



# CMS32H6157 User Manual

**Ultra-low-power 32-bit measurement SOC based on the ARM® Cortex®-M0+**

**Rev. 0.1.1**

Please note the following CMS IP policy

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

\* The name and logo of Zhongwei Semiconductor (Shenzhen) Co., Ltd. are registered trademarks of the Company.

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

## Document Instructions

This manual is the user manual for the CMS32H6157 microcontroller product. The user 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. Each function module is introduced in a special section.

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

The [data sheet](#) information is as follows::

CMS32H6157datasheet\_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 [user 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 [user manual](#) in detail to understand the specific implementation and register configuration of each function. Refer to the [datasheet](#) for information on voltages, currents, drive capabilities and pin assignments when designing the hardware.

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

## Contents

Document Instructions .....	2
<b>CHAPTER 1 CPU.....</b>	<b>19</b>
1.1 Overview .....	19
1.2 Cortex-M0+ core features .....	19
1.3 Debugging features.....	19
1.4 SWD interface pins .....	21
1.5 ARM reference document.....	22
<b>CHAPTER 2 PORT FUNCTION.....</b>	<b>23</b>
2.1 Port general function .....	23
2.2 Port multiplexing function.....	24
2.3 Registers for controlling port functions .....	27
2.3.1 Port output control register (PMxx) .....	29
2.3.2 Port register (Pxx) .....	30
2.3.3 Port set control register (PSETxx) .....	31
2.3.4 Port clear control register (PCLRxx) .....	32
2.3.5 Pull-up resistor selection register (PUxx) .....	33
2.3.6 Pull-down resistor selection register (PDxx).....	34
2.3.7 Port output mode register (POMxx) .....	35
2.3.8 Port mode control register (PMCxx) .....	36
2.3.9 Port readback register (PREADxx).....	37
2.3.10 Port multiplexing function configuration register (PxxCFG) .....	38
2.3.11 External interrupt port selection register (INTPnPCFG) .....	41
2.3.12 External reset port mask register (RSTM) .....	44
2.4 Handling of unused ports .....	45
2.5 Register settings when using the multiplexing function.....	46
Basic ideas when using multiplexing functions .....	46
<b>CHAPTER 3 SYSTEM STRUCTURE .....</b>	<b>65</b>
3.1 Overview .....	65
3.2 System address division .....	66
3.3 Peripheral address assignment .....	67
<b>CHAPTER 4 CLOCK GENERATION CIRCUIT .....</b>	<b>68</b>
4.1 Function of clock generation circuit .....	68
4.2 Structure of clock generation circuit.....	70
4.3 Registers for controlling clock generation circuit .....	73
4.3.1 Clock operation mode control register (CMC) .....	73
4.3.2 System clock control register (CKC).....	75
4.3.3 Clock operation status control register (CSC) .....	76
4.3.4 Status register of the oscillation stabilization time counter (OSTC) .....	78

4.3.5	Oscillation stabilization time selection register (OSTS).....	80
4.3.6	Peripheral enabled registers 0, 1 (PER0, PER1) .....	82
4.3.7	Subsystem clock supply mode control register (OSMC) .....	85
4.3.8	High-speed internal oscillator frequency selection register (HOCODIV) .....	86
4.3.9	High-speed internal oscillator trim register (HIOTRM) .....	87
4.3.10	Subsystem clock selection register (SUBCKSEL).....	88
4.4	System clock oscillation circuit.....	89
4.4.1	X1 oscillation circuit .....	89
4.4.2	XT1 oscillation circuit .....	90
4.4.3	High-speed internal oscillator .....	93
4.4.4	Low-speed internal oscillator .....	93
4.5	Operation of clock generation circuit .....	94
4.6	Clock control .....	96
4.6.1	Example of setting up a high-speed internal oscillator .....	96
4.6.2	Example of setting X1 oscillation circuit .....	98
4.6.3	Example of setting XT1 oscillation circuit .....	99
4.6.4	CPU clock state transition diagram.....	100
4.6.5	Conditions before CPU clock transfer and post-transfer processing .....	106
4.6.6	Time required to switch CPU clock and main system clock.....	108
4.6.7	Conditions before clock oscillation stops.....	109
4.7	High-speed internal oscillation correction .....	110
4.7.1	High-speed internal oscillation self-adjustment function .....	110
4.7.2	Register description .....	111
4.7.2.1	High-speed internal oscillation frequency correction control register (HOCOFC).....	111
4.7.3	Action description.....	113
4.7.3.1	Action summary.....	113
4.7.3.2	Action setting flow .....	116
4.7.4	Notes on use .....	117
4.7.4.1	SFR Access.....	117
4.7.4.2	Actions on resetting.....	117
4.8	Oscillation stop detection circuit .....	118
4.8.1	Composition of the oscillation-stop detection circuit.....	118
4.8.2	The registers used by the oscillation-stop detection circuit.....	119
4.8.2.1	Peripheral enabled Register 1(PER1) .....	119
4.8.2.2	Oscillation stop detection control register (SCMCTL).....	119
4.8.2.3	Oscillation stop detection mode register (SCMMD).....	120
4.8.2.4	Oscillation stop detection status register (SCMST) .....	120
4.8.3	Operation of oscillation stop detection circuit .....	121
4.8.3.1	Operation method of oscillation stop detection circuit .....	121



4.8.4	Operation of oscillation stop detection circuit in deep sleep mode .....	122
4.8.5	Notes on the oscillation stop detection function .....	122
<b>CHAPTER 5</b>	<b>GENERAL-PURPOSE TIMER UNIT TIMER8 .....</b>	<b>123</b>
5.1	General-purpose timer unit functions.....	125
5.1.1	Independent channel operation functions.....	125
5.1.2	Multi-channel linkage operation functions .....	127
5.1.3	LIN-bus support function (channel 3 only).....	129
5.2	Structure of the general-purpose timer unit .....	130
5.2.1	General-purpose timer unit register list .....	132
5.2.2	Timer count register mn (TCRmn) .....	133
5.2.3	Timer data register mn (TDRmn).....	135
5.3	Registers that control the general-purpose timer unit .....	136
5.3.1	Peripheral enable register 0(PER0).....	137
5.3.2	Timer clock selection register m (TPSm).....	138
5.3.3	Timer mode register mn (TMRmn) .....	141
5.3.4	Timer status register mn (TSRmn) .....	145
5.3.5	Timer channel enable status register m (TEm) .....	146
5.3.6	Timer channel start register m (TSM).....	147
5.3.7	Timer channel stop register m (TPSm).....	148
5.3.8	Timer input output selection register (TIOSt).....	149
5.3.9	Timer output enable register m (TOEm).....	150
5.3.10	Timer output register m (TOM) .....	151
5.3.11	Timer output level register m (TOLm).....	152
5.3.12	Timer output mode register m (TOMm) .....	153
5.3.13	Input switching control register (ISC).....	154
5.3.14	Noise filter enable register (NFEN1).....	155
5.3.15	Registers for controlling timer input/output pin port functions .....	156
5.4	Basic rules of the general-purpose timer unit .....	157
5.4.1	Basic rules of multi-channel linkage operation function .....	157
5.4.2	Timer channel start register m (TSM).....	160
5.5	Operation of counters .....	161
5.5.1	Counting clock ( $F_{TCLK}$ ).....	161
5.5.2	Start timing of counter .....	163
5.5.3	Operation of counters .....	164
5.6	Control of channel outputs (TOMn pins).....	169
5.6.1	Block diagram of the TOMn pin output circuit.....	169
5.6.2	Settings of the TOMn pin output .....	170
5.6.3	Cautions for channel output operation.....	171
5.6.4	One-time operation of TOMn bit .....	176

5.6.5	Timer interrupt and TOMn pin output when counting starts .....	177
5.7	Control of Timer Input (TImn).....	178
5.7.1	Block diagram of the TImn pin input circuit.....	178
5.7.2	Noise filter .....	178
5.7.3	Cautions for channel input operation .....	179
5.8	Independent channel operation function for general purpose timer units .....	180
5.8.1	Operation as interval timer/square wave output.....	180
5.8.2	Operation as external event counter.....	184
5.8.3	Operation as frequency divider .....	187
5.8.4	Operation as input pulse interval measurement.....	190
5.8.5	Operation as input signal high and low level width measurement .....	194
5.8.6	Operation as delay counter.....	198
5.9	Multi-channel linkage operation function for general purpose timer units.....	201
5.9.1	Operation as single trigger pulse output function .....	201
5.9.2	Operation as PWM function.....	208
5.9.3	Operation as multiple PWM output function .....	215
5.10	Cautions when using the general-purpose timer unit .....	224
5.10.1	Cautions when using timer output .....	224
<b>CHAPTER 6</b>	<b>TIMERA .....</b>	<b>225</b>
6.1	TimerA function .....	225
6.2	Structure of Timer A .....	226
6.3	Registers for controlling Timer A.....	227
6.3.1	Peripheral enable register 0(PER0).....	228
6.3.2	Subsystem Clock Supply Mode Control Register (OSMC) .....	229
6.3.3	Timer A count register 0 (TA0) .....	230
6.3.4	Timer A control register 0 (TACR0) .....	231
6.3.5	Timer AI/O control register 0 (TAIOC0) .....	232
6.3.6	Timer A control register 0 (TAMR0).....	234
6.3.7	Timer A event pin selection register 0 (TAISR0) .....	235
6.4	Timer A operation.....	236
6.4.1	Rewriting the reload register and counter.....	236
6.4.2	Timer mode .....	237
6.4.3	Pulse output mode .....	238
6.4.4	Event counter mode.....	239
6.4.5	Pulse width measurement mode .....	241
6.4.6	Pulse period measurement mode.....	242
6.4.7	Collaboration with EVENTC.....	243
6.4.8	Output settings for each mode.....	244
6.5	Cautions when using timer A .....	245

6.5.1	Start and stop control of counting .....	245
6.5.2	Flag access (TEDGF bit and TUNDF bit of TACR0 register) .....	245
6.5.3	Access to counting registers .....	246
6.5.4	Change in mode .....	246
6.5.5	Setting procedure for TAO pin and TAIO pin .....	246
6.5.6	When timer A is not used .....	247
6.5.7	Stop of timer A operation clock .....	247
6.5.8	Setting steps for deep sleep mode (event counter mode) .....	247
6.5.9	Function limitations in deep sleep mode (event counter mode only) .....	248
6.5.10	Forced count stop via the TSTOP bit .....	248
6.5.11	Digital Filters .....	248
6.5.12	Selecting F <sub>IL</sub> as the count source .....	249
<b>CHAPTER 7</b>	<b>REAL TIME CLOCK .....</b>	<b>250</b>
7.1	Functions of real-time clock .....	250
7.2	Structure of real-time clock .....	250
7.3	Registers for controlling real-time clock .....	252
7.3.1	Peripheral enable register 0 (PER0) .....	253
7.3.2	Real-time clock selection register (RTCCL) .....	254
7.3.3	Real-time clock control register 0 (RTCC0) .....	255
7.3.4	Real-time clock control register 1 (RTCC1) .....	256
7.3.5	Watch error correction register (SUBCUD) .....	258
7.3.6	Second count register (SEC) .....	259
7.3.7	Minute count register (MIN) .....	260
7.3.8	Hour count register (HOUR) .....	261
7.3.9	Day count register (DAY) .....	263
7.3.10	Week count register (WEEK) .....	264
7.3.11	Month count register (MONTH) .....	265
7.3.12	Year count register (YEAR) .....	266
7.3.13	Alarm minute register (ALARMWM) .....	267
7.3.14	Alarm hour register (ALARMWH) .....	268
7.3.15	Alarm week register (ALARMWW) .....	269
7.3.16	Starting operation of real-time clock .....	270
7.3.17	Shifting to sleep mode after starting operation .....	271
7.3.18	Reading/writing real-time clock .....	272
7.3.19	Setting alarm of real-time clock .....	274
7.3.20	1 Hz output of real-time clock .....	275
7.3.21	Example of watch error correction of real-time clock .....	276
<b>CHAPTER 8</b>	<b>15-BIT INTERVAL TIMER .....</b>	<b>278</b>
8.1	Functions of 15-bit Interval Timer .....	278

8.2	Structure of 15-bit Interval Timer .....	278
8.3	Registers controlling 15-bit Interval Timer .....	279
8.3.1	Peripheral enable register 0 (PER0) .....	279
8.3.2	Real-time clock selection register (RTCCL) .....	280
8.3.3	15-bit interval timer control register (ITMC) .....	281
8.4	15-bit interval timer operation .....	282
8.4.1	15-bit interval timer operation timing .....	282
8.4.2	Start of count operation and re-enter to sleep mode after returned from sleep mode .....	283
<b>CHAPTER 9</b>	<b>CLOCK OUTPUT/BUZZER OUTPUT CONTROLLER .....</b>	<b>284</b>
9.1	Functions of clock output/buzzer output controller .....	284
9.2	Registers for controlling clock output/buzzer output controller .....	286
9.2.1	Clock output select registers n (CKSn) .....	286
9.3	Operations of clock output/buzzer output controller .....	288
9.4	Cautions of clock output/buzzer output controller .....	288
<b>CHAPTER 10</b>	<b>WATCH DOG TIMER .....</b>	<b>289</b>
10.1	Functions of watchdog timer .....	289
10.2	Structure of watch dog timer .....	289
10.3	Registers for controlling watchdog timer .....	291
10.3.1	Watchdog timer enable register (WDTE) .....	291
10.3.2	LOCKUP control register (LOCKCTL) and its protection register (PRCR) .....	292
10.3.3	WDTCFG configuration registers (WDTCFG0/1/2/3) .....	293
10.4	Operation of watchdog timer .....	294
10.4.1	Operational control of watchdog timer .....	294
10.4.2	Setting overflow time of watchdog timer .....	296
10.4.3	Setting window open period of watchdog timer .....	297
10.4.4	Setting watchdog timer interval interrupt .....	298
10.4.4.1	Operation of the watchdog timer during LOCKUP .....	298
10.4.4.2	Operation of the watchdog timer without WDTCFG configured .....	298
<b>CHAPTER 11</b>	<b>AD CONVERTER .....</b>	<b>299</b>
11.1	Function of A/D converter .....	299
11.2	Registers for controlling A/D converter .....	301
11.2.1	Peripheral enable register (PER1) .....	302
11.2.2	A/D converter mode register 0 (ADM0) .....	303
11.2.3	A/D converter mode register 1 (ADM1) .....	304
11.2.4	A/D converter mode register 2 (ADM2) .....	305
11.2.5	A/D converter trigger mode register (ADTRG) .....	306
11.2.6	Analog input channel specification register (ADS) .....	307
11.2.7	12-bit A/D conversion result register (ADCR) .....	310
11.2.8	8-bit A/D conversion result register (ADCRH) .....	311

11.2.9	Conversion result comparison upper limit setting register (ADUL) .....	311
11.2.10	Conversion result comparison lower limit setting register (ADLL).....	312
11.2.11	A/D converter sampling time control register (ADNSMP).....	313
11.2.12	A/D converter sampling time extension control register (ADSMPWAIT).....	315
11.2.13	A/D test register (ADTES) .....	316
11.2.14	A/D converters charge-discharge control register (ADNDIS) .....	317
11.3	Input voltage and conversion result .....	318
11.4	Operation mode of A/D converter .....	319
11.4.1	Software trigger mode (select mode, sequential conversion mode) .....	319
11.4.2	Software trigger mode (select mode, single conversion mode) .....	320
11.4.3	Software trigger mode (scan mode, sequential conversion mode).....	321
11.4.4	Software trigger mode (scan mode, single conversion mode) .....	322
11.4.5	Hardware trigger no-wait mode (select mode, sequential conversion mode) .....	323
11.4.6	Hardware trigger no-wait mode (select mode, single conversion mode) .....	324
11.4.7	Hardware trigger no-wait mode (scan mode, sequential conversion mode).....	325
11.4.8	Hardware trigger no-wait mode (scan mode, single conversion mode) .....	326
11.4.9	Hardware trigger wait mode (select mode, sequential conversion mode) .....	327
11.4.10	Hardware trigger wait mode (select mode, single conversion mode) .....	328
11.4.11	Hardware trigger wait mode (scan mode, sequential conversion mode) .....	329
11.4.12	Hardware trigger wait mode (scan mode, single conversion mode) .....	330
11.5	A/D converter setup flowchart.....	331
11.5.1	Setting up software trigger mode.....	331
11.5.2	Setting up hardware trigger no-wait mode.....	332
11.5.3	Setting up hardware trigger wait mode .....	333
11.5.4	Setup when temperature sensor output voltage/internal reference voltage is selected .....	334
11.5.5	Setting up test mode .....	335

## **CHAPTER 12 SIGMA-DELTA ADC.....336**

12.1	Overview .....	336
12.2	Description of basic functions .....	336
12.3	Description of ADC working principle .....	337
12.3.1	LDO .....	337
12.3.2	Analog inputs .....	337
12.3.3	Temperature sensor.....	337
12.3.4	Low noise programmable gain amplifier.....	337
12.3.5	ADC clock, output data rate .....	338
12.3.6	Reset and sleep mode .....	339
12.3.7	Setup time .....	339
12.4	SPI Serial Interface .....	340
12.4.1	Digital output codes .....	340

12.4.2	Data ready/data input and output (DRDYB/DOUT) .....	340
12.4.3	Serial input clock (SCLK) .....	340
12.4.4	Serial data transmission .....	341
12.4.5	Function configuration .....	342
12.4.6	Description of SPI opcode command .....	343
12.4.7	Cautions for SPI Communication .....	343
12.5	Related registers .....	344
12.5.1	Sigma-Delta ADC control register 1 .....	344
12.5.2	Sigma-Delta ADC control register 2 .....	345
12.5.3	Sigma-Delta ADC control register 3 .....	346
12.5.4	Sigma-Delta ADC control register 4 .....	347
<b>CHAPTER 13</b>	<b>D/A CONVERTER .....</b>	<b>348</b>
13.1	Function of D/A converter .....	348
13.2	Structure of D/A converter .....	349
13.3	Registers for controlling D/A converter .....	350
13.3.1	Peripheral enable register 1 (PER1) .....	350
13.3.2	D/A converter mode register (DAM) .....	351
13.3.3	D/A conversion value setting register i (DACSi) (i=0) .....	351
13.3.4	Event output destination select register n (ELSELRn), n= 00~15 .....	351
13.4	Operation of D/A Converter .....	352
13.4.1	Operation in Normal Mode .....	352
13.4.2	Operation in real-time output mode .....	353
13.4.3	D/A conversion output timing .....	354
13.5	Cautions for D/A Converter .....	355
<b>CHAPTER 14</b>	<b>COMPARATOR .....</b>	<b>356</b>
14.1	Function of comparator .....	356
14.2	Structure of comparator .....	357
14.3	Registers for controlling comparator .....	359
14.3.1	Peripheral enable register 1 (PER1) .....	360
14.3.2	Comparator mode setting register (COMPMDR) .....	361
14.3.3	Comparator filter control register (COMPFIR) .....	362
14.3.4	Comparator output control register (COMPOCR) .....	364
14.3.5	Comparator negative reference selection register (CnREFS) .....	366
14.3.6	Comparator positive input selection register (CMPSELn) .....	367
14.3.7	Comparator hysteresis control register (CMPnHY) .....	368
14.4	Operation .....	369
14.4.1	Digital filter of comparator n (n=0, 1) .....	371
14.4.2	Comparator n interrupt (n=0, 1) .....	371
14.4.3	Event signal output to the coordination controller (EVENTC) .....	372

14.4.4	Output of comparator n (n=0, 1)	373
14.4.5	Stopping or Supplying comparator clock	374
<b>CHAPTER 15</b>	<b>OPERATIONAL AMPLIFIER (OPA)</b>	<b>375</b>
15.1	Function of operational amplifier	375
15.2	Register of operational amplifier	376
15.2.1	Peripheral enable register 1 (PER1)	376
15.2.2	Operational amplifier control register (OPACTL)	377
15.2.3	Operational amplifier digital-to-analog control register (OPADAC)	378
<b>CHAPTER 16</b>	<b>GENERAL-PURPOSE SERIAL COMMUNICATION UNIT</b>	<b>379</b>
16.1	Function of general-purpose serial communication unit	380
16.1.1	3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)	380
16.1.2	UART (UART0~UART2)	381
16.1.3	Simplified I <sup>2</sup> C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)	382
16.2	Structure of general-purpose serial communication unit	383
16.2.1	Shift register	385
16.2.2	Serial data register mn (SDRmn)	385
16.3	Registers for controlling general-purpose serial communication unit	387
16.3.1	Peripheral enable register 0 (PER0)	389
16.3.2	Serial clock selection register m (SPSm)	390
16.3.3	Serial mode register mn (SMRmn)	391
16.3.4	Serial communication run setting register mn (SCRmn)	393
16.3.5	Serial data register mn (SDRmn)	396
16.3.6	Serial flag clear trigger register mn(SIRmn)	397
16.3.7	Serial status register mn (SSRmn)	398
16.3.8	Serial channel start register m(SSm)	400
16.3.9	Serial channel stop register m(STm)	401
16.3.10	Serial channel enable status register m (SEm)	402
16.3.11	Serial output enable register m (SOEm)	403
16.3.12	Serial output register m (SOM)	404
16.3.13	Serial output level register m (SOLm)	405
16.3.14	Serial standby control register m (SSCm)	406
16.3.15	Slave select function enable register m (SSEm)	407
16.3.16	Input switching control register (ISC)	408
16.3.17	Noise filter enable register 0 (NFEN0)	409
16.4	Operation stop mode	410
16.4.1	Stopping the operation by units	410
16.4.2	Stopping the operation by channels	411
16.5	3-wire serial I/O(SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) communication	412
16.5.1	Master transmission	413



16.5.2	Master reception .....	421
16.5.3	Master transmission and reception .....	429
16.5.4	Slave transmission .....	437
16.5.5	Slave reception .....	445
16.5.6	Slave transmission and reception .....	451
16.5.7	Calculation of transmission clock frequency .....	460
16.5.8	Procedure for handling errors during 3-wire serial I/O communication (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) .....	462
16.6	Operation of clock-synchronous serial communication with slave selection input function .....	463
16.6.1	Slave transmission .....	466
16.6.2	Slave reception .....	476
16.6.3	Slave transmission and reception .....	483
16.6.4	Calculation of transmission clock frequency .....	493
16.6.5	Procedure for handling errors during clock-synchronous serial communication with the slave selection input function .....	494
16.7	Operation of UART (UART0~UART2) communication .....	495
16.7.1	UART transmission .....	496
16.7.2	UART reception .....	504
16.7.3	Low-power UART mode function .....	511
16.7.4	Calculation of baud rate .....	517
16.7.5	Handling steps when an error occurs during UART (UART0~UART2) communication .....	521
16.8	Operation of LIN communication .....	522
16.8.1	LIN transmission .....	522
16.8.2	LIN reception .....	525
16.9	Operation of simple I <sup>2</sup> C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication .....	530
16.9.1	Address segment sending .....	531
16.9.2	Data transmission .....	536
16.9.3	Data reception .....	539
16.9.4	Generation of stop condition .....	543
16.9.5	Calculation of transfer rate .....	544
16.9.6	Processing steps when an error occurs in a simple I <sup>2</sup> C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication process .....	546

## **CHAPTER 17 SERIAL INTERFACE SPI.....547**

17.1	Function of SPI .....	547
17.2	Structure of SPI .....	547
17.3	Registers for controlling SPI .....	548
17.3.1	Peripheral enable register 1 (PER1) .....	548
17.3.2	SPI operating mode register (SPIM) .....	549
17.3.3	SPI clock selection register (SPIC) .....	550
17.3.4	SPI status register (SPIS) .....	551



17.3.5	Transmit buffer register (SOTB) .....	552
17.3.6	Receive buffer register (SIO) .....	552
17.4	Operation of SPI.....	553
17.4.1	Master transmission and reception.....	554
17.4.2	Master reception .....	557
17.4.3	Slave transmission and reception.....	560
17.4.4	Slave reception .....	563
<b>CHAPTER 18</b>	<b>SERIAL INTERFACE IICA .....</b>	<b>566</b>
18.1	Function of IICA .....	566
18.1.1	Run stop mode.....	566
18.1.2	I <sup>2</sup> C-bus mode (supports multi-master).....	566
18.1.3	Wake-up mode.....	566
18.2	Structure of IICA.....	569
18.2.1	IICA shift register n (IICAn) .....	570
18.2.2	Slave address register n(SVAn) .....	570
18.2.3	SO latch .....	570
18.2.4	Wake-up control circuitry .....	571
18.2.5	Serial clock counter.....	571
18.2.6	Interrupt request signal generation circuit .....	571
18.2.7	Serial clock control circuitry .....	571
18.2.8	Serial clock wait control circuit.....	571
18.2.9	Ack generation circuit, stop condition detection circuit, start condition detection circuit, Ack detection circuit.....	571
18.2.10	Data hold time correction circuit.....	571
18.2.11	Start conditional generation circuitry.....	571
18.2.12	Stop condition generation circuitry.....	572
18.2.13	Bus status detection circuitry .....	572
18.3	Registers for controlling IICA .....	573
18.3.1	Peripheral enable register 0 (PER0).....	573
18.3.2	IICA control register n0 (IICCTLn0) .....	574
18.3.3	IICA status register n(IICSn) .....	578
18.3.4	IICA flag register n(IICFn).....	581
18.3.5	IICA control register n1(IICCTLn1) .....	583
18.3.6	IICA low-level width setting register n(IICWLn) .....	585
18.3.7	IICA high level width setting register n(IICWHn) .....	585
18.3.8	Registers for controlling the IICA pin port function .....	586
18.4	Function of I <sup>2</sup> C-bus mode .....	587
18.4.1	Pin structure .....	587
18.4.2	Setting the transmit clock via IICWLn register and IICWHn register.....	588
18.5	Definition and control method of I <sup>2</sup> C-bus.....	590

18.5.1	Start condition .....	591
18.5.2	Address .....	592
18.5.3	Designation of transmission direction .....	592
18.5.4	ACK .....	593
18.5.5	Stop Condition .....	594
18.5.6	Await .....	595
18.5.7	Method of release from wait state .....	597
18.5.8	Generation timing and waiting control of interrupt requests (INTIICAn) .....	598
18.5.9	Detection method for address matching .....	599
18.5.10	Error detection .....	599
18.5.11	Extension code .....	600
18.5.12	Arbitration .....	601
18.5.13	Wake-up function .....	603
18.5.14	Communication appointment .....	606
18.5.15	Other cautions .....	610
18.5.16	Communication operation .....	611
18.5.17	Timing of I <sup>2</sup> C interrupt request (INTIICAn) generation .....	621
18.6	Timing diagram .....	641
<b>CHAPTER 19</b>	<b>IRDA .....</b>	<b>658</b>
19.1	Function of IrDA .....	658
19.2	Registers for controlling IrDA .....	659
19.2.1	Peripheral enable register 0 (PER0) .....	659
19.2.2	IrDA control register (IRCR) .....	660
19.3	Operation of IrDA .....	661
19.3.1	Procedure for IrDA communication .....	661
19.3.2	Transmission .....	662
19.3.3	Reception .....	662
19.3.4	Selection of high level pulse width .....	663
19.4	Cautions when using IrDA .....	664
<b>CHAPTER 20</b>	<b>LCD CONTROLLER/DRIVER .....</b>	<b>665</b>
20.1	Function of LCD controller / driver .....	665
20.2	Structure of LCD controller / driver .....	666
20.3	Registers for controlling LCD controller/driver .....	668
20.3.1	LCD mode register 0(LCDM0) .....	669
20.3.2	LCD mode register 1(LCDM1) .....	671
20.3.3	Subsystem Clock Supply Mode Control Register (OSMC) .....	673
20.3.4	LCD clock control register 0 (LCDC0) .....	674
20.3.5	LCD Boost Level Control Register (VLCD) .....	675
20.3.6	LCD input switching control register (ISCLCD) .....	677

20.3.7	LCD port function register .....	678
20.4	LCD display data register .....	679
20.5	LCD display register selection .....	681
20.5.1	Data display in graphic area A and graphic area B .....	682
20.5.2	Blinking display (alternate display of data in graph area A and graph area B) .....	682
20.6	LCD drive voltage provided by $V_{L1}$ , $V_{L2}$ , $V_{L3}$ , $V_{L4}$ .....	683
20.6.1	Internal resistor splitting method .....	683
20.6.2	External resistance splitting method .....	684
20.6.3	Internal boost method .....	685
20.6.4	Capacitance splitting method .....	686
<b>CHAPTER 21</b>	<b>ENHANCED DMA .....</b>	<b>687</b>
21.1	Function of DMA .....	687
21.2	Structure of DMA .....	689
21.3	Registers for controlling DMA .....	690
21.3.1	DMA control data areas and DMA vector table areas allocation .....	691
21.3.2	Control data allocation .....	692
21.3.3	Vector table .....	694
21.3.4	Peripheral enable register 1 (PER1) .....	696
21.3.5	DMA control register j (DMACRj)(j=0~23) .....	697
21.3.6	DMA block size register j (DMBLSj)(j=0~23) .....	699
21.3.7	DMA transmit count register j (DMACTj)(j=0~23) .....	700
21.3.8	DMA transfer count reload register j (DMRLDj)(j=0~23) .....	701
21.3.9	DMA source address register j (DMSARj)(j=0~23) .....	702
21.3.10	DMA destination address register j (DMDARj)(j=0~23) .....	703
21.3.11	DMA start enable register i (DMAENi)(i=0~2) .....	704
21.3.12	DMA base address register (DMABAR) .....	706
21.3.13	DMAENi set register i (DMSETi)(i=0~2) .....	707
21.3.14	DMAENi reset register i (DMCLRi)(i=0~2) .....	707
21.4	Operation of DMA .....	708
21.4.1	Startup Source .....	708
21.4.2	Normal mode .....	709
21.4.3	Repeat pattern .....	712
21.4.4	Chain transmission .....	715
21.5	Cautions when using DMA .....	717
21.5.1	DMA control data and vector table settings .....	717
21.5.2	Allocation of DMA control data area and DMA vector table area .....	717
21.5.3	Number of execution clocks for DMA .....	718
21.5.4	Response time of DMA .....	719
21.5.5	Startup source of DMA .....	719

21.5.6	Operation in standby mode.....	720
<b>CHAPTER 22</b>	<b>LINKAGE CONTROLLER (EVENTC) .....</b>	<b>721</b>
22.1	Function of EVENTC.....	721
22.2	Structure of EVENTC.....	721
22.3	Control register.....	722
22.3.1	Output target selection register n(ELSELRn)(n=00~15) .....	723
22.4	Operation of EVENTC.....	726
<b>CHAPTER 23</b>	<b>INTERRUPT FUNCTION .....</b>	<b>728</b>
23.1	Types of interrupt function .....	728
23.2	Interrupt Sources and Structures.....	728
23.3	Registers for controlling interrupt function .....	732
23.3.1	Interrupt request flag register (IF00~IF31).....	732
23.3.2	Interrupt mask flag register (MK00~MK31) .....	733
23.3.3	External interrupt rising edge enable register (EGP0), External interrupt falling edge enable register (EGN0).....	735
23.4	Operation of interrupt handling .....	736
23.4.1	Acceptance of maskable interrupt requests .....	736
23.4.2	Acceptance of non-maskable interrupt requests .....	736
<b>CHAPTER 24</b>	<b>KEY INTERRUPT FUNCTION.....</b>	<b>737</b>
24.1	Function of key interrupt .....	737
24.2	Structure of key interrupt.....	737
24.3	Register controlling key interrupts .....	739
24.3.1	Key return mode register (KRM).....	739
<b>CHAPTER 25</b>	<b>STANDBY FUNCTION .....</b>	<b>740</b>
25.1	Standby function .....	740
25.2	Sleep mode .....	741
25.2.1	Setting of sleep mode .....	741
25.2.2	Release of sleep mode .....	745
25.3	Deep sleep mode.....	746
25.3.1	Setting of deep sleep mode .....	746
25.3.2	Release of deep sleep mode .....	749
<b>CHAPTER 26</b>	<b>RESET FUNCTION.....</b>	<b>750</b>
26.1	Reset timing .....	752
26.2	Register for confirming the reset source.....	755
26.2.1	Reset control flag register (RESF).....	755
<b>CHAPTER 27</b>	<b>POWER-ON RESET CIRCUIT.....</b>	<b>757</b>
27.1	Function of power-on reset circuit.....	757
27.2	Structure of power-on reset circuit.....	758
27.3	Operation of power-on reset circuit.....	759

<b>CHAPTER 28 VOLTAGE DETECTION CIRCUIT .....</b>	<b>763</b>
28.1 Function of voltage detection circuit .....	763
28.2 Structure of voltage detection circuit.....	764
28.3 Registers for controlling voltage detection circuit .....	765
28.3.1 Voltage detection register (LVIM) .....	765
28.3.2 Voltage detection level register (LVIS) .....	766
28.4 Operation of voltage detection circuit .....	769
28.4.1 Settings when used in reset mode.....	769
28.4.2 Settings when used in interrupt mode .....	770
28.4.3 Settings for interrupt & reset mode .....	772
28.5 Cautions for voltage detection circuits .....	779
<b>CHAPTER 29 SECURITY FEATURE .....</b>	<b>781</b>
29.1 Overview .....	781
29.2 Registers Used for Security Functions .....	782
29.3 Operation of security features.....	783
29.3.1 Flash CRC operational function (high-speed CRC) .....	783
29.3.1.1 Flash CRC control register (CRC0CTL) .....	783
29.3.1.2 Flash CRC operation result register (PGCRCL).....	784
29.3.2 CRC operation function (general-purpose CRC).....	786
29.3.2.1 CRC input register (CRCIN).....	787
29.3.2.2 CRC data register (CRCD) .....	787
29.3.3 RAM parity error detection function .....	789
29.3.3.1 RAM parity error control register (RPECTL).....	789
29.3.4 SFR protection function .....	791
29.3.4.1 SFR protect control register(SFRGD).....	791
29.3.5 Frequency detection function.....	792
29.3.5.1 Timer input selection register0(TIS0).....	792
29.3.6 A/D test function.....	793
29.3.6.1 A/D test register (ADTES).....	795
29.3.6.2 Analog input channel specification register (ADS) .....	795
29.3.7 Product Unique Identification Register .....	796
<b>CHAPTER 30 TEMPERATURE SENSOR AND INTERNAL REFERENCE VOLTAGE ....</b>	<b>797</b>
30.1 Temperature sensor.....	797
30.2 Register for temperature sensor .....	798
30.2.1 Temperature sensor calibration data register TSN25 .....	798
30.2.2 Temperature sensor calibration data register TSN85 .....	798
30.3 Instructions for using the temperature sensor .....	799
30.3.1 How temperature sensors are used.....	799
30.3.2 How to use temperature sensor.....	800

30.4	Internal reference voltage .....	801
30.4.1	VDD calibration data register VDDCDR .....	801
30.4.2	Instructions for using the internal reference voltage.....	801
<b>CHAPTER 31</b>	<b>OPTION BYTES .....</b>	<b>802</b>
31.1	Function of option bytes .....	802
31.1.1	User option bytes (000C0H~000C2H) .....	802
31.1.2	Flash data protection option bytes (000C3H, 500004H) .....	803
31.2	Format of the user option bytes .....	804
31.3	Format of flash data protection option bytes .....	810
<b>CHAPTER 32</b>	<b>FLASH CONTROL.....</b>	<b>811</b>
32.1	Description of FLASH control.....	811
32.2	Structure of FLASH memory.....	811
32.3	Registers for controlling FLASH .....	812
32.3.1	Flash write protection register (FLPROT).....	812
32.3.2	FLASH operation control register (FLOPMD1,FLOPMD2) .....	813
32.3.3	Flash erase control register (FLERMD).....	814
32.3.4	Flash status register (FLSTS).....	815
32.3.5	Flash full-chip erase time control register (FLCERCNT).....	815
32.3.6	Flash sector erase time control register (FLSERCNT).....	816
32.3.7	Flash write time control register (FLPROCNT).....	817
32.3.8	Flash erase protection control register (FLSECPR) .....	818
32.4	How to operate FLASH .....	819
32.4.1	Sector erase.....	819
32.4.2	Chip erase.....	820
32.4.3	Word program .....	820
32.5	Flash Read.....	821
32.6	Cautions for FLASH operation .....	821
<b>CHAPTER 33</b>	<b>REVISION HISTORY .....</b>	<b>822</b>

# Chapter 1 CPU

## 1.1 Overview

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

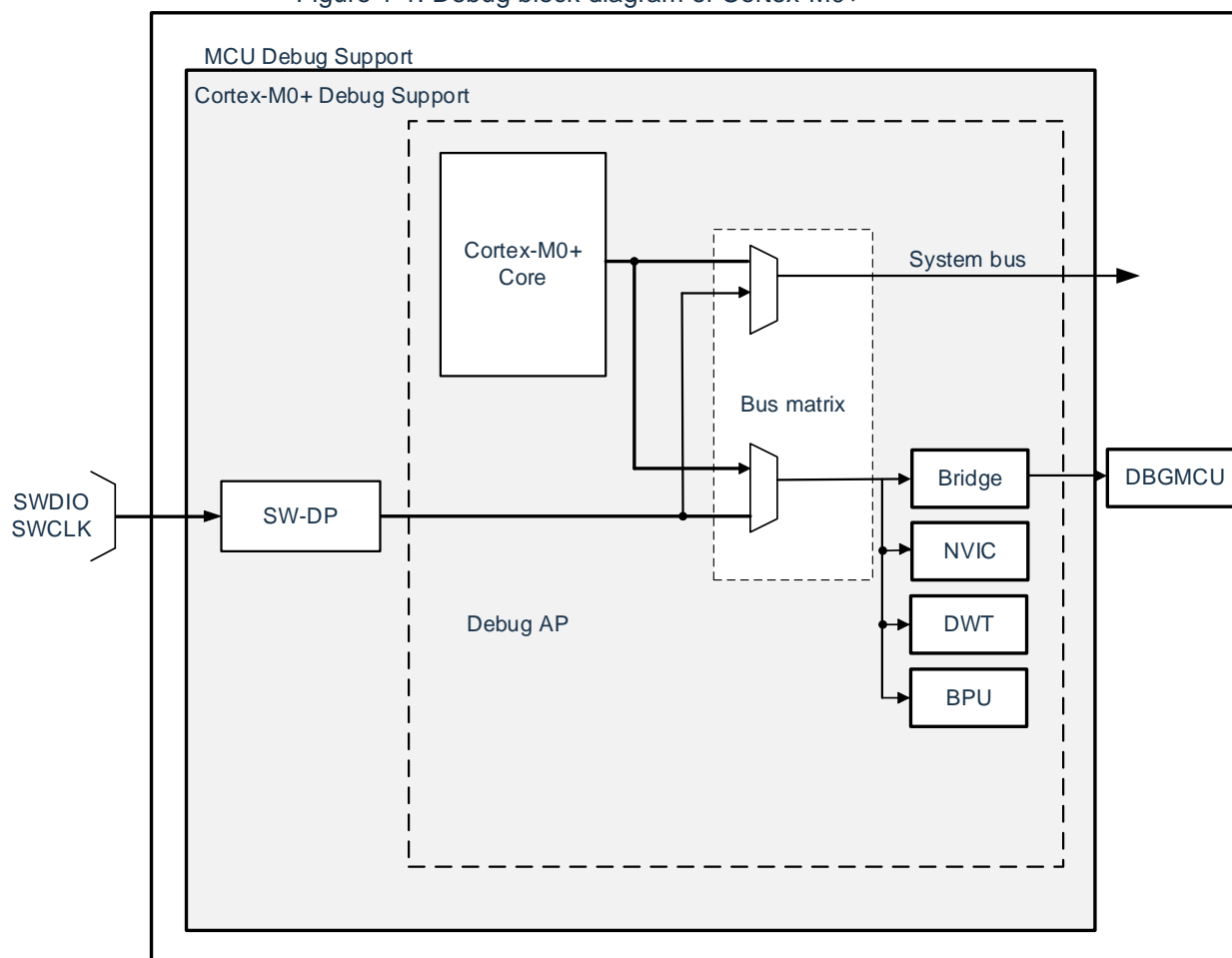
## 1.2 Cortex-M0+ core features

- ARM Cortex-M0+ processor is a 32-bit RISC core with a 2-stage pipeline that supports privileged and user modes
- 32-cycle hardware multiplier
- Nested Vector Interrupt Controller (NVIC)
  - 1 non-maskable interrupt (NMI)
  - Support 32 maskable interrupt requests (IRQ)
  - 4 interrupt priority levels
- System Timer (SysTick) is a 24-bit countdown timer with a choice of  $F_{CLK}$  or  $F_{IL}$  count clock
- Vector Table Offset Register (VTOR)
  - The software can write VTOR to relocate the vector table start address to a different location.
  - The default value of this register is 0x0000\_0000, the lower 8 bits are ignored for writing and zero for reading, which means the offset is 256 bytes aligned.

## 1.3 Debugging features

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

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



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



## 1.4 SWD interface pins

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

Table 1-1: SWD debug port pins

SWD port name	Debugging functions	Pin assignment
SWCLK	Serial clock	PA14
SWDIO	Serial data input/output	PA13

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

Symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DBGSTOPCR <31:16>	-							SWD IS	-							
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBGSTOPCR <15:0>	-														FRZ EN1	FRZ EN0

- Bit24      SWDIS:    SWD debug interface disabled
- 0=    SWD debug interface enable. PA13 cannot be used as GPIO (because ENO and DOUT of this IOBUF are controlled by the debugger at this time)
- 1=    SWD debug interface disabled. PA13 can be used as GPIO
- Bit1      FRZEN0:    When the debugger is connected and the CPU is in debug state (HALTED=1), the timer system peripheral module acts/stops.<sup>Note1</sup>
- 0=    Peripheral actions
- 1=    Peripheral Stop
- Bit0      FRZEN1:    When the debugger is connected and the CPU is in debug state (HALTED=1), the peripheral module of the communication system acts/stops.<sup>Note2</sup>
- 0=    Peripheral actions
- 1=    Peripheral stop

Note1: The timer system peripheral module of this product includes: Timer8, a general-purpose timer unit, Timer A, a 15-bit interval timer and a real-time clock (RTC).

Note2: The communication system peripheral module of this product includes: communication serial communication unit, serial interface SPI and serial interface IICA.

## 1.5 ARM reference document

The built-in debugging features in the Cortex®-M0+ kernel are part of the ARM® CoreSight design suite.

For documentation, refer to:

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

## Chapter 2 Port Function

### 2.1 Port general function

The general purpose I/O ports (GPIO) for this product are PA00~PA15, PB00~PB15, PC00~PC12, PD02~PD09, and PH00~PH04. The general purpose I/O ports used vary by product model, so please refer to the datasheet for each product series for details.

Each GPIO port has associated control and configuration registers that can be individually configured by software to multiple modes to meet the needs of specific applications. The basic operating modes are:

- Input floating
- Input pull-up
- Input pull-down
- Open-drain output (support pull-up)
- Push-pull output
- Analog channel

When the port is configured as an analog port, the digital function is isolated and the numbers "1" and "0" cannot be output, so the result of reading the port readback register PREADxx is "0".

PA00~PA07, PB00~PB02, PB10~PB15, PC00~PC06, PD04, PD05, PD08, PD09 ports can be used as analog channels and these ports are analog ports by default after reset. To use their digital function, you need to configure the corresponding bit of PMCxx register to 0.

PA08~PA15, PB03~PB08, PC07~PC12, PD02, PD06, PD07, PH00~PH04 can only be used as digital ports.

When used as digital GPIOs, all ports except PH00, PH03, and PH04 support output (push-pull or open-drain), input, pull-up, and pull-down functions. PH00 does not support pull-down, and PH03 and PH04 do not support pull-up and pull-down.

In order to avoid through-current, turn off the input function by default after reset PB03~PB08. If you want to turn on the input function of GPIO, you need to configure ISCLCD register, and when ISCLCD.ISCCAP is set to 1, the Schmitt input of PB03 and PB04 is enabled. When ISCLCD.ISCCAP is set to 1, the Schmitt input of PB05~PB08 will be enabled. For details, please refer to "Chapter 20 LCD Controller/Driver".

PA13 and PA14 ports are used as debug ports by default and are pulled up by default after reset. To use their GPIO function, you need to configure DBGSTOPCR. SWDIS register bit to 1 to mask the debug function. The corresponding relationship and default status of the debug ports are shown in Table 2-1: SW debug port pins, please refer to "Chapter 1 CPU" for the specific usage of the debug function.

Table 2-1: SW debug port pins

Port name	SW debug port	IO Type	Default State
PA13	SWDIO	I/O	Internal pull-up
PA14	SWCLK	I	Internal pull-up

The default state of the common digital ports (PA08~PA12, PA15, PC07~PC12, PD02, PD06, PD07, PH01~PH04) after chip reset is high resistance input (floating input), which aims to prevent abnormal actions to external devices when the chip is abnormally reset. However, in order to avoid the leakage caused by high resistance input, users should configure the ports accordingly (configured as internal pull-up/pull-down inputs or outputs) after the chip is started.

## 2.2 Port multiplexing function

In addition to supporting general-purpose GPIO functions, each port can also be multiplexed as a function port for peripheral modules. Such as input and output signals of analog module ADC/DAC/CMP/AMP/ LCD, input and output signals of digital function modules (such as SPI, UART, I<sup>2</sup>C, Timer, etc.). When using the port multiplexing function, please follow the following instructions for configuration:

- Each port can be configured as an external interrupt, and the interrupt trigger type can be configured as rising edge triggered, falling edge triggered or double edges triggered. The interrupt port selection requires the configuration of INTPnPCFG registers (n=0~5) for details please refer to 2.3.11 External interrupt port selection register (INTPnPCFG) To select the trigger type, please configure the EPG0 and EGN0 registers, please refer to "Chapter 23 Interrupt Function" for details.
- PH00 (RESETB) is used as an external reset input port by default and is pulled up by default. To use its GPIO function you need to configure the RSTM register to mask the reset function of PH00 port. PH00 port does not support pull-down function.
- PH01 (X1) and PH02 (X2/EXCLK) can be configured as external high-speed crystal oscillator ports. For details on the configuration and use of these oscillator ports, see "Chapter 4 Clock generation circuit".
- PH03 (XT1) and PH04 (XT2/EXCLKS) can be configured as external low-speed crystal oscillator ports. The configuration and usage of these ports as oscillator ports are described in "Chapter 4 Clock generation circuit". These two ports do not support pull-up and pull-down functions.
- When the port is multiplexed as LCD port (COM0~7, SEG0~41, VLCD1~4, CAPL, CAPH), the corresponding bits of the port mode control register PMCxx should be configured to 0, and the corresponding bits of the LCD port mode registers SEG0~SEG3 should be configured to 1. For details, please refer to "Chapter 20 LCD Controller/Driver".
- Each port with multiplexing function corresponds to a port multiplexing function configuration register PxxCFG, by setting PxxCFG you can select function 1~7 for each port respectively, be careful not to configure the same multiplexing input function to different ports, please refer to the operation description of PxxCFG register 2.3.10 Port multiplexing function configuration register (PxxCFG). For the detailed setting of the port multiplexing function, please refer to Configuration methods of the multiplexing function.

Table 2-2: List of multiplexing functions by port

Port function configuration							
Default Function	Function 1	Function 2	Function 3	Function 4	Function 5	Function 6	Function 7
PA00	TXD2/SDO20	TI00	TO00	-	-	VC0OUT	-
PA01	RXD2/SDI20/ SDA20	-	TI01/TO01	-	SPI0_MOSI	-	PCLBUZ0
PA02	TXD1/SDO10	-	TI02/TO02	-	SPI0_MISO	VC1OUT	PCLBUZ1
PA03	RXD1/SDI10/ SDA10	-	TI03/TO03	TI00_GATE	SPI0_NSS	-	-
PA04	TXD1/SDO10	-	TI04/TO04		SPI0_NSS	-	-
PA05	SS10	-	TI05/TO05	TI06_GATE	SPI0_SCK	-	PCLBUZ0
PA06	SS20	-	TI06/TO06	TI07_GATE	SPI0_MISO	VC0OUT	-
PA07	SCLK10/SCL10	-	TI07/TO07	-	SPI0_MOSI	VC1OUT	PCLBUZ1
PA08	TXD0/SDO00	TI00	TO00	TI01_GATE	-	-	-
PA09	TXD0/SDO00	SCLA0	TI01/TO01	TI02_GATE	-	-	PCLBUZ0
PA10	RXD0/SDI00/ SDA00	SDAA0	TI02/TO02	TI03_GATE	-	-	PCLBUZ1
PA11	SS00	SCLA0	TI03/TO03	TI04_GATE	SPI0_MISO	VC0OUT	-
PA12	SS11	SDAA0	TI04/TO04	TI05_GATE	SPI0_MOSI	VC1OUT	-
PA13/ SWDIO	RXD0/SDI00/ SDA00	-	TI05/TO05	RTC1HZ	-	KR4	PCLBUZ0
PA14/ SWCLK	TXD0/SDO00	-	TI06/TO06	-	-	KR1	PCLBUZ1
PA15	RXD0/SDI00/ SDA00	-	TI07/TO07	-	SPI0_NSS	KR0	-
PB00	TXD1/SDO10	TI00	TO00	-	-	-	PCLBUZ0
PB01	SCLK20/SCL20	-	TI01/TO01	-	-	-	PCLBUZ1
PB02	TXD2/SDO20	-	TI02/TO02	-	-	TA_TI/TA_TO	
PB03	SCLK11/SCL11	-	TI03/TO03	TI04_GATE	SPI0_SCK		PCLBUZ0
PB04	SDI11/SDA11	-	TI04/TO04	TI05_GATE	SPI0_MISO	TA_TI/TA_TO	-
PB05	SDO11	-	TI05/TO05	TI06_GATE	SPI0_MOSI		-
PB06	TXD0/SDO00	SCLA0	TI06/TO06	-	-	TA_TI/TA_TO	-
PB07	RXD0/SDI00/ SDA00	SDAA0	TI07/ TO07	-	-	TA_TON	-
PB08	SCLK00/SCL00	-	-	-	-	-	-
PB10	TXD1/SDO10	SCLA0	TI02/TO02	-	SPI0_SCK	-	-
PB11	RXD1/SDI10/ SDA10	SDAA0	TI03/TO03	TI00_GATE	--	-	-
PB12	TXD1/SDO10	-	TI04/TO04	-	SPI0_NSS	VC0OUT	-
PB13	SCLK10/SCL10	SCLA0	TI05/TO05	TI01_GATE	SPI0_SCK	-	-
PB14	SS01	SDAA0	TI06/TO06	RTC1HZ	SPI0_MISO	-	-
PB15	RXD2/SDI20/ SDA20	-	TI07/TO07	TI02_GATE	SPI0_MOSI	-	-

Port function configuration							
Default Function	Function 1	Function 2	Function 3	Function 4	Function 5	Function 6	Function 7
PC00	SS21	TI00	TO00	TI03_GATE	-	-	-
PC01	SCLK21/SCL21	-	TI01/TO01	-	-	TA_TI/ TA_TO	-
PC02	SDI21/SDA21	-	TI02/TO02	-	SPI0_MISO	TA_TON	-
PC03	SDO21	-	TI03/TO03	-	SPI0_MOSI	TA_TI/ TA_TO	-
PC04	TXD1/SDO10	-	TI04/TO04	-	-	-	PCLBUZ1
PC05	RXD1/SDI10/ SDA10	-	TI05/TO05	-	-	-	-
PC06	SCLK01/SCL01	-	TI06/TO06	-	-	-	-
PC07	SDI01/SDA01	-	TI07/TO07	-	-	KR7	-
PC08	SDO01	TI00	TO00	-	-	KR6	-
PC09	SCLK00/SCL00	-	TI01/TO01	-	-	KR5	-
PC10	TXD2/SDO20	-	TI02/TO02	-	-	-	-
PC11	RXD2/SDI20/ SDA20	-	TI03/TO03	-	-	-	-
PC12	TXD2/SDO20	-	TI04/TO04	-	-	-	-
PD02	SCLK20/SCL20	-	TI02/TO02	-	-	-	-
PD04	SCLK10/SCL10		-	-	-	-	-
PD05	SCLK20/SCL20		-	-	-	-	-
PD06	-	SCLA0	-	-	-	KR3	-
PD07	SCLK00/SCL00	SDAA0	-	-	-	KR2	-
PH00/ RESETB	-		-	-	-	-	-
PH01	TXD1/SDO10	SDAA0	-	-	-	-	-
PH02	RXD1/SDI10/ SDA10	SCLA0	TI01/ TO01	-	-	-	-
PH03	-	-	-	-	-	-	-
PH04	-	-	-	-	-	-	-

## 2.3 Registers for controlling port functions

The port function is controlled through the following registers.

- Port Output Control 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 Status Readback Register (PREADxx)
- Port Multiplexing Function Configuration Register (PxxCFG, see 2.3.10)
- External Interrupt Port Selection Register (INTPnPCFG, n=0~5)

Note: As the number of ports used and the configuration of the ports varies from product model to product, the public port control register varies from product to product. or the list of public port control registers for each product, please refer to Table 2-3.

Table 2-3: Port control registers assigned to each product

Port		Register Name								
		PMxx Register	Pxx Register	PSETxx Register	PCLRxx Register	PUxx Register	PDxx Register	POMxx Register	PREADxx Register	PMCxx Register
Port A	00	PMA0	PA0	PSETA0	PCLRA0	PUA0	PDA0	POMA0	PREADA0	PMCA0
	01	PMA1	PA1	PSETA1	PCLRA1	PUA1	PDA1	POMA1	PREADA1	PMCA1
	02	PMA2	PA2	PSETA2	PCLRA2	PUA2	PDA2	POMA2	PREADA2	PMCA2
	03	PMA3	PA3	PSETA3	PCLRA3	PUA3	PDA3	POMA3	PREADA3	PMCA3
	04	PMA4	PA4	PSETA4	PCLRA4	PUA4	PDA4	POMA4	PREADA4	PMCA4
	05	PMA5	PA5	PSETA5	PCLRA5	PUA5	PDA5	POMA5	PREADA5	PMCA5
	06	PMA6	PA6	PSETA6	PCLRA6	PUA6	PDA6	POMA6	PREADA6	PMCA6
	07	PMA7	PA7	PSETA7	PCLRA7	PUA7	PDA7	POMA7	PREADA7	PMCA7
	08	PMA8	PA8	PSETA8	PCLRA8	PUA8	PDA8	POMA8	PREADA8	-
	09	PMA9	PA9	PSETA9	PCLRA9	PUA9	PDA9	POMA9	PREADA9	-
	10	PMA10	PA10	PSETA10	PCLRA10	PUA10	PDA10	POMA10	PREADA10	-
	11	PMA11	PA11	PSETA11	PCLRA11	PUA11	PDA11	POMA11	PREADA11	-
	12	PMA12	PA12	PSETA12	PCLRA12	PUA12	PDA12	POMA12	PREADA12	-
	13	PMA13	PA13	PSETA13	PCLRA13	PUA13	PDA13	POMA13	PREADA13	-
	14	PMA14	PA14	PSETA14	PCLRA14	PUA14	PDA14	POMA14	PREADA14	-
	15	PMA15	PA15	PSETA15	PCLRA15	PUA15	PDA15	POMA15	PREADA15	-

Port		Bit Name								
		PMxx Register	Pxx Register	PSETxx Register	PCLRxx Register	PUxx Register	PDxx Register	POMxx Register	PREADxx Register	PMCxx Register
Port B	00	PMB0	PB0	PSETB0	PCLRB0	PUB0	PDB0	POMB0	PREADB0	PMCB0
	01	PMB1	PB1	PSETB1	PCLRB1	PUB1	PDB1	POMB1	PREADB1	PMCB1
	02	PMB2	PB2	PSETB2	PCLRB2	PUB2	PDB2	POMB2	PREADB2	PMCB2
	03	PMB3	PB3	PSETB3	PCLRB3	PUB3	PDB3	POMB3	PREADB3	-
	04	PMB4	PB4	PSETB4	PCLRB4	PUB4	PDB4	POMB4	PREADB4	-
	05	PMB5	PB5	PSETB5	PCLRB5	PUB5	PDB5	POMB5	PREADB5	-
	06	PMB6	PB6	PSETB6	PCLRB6	PUB6	PDB6	POMB6	PREADB6	-
	07	PMB7	PB7	PSETB7	PCLRB7	PUB7	PDB7	POMB7	PREADB7	-
	08	PMB8	PB8	PSETB8	PCLRB8	PUB8	PDB8	POMB8	PREADB8	-
	10	PMB10	PB10	PSETB10	PCLRB10	PUB10	PDB10	POMB10	PREADB10	PMCB10
	11	PMB11	PB11	PSETB11	PCLRB11	PUB11	PDB11	POMB11	PREADB11	PMCB11
	12	PMB12	PB12	PSETB12	PCLRB12	PUB12	PDB12	POMB12	PREADB12	PMCB12
	13	PMB13	PB13	PSETB13	PCLRB13	PUB13	PDB13	POMB13	PREADB13	PMCB13
	14	PMB14	PB14	PSETB14	PCLRB14	PUB14	PDB14	POMB14	PREADB14	PMCB14
	15	PMB15	PB15	PSETB15	PCLRB15	PUB15	PDB15	POMB15	PREADB15	PMCB15
Port C	00	PMC0	PC0	PSETC0	PCLRC0	PUC0	PDC0	POMC0	PREADC0	PMCC0
	01	PMC1	PC1	PSETC1	PCLRC1	PUC1	PDC1	POMC1	PREADC1	PMCC1
	02	PMC2	PC2	PSETC2	PCLRC2	PUC2	PDC2	POMC2	PREADC2	PMCC2
	03	PMC3	PC3	PSETC3	PCLRC3	PUC3	PDC3	POMC3	PREADC3	PMCC3
	04	PMC4	PC4	PSETC4	PCLRC4	PUC4	PDC4	POMC4	PREADC4	PMCC4
	05	PMC5	PC5	PSETC5	PCLRC5	PUC5	PDC5	POMC5	PREADC5	PMCC5
	06	PMC6	PC6	PSETC6	PCLRC6	PUC6	PDC6	POMC6	PREADC6	PMCC6
	07	PMC7	PC7	PSETC7	PCLRC7	PUC7	PDC7	POMC7	PREADC7	-
	08	PMC8	PC8	PSETC8	PCLRC8	PUC8	PDC8	POMC8	PREADC8	-
	09	PMC9	PC9	PSETC9	PCLRC9	PUC9	PDC9	POMC9	PREADC9	-
	10	PMC10	PC10	PSETC10	PCLRC10	PUC10	PDC10	POMC10	PREADC10	-
	11	PMC11	PC11	PSETC11	PCLRC11	PUC11	PDC11	POMC11	PREADC11	-
	12	PMC12	PC12	PSETC12	PCLRC12	PUC12	PDC12	POMC12	PREADC12	-
Port D	02	PMD2	PD2	PSETD2	PCLRD2	PUD2	PDD2	POMD2	PREADD2	-
	04	PMD4	PD4	PSETD4	PCLRD4	PUD4	PDD4	POMD4	PREADD4	PMCD4
	05	PMD5	PD5	PSETD5	PCLRD5	PUD5	PDD5	POMD5	PREADD5	PMCD5
	06	PMD6	PD6	PSETD6	PCLRD6	PUD6	PDD6	POMD6	PREADD6	-
	07	PMD7	PD7	PSETD7	PCLRD7	PUD7	PDD7	POMD7	PREADD7	-
RESETB/PH00		PMH0	PH0	PSETH0	PCLRH0	PUH0	-	POMH0	PREADH0	-
X1/PH01		PMH1	PH1	PSETH1	PCLRH1	PUH1	PDH1	POMH1	PREADH1	-
X2/PH02		PMH2	PH2	PSETH2	PCLRH2	PUH2	PDH2	POMH2	PREADH2	-
XT1/PH03		PMH3	PH3	PSETH3	PCLRH3	-	-	POMH3	PREADH3	-
XT2/PH04		PMH4	PH4	PSETH4	PCLRH4	-	-	POMH4	PREADH4	-



### 2.3.1 Port output control register (PMxx)

When the port is used as a digital channel, this is the register that sets whether its output is enabled or not in bit units. After a reset signal is generated, each port defaults to the input state. When the port is used as a multiplexed port, it must be set as described in "2.5 Register settings when using 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 the port output control register

After reset:	FFFFH																Offset addresses
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
PMA	PMA15	PMA14	PMA13	PMA12	PMA11	PMA10	PMA9	PMA8	PMA7	PMA6	PMA5	PMA4	PMA3	PMA2	PMA1	PMA0	0x010
PMB	PMB15	PMB14	PMB13	PMB12	PMB11	PMB10	PMB9	PMB8	PMB7	PMB6	PMB5	PMB4	PMB3	PMB2	PMB1	PMB0	0x012
PMC	1	PMC14	PMC13	PMC12	PMC11	PMC10	PMC9	PMC8	PMC7	PMC6	PMC5	PMC4	PMC3	PMC2	PMC1	PMC0	0x014
PMD	1						PMD9	PMD8	PMD7	PMD6	PMD5	PMD4	PMD3	PMD2	PMD1	PMD0	0x016
PMH	1											PMH4	PMH3	PMH2	PMH1	PMH0	0x01E
R/W	R/W																

PMmn: Selection of output mode of Pmn port (m=A,B,C,D,H, n=0~15)

0= Output mode (used as output port (output buffer ON))

1= Input mode (used as input port (output buffer OFF))

Note: The initial value must be set for the unassigned bits.

## 2.3.2 Port register (Pxx)

This register is used to set the value of the output latch for each port. Reading this register when PMxx is 0 gives the value of the output latch of the corresponding port, while reading this register when PMxx is 1 gives the port level of the corresponding port.

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 the port register

After reset:	0000H (output latch)																Offset addresses
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
PA	PA15	PA14	PA13	PA12	PA11	PA10	PA9	PA8	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	0x000
PB	PB15	PB14	PB13	PB12	PB11	PB10	0	PB8	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0	0x002
PC	0			PC12	PC11	PC10	PC9	PC8	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0	0x004
PD	0						PD9	PD8	PD7	PD6	PD5	PD4	0	PD2	0	0	0x006
PH	0											PH4	PH3	PH2	PH1	PH0	0x00E
R/W	R/W																

Pmn: m=A,B,C,D,H, n=0~15

Control of output data (output mode)      Reading of input data (input mode)

0=    The port outputs "0".                      Port input low level.

1=    The port outputs "1".                      Port input high level.

Note: The initial value must be set for the unassigned bits.

### 2.3.3 Port set control register (PSETxx)

This is the register to set the port output latch in bit units. After a reset signal is generated, the value of these registers becomes "0000H".

Register address = base address + offset address; the base address of Port Set Control Register is 0x40040000, and the offset address is shown in the figure below.

Figure 2-3: Format of the port set control register

After reset:	0000H																Offset addresses
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
PSETA	PSET A15	PSET A14	PSET A13	PSET A12	PSET A11	PSET A10	PSET A9	PSET A8	PSET A7	PSET A6	PSET A5	PSET A4	PSET A3	PSET A2	PSET A1	PSET A0	0x060
PSETB	PSET B15	PSET B14	PSET B13	PSET B12	PSET B11	PSET B10	PSET B9	PSET B8	PSET B7	PSET B6	PSET B5	PSET B4	PSET B3	PSET B2	PSET B1	PSET B0	0x062
PSETC	0	PSET C14	PSET C13	PSET C12	PSET C11	PSET C10	PSET C9	PSET C8	PSET C7	PSET C6	PSET C5	PSET C4	PSET C3	PSET C2	PSET C1	PSET C0	0x064
PSETD	0						PSET D9	PSET D8	PSET D7	PSET D6	PSET D5	PSET D4	PSET D3	PSET D2	PSET D1	PSET D0	0x066
PSETH	0											PSET H4	PSET H3	PSET H2	PSET H1	PSET H0	0x06E
R/W	W																

PSETmn: Set control of Pmn port (m=A,B,C,D,H, n=0~15)

0= No operation

1= Set the corresponding Pmn to 1

Note: The initial value must be set for the unassigned bits.

### 2.3.4 Port clear control register (PCLRxx)

This is the register to set the port output latch in bit units. After a reset signal is generated, the value of these registers becomes "0000H".

Register address = base address + offset address; the base address of Port Clear Control Register is 0x40040000, and the offset address is shown in the figure below.

Figure 2-4: Format of the port clear control register

After reset:	0000H																Offset addresses
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
PCLRA	PCLR A15	PCLR A14	PCLR A13	PCLR A12	PCLR A11	PCLR A10	PCLR A9	PCLR A8	PCLR A7	PCLR A6	PCLR A5	PCLR A4	PCLR A3	PCLR A2	PCLR A1	PCLR A0	0x070
PCLRB	PCLR B15	PCLR B14	PCLR B13	PCLR B12	PCLR B11	PCLR B10	PCLR B9	PCLR B8	PCLR B7	PCLR B6	PCLR B5	PCLR B4	PCLR B3	PCLR B2	PCLR B1	PCLR B0	0x072
PCLRC	0	PCLR C14	PCLR C13	PCLR C12	PCLR C11	PCLR C10	PCLR C9	PCLR C8	PCLR C7	PCLR C6	PCLR C5	PCLR C4	PCLR C3	PCLR C2	PCLR C1	PCLR C0	0x074
PCLRD	0						PCLR D9	PCLR D8	PCLR D7	PCLR D6	PCLR D5	PCLR D4	PCLR D3	PCLR D2	PCLR D1	PCLR D0	0x076
PCLRH	0											PCLR H4	PCLR H3	PCLR H2	PCLR H1	PCLR H0	0x07E
R/W	W																

PCLRmn: Clear control of Pmn port (m=A,B,C,D,H, n=0~15)

0= No operation

1= Set the corresponding Pmn to 0

Note: The initial value must be set for the unassigned bits.

## 2.3.5 Pull-up resistor selection register (PUxx)

Pull-up resistor selection register. Setting this register allows the port in input mode (PMmn=1) or in N-channel open drain output mode to be pulled up by using an internal pull-up resistor in bit units. For ports set to output mode, independent of the pull-up resistor selection register setting, no internal pull-up resistor is connected. Pull-ups are also not possible when the output port is used as a multiplexed function or when it is set to an analogue function.

After the reset signal is generated, the pull-up function of the three ports PA13, PA14 and PH00 is turned on by default (PUA13, PUA14 and PUH0 reset value is "1"), while the pull-up function of the other ports is not turned on by default.

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

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

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset address	After reset:
PUA	PUA 15	PUA 14	PUA 13	PUA 12	PUA 11	PUA 10	PUA 9	PUA 8	PUA 7	PUA 6	PUA 5	PUA 4	PUA 3	PUA 2	PUA 1	PUA 0	0x020	0x02E
PUB	PUB 15	PUB 14	PUB 13	PUB 12	PUB 11	PUB 10	PUB 9	PUB 8	PUB 7	PUB 6	PUB 5	PUB 4	PUB 3	PUB 2	PUB 1	PUB 0	0x022	0000H
PUC	0	PUC 14	PUC 13	PUC 12	PUC 11	PUC 10	PUC 9	PUC 8	PUC 7	PUC 6	PUC 5	PUC 4	PUC 3	PUC 2	PUC 1	PUC 0	0x024	0000H
PUD	0						PUD 9	PUD 8	PUD 7	PUD 6	PUD 5	PUD 4	PUD 3	PUD 2	PUD 1	PUD 0	0x026	0000H
PUH	0												PUH 2	PUH 1	PUH 0		0x02E	0001H
R/W	R/W																	

PUmn: Selection of internal pull-up resistors of Pmn port (m=A,B,C,D,H, n=0~15)

0= No internal pull-up resistor is connected.

1= Internal pull-up resistor is connected.

Note: The initial value must be set for the unassigned bits.

## 2.3.6 Pull-down resistor selection register (PDxx)

Internal pull-down resistor selection register. Setting this register allows the port in input mode (PMmn=1) to be pulled down by using an internal pull-down resistor in bit units. For ports set to output mode, independent of the pull-down resistor selection register setting, no internal pull-down resistor is connected. Pull-downs are also not possible when the output port is used as a multiplexed function or when it is set to an analogue function.

After the reset signal is generated, the pull-down function of the port is turned off by default and the reset value of the register is "0000H".

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

After reset:	0000H																
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset addresses
PDA	PDA15	PDA14	PDA13	PDA12	PDA11	PDA10	PDA9	PDA8	PDA7	PDA6	PDA5	PDA4	PDA3	PDA2	PDA1	PDA0	0x030
PDB	PDB15	PDB14	PDB13	PDB12	PDB11	PDB10	PDB9	PDB8	PDB7	PDB6	PDB5	PDB4	PDB3	PDB2	PDB1	PDB0	0x032
PDC	0	PDC14	PDC13	PDC12	PDC11	PDC10	PDC9	PDC8	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0	0x034
PDD	0						PDD9	PDD8	PDD7	PDD6	PDD5	PDD4	PDD3	PDD2	PDD1	PDD0	0x036
PDH	0													PDH2	PDH1	0	0x03E
R/W	R/W																

PDmn: Selection of internal pull-down resistors of Pmn port (m=A,B,C,D,H, n=0~15)

0= No internal pull-down resistor is connected.

1= Internal pull-down resistor is connected.

Note: The initial value must be set for the unassigned bits.

## 2.3.7 Port output mode register (POMxx)

This is a register for setting the output mode in bit units. The N-channel open drain output mode is selected for serial communication with external devices of different potentials and for I<sup>2</sup>C communication with external devices of the same potential.

After a reset signal is generated, the value of these registers becomes "0000H".

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

Figure 2-7: Format of the port output mode register

After reset:	0000H																Offset address
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
POMA	POMA15	POMA14	POMA13	POMA12	POMA11	POMA10	POMA9	POMA8	POMA7	POMA6	POMA5	POMA4	POMA3	POMA2	POMA1	POMA0	0x040
POMB	POMB15	POMB14	POMB13	POMB12	POMB11	POMB10	POMB9	POMB8	POMB7	POMB6	POMB5	POMB4	POMB3	POMB2	POMB1	POMB0	0x042
POMC	0	POMC14	POMC13	POMC12	POMC11	POMC10	POMC9	POMC8	POMC7	POMC6	POMC5	POMC4	POMC3	POMC2	POMC1	POMC0	0x044
POMD	0						POMD9	POMD8	POMD7	POMD6	POMD5	POMD4	POMD3	POMD2	POMD1	POMD0	0x046
POMH	0											POMH4	POMH3	POMH2	POMH1	POMH0	0x04E
R/W	R/W																

POMmn: Selection of output mode of Pmn port (m=A,B,C,D,H, n=0~15)

0= Usual output mode

1= N-channel open-drain output mode

Note: The initial value must be set for the unassigned bits.

## 2.3.8 Port mode control register (PMCxx)

The PMC register sets the port in bit units for use as an analog channel or for other functions, which include digital port functions and LCD port functions.

After generating the reset signal, PA00~PA07, PB00~PB03, PB10~PB15, PC00~PC06, PD04, PD05, PD08, PD09 are used as analog channels by default, and other ports are used as digital channels by default. The ports without PMC register have only digital function 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 the port mode control register

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Offset address	After reset:
PMCA	0							PMC A7	PMC A6	PMC A5	PMC A4	PMC A3	PMC A2	PMC A1	PMC A0		0x050	00FFH
PMCB	PMC B15	PMC B14	PMC B13	PMC B12	PMC B11	PMC B10	0					PMC B2	PMC B1	PMC B0		0x052	FC07H	
PMCC	0								PMC C6	PMC C5	PMC C4	PMC C3	PMC C2	PMC C1	PMC C0		0x054	407FH
PMCD	0									PMC D5	PMC D4	0					0x056	0333H
R/W	R/W																	

PMCMn: Pmn port is selected as an analog channel (m=A, B, C, D, n=0~15)

0= As a port other than analog, such as a digital port or LCD port

1= Analog Channel

Note: The initial value must be set for the unassigned bits.



## 2.3.9 Port readback register (PREADxx)

This is a read-only register and the port level can be obtained by reading this register when the port is used as a digital port.

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 2-9: Format of the port readback register

After reset:	xxxxH																Offset addresses
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
PREADA	PREA DA15	PREA DA14	PREA DA13	PREA DA12	PREA DA11	PREA DA10	PREA DA9	PREA DA8	PREA DA7	PREA DA6	PREA DA5	PREA DA4	PREA DA3	PREA DA2	PREA DA1	PREA DA0	0x080
PREADB	PREA DB15	PREA DB14	PREA DB13	PREA DB12	PREA DB11	PREA DB10	PREA DB9	PREA DB8	PREA DB7	PREA DB6	PREA DB5	PREA DB4	PREA DB3	PREA DB2	PREA DB1	PREA DB0	0x082
PREADC	0	PREA DC14	PREA DC13	PREA DC12	PREA DC11	PREA DC10	PREA DC9	PREA DC8	PREA DC7	PREA DC6	PREA DC5	PREA DC4	PREA DC3	PREA DC2	PREA DC1	PREA DC0	0x084
PREADD	0						PREA DD9	PREA DD8	PREA DD7	PREA DD6	PREA DD5	PREA DD4	PREA DD3	PREA DD2	PREA DD1	PREA DD0	0x086
PREADH	0											PREA DH4	PREA DH3	PREA DH2	PREA DH1	PREA DH0	0x08E
R/W	R																

Remark: The readout value of the register after reset depends on the status of each port.

PREADmn: m=A,B,C,D,H, n=0~15

Output Mode/Input Mode

0= The port is low level.

1= The port is high level.

## 2.3.10 Port multiplexing function configuration register (PxxCFG)

The port multiplexing function configuration registers enables redirecting the digital input and output functions of peripheral modules to different ports. Each port corresponds to a port multiplexing function configuration register. The default function of ports other than PH00, PA13, PA14 is GPIO function after reset. In order to avoid through-current, turn off the input function by default after reset PB03~PB08. If you want to turn on the input function of the ports, you need to configure ISCLCD register, and when ISCLCD.ISCCAP is set to 1, the digital inputs of PB03 and PB04 are enabled. When ISCLCD.ISCCAP is set to 1, the digital inputs of PB05~PB08 are enabled. The reset value of Port Multiplexing Configuration Register is "00H".

Register address = base address + offset address; the base address of PxxCFG register is 0x40040400, and the offset address is shown in Table 2-5: List of port multiplexing function configuration registers.

Figure 2-10: Format of the port multiplexing function configuration register

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

Table 2-4: Selection method of port multiplexing function

Register Name	Register Configuration	Port Function
PxxCFG[2:0]	3'b000(default)	Default Function
	3'b001	Function 1
	3'b010	Function 2
	3'b011	Function 3
	3'b100	Function 4
	3'b101	Function 5
	3'b110	Function 6
	3'b111	Function 7

Each port corresponds to a PxxCFG register, by setting PxxCFG you can select the 1st function to the 7th function for each port respectively, see Table 2-2: List of multiplexing functions by port. Be careful not to configure the same multiplexed input function to different ports.

Table 2-5: List of port multiplexing function configuration registers

Base Address	Offset Address	Register Name	R/W	Bit Width	Reset Value
0x40040400	0x000	PA00CFG[2:0]	R/W	8	00H
	0x001	PA01CFG[2:0]	R/W	8	00H
	0x002	PA02CFG[2:0]	R/W	8	00H
	0x003	PA03CFG[2:0]	R/W	8	00H
	0x004	PA04CFG[2:0]	R/W	8	00H
	0x005	PA05CFG[2:0]	R/W	8	00H
	0x006	PA06CFG[2:0]	R/W	8	00H
	0x007	PA07CFG[2:0]	R/W	8	00H
	0x008	PA08CFG[2:0]	R/W	8	00H
	0x009	PA09CFG[2:0]	R/W	8	00H
	0x00a	PA10CFG[2:0]	R/W	8	00H
	0x00b	PA11CFG[2:0]	R/W	8	00H
	0x00c	PA12CFG[2:0]	R/W	8	00H
	0x00d	PA13CFG[2:0]	R/W	8	00H
	0x00e	PA14CFG[2:0]	R/W	8	00H
	0x00f	PA15CFG[2:0]	R/W	8	00H
	0x010	PB00CFG[2:0]	R/W	8	00H
	0x011	PB01CFG[2:0]	R/W	8	00H
	0x012	PB02CFG[2:0]	R/W	8	00H
	0x013	PB03CFG[2:0]	R/W	8	00H
	0x014	PB04CFG[2:0]	R/W	8	00H
	0x015	PB05CFG[2:0]	R/W	8	00H
	0x016	PB06CFG[2:0]	R/W	8	00H
	0x017	PB07CFG[2:0]	R/W	8	00H
	0x018	PB08CFG[2:0]	R/W	8	00H
	0x01a	PB10CFG[2:0]	R/W	8	00H
	0x01b	PB11CFG[2:0]	R/W	8	00H
	0x01c	PB12CFG[2:0]	R/W	8	00H
	0x01d	PB13CFG[2:0]	R/W	8	00H
	0x01e	PB14CFG[2:0]	R/W	8	00H
	0x01f	PB15CFG[2:0]	R/W	8	00H

Base Address	Offset Address	Register Name	R/W	Bit Width	Reset Value
0x40040400	0x020	PC00CFG[2:0]	R/W	8	00H
	0x021	PC01CFG[2:0]	R/W	8	00H
	0x022	PC02CFG[2:0]	R/W	8	00H
	0x023	PC03CFG[2:0]	R/W	8	00H
	0x024	PC04CFG[2:0]	R/W	8	00H
	0x025	PC05CFG[2:0]	R/W	8	00H
	0x026	PC06CFG[2:0]	R/W	8	00H
	0x027	PC07CFG[2:0]	R/W	8	00H
	0x028	PC08CFG[2:0]	R/W	8	00H
	0x029	PC09CFG[2:0]	R/W	8	00H
	0x02a	PC10CFG[2:0]	R/W	8	00H
	0x02b	PC11CFG[2:0]	R/W	8	00H
	0x02c	PC12CFG[2:0]	R/W	8	00H
	0x032	PD02CFG[2:0]	R/W	8	00H
	0x034	PD04CFG[2:0]	R/W	8	00H
	0x035	PD05CFG[2:0]	R/W	8	00H
	0x036	PD06CFG[2:0]	R/W	8	00H
	0x037	PD07CFG[2:0]	R/W	8	00H
	0x038	PD08CFG[2:0]	R/W	8	00H
	0x039	PD09CFG[2:0]	R/W	8	00H
	0x080	PH00CFG[2:0]	R/W	8	00H
	0x081	PH01CFG[2:0]	R/W	8	00H
	0x082	PH02CFG[2:0]	R/W	8	00H
	0x083	PH03CFG[2:0]	R/W	8	00H
	0x084	PH04CFG[2:0]	R/W	8	00H

## 2.3.11 External interrupt port selection register (INTPnPCFG)

This product supports 6 external interrupts INTP0~5, and each external interrupt can be redirected to multiple ports. By configuring the external interrupt port selection register (INTPnPCFG), the input function of INTPn can be redirected to a different port. The reset value of the external interrupt port selection register is "00H". (n=0~5)

Register address = base address + offset address; the base address of INTPnPCFG register is 0x40040400, and the offset address is shown in the following table.

Table 2-6: List of external interrupt port selection register

Register Name	Offset Address	Function	R/W	Reset Value
INTP0PCFG	0x0E0	INTP0 redirect port selection	R/W	00H
INTP1PCFG	0x0E1	INTP1 redirect port selection	R/W	00H
INTP2PCFG	0x0E2	INTP2 redirect port selection	R/W	00H
INTP3PCFG	0x0E3	INTP3 redirect port selection	R/W	00H
INTP4PCFG	0x0E4	INTP4 redirect port selection	R/W	00H
INTP5PCFG	0x0E5	INTP5 redirect port selection	R/W	00H

Figure 2-11: Format of external interrupt port selection register

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

No.	Register Name	Register Configuration	INTP0 port selection
0	INTP0PCFG	4'h0(default)	PA00
1		4'h1	PA01
2		4'h2	PA02
3		4'h3	PA03
4		4'h4	PA04
5		4'h5	PA05
6		4'h6	PA06
7		4'h7	PA07
8		4'h8	PA08
9		4'h9	PA09
10		4'hA	PA10
11		4'hB	PA11
12		4'hC	PA12
13		4'hD	PA13
14		4'hE	PA14
15		4'hF	PA15

No.	Register Name	Register Configuration	INTP1 port selection
0	INTP1PCFG	4'h0(default)	PB00
1		4'h1	PB01
2		4'h2	PB02
3		4'h3	PB03
4		4'h4	PB04
5		4'h5	PB05
6		4'h6	PB06
7		4'h7	PB07

No.	Register Name	Register Configuration	INTP2 port selection
0	INTP2PCFG	4'h0(default)	PC00
1		4'h1	PC01
2		4'h2	PC02
3		4'h3	PC03
4		4'h4	PC04
5		4'h5	PC05
6		4'h6	PC06
7		4'h7	PC07
8		4'h8	PC08
9		4'h9	PC09
10		4'hA	PC10
11		4'hB	PC11
12		4'hC	PC12
13		4'hD	PC13
14		4'hE	PH03
15		4'hF	PH04

No.	Register Name	Register Configuration	INTP3 port selection
0	INTP3PCFG	4'h0(default)	PD00
1		4'h1	PD01
2		4'h2	PD02
3		4'h3	PB08
4		4'h4	PH00
5		4'h5	PH01
6		4'h6	PH02

No.	Register Name	Register Configuration	INTP4 port selection
0	INTP4PCFG	4'h0(default)	PC14
1		4'h1	PD04
2		4'h2	PD05
3		4'h3	PD06
4		4'h4	PD07
5		4'h5	PD08
6		4'h6	PD09

No.	Register Name	Register Configuration	INTP5 port selection
0	INTP5PCFG	4'h0(default)	PD03
1		4'h1	PB09
2		4'h2	PB10
3		4'h3	PB11
4		4'h4	PB12
5		4'h5	PB13
6		4'h6	PB14
7		4'h7	PB15

## 2.3.12 External reset port mask register (RSTM)

PH00 (RESETB) is used as an external reset input port by default, and system reset occurs when it is at low level. When you need to use the GPIO function of PH00, you need to configure the register RSTM to mask its external reset function first.

Figure 2-12: Format of the external reset port mask register

Base address: 0x40020400

Base Address:	0x40020400	After reset:	00H	Offset address:	0x00C				
Symbol	7	6	5	4	3	2	1	0	
RSTM	0							RSTM	
R/W	R/W								

Bit0      RSTM: External reset function mask for PH00 (RESETB) port  
             0= PH00 as an external reset port  
             1= PH00 as an GPIO port



## 2.4 Handling of unused ports

The processing of each unused port is Table 2-7 as follows.

Table 2-7: Handling of each unused port

Port name	I/O	Recommended connection method when not in use
PA00~PA15	I/O	Input: Connect to $EV_{DD}$ or $EV_{SS}$ via separate resistor. Output: Set to open circuit.
PB00~PB15		
PC00~PC12		
PD02~PD09		
PH01~PH04		Connect to $V_{DD}$ or $V_{SS}$ via separate resistor.
RESETB/PH00	I/O	Connect to $V_{DD}$ directly or via separate resistor.

Note: For products without  $EV_{DD}$  and  $EV_{SS}$  ports,  $EV_{DD}$  must be replaced with  $V_{DD}$  and  $EV_{SS}$  with  $V_{SS}$ .

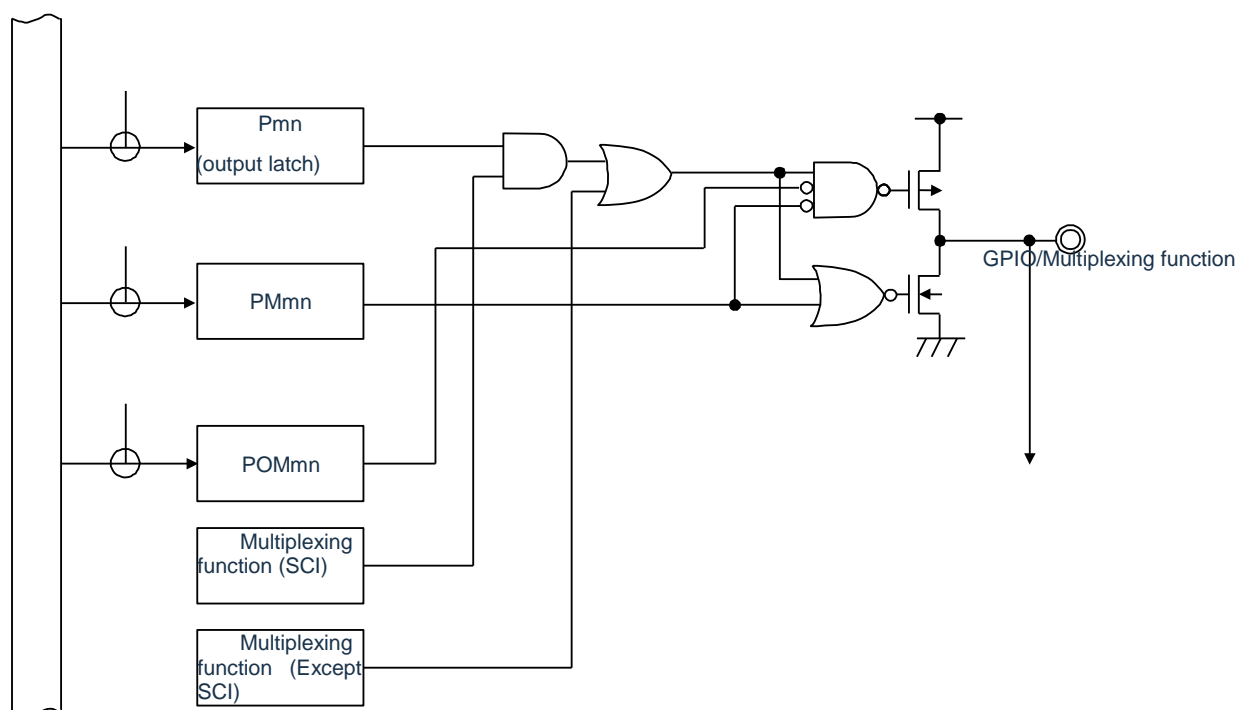
## 2.5 Register settings when using the multiplexing function

### Basic ideas when using multiplexing functions

Firstly, for ports with analogue functions, the Port Mode Control Register (PMCxx) sets whether the port is to be used for analogue or other functions.

The basic structure of the circuit when used as a digital input/output is shown in Figure2-13. The output of the multiplexed SCI function is connected to the AND gate together with the output latch of the port, and then connected to the OR gate together with the output of other multiplexed non-SCI functions (timer, RTC, clock/buzzer output, IICA, etc.). When using the port as a general-purpose output port or as a multiplexed output port, the basic idea of the setting is that the multiplexed function not in use cannot affect the output of the function to be used. As Table 2-8: Basic idea of setting is shown.

Figure2-13: Basic structure of the port output



Note 1: When there is no POM register, this signal is Low level (0).

Note 2: When there is no multiplexing function, this signal is High level (1).

Note 3: When there is no multiplexing function, this signal is Low level (0).

Table 2-8: Basic idea of setting

Port output functions in use	Output settings for multiplexed functions not in use		
	Port Function	SCI output function	Non-SCI output function
Universal digital output function	-	High level output (1)	Low level output (0)
SCI output function	High (1)	-	Low level output (0)
Non-SCI output function	Low (0)	High level output (1)	Low level output (0) <sup>Note</sup>

Since it is possible to multiplex multiple output functions other than SCI on one port, it is necessary to set the output of the multiplexed function not in use to Low level (0). For the specific setting method, refer to "Table 2-9: Configuration methods of the multiplexing function".

## Configuration methods of the multiplexing function

Table 2-10: Configuration methods of the multiplexing function

Port Name	Port Mode	Port Function									Register Settings			
		Default Function	Function 1 Function n	Function 2 Function	Function 3 Function	Function 4 Function	Function 5 Function	Function 6 Function	Function 7 Function	LCD Function	PMc xx	PMx x	Pxx	POM xx
PA00	Analog Channel	ANI00	-	-	-	-	-	-	-	-	1	x	x	x
		VC0_INP0	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel	-	-	-	-	-	-	-	-	SEG23	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type AND	-	TXD2/ SDO20	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output type OR	-	-	-	TO0 0	-	-	VC0OUT	-	-	0	0	0	0
PA01	Analog Channel	ANI01	-	-	-	-	-	-	-	-	1	x	x	x
		VC0_INP1	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel	-	-	-	-	-	-	-	-	SEG22	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed bidirectional Open-drain output type AND	-	SDA20	-	-	-	-	-	-	-	0	0	1	1
	Multiplexed output type OR	-	-	-	TO0 1	-	SPI0_MO	-	PCLBUZ 0	-	0	0	0	0
PA02	Analog Channel	ANI02	-	-	-	-	-	-	-	-	1	x	x	x
		VC0_INP2	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel	-	-	-	-	-	-	-	-	SEG21	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type AND	-	TXD1/ SDO10	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output type OR	-	-	-	TO0 2	-	SPI0_SO	VC1OUT	PCLBUZ 1	-	0	0	0	0
PA03	Analog Channel	ANI03	-	-	-	-	-	-	-	-	1	x	x	x
		VC0_INP3	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel	-	-	-	-	-	-	-	-	SEG20	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed bidirectional Open-drain output type AND	-	SDA10	-	-	-	-	-	-	-	0	0	1	1
	Multiplexed output type OR	-	-	-	TO0 3	-	-	-	-	-	0	0	0	0
PA04	Analog Channel	ANI04	-	-	-	-	-	-	-	-	1	x	x	x
		VC0_INP4	-	-	-	-	-	-	-	-	1	x	x	x

		OPA0_DA												
	LCD channel	-	-	-	-	-	-	-	-	SEG19	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type AND	-	TXD1/SDO10	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output type OR	-	-	-	TO04	-	-	-	-	-	0	0	0	0
	Multiplexed output	-	-	-	TI04	-	SPI0_NSS	-	-	-	0	1	x	x

Port Name	Port Mode	Port Function									Register Settings			
		Default Function	Function 1 Function	Function 2 Function	Function 3 Function	Function 4 Function	Function 5 Function	Function 6 Function	Function 7 Function	LCD Function	PMc	PM	Px	PO Mxx
PA05	Analog Channel	ANI05	-	-	-	-	-	-	-	-	1	x	x	x
		VC0_INN2 / VC1_INN2												
		DAC0												
	LCD channel	-	-	-	-	-	-	-	-	SEG18	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO05	-	SPI0_SCK	-	PCLBUZ0	-	0	0	0	0
Multiplexed output	-	SS10	-	TI05	TI06_GATE	SPI0_SCK	-	-	-	0	1	x	x	
PA06	Analog Channel	ANI06	-	-	-	-	-	-	-	-	1	x	x	x
		VC0_INN0												
	LCD channel		-	-	-	-	-	-	-	SEG17	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO06	-	SPI0_SO	VC0OUT	-	-	0	0	0	0
	Multiplexed output	-	SS20	-	TI06	TI07_GATE	SPI0_MI	-	-	-	0	1	x	x
PA07	Analog Channel	ANI07	-	-	-	-	-	-	-	-	1	x	x	x
		VC0_INN1												
	LCD channel	-	-	-	-	-	-	-	-	SEG16	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type AND	-	SCLK10/ SCL10	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output type OR	-	-	-	TO07	-	SPI0_MO	VC1OUT	PCLBUZ1	-	0	0	0	0
Multiplexed output	-	SCLK10	-	TI07	-	SPI0_SI	-	-	-	0	1	x	x	
PA08	LCD channel	-	-	-	-	-	-	-	-	SEG35	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type AND	-	TXD0/ SDO00	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output type OR	-	-	-	TO00	-	-	-	-	-	0	0	0	0
	Multiplexed output	-	-	TI00	-	TI01_GATE	-	-	-	-	0	1	x	x
PA09	LCD channel		-	-	-	-	-	-	-	COM0	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output	-	-		TI01	TI02_GATE	-	-	-	-	0	1	x	x
	Multiplexed bidirectional Open-drain output type OR	-	-	SCLA0	-	-	-	-	-	-	0	0	0	1
	Multiplexed output type OR	-	-	-	TO01		-	-	PCLBUZ0	-	0	0	0	0
	Multiplexed output type AND	-	TXD0/ SDO00	-	-	-	-	-	-	-	0	0	1	0

PA10	LCD channel	-	-	-	-	-	-	-	-	COM1	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output	-	RXD0/S DI00	-	TI02	TI03_GATE	-	-	-	-	0	1	x	x
	Multiplexed bidirectional Open-drain output type OR	-	-	SDAA 0	-	-	-	-	-	-	0	0	0	1
	Multiplexed bidirectional Open-drain output type AND	-	SDA00	-	-	-	-	-	-	-	0	0	1	1
	Multiplexed output type OR	-	-	-	TO02	-	-	-	PCLBUZ 1	-	0	0	0	0

Port Name	Port Mode	Port Function									Register Settings			
		Default Function	Function 1	Function 2	Function 3	Function 4	Function 5	Function 6	Function 7	LCD Function	PMc	PMx	Pxx	PO Mxx
PA11	LCD channel	-	-	-	-	-	-	-	-	COM2	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO03	-	SPI0_SO	VC0OUT	-	-	0	0	0	0
	Multiplexed bidirectional Open-drain output type OR	-	-	SCLA0	-	-	-	-	-	-	0	0	0	1
	Multiplexed output	-	SS00	-	TI03	TI04_GATE	SPI0_MI	-	-	-	0	1	x	x
PA12	LCD channel	-	-	-	-	-	-	-	-	COM3	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO04	-	SPI0_MO	VC1OUT	-	-	0	0	0	0
	Multiplexed bidirectional Open-drain output type OR	-	-	SDAA0	-	-	-	-	-	-	0	0	0	1
	Multiplexed output	-	SS11	-	TI04	TI05_GATE	SPI0_SI	-	-	-	0	1	x	x
PA13	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO05	RTC1HZ	-	-	PCLBUZ0	-	0	0	0	0
	Multiplexed bidirectional Open-drain output type AND	-	SDA00	-	-	-	-	-	-	-	0	0	1	1
	Multiplexed bidirectional	SWDIO	-	-	-	-	-	-	-	-	0	x	x	0
	Multiplexed output	-	RXD0/SDI00	-	TI05	-	-	KR4	-	-	0	1	x	x
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
PA14	Multiplexed output type OR	-	-	-	TO06	-	-	-	PCLBUZ1	-	0	0	0	0
	Multiplexed output type AND	-	TXD0/SDO00	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output	SWCLK	-	-	TI06	-	-	KR1	-	-	0	1	x	x
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
PA15	LCD channel	-	-	-	-	-	-	-	-	SEG32	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO07	-	-	-	-	-	0	0	0	0
	Multiplexed bidirectional Open-drain output type AND	-	SDA00	-	-	-	-	-	-	-	0	0	1	1
	Multiplexed output	-	RXD0/SDI00	-	TI07	-	SPI0_NSS	KR0	-	-	0	1	x	x

PB00	Analog Channel	ANI08/ AVREF M	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel		-	-	-	-	-	-	-	SEG13	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type AND	-	TXD1/ SDO1 0	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output type OR	-	-	-	TO00	-	-	-	PCLBUZ 0	-	0	0	0	0
	Multiplexed output	-	-	TI00	-	-	-	-	-	-	0	1	x	x
PB01	Analog Channel	ANI09/ AVREFP	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel		-	-	-	-	-	-	-	SEG12	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type AND	-	SCLK20 / SCL20	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output type OR	-	-	-	TO01	-	-	-	PCLBUZ 1	-	0	0	0	0
	Multiplexed output	-	SCLK20	-	TI01	-	-	-	-	-	0	1	x	x



Port Name	Port Mode	Port Function									Register Settings			
		Default Function	Function 1	Function 2	Function 3	Function 4	Function 5	Function 6	Function 7	LCD Function	PM Cxx	PMx x	Pxx	PO Mxx
PB02	Analog Channel	ANI10 VC1_INP0	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel		-	-	-	-	-	-	-	SEG11	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type AND	-	TXD2/ SDO20	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output type OR	-	-	-	TO02	-	-	TA_TO	-	-	0	0	0	0
	Multiplexed output	-	-	-	TI02	-	-	TA_TI	-	-	0	1	x	x
PB03	LCD channel		-	-	-	-	-	-	-	CAPH	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO03	-	SPI0_SCK	-	PCLBUZ0	-	0	0	0	0
	Multiplexed output type AND	-	SCLK11 / SCL11	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output	-	SCLK11	-	TI03	TI04_GATE	SPI0_SCK	-	-	-	0	1	x	x
PB04	LCD channel		-	-	-	-	-	-	-	CAPL	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO04	-	SPI0_SO	TA_TO	-	-	0	0	0	0
	Multiplexed bidirectional Open-drain output type AND	-	SDA11	-	-	-	-	-	-	-	0	0	1	1
	Multiplexed output	-	SDI11	-	TI04	TI05_GATE	SPI0_MI	TA_TI	-	-	0	1	x	x
PB05	LCD channel		-	-	-	-	-	-	-	VL4	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type AND	-	SDO11	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output type OR	-	-	-	TO05	-	SPI0_MO	-	-	-	0	0	0	0
	Multiplexed output	-	-	-	TI05	TI06_GATE	SPI0_SI	-	-	-	0	1	x	x
PB06	LCD channel		-	-	-	-	-	-	-	VL3	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed bidirectional Open-	-	-	SCLA0	-	-	-	-	-	-	0	0	0	1

	drain output type OR													
	Multiplexed output type OR	-	-	-	TO06	-	-	TA_TO	-	-	0	0	0	0
	Multiplexed output type AND	-	TXD0/SDO00	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output	-	-	-	TI06	-	-	TA_TI	-	-	0	1	x	x
PB07	LCD channel		-	-	-	-	-	-	-	VL2	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO07	-	-	TA_TON	-	-	0	0	0	0
	Multiplexed bidirectional Open-drain output type OR	-	-	SDAA0	-	-	-	-	-	-	0	0	0	1
	Multiplexed bidirectional Open-drain output type AND	-	SDA00	-	-	-	-	-	-	-	0	0	1	1
	Multiplexed output	-	RXD0/SDI00	-	TI07	-	-	-	-	-	0	1	x	x

Port Name	Port Mode	Port Function									Register Settings			
		Default Function	Function 1 Function	Function 2 Function	Function 3 Function	Function 4 Function	Function 5 Function	Function 6 Function	Function 7 Function	LCD Function	PM Cxx	PMx x	Pxx	PO Mxx
PB08	LCD channel	-	-	-	-	-	-	-	-	VL1	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	-	-	-	-	-	-	0	0	0	0
	Multiplexed output type AND	-	SCLK00/SCL00	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output	-	SCLK00	-	-	-	-	-	-	-	0	1	x	x
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed bidirectional Open-drain output type OR			SCLA0							0	0	0	1
	Multiplexed output	-	-			TI07_GATE	TI00	-	-	-	0	1	x	x
	Multiplexed output type OR	-	-	-	TO00	-	-	-	-	-	0	0	0	0
PB10	Analog Channel	ANI11	-	-	-	-	-	-	-	-	1	x	x	x
		VC1_INP1	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel		-	-	-	-	-	-	-	SEG10	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed bidirectional Open-drain output type OR			SCLA0							0	0	0	1
	Multiplexed output	-	-		TI02	-	SPI0_SCK	-	-	-	0	1	x	x
	Multiplexed output type AND	-	TXD1/SDO10	-	-	-	-	-	-	-	0	0	1	0
PB11	Analog Channel	ANI12	-	-	-	-	-	-	-	-	1	x	x	x
		VC1_INP2	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel		-	-	-	-	-	-	-	SEG09	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO03	-	-	-	-	-	0	0	0	0
PB11	Multiplexed bidirectional Open-	-	-	SDAA0	-	-	-	-	-	-	0	0	0	1

	drain output type OR													
	Multiplexed bidirectional Open-drain output type AND	-	SDA10	-	-	-	-	-	-	-	0	0	1	1
	Multiplexed output	-	RxD1/SDI10	-	TI03	TI00_GATE	-	-	-	-	0	1	x	x
PB1 2	Analog Channel	ANI13	-	-	-	-	-	-	-	-	1	x	x	x
		VC1_INP3	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel		-	-	-	-	-	-	-	SEG08	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-		TO04	-	-	VC0OUT	-	-	0	0	0	0
	Multiplexed output type AND	-	TXD1/SDO10			-	-	-	-	-	0	0	1	0
	Multiplexed output	KR7	-	-	TI04	-	SPI0_NSS	-	-	-	0	1	x	x

Port Name	Port Mode	Port Function									Register Settings			
		Default Function	Function 1 Function	Function 2 Function	Function 3 Function	Function 4 Function	Function 5 Function	Function 6 Function	Function 7 Function	LCD Function	PMC <sub>xx</sub>	PM <sub>xx</sub>	P <sub>xx</sub>	POM <sub>xx</sub>
PB13	Analog Channel	ANI14 VC1_INN0	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel									SEG07	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed bidirectional Open-drain output type OR	-	-	SCLA0	-	-	-	-	-	-	0	0	0	1
	Multiplexed output type OR	-	-	-	TO05	-	SPI0_SCK	-	-	-	0	0	0	0
	Multiplexed output type AND	-	SCLK10/ SCL10	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output	-	SCLK10	-	TI05	TI01_GATE	SPI0_SCK	-	-	-	0	1	x	x
PB14	Analog Channel	ANI15	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel									SEG06	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO06	RTC1HZ	SPI0_SO	-	-	-	0	0	0	0
	Multiplexed bidirectional Open-drain output type OR	-	-	SDAA0	-	-	-	-	-	-	0	0	0	1
	Multiplexed output	-	SS01	-	TI06	-	SPI0_MI	-	-	-	0	1	x	x
PB15	Analog Channel	ANI16	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel		-	-	-	-	-	-	-	SEG05	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO07	-	SPI0_MO	-	-	-	0	0	0	0
	Multiplexed bidirectional Open-drain output type AND	-	SDA20	-	-	-	-	-	-	-	0	0	1	1
	Multiplexed output	-	RxD2/ SDI20	-	TI07	TI02_GATE	SPI0_SI	-	-	-	0	1	x	x
PC00	Analog Channel	ANI17 VC1_INP4	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel		-	-	-	-	-	-	-	SEG27	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-			TO00					-	0	0	0	0
	Multiplexed output	-	SS21	TI00	-	TI03_GATE	-	-	-	-	0	1	x	x
PC01	Analog Channel	ANI18 VC1_INP5	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel		-	-	-	-	-	-	-	SEG26	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		

	Multiplexed output type AND	-	SCLK21/ SCL21							-	0	0	1	0
	Multiplexed output type OR	-			TO01			TA_TO		-	0	0	0	0
	Multiplexed output	-	SCLK21	-	TI01	-	-	TA_TI	-	-	0	1	x	x

Port Name	Port Mode	Port Function									Register Settings			
		Default Function	Function 1 Function	Function 2 Function	Function 3 Function	Function 4 Function	Function 5 Function	Function 6 Function	Function 7 Function	LCD Function	PMCxx	PMxx	Pxx	POMxx
PC02	Analog Channel	ANI19	-	-	-	-	-	-	-	-	1	x	x	x
		VC1_INN1	-	-	-	-	-	-	-	-	1	0	0	0
	LCD channel		-	-	-	-	-	-	-	SEG25	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-			TO02		SPI0_SO	TA_TON		-	0	0	0	0
	Multiplexed bidirectional Open-drain output type AND	-	SDA21							-	0	0	1	0
PC03	Multiplexed output	-	SDI21	-	TI02	-	SPI0_MI	-	-	-	0	1	x	x
	Analog Channel	ANI20	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel		-	-	-	-	-	-	-	SEG24	0	1	0	0
	Multiplexed output type OR	-			TO03		SPI0_MO	TA_TO		-	0	0	0	0
	Multiplexed output type AND	-	SDO21							-	0	0	1	0
PC04	Multiplexed output	-	-	-	TI03	-	SPI0_SI	TA_TI	-	-	0	1	x	x
	Analog Channel	ANI21	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel		-	-	-	-	-	-	-	SEG15	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-			TO04				PCLBUZ1	-	0	0	0	0
	Multiplexed output type AND	-	TXD1/SDO10							-	0	0	1	0
PC05	Multiplexed output	-	-	-	TI04	-	-	-	-	-	0	1	x	x
	Analog Channel	ANI22	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel		-	-	-	-	-	-	-	SEG14	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed bidirectional Open-drain output type AND	-	SDA10	-	-	-	-	-	-	-	0	0	1	1
	Multiplexed output type OR	-	-	-	TO05	-	-	-	-	-	0	0	0	0
PC06	Multiplexed output	-	RxD1/SDI10	-	TI05	-	-	-	-	-	0	1	x	x
	Analog Channel	ANI23	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel		-	-	-	-	-	-	-	SEG04	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-			TO06	-	-	-	-	-	0	0	0	0
	Multiplexed output type AND	-	SCLK01/SCL01	-	-	-	-	-	-	-	0	0	1	0
PC07	Multiplexed output	-	SCLK01	-	TI06	-	-	-	-	-	0	1	x	x
	LCD channel		-	-	-	-	-	-	-	SEG38	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO07	-	-	-	-	-	0	0	0	0
	Multiplexed bidirectional	-	SDA01	-	-	-	-	-	-	-	0	0	1	1

	Open-drain output type AND													
	Multiplexed output	-	SDI01	-	TI07	-	-	KR7	-	-	0	1	x	x



Port Name	Port Mode	Port Function									Register Settings			
		Default Function	Function 1	Function 2	Function 3	Function 4	Function 5	Function 6	Function 7	LCD Function	PMC <sub>x</sub>	PM <sub>x</sub>	P <sub>xx</sub>	POM <sub>x</sub>
PC08	LCD channel	-	-	-	-	-	-	-	-	SEG37	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO00	-	-	-	-	-	0	0	0	0
	Multiplexed output type AND	-	SDO01	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output	-	-	TI00	-	-	-	KR6	-	-	0	1	x	x
PC09	LCD channel	-	-	-	-	-	-	-	-	SEG36	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output	-	SCLK0	-	TI01	-	-	KR5	-	-	0	1	x	x
	Multiplexed output type OR	-	-	-	TO01	-	-	-	-	-	0	0	0	0
	Multiplexed output type AND	-	SCLK0/SCL00	-	-	-	-	-	-	-	0	0	1	0
PC10	LCD channel	-	-	-	-	-	-	-	-	SEG00/COM4	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output	-	-	-	TI02	-	-	-	-	-	0	1	x	x
	Multiplexed output type AND	-	TXD2/SDO20	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output type OR	-	-	-	TO02	-	-	-	-	-	0	0	0	0
PC11	LCD channel	-	-	-	-	-	-	-	-	SEG01/COM5	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO03	-	-	-	-	-	0	0	0	0
	Open-drain output type AND	-	SDA20	-	-	-	-	-	-	-	0	0	1	1
	Multiplexed output	-	RxD2/SDI20	-	TI03	-	-	-	-	-	0	1	x	x
PC12	LCD channel	-	-	-	-	-	-	-	-	SEG02/COM6	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO04	-	-	-	-	-	0	0	0	0
	Multiplexed output type AND	-	TXD2/SDO20	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output	-	-	-	TI04	-	-	-	-	-	0	1	x	x
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output	-	SS10	-	-	-	-	-	-	-	0	1	x	x
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		

Port Name	Port Mode	Port Function									Register Settings			
		Default Function	Function 1 Function	Function 2 Function	Function 3 Function	Function 4 Function	Function 5 Function	Function 6 Function	Function 7 Function	LCD Function	PMCxx	PMxx	Pxx	POMxx
PD02	LCD channel		-	-	-	-	-	-	-	SEG03/COM7	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO02	-	-	-	-	-	0	0	0	0
	Multiplexed output type AND	-	SCLK20/SCL20	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed output	-	SCLK20	-	TI02	-	-	-	-	-	0	1	x	x
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO01	-	-	-	PCLBUZ1	-	0	0	0	0
	Multiplexed bidirectional Open-drain output type AND	-	SDA00	-	-	-	-	-	-	-	0	0	1	1
	Multiplexed bidirectional Open-drain output type OR	-	-	SDAA0	-	-	-	-	-	-	0	0	0	1
	Multiplexed output	-	RXD0/SDI00	-	TI01	-	SPI0_NSS	-	-	-	0	1	x	x
PD04	Analog Channel	ANI24	-	-	-	-	-	-	-	-	1	x	x	x
		VC0_INP5												
		OPA00												
	LCD channel		-	-	-	-	-	-	-	-	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
PD05	Multiplexed output type AND	-	SCLK10/SCL10						-	-	0	0	1	0
	Multiplexed output	-	SCLK10	-		-	-	-	-	-	0	1	x	x
	Analog Channel	ANI25	-	-	-	-	-	-	-	-	1	x	x	x
	LCD channel	-	-	-	-	-	-	-	-	SEG39	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
PD06	Multiplexed output type AND	-	SCLK20/SCL20						-	-	0	0	1	0
	Multiplexed output	-	SCLK20	-		-	-	-	-	-	0	1	x	x
	LCD channel		-	-	-	-	-	-	-	SEG34	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed bidirectional Open-drain output type OR	-	-	SCLA0	-	-	-		-	-	0	0	0	1
PD07	Multiplexed output							KR3			0	1	x	x
	LCD channel		-	-	-	-	-	-	-	SEG33	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type AND	-	SCLK00/SCL00	-	-	-	-	-	-	-	0	0	1	1
	Multiplexed bidirectional Open-drain output type OR	-	-	SDAA0	-	-	-	-	-	-	0	0	0	1
PH00	Multiplexed output	-	SCLK00					KR2	-	-	0	1	x	x
	LCD channel	-	-	-	-	-	-	-	-	-	0	1	0	0
PH00	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		

Port Name	Port Mode	Port Function									Register Settings			
		Default Function	Function 1	Function 2	Function 3	Function 4	Function 5	Function 6	Function 7	LCD Function	PMCx	PMx	Pxx	POMx
PH0 1	LCD channel	-	-	-	-	-	-	-	-	-	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type AND	-	TXD1/SDO10	-	-	-	-	-	-	-	0	0	1	0
	Multiplexed bidirectional Open-drain output type OR	-	-	SDAA0	-	-	-	-	-	-	0	0	0	1
PH0 2	LCD channel	-	-	-	-	-	-	-	-	-	0	1	0	0
	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Multiplexed output type OR	-	-	-	TO01	-	-	-	-	-	0	0	0	0
	Multiplexed bidirectional	-	SDA10	-	-	-	-	-	-	-	0	0	1	1
	Open-drain output type AND	-	-	SCLA0	-	-	-	-	-	-	0	0	0	1
	Multiplexed bidirectional	-	-	-	-	-	-	-	-	-	0	0	0	1
PH0 3	Multiplexed output	-	RxD1/SDI10	-	TI01	-	-	-	-	-	0	1	x	x
	Analog Channel	-	-	-	-	-	-	-	-	-	1	x	x	x
PH0 4	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Analog Channel	-	-	-	-	-	-	-	-	-	1	x	x	x
PH0 4	Digital general	GPIO	-	-	-	-	-	-	-	-	0	Configure according to requirements		
	Analog Channel	-	-	-	-	-	-	-	-	-	1	x	x	x

**Configuration Description:**

- PA00~PA07, PB00~PB03, PB10~PB15, PC00~PC06, PD04, PD05, PD08, PD09 are used as analog ports by default after power on. If they are to be used for digital general purpose GPIO or digital multiplexing function, the ports need to be configured to digital mode (PMCxx=0).
- PA13 and PA14 ports are used as debug ports by default and are pulled up by default after reset. To use their GPIO function, you need to configure DBGSTOPCR. SWDIS register bit to 1 to mask the debug function. The DBGSTOPCR.SWDIS register is described in "Chapter 1 CPU".
- PA08~PA12, PA15, PC07~PC12, PD02, PD06, PD07, PH01~PH04 ports are used as digital GPIO by default after power on.
- To use the digital input function of the port, the port must be configured to input mode (PMxx=1).
- To use the digital output function of the port, the port must be configured to output mode (push-pull or open-drain) (PMxx=0).
- To use the multiplexed output function of the port, set the output latch Pxx of the port according to the above table.
- To use the GPIO function of PH00 port, mask the reset function of this port and set RSTM=1 at first.
- To use the GPIO function or multiplexing function of PH01 and PH02 ports, make sure that the X1 oscillation mode and external clock input mode are not turned on. Please refer to "Chapter 4 Clock generation circuit".
- To use the GPIO function or multiplexing function of PH03 and PH04 ports, make sure that the XT1 oscillation mode and external clock input mode are not turned on. Please refer to "Chapter 4 Clock generation circuit".
- Each port can be configured as an external interrupt, and the interrupt trigger type can be configured as rising edge triggered, falling edge triggered or double edges triggered. The interrupt port selection requires the configuration of INTPnPCFG registers (n=0~5) for details please refer to 2.3.11 External interrupt port selection register (INTPnPCFG) To select the trigger type, please configure the EPG0 and EGN0 registers, please refer to "Chapter 23 Interrupt Function" for details.
- When the port is multiplexed as LCD port (COM0~7, SEG0~41, VLCD1~4, CAPL, CAPH), the corresponding bits of the port mode control register PMCxx should be configured to 0, and the corresponding bits of the LCD port mode registers SEG0~SEG3 should be configured to 1. For details, please refer to "Chapter 20 LCD Controller/Driver".
- In order to avoid through-current, turn off the input function by default after reset PB03~PB08. If you want to turn on the input function of GPIO, you need to configure ISCLCD register, and when ISCLCD.ISCCAP is set to 1, the Schmitt input of PB03 and PB04 is enabled. When ISCLCD.ISCCAP is set to 1, the Schmitt input of PB05~PB08 will be enabled. For details, please refer to "Chapter 20 LCD Controller/Driver".

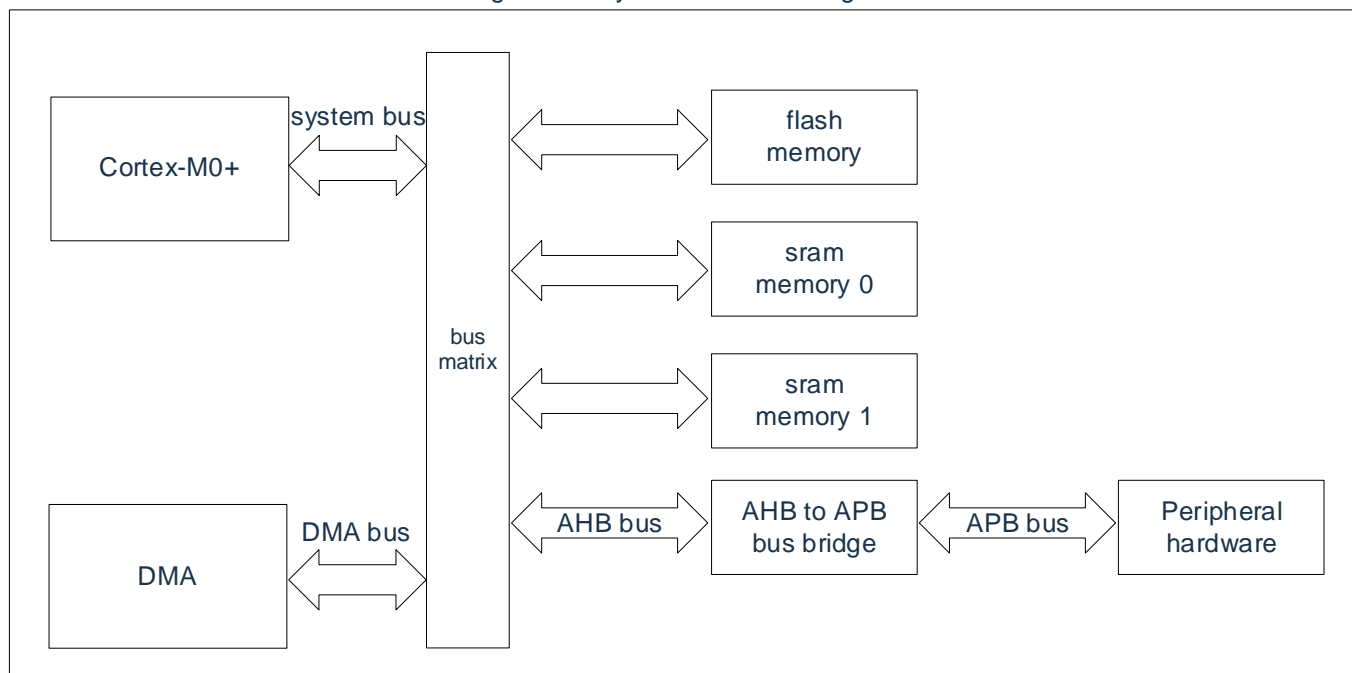
# Chapter 3 System Structure

## 3.1 Overview

This product system consists of the following components:

- 2 AHB buses Master:
  - Cortex-M0+
  - Enhanced DMA
- 4 AHB buses Slaves:
  - FLASH Storage
  - SRAM Memory 0
  - SRAM Memory 1
  - AHB to APB Bridge, contains all APB interface peripherals

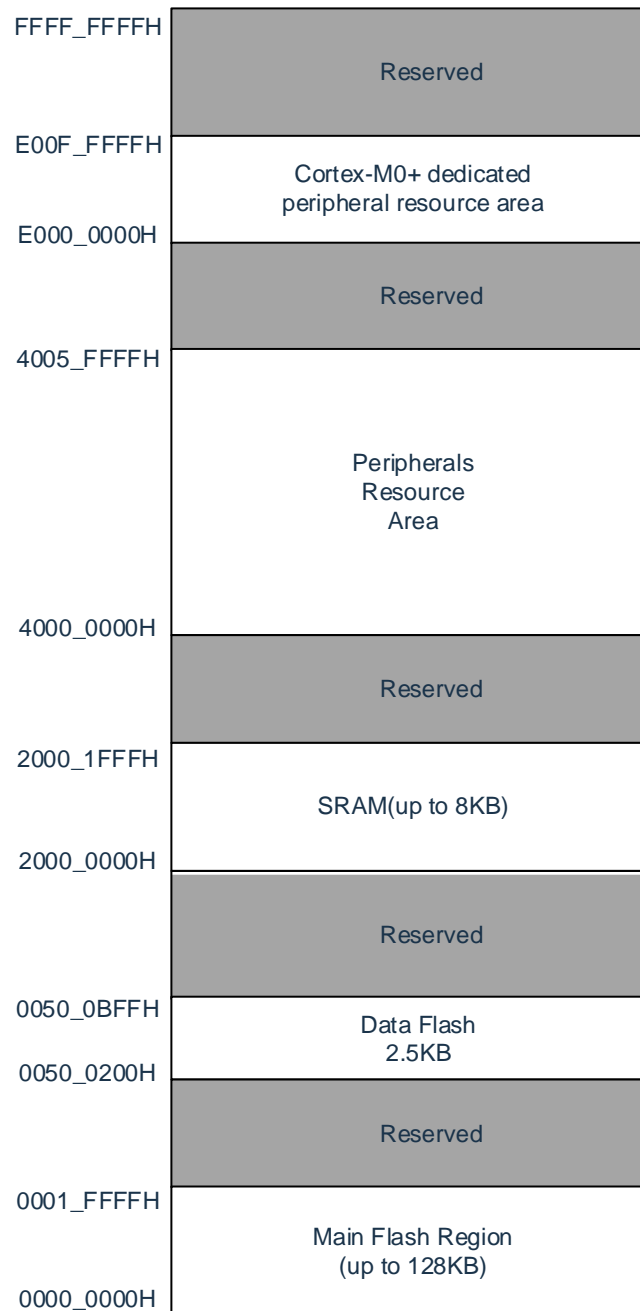
Figure3-1: System structure diagram



- System bus: This bus connects the Cortex-M0+ core's system bus (peripheral bus) to the bus matrix, which coordinates accesses between the kernel and the 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 access arbitration between the kernel system bus and the DMA master bus, with fixed priority for arbitration and high priority for DMA.
- AHB to APB Bridge: The AHB to APB Bridge provides a synchronous connection between the AHB and APB buses. For the address mapping of the different peripherals connected to each bridge, please refer to Figure3-1.

## 3.2 System address division

Figure3-2: Address division diagram



### 3.3 Peripheral address assignment

Figure3-1: Start address of the peripheral register group

Start Address	Peripheral	Remark
0x4000_0000 - 0x4000_4FFF	Retain	
0x4000_5000 - 0x4000_5FFF	DMA	
0x4000_6000 - 0x4000_6FFF	Interrupt control	
0x4000_7000 - 0x4001_8FFF	Retain	
0x4001_9000 - 0x4001_AFFF	Retain	
0x4001_B000 - 0x4001_BFFF	DBGREG	
0x4001_C000 - 0x4001_CFFF	DIV	
0x4001_D000 - 0x4001_FFFF	Retain	
0x4002_0000 - 0x4002_03FF	FLASH control	
0x4002_0400 - 0x4002_0FFF	Clock control	
0x4002_1000 - 0x4002_1002	Watch dog timer	
0x4002_1003 - 0x4002_1800	Retain	
0x4002_1800 - 0x4002_1BFF	High-speed CRC	See Chapter 29 Security Feature for more details
0x4002_1C00 - 0x4002_1FFF	FLASH control	
0x4002_2000 - 0x4003_FFFF	Retain	
0x4004_0000 - 0x4004_0FFF	GPIO	
0x4004_1000 - 0x4004_13FF	pcbz	
0x4004_1400 - 0x4004_17FF	General CRC	See Chapter 29 Security Feature for more details
0x4004_1800 - 0x4004_1BFF	Linkage controller	
0x4004_1C00 - 0x4004_1FFF	External interrupt control	
0x4004_2000 - 0x4004_23FF	Key interrupt	
0x4004_2400 - 0x4004_27FF	Real time clock	
0x4004_2800 - 0x4004_2FFF	Retain	
0x4004_3000 - 0x4004_33FF	General purpose timer unit	
0x4004_3400 - 0x4004_3FFF	Retain	
0x4004_4000 - 0x4004_43FF	TimerA	
0x4004_4400 - 0x4004_4FFF	Retain	
0x4004_5000 - 0x4004_53FF	AD converter	
0x4004_5400 - 0x4004_57FF	DAconverter	
0x4004_5800 - 0x4004_5BFF	Comparator	
0x4004_5C00 - 0x4004_5FFF	Amplifier	
0x4004_6000 - 0x4004_6BFF	Serial Communication Unit	
0x4004_6C00 - 0x4004_6FFF	Retain	
0x4004_7000 - 0x4004_73FF	Serial interfacellCA0	
0x4004_7400 - 0x4004_7FFF	Retain	
0x4004_7800 - 0x4004_7BFF	SPIHS0	
0x4004_7C00 - 0x4004_7FFF	Retain	
0x4004_8000 - 0x4004_83FF	IrDA	
0x4004_8400 - 0x4004_8FFF	Retain	
0x4004_9000 - 0x4004_93FF	LCD	

## Chapter 4 Clock generation circuit

### 4.1 Function of clock generation circuit

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

#### (1) Main system Clock

##### ① X1 oscillation circuit

The clock can oscillate at  $F_X = 1\sim 20\text{MHz}$  by connecting resonators to pins X1 and X2, and can be stopped by entering deep sleep mode or by setting the MSTOP bit (bit 7 of the Clock Operation Status Control Register (CSC)).

##### ② High-speed internal oscillator (high-speed OCO)

It is possible to oscillate at a frequency selected from  $F_{HOCO} = 32\text{MHz}, 16\text{MHz}, 8\text{MHz}, 4\text{MHz}, 2\text{MHz}$  and  $1\text{MHz}$  (typical values) by means of the option byte (000C2H), and the same frequency for  $F_{IH}$  and  $F_{HOCO}$ . The CPU must start running with this high-speed internal oscillator clock after the reset is released. Oscillation can be stopped by entering deep sleep mode or by setting the HIOSTOP bit (bit 0 of the CSC register). The frequency of the option byte setting can be changed by the frequency selection register (HOCODIV) of the high-speed internal oscillator. For frequency settings, please refer to "Figure4-10: Format of the high-speed internal oscillator frequency selection register (HOCODIV)".

In addition, an external main system clock ( $F_{EX} = 1\sim 20\text{MHz}$ ) can be provided by the EXCLK/X2/PH02 pin and the input to the external main system clock can be disabled by entering deep sleep mode or by setting the MSTOP bit.

It is possible to switch between the high-speed system clock (X1 clock or external main system clock) and the high-speed internal oscillator clock by setting the MCM0 bit (bit 4 of the system clock control register (CKC)).

#### (2) Subsystem Clock

##### • XT1 oscillation circuit

The clock can oscillate at  $F_{XT} = 32.768\text{KHz}$  by connecting resonators to pins XT1 and XT2, and can be stopped by setting the XTSTOP bit (bit 6 of the Clock Operation Status Control Register (CSC)).

In addition, an external subsystem clock can be provided by the EXCLKS/XT2/PH04 pin ( $F_{EXS} = 32.768\text{KHz}$ ) and the input to the external subsystem clock can be disabled by setting the XTSTOP bit.

#### (3) Low-speed internal oscillator clock (low-speed OCO)

It can make the clock oscillate at  $F_{IL} = 32.768\text{KHz}$  (typical value).

The low-speed internal oscillator clock can be used as the CPU clock.

The SysTick timer uses the low-speed internal oscillator clock as the external reference clock.

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

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



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

Remarks:  $F_X$  : X1 clock oscillation frequency

$F_{HOCO}$  : High-speed internal oscillator clock frequency (max. 32MHz)

$F_{IH}$  : High-speed internal oscillator clock frequency (max. 32MHz)

$F_{EX}$  : External main system clock frequency

$F_{XT}$  : XT1 clock oscillation frequency

$F_{EXS}$  : External subsystem system clock frequency

$F_{IL}$  : Clock frequency of the low-speed internal oscillator

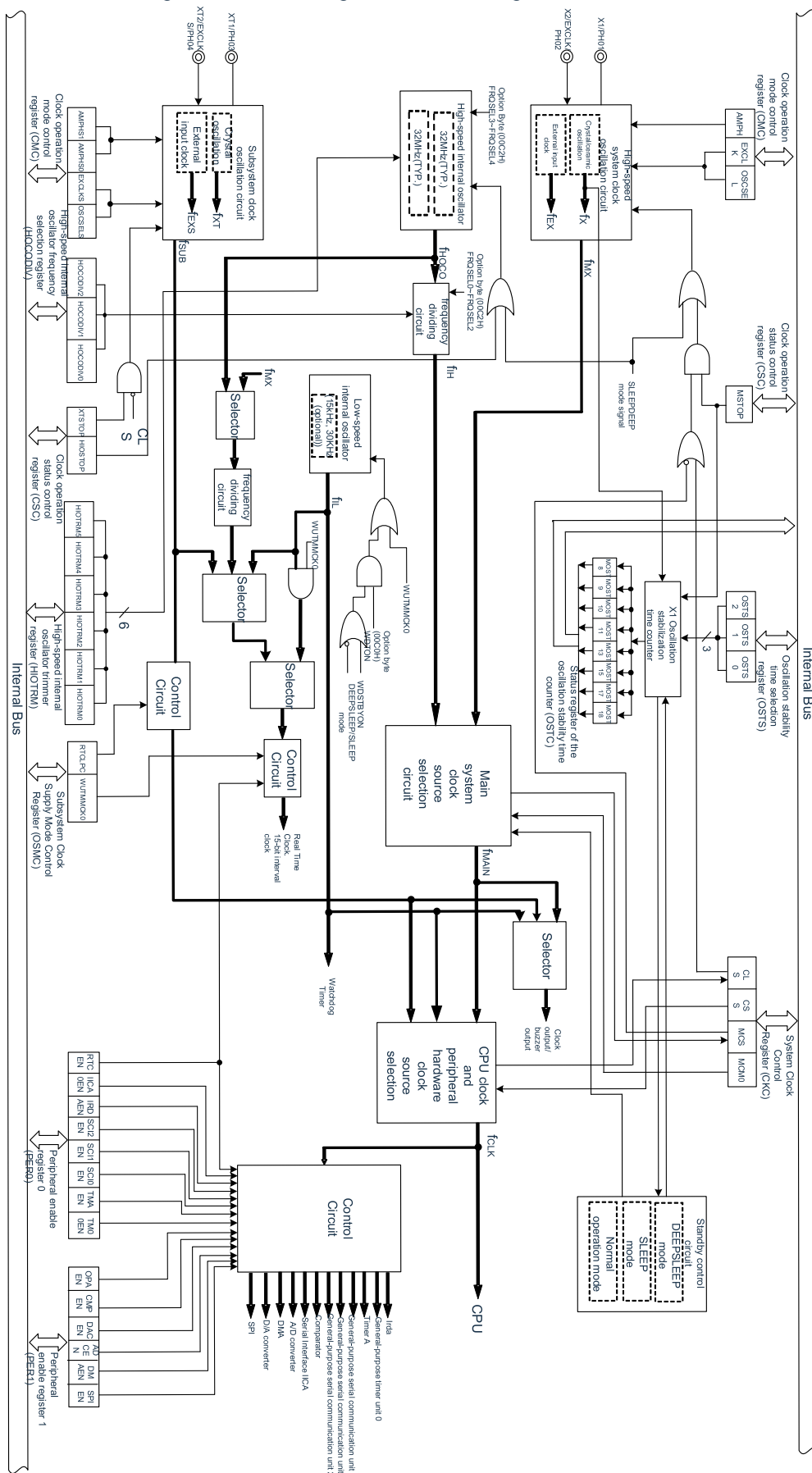
## 4.2 Structure of clock generation circuit

The clock generation circuit consists of the following hardware.

Table 4-1: Structure of the clock generation circuit

Item	Structure
Control register	Clock operation mode control register (CMC) System Clock Control Register (CKC) Clock operation status control register (CSC) Status register of the oscillation stabilization time counter (OSTC) Oscillation stabilization time selection register (OSTS) Peripheral enabled registers 0, 1 (PER0, PER1) Subsystem Clock Supply Mode Control Register (OSMC) High-speed internal oscillator frequency selection register (HOCODIV) High-speed internal oscillator trim register (HIOTRM) Subsystem Clock Selection Register (SUBCKSEL)
Oscillation circuit	X1 oscillation circuit XT1 oscillation circuit High-speed internal oscillator Low-speed internal oscillator

Figure4-1: Block diagram of the clock generation circuit



Remarks: FX : X1 clock oscillation frequency

F<sub>HOCO</sub> : High-speed internal oscillator clock frequency (max. 32MHz)

F<sub>IH</sub> : High-speed internal oscillator clock frequency (max. 32MHz)

F<sub>EX</sub> : External main system clock frequency

F<sub>MX</sub> : High-speed system clock frequency

F<sub>MAIN</sub> : Main system clock frequency

F<sub>XT</sub> : XT1 clock oscillation frequency

F<sub>EXS</sub> : External subsystem system clock frequency

F<sub>SUB</sub> : Subsystem clock frequency

F<sub>CLK</sub> : Clock frequency of CPU/peripheral hardware

F<sub>IL</sub> : Clock frequency of the low-speed internal oscillator

## 4.3 Registers for controlling clock generation circuit

The clock generation circuit is controlled through the following registers.

- Clock operation mode control register (CMC)
- System Clock Control Register (CKC)
- Clock operation status control register (CSC)
- Status register of the oscillation stabilization time counter (OSTC)
- Oscillation stabilization time selection register (OSTS)
- Peripheral enabled registers 0, 1 (PER0, PER1)
- Subsystem Clock Supply Mode Control Register (OSMC)
- High-speed internal oscillator frequency selection register (HOCODIV)
- High-speed internal oscillator trim register (HIOTRM)

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

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

This is the register to set the operation mode of X1/PH01, X2/EXCLK/PH02, XT1/PH03, XT2/EXCLKS/PH04 pins and to select the gain of the oscillation circuit.

After the reset is released, the CMC register can only be written 1 time by 8-bit memory manipulation instruction. It is possible to read this register by means of an 8-bit memory manipulation instruction.

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

Figure4-2: Format of the Clock Operation Mode Control Register (CMC)

Address:	40020400H		After reset: 00H					
Symbol	7	6	5	4	3	2	1	0
CMC	EXCLK	OSCSEL	EXCLKS <sup>Note</sup>	OSCSELS <sup>Note</sup>	0	AMPHS1 <sup>Note</sup>	AMPHS0 <sup>Note</sup>	0
R/W	R/W							

Bit7~ Bit6	EXCLK OSCSEL:	High-speed system clock pin	X1/PH01 pins	X2/EXCLK/PH02 pins
		operation mode		
		00= Port Mode	IO Port	
		01= X1 oscillation mode	Connect to a crystal or ceramic resonator.	
		10= Port Mode	IO Port	
Bit5~ Bit4	EXCLKS OSCSELS:	11= External clock input mode	IO Port	External clock input
		Operating mode of the subsystem clock pins	XT1/PH03 pins	XT2/EXCLK/PH04 pins
		00= Port Mode	IO Port	
		01= XT1 oscillation mode	Connect to a crystal resonator.	
		10= Port Mode	IO Port	
Bit2~ Bit1	AMPHS1 AMPHS0:	11= External clock input mode	IO Port	External clock input
		Oscillation mode selection of XT1 oscillation circuit		
		00= Low-power consumption oscillation (default)		
		01= Usual oscillation		
		10= Ultra-low power consumption oscillation		
	AMPH:	Settings are disabled.		
		Control of X1 clock oscillation frequency		
		0= 1MHz≤F <sub>x</sub> ≤10MHz		
		1= 10MHz<F <sub>x</sub> ≤20MHz		

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

Notice:

1. After the reset is released, the CMC register can only be written 1 time by an 8-bit memory manipulation instruction. When the CMC register is used with the initial value ("00H"), the CMC register must be set to "00H" after the reset is released in order to prevent malfunction when the program is out of control (if a value other than "00H" is written by mistake, it cannot be recovered).
2. The CMC register must be set after the reset is released and before the X1 or XT1 oscillation is started by setting the clock running status control register (CSC).
3. When the X1 clock oscillation frequency exceeds 10MHz, the AMPH need be set to "1".
4. The AMPH bit, AMPHS1 bit and AMPHS0 bit must be set after the reset is released and with F<sub>IH</sub> selected as the state of F<sub>CLK</sub> (before switching from F<sub>CLK</sub> to F<sub>MX</sub> or F<sub>SUB</sub>).
5. The oscillation stabilization time of F<sub>XT</sub> must be counted by software.
6. The upper frequency limit of the system clock is 32MHz, but the upper frequency limit of the X1 oscillation circuit is 20MHz.

Remarks: F<sub>X</sub> : X1 clock oscillation frequency

### 4.3.2 System clock control register (CKC)

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

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

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

Figure4-3: Format of the system clock control register (CKC)

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

- Bit7      CLS:    Status of the CPU/peripheral hardware clock ( $F_{CLK}$ )  
                  0=    Main system Clock ( $F_{MAIN}$ )  
                  1=    Sub-system Clock ( $F_{SUB}$ )
- Bit6      CSS<sup>Note2</sup>:    Selection of the CPU/peripheral hardware clock ( $F_{CLK}$ )  
                  0=    Main system Clock ( $F_{MAIN}$ )  
                  1=    Sub-system Clock ( $F_{SUB}$ )
- Bit5      MCS:    Status of the main system clock ( $F_{MAIN}$ )  
                  0=    High-speed internal oscillator clock ( $F_{IH}$ )  
                  1=    High-speed system Clock ( $F_{MX}$ )
- Bit4      MCM0<sup>Note2</sup>    Operation control of the main system clock ( $F_{MAIN}$ )  
                  0=    Select the high-speed internal oscillator clock ( $F_{IH}$ ) as the main system 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, bit0~3 must be set to "0".

Note 2: It is disabled to change the value of MCM0 bit when the CSS bit is "1".

Remark:

- Provides clock to the CPU and the CSS bit of the peripheral hardware. If the CPU clock is changed, the peripheral hardware clock is changed at the same time (except for the real-time clock, 15-bit interval timer, clock output/buzzer output, and watchdog timer). Therefore, if the CPU/peripheral hardware clock is changed, each peripheral function must be stopped.
- If the subsystem clock is used as the peripheral hardware clock, the operation of the A/D converter and IICA cannot be guaranteed. For the operating characteristics of the peripheral hardware, please refer to the electrical characteristics in the chapter of each peripheral hardware and the data sheet.
- $F_{HOCO}$  : High-speed internal oscillator clock frequency (max. 32MHz)  
 $F_{IH}$  : High-speed internal oscillator clock frequency (max. 32MHz)  
 $F_{MX}$  : High-speed system clock frequency  
 $F_{MAIN}$  : Main system clock frequency  
 $F_{SUB}$  : Subsystem clock frequency

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

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

After a reset signal is generated, the value of these registers becomes "C0H".

Figure4-4: Format of the clock operation status control register (CSC)

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

Bit7	MSTOP:	High-speed system clock operation control											
		X1 oscillation mode					External clock input mode					Input Port Mode	
		0=	X1 oscillation circuit operated					External clock valid on EXCLK pin					Input Port
		1=	X1 oscillation circuit stopped					External clock invalid on EXCLK pin					
Bit6	XTSTOP:	Operation control of subsystem clocks											
		XT1 oscillation mode					External clock input mode					Input Port Mode	
		0=	XT1 oscillation circuit operated					External clock valid on EXCLKS pin					Input Port
		1=	XT1 oscillation circuit stopped					External clock invalid on EXCLKS pin					
Bit0	HIOSTOP:	High-speed internal oscillator clock operation control											
		0=	High-speed internal oscillator operated										
		1=	High-speed internal oscillator stopped										

Notice:

1. After the reset is released, the CSC register must be set after setting the Clock Run Mode Control Register (CMC).
2. The oscillation stabilization time selection register (OSTS) must be set after the reset is released and before the MSTOP bit is set to "0". However, when using the OSTS register with its initial value, it is not necessary to set the OSTS register.
3. To start X1 oscillation by setting the MSTOP bit, the oscillation stabilization time of the X1 clock must be confirmed by the status register (OSTC) of the oscillation stabilization time counter.
4. To start XT1 oscillation by setting the XSTOP bit, you must wait for the required oscillation stabilization time of the subsystem clock by software.
5. The clock selected as the CPU/peripheral hardware clock ( $F_{CLK}$ ) cannot be stopped via the CSC register.

Note: For the register flag settings used to stop clock oscillation (external clock input is invalid) and the conditions before stopping, please refer to Table4-2.

Table4-2: Clock stopping method

Clock	Conditions before clock stop (invalid external clock input)	Flag settings of CSC register
X1 Clock	CPU/peripheral hardware clock runs at a clock other than the high-	MSTOP=1



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

#### 4.3.4 Status register of the oscillation stabilization time counter (OSTC)

This is the register that indicates the count status of the oscillation stabilization time counter of the X1 clock. It can confirm the oscillation stabilization time of X1 clock in the following cases:

- When the CPU clock is a high speed internal oscillator clock or a subsystem clock and the oscillation of the X1 clock is started
- When the CPU clock is a high speed internal 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 manipulation instruction.

The value of this register changes to "00H" by generating a reset signal, entering deep sleep mode or setting the MSTOP bit (bit7 of bit 7 of the Clock Operation Status Control Register (CSC)) to 1.

Note: The oscillation stabilization time counter starts counting when:

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

Figure4-5: Format of the status register (OSTC) of the oscillation stabilization time counter

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

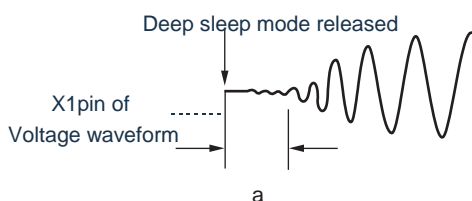
Bit7~Bit0

OSTC: Oscillation stabilization time state

		$F_X=10\text{MHz}$	$F_X=20\text{MHz}$
00000000=	Less than $2^8/F_X$	Less than 25.6us	Less than 12.8us
10000000=	At least $2^8/F_X$	At least 25.6us	At least 12.8us
11000000=	At least $2^9/F_X$	At least 51.2us	At least 25.6us
11100000=	At least $2^{10}/F_X$	At least 102us	At least 51.2us
11110000=	At least $2^{11}/F_X$	At least 204us	At least 102us
11111000=	At least $2^{13}/F_X$	At least 819us	At least 409us
11111100=	At least $2^{15}/F_X$	At least 3.27ms	At least 1.63ms
11111110=	At least $2^{17}/F_X$	At least 13.1ms	At least 6.55ms
11111111=	At least $2^{18}/F_X$	At least 26.2ms	At least 13.1ms

Notice:

- After the above time, each bit from MOST8 becomes "1" and stays "1" in turn.
- The oscillation stable time counter counts only within the oscillation stable time set by the OSTC. In the following cases, the setting value of the oscillation stability time of the OSTC 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 oscillation of the X1 clock is started
  - When the CPU clock is a high speed internal oscillator clock and the deep sleep mode is released after transferring to deep sleep mode in the X1 clock oscillation state (therefore, it must be noted that the OSTC register after released from deep sleep mode only sets the state within the oscillation stabilization time set by the OSTC register)
- The oscillation stabilization time of the X1 clock does not include the time before the clock starts oscillating (See Figure a as below).


Remarks:  $F_X$  : X1 clock oscillation frequency

### 4.3.5 Oscillation stabilization time selection register (OSTS)

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

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

If the CPU clock is switched from the high-speed internal oscillator clock or the subsystem clock to the X1 clock, or if the CPU clock is the high-speed internal oscillator clock and is released from deep sleep mode after shifting to deep sleep mode with the X1 clock oscillating, it is necessary to confirm whether the oscillation stabilization time has elapsed by the status register (OSTC) of the oscillation stabilization time counter.

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

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

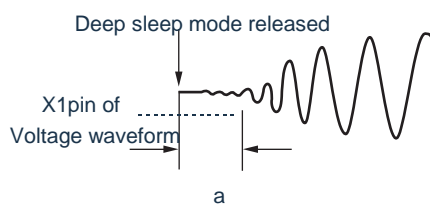
Figure4-6: Format of the oscillation stabilization time selection register (OSTS)

Address:	40020403H			After reset:	07H			
Symbol	7	6	5	4	3	2	1	0
OSTS	0					OSTS2	OSTS1	OSTS0
R/W	R/W							

Bit2~Bit0	OSTS<2:0>:	Oscillation stabilization time state	
		$F_X=10\text{MHz}$	$F_X=20\text{MHz}$
000=	$2^8/F_X$	25.6us	12.8us
001=	$2^9/F_X$	51.2us	25.6us
010=	$2^{10}/F_X$	102us	51.2us
011=	$2^{11}/F_X$	204us	102us
100=	$2^{13}/F_X$	819us	409us
101=	$2^{15}/F_X$	3.27ms	1.63ms
110=	$2^{17}/F_X$	13.1ms	6.55ms
111=	$2^{18}/F_X$	26.2ms	13.1ms

Notice:

- To change the setting of the OSTS register, you must set the MSTOP bit to "0" in the Clock Operation Status Control Register (CSC) before changing it.
- The oscillation stabilization time counter counts only during the oscillation stabilization time set by the OSTC register. In the following cases, the setting value of the oscillation stability time of the OSTS register must be greater than the count value confirmed by the OSTC register after the start of oscillation.
  - When the CPU clock is a high speed internal oscillator clock or a subsystem clock and the oscillation of the X1 clock is started
  - When the CPU clock is a high speed internal oscillator clock and the deep sleep mode is released after transferring to deep sleep mode in the X1 clock oscillation state (therefore, it must be noted that the OSTC register after released from deep sleep mode only sets the state within the oscillation stabilization time set by the OSTS register)
- The oscillation stabilization time of the X1 clock does not include the time before the clock starts oscillating (See Figure a as below).



Remarks: FX : X1 clock oscillation frequency

### 4.3.6 Peripheral enabled 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 clocks to hardware that is not in use.

When the following peripheral functions controlled by these registers are used, the corresponding bit should be set to "1" before the initial setting of the peripheral function is performed.

- Real-time clock, 15-bit interval timer
- IrDA
- AD converter
- Serial interface IICA0
- Serial Communication Unit 0/1/2
- Timer80
- D/A converter
- Comparator
- Enhanced DMA
- TimerA

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

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

Figure4-7: Format of peripheral enabled register 0 (PER0)

Address:	40020420H				After reset: 00H			
Symbol	7	6	5	4	3	2	1	0
PER0	RTCEN <sup>Note</sup>	IICAEN	IRDAEN	SCI2EN	SCI1EN	SCI0EN	TMAEN	TM80EN
R/W	R/W							

- |      |         |   |
|------|---------|---|
| Bit7 | RTCEN:  | Provides control of the input clock for the real-time clock (RTC) and 15-bit interval timer   |
|      | 0=      | Stop providing input clock.   |
|      |         | <ul style="list-style-type: none"> <li>• The SFR used by the Real Time Clock (RTC) and the 15-bit interval timer cannot be written.</li> <li>• The real-time clock (RTC) and 15-bit interval timer are in the reset state.</li> </ul> |
|      | 1=      | Provides input clock.   |
|      |         | <ul style="list-style-type: none"> <li>• The SFR used by the Real Time Clock (RTC) and the 15-bit interval timer can be read and written.</li> </ul>  |
| Bit6 | IICAEN: | Provides control of the input clock of the serial interface IICA  |
|      | 0=      | Stop providing input clock.   |
|      |         | <ul style="list-style-type: none"> <li>• The SFR used by the serial interface IICA cannot be written.</li> <li>• The serial interface IICA is in the reset state.</li> </ul>  |
|      | 1=      | Provides input clock.   |
|      |         | <ul style="list-style-type: none"> <li>• The SFR used by the serial interface IICA0 can be read and written.</li> </ul>   |
| Bit5 | IRDAEN: | Provides control of the input clock of the serial interface IRDA  |
|      | 0=      | Stop providing input clock.   |
|      |         | <ul style="list-style-type: none"> <li>• The SFR used by IRDA cannot be written.</li> <li>• IRDA is in the reset state.</li> </ul>  |
|      | 1=      | Provides input clock.   |
|      |         | <ul style="list-style-type: none"> <li>• The SFR used by IRDA can be read and written.</li> </ul>   |
| Bit4 | SCI2EN: | Provides control of the input clock of Universal Serial Communication Unit 2  |
|      | 0=      | Stop providing input clock.   |
|      |         | <ul style="list-style-type: none"> <li>• The SFR used by Universal Serial Communication Unit 2 cannot be written.</li> <li>• Universal Serial Communication Unit 2 is in the reset state.</li> </ul>                                  |
|      | 1=      | Provides input clock.   |
|      |         | <ul style="list-style-type: none"> <li>• The SFR used by Universal Serial Communication Unit 2 can be read and written.</li> </ul>  |
| Bit3 | SCI1EN: | Provides control of the input clock of Universal Serial Communication Unit 1  |
|      | 0=      | Stop providing input clock.   |
|      |         | <ul style="list-style-type: none"> <li>• The SFR used by Universal Serial Communication Unit 1 cannot be written.</li> <li>• Universal Serial Communication Unit 1 is in the reset state.</li> </ul>                                  |

		1=	Provides input clock. <ul style="list-style-type: none"><li>• The SFR used by Universal Serial Communication Unit 1 can be read and written.</li></ul>
Bit2	SCI0EN:		Provides control of the input clock of Universal Serial Communication Unit 0
		0=	Stop providing input clock. <ul style="list-style-type: none"><li>• The SFR used by Universal Serial Communication Unit 0 cannot be written.</li><li>• Universal Serial Communication Unit 0 is in the reset state.</li></ul>
		1=	Provides input clock. <ul style="list-style-type: none"><li>• The SFR used by Universal Serial Communication Unit 0 can be read and written.</li></ul>
Bit1	TMAEN:		Provides control of the input clock of the TMA module
		0=	Stop providing input clock. <ul style="list-style-type: none"><li>• The SFR used by TMA cannot be written.</li><li>• TMA is in the reset state.</li></ul>
		1=	Provides input clock. <ul style="list-style-type: none"><li>• The SFR used by TMA can be read and written.</li></ul>
Bit0	TM80EN:		Provides control of the input clock of Timer8
		0=	Stop providing input clock. <ul style="list-style-type: none"><li>• The SFR used by the general-purpose timer unit 0 cannot be written.</li><li>• General purpose timer unit 0 is in the reset state.</li></ul>
		1=	Provides input clock. <ul style="list-style-type: none"><li>• The SFR used by the general-purpose timer unit 0 can be read and written.</li></ul>

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

Figure4-8: Format of peripheral enabled register 1 (PER1)

Address: 4002081AH

After reset: 00H

Symbol	7	6	5	4	3	2	1	0
PER1	OPAEN	CMPEN	DACEN	ADCEN	0	OSDCEN	DMAEN	SPIHSEN
R/W	R/W							

- Bit7 OPAEN: Provides control of the input clock of the OPA
- 0= Stop providing input clock.
    - The SFR used by OPA cannot be written.
    - OPA is in the reset state.
  - 1= Provides input clock.
    - The SFR used by OPA can be read and written.
- Bit6 CMPEN: Provides control of the input clock of Comparator
- 0= Stop providing input clock.
    - The SFR used by Comparator cannot be written.
    - Comparator is in the reset state.
  - 1= Provides input clock.
    - The SFR used by Comparator can be read and written.
- Bit5 DACEN: Provides control of the input clock of D/A
- 0= Stop providing input clock.
    - The SFR used by D/A cannot be written.
    - D/A is in the reset state.
  - 1= Provides input clock.
    - The SFR used by D/A can be read and written.
- Bit4 ADCEN: Provides control of the input clock of the A/D
- 0= Stop providing input clock.
    - The SFR used by the A/D cannot be written.
    - A/D is in the reset state.
  - 1= Provides input clock.
    - The SFR used by A/D can be read and written.
- Bit2 OSDCEN: Provides control of the input clock of the OSDC
- 0= Stop providing input clock.
    - The SFR used by OSDC cannot be written.
    - OSDC is in the reset state.
  - 1= Provides input clock.
    - The SFR used by OSDC can be read and written.
- Bit1 DMAEN: Provides control of the input clock of the DMA
- 0= Stop providing input clock.
    - DMA does not operate.
  - 1= Provides input clock.
    - DMA can operate.
- Bit0 SPIHSEN: Provides control of the input clock of the SPIHS
- 0= Stop providing input clock.
    - The SFR used by SPIHS cannot be written.
    - SPIHS is in the reset state.
  - 1= Provides input clock.
    - The SFR used by SPIHS can be read and written.



### 4.3.7 Subsystem clock supply mode control register (OSMC)

OSMC registers are registers that reduce power consumption by stopping unneeded clock functions.

If RTCLPC bit set to "1", it stops providing clock to 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 real-time clock and the running clock of the 15-bit interval timer can be selected via the OSMC register.

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

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

Figure4-9: Format of the system clock supply mode control register (OSMC)

Address: 40020423H		After reset: 00H						
Symbol	7	6	5	4	3	2	1	0
OSMC	RTCLPC	0		WUTMMCK0	0			
R/W	R/W							

- Bit7 RTCLPC: Settings in deep sleep mode and sleep mode where the CPU runs on the subsystem clock
- 0= Enable subsystem clocks to be provided to peripheral functions  
(Refer to Table 25-1 to Table 25-3 for the peripheral functions enabled to operate)
  - 1= Stop providing a subsystem clock to peripheral functions other than the real-time clock and 15-bit interval timer.
- Bit4 WUTMMCK0: Selection of operating clock for real-time clock, 15-bit interval timer and Timer A
- 0=
    - The subsystem clock is the running clock for the real-time clock and the 15-bit interval timer.
    - The low-speed internal oscillator cannot be selected as the counting source for Timer A.
  - 1=
    - The low-speed internal oscillator clock is the running clock for the real-time clock and the 15-bit interval timer.
    - The low-speed internal oscillator or subsystem clock can be selected as the counting source for Timer A.

### 4.3.8 High-speed internal oscillator frequency selection register (HOCODIV)

This is a register that changes the high-speed internal oscillator frequency set by the option byte (000C2H).

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

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

Figure4-10: Format of the high-speed internal oscillator frequency selection register (HOCODIV)

Address: 40021C20H

After reset: Setting value of FRQSEL2~FRQSEL0 bits of option byte (000C2H)

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

Bit2~Bit0 HOCODIV<2:0>: High-speed internal oscillator clock frequency selection

FRQSEL4,3=00

000= F<sub>HOCO</sub>=32MHz

F<sub>IH</sub>=32MHz

001= F<sub>HOCO</sub>=32MHz

F<sub>IH</sub>=16MHz

010= F<sub>HOCO</sub>=32MHz

F<sub>IH</sub>=8MHz

011= F<sub>HOCO</sub>=32MHz

F<sub>IH</sub>=4MHz

100= F<sub>HOCO</sub>=32MHz

F<sub>IH</sub>=2MHz

101= F<sub>HOCO</sub>=32MHz

F<sub>IH</sub>=1MHz

Others: Settings are disabled.

Notice:

1. The HOCODIV register must be set in the state where the high-speed internal oscillator clock (FIH) is selected as the CPU/peripheral hardware clock (F<sub>CLK</sub>).
2. After changing the frequency via the HOCODIV register, frequency switching is performed after the following transfer times:
  - 1) Runs for up to 3 clocks at the frequency before the change.
  - 2) Wait for up to 3 CPU/peripheral hardware clocks at the changed frequency.

### 4.3.9 High-speed internal oscillator trim register (HIOTRM)

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

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

In case of changes in temperature and voltage at the  $V_{DD}$  pin, corrections need to be made before the required frequency accuracy or at regular intervals.

Figure4-11: Format of the high-speed internal oscillator trim register (HIOTRM)

Address:	40021C00H		After reset:	Note				
Symbol	7	6	5	4	3	2	1	0
HIOTRM	0		HIOTRM5	HIOTRM4	HIOTRM3	HIOTRM2	HIOTRM1	HIOTRM0
R/W	R/W							

Bit5~Bit0    HIOTRM<5:0>:    High-speed internal oscillator

000000=    minimum speed

000001=    ↑

000010=    |

000011=    |

000100=    |

      •    |

      •    |

      •    |

111110=    ↓

111111=    maximum speed

Note: The reset value is the adjustment value at shipment.

Remark: Every 1 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 the register for selecting the subsystem clock  $F_{SUB}$  and the low-speed internal oscillator clock  $F_{IL}$  and for selecting the low-speed internal oscillator clock frequency.

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

After a reset signal is generated, the value of these registers becomes "02H".

Figure4-12: Format of the subsystem Clock Selection Register (SUBCKSEL)

Address:	40020407H	After reset:	02H					
Symbol	7	6	5	4	3	2	1	0
SUBCKSEL	0						LOCOSSEL	SELLOSC
R/W	R/W							

- Bit1      LOCOSSEL: Low-speed internal oscillator clock frequency selection
- 0=      • The low-speed internal oscillator clock frequency is 15K.
  - 1=      • The low-speed internal oscillator clock frequency is 30K (Default).
- Bit0      SELLOSC: Selection of subsystem clock and low-speed internal oscillator clock
- 0=      • Select the secondary system clock (Default).
  - 1=      • Selects the low-speed internal oscillator clock.

## 4.4 System clock oscillation circuit

### 4.4.1 X1 oscillation circuit

The X1 oscillator circuit is oscillated by a crystal resonator or ceramic resonator (1~20MHz) connected to pins X1 and X2. An external clock can also be input, where a clock signal must be input to the EXCLK pin.

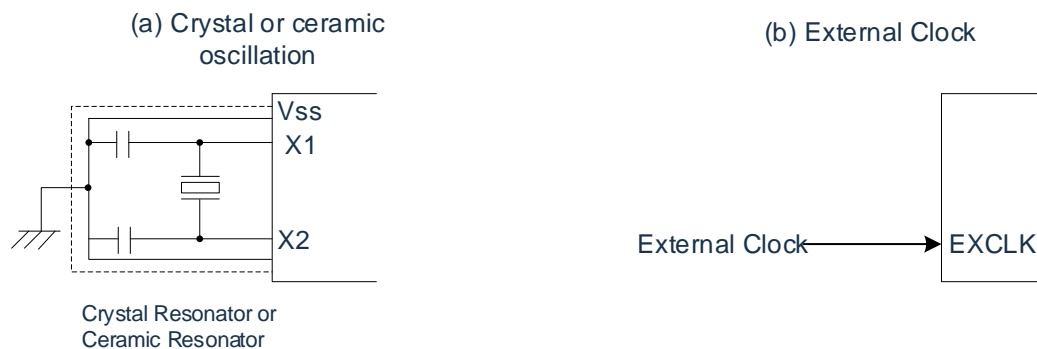
When using the X1 oscillation circuit, bit 7 and bit 6 (EXCLK, OSCSEL) of the clock operation mode control register (CMC) must be set as follows:

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

When the X1 oscillation circuit is not used, the input port mode must be set (EXCLK, OSCSEL=0, 0). Also, refer to "2.4 Handling of unused ports" when it is not used as an input port either.

An example of an external circuit for the X1 oscillation circuit is shown in Figure4-13.

Figure4-13: Example of an external circuit for X1 oscillation circuit



Notices are shown on the following page.

## 4.4.2 XT1 oscillation circuit

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

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

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

When the XT1 oscillation circuit is not used, the input port mode must be set (EXCLK, OSCSEL=0, 0). Also, when not used as an input port either, refer to "2.4 Handling of unused ports". An example of an external circuit for the XT1 oscillation circuit is shown in Figure4-14.

Figure4-14: Example of an external circuit for XT1 oscillation circuit

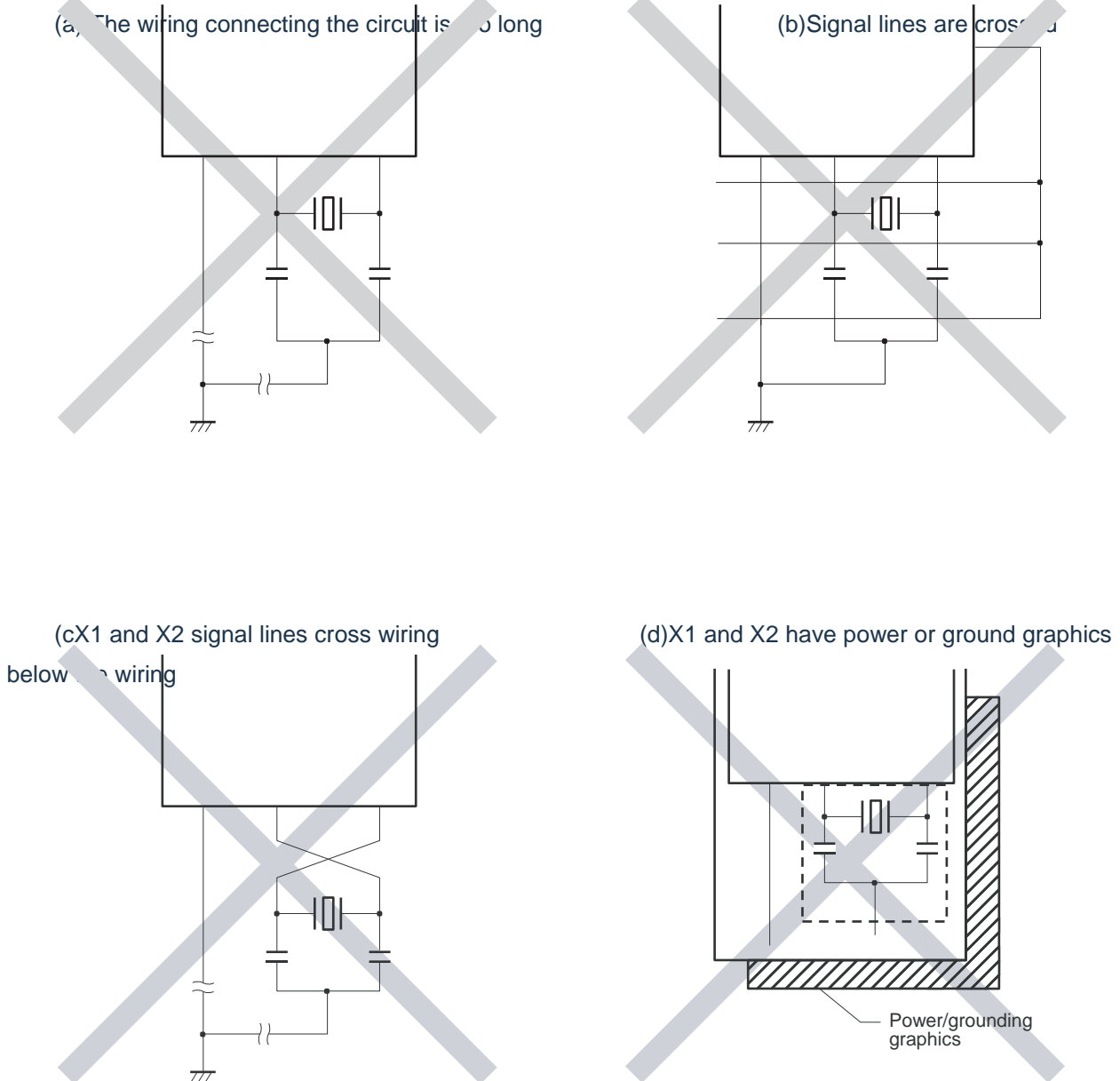


Note: When using the X1 oscillation circuit and XT1 oscillation circuit, in order to avoid the effects of wiring capacitors, etc., the dashed portions of Figures 4-13 and 4-14 must be wired by.

- 1) The wiring must be kept as short as possible.
- 2) It must not cross with other signal lines and must not be close to wiring with varying high currents flowing through it.
- 3) The capacitor ground point of the oscillating circuit must always be kept at the same potential as  $V_{SS}$  and must not be grounded to a ground pattern through which a large current flows.
- 4) The signal cannot be removed from the oscillation circuit.

An incorrect example of a resonator connection is shown in Figure.

Figure4-15: Example of incorrect resonator connection (1/2)

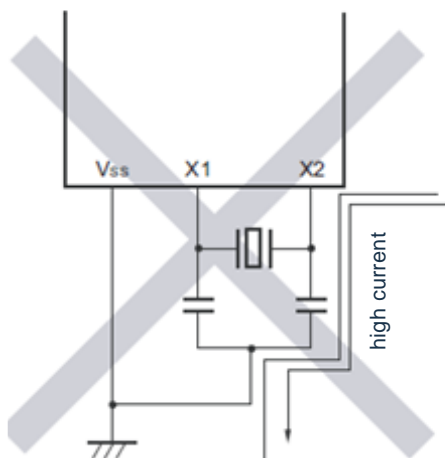


**Notice:**

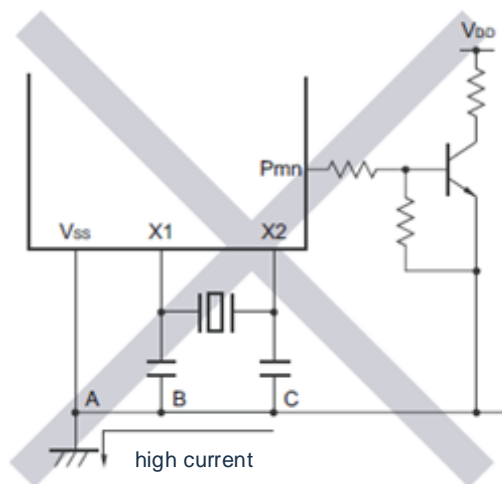
1. In multilayer boards or double-sided boards, power or ground graphics should not be configured below the wiring area (dashed portion of the diagram) for pins X1, X2 and the resonator. The wiring must not create a capacitive component that would affect the oscillation characteristics.
2. In the case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

Figure4-15: Example of incorrect resonator connection (2/2)

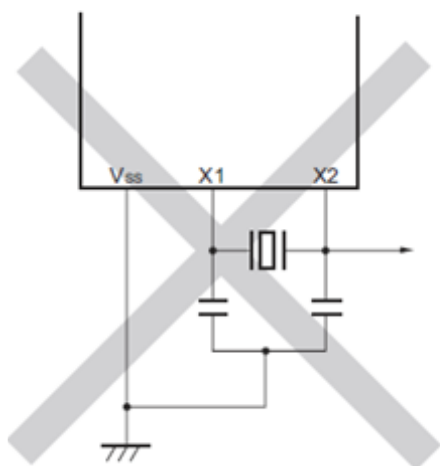
(e) varying high current source close to singal lines



(f) Current flows along grounding of oscilation circuit  
(Point A, B, C has difference in electric potential)



(g) extracted signal



Notice:

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



### 4.4.3 High-speed internal oscillator

The CMS32H6157 has a built-in high-speed internal oscillator. The frequency can be selected from 32MHz, 16MHz, 12MHz, 8MHz, 6MHz, 4MHz, 3MHz, 2MHz and 1MHz by the option byte (000C2H). The oscillation can be controlled by bit0 (HISTOP) of the clock operation status control register (CSC).

The high-speed internal oscillator starts oscillating automatically after the reset is released.

### 4.4.4 Low-speed internal oscillator

The CMS32H6157 has a built-in low-speed internal oscillator.

The low-speed internal oscillator clock is used as the watchdog timer, the real-time clock, the clock for the 15-bit interval timer, and the clock for Timer A, as well as the external reference clock for the SysTick timer, and 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", the low-speed internal oscillator oscillates.

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

## 4.5 Operation of clock generation circuit

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

$F_{MAIN}$  : Main system clock

$F_{MX}$  : High-speed system clock

$F_X$ : X1 Clock

$F_{EX}$  : External main system clock

$F_{IH}$ : High-speed internal oscillator clock

$F_{SUB}$  : Subsystem clock

$F_{XT}$ : XT1 Clock

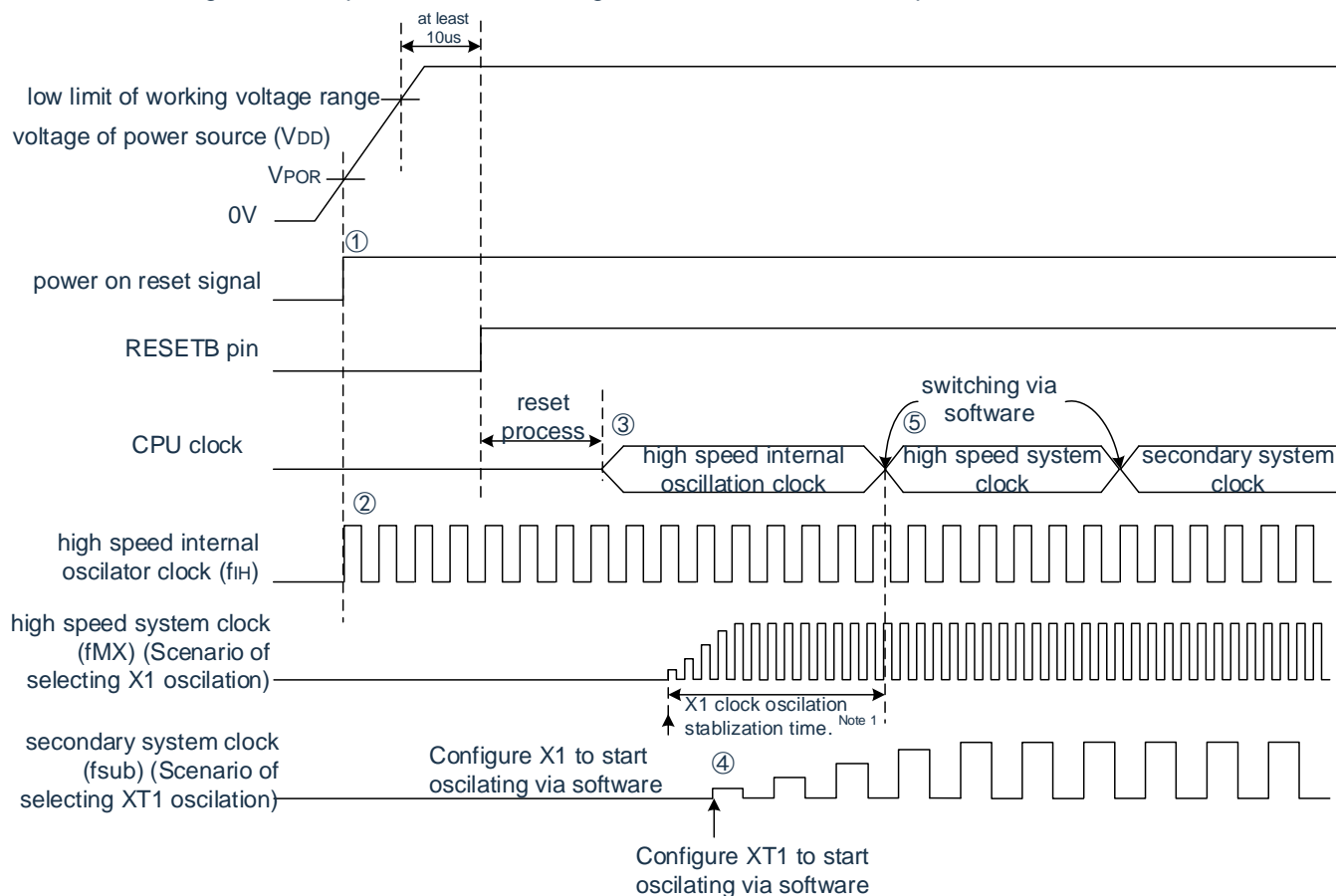
$F_{EXS}$  : External subsystem system clock

$F_{IL}$ : High-speed internal oscillator clock

$F_{CLK}$ : CPU/peripheral hardware clock

After the CMS32H6157 is released from reset, the CPU starts to operate through the output of the high-speed internal oscillator. The operation of the clock generation circuit when power is turned on is shown in Figure.

Figure4-16: Operation of the clock generation circuit when the power is turned on



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

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

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

## 4.6 Clock control

### 4.6.1 Example of setting up a high-speed internal oscillator

After the reset is released, the CPU/peripheral hardware clock ( $F_{CLK}$ ) must run as the high-speed internal oscillator clock. The frequency of the high-speed internal oscillator can be selected from 32MHz, 16MHz, 8MHz, 4MHz, 2MHz, and 1MHz by using the FRQSEL0 to FRQSEL4 bits of the option byte (000C2H). In addition, the frequency can be changed by the frequency selection register (HOCODIV) of the high-speed internal oscillator.

#### 【Option byte setting】

Address: 000C2H

Symbol	7	6	5	4	3	2	1	0
Byte (000C2H)	1			FRQS EL4 0	FRQS EL3 0	FRQS EL2 0/1	FRQS EL1 0/1	FRQS EL0 0/1

Bit4~Bit0 FRQSEL<4:0>: High-speed internal oscillator frequency

	$F_{HOCO}$	$F_{IH}$
00000=	32MHz	32MHz
00001=	32MHz	16MHz
00010=	32MHz	8MHz
00011=	32MHz	4MHz
00100=	32MHz	2MHz
00101=	32MHz	1MHz
Others:	Settings are disabled.	

### 【Setting of high-speed internal oscillator frequency selection register (HOCODIV)】

Address: 0x40021C20

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

Bit2~Bit0 HOCODIV&lt;2:0&gt;: High-speed internal oscillator frequency

000=  $F_{HOCO}=32\text{MHz}$ 
 $F_{IH}=32\text{MHz}$ 

001=  $F_{HOCO}=32\text{MHz}$ 
 $F_{IH}=16\text{MHz}$ 

010=  $F_{HOCO}=32\text{MHz}$ 
 $F_{IH}=8\text{MHz}$ 

011=  $F_{HOCO}=32\text{MHz}$ 
 $F_{IH}=4\text{MHz}$ 

100=  $F_{HOCO}=32\text{MHz}$ 
 $F_{IH}=2\text{MHz}$ 

101=  $F_{HOCO}=32\text{MHz}$ 
 $F_{IH}=1\text{MHz}$ 

Others: Settings are disabled.

## 4.6.2 Example of setting X1 oscillation circuit

After the reset is released, the CPU/peripheral hardware clock ( $F_{CLK}$ ) must run as the high-speed internal oscillator clock. Thereafter, if the X1 oscillation clock is changed, the setting of the oscillation circuit and the control of the start of oscillation are carried out via the oscillation stabilization time selection register (OSTS), the clock operation mode control register (CMC) and the clock operation status control register (CSC), and the oscillation stability is waited for via the status register of the oscillation stabilization time counter (OSTC). The X1 oscillation clock is set to  $F_{CLK}$  via the system clock control register (CKC) after waiting for the oscillation to stabilize.

【Register settings】 The registers must be set in the order ① to ⑤.

- ① Set OSCSEL bit to "1" in the CMC register and AMPH bit to "1" to operate the X1 oscillation circuit when the  $F_X$  is greater than or equal to 10MHz.

Symbol	7	6	5	4	3	2	1	0
CMC	EXCLK0	OSCSEL1	EXCLKS0	OSCS ELS0	0	AMPHS10	AMPHS00	AMPH 0/1

- ② The oscillation stabilization time of the X1 oscillation circuit during deep sleep mode is released by OSTS register selection.

(Example) To wait at least 102us by the 10MHz resonator, the value must be set to the following.

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

- ③ Clear the MSTOP bit of the CSC register to "0" to start the X1 oscillation circuit.

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

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

(Example) To wait at least 102us by the 10MHz resonator, the bits must be changed to the following value.

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

- ⑤ The X1 oscillation clock is set to the CPU/peripheral hardware clock via the MCM0 bit of the CKC register

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

### 4.6.3 Example of setting XT1 oscillation circuit

After the reset is released, the CPU/peripheral hardware clock ( $F_{CLK}$ ) must run as the high-speed internal oscillator clock. Thereafter, if changed to the XT1 oscillation clock, the oscillation circuit is set and the oscillation start is controlled by the Subsystem Clock Supply Mode Control Register (OSMC), Clock Operation Mode Control Register (CMC) and Clock Operation Status Control Register (CSC), and the XT1 oscillation clock is set to  $F_{CLK}$  by the System Clock Control Register (CKC).

【Register settings】 The registers must be set in the order ① to ⑤.

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

Symbol	7	6	5	4	3	2	1	0
OSMC	RTCLPC 0/1	0		WUTM MCK00	0			

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

Symbol	7	6	5	4	3	2	1	0
CMC	EXCLK0	OSCSEL0	EXCLKS0	OSCS ELS1	0	AMPHS1 0/1	AMPHS0 0/1	AMPH0

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

- ③ Clear the XTSTOP bit of the CSC register to "0" to start the XT1 oscillation circuit.

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

- ④ It is necessary to wait for the required oscillation stabilization time of the subsystem clock by software and timer functions, etc.

- ⑤ The XT1 oscillation clock is set to the CPU/peripheral hardware clock via the CSS bit of the CKC register.

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





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

Table 4-3: Examples of CPU clock transfer and SFR register set-up (1/5)

- (1) After the reset (A) is released, the CPU is transferred to the high speed internal oscillator clock to operate (B).

State transition	Settings for the SFR register
(A)→(B)	The SFR register (initial state after reset released) does not need to be set.

- (2) After the reset (A) is released, the CPU is transferred to the high speed system clock to operate (C).  
(The CPU operates (B) with a high speed internal oscillator clock immediately after the reset is released)

(SFR register setting order) →

SFR register setting flag State transition	CMC register <sup>Note1</sup>			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	x	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.

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

The expected oscillation stabilization time of the state register (OSTC) of the oscillation stabilization time counter ≤ the oscillation stabilization time set by the OSTS register

Notice: The clock must be set after the supply voltage reaches the set clock operable voltage (refer to the datasheet).

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

(SFR register setting order) →

SFR register setting flag State transition	CMC register <sup>Note</sup>				CSC Register	Oscillation stabilization waiting	CKC Register
	EXCLKS	OSCSEL	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	x	x	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.

Remark:

- x: Ignore
- Table 4-3 of (A) ~ (I) corresponds to Figure of (A)~(I).

Table 4-3: Examples of CPU clock transfer and SFR register set-up (2/5)

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

(SFR register setting order)

SFR register setting flag State transition	CMC register <sup>Note1</sup>			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	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 after the reset is released.

Not required if already set.

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

The expected oscillation stabilization time of the state register (OSTC) of the oscillation stabilization time counter ≤ the oscillation stabilization time set by the OSTS register

Notice: The clock must be set after the supply voltage reaches the set clock operable voltage (refer to the datasheet).

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

(SFR register setting order)

SFR register setting flag State transition	CMC register <sup>Note</sup>			CSC Register	Oscillation stabilization waiting	CKC Register
	EXCLKS	OSCSELS	AMPHS1, 0	XTSTOP		CSS
(B)→(D) (XT1 Clock)	0	1	00: Low power consumption oscillation 01: Usual	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.

Remark:

1. ×: Ignore
2. Table 4-3 of (A) ~ (I) corresponds to Figure of (A) ~ (I).

Table 4-3: Examples of CPU clock transfer and SFR register set-up (3/5)

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

(SFR register setting order) →

SFR register setting flag State transition	CSC Register	Oscillation accuracy stabilization waiting	CKC Register
	HIOSTOP		MCM0
(C)→(B)	0	Note	0

Not required for high-speed internal oscillator clock

operation.

Note: When FRQSEL4=0: 45us~65us

When FRQSEL4=0: 45us~135us

Note: The oscillation accuracy of the high-speed internal oscillator clock varies steadily depending on temperature conditions and during deep sleep mode.

- (7) The CPU moves from high-speed system clock operation (C) to subsystem clock operation (D).

(SFR register setting order) →

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

Not required for subsystem clock operation.

- (8) The CPU moves from subsystem clock operation (D) to high-speed internal oscillator clock operation (B).

(SFR register setting order) →

SFR register setting flag State transition	CSC Register	Oscillation accuracy stabilization waiting	CKC Register
	HIOSTOP		CSS
(D)→(B)	0	Note	0

Not required for high-speed internal oscillator clock operation.

Note 1: When FRQSEL4=0: 45us~65us

Note 2: When FRQSEL4=1: 45us~135us

Remark:

- Table 4-3 of (A) ~ (I) corresponds to Figure of (A)~(I).
- Note: The oscillation accuracy of the high-speed internal oscillator clock varies steadily depending on temperature conditions and during deep sleep mode.

Table 4-3: Examples of CPU clock transfer and SFR register set-up (4/5)

(9) The CPU moves from subsystem clock operation (D) to high-speed system clock operation (C).

(SFR register setting order)

State transition SFR register setting flag	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

Not required for high-speed system clock operation.

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

The expected oscillation stabilization time of the state register (OSTC) of the oscillation stabilization time counter ≤ the oscillation stabilization time set by the OSTS register

Notice: The clock must be set after the supply voltage reaches the set clock operable voltage (refer to the datasheet).

(10) The CPU moves from high-speed internal 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: Table 4-3 of (A) ~ (I) corresponds to Figure of (A)~(I).

Table 4-3: Examples of CPU clock transfer and SFR register set-up (5/5)

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

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

(Setting order) —————→

State transition		Setting contents:		
(B)→(H)		Stop	-	SCR register bit2 (SLEEPDEEP) is set to 1 and the WFI instruction is executed.
(C)→(I)	X1 oscillation	Peripheral functions that cannot be run in deep sleep mode.	OSTS register setting	
	(External clock)		-	

Remark: Table 4-3 of (A) ~ (I) corresponds to Figure of (A)~(I).

## 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.

Table4-4: Transfer of CPU clocks (1/2)

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

## 4.6.6 Time required to switch CPU clock and main system clock

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

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

Remark:

- The number of clocks in Table 4-8 and
- Table 4-7 is the number of CPU clocks before the switch.
- The number of clocks in Table 4-9 and
- Table 4-7 is the number of clocks rounded to the decimal portion.

Example: case where the main system clock is switched from the high-speed system clock to the high-speed internal oscillator clock ( $F_{IH}=8\text{MHz}$ ,  $F_{MX}=10\text{MHz}$  oscillation is selected)

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



### 4.6.7 Conditions before clock oscillation stops

The register flag settings for stopping clock oscillations (invalid external clock input) and the conditions before stopping are as follows.

Table 4-10: Conditions and flag settings before clock oscillation stops

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

## 4.7 High-speed internal oscillation correction

### 4.7.1 High-speed internal oscillation self-adjustment function

This function measures the frequency of the high-speed internal oscillator using the subsystem clock  $F_{SX}$  (32.768KHz) or the main system clock  $F_{MX}$  (8M) as the reference, and corrects the frequency accuracy of the high-speed internal oscillator  $F_{HOCO}$  in real time.

Table4-11 is the operating specifications for the high-speed internal oscillation frequency correction function. Figure4-18 is the action block diagram of the high-speed internal oscillation frequency correction function.

Table4-11: Operating specifications for the high-speed internal oscillation frequency correction function

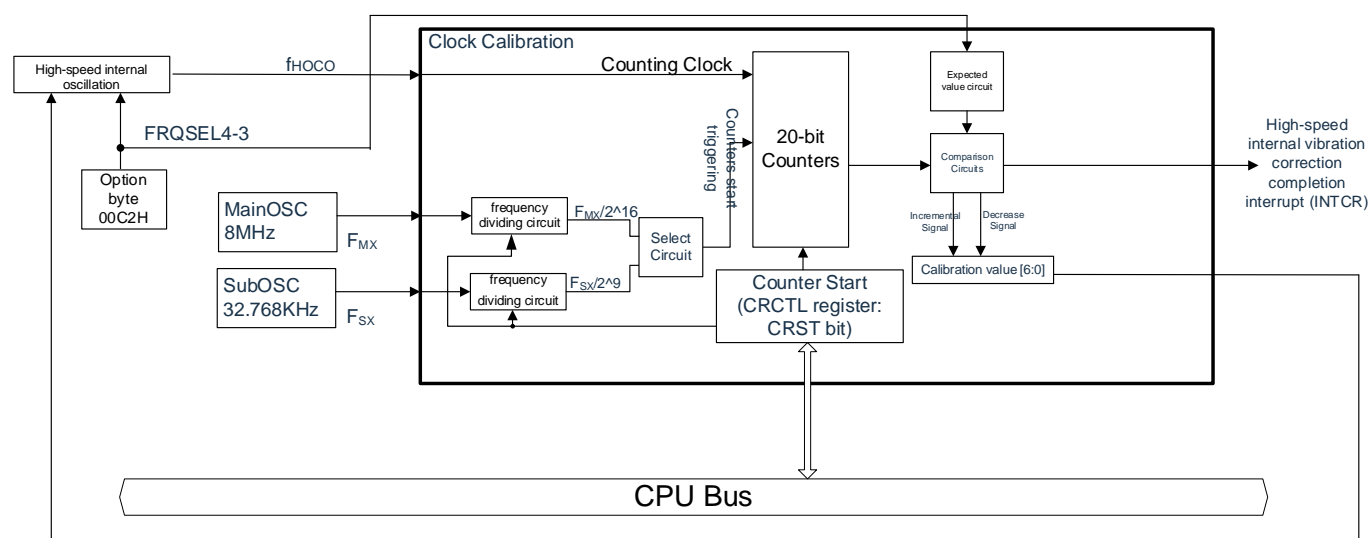
Item	Content
Reference Clock	• $F_{SX}/2^9$ (subsystem clock 32.768KHz) or $F_{MX}/2^{16}$ (main system clock 8MHz)
Corrections on object clocks	• $F_{HOCO}$ (High-speed internal oscillation)
Action Mode	• Continuous action mode Continuous high-speed internal oscillation frequency correction mode • Interval action mode A mode of high speed internal oscillation frequency correction at intervals using timer clock terminals, etc.
Clock accuracy adjustment function	• Calibration time: The reference clock is $F_{SX}/2^9$ : calibration period (31.25ms) $\times$ (calibration count - 0.5) <sup>Note</sup> The reference clock is $F_{MX}/2^{16}$ : calibration period (16.384ms) $\times$ (calibration count - 0.5) <sup>Note</sup>
Interrupt	• Interrupt generated when high speed internal oscillation frequency correction is completed (when interrupt permit is open)

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

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

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

Figure4-18: Operating specifications for the high-speed internal oscillation frequency correction function



## 4.7.2 Register description

Table 4-12 is the list of registers used for the high-speed internal oscillation frequency correction function.

Table 4-12: High-speed internal oscillation frequency correction function register

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

### 4.7.2.1 High-speed internal oscillation frequency correction control register (HOCOFC)

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

The HOCOFC register is set by a 32-bit memory manipulation instruction.

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

Figure4-19: Format of the High Speed Internal Oscillation Frequency Correction Control Register (HOCOFC)

Address: 0x40022400

After reset: 00H

R/W

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

FCMD <sup>Note1</sup>	High-speed internal oscillation correction action mode
0	Continuous action mode
1	Interval action mode

FCREF	High-speed internal oscillator frequency correction reference clock selection
0	High-speed internal oscillator frequency correction selects the subsystem clock $F_{SX}$ as the
1	High-speed internal oscillator frequency correction selects the main system clock $F_{MX}$ as the

FCEND <sup>Note2</sup>	High-speed internal oscillation frequency correction completion flag
0	High-speed internal oscillation frequency correction is not completed
1	High-speed internal oscillation frequency correction is completed

FCST <sup>Note 3</sup>	High-speed internal oscillation frequency correction circuit action control/status
0	High-speed internal oscillation frequency correction circuit operation stop/in stop progress
1	High-speed internal oscillation frequency correction circuit operation start/in operation
In continuous action mode, the software writes 0 to stop the action.	
In interval action mode, the hardware clears the FCST bit when the correction is complete.	

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

Note 2: FCEND represents the flag bit of high speed internal oscillation frequency correction completion, which is automatically set to 1 when high speed internal oscillation frequency correction is completed and cleared by writing 0 to the register.

Note 3: When writing 1 to FCST bit, confirm the current FCST bit value is 0 before writing 1 to it at first. Due to the hardware clear priority, when writing a 1 to the FCST bit immediately after the completion of the interval action (when the high-speed internal oscillator frequency correction completion interrupt is generated), the operation should be executed at least 1 cycle after the high-speed internal oscillator

frequency correction completion interrupt is generated by  $F_{HOCO}$ . After writing 0 to FCST bit (high-speed internal oscillation frequency correction circuit operation is stopped), writing 1 to FCST bit (high-speed internal oscillation frequency correction circuit operation is started) is disabled within 2 cycles of  $F_{HOCO}$ .

Notice: Bit 6, 3~1 must be written 0.

## 4.7.3 Action description

### 4.7.3.1 Action summary

The high speed internal oscillator frequency correction function generates a correction cycle based on the subsystem clock ( $F_{SX}$ ) or the main system clock ( $F_{MX}$ ), measures the frequency of the high speed internal oscillator, and corrects the frequency accuracy of the high speed internal oscillator in real time. The clock adjustment repeats the operations of the frequency measurement stage and the frequency correction stage. The calibration algorithm is performed during the frequency measurement phase, and the correction value reflecting the calibration algorithm result is saved during the frequency calibration.

Table 4-13 is the high speed internal oscillator input frequency and calibration cycle. Figure 4-20 is the action block diagram of the high-speed internal oscillation frequency correction function.

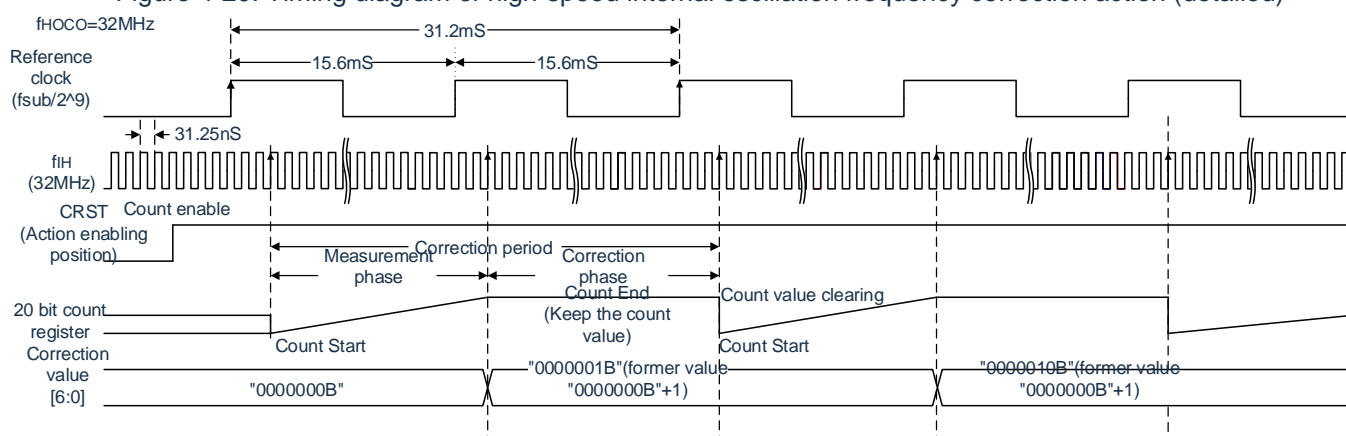
Table 4-13: High-speed internal oscillator input frequency and calibration cycle

$F_{HOCO}$ (MHz)	Correction cycle (ms)
32	$F_{SX}$ as a benchmark: 31.25 $F_{MX}$ as a benchmark: 16.384 (Frequency measurement stage + frequency correction stage)

During the frequency measurement of the calibration cycle, the frequency of the high speed internal oscillation is calibrated according to the result of the magnitude of the count value and the expected value, using the high speed internal oscillation count.

The subsequent action descriptions in this section use the subsystem clock ( $F_{SX}$ ) as the reference clock. To use  $F_{MX}$  as the reference as the base clock, simply set bit 5 (FCREF) of the HOCOFC register to 1.

Figure 4-20: Timing diagram of high-speed internal oscillation frequency correction action (detailed)



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

### (1) Continuous action mode

In continuous action mode, the high speed internal oscillator clock frequency correction action is always performed. The FCMD bit of the HOCOFC register is set to 0, which means continuous action mode.

When the FCST bit in the HOCOFC register is set to 1, the high-speed internal oscillator clock frequency correction action begins. Similarly, the high speed internal oscillator clock frequency correction action stops when the FCST bit is set to 0.

After the high speed internal oscillator clock frequency correction action, the frequency counter starts counting at the rising edge of the reference clock ( $F_{SUB}/2^9$ ) and stops counting at the rising edge of the next reference clock ( $F_{SUB}/2^9$ ). (Frequency measurement stage)

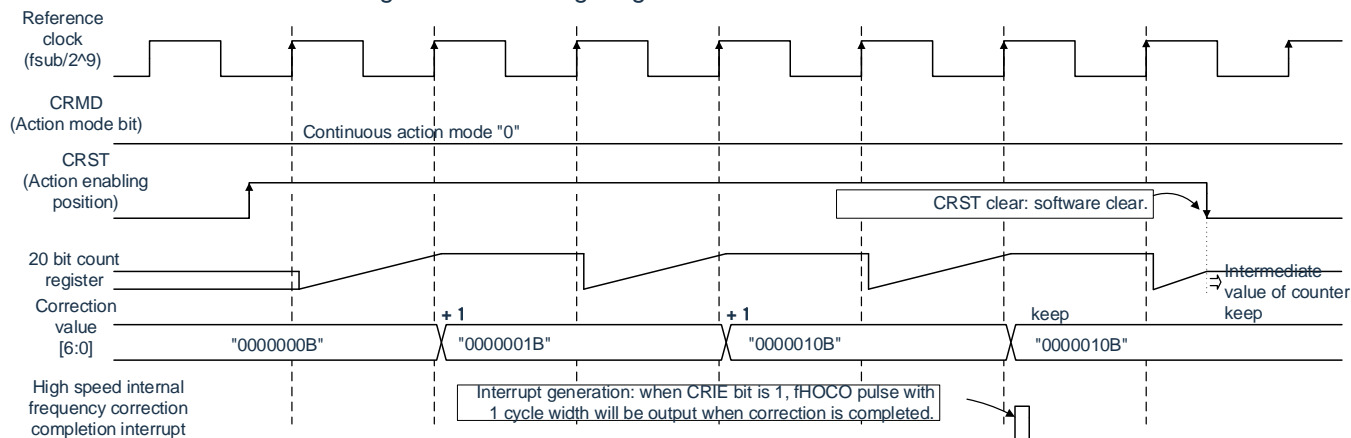
Then, the count value is compared with the expected value, and the correction value is adjusted according to the following description. (Frequency correction stage)

- When the count value is larger than the expected value: correction value -1
- When the count value is less than the expected value: correction value +1
- When the count value is within the expected value: hold the correction value (high-speed internal oscillator clock frequency correction ends)

If the FCIE bit of the HOCOFC register is set to 1, the high-speed internal oscillator clock frequency correction completion interrupt is generated after the high-speed internal oscillator clock frequency correction is completed. In continuous operation mode, the high-speed internal oscillator clock frequency correction function repeats the frequency measurement and the frequency correction until the high-speed internal oscillator clock frequency correction function is stopped.

Figure 4-21 is the timing diagram of the continuous action mode.

Figure 4-21: Timing diagram of continuous action mode



## (2) Interval action mode

In interval action mode, high-speed internal oscillator clock frequency correction is performed intermittently using timer interrupts, etc. The FCMD bit of the HOCOFC register is set to 1, which means that the interval action mode is set.

When the FCST bit in the HOCOFC register is set to 1, the high-speed internal oscillator clock frequency correction action begins.

After the high speed internal oscillator clock frequency correction action, the frequency counter starts counting at the rising edge of the reference clock ( $F_{SUB}/2^9$ ) and stops counting at the rising edge of the next reference clock ( $F_{SUB}/2^9$ ). (Frequency measurement stage)

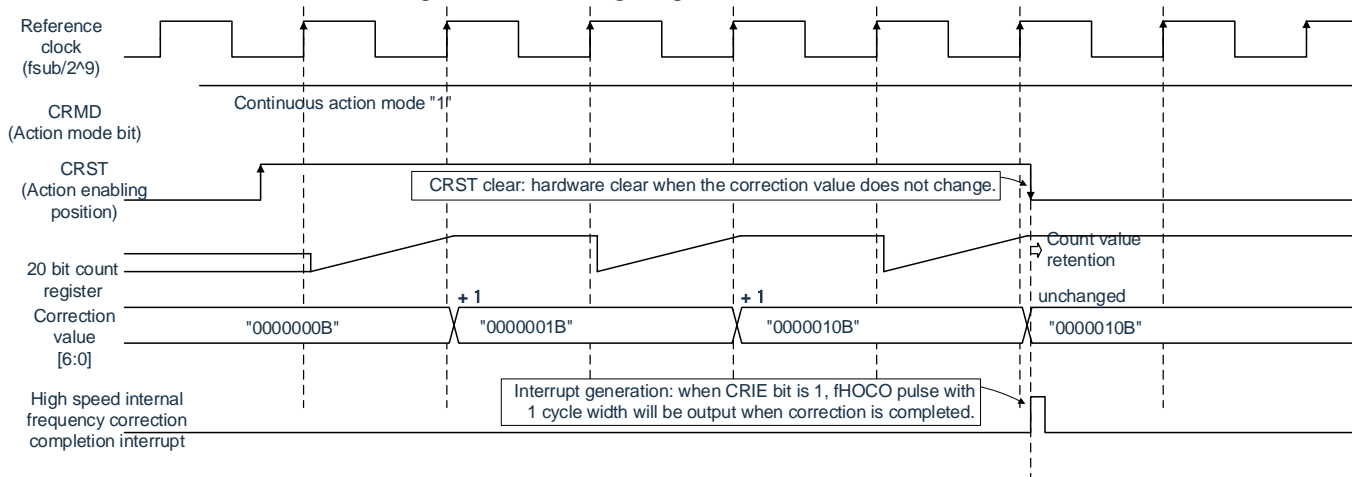
Then, the count value is compared with the expected value, and the correction value is adjusted according to the following description. (Frequency correction stage)

- When the count value is larger than the expected value: correction value -1
- When the count value is less than the expected value: correction value +1
- When the count value is within the expected value: hold the correction value (high-speed internal oscillator clock frequency correction ends)

If the FCIE bit of the HOCOFC register is set to 1, the high-speed internal oscillator clock frequency correction completion interrupt is generated after the high-speed internal oscillator clock frequency correction is completed. In the interval operation mode, the high speed internal oscillator clock frequency correction function repeats the frequency measurement and the frequency correction phase, and stops the high speed internal oscillator clock frequency correction function when the high speed internal oscillator clock frequency correction is completed.

Figure 4-22 is the timing diagram of the continuous action mode.

Figure 4-22: Timing diagram of interval action mode

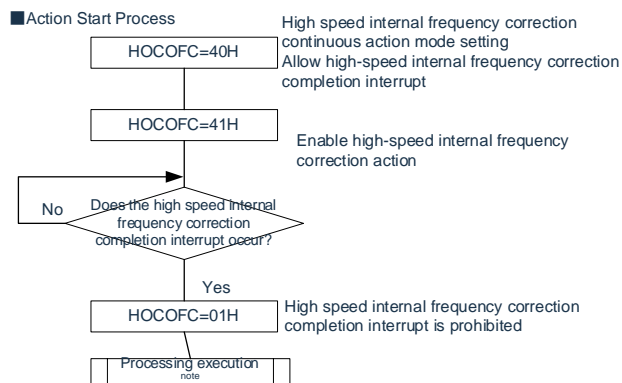


### 4.7.3.2 Action setting flow

The action start/stop flow when using the high-speed internal oscillator clock frequency correction function is shown in the figure below

Figure 4-23: Action mode setting flow (example)

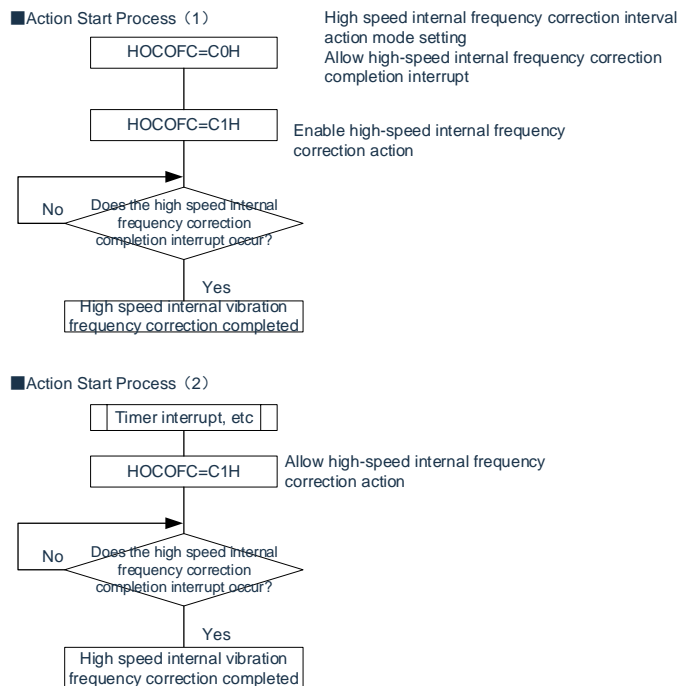
<Continuous action mode>



■ Action stop process



<Interval action mode>



Note: The high-speed internal oscillator clock frequency correction action is performed repeatedly before the high-speed internal oscillator clock frequency correction function is stopped.



## 4.7.4 Notes on use

### 4.7.4.1 SFR Access

Regarding the control of the FCST bit during interval action mode, when writing a 1 to the FCST bit, you must first confirm that the current FCST bit is 0 before writing a 1 to it. Due to the hardware clear priority, when writing a 1 to the FCST bit immediately after the completion of the interval action (when the high-speed internal oscillator frequency correction completion interrupt is generated), the operation should be executed at least 1 cycle after the high-speed internal oscillator frequency correction completion interrupt is generated by  $F_{HOCO}$ .

### 4.7.4.2 Actions on resetting

Before entering deep sleep, the high-speed internal oscillator clock frequency correction function must be stopped.

## 4.8 Oscillation stop detection circuit

The oscillation stop detection function uses the internal low-speed oscillation clock ( $F_{IL}$ ) to monitor the action state of the main system clock ( $F_{MX}$ ) or the sub-system clock ( $F_{SX}$ ), and when the action is detected to stop within a period of time, the X1 oscillation circuit or XT1 oscillation circuit is judged to be abnormal, and the oscillation stop detection signal is output, which can be used as an interrupt signal or a reset signal.

The oscillation stop detection circuit, which needs to be enabled by software setting after the reset is released.

The oscillation stop detection circuit, set by software to stop the detection action. Alternatively, the oscillation detection action is stopped because a terminal reset or other internal reset has occurred. After the reset has occurred, the software setting is needed again to enable the oscillation stop detection action.

The time for the oscillation stop detection circuit to judge the oscillation stop (oscillation stop determination time) is set by the OSDCCMP11 to OSDCCMP0 of the oscillation stop detection control register (OSDC).

Stop time = Internal low speed oscillation clock ( $F_{IL}$ ) cycle  $\times ((\text{OSDCCMP11} \sim \text{OSDCCMP0} \text{ setting value}) + 1)$

Take the internal low-speed oscillation clock ( $F_{IL}$ ) frequency of 15K as an example:

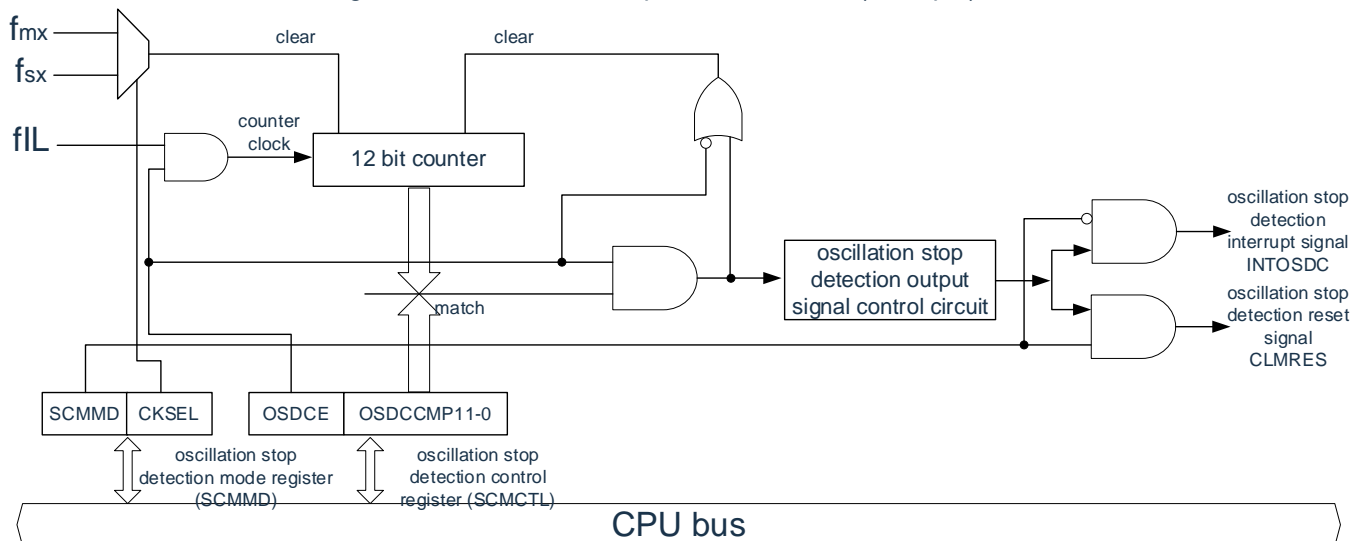
When OSDCCMP11~OSDCCMP0=003H: 232us(MIN.),267us(TYP.),314us(MAX.)

When OSDCCMP11~OSDCCMP0=FFFH: 237ms(MIN.),273ms(TYP.),322ms(MAX.)

### 4.8.1 Composition of the oscillation-stop detection circuit

The oscillation stop detection circuit consists of the following block diagram.

Figure 4-24: Oscillation stop detection circuit (example)



## 4.8.2 The registers used by the oscillation-stop detection circuit

### 4.8.2.1 Peripheral enabled Register 1(PER1)

When using the oscillation-stop detection circuit, the bit4 (OSDCEN) of PER1 must be set to 1.

Registers are described in "4.3.6 Peripheral enabled registers 0, 1 (PER0, PER1)".

### 4.8.2.2 Oscillation stop detection control register (SCMCTL)

The oscillation stop detection control register (SCMCTL) is a register that controls the start and stop of the oscillation stop detection circuit, as well as setting the oscillation stop determination time.

When the OSCDE bit is 0, the oscillation stop detection circuit does not start to operate.

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

Figure4-25: Format of the Oscillation Stop Detection Control Register (SCMCTL)

Address: 0x40022200 After reset: 0FFFH R/W

Symbol	15	14	13	12	11	10	9	8
SCMCTL	OSCDE	0	0	0	OSDCCMP11-8			
Symbol	7	6	5	4	3	2	1	0
SCMCTL	OSDCCMP7-0							

OSDCE	Action of the oscillation stop detection
0	Oscillation stop and detection action stop
1	Oscillation stop and detection action start

OSDCCMP11-0	Oscillation stop determining time
000H ... 002H	Settings are disabled.
003H ... FFFH	Set oscillation stop determining time. Oscillation stop determining time = internal low-speed oscillation clock ( $f_{IL}$ ) cycle $\times$ ((OSDCCMP11~ OSDCCMP0 setting value) +1).

Notice:

1. When modifying the setting value of OSDCCMP11~OSDCCMP0, OSDCE must be set to 0.
2. The oscillation stop detection circuit stops the oscillation detection action by setting OSDCE=0 (oscillation stop detection action stops) or by generating a terminal reset and other internal resets by software.
3. Bit14-12 must be set to 0.

### 4.8.2.3 Oscillation stop detection mode register (SCMMD)

The Oscillation Stop Detection Mode Register (SCMMD) is a register that selects the object of oscillation stop detection as the main system clock ( $F_{MX}$ ) or the subsystem clock ( $F_{SX}$ ), and whether the action after the oscillation stop is detected is to generate a reset or an interrupt.

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

Figure4-26: Format of the Oscillation Stop Detection Mode Register (SCMMD)

Address: 0x40022202 After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8
SCMMD	KEY <sup>Note</sup>							

Symbol	7	6	5	4	3	2	1	0
SCMMD	0	0	0	0	0	0	MDSEL	CKSEL

CKSEL	Object of the oscillation stop detection
0	Oscillation status of the main system clock ( $F_{MX}$ )
1	Oscillation status of the subsystem clock ( $F_{SX}$ )

MDSEL	Action after oscillation stop detection
0	Interrupt generated after oscillation stop detection
1	Reset generated after oscillation stop detection

Note: When rewriting MDSEL and CKSEL, “0x3C” should be written to the high 8 bit (KEY) of SCMMD at the same time.

For example, after resetting, the initial value of the SCMMD register is 0x00, and the CKSEL bit set to 1 by writing 0x3C01 to the SCMMD register.

### 4.8.2.4 Oscillation stop detection status register (SCMST)

The Oscillation Stop Detection Status Register (SCMST) is a register that displays the oscillation stop detection status.

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

Figure 4-27: Format of the Oscillation Stop Detection Status Register (SCMST)

Address: 0x40022204 After reset: 000H R/W <sup>Note</sup>

Symbol	7	6	5	4	3	2	1	0
SCMST	0	0	0	0	0	0	0	OSTDF

OSTDF	Status of oscillation stop detection
0	oscillation stop undetected
1	oscillation stop detected

Note: After the oscillation stop has been detected, set the OSTDF bit to 1, write it to 0 by writing the register.

## 4.8.3 Operation of oscillation stop detection circuit

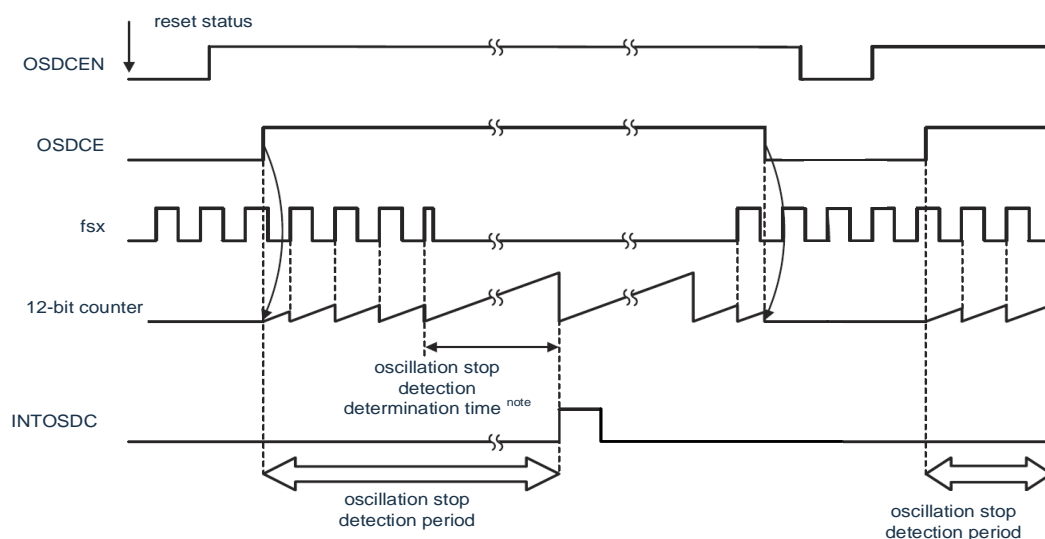
### 4.8.3.1 Operation method of oscillation stop detection circuit

1. After the external reset is released, the main system clock ( $F_{MX}$ )/subsystem clock ( $F_{SX}$ ) starts to oscillate.
2. Writing to the peripheral enable register (PER2) enables the oscillation stop detection circuit<sup>Note</sup>.
3. Write the oscillation stop detection mode register (SCMMD) to select whether to detect the oscillation status of the main system clock ( $F_{MX}$ ) or the subsystem clock ( $F_{SX}$ ), and select the action after the oscillation stop is detected, whether to generate a reset or an interrupt.
4. Write the OSDCE bit of the oscillation stop detection control register (SCMCTL) to start the oscillation stop detection circuit.
5. During the operation of the oscillation stop detection circuit, the main system clock ( $F_{MX}$ )/subsystem clock ( $F_{SX}$ ) is always stopped during the oscillation stop determination time, and the oscillation stop detection signal is output to generate a reset or interrupt.
6. When the CPU clock is the main system clock ( $F_{MX}$ ) or the CPU clock is the PLL clock and the main system clock ( $F_{MX}$ ) is used as the PLL input, and there is a main system clock ( $F_{MX}$ ) oscillation stop detection, the CPU clock will switch to the internal high-speed oscillation clock of 8 divisions ( $F_{HOCO}/8$ ), when the CPU clock is the subsystem clock ( $F_{SX}$ ) and there is a subsystem clock ( $F_{SX}$ ) oscillation stop detection, the CPU clock will switch to the internal low-speed oscillation clock ( $F_{IL}$ ). Clearing the OSTDF by writing the SCMST register to 0 in software will cut the CPU clock back to the original clock.

Note: Since there may be false detection when bit4 (OSDCEN) of PER2 is written to 1, when using the oscillation stop detection interrupt, you must clear the interrupt flag bit after writing PER2, and then turn on the interrupt enable. For the interrupt register corresponding to the oscillation stop detection interrupt, please refer to “Chapter 23 Interrupt Function”.

Figure 4-28: Timing of the oscillation stop detection circuit

(Take the detection object as fsx, and the interrupt occurs after the oscillation stop is detected)



Note: Oscillation stop determination time = internal low-speed oscillation clock ( $f_{IL}$ ) cycle  $\times$  ((OSDCCMP11~OSDCCMP0 setting value) + 1).

#### 4.8.4 Operation of oscillation stop detection circuit in deep sleep mode

If the oscillation stop detection circuit is enabled before entering deep sleep, the oscillation stop detection function will be switched off automatically after entering deep sleep, and the oscillation stop detection function will be switched on again after the standby release signal comes.

For standby de-signalling, please refer to "Standby Function".

#### 4.8.5 Notes on the oscillation stop detection function

The oscillation stop detection circuit is used together with the watchdog timer.

The oscillation stop detection, which can be used under any of the following conditions:

- When bit0 (WDSTBYON) and bit4 (WDTON) of option byte (00C0H) are 1 and bit4 (WUTMMCK0) of OSMC register is 0
- When bit4 of OSMC register (WUTMMCK0) is 1.

## Chapter 5 General-purpose timer unit Timer8

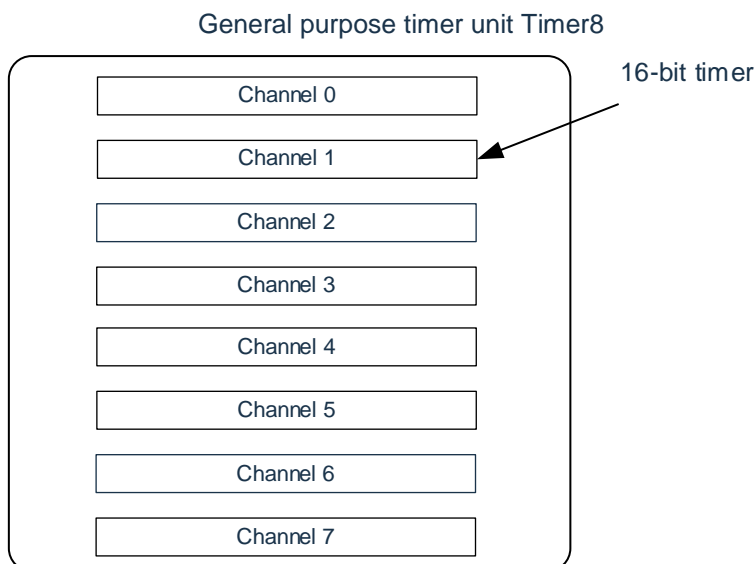
This product is equipped with a general-purpose timer unit Timer8, containing 8 channels. The number of channels of the timer unit varies from product to product.

Description:

1. The symbol "m" in the following part of this chapter represents the unit number, and this product is equipped with a general-purpose Timer8, so m=0.
2. In the following part of this chapter, the symbol "n" represents the channel number (in this chapter, n=0~7), and the availability of timer input/output pins for each channel varies from product to product. For details, please refer to "Chapter 2 Port Function".

The general-purpose timer unit Timer8 has 8 16-bit timers.

Each 16-bit timer is called a "channel" and can be used separately as a standalone timer or combined 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"><li>• Interval timer (refer to 5.8.1)</li><li>• Square wave output (refer to 5.8.1)</li><li>• External event counter (refer to 5.8.2)</li><li>• Frequency divider (refer to 5.8.3)</li><li>• Measurement of input pulse interval (refer to 5.8.4)</li><li>• Measurement of the high and low level width of the input</li><li>• Delay counter (refer to 5.8.6)</li></ul>	<ul style="list-style-type: none"><li>• Single trigger pulse output (refer to 5.9.1)</li><li>• PWM output (refer to 5.9.2)</li><li>• Multiple PWM outputs(refer to 5.9.3)</li></ul>

Able to realize LIN-bus communication by coordination of channel 3 and UART0 of general-purpose serial communication unit.



## 5.1 General-purpose timer unit functions

The general-purpose timer unit has the following functions:

### 5.1.1 Independent channel operation functions

Independent channel operation function is the function that can use any channel independently without being affected by the operation mode of other channels.

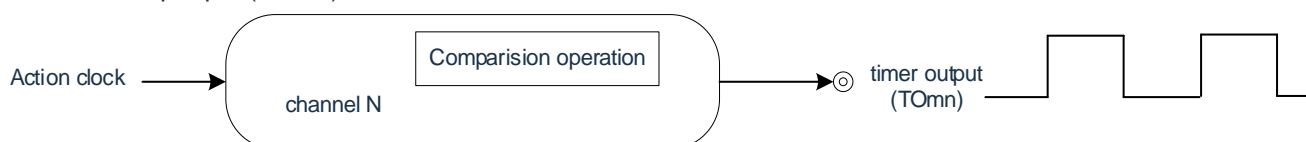
#### (1) Interval Timer

It can be used as a reference timer to generate interrupts (INTTMn) at fixed intervals.



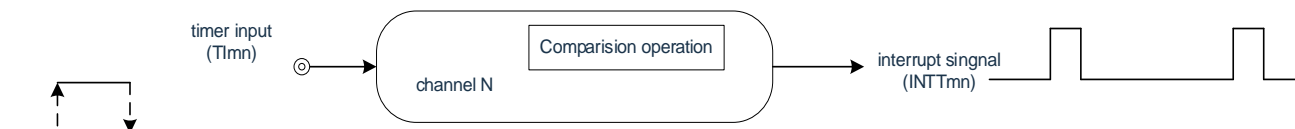
#### (2) Square wave output

Whenever the INTTMn interrupt is generated, it runs alternately and outputs a 50% duty cycle square wave from the output pin (TOMn) of the timer.



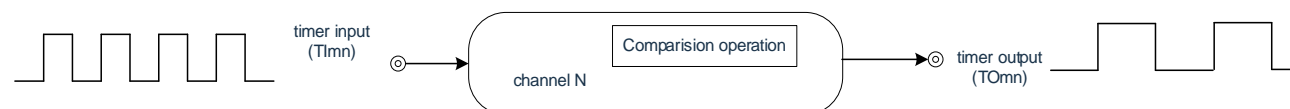
#### (3) External event counter

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.



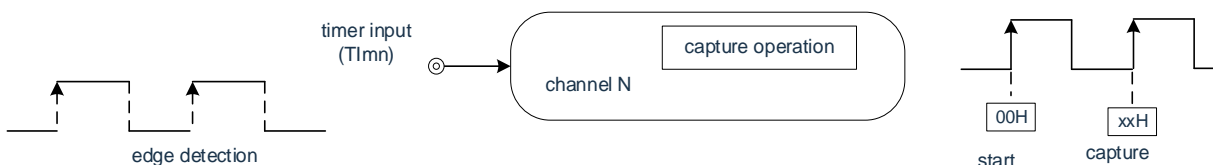
#### (4) Frequency divider function (channel 0 only)

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



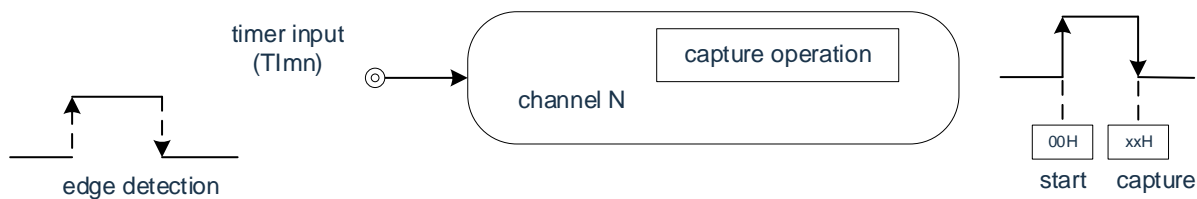
#### (5) Measurement of input pulse interval

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.



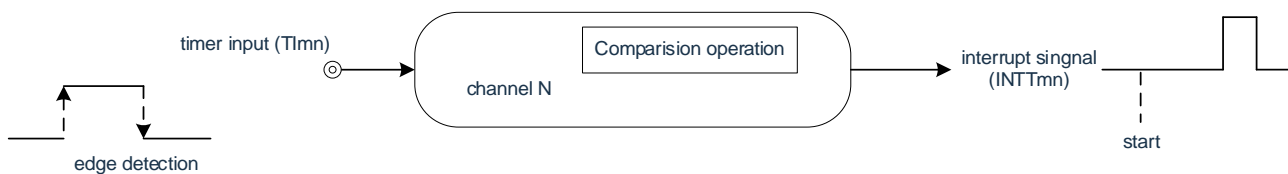
#### (6) Measurement of the high and low level width of the input signal

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

The count starts at the active edge of the input signal at the timer input pin (TImn) and an interrupt is generated after an arbitrary delay period.



Remark:

1. m: unit number (m=0) n: channel number (n=0~7)
2. The availability of timer input/output pins for each channel varies from product to product.

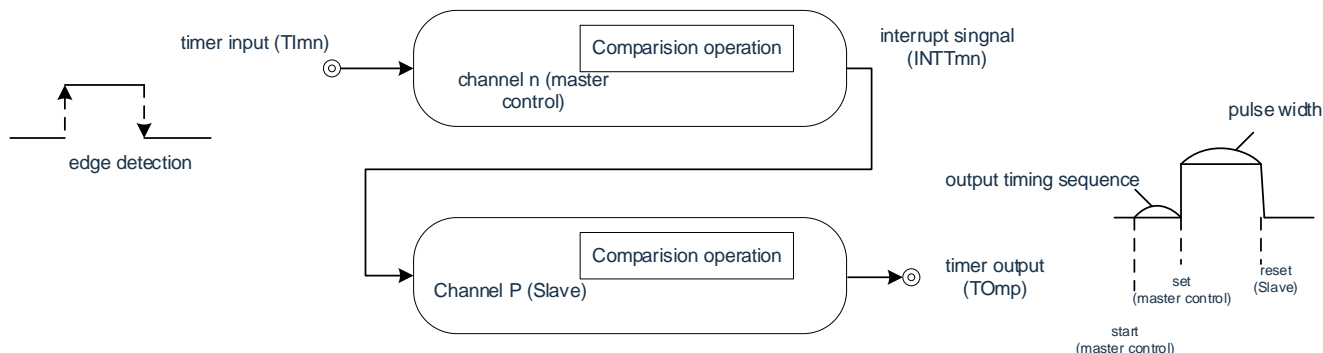
## 5.1.2 Multi-channel linkage operation functions

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

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

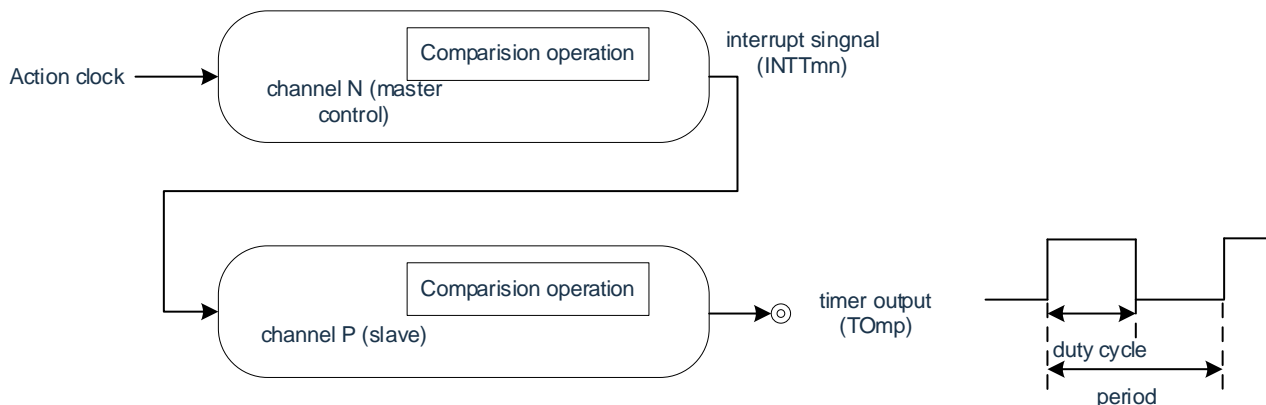
### (1) Single trigger pulse output

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



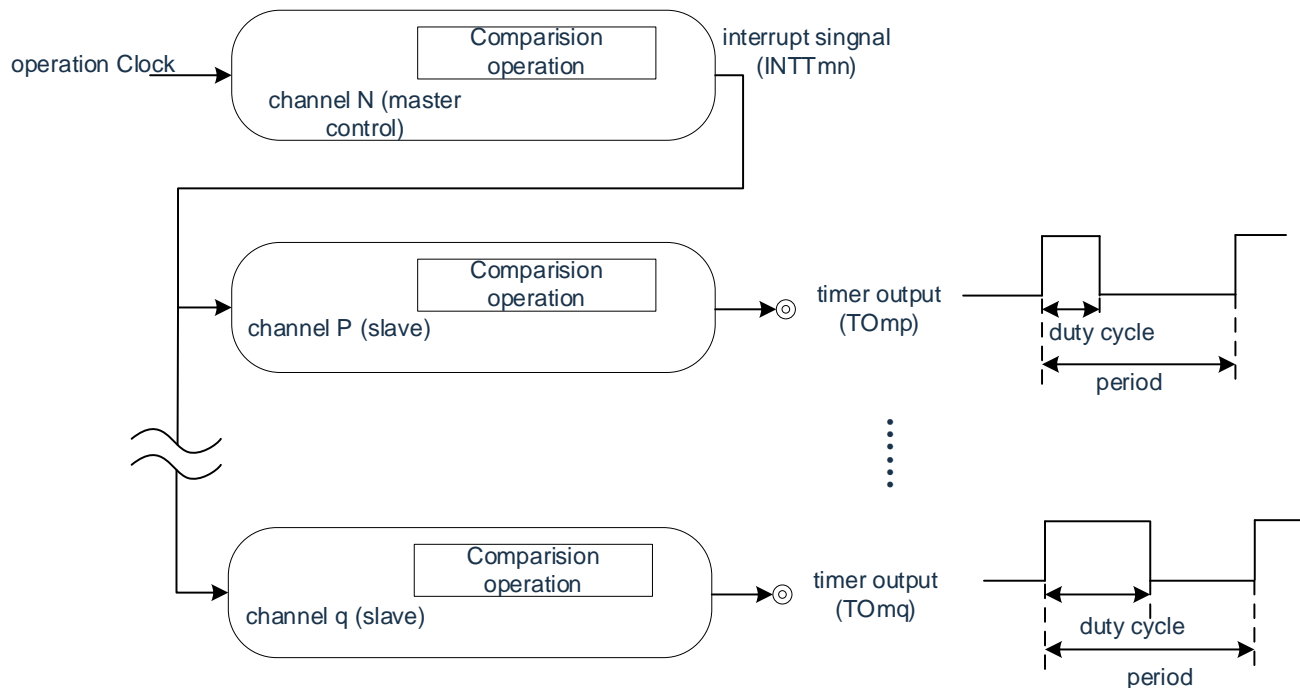
### (2) PWM(Pulse Width Modulation) output

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



### (3) Multiple PWM (Pulse Width Modulation) outputs

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



Note: Please refer to "5.4.1 Basic rules of multi-channel linkage operation function" for the rule details of multi-channel linkage operation function.

Remark: m: unit number (m=0) n: channel number (n=0~7)

p, q: slave channel number ( $n < p < q \leq 7$ )

### 5.1.3 LIN-bus support function (channel 3 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.

**Note:** 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 the general-purpose timer unit

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

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

Item	Structure
Counter	Timer Count Register mn (TCRmn)
Register	Timer data register mn (TDRmn)
Timer input	TI00~TI07 <sup>Note1</sup> , RxD0 pins (for LIN-bus)
Timer output	TO00~TO07 <sup>Note1</sup> , output control circuit
Control register	<p>&lt; Register of Unit Setting Section &gt;</p> <ul style="list-style-type: none"> <li>• Peripheral enable register 0(PER0)</li> <li>• Timer clock selection register m (TPSm)</li> <li>• Timer channel enable status register m (TEm)</li> <li>• Timer channel start register m (TSM)</li> <li>• Timer channel stop register m (TPSm)</li> <li>• Timer input selection register 0 (TIS0)</li> <li>• Timer output enable register m (TOEm)</li> <li>• Timer output register m (TOM)</li> <li>• Timer output level register m (TOLm)</li> <li>• Timer output mode register m (TOMm)</li> </ul>
	<p>&lt; Register per channel &gt;</p> <ul style="list-style-type: none"> <li>• Timer mode register mn (TMRmn)</li> <li>• Timer status register mn (TSRmn)</li> <li>• Input switching control register (ISC)</li> <li>• Noise filter enable register (NFEN1)</li> <li>• Port mode control register (PMCxx)<sup>Note2</sup></li> <li>• Port mode register (PMxx)<sup>Note2</sup></li> <li>• Port register (Pxx)<sup>Note2</sup></li> </ul>

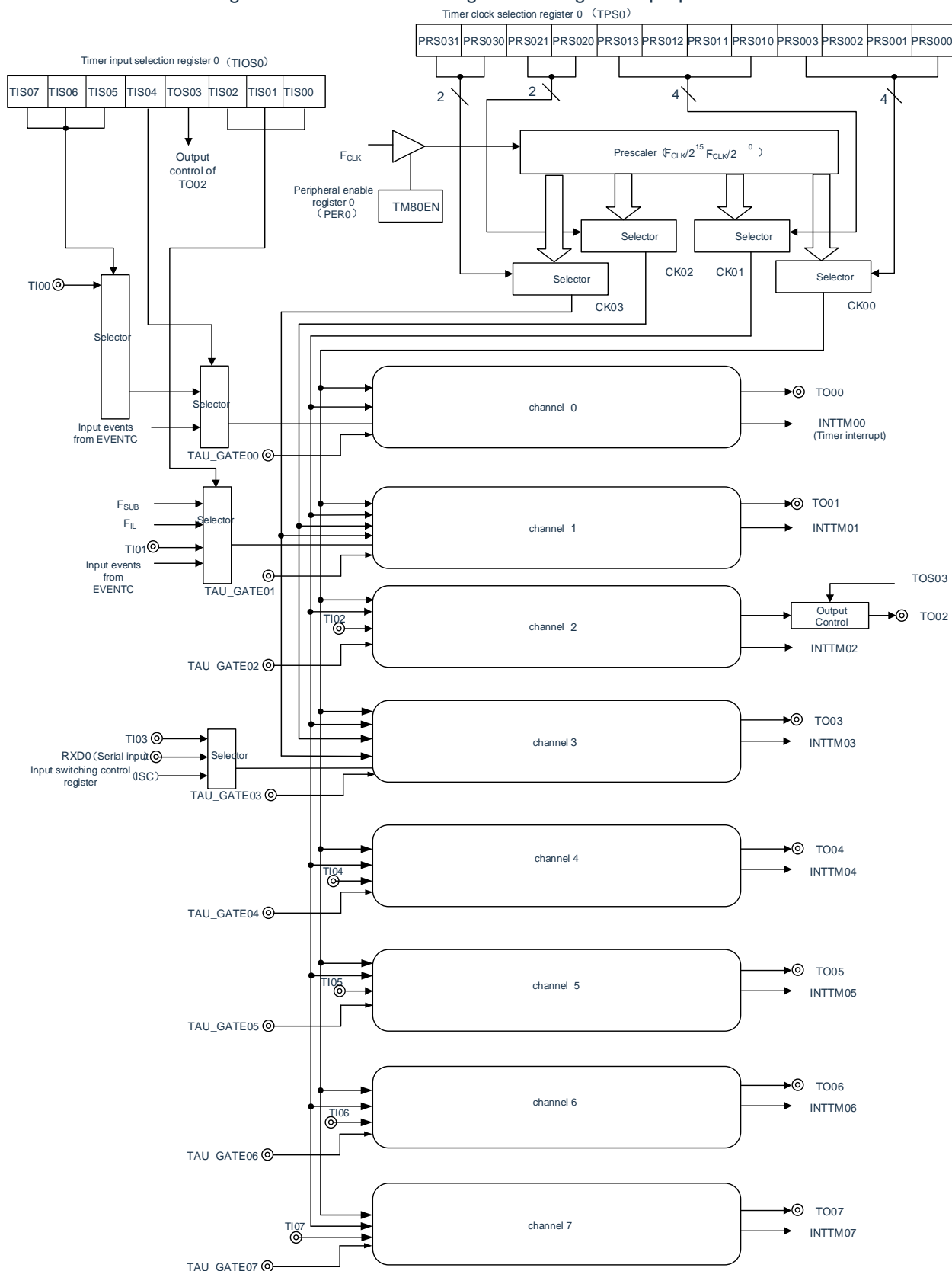
Note 1: The availability of timer input/output pins for each channel varies from product to product.

Note 2: The set Port Mode Control Register (PMCxx), Port Mode Register (PMxx) and Port Register (Pxx) vary from product to product. For details, please refer to "Chapter 2 Port Function".

Remark: m: unit number (m=0) n: channel number (n=0~7)

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

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



Remark:  $F_{SUB}$  : Subsystem clock frequency

$F_{IL}$  : Clock frequency of the low-speed internal oscillator

## 5.2.1 General-purpose timer unit register list

Register base address: 0x40043000

Offset Address	Register Name	R/W	Bit Width	Reset Value
0x000	TCR00	R	16	FFFFH
0x002	TCR01	R	16	FFFFH
0x004	TCR02	R	16	FFFFH
0x006	TCR03	R	16	FFFFH
0x008	TCR04	R	16	FFFFH
0x00A	TCR05	R	16	FFFFH
0x00C	TCR06	R	16	FFFFH
0x00E	TCR07	R	16	FFFFH
0x010	TMR00	R/W	16	0000H
0x012	TMR01	R/W	16	0000H
0x014	TMR02	R/W	16	0000H
0x016	TMR03	R/W	16	0000H
0x018	TMR04	R/W	16	0000H
0x01A	TMR05	R/W	16	0000H
0x01C	TMR06	R/W	16	0000H
0x01E	TMR07	R/W	16	0000H
0x020	TSR00	R	16	0000H
0x022	TSR01	R	16	0000H
0x024	TSR02	R	16	0000H
0x026	TSR03	R	16	0000H
0x028	TSR04	R	16	0000H
0x02A	TSR05	R	16	0000H
0x02C	TSR06	R	16	0000H
0x02E	TSR07	R	16	0000H
0x030	TE0	R	16	0000H
0x032	TS0	R/W	16	0000H
0x034	TT0	R/W	16	0000H
0x036	TPS0	R/W	16	0000H
0x038	TO0	R/W	16	0000H
0x03A	TOE0	R/W	16	0000H
0x03C	TOL0	R/W	16	0000H
0x03E	TOM0	R/W	16	0000H
0x040	TDR00	R/W	16	0000H
0x042	TDR01	R/W	16	0000H
0x044	TDR02	R/W	16	0000H
0x046	TDR03	R/W	16	0000H
0x048	TDR04	R/W	16	0000H
0x04A	TDR05	R/W	16	0000H
0x04C	TDR06	R/W	16	0000H
0x04E	TDR07	R/W	16	0000H



## 5.2.2 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)”).

Figure5-3: Table of Timer Count Register mn (TCRmn)

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

Remark: m: unit number (m=0) n: channel number (n=0~7)

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 TM80EN bit of the Peripheral Permitted Registration (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

Note: 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.

Figure5-2: The read value of the Timer Count Register mn (TCRmn) in each operating mode

Operation mode	Counting method	Timer Count Register mn (TCRmn) read value <sup>Note</sup>			
		Change after un-reset value when in run mode	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	Decrement count	FFFFH	The value when stopped	Indefinite value	-
Capture Mode	Incremental counting	0000H	The value when stopped	Indefinite value	-
Event counter mode	Decrement count	FFFFH	The value when stopped	Indefinite value	-
Single count mode	Decrement count	FFFFH	The value when stopped	Indefinite value	FFFFH
Capture & Single Count Mode	Incremental counting	0000H	The value when stopped	Indefinite value	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) n: channel number (n=0~7)

### 5.2.3 Timer data register mn (TDRmn)

This is a 16-bit register that can be switched between capture and comparison functions. The operation mode is selected via the MDmn3 to MDmn0 bits of the Timer Mode Register mn (TMRmn) for switching between the capture and comparison functions.

TDRmn registers can be overwritten at any time. This register can be read and written in 16-bit increments.

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

Figure5-4: Table of timer data registers mn (TDRmn) (n=0~7)

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

- (i) The timer data register mn (TDRmn) is used as a comparison register for the case

The count is decremented from the setpoint of the TDRmn register, and an interrupt signal (INTTMmn) is generated when the count value changes to "0000H". The value of the TDRmn register is held until it is rewritten.

Note: The TDRmn register set to the compare function does not perform capture operation even if a capture trigger signal is input.

- (ii) The timer data register mn (TDRmn) is used as a capture register in the case

The count value of the timer count register mn (TCRmn) is captured to the TDRmn register by input capture triggering.

A valid edge of the TImn pin can be selected as the capture trigger signal. The selection of capture triggers is set by the timer mode register mn (TMRmn).

Remark: m: unit number (m=0) n: channel number (n=0~7)

## 5.3 Registers that control the general-purpose timer unit

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

- Peripheral enable register 0(PER0)
- Timer clock selection 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 (TPSm)
- 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)
- Input switching control register (ISC)
- Noise filter enable register (NFEN1)
- Port mode control register (PMCxx)
- Port mode register (PMxx)
- Port register (Pxx)

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

Remark: m: unit number (m=0) n: channel number (n=0~7)

### 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 the general purpose timer, bit0 (TM80EN) must be set to "1".

The PER0 register is set by a 32-bit memory manipulation instruction.

After a reset signal is generated, the value of PER0 register becomes "000000H".

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

Address: 40020420H		After reset: 00000000H				R/W		
Symbol	7	6	5	4	3	2	1	0
PER0	RTCEN <sup>Note</sup>	IICAEN	IRDAEN	SCI2EN	SCI1EN	SCI0EN	TMAEN	TM80EN

TM80EN	Control of the input clock of the general-purpose timer
0	Stop providing input clock. • The SFR used by the general-purpose timers cannot be written. • General purpose timers are in the reset state.
1	Provides input clock. • The SFR used by the general-purpose timers can be read and written.

Note: To set the general-purpose timer unit, the following registers must first be set with the TM80EN bit at "1".

When the TM80EN bit is "0", the value of the control registers of the Timer Array Unit is the initial value, and the write operation is ignored (except for Timer Input/Output Selection Register 0 (TIOS0), Input Switching Control Register (ISC), Noise Filter Enable Register (NFEN1), Port Mode Control Register (PMCx), Port Mode Register PMx and Port Register Px).

- Timer status register mn (TSRmn)
- Timer channel enable status register m (TEm)
- Timer channel start register m (TSM)
- Timer channel stop register m (TPSM)
- 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 selection register m (TPSm)

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

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

- 1) Cases where bits PRSm00 to PRSm03 can be rewritten (n=0 to 7):

All channels with CKm0 selected as the operating clock (CKSmn1, CKSmn0 = 0, 0) are in the stop state (TEmn = 0).

- 2) Cases where bits PRSm10 to PRSm13 can be rewritten (n=0 to 7):

All channels with CKm2 selected as the operating clock (CKSmn1, CKSmn0 = 0, 1) are in the stop state (TEmn = 0).

- 3) Cases where bits PRSm20 and PRSm21 can be rewritten (n=1, 3):

All channels with CKm1 selected as the operating clock (CKSmn1, CKSmn0 = 1, 0) are in the stop state (TEmn = 0).

- 4) Cases where bits PRSm30 and PRSm31 can be rewritten (n=1, 3):

All channels with CKm3 selected as the operating clock (CKSmn1, CKSmn0 = 1, 1) are in the stop state (TEmn = 0).

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

Figure 5-6: 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

PRS mk3	PRS mk2	PRS mk1	PRS mk0	Selection of operating clock (CKmk) <sup>Note</sup> (k=0, 1)					
					F <sub>CLK</sub> =2MHz	F <sub>CLK</sub> =4MHz	F <sub>CLK</sub> =8MHz	F <sub>CLK</sub> =20MHz	F <sub>CLK</sub> =32MHz
0	0	0	0	F <sub>CLK</sub>	2MHz	4MHz	8MHz	20MHz	32MHz
0	0	0	1	F <sub>CLK</sub> /2	1MHz	2MHz	4MHz	10MHz	16MHz
0	0	1	0	F <sub>CLK</sub> /2 <sup>2</sup>	500KHz	1MHz	2MHz	5MHz	8MHz
0	0	1	1	F <sub>CLK</sub> /2 <sup>3</sup>	250KHz	500KHz	1MHz	2.5MHz	4MHz
0	1	0	0	F <sub>CLK</sub> /2 <sup>4</sup>	125KHz	250KHz	500KHz	1.25MHz	2MHz
0	1	0	1	F <sub>CLK</sub> /2 <sup>5</sup>	62.5KHz	125KHz	250KHz	625KHz	1MHz
0	1	1	0	F <sub>CLK</sub> /2 <sup>6</sup>	31.3KHz	62.5KHz	125KHz	313KHz	500KHz
0	1	1	1	F <sub>CLK</sub> /2 <sup>7</sup>	15.6KHz	31.3KHz	62.5KHz	156KHz	250KHz
1	0	0	0	F <sub>CLK</sub> /2 <sup>8</sup>	7.81KHz	15.6KHz	31.3KHz	78.1KHz	125KHz
1	0	0	1	F <sub>CLK</sub> /2 <sup>9</sup>	3.91KHz	7.81KHz	15.6KHz	39.1KHz	62.5KHz
1	0	1	0	F <sub>CLK</sub> /2 <sup>10</sup>	1.95KHz	3.91KHz	7.81KHz	19.5KHz	31.25KHz
1	0	1	1	F <sub>CLK</sub> /2 <sup>11</sup>	977Hz	1.95KHz	3.91KHz	9.77KHz	15.6KHz
1	1	0	0	F <sub>CLK</sub> /2 <sup>12</sup>	488Hz	977Hz	1.95KHz	4.88KHz	7.81KHz
1	1	0	1	F <sub>CLK</sub> /2 <sup>13</sup>	244Hz	488Hz	977Hz	2.44KHz	3.91KHz
1	1	1	0	F <sub>CLK</sub> /2 <sup>14</sup>	122Hz	244Hz	488Hz	1.22KHz	1.95KHz
1	1	1	1	F <sub>CLK</sub> /2 <sup>15</sup>	61.0Hz	122Hz	244Hz	610Hz	977Hz

Note: In case of changing the clock selected as F<sub>CLK</sub> (changing the value of the system clock control register (CKC)), the general-purpose timer unit must be stopped (TTm=0,100FH). The General Timer Unit needs to be stopped even when the running clock (F<sub>MCK</sub>) is selected or when the active edge of the TIMn pin input signal is used.

Notice:

1. The bit 15, 14, 11 and 10 must be set to "0".
2. If F<sub>CLK</sub> (no divider) is selected as the running 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.

Remark:

1. F<sub>CLK</sub>: Clock frequency of CPU/peripheral hardware
2. The clock waveform selected by TPSm register is high for only 1 FCLK cycle (m=1~15) from the rising edge. For details, please refer to "5.5. 1 Counting Clock (F<sub>TCLK</sub>)".

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

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPSm	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 the running clock (CKm2) <sup>Note</sup>					
			F <sub>CLK</sub> =2MHz	F <sub>CLK</sub> =4MHz	F <sub>CLK</sub> =8MHz	F <sub>CLK</sub> =20MHz	F <sub>CLK</sub> =32MHz
0	0	F <sub>CLK</sub> /2	1MHz	2MHz	4MHz	10MHz	16MHz
0	1	F <sub>CLK</sub> /22	500KHz	1MHz	2MHz	5MHz	8MHz
1	0	F <sub>CLK</sub> /24	125KHz	250KHz	500KHz	1.25MHz	2MHz
1	1	F <sub>CLK</sub> /26	31.3KHz	62.5KHz	125KHz	313KHz	500KHz

PRS m31	PRS m30	Selection of the running clock (CKm3) <sup>Note</sup>					
			F <sub>CLK</sub> =2MHz	F <sub>CLK</sub> =4MHz	F <sub>CLK</sub> =8MHz	F <sub>CLK</sub> =20MHz	F <sub>CLK</sub> =32MHz
0	0	F <sub>CLK</sub> /28	7.81KHz	15.6KHz	31.3KHz	78.1KHz	125KHz
0	1	F <sub>CLK</sub> /210	1.95KHz	3.91KHz	7.81KHz	19.5KHz	31.3KHz
1	0	F <sub>CLK</sub> /212	488Hz	977Hz	1.95KHz	4.88KHz	7.81KHz
1	1	F <sub>CLK</sub> /214	122Hz	244Hz	488Hz	1.22KHz	1.95KHz

Note: In case of changing the clock selected as F<sub>CLK</sub> (changing the value of the system clock control register (CKC)), the general-purpose timer unit must be stopped (TTm=0,100FH). The General Timer Unit needs to be stopped even when the running clock (F<sub>MCK</sub>) is selected or when the active edge of the TIMn pin input signal is used.

Notice: The bit 15, 14, 11 and 10 must be set to "0".



### 5.3.3 Timer mode register mn (TMRmn)

The MRmn register is the register for setting the operation mode of channel n. It carries out the selection of the operation clock ( $F_{MCK}$ ), the selection of the count clock, the selection of master/slave, the setting of start trigger and capture trigger, the selection of the active edge of the timer input and the setting of the operation mode (interval, capture, event counter, single count, capture & single count).

It is forbidden to overwrite the TMRmn register during operation ( $TE_{mn}=1$ ). However, it is possible to rewrite bit7 and bit6 ( $CIS_{mn1}$ ,  $CIS_{mn0}$ ) in a part of the operation ( $TE_{mn}=1$ ) (for details please refer to "5.8 Independent channel operation function for general purpose timer units" and "5.9 Multi-channel linkage operation function for general purpose timer units").

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".

Note: The bit11 of the TMRmn register varies by channel.

TMRm2, TMRm4, TMRm6: MASTERmn bit ( $n=2, 4, 6$ )

TMRm0, TMRm5, TMRm7: Fixed to "0".

Figure 5-8: 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 ( $n=2,4,6$ )	CKS mn1	CKS mn0	0	CCS mn	MAS TERmn	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	GTS mn1	GTS mn0	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 ( $n=0,1,3,5,7$ )	CKS mn1	CKS mn0	0	CCS mn	0 <sup>Note 1</sup>	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	GTS mn1	GTS mn0	MD mn3	MD mn2	MD mn1	MD mn0

CKSmn1	CKSmn0	Selection of channel n operating clock ( $F_{MCK}$ )
0	0	The operating clock CKm0 set by the timer clock selection register m (TPSm).
0	1	The operating clock CKm2 set by the timer clock selection register m (TPSm).
1	0	The operating clock CKm1 set by the timer clock selection register m (TPSm).
1	1	The operating clock CKm3 set by the timer clock selection register m (TPSm).
The operating 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 choose operation clocks CKm2 and CKm3.		

CCSmn	Selection of channel n counting 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 Channel 3: The active edge of the input signal selected by ISC
Counting clocks ( $F_{TCLK}$ ) are used in counters, output control circuits, and interrupt control circuits.	

Note: bit11 is a read-only bit, fixed at "0", ignoring write operations.

Notice:

- The bit 13 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 ( $TTm=0,10FFH$ ) must be stopped even if the operating clock ( $F_{MCK}$ ) specified by the CKSmn0 bit and the CKSmn bit or the active edge of the TImn pin input signal is selected as the counting clock ( $F_{TCLK}$ ).

Remark: m: unit number ( $m=0$ ) n: channel number ( $n=0\sim7$ )

Figure5-9: Table of timer mode register mn (TMRmn) (2/ 4)

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

(Bit11 of TMRmn (n=2,4,6))

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 2,4,6 can be set as master channel (MASTERmn=1). Channel 0 is fixed to "0" (because channel 0 is the highest bit channel, it is not relevant to the setting of this bit and is used as a master channel). For channels that are used as independent channel operation functions, the MASTERmn bit should be set to "0".	

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).
Others:			Settings are disabled.

Note 1: Bit11 is a read-only bit, fixed to "0", ignoring write operations.

Remark: m: unit number (m=0) n: channel number (n=0~7)

Figure 5-10: 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=2,4,6)	CKS mn1	CKS mn0	0	CCS mn	MAS TERmn	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	GTS mn1	GTS mn0	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=0,1,3, 5,7)	CKS mn1	CKS mn0	0	CCS mn	0 <sup>Note 1</sup>	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	GTS mn1	GTS mn0	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	Both edges Start triggering, falling edge, capture triggering, rising edge
1	1	Both edges(when measuring high level width) Start triggering, rising edge, capture triggering, falling edge
The CISmn1~CISmn0 bits must be set to "10B" when the STSmn2~STSmn0 bits are not "010B" and the both edges are used.		

GTSmn1	GTSmn0	Counting clock source start-up setting
0	0	Invalid gate control signal
0	1	Gating signal is high and counting clock source start
1	0	Gating signal is low and counting clock source start
1	1	Invalid gate control signal

Note 1: Bit11 is a read-only bit, fixed to "0", ignoring write operations.

Remark: m: unit number (m=0) n: channel number (n=0~7)

Figure 5-11: 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=2,4,6)	CKS mn1	CKS mn0	0	CCS mn	MAS TERmn	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	GTS mn1	GTS mn0	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=0,1,3, 5,7)	CKS mn1	CKS mn0	0	CCS mn	0 <sup>Note 1</sup>	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	GTS mn1	GTS mn0	MD mn3	MD mn2	MD mn1	MD mn0

MD mn3	MD mn2	MD mn1	Setting of channel n operation mode	Corresponding functions	Counting operation of TCR
0	0	0	Interval timer mode	Interval timer/square wave output Frequency divider function/PWM	Decrement count
0	1	0	Capture Mode	Measurement of input pulse interval	Incremental counting
0	1	1	Event counter mode	External event counter	Decrement count
1	0	0	Single count mode	Delay counter/single trigger pulse output/PWM output	Decrement count
1	1	0	Capture & Single Count Mode	Measurement of the high and low level width of the input signal	Incremental counting
Others:			Settings are disabled.		
The operation of each mode varies depending on the MDmn0 bit (refer to the table below).					

Operation mode (MDmn3~MDmn1 bit setting (refer to the table above))	MD mn0	Start counting and interrupt settings
<ul style="list-style-type: none"> <li>Interval timer mode (0, 0, 0)</li> <li>Capture mode (0, 1, 0)</li> </ul>	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"> <li>Event counter mode (0, 1, 1)</li> </ul>	0	No timer interrupt is generated when counting starts (the output of the timer does not change).
<ul style="list-style-type: none"> <li>Single Count Mode<sup>Note 2</sup> (1, 0, 0)</li> </ul>	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 <sup>Note 3</sup> . No interruption at this time.
<ul style="list-style-type: none"> <li>Capture &amp; Single Count Mode (1, 1, 0)</li> </ul>	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", ignoring write operations.

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

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

Remark: m: unit number (m=0) n: channel number (n=0~7)

### 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 5-5 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.

Figure5-12: 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) n: channel number (n=0~7)

Figure 5-3: OVF bit changes and set/clear conditions in each operating mode

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

Note: Even if the counter overflows, the OVF bit does not change immediately, but changes at the time of capture thereafter.

### 5.3.5 Timer channel enable status register m (TE<sub>m</sub>)

The TE<sub>m</sub> register is a register that indicates the enable or stop status of each channel timer operation.

Each of the TE<sub>m</sub> register corresponds to each of the timer channel start register m (TS<sub>m</sub>) and timer channel stop register m (TT<sub>m</sub>). If each bit of the TS<sub>m</sub> register is "1", the corresponding bit of the TE<sub>m</sub> register is "1". If each bit of the TT<sub>m</sub> register is "1", the corresponding bit of the TE<sub>m</sub> register is cleared to "0".

The TE<sub>m</sub> register is read by a 16-bit memory manipulation instruction.

Figure5-13: Table of timer channel enable status register m (TE<sub>m</sub>)

Symbo	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TE <sub>m</sub>	0	0	0	0	0	0	0	0	TE <sub>m</sub> 7	TE <sub>m</sub> 6	TE <sub>m</sub> 5	TE <sub>m</sub> 4	TE <sub>m</sub> 3	TE <sub>m</sub> 2	TE <sub>m</sub> 1	TE <sub>m</sub> 0

Remark: m=0

TE <sub>m</sub> n	An indication of the enabled or stopped state of channel n
0	Stop state
1	Operation enable state

Remark: m: unit number (m=0) n: channel number (n=0~7)

### 5.3.6 Timer channel start register m (TSm)

The TSm register is a trigger register to initialize Timer Count Register mn (TCRmn) and set the start of each channel count operation. If each bit is set to "1", the corresponding bit of Timer Channel Enable Status Register m (TEm) is set to "1". Since the TSmn bit is the trigger bit, the TSmn bit is cleared immediately if the operation is enabled (TEmn = 1).

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

Figure 5-14: Table of timer channel start register m (TSm)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSm	0	0	0	0	0	0	0	0	TSm7	TSm6	TSm5	TSm4	TSm3	TSm2	TSm1	TSm0

Remark: m=0

TSmn	Operation of channel n enables (start) triggering
0	No triggering.
1	Set TEMn bit to "1" to and enter the count-enabled state. The count start of the TCRmn register in the count-enable state varies depending on the operation modes (refer to <b>Table 5-6</b> in "5.5.2 Start timing of the counter").

Notice:

1. The bit 15~8 must be set to "0".
2. When switching from a function that does not use TImn pin input to a function that uses TImn pin input, the following period of waiting is required from setting the timer mode register mn (TMRmn) until the TSmn bit is set to "1":

When the TImn pin noise filter is valid (TNFENmn=1): 4 operating clocks ( $F_{MCK}$ )

When the TImn pin noise filter is invalid (TNFENmn=0): 2 operating clocks ( $F_{MCK}$ )

Remark: The read value of the TSm register is always "0".

m: unit number (m=0) n: channel number (n=0~7)

### 5.3.7 Timer channel stop register m (TPSm)

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 of Timer Channel Enable Status Register m (TEm) is cleared to "1". Since the TTmn bit is a trigger bit, if the operation stop state is changed (TEmn=0), the TTmn bit is cleared immediately

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

Figure5-15: Table of timer channel stop register m (TTm)

Symb ol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TTm	0	0	0	0	0	0	0	0	TTm7	TTm6	TTm5	TTm4	TTm3	TTm2	TTm1	TTm0

Remark: m=0

TTmn	Operation of channel n stop triggering
0	No triggering.
1	Set TEmn bit to "0" to and enter the count-stopped state.

Notice: The bit 15~8 must be set to "0".

Remark:

1. The TTm register is always read as "0".
2. m: unit number (m=0) n: channel number (n=0~7)



### 5.3.8 Timer input output selection register (TIOS0)

The TIOS0 register selects the timer inputs for Channel 0 and Channel 1 of Unit 0 and the timer outputs for Channel 2. 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 5-16: Table of timer input selection register 0 (TIOS0)

Address: 0x40040C04H	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)
Others:			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 the timer output of channel 2
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 EVENTC
0	1	0	Input signal for timer input pin (TI01)
0	1	1	
1	0	0	Low-speed internal oscillator clock (F <sub>IL</sub> )
1	0	1	Sub-system Clock (F <sub>SUB</sub> )
Others:			Settings are disabled.

Notice:

1. The high and low level width of the selected timer input needs to be greater than or equal to  $1/F_{MCK} + 10ns$ . Therefore, when F<sub>SUB</sub> is selected as F<sub>CLK</sub> (CSS=1 in the CKC register), TIS02 cannot be set to "1".
2. When selecting the event input signal for ELC via Timer Input Selection Register 0 (TIOS0), F<sub>CLK</sub> must be selected via Timer Clock Selection 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.

For channel n, which allows timer output, the value of the TOMn bit of the latter timer output register m (TOM) cannot be rewritten by software, and the value reflected by the timer output function of the count operation is output from the output pin (TOMn) of the timer.

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

Figure 5-17: 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	TOE <sub>m7</sub>	TOE <sub>m6</sub>	TOE <sub>m5</sub>	TOE <sub>m4</sub>	TOE <sub>m3</sub>	TOE <sub>m2</sub>	TOE <sub>m1</sub>	TOE <sub>m0</sub>

Remark: m=0

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 operation of the TOMn bit is ignored.

Notice: The bit 15~8 must be set to "0".

Remark: m: unit number (m=0) n: channel number (n=0~7)

### 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 the timer output is disabled (TOEmn=0).

When the timer output is enabled (TOEmn=1), the rewrite operation by software is ignored and the value is changed only by the timer operation.

To use the TOMn pin as a port function, the corresponding TOMn bit must be set to "0".

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

Figure 5-18: 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	TOM7	TOM6	TOM5	TOM4	TOM3	TOM2	TOM1	TOM0

Remark: m=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: The bit 15~8 must be set to "0".

Remark: m: unit number (m=0) n: channel number (n=0~7)

### 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 link 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.

Figure5-19: Table of timer output level register m (TOLm)

Symb ol	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	TOL m7	TOL m6	TOL m5	TOL m4	TOL m3	TOL m2	TOL m1	0

Remark: m=0

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

Notice: The bit 15~8 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) n: channel number (n=0~7)

### 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.

Figure5-20: 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	TOM <sub>m7</sub>	TOM <sub>m6</sub>	TOM <sub>m5</sub>	TOM <sub>m4</sub>	TOM <sub>m3</sub>	TOM <sub>m2</sub>	TOM <sub>m1</sub>	0

Remark: m=0

TOMmn	Control of timer output mode of channel n
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: The bit 15~8 and bit0 must be set to "0".

Remark: m: unit number (m=0),

n: channel number (n=0~7)

Master control channel number: n=0, 2,4, 6

Slave channel number p:  $n < p \leq 7$

(For details of the relationship between master and slave channels, please refer to "5.4.1 Basic rules of multi-channel linkage operation function")

### 5.3.13 Input switching control register (ISC)

The ISC1 and ISC0 bits of the ISC register are used for the coordination of channel 3 and the general-purpose serial communication unit to implement LIN-bus communication. If the ISC1 bit is set to "1", the input signal of the serial data input pin (RxD0) is selected as the input of the timer.

For the setting of SSIE00 bit, please refer to "Input switching control register (ISC)".

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

Figure5-21: Table of input switching control register (ISC)

Address: 0x40040C03H	After reset: 00H		R/W					
Symbol	7	6	5	4	3	2	1	0
ISC	SSIE00	0	0	0	0	0	ISC1	ISC0

SSIE00	SSI00 pin input setting for channel 0 in slave mode for CSI00 communication
0	SSI00 pin input is invalid.
1	SSI00 pin input is valid.

ISC1	Input switching for channel 3 of general-purpose timer unit 0
0	Use the input signal from the TI03 pin as an input to the timer (normally operating).
1	Use the input signal on the RxD0 pin as a timer input (detects the wake-up signal and measures the low level width of the break field and the pulse width of the sync field).

ISC0	Input switching of external interrupt (INTP0)
0	Use the input signal on the INTP0 pin as an external interrupt input (normally operating).
1	Use the input signal on the RxD0 pin as an external interrupt input (detecting a wake-up signal).

Notice: The bit 6~2 must be set to "0".

Remark: To use LIN-bus for communication, you must set ISC1 bit to "1" and select the input signal of RxD0 pin.

### 5.3.14 Noise filter enable register (NFEN1)

The NFEN1 register sets whether the noise filter is used for the input signal of each channel timer input pin. For the pin that needs to eliminate noise, the corresponding bit must be set to "1" and make the noise filter effective. When the noise filter is valid, it checks whether the 2 clocks are the same after synchronizing via the operating clock of the object channel ( $F_{MCK}$ ), when the noise filter is invalid, it only synchronizes via the operating clock of the object channel ( $F_{MCK}$ )<sup>Note</sup>.

Note: For details, refer to "5.5. 1(2) Selecting the active edge of TImn pin input signal (CCSmn=1)", "5.5.2 Start timing of counter" and "5.7 Control of Timer Input (TImn)".

Figure5-22: Table of noise filter enable register (NFEN1)

Address: 0x40040C01	After reset: 00H		R/W					
Symbol	7	6	5	4	3	2	1	0
NFEN1	TNFEN07	TNFEN06	TNFEN05	TNFEN04	TNFEN03	TNFEN02	TNFEN01	TNFEN00

TNFEN07	Usage of input signal noise filters on the TI07 pin
0	Noise filter OFF
1	Noise filter ON

TNFEN06	Usage of input signal noise filters on the TI06 pin
0	Noise filter OFF
1	Noise filter ON

TNFEN05	Usage of input signal noise filters on the TI05 pin
0	Noise filter OFF
1	Noise filter ON

TNFEN04	Usage of input signal noise filters on the TI04 pin
0	Noise filter OFF
1	Noise filter ON

TNFEN03	Usage of input signal noise filter on TI03 pin or RxD0 pin <sup>Note</sup>
0	Noise filter OFF
1	Noise filter ON

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

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

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

Note: The applicable pin can be switched by setting the ISC1 bit of the Input Switching Control Register (ISC).

ISC1=0: You can choose whether to use the noise filter of the TI03 pin. ISC1=1: You can choose whether to use the noise filter of the RxD0 pin.

Remark: The availability of timer input/output pins for channels varies from product to product.

### 5.3.15 Registers for controlling timer input/output pin port functions

When using the general-purpose timer unit, the control registers (Port Mode Register (PMxx), Port Register (Pxx) and Port Mode Control Register (PMCxx)) of the port function multiplexed with the object channel must be set. For details, please refer to "2.3. 1 Port mode register (PMxx)", "2.3. 2 Port register (Pxx)" and "2.3. 8 Port mode control register (PMCxx)".

The Port Mode Register (PMxx), Port Register (Pxx) and Port Mode Control Register (PMCxx) are set differently depending on the product. For details, please refer to "2.5 Register settings when using the multiplexing function".

When using the multiplexed ports of the timer output pins as timer outputs, 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 must be set to "0".

When using the multiplexed ports of the timer input pins as timer inputs, the bit of the corresponding Port Mode Register (PMxx) must be set to "1" for each port and the bit of the Port Mode Control Register (PMCxx) must be set to "0". In this case, the bits of the Port Register (Pxx) can be "0" or "1".



## 5.4 Basic rules of the 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.

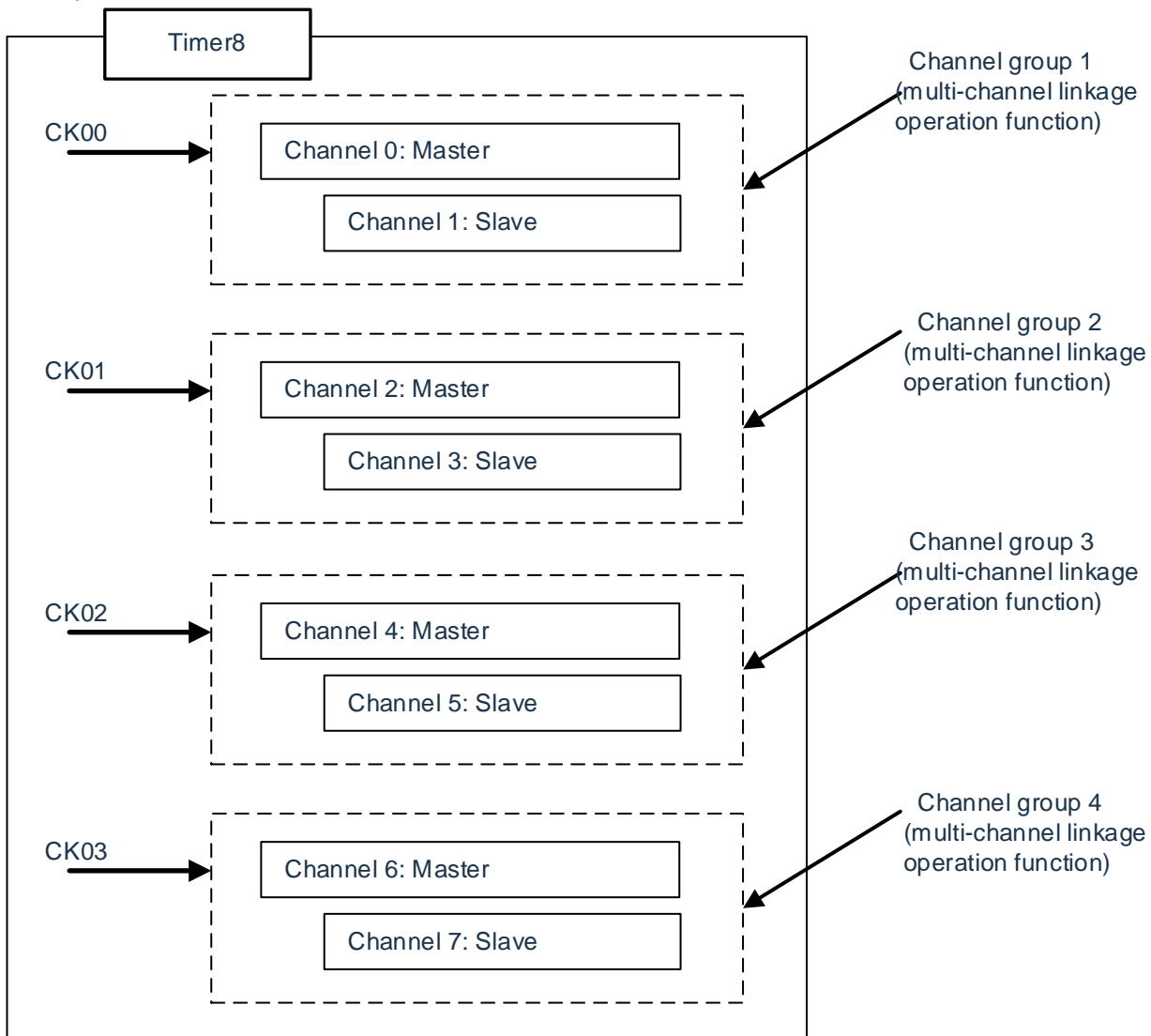
- 1) Only the even-number channel (channel 0, channel 2, channel 4, channel 6) can be set as a master channel.
- 2) Any channel other than channel 0 can be set as a slave channel.
- 3) Only the lower channel of the master channel can be set as a slave channel.  
For example, when setting channel 0 as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

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

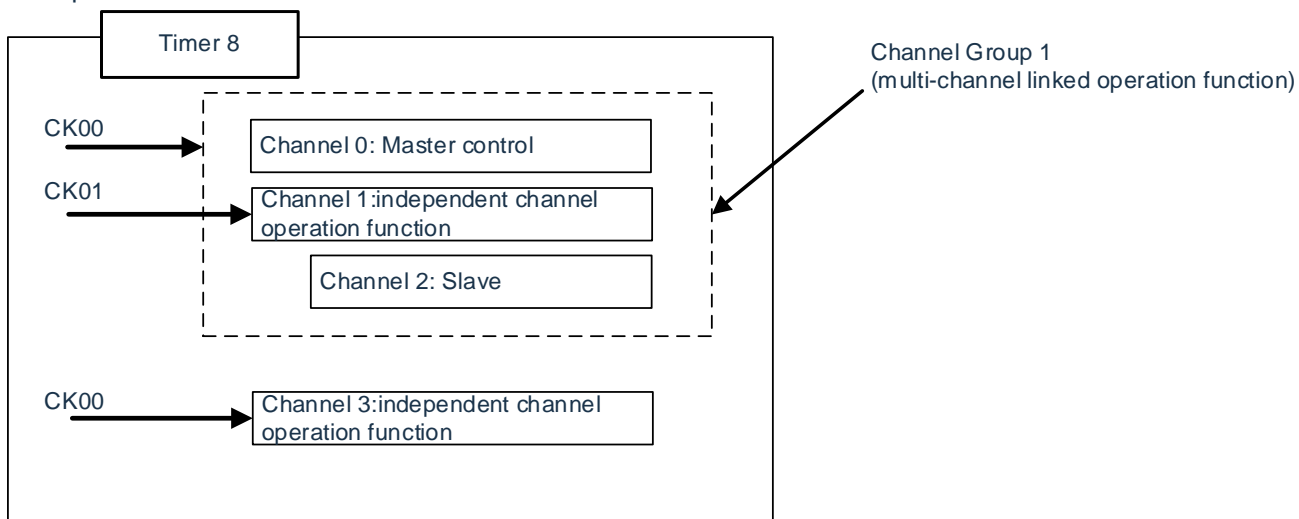
If you set 2 or more channel groups that are not linked to each other, the above basic rules do not apply to the channel groups.

Remark: m: unit number (m=0) n: channel number (n=0~7)

Example 1



Example 2



## 5.4.2 Timer channel start register m (TSM)

The TSM register is a trigger register to initialize Timer Count Register mn (TCRmn) and set the start of each channel count operation. If each bit is set to "1", the corresponding bit of Timer Channel Enable Status Register m (TEM) is set to "1". Since the TSMn bit is the trigger bit, the TSMn bit is cleared immediately if the operation is enabled (TEMn = 1).

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

Figure 5-23: Table of timer channel start register m (TSM)

Symb ol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSM	0	0	0	0	0	0	0	0	TSM7	TSM6	TSM5	TSM4	TSM3	TSM2	TSM1	TSM0

Remark: m=0

TSMn	Operation of channel n enables (start) triggering
0	No triggering.
1	Set TEMn bit to "1" to and enter the count-enabled state. The count start of the TCRmn register in the count-enable state varies depending on the operation modes (refer to <b>Table 5-6</b> in "5.5.2 Start timing of the counter").

Notice:

1. The bit 15~8 must be set to "0".
2. When switching from a function that does not use TIMn pin input to a function that uses TIMn pin input, the following period of waiting is required from setting the timer mode register mn (TMRmn) until the TSMn bit is set to "1":

When the TIMn pin noise filter is valid (TNFENmn=1): 4 operating clocks ( $F_{MCK}$ )

When the TIMn pin noise filter is invalid (TNFENmn=0): 2 operating clocks ( $F_{MCK}$ )

Remark:

1. The TSM register is always read as "0".
2. m: unit number (m=0) n: channel number (n=0~7)

## 5.5 Operation of counters

### 5.5.1 Counting 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:

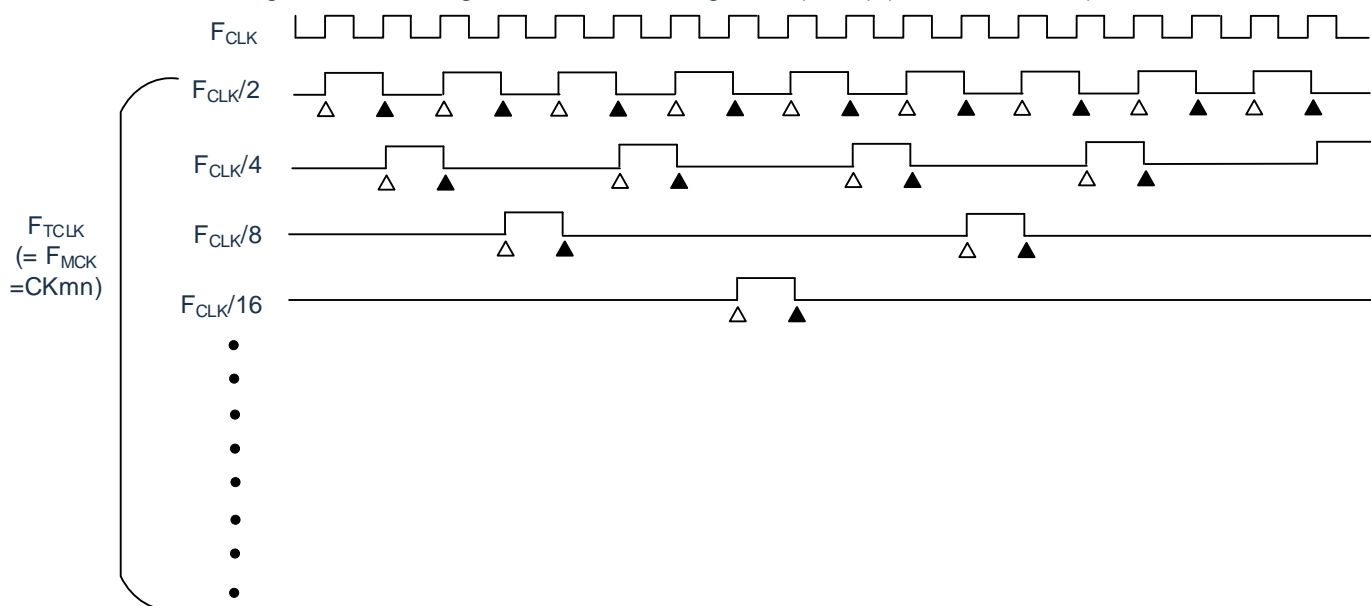
- The CKSmn0 bit and CKSmn1 bit specified operation clock ( $F_{MCK}$ )
- The active edge of the TImn pin input signal

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

- (1) Select the case where the CKSmn0 bit and the CKSmn1 bit specified operation clock ( $F_{MCK}$ ) (CCSmn=0)

According to the setting of Timer Clock Selection Register m (TPSm), the counting clock ( $F_{TCLK}$ ) is  $F_{CLK} \sim F_{CLK}/215$ . However, when the 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 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 counting clock ( $F_{TCLK}$ ) (when CCSmn=0)



Remark:

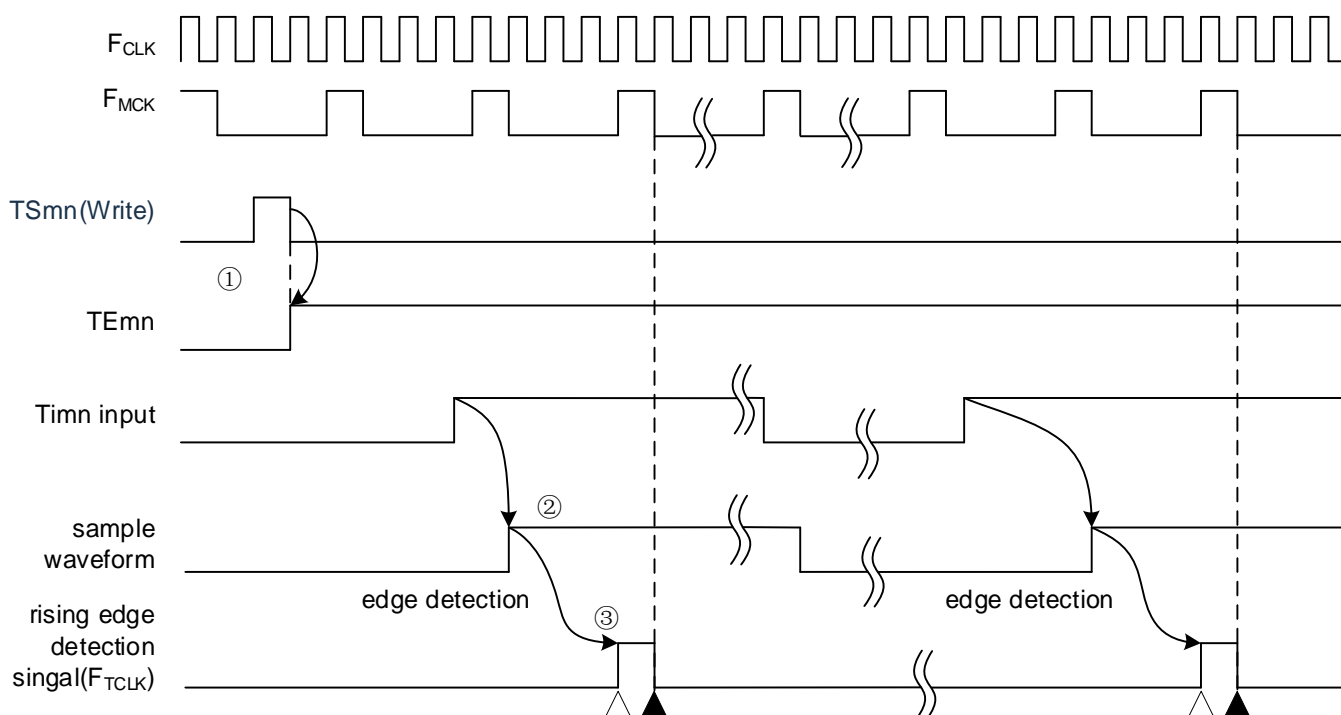
1.  $\Delta$ : The rising edge of the counting clock  
 $\blacktriangle$ : Synchronization, counter increment/decrement
2.  $F_{CLK}$ : CPU/peripheral hardware clock

(2) The case of selecting an active edge of the TImn pin input signal (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.

Figure5-25: Timing of the counting clock ( $F_{TCLK}$ ) (CCSmn=1, without noise filter)



- ① Start the timer by setting the TSmn bit and wait for a valid edge of the TImn input.
- ② The rising edge of the TImn input is sampled via  $F_{MCK}$ .
- ③ The edge is detected at the rising edge of the sampling signal and the detection signal (counting clock) is output.

Remark:

1.  $\Delta$ : The rising edge of the counting clock  
 $\blacktriangle$ : Synchronization, counter increment/decrement
2.  $F_{CLK}$ : CPU/peripheral hardware clock  
 $F_{MCK}$ : Operating 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 5-4.

Figure 5-4: Operation from the counting enable state to the start of the timer count register mn (TCRmn)

Timer operation mode	Operation after setting TSmn bit to "1"
• Interval timer mode	No operation is performed from the detection of the start trigger (TSmn=1) until the count clock is generated. The value of the TDRmn register is loaded into the TCRmn register by the first count clock and decremented by subsequent count clocks(refer to“5.5.3(1) Operation of the interval timer mode”).
• Event counter mode	The value of the TDRmn register is loaded into the TCRmn register by writing a "1" to the TSmn bit. If the input edge of TImn is detected, the count is decremented by the subsequent count clocks. (Refer to “5.5.(2) Operation of the event counter mode”).
• Capture mode	No operation is performed from the time the start trigger is detected until the count clock is generated. The “0000H” is loaded into the TCRmn register by the first count clock, and incremental counting is performed by the subsequent count clocks (refer to“5.5.3(3) Operation of the capture mode (input pulse interval measurement)”).
• Single count mode	By writing "1" to the TSmn bit while the timer is stopped (TEmn=0), it enters the wait state for the start of the trigger. No operation is performed from the time the start trigger is detected until the count clock is generated. The value of the TDRmn register is loaded into the TCRmn register by the first count clock, and decremental counting by subsequent count clocks (refer to5.5.3“(4) Operation of the single count mode”).
• Capture & single count mode	By writing "1" to the TSmn bit while the timer is stopped (TEmn=0), it enters the wait state for the start of the trigger. No operation is performed from the time the start trigger is detected until the count clock is generated. The “0000H” is loaded into the TCRmn register by the first count clock, and incremental counting is performed by the subsequent count clocks (refer to“5.5.3(5) Operation of capture & single count mode (measurement of high level width) ”).

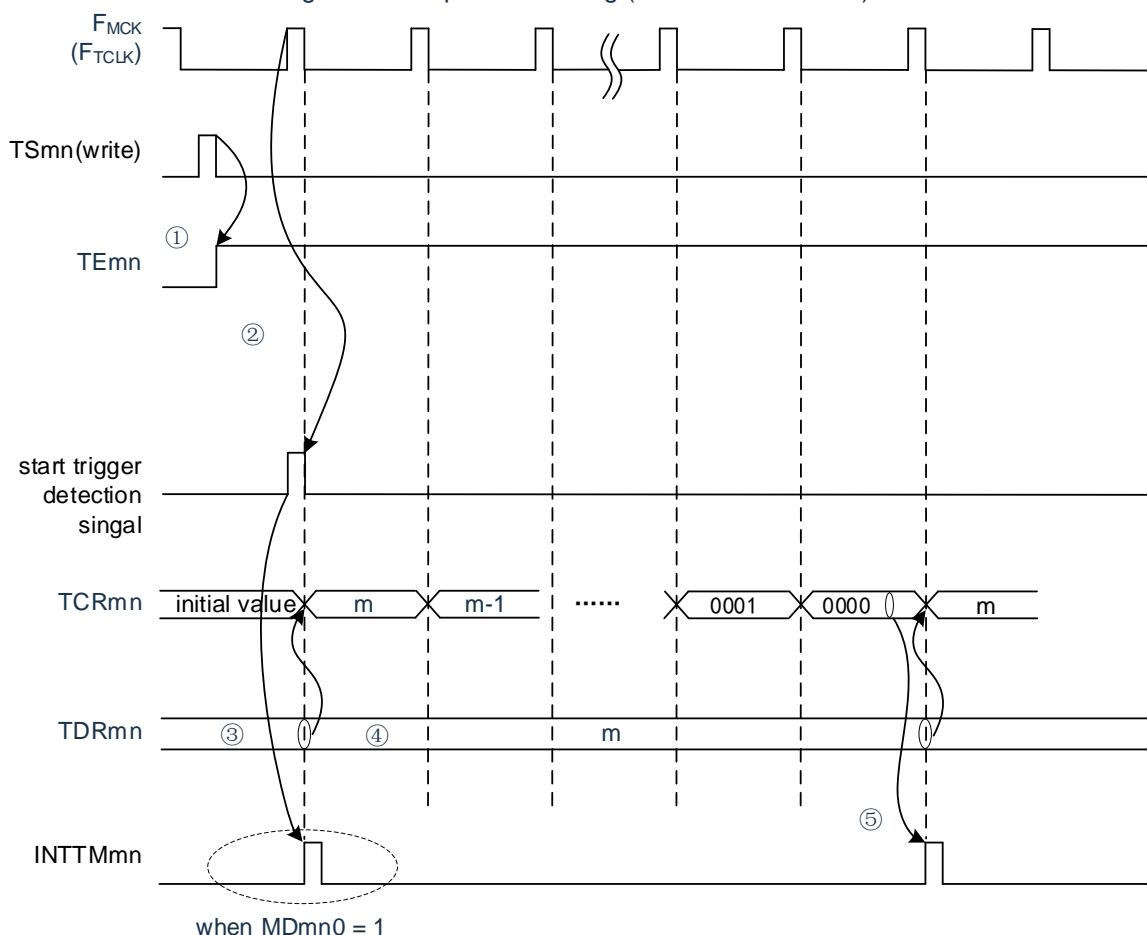
### 5.5.3 Operation of counters

The following describes the counter operation for each mode.

#### (1) Operation of the 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.

Figure5-26: Operation timing (interval timer mode)



Note: 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.

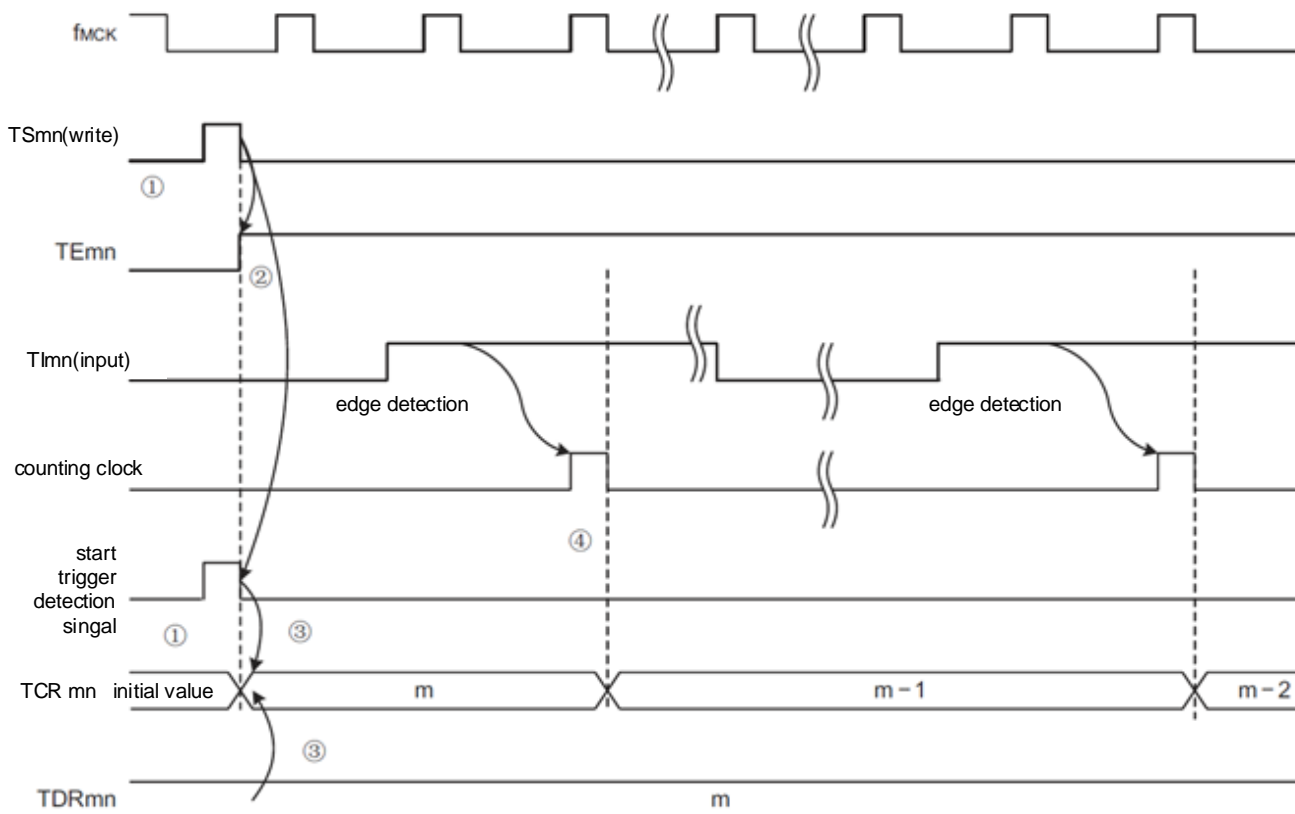
Remark:  $F_{MCK}$ , the start trigger detection signal and INTTMmn are synchronized with FCLK and are valid within 1 clock.



## (2) Operation of the event counter mode

- ① The timer count register mn (TCRmn) remains during the operation stop state (TEmn=0).
- ② The operation enable state is entered 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.

Figure5-27: Operation timing (event counter mode)

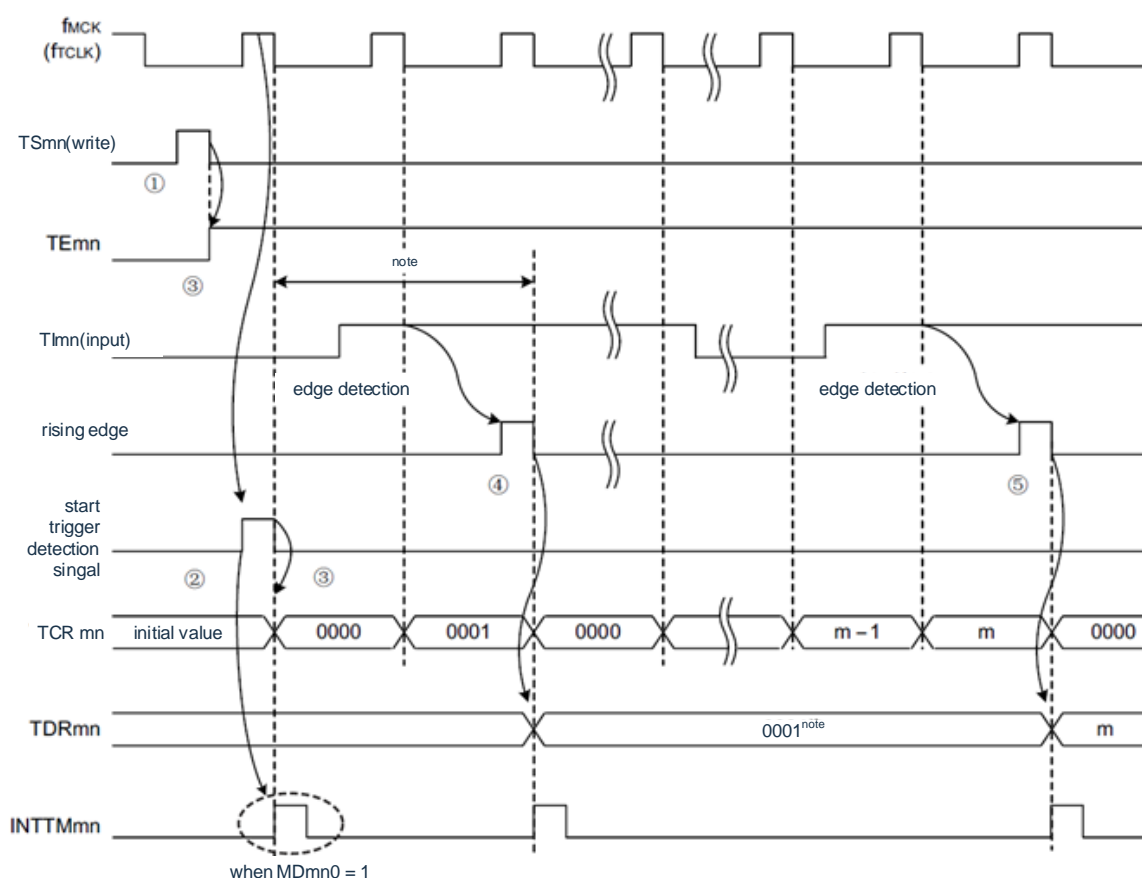


Remark: This is a timing without the noise filter. If the noise filter is used, the edge detection is delayed by 2 more  $F_{MCK}$  cycles (3~4 cycles in total) from the TImn input. The 1 cycle error is because the TImn input is not synchronized with the count clock ( $F_{MCK}$ ).

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

- ① The operation enable state is entered by writing "1" to the TSmn bit (TEmn=1).
- ② The timer count register mn (TCRmn) remains at its initial value until a count clock is generated.
- ③ A start trigger signal is generated by enabling the 1st count clock ( $F_{MCK}$ ) after the operation. Then, the "0000H" is loaded into the TCRmn register and counting starts in capture mode (INTTMmn is generated by the start trigger signal when MDmn0 bit is "1").
- ④ If an active edge of TImn input is detected, the value of TCRmn register is captured to TDRmn register and INTTMmn interrupt is generated. The capture value is meaningless at this point. The TCRmn register continues counting from the "0000H".
- ⑤ If an active edge of the next TImn input is detected, the value of the TCRmn register is captured to the TDRmn register and the INTTMmn interrupt is generated.

Figure5-28: 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.

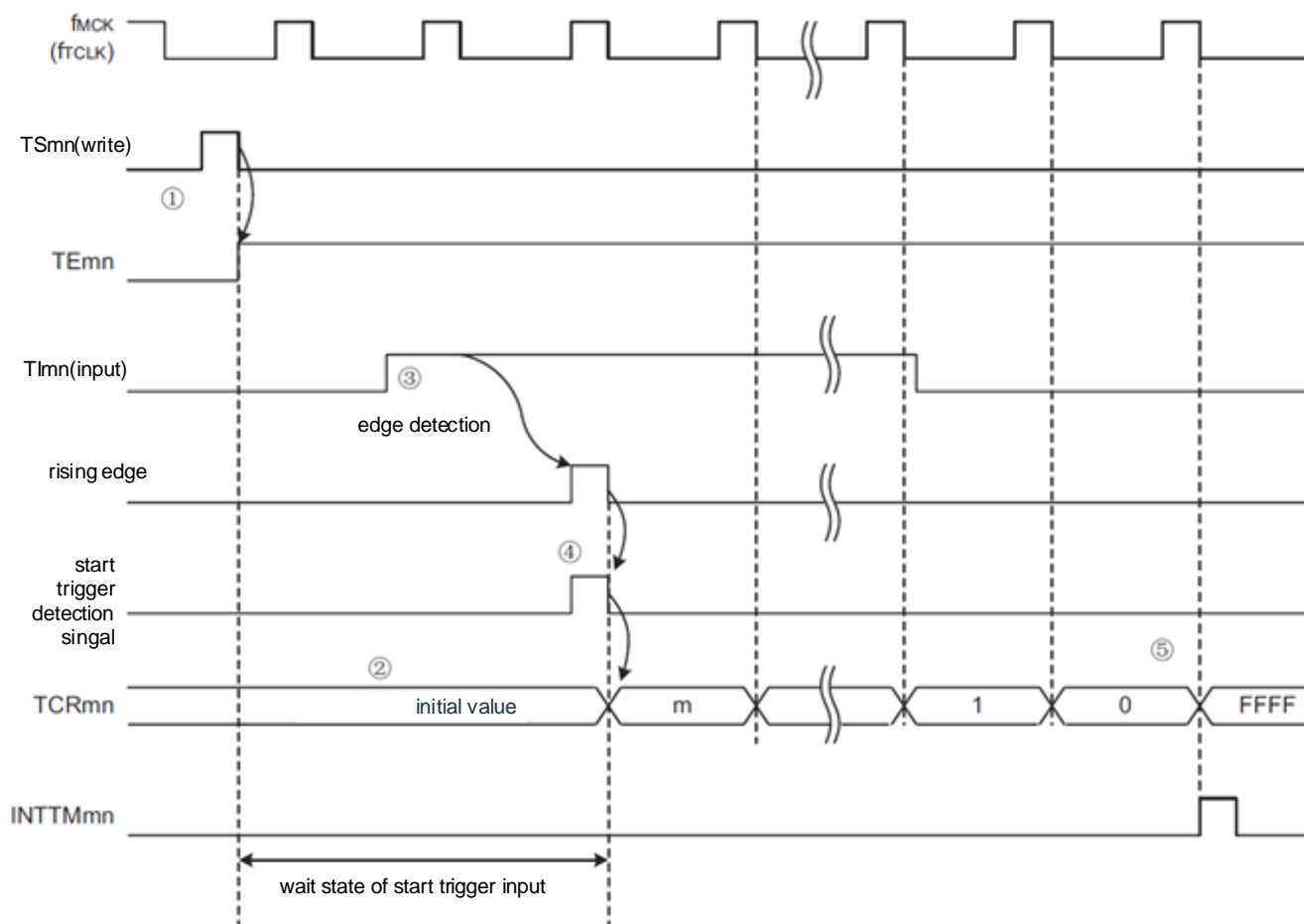
**Notice:** Because the 1st count clock cycle runs after the TSmn bit is written and delays the start of counting before generating the count clock, an error of up to 1 clock cycle is generated. Also, if you need information about the start of the count timing, set MDmn0 to "1" so that an interrupt can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

#### (4) Operation of the 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.

Figure5-29: Operation timing (single count mode)

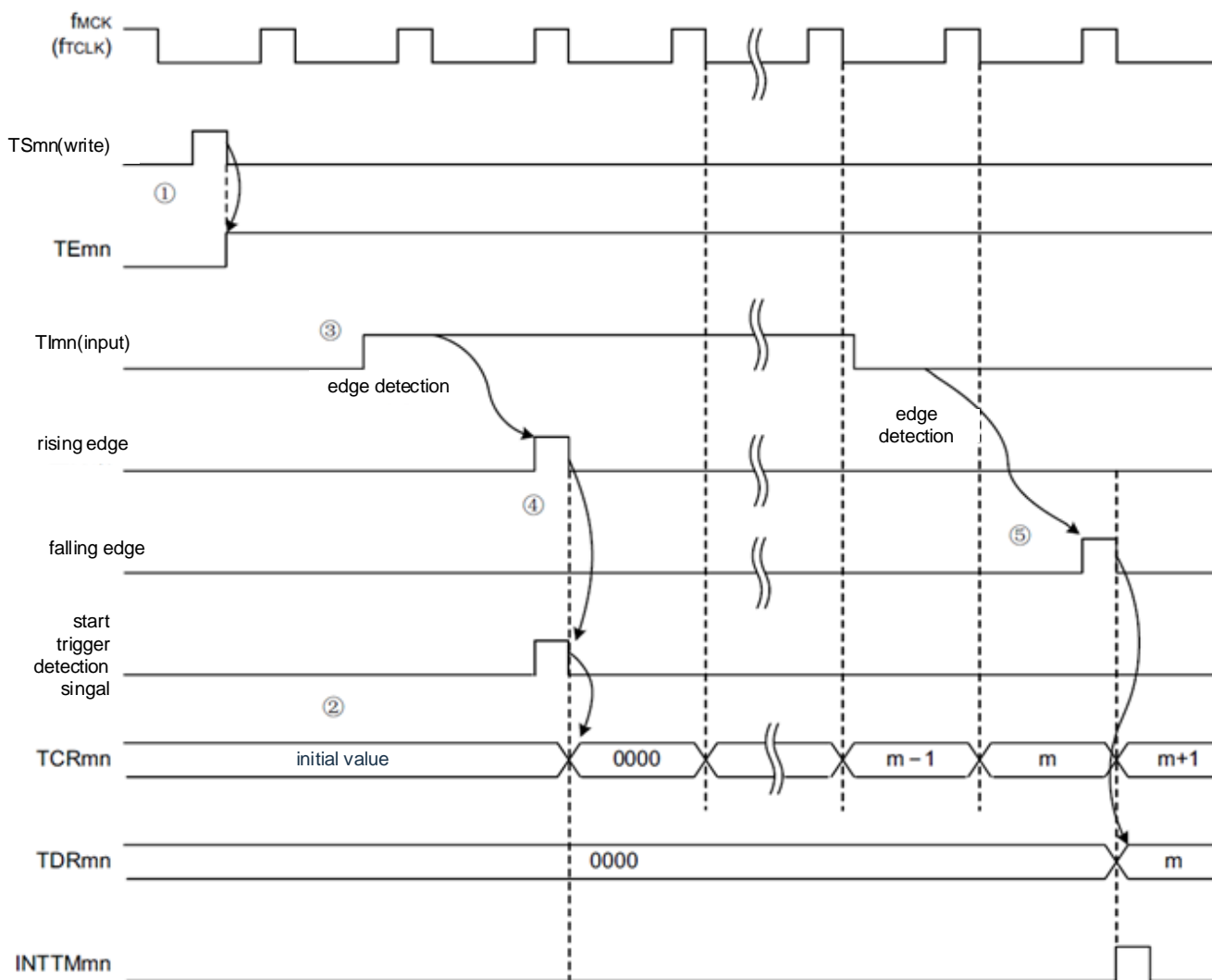


Remark: This is a timing without the noise filter. If the noise filter is used, the edge detection is delayed by 2 more  $F_{MCK}$  cycles (3~4 cycles in total) from the  $TImn$  input. The 1 cycle error is because the  $TImn$  input is not synchronized with the count clock ( $F_{MCK}$ ).

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

- ① The operation enable state is entered by writing "1" to the TSmn bit of the timer channel start register m (TSM)(TEmn=1).
- ② The timer count register mn (TCRmn) remains the initial value until a start trigger signal is generated.
- ③ Detects the rising edge of the TImn input.
- ④ After the start trigger signal is generated, "0000H" is loaded into the TCRmn register and counting starts.
- ⑤ If the falling edge of TImn input is detected, the value of the TCRmn register is captured to the TDRmn register and an INTTMmn interrupt is generated.

Figure5-30: Operation timing (capture & single count mode: measurement of high level width)

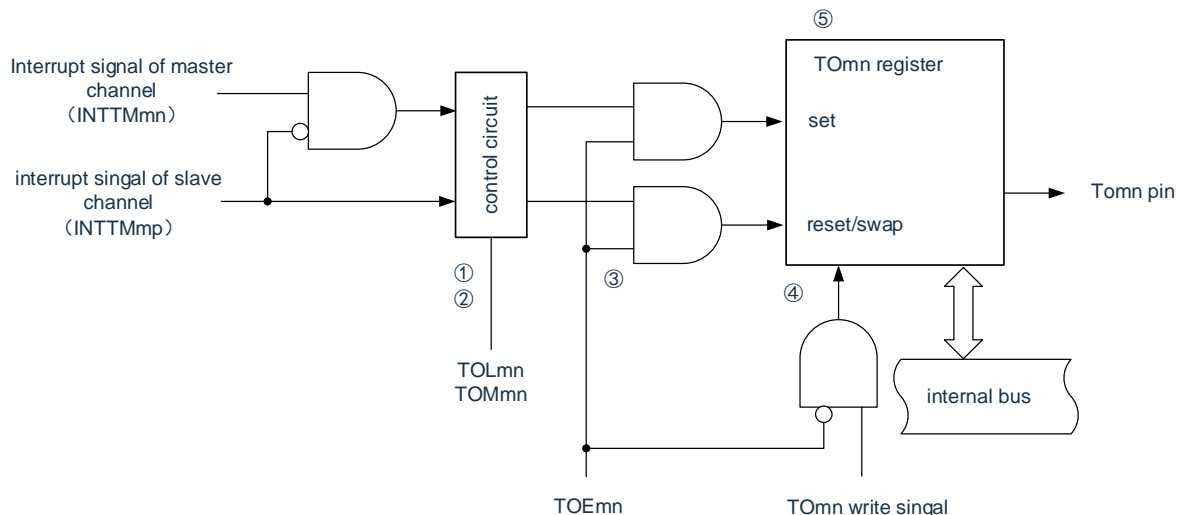


Remark: This is a timing without the noise filter. If the noise filter is used, the edge detection is delayed by 2 more  $F_{MCK}$  cycles (3~4 cycles in total) from the TImn input. The 1 cycle error is because the TImn input is not synchronized with the count clock ( $F_{MCK}$ ).

## 5.6 Control of channel outputs (TOmn pins)

### 5.6.1 Block diagram of the TOmn pin output circuit

Figure5-31: Block Diagram of the output circuit



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.  
In this case, the TOLm register is active and the following signals are controlled:  
When TOLmn=0: forward-direction operation (INTTMmn→set, INTTMmp→reset) When TOLmn=1: reverse-direction operation (INTTMmn→reset, INTTMmp→set)  
When both INTTMmn and INTTMmp are generated (0% output of the PWM output), priority is given to INTTMmp (reset signal) 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),

n: channel number, n=0~7 (master control channel: n=0, 2, 4, 6)

p: slave channel number

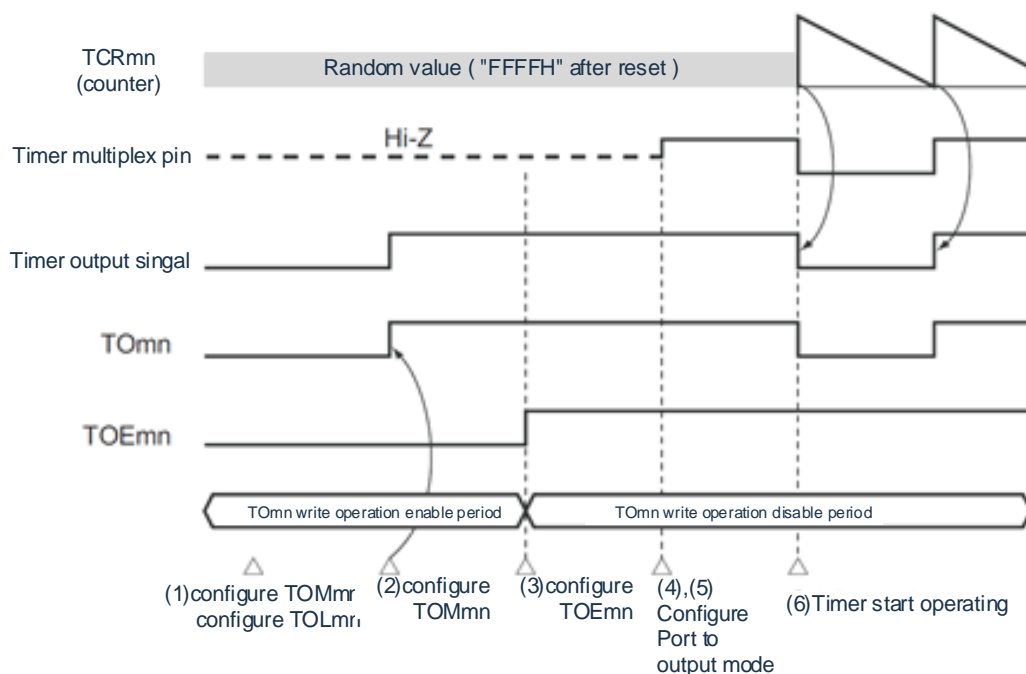
n=0: p=1, 2, 3

n=2: p=3

## 5.6.2 Settings of the TOMn pin output

The steps and state changes from the initial setting of the TOMn output pin to the start of timer operation are shown below.

Figure5-32: State change from the setting timer output to the start of operation



- ① Sets 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).
- ③ Writing "1" to TOEmn bit enables timer output (writing to TOM register is disabled).
- ④ The port is set to digital input/output via the Port Mode Control Register (PMCxx)(refer to "5.3.15 Registers for controlling timer input/output pin port functions").
- ⑤ Set the input/output of the port to output (refer to "5.3.15 Registers for controlling timer input/output pin port functions").
- ⑥ Enable timer operation (TSmn=1).

Remark: m: unit number (m=0) n: channel number (n=0~7)

### 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 Allow 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) n: channel number (n=0~7)

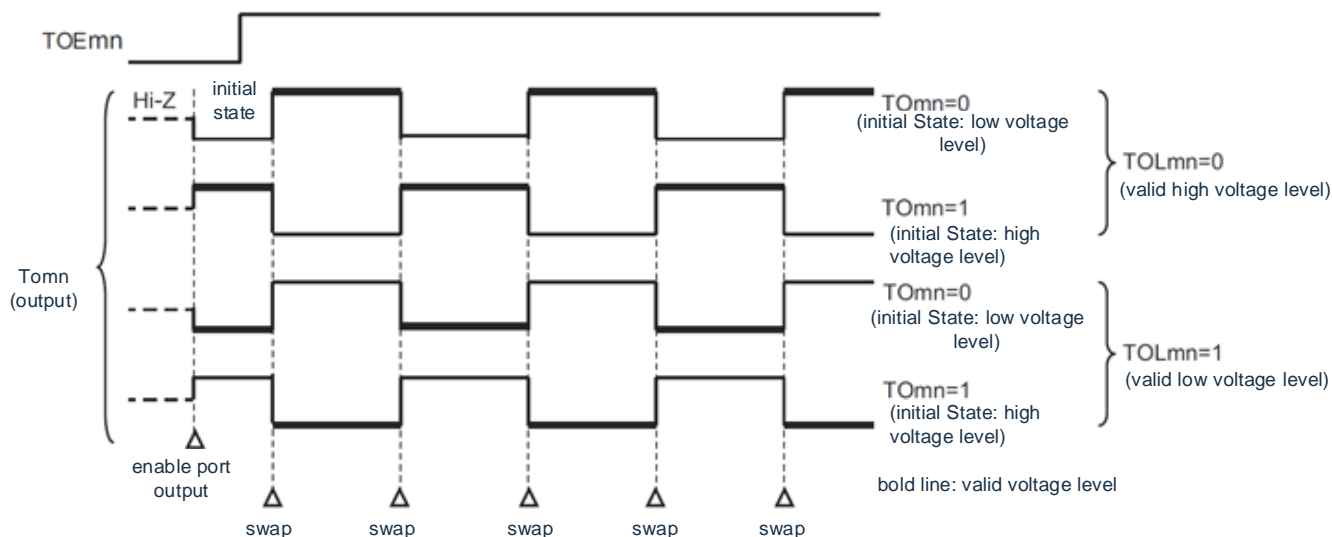
#### (2) Initial level for the TOMn pin and output level after the timer starts to operate

The timer output register m (TOM) is written before the port output is enabled and the timer output is disabled (TOEmn=0). The change of the TOMn pin output level when the initial level is set to the timer output enable state (TOEmn=1) is shown below.

##### (a) Operation start in master channel output mode (TOMmn=0)

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

Figure5-33: Output state of TOMn pin at alternate output (TOMmn=0)



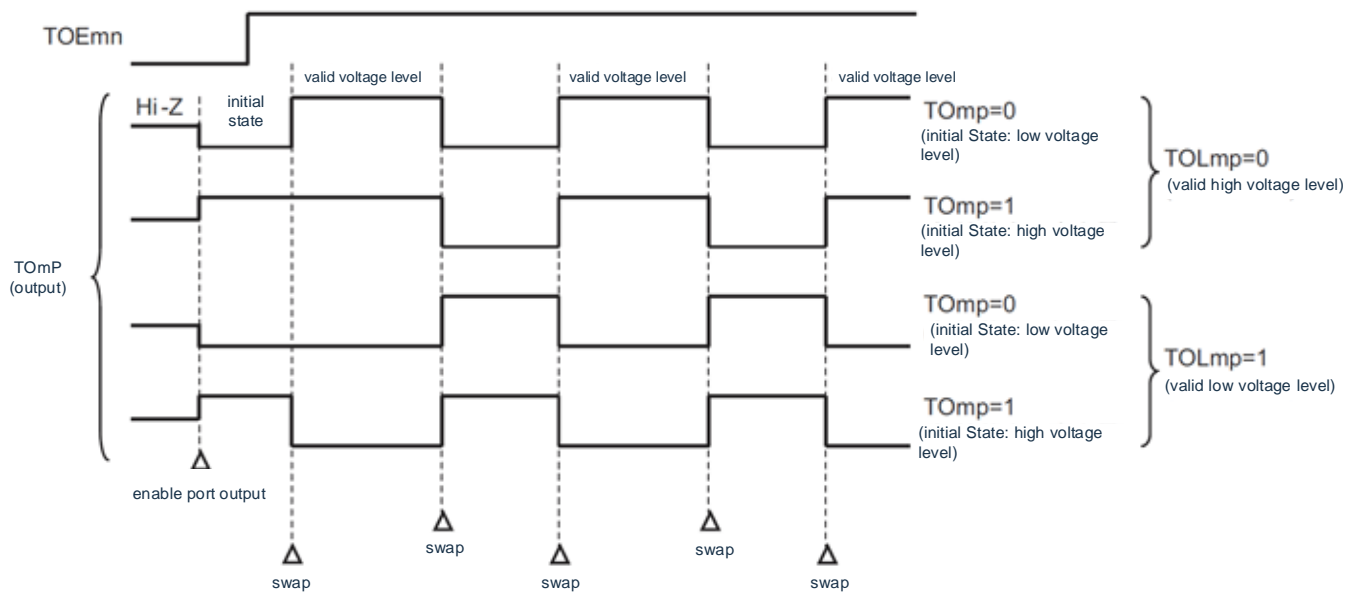
Remark:

1. Alternating: Output state of the inverted TOMn pin.
2. m: unit number (m=0) n: channel number (n=0~7)

##### (b) Operation start in slave channel output mode (TOMmn=1) (PWM output)

In slave channel output mode (TOMmn=1), the active level depends on the setting of Timer Output Level Register m (TOLmn).

Figure5-34: Output state of TOmn pin at PWM output (TOMmn=1)



#### Remark:

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



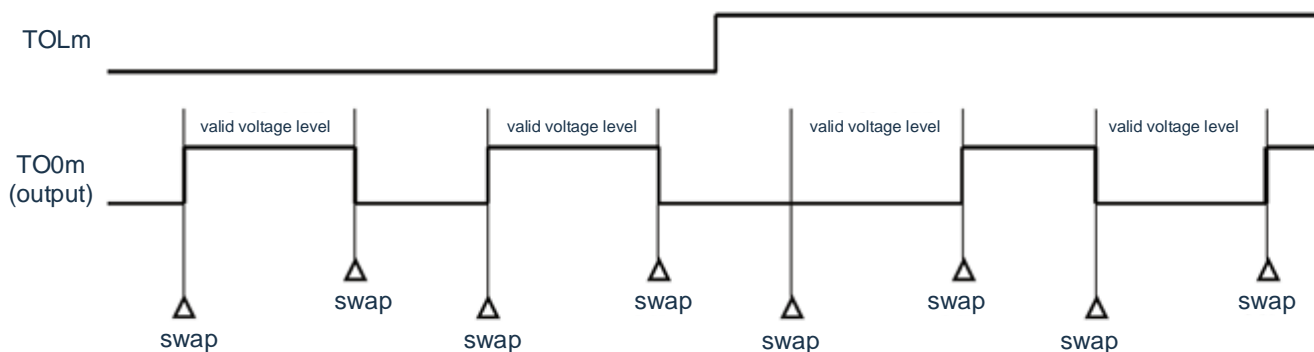
### (3) TOMn pin change for slave channel output mode (TOMmn=1)

#### (a) Settings of changing the timer output level register m (TOLm) during timer operation

If the setting of the TOLm register is changed during timer operation, the setting is valid when the TOMn pin change condition is generated. It is not possible to change the output level of the TOMn pin by rewriting the TOLm register.

When the TOMmn bit is "1", the operation when the value of the TOLm register is changed during timer operation (TEmn=1) is shown below.

Figure5-35: 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 an active level.  
Reset: The output signal from the TOMn pin changes from an active level to an invalid level.
2. m: unit number (m=0) n: channel number (n=0~7)

#### (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.

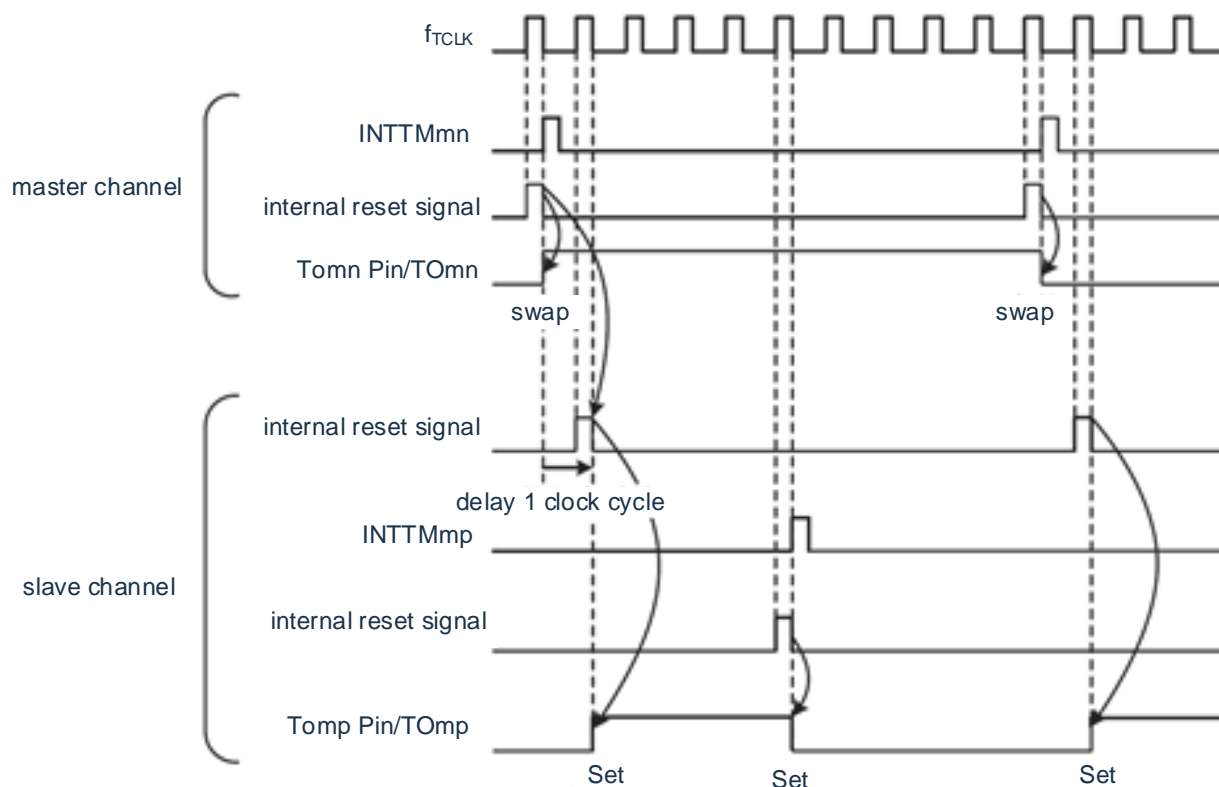
The set/reset operation status when setting the master/slave channel according to the following method is shown in Figure 5-35.

Main control channel: TOEmn=1, TOMmn=0, TOLmn=0

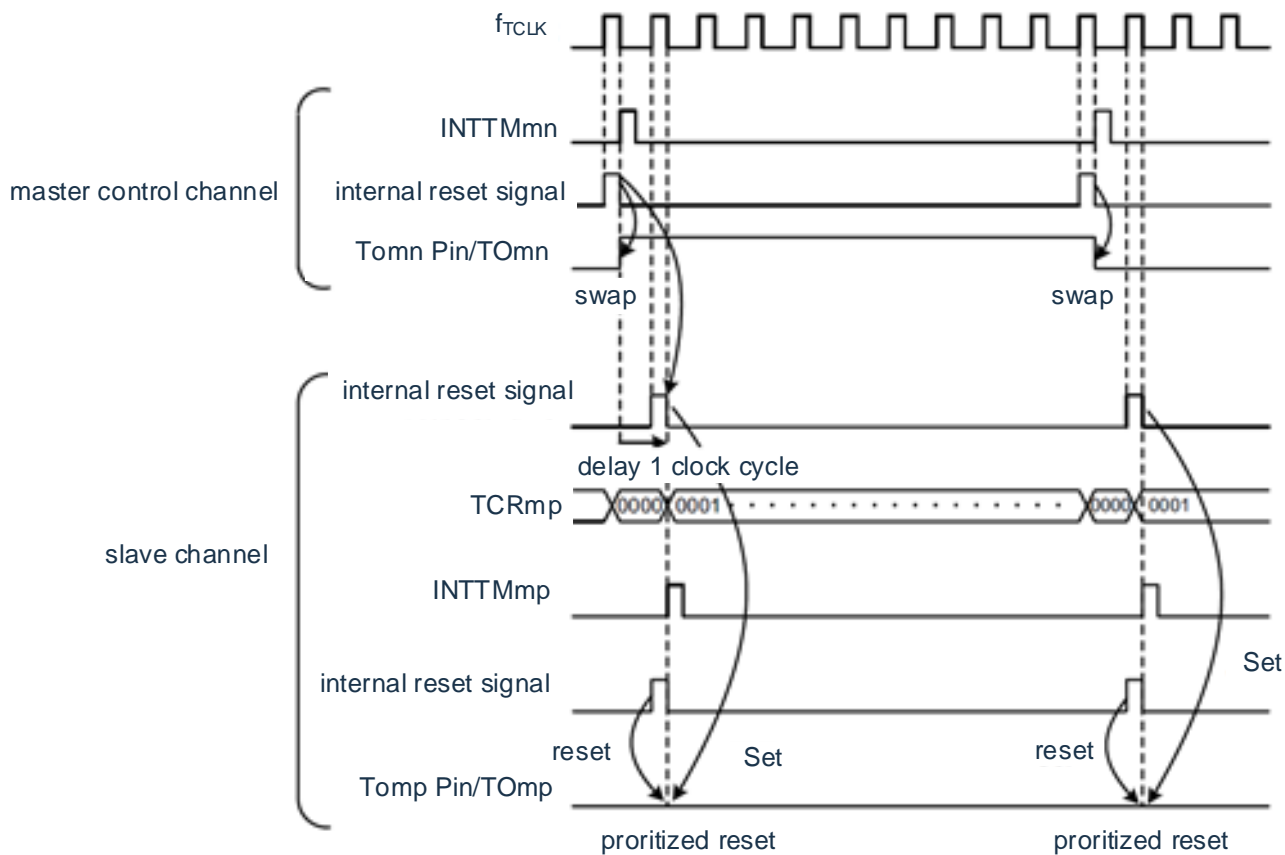
Slave channels: TOEmp=1, TOMmp=1, TOLmp=0

Figure5-36: Set/reset timing operation status

## (1) Basic operation timing



## (2) Operation timing with 0% duty cycle



## Remark:

1. Internal reset signal: reset/alternating signal on T0mn pin  
Internal set signal: set signal on T0mn pin
2. m: unit number (m=0)  
n: channel n=0~7 (master control channel: n=0, 2, 4, 6)  
p: slave channel number  
n=0: p=1, 2, 3  
n=2: p=3

## 5.6.4 One-time operation of TOn bit

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

Figure5-37: Example of a one-time operation with TOn bits

Before writing

TO0	0	0	0	0	0	0	0	0	0	0	0	TO0 3 1	TO0 2 0	TO0 1 1	TO0 0 0
TOE0	0	0	0	0	0	0	0	0	0	0	0	TOE03 0	TOE02 0	TOE01 0	TOE00 1

Data to be written

0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

After writing

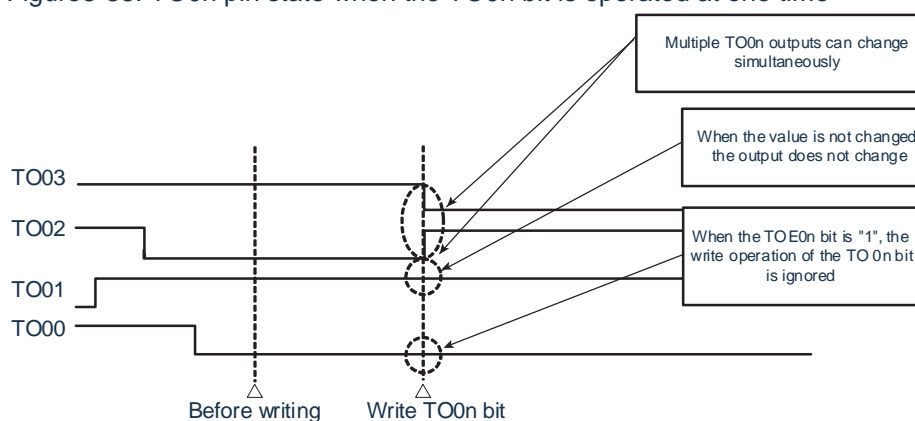
After writing

</

Only TOn bits with TOEmn bit "0" can be written, and TOn bits with "1" are ignored.

TOn (channel output) with bit "1" is not affected by the write operation, even if the TOn bit is written, the output change caused by the timer operation is carried out normally.

Figure5-38: TOn pin state when the TOn bit is operated at one time



Remark: m: unit number (m=0) n: channel number (n=0~7)

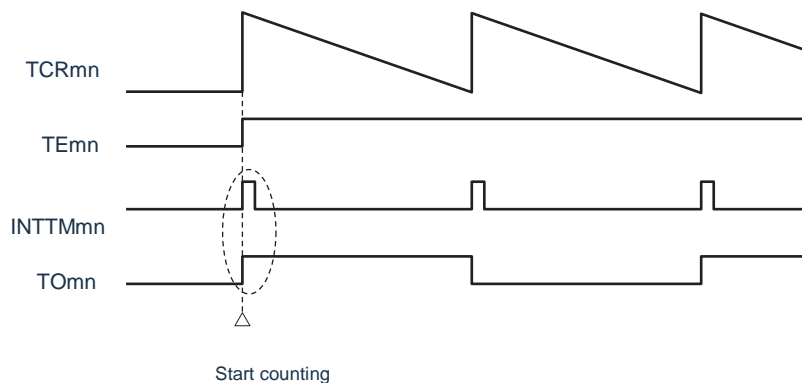
## 5.6.5 Timer interrupt and TOMn pin output when counting starts

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.

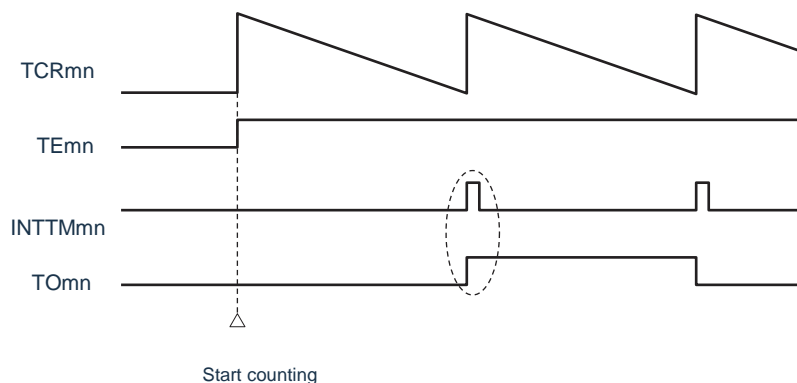
When the MDmn0 bit is "1", the start timing of the count can be known by generating a timer interrupt (INTTMmn). In other modes, the timer interrupt and TOMn output at the start of counting are not controlled. An example of operation when set to interval timer mode (TOEmn=1, TOMmn=0) is shown below

Figure5-39: An operation example of timer interrupt and TOMn output at start count

(a) MDmn0 bit is "1"



(b) MDmn0 bit is "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.

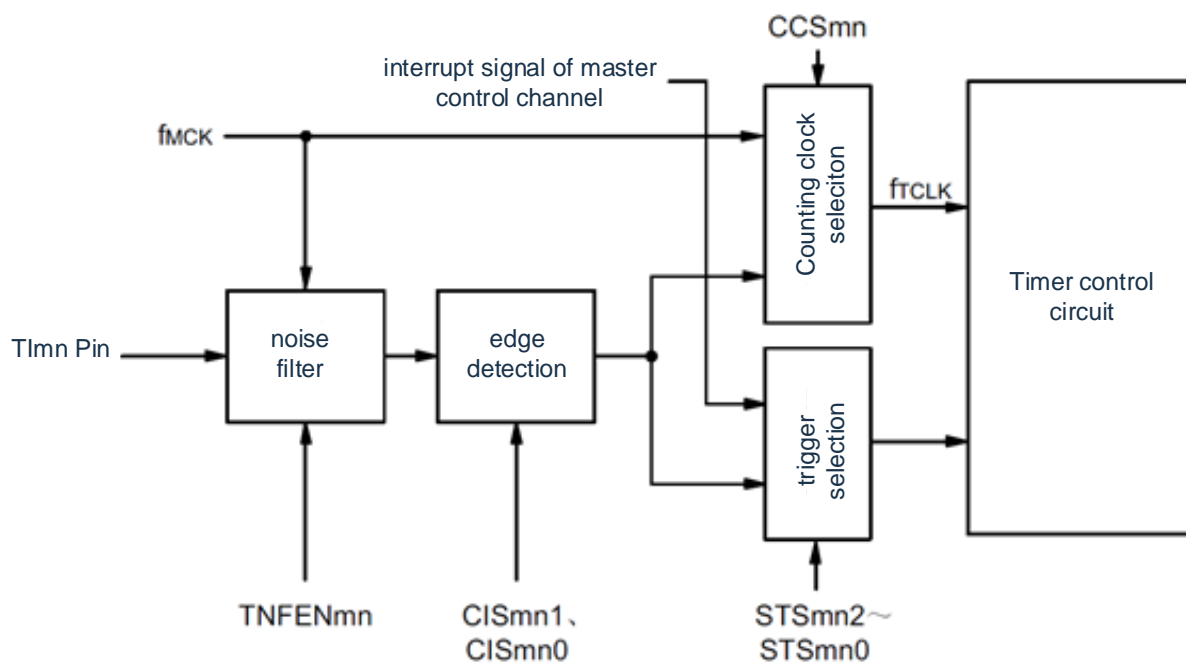
Remark: m: unit number (m=0) n: channel number (n=0~7)

## 5.7 Control of Timer Input (TImn)

### 5.7.1 Block diagram of the 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 eliminated from noise, the corresponding pin noise filter must be set to enable. The block diagram of the input circuit is as follows.

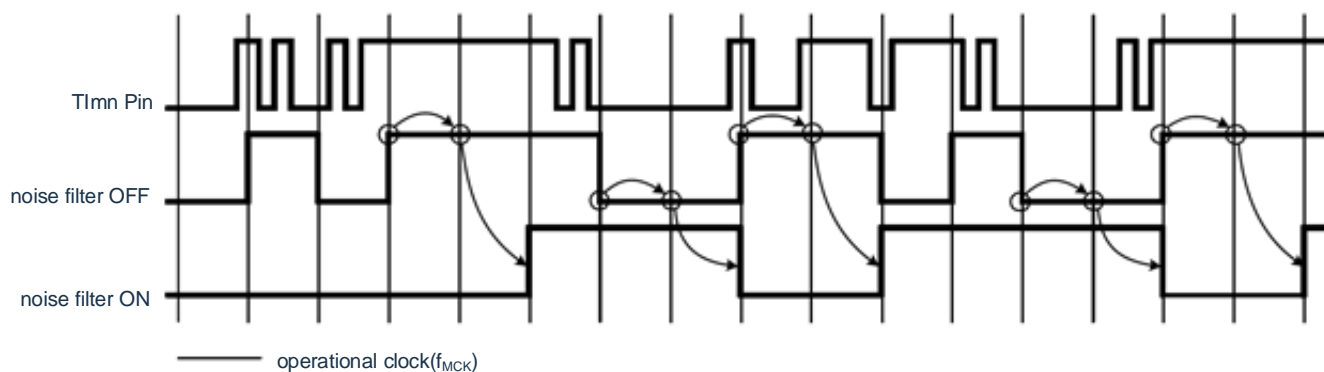
Figure5-40: Block Diagram of the input circuit



### 5.7.2 Noise filter

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

Figure5-41: 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. In realistic case, the input must be made in accordance with the TImn input high and low level width shown in "AC Characteristics".

### 5.7.3 Cautions for 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

If any of the bits 12 (CCSmn), 9 (STSmn1) and 8 (STSmn0) of the timer mode register mn (TMRmn) are all "0", the timer channel start must be set after at least 2 cycles of the operation clock ( $F_{MCK}$ ).

(2) Noise filter ON

If any of the bits 12 (CCSmn), 9 (STSmn1) and 8 (STSmn0) of the timer mode register mn (TMRmn) are all "0", the timer channel start must be set after at least 4 cycles of the operation clock ( $F_{MCK}$ ).

## 5.8 Independent channel operation function for general purpose timer units

### 5.8.1 Operation as interval timer/square wave output

#### (a) Interval Timer

It can be used as a reference timer to generate INTTMmn (timer interrupt) at fixed intervals. The interrupt

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

generation period can be calculated using the following equation:

#### (b) 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 TOMn output square wave can be calculated using the following equation:

- Square wave period of TOMn output = counting clock period  $\times$  (setting value of TDRmn + 1)  $\times$  2

- Square wave frequency of TOMn output = counting clock frequency / {setting value of TDRmn + 1}  $\times$  2}

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

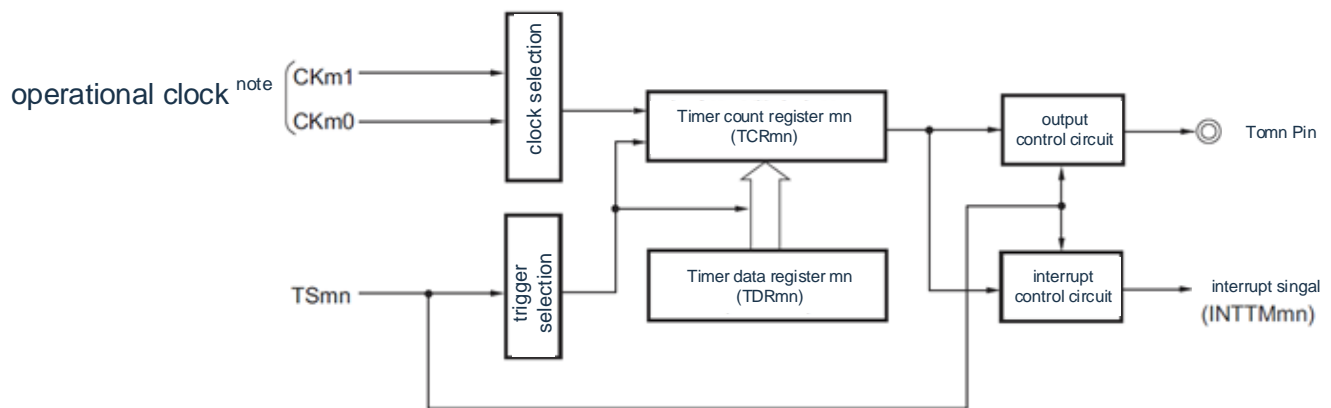
After setting the channel start trigger bit (TSMn) of the timer channel start register m (TSM) to "1", the value of timer data register mn (TDRmn) is loaded into the TCRmn register by the first count clock. At this time, if the MDmn0 bit of the timer mode register n (TMRmn) is "0", INTTMmn is not output 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.

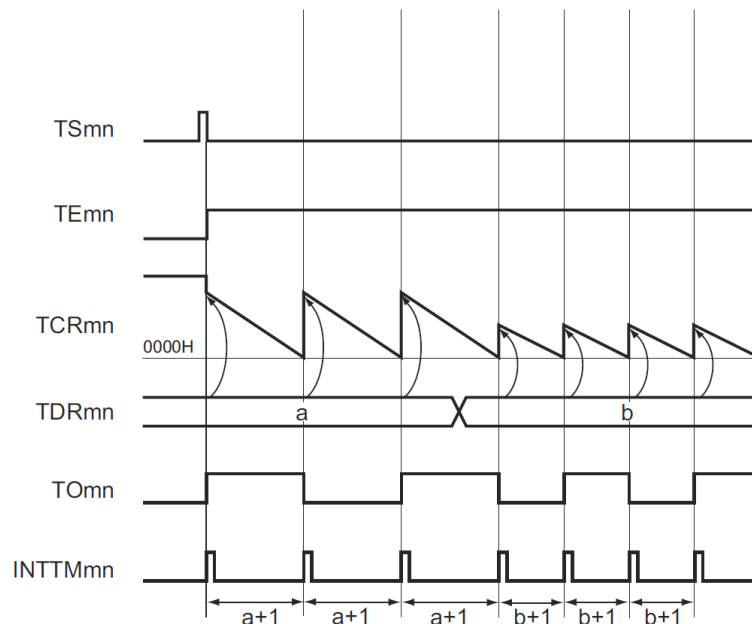


Figure5-42: Example of basic timing operating as an interval timer/square wave output (MDmn0=1)



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

Figure5-43: Example of basic timing operating as an interval timer/square wave output (MDmn0=1)

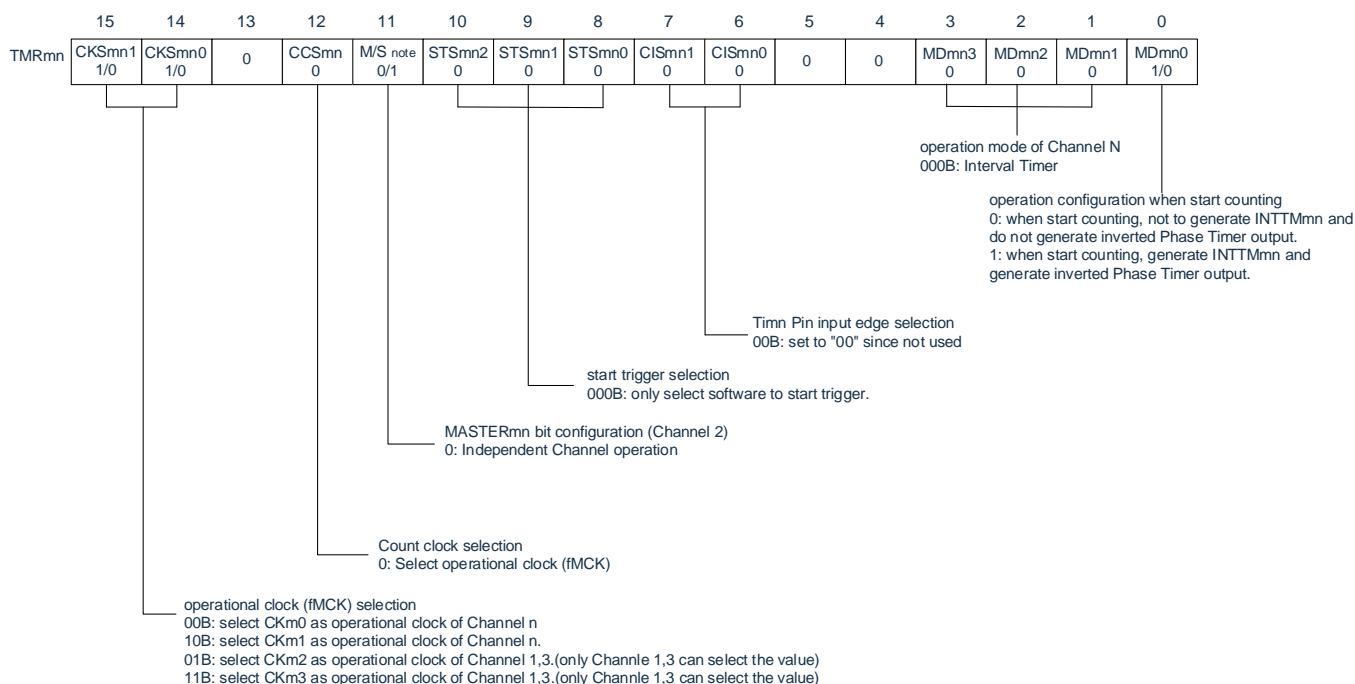


Remark:

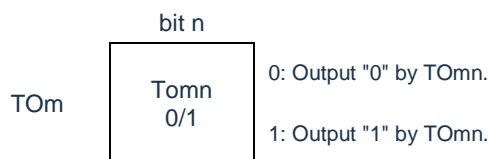
1. m: unit number (m=0) n: channel number (n=0~7)
2. TSmn: bit n of timer channel start register m (TSM)
- TEmn: bit n of timer channel enable status register m (TEM)
- TCRmn: timer count register mn (TCRmn)
- TDRmn: timer data register mn (TDRmn)
- TOMn: TOMn pin output signal

Figure5-44: Example of register setting contents for interval timer/square wave output

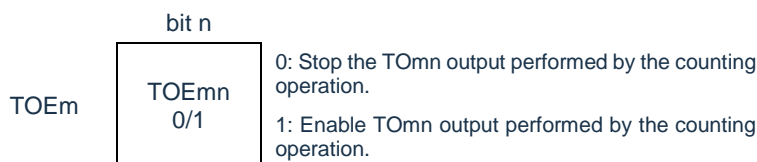
## (a) Timer mode register mn (TMRmn)



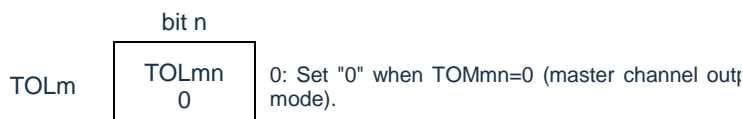
## (b) Timer output register m (TOM)



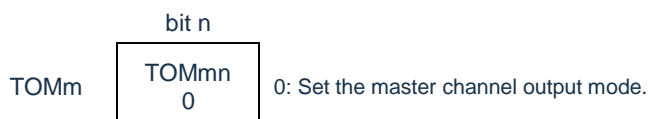
## (c) Timer output enable register m (TOEm)



## (d) Timer output level register m (TOLm)



## (e) Timer output mode register m (TOMm)



Note: TMRm2: MASTERmn bit

TMRm0: Fixed to "0".

Remark: m: unit number (m=0) n: channel number (n=0~7)

Figure5-45: Procedure for the interval timer/square wave output function

	Software operation	Hardware status
TAU initial settings		The input clock of timer unit m is in the stop-providing state. (Stop providing clock, cannot write to each register)
	Set the TM80EN bit of peripheral enable register 0(PER0) to "1".	The input clock of timer unit m is in the providing state. (Start providing clock, can write to each register)
	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3.	
Initial setting of the channels	Set the timer mode register mn (TMRmn) (to determine the channel's operation mode). Set the interval (period) value for the timer data register mn (TDRmn).	The channel is in the stop state. (Provides clock, and consumes some power)
	Using TOMn output: Set the TOMmn bit of Timer Output Mode Register m (TOMm) to "0" (master channel output mode). Set the TOLmn bit to "0". Set the TOMn bit to determine the initial level of the TOMn output. Set the TOEmn bit to "1" and enable TOMn output. Set the Port Register and Port Mode Register to "0".	The TOMn pin is in Hi-Z output state.  When the port mode register is in output mode and the port register is "0", the TOMn initial set level is output. The TOMn remains unchanged because the channel is in the stop state. The TOMn pin outputs the level set by the TOMn.
Start Operate	(The TOEmn bit set to "1" only when the TOMn output is used and restarted) Set the TSmn bit to "1". Since the TSmn bit is a trigger bit, it automatically returns to "0".	The TEMn bit becomes "1" and starts counting. Load the value of the TDRmn register into the Timer Count Register mn (TCRmn). When the MDmn0 bit of TMRmn register is "1", INTTMmn is generated and TOMn is output alternately
In operation	The setting of the TDRmn register can be changed at will. The TCRmn register can be read at any time. The TSRmn register is not used. The TOM register and TOEm register settings can be changed. The setting of the TMRmn register, the TOMmn bit and the TOLmn bit cannot be changed.	The counter (TCRmn) performs decremental counting. If the count reaches "0000H", the value of the TDRmn register is loaded into the TCRmn register again and the count continues. When TCRmn is detected as "0000H", INTTMmn is generated and TOMn is alternately output. Thereafter, repeat this operation.
Stop Operation	Set the TTmn bit to "1". The operation automatically returns to "0" because the TTmn bit is a trigger bit.	The TEMn bit becomes "0" and stops counting. The TCRmn register holds the count value and stops counting. The TOMn output is not initialized but remains its state.
	Set the TOEmn bit to "0" and set the value for the TOMn bit.	The TOMn pin outputs the level set by the TOMn bit.
TAU Stop	To maintain the output level of the TOMn pin: Set TOMn bit to "0" after setting the value to be held for the port register. No need to maintain the output level of the TOMn pin: No need to set.	The output level of the TOMn pin is maintained by the port function.
	Set the TM80EN bit of the PER0 register to "0".	The input clock of timer unit m is in the stop-providing state. Initialize all circuits and the SFR for each channel. (TOMn bit becomes "0" and TOMn pin becomes port function)

Restart the operation

Remark: m: unit number (m=0) n: channel number (n=0~7)

## 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

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

the following equation:

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) of Timer Channel Start Register m (TSm) to "1".

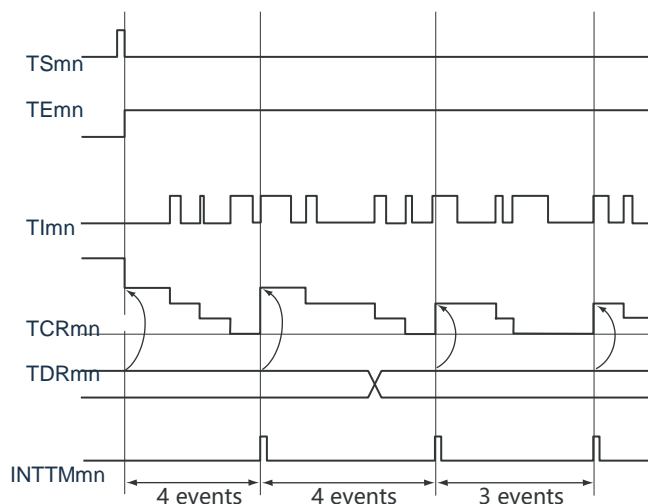
The TCRmn register decrements the count while detecting the active edge of the TImn pin input. If TCRmn becomes "0000H", the value of TDRmn register is loaded again and INTTMmn is output.

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.

Figure5-46: Example of basic timing operating as an external event counter

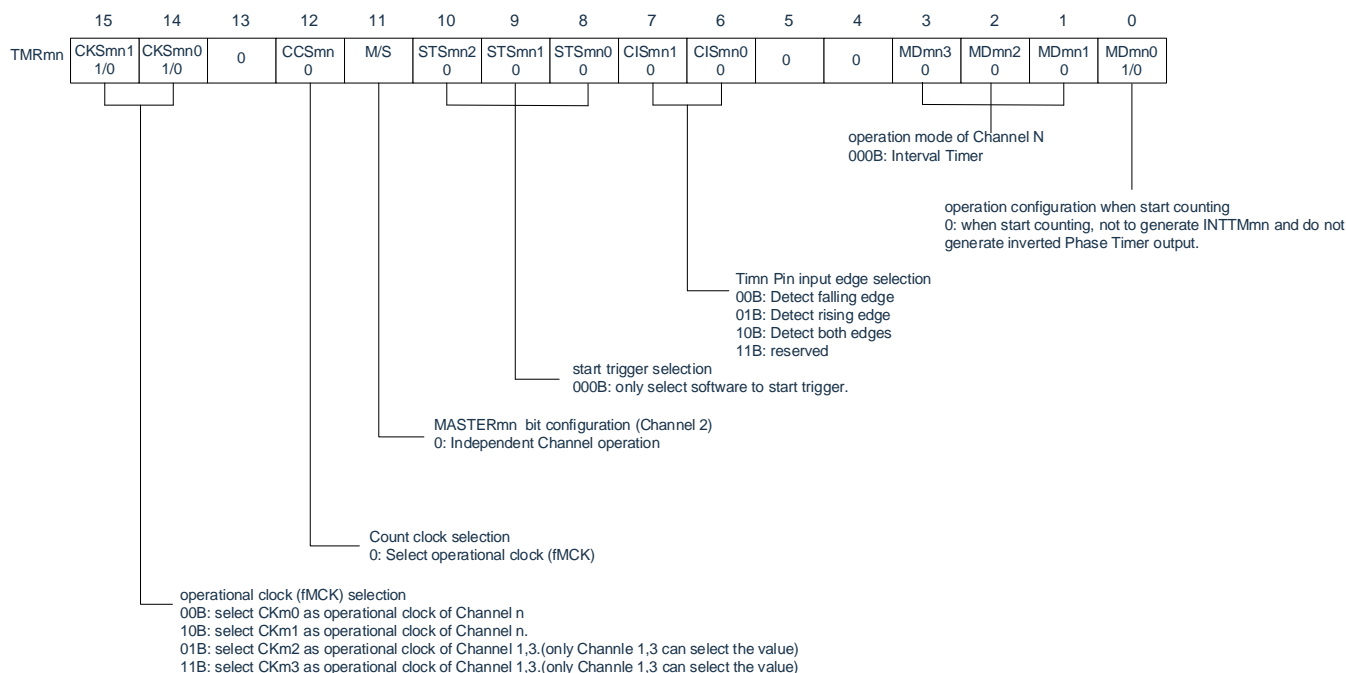


Remark:

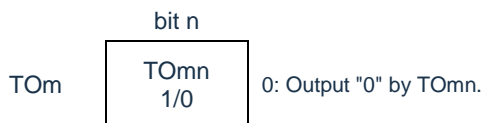
1. m: unit number (m=0) n: channel number (n=0~7)
2. TSmn: bit n of timer channel start register m (TSm)  
TE mn: bit n of timer channel enable status register m (TEm)  
TImn: TImn pin input signal  
TCRmn: timer count register mn (TCRmn)  
TDRmn: timer data register mn (TDRmn)

Figure5-47: Example of register contents setting in external event counter mode

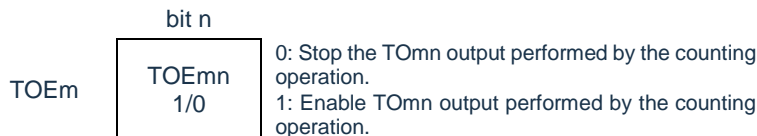
## (a) Timer mode register mn (TMRmn)



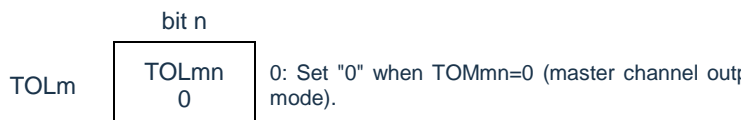
## (b) Timer output register m (TOM)



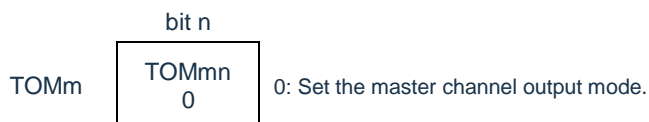
## (c) Timer output enable register m (TOEm)



## (d) Timer output level register m (TOLm)



## (e) Timer output mode register m (TOMm)



Note: TMRm2, TMRm4, TMRm6: MASTERmn bit

TMRm0, TMRm5, TMRm7: Fixed to "0".

Remark: m: unit number (m=0) n: channel number (n=0~7)

Figure5-48: Procedure for the external event counter function

	Software operation	Hardware status
Timer8 initial settings		The input clock of timer unit m is in the stop-providing state. (Stop providing clock, cannot write to each register)
	Set the TM80EN bit of peripheral enable register 0(PER0) to "1".	The input clock of timer unit m is in the providing state and the channels are in the stop state. (Start providing clock, can write to each register)
	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3.	
Initial setting of the channels	Set the corresponding bit of the Noise Filter Enable Register (NFEN1) to "0" (OFF) or "1" (ON). Set the timer mode register mn (TMRmn)(to determines the operating mode of the channel). Set the count value for the timer data register mn (TDRmn). Set the TOEmn bit of the timer output enable register m (TOEm) to "0".	The channel is in the stop state. (Provides clock, and consumes some power)
Start operating	Set the TSmn bit to "1". Since the TSmn bit is a trigger bit, it automatically returns to "0".	The TEMn bit becomes "1" and starts counting. The value of the TDRmn register is loaded into the timer count register mn (TCRmn) and enter the detection wait state of the input edge of the TImn pin.
In operation	The setting of the TDRmn register can be changed at will. The TCRmn register can be read at any time. The TSRmn register is not used. The setting of the TMRmn register, the TOMmn bit, the TOLmn bit, the Tomn bit and the TOEmn bit cannot be changed.	Whenever the input edge of the TImn pin is detected, the counter (TCRmn) is decremented. If the count reaches "0000H", the value of the TDRmn register is loaded into the TCRmn register again and the count continues. When TCRmn is detected as "0000H", INTTMmn is generated. Thereafter, repeat this operation.
Stop operating	Set the TTmn bit to "1". The operation automatically returns to "0" because the TTmn bit is a trigger bit.	The TEMn bit becomes "0" and stops counting. The TCRmn register holds the count value and stops counting.
Timer8 stop	Set the TM80EN 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) n: channel number (n=0~7)

### 5.8.3 Operation as frequency divider

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

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

- Select rising or falling edge:

$$\text{Divider clock frequency} = \text{input clock frequency} / \{(\text{setting value of TDRmn} + 1) \times 2\}$$

- Select rising and falling edges:

$$\text{Divider clock frequency} \approx \text{input clock frequency} / (\text{setting value of TDRmn} + 1)$$

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

After setting the channel start trigger bit (TSmn) of the timer channel start register (TSM) to "1", the value of the timer data register (TDRmn) is loaded into the TCRmn register by detecting an active edge of TImn. In this case, if the MDmn0 bit of the Timer Mode Register (TMRmn) is "0", no INTTMmn is output and TOmn is not alternately output. If the MDmn0 bit of the TMRmn register is "1", INTTMmn is output and TOmn is alternately output.

The TCRmn register is then decremented by counting the active edges of the TImn pin input. If TCRmn becomes "0000H", TOmn is alternately output. At the same time, the value of TDRmn register is loaded into TCRmn register and the count continues.

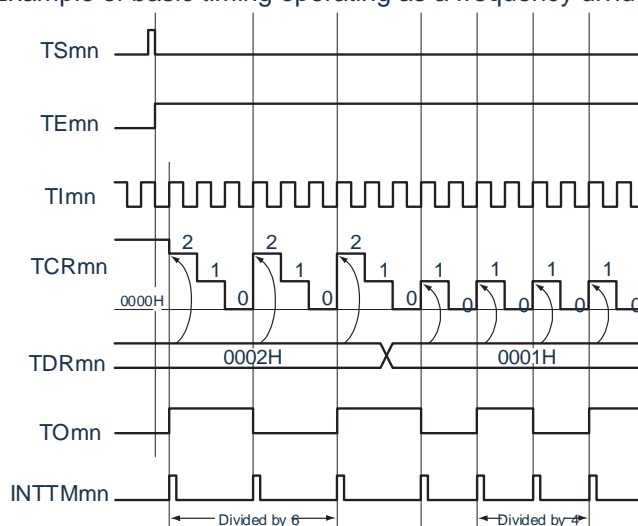
If detection of the both edges is selected for the TImn pin input, the duty cycle error of the input clock affects the divider clock period of the TOmn output.

The clock period of the TOmn output contains the sampling error of 1 operating clock period.

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

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

Figure5-49: Example of basic timing operating as a frequency divider (MDmn0=1)



Remark: TSmn: bit n of timer channel start register (TSM)

TE mn: bit n of timer channel enable status register TEM)

TImn: TImn pin input signal

TCRmn: timer count register (TCRmn)

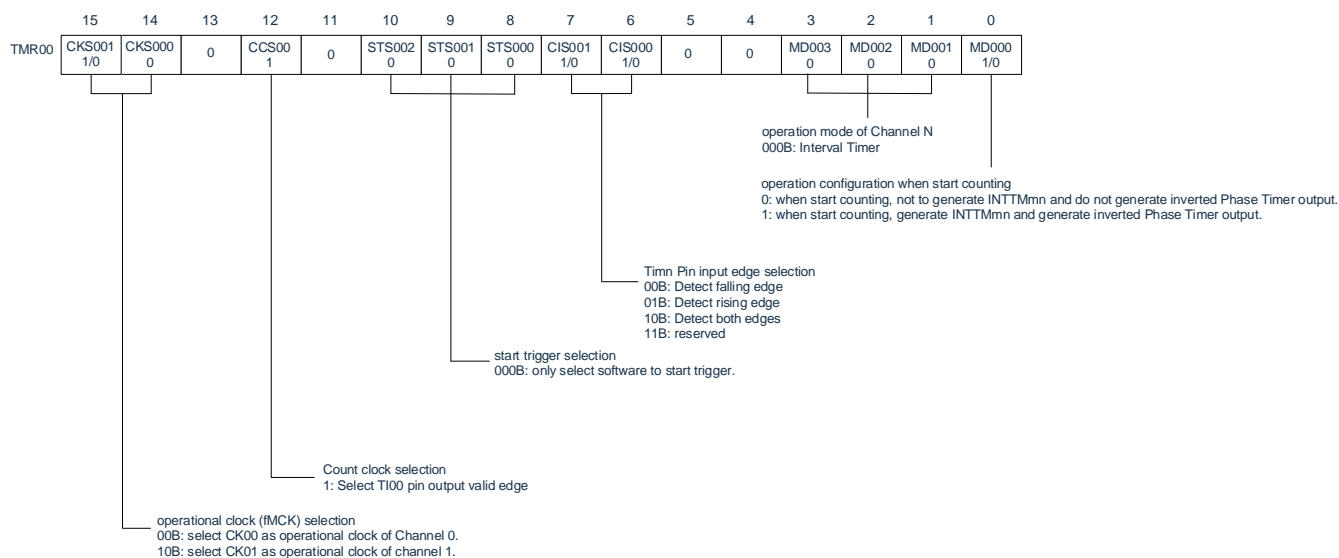
TDRmn: timer data register (TDRmn)

TOmn: TOmn pin output signal

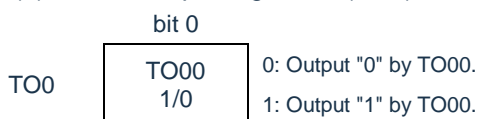
m: unit number (m=0) n: channel number (n=0~7)

Figure5-50: Example of register contents setting when operating as a frequency divider (channel 0 of unit 0)

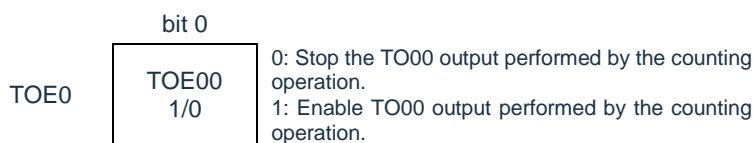
## (a) Timer mode register 00 (TMR00)



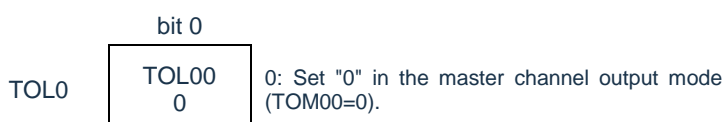
## (b) Timer output register 0 (TO0)



## (c) Timer output enable register 0 (TOE0)



## (d) Timer output level register 0 (TOL0)



## (e) Timer output mode register 0 (TOM0)

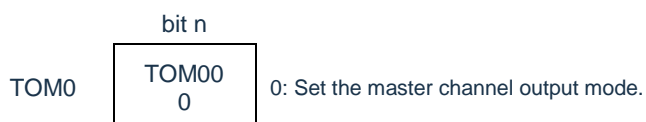




Figure5-51: Procedure for the frequency divider function

	Software operation	Hardware status
Timer8 initial settings		The input clock of timer unit 0 is in the stop-providing state. (Stop providing clock, cannot write to each register)
	Set the TM80EN bit of peripheral enable register 0(PER0) to "1".	The input clock of timer unit 0 is in the providing state and the channels are in the stop state. (Start providing clock, can write to each register)
	Set the timer clock selection register 0 (TPS0). Determine the clock frequency of CK00 ~ CK03.	
Initial setting of the channels	Set the corresponding bit of the Noise Filter Enable Register (NFEN1) to "0" (OFF) or "1" (ON). Set the timer mode register 00 (TMR00)(to determines the operating mode of the channel, select edge detection). Set the interval (cycles) value for the timer data register 00 (TDR00).	The channel is in the stop state. (Provides clock, and consumes some power)
	Set TOM00 bit of timer output mode register 0 (TOM0) to "0"(master control channel output mode). The TOL00 bit must be set to "0". Set the TO00 bit and determine the initial level of TO00 output Set TOE00 bit to "1" and enable TO00 output. Set the Port Register and Port Mode Register to "0"	The TO00 pin is in Hi-Z output state. When the port mode register is in output mode and the port register is "0", the TO00 initial set level is output. Since the channel is in stop state, TO00 changes. The TO00 pin outputs the level set by the TO00.
Start operating	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.
In operation	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.
Timer8 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 TM80EN bit of the PER0 register to "0".	The input clock of timer unit 0 is in the stop-providing state. Initialize all circuits and the SFR for each channel. (TO00 bit becomes "0" and TO00 pin becomes port function)

Restart Operation

## 5.8.4 Operation as input pulse interval measurement

The count value can be captured at the active edge of TImn and the interval between TImn input pulses can be measured. The software operation (TSmn=1) can also be set to capture the count value during the period when the TEMn bit is "1".

The pulse interval can be calculated using the following equation:

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

Note: Because the TImn pin input is sampled by the operating clock selected by the CKSmn bit of the timer mode register mn (TMRmn), an error of one operating clock is generated.

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.

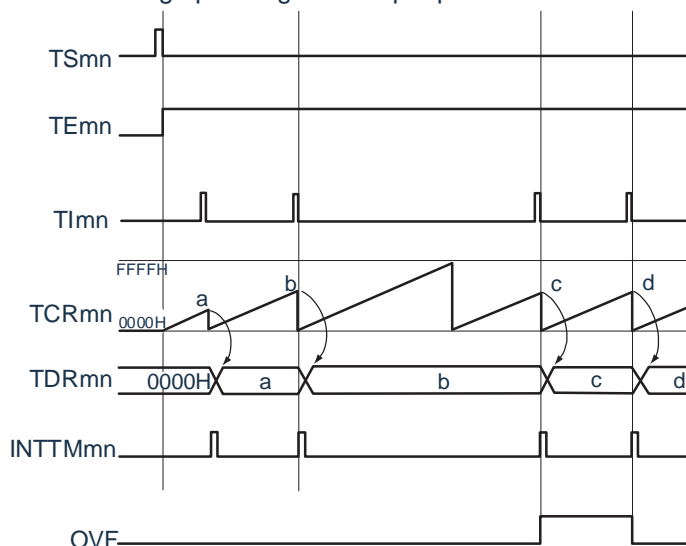
If the active edge of the TImn pin input is detected, the count value of TCRmn register is transferred (captured) to Timer Data Register mn (TDRmn), and the TCRmn register is cleared to "0000H", and then INTTMmn is output. If the counter overflows, the OVF bit of Timer Status Register mn (TSRmn) is set 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.

Figure5-52: Example of basic timing operating as an input pulse interval measurement (MDmn0=0)



Remark:

1. m: unit number (m=0) n: channel number (n=0~7)
2. TSmn: bit n of timer channel start register m (TSM)  
TEMn: bit n of timer channel enable status register (TEM)

TImn: TImn pin input signal

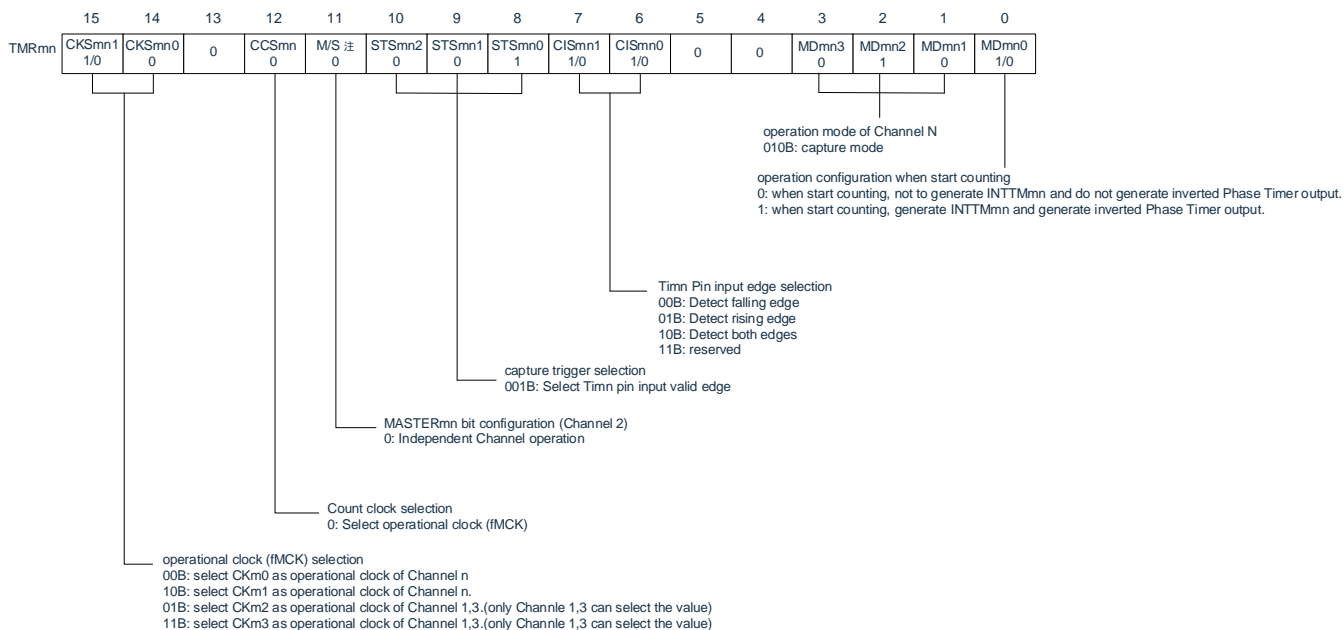
TCRmn: timer count register mn (TCRmn)

TDRmn: timer data register mn (TDRmn)

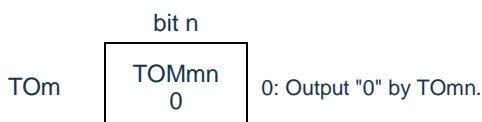
OVF: bit 0 of Timer Status Register mn (TSRmn)

Figure5-53: Example of register contents setting in measuring input pulse interval

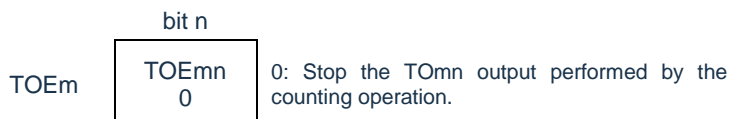
## (a) Timer mode register mn (TMRmn)



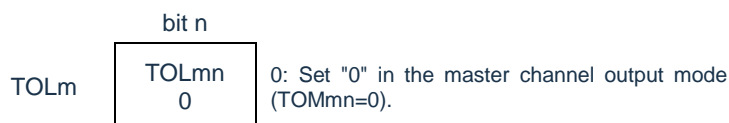
## (b) Timer output register m (TOM)



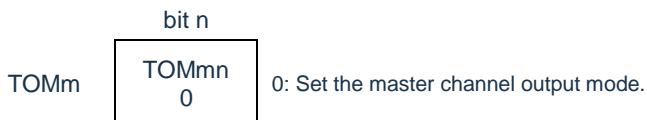
## (c) Timer output enable register m (TOEm)



## (d) Timer output level register m (TOLm)



## (e) Timer output mode register m (TOMm)



Note: TMRm2, TMRm4, TMRm6: MASTERmn bit

TMRm0, TMRm5, TMRm7: Fixed to "0".

Remark: m: unit number (m=0) n: channel number (n=0~7)

Figure5-54: Procedure for the input pulse interval measurement function

	Software operation	Hardware status
TAU initial settings		The input clock of timer unit m is in the stop-providing state. (Stop providing clock, cannot write to each register)
	Set the TM80EN bit of the peripheral enable register 0(PER0) to "1".	The input clock of timer unit m is in the providing state and the channels are in the stop state. (Start providing clock, can write to each register)
	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3.	
Initial setting of the 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 Operate	Set the TSmn bit to "1". Since the TSmn bit is a trigger bit, it automatically returns to "0".	The TEMn bit becomes "1" and starts counting. Clear the timer count register mn (TCRmn) to "0000H". When the MDmn0 bit of TMRmn register is "1", INTTMmn is generated.
In operation	The setting values of the CISmn1 bit and the CISmn0 bit of the TMRmn register can be changed. The TDRmn register can be read at any time. The TCRmn register can be read at any time. The TSRmn register can be read at any time. The setting of the the TOMmn bit, the TOLmn bit, the Tomn bit and the TOEmn bit cannot be changed.	The counter (TCRmn) starts incremental counting from "0000H" and transfers (captures) the count value to the timer data register mn (TDRmn) if it detects an active edge on the TImn pin input or sets TSmn bit to "1". If the counter overflows, set the OVF bit of Timer Status Register mn (TSRmn). If the counter does not overflow, the OVF bit is cleared. Thereafter, repeat this operation.
Stop Operation	Set the TTmn bit to "1". The operation automatically returns to "0" because the TTmn bit is a trigger bit.	The TEMn bit becomes "0" and stops counting. The TCRmn register holds the count value and stops counting. The OVF bit of the TSRmn register remains unchanged.
TAU Stop	Set the TM80EN 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) n: channel number (n=0~7)

## 5.8.5 Operation as input signal high and low level width measurement

Note: When used as a LIN-bus support feature, the bit1 (ISC1) of the input switching control register (ISC) must be set to "1", and in the instructions below, please use RxD0 Instead of TImn.

The signal width (high and 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

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

be calculated using the following equation:

Note: Because the TImn pin input is sampled by the operating clock selected by the CKSmn bit of the timer mode register mn (TMRmn), an error of one operating 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.

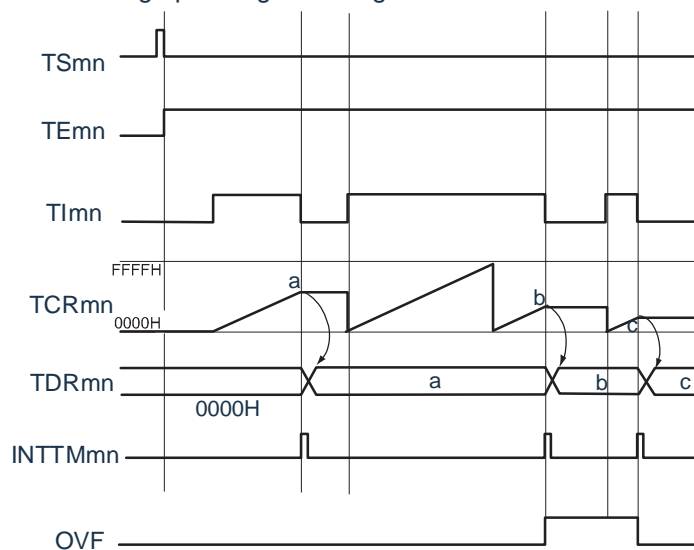
Even if the counter counts 2 or more complete cycles, the overflow is considered to have occurred and the OVF bit of the TSRmn register is set to "1". However, when two or more overflows occur, the interval value cannot be measured normally by the OVF bit.

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.

Figure5-55: Example of basic timing operating as an high and low level width measurement of input signal

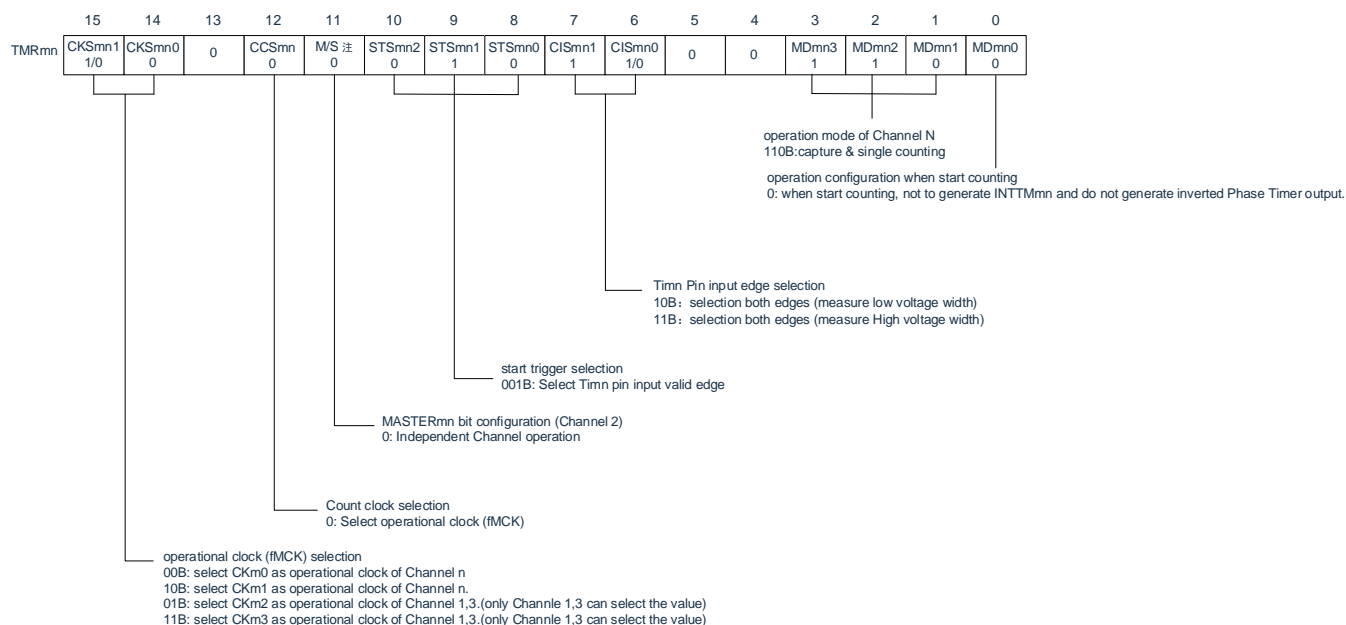


Remark:

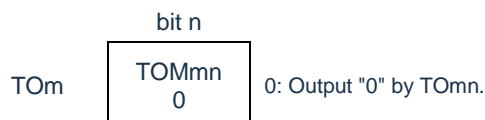
1. m: unit number (m=0) n: channel number (n=0~7)
2. TSmn: bit n of timer channel start register m (TSm)  
TE mn: bit n of timer channel enable status register TE m)  
Tl mn: Tl mn pin input signal  
TCR mn: timer count register mn (TCR mn)  
TDR mn: timer data register mn (TDR mn)  
OVF: bit0 of timer status register mn (TSR mn)

Figure5-56: Example of register contents setting in measuring high and low level width of input signal

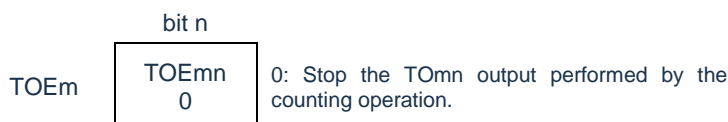
## (a) Timer mode register mn (TMRmn)



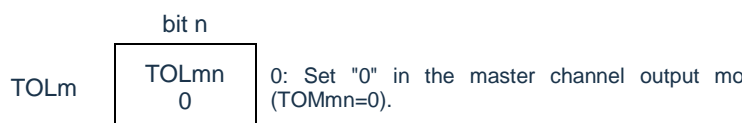
## (b) Timer output register m (TOM)



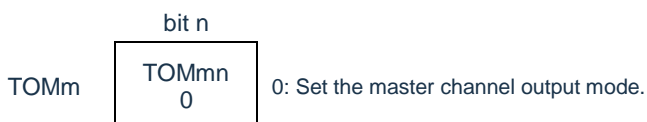
## (c) Timer output enable register m (TOEm)



## (d) Timer output level register m (TOLm)



## (e) Timer output mode register m (TOMm)



Note: TMRm2, TMRm4, TMRm6: MASTERmn bit

TMRm0, TMRm5, TMRm7: Fixed to "0".

Remark: m: unit number (m=0) n: channel number (n=0~7)



Figure5-57: Procedure for high and low level width measurement function of input signal

	Software operation	Hardware status
Timer8 initial settings		The input clock of timer unit m is in the stop-providing state. (Stop providing clock, cannot write to each register)
	Set the TM8mEN bit of the peripheral enable register 0(PER0) to "1".	The input clock of timer unit m is in the providing state and the channels are in the stop state. (Start providing clock, can write to each register)
	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3.	
Initial setting of the 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 TOEmn bit to "0" and stop TOMn operation.	The channel is in the stop state. (Provides clock, and consumes some power)
Start Operate	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 enters the start edge detection wait state of the TImn pin.
	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.
Timer8 Stop	Set the TM8mEN 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 the operation

Remark: m: unit number (m=0) n: channel number (n=0~7)

## 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 INTTMMmn (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 INTTMMmn (timer interrupt) at any set interval.

The interrupt generation period can be calculated using the following equation:

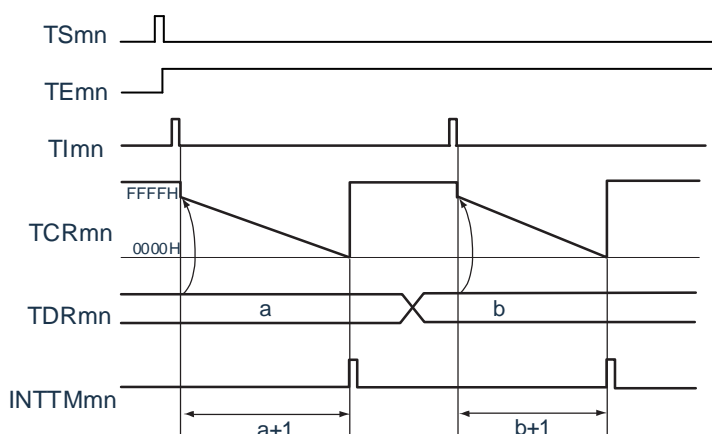
$$\text{INTTMMmn (timer interrupt) generation period} = \text{counting clock period} \times (\text{setting value of TDRmn} + 1)$$

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

If the channel start trigger bit (TSmn) of the timer channel start register m(TSm) is set to "1", the TEMn7 bit becomes "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", INTTMMmn 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.

Figure5-58: Example of basic timing operating as a delay counter

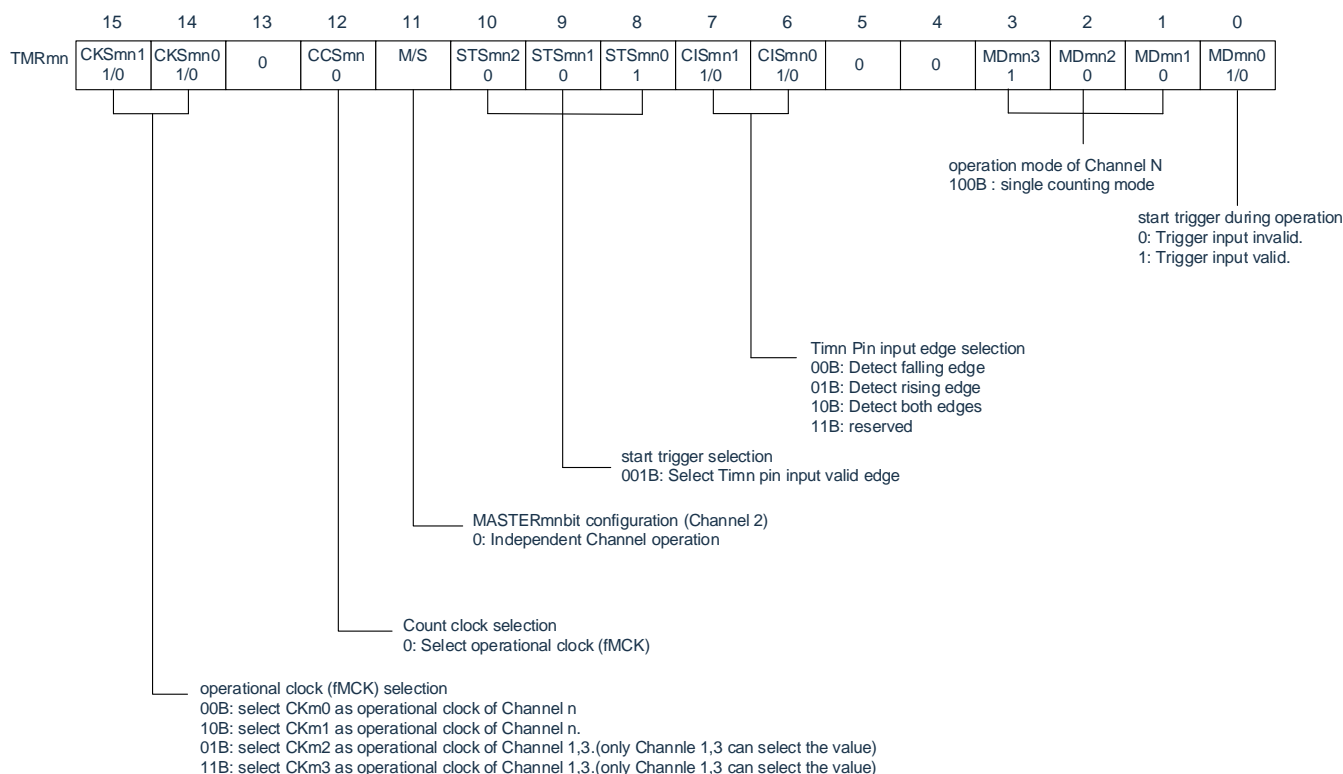


Remark:

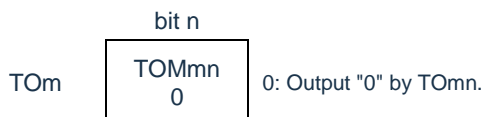
1. m: unit number (m=0) n: channel number (n=0~7)
2. TSmn: bit n of timer channel start register m (TSm)  
TEMn: bit n of timer channel enable status register m (TEm)  
TImn: TImn pin input signal  
TCRmn: timer count register mn (TCRmn)  
TDRmn: timer data register mn (TDRmn)

Figure5-59: Example of register contents setting for delay counter function

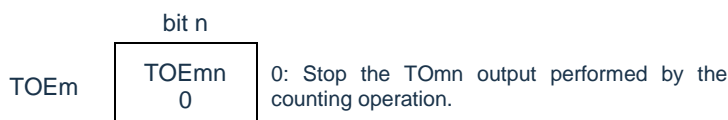
## (a) Timer mode register mn (TMRmn)



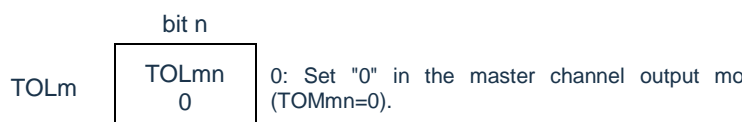
## (b) Timer output register m (TOM)



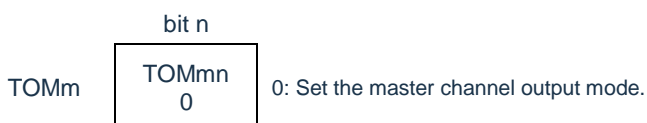
## (c) Timer output enable register m (TOEm)



## (d) Timer output level register m (TOLm)



## (e) Timer output mode register m (TOMm)



Note: TMRm2, TMRm4, TMRm6: MASTERmn bit

TMRm0, TMRm5, TMRm7: Fixed to "0".

Remark: m: unit number (m=0) n: channel number (n=0~7)

Figure5-60: Procedure for the delay counter function

	Software operation	Hardware status
Timer8 initial settings		The input clock of timer unit m is in the stop-providing state. (Stop providing clock, cannot write to each register)
	Set the TM80EN bit of the peripheral enable register 0(PER0) to "1".	The input clock of timer unit m is in the providing state and the channels are in the stop state. (Start providing clock, can write to each register)
	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 ~ CKm3.	
Initial setting of the 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.	
Start Operate	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 Operatio n	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.
Timer8 Stop	Set the TM80EN 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) n: channel number (n=0~7)

## 5.9 Multi-channel linkage operation function for general purpose timer units

### 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:

$$\begin{aligned}\text{Delay} &= \{\text{TDRmn (master) set value} + 2\} \times \text{counting clock period} \\ \text{Pulse width} &= \{\text{TDRmp (slave) set value} + 1\} \times \text{counting clock period}\end{aligned}$$

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.

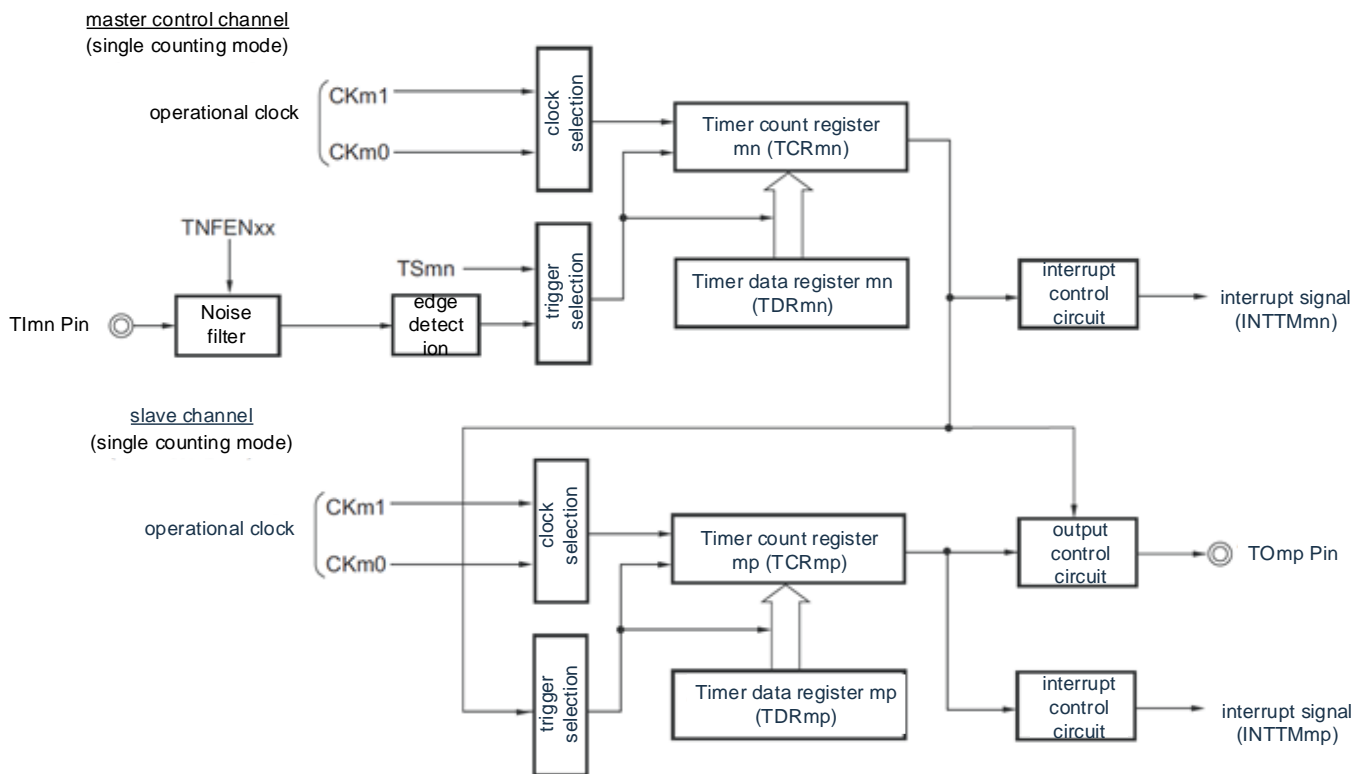
The software operation (TSMn=1) can also be used as a start trigger to output a single trigger pulse without using the TImn pin input.

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.

Remarks: m: unit number (m=0) n: master channel number (n=0, 2, 4, 6)

p: slave channel number ( $n < p \leq 7$ )

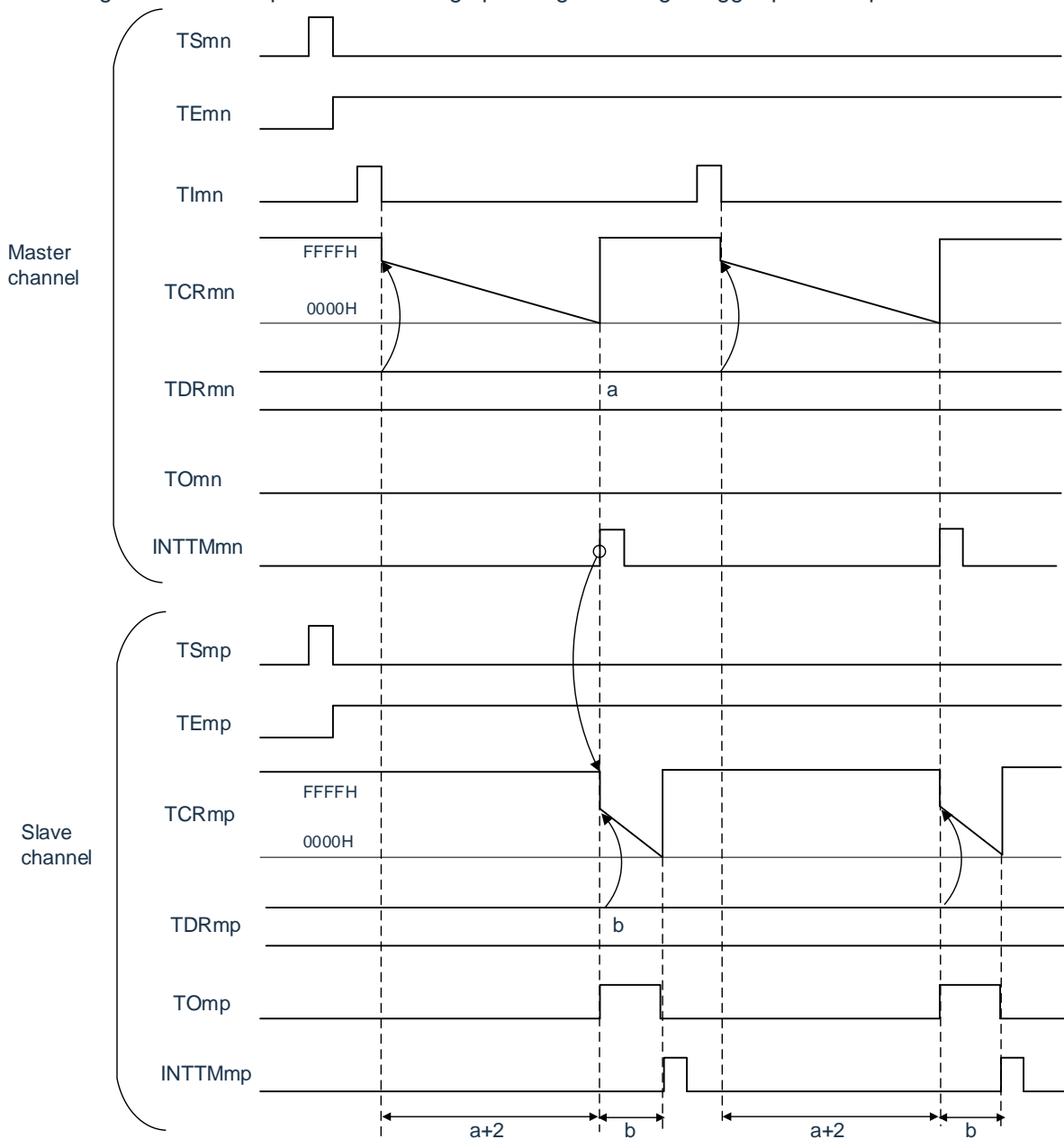
Figure5-61: Block diagram of operation as single trigger pulse output function



Remark: m: unit number (m=0) n: master channel number (n=0, 2, 4, 6)

p: slave channel number ( $n < p \leq 7$ )

Figure5-62: Example of basic timing operating as a single trigger pulse output function

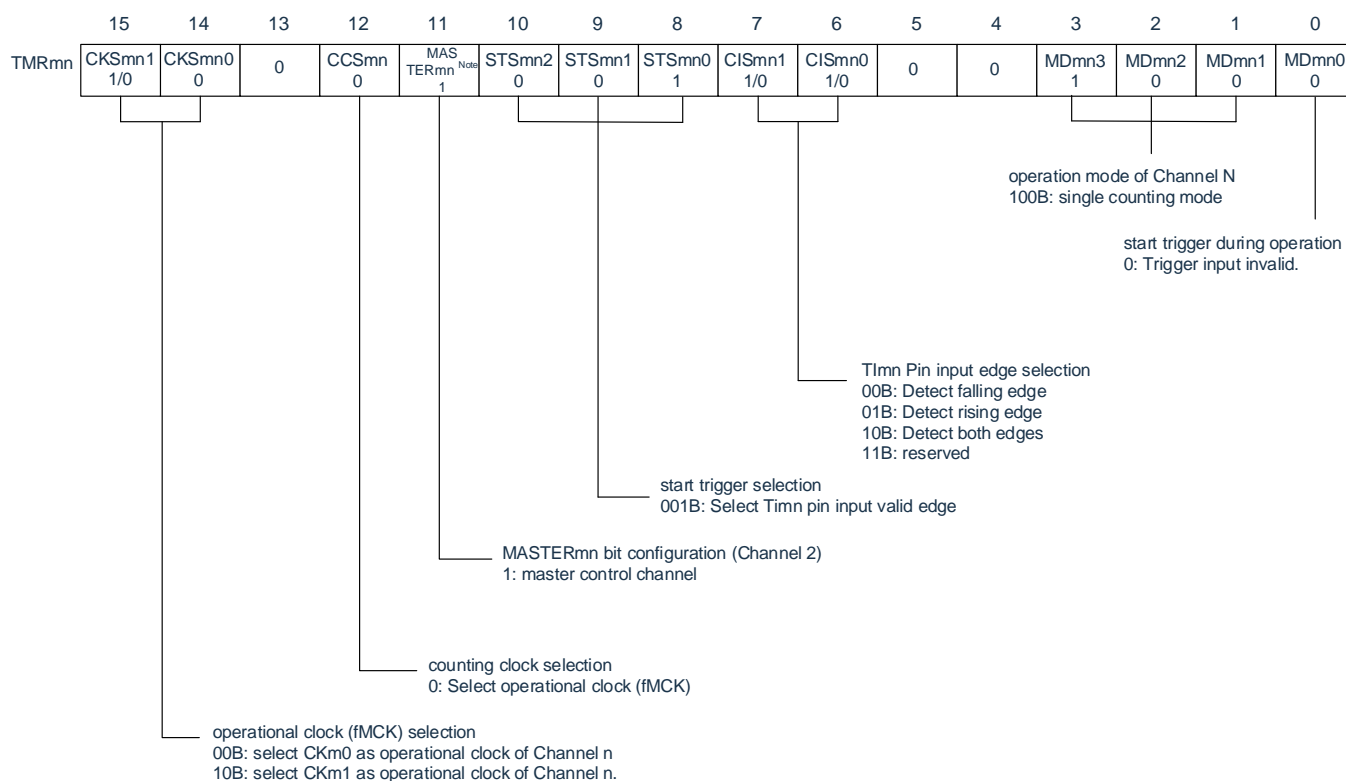


Remark:

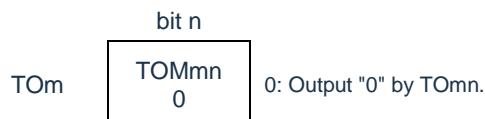
1. m: unit number (m=0) n: master channel number (n=0, 2, 4, 6)  
p: slave channel number (n < p ≤ 7)
2. TSmn, TSmp: bit n, p of timer channel start register m (TSm)  
TE mn, TE mp: bit n, p of timer channel enable status register m (TEm)  
TImn, TImp: input signal of TImn pin and TImp pin  
TCRmn, TCRmp: 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

Figure5-63: Example of register contents setting for single trigger pulse output function (master channel)

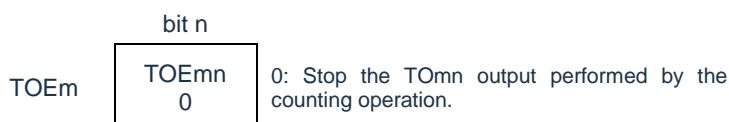
## (a) Timer mode register mn (TMRmn)



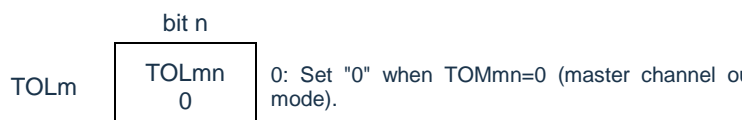
## (b) Timer output register m (TOM)



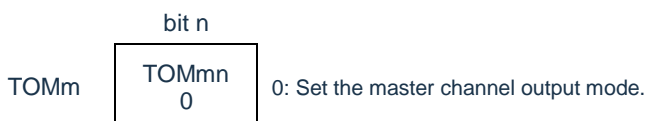
## (c) Timer output enable register m (TOEm)



## (d) Timer output level register m (TOLm)



## (e) Timer output mode register m (TOMm)

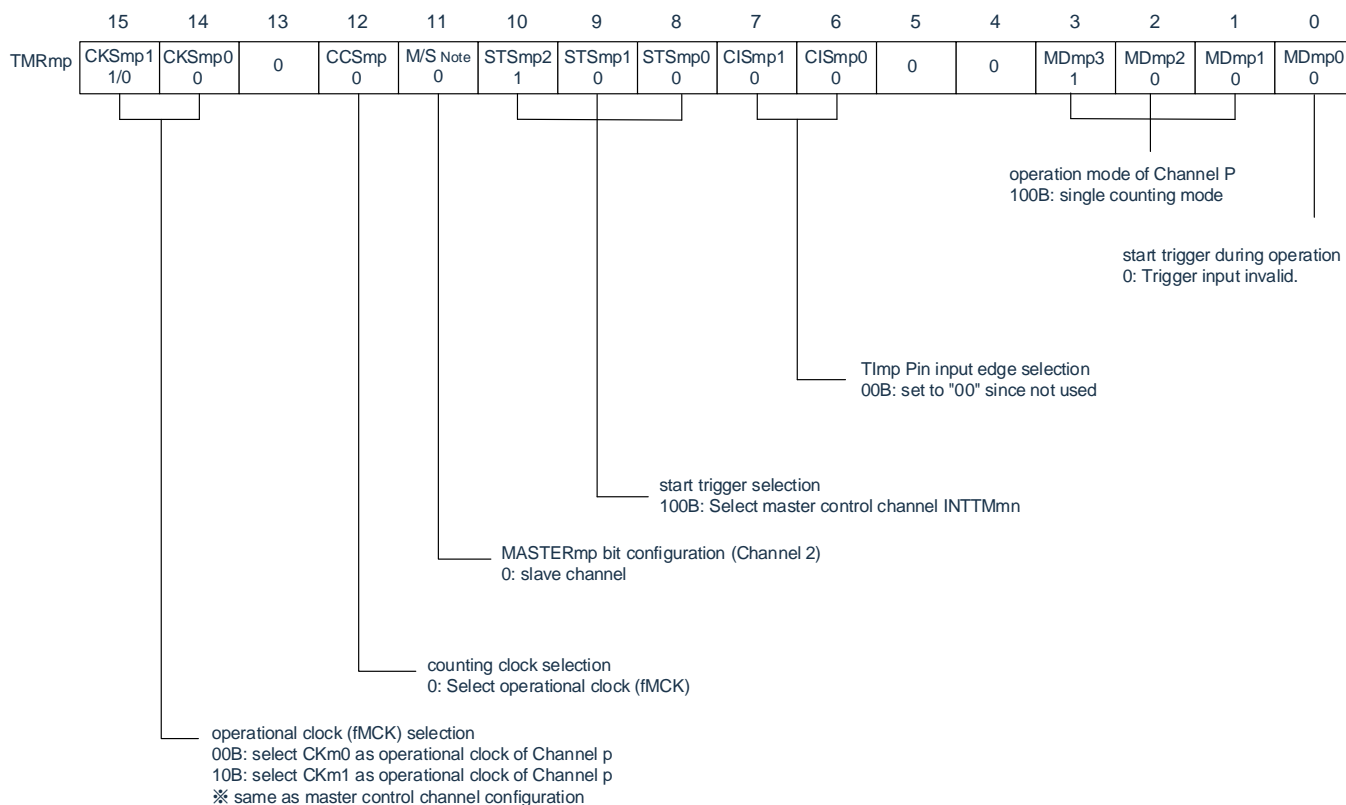


Remark: m: unit number (m=0) n: master channel number (n=0, 2, 4, 6)

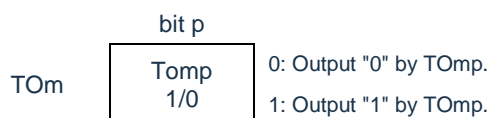


Figure5-64: Example of register contents setting for single trigger pulse output function (slave channel)

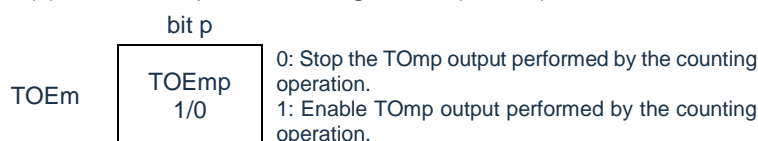
## (a) Timer mode register mn (TMRmn)



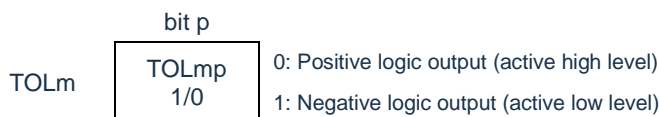
## (b) Timer output register m (TOM)



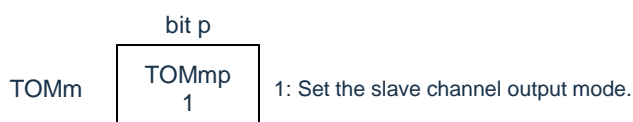
## (c) Timer output enable register m (TOEm)



## (d) Timer output level register m (TOLm)



## (e) Timer output mode register m (TOMm)



Note: TMRm2, TMRm4, TMRm6: MASTERmp bit

Remark: m: unit number (m=0) n: master channel number (n=0, 2, 4, 6) p: slave channel number (n &lt; p ≤ 7)

Figure5-65: Procedure for the single trigger pulse output function (1/2)

	Software operation	Hardware status
Timer8 initial settings		The input clock of timer unit m is in the stop-providing state. (Stop providing clock, cannot write to each register)
	Set the TM80EN bit of the peripheral enable register 0(PER0) to "1".	The input clock of timer unit m is in the providing state and the channels are in the stop state. (Start providing clock, can write to each register)
	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 and CKm1.	
Initial setting of the 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.

Remark: m: unit number (m=0) n: channel number (n=0~7)

Figure5-66: Procedure for the single trigger pulse output function (2/2)

	Software operation	Hardware status
Start operation n	<p>The setting values of the CISmn1 bit and the CISmn0 bit of the TMRmn register can be changed.</p> <p>The setting values of the TMRmp, TDRmn, TDRmp registers and the TOMmn bit, TOMmp bit, TOLmn bit, and TOLmp bit cannot be changed.</p> <p>The TCRmn and TCRmp registers can be read at any time.</p> <p>The TSRmn and TSRmp registers are not used.</p> <p>The setting of TOM register and TOEm register of the slave channel can be changed.</p>	<p>The master channel loads the value of the TDRmn register into the timer count register mn (TCRmn) by detecting the start trigger (detecting the active edge of the TImn pin input or setting the TSmn bit to "1" of the master channel) and decrementing the count. If TCRmn counts to "0000H", INTTMmn is generated and the count is stopped before the next TImn pin input.</p> <p>The slave channel uses the master channel's INTTMmn as the trigger, loads the value of the TDRmp register into the TCRmp register and the counter starts decrementing 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.</p> <p>Then, if TCRmp counts to "0000H", it stops counting after setting the output level of TOmp to an invalid level.</p> <p>Thereafter, repeat this operation.</p>
In operation n	<p>Set the TTmn bit (master) and TTmp bit (slave) to "1" at the same time.</p> <p>The operation automatically returns to "0" because the TTmn and Ttmp bits are trigger bits.</p>	<p>The TEMn and TEm bits become "0" and stops counting.</p> <p>The TCRmn register and TCRmp register hold the count value and stop counting. The TOmp output is not initialized but remains its state.</p>
	<p>Set the TOEmp bit of slave channel to "0" and set the value for the TOmp bit.</p>	<p>The TOmp pin outputs the level set by the TOmp.</p>
Stop Operation n	<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.</p> <p>No need to maintain the output level of the TOmp pin: No need to set.</p>	<p>The output level of the TOmp pin is maintained by the port function.</p>
Timer8 Stop	<p>Set the TM80EN 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.</p> <p>(TOMn bit becomes "0" and TOmp pin becomes port function)</p>

Remark: m: unit number (m=0) n: master channel number (n=0, 2, 4, 6) p: slave channel number (n < p ≤ 7)

## 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

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

cycle of the output pulses can be calculated using the following equations:

Remark: When the setting value of TDRmp (slave) > {TDRmn (master) set value+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.

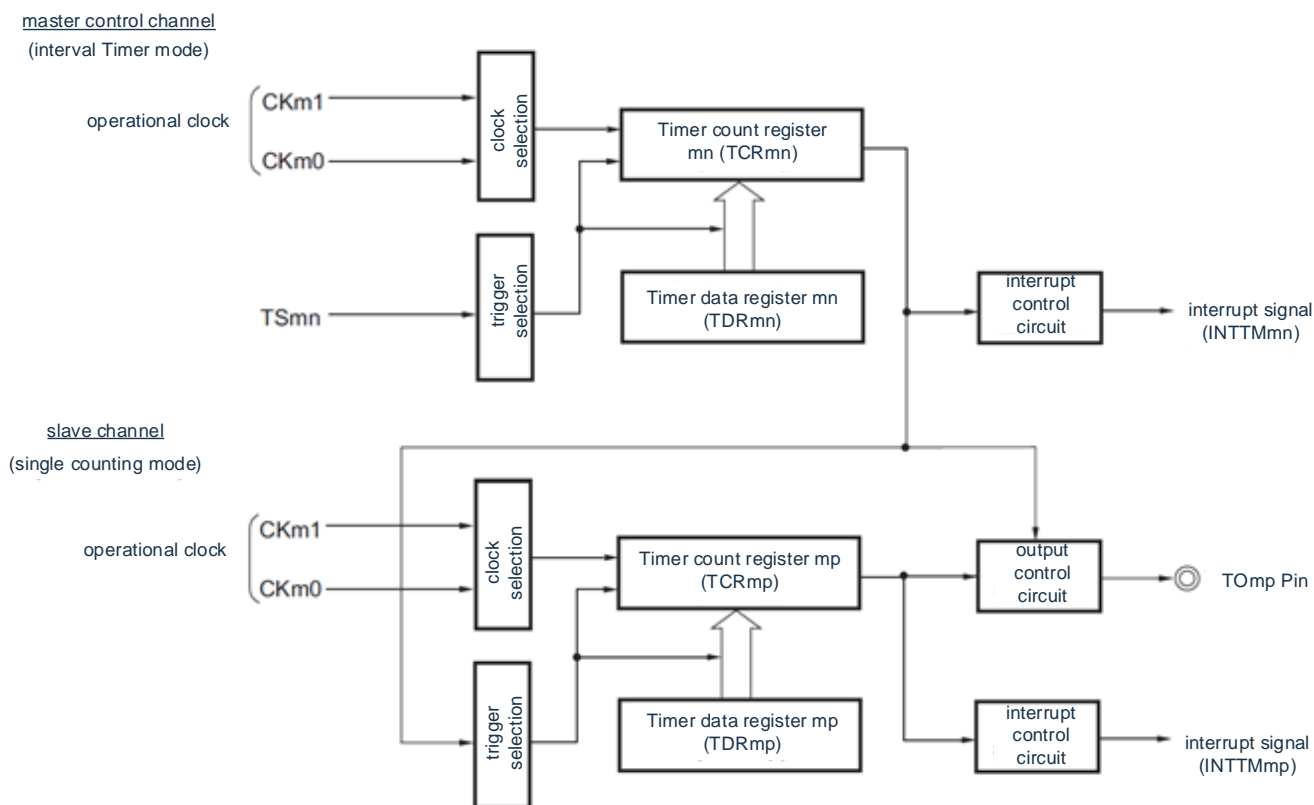
When used as PWM function, the slave channel decrements the count and the duty cycle of the PWM output (TOmp) for the period until "0000H" is counted.

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 the timer data register mn (TDRmn) of the master channel and the TDRmp register of the slave channel at the same time, 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 2 registers must be rewritten immediately after the master channel generates INTTMmn.

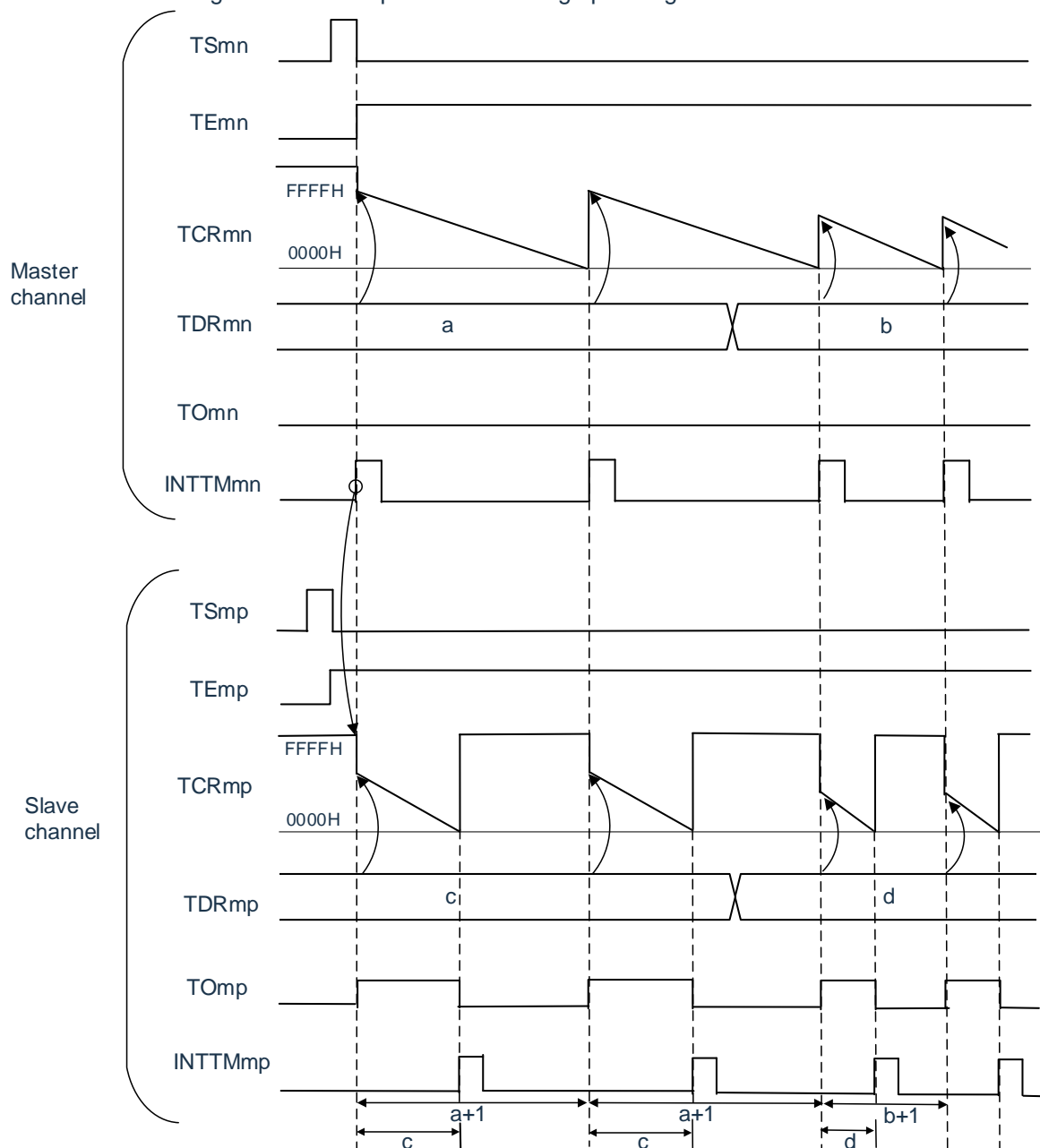
Remark: m: unit number (m = 0) n: master channel number (n = 0, 2, 4, 6) p: slave channel number (when m = 0: n < p ≤ 7)

Figure5-67: Block diagram of operation as PWM function



Remark: m: unit number (m = 0) n: master channel number (n = 0, 2, 4, 6) p: slave channel number (n < p ≤ 7)

Figure5-68: Example of basic timing operating as PWM function

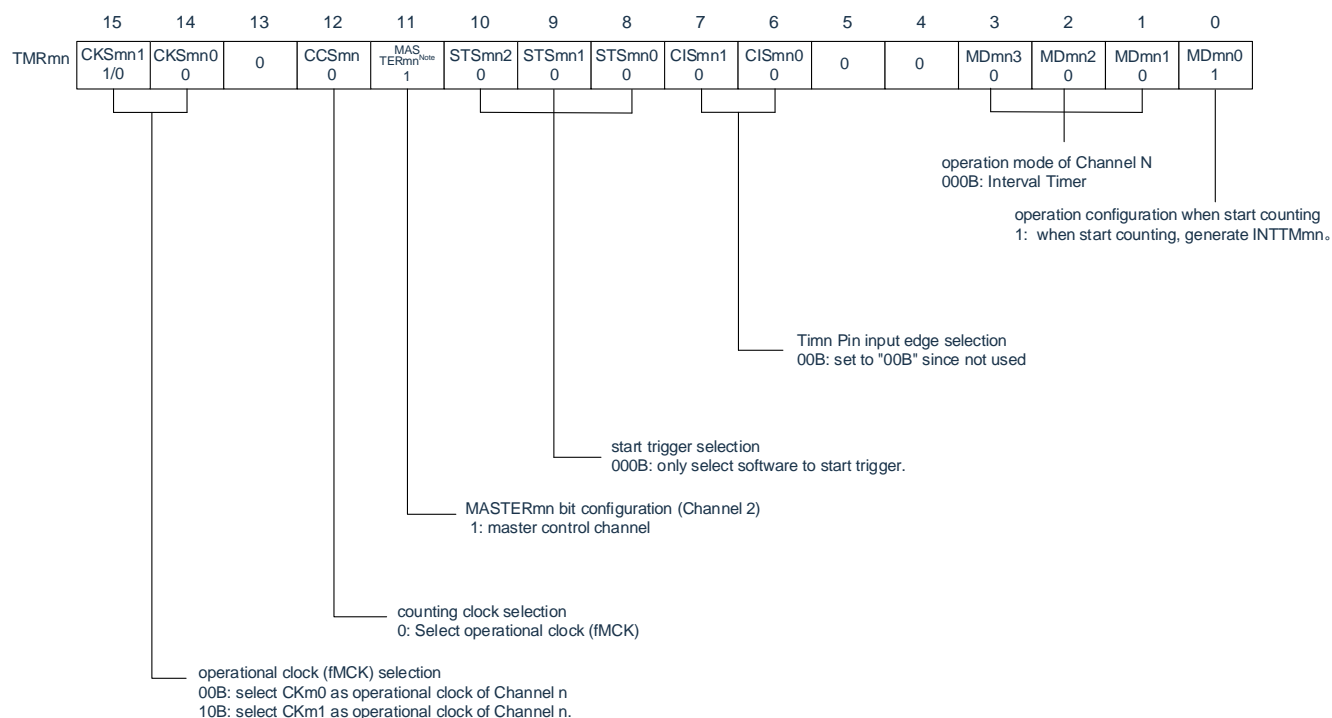


Remark:

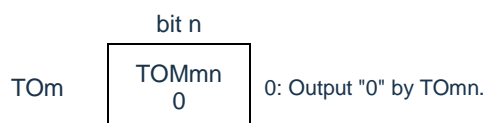
1.  $m$ : unit number ( $m=0$ )  $n$ : master channel number ( $n=0, 2, 4, 6$ )  
 $p$ : slave channel number ( $n < p \leq 7$ )
2.  $TS_{mn}$ ,  $TS_{mp}$ : bit  $n$ ,  $p$  of timer channel start register  $m$  ( $TS_m$ )  
 $TE_{mn}$ ,  $TE_{mp}$ : bit  $n$ ,  $p$  of timer channel enable status register  $m$  ( $TE_m$ )  
 $TCR_{mn}$ ,  $TCR_{mp}$ : timer count registers  $mn$ ,  $mp$  ( $TCR_{mn}$ ,  $TCR_{mp}$ )  
 $TDR_{mn}$ ,  $TDR_{mp}$ : timer data registers  $mn$ ,  $mp$  ( $TDR_{mn}$ ,  $TDR_{mp}$ )  
 $TO_{mn}$ ,  $TO_{mp}$ : output signals of  $TO_{mn}$  pin and  $TO_{mp}$  pin

Figure5-69: Example of register contents setting for PWM function (master channel)

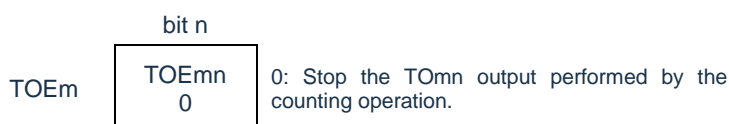
## (a) Timer mode register mn (TMRmn)



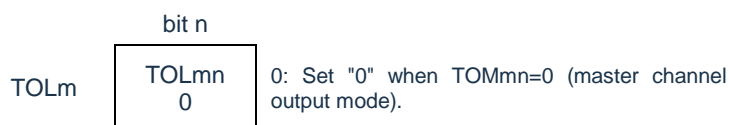
## (b) Timer output register m (TOM)



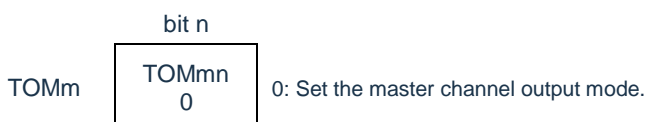
## (c) Timer output enable register m (TOEm)



## (d) Timer output level register m (TOLm)



## (e) Timer output mode register m (TOMm)



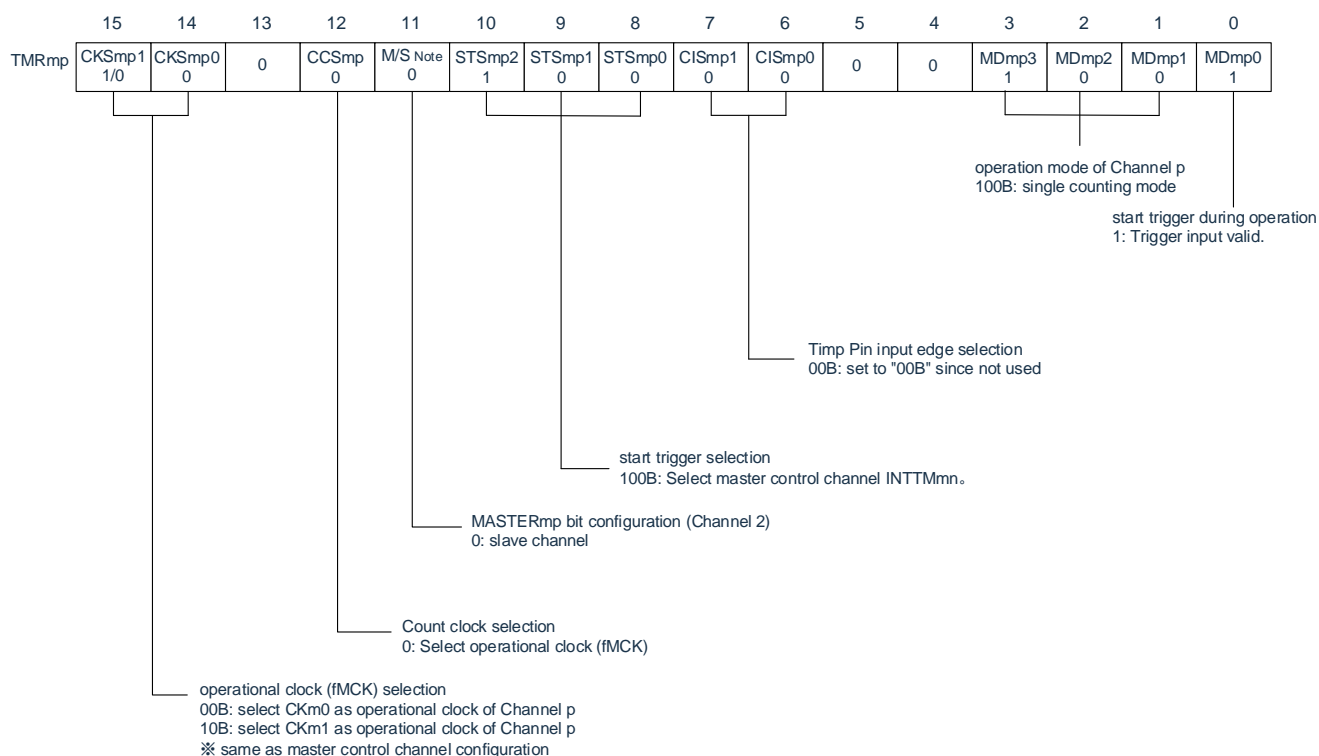
Note: TMRm2: MASTERmn= 1

TMRm0: Fixed to "0".

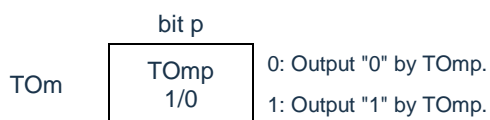
Remark: m: unit number (m=0) n: master channel number (n=0, 2, 4, 6)

Figure5-70: Example of register contents setting for PWM function (master channel)

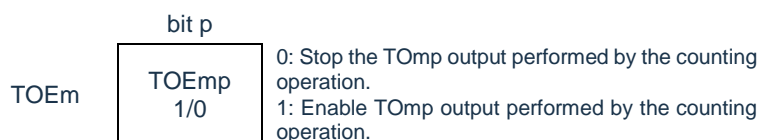
## (a) Timer mode register mn (TMRmn)



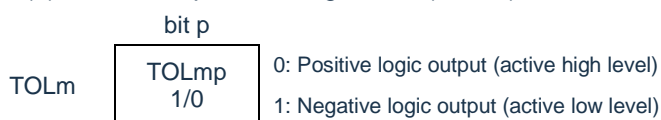
## (b) Timer output register m (TOM)



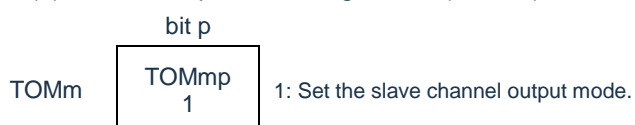
## (c) Timer output enable register m (TOEm)



## (d) Timer output level register m (TOLm)



## (e) Timer output mode register m (TOMm)



Note: TMRm2, TMRm4, TMRm6: MASTERmp bit

Remark: m: unit number (m = 0) n: master channel number (n = 0, 2, 4, 6) p: slave channel number (when m = 0: n &lt; p ≤ 7)



Figure5-71: Procedure for the PWM function (1/2)

	Software operation	Hardware status
Timer8 initial settings		The input clock of timer unit m is in the stop-providing state. (Stop providing clock, cannot write to each register)
	Set the TM80EN bit of the peripheral enable register 0(PER0) to "1".	The input clock of timer unit m is in the providing state and the channels are in the stop state. (Start providing clock, can write to each register)
	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 and CKm1.	
Initial setting of the 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 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.

Figure5-72: Procedure for the PWM function (2/2)

	Software operation	Hardware status
Restart the operation	<b>Start operation n</b> Set the TOEmp bit to "1" (only limited to restart operation). Set both the TSmn bit (master) and TSmp bit (slave) of the timer channel start register m (TSm) to "1". The operation automatically returns to "0" because the TSmn and TSmp bits are trigger bits.	The TEMn and TEmP bits become "1". The master channel starts counting and generates INTTMmn. With this as a trigger, the slave channel also starts counting.
	<b>In operation n</b> The setting values of the TMRmn and TMRmp registers and the TOMmn bit, TOMmp bit, TOLmn bit, and TOLmp bit cannot be changed. Ability to change the setting of the TDRmn register and the TDRmp register after the master channel has generated INTTMmn. The TCRmn and TCRmp registers can be read at any time. The TSRmn and TSRmp registers are not used.	The master channel loads the value of the TDRmn register into the timer count register mn (TCRmn) and perform decremental counting. If TCRmn counts till "0000H", then generating INTTMmn. At the same time, load the TDRmn register value into the TCRmn register and restart decremental counting. The slave channel use INTTMmn of master channel as a trigger, load the TDRmp register value into the TCRmp register and counter start decremental counting. After INTTMmn is output from the master channel and one count clock has elapsed, the output level of TOmp is set to an active level. Then, if TCRmp counts to "0000H", it stops counting after setting the output level of TOmp to an invalid level. Thereafter, repeat this operation.
	<b>Stop Operation</b> 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.	The TEMn and TEmP bits become "0" and stops counting. The TCRmn register and TCRmp register hold the count value and stop counting. The TOmp output is not initialized but remains its state.
	Set the TOEmp bit of slave channel to "0" and set the value for the TOmp bit.	The TOmp pin outputs the level set by the TOmp.
	<b>Timer8 Stop</b> 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. No need to maintain the output level of the TOmp pin: No need to set.	The output level of the TOmp pin is maintained by the port function.
	Set the TM80EN bit of the PER0 register to "0".	The input clock of timer unit m is in the stop-providing state. Initialize all circuits and the SFR for each channel. (TOMn bit becomes "0" and TOmp pin becomes port function)

Remark: m: unit number (m = 0) n: master channel number (n = 0, 2, 4, 6) p: slave channel number (when m = 0: n < p ≤ 7)

### 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

$$\begin{aligned} \text{Pulse period} &= \{\text{TDRmn (master) set value} + 1\} \times \text{counting clock period} \\ \text{Duty cycle 1 [\%]} &= \{\text{TDRmp (slave1) set value}\} / \{\text{TDRmn (master) set value} + 1\} \times 100 \end{aligned}$$

calculated by using the following equation:

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).

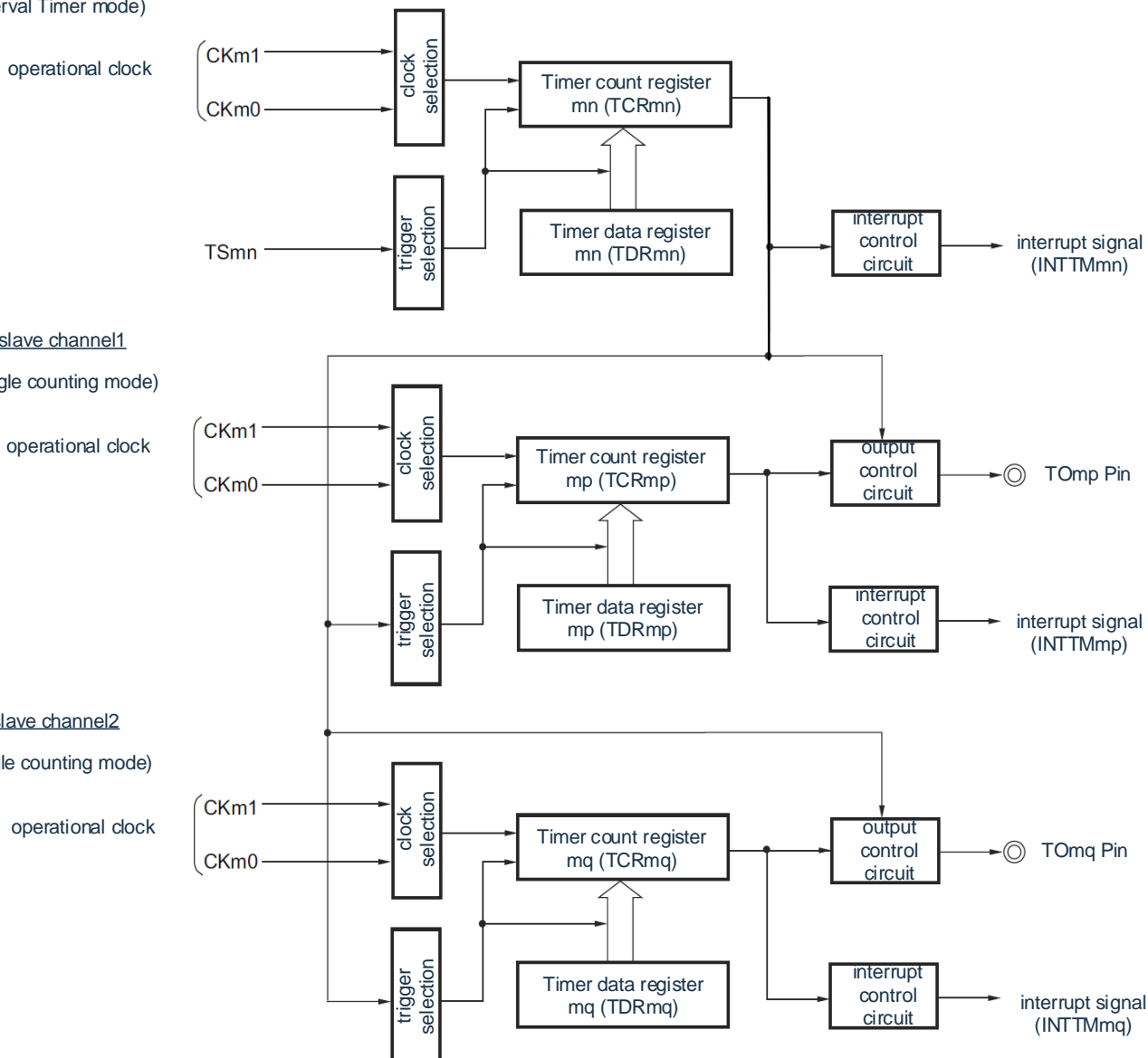
Remark: m: unit number (m=0) n: master channel number (n= 2, 4, 6)

p: slave channel number q: slave channel number  $n < p < q \leq 7$  (p and q are integers greater than n)

Figure5-73: Block diagram of operation as multiple PWM output function (output two types of PWMs)

master control channel

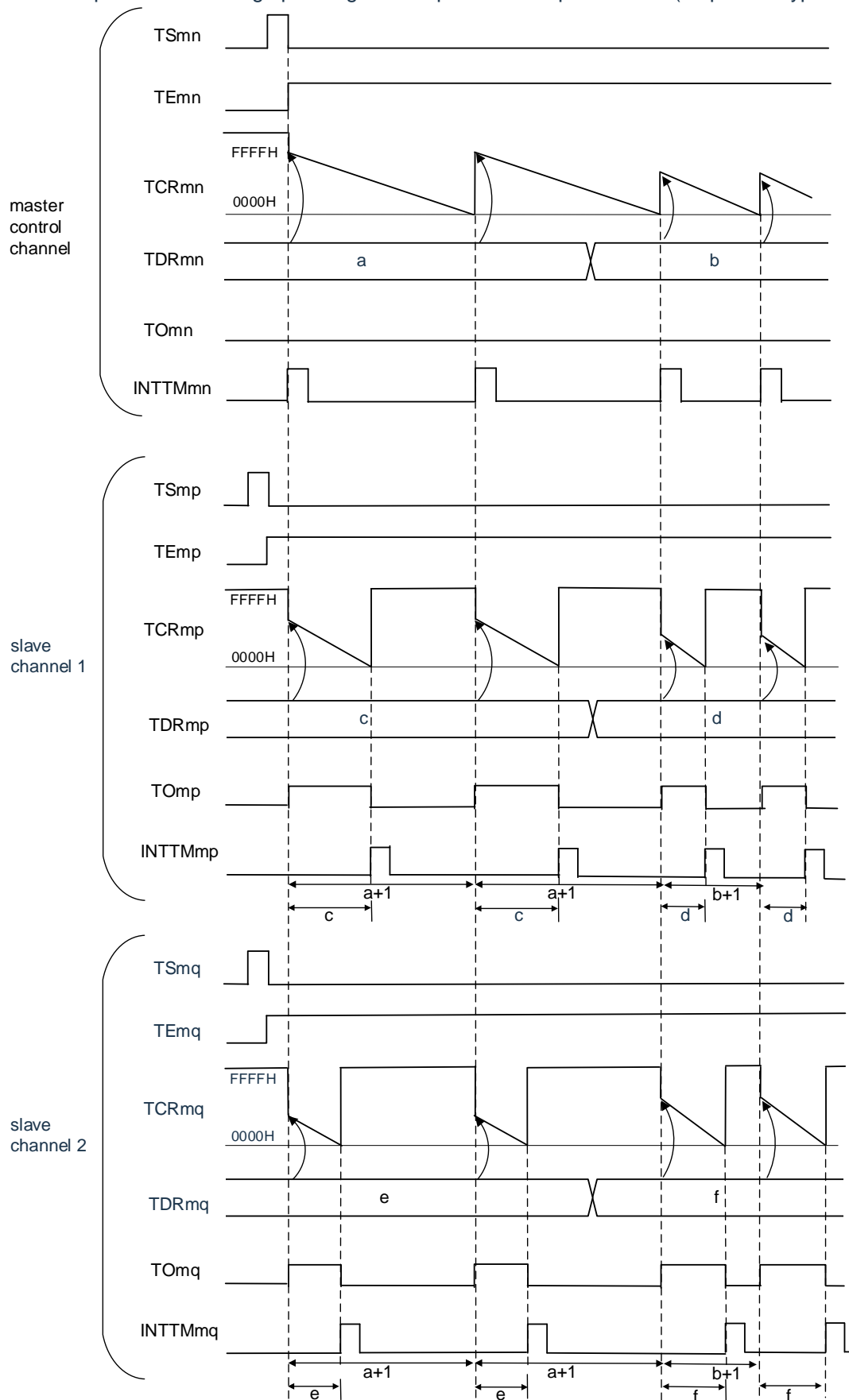
(interval Timer mode)



Remark: m: unit number (m=0) n: master channel number (n= 2, 4, 6)

p: slave channel number q: slave channel number  $n < p < q \leq 7$  (p and q are integers greater than n)

Figure5-74: Example of basic timing operating as multiple PWM output function (output two types of PWMs)

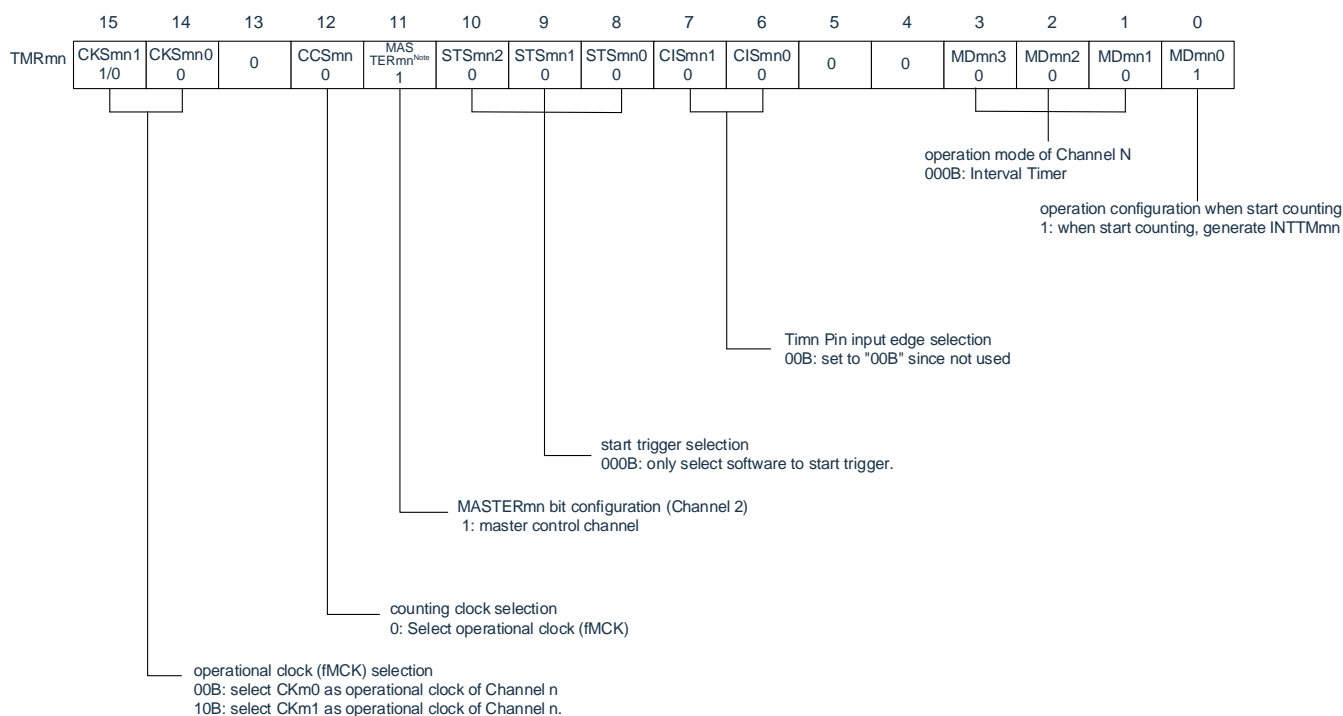


## Remark:

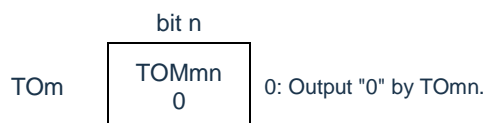
1. m: unit number (m=0)  
n: master channel number (n=0, 2, 4)  
p: slave channel number  
q: slave channel number  
For m = 0:  $n < p < q \leq 7$  (p and q are integers greater than n)
2. TSmn, TSmp, TSmq: bitn, p, q of timer channel start register m (TSm)  
TEmn, TEmq, TEmq: bitn, p, q of timer channel enable status register m (TEm)  
TCRmn, TCRmp, TCRmq: timer count registers mn, mp, mq (TCRmn, TCRmp, TCRmq)  
TDRmn, TDRmp, TDRmq: timer data registers mn, mp, mq (TDRmn, TDRmp, TDRmq)  
TOMn, TOMp, TOMq: output signals of TOMn, TOMp, TOMq pins

Figure5-75: Example of register contents setting for multiple PWM output function (master channel)

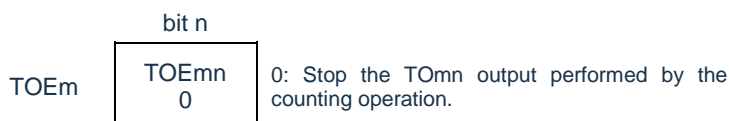
## (a) Timer mode register mn (TMRmn)



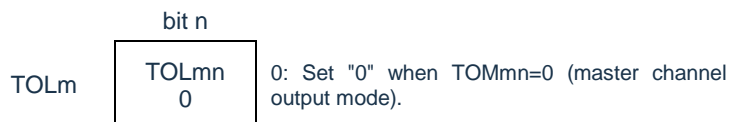
## (b) Timer output register m (TOM)



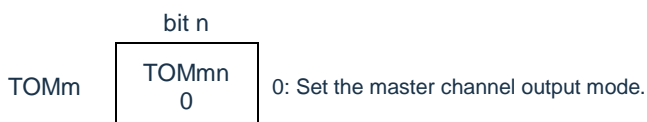
## (c) Timer output enable register m (TOEm)



## (d) Timer output level register m (TOLm)



## (e) Timer output mode register m (TOMm)



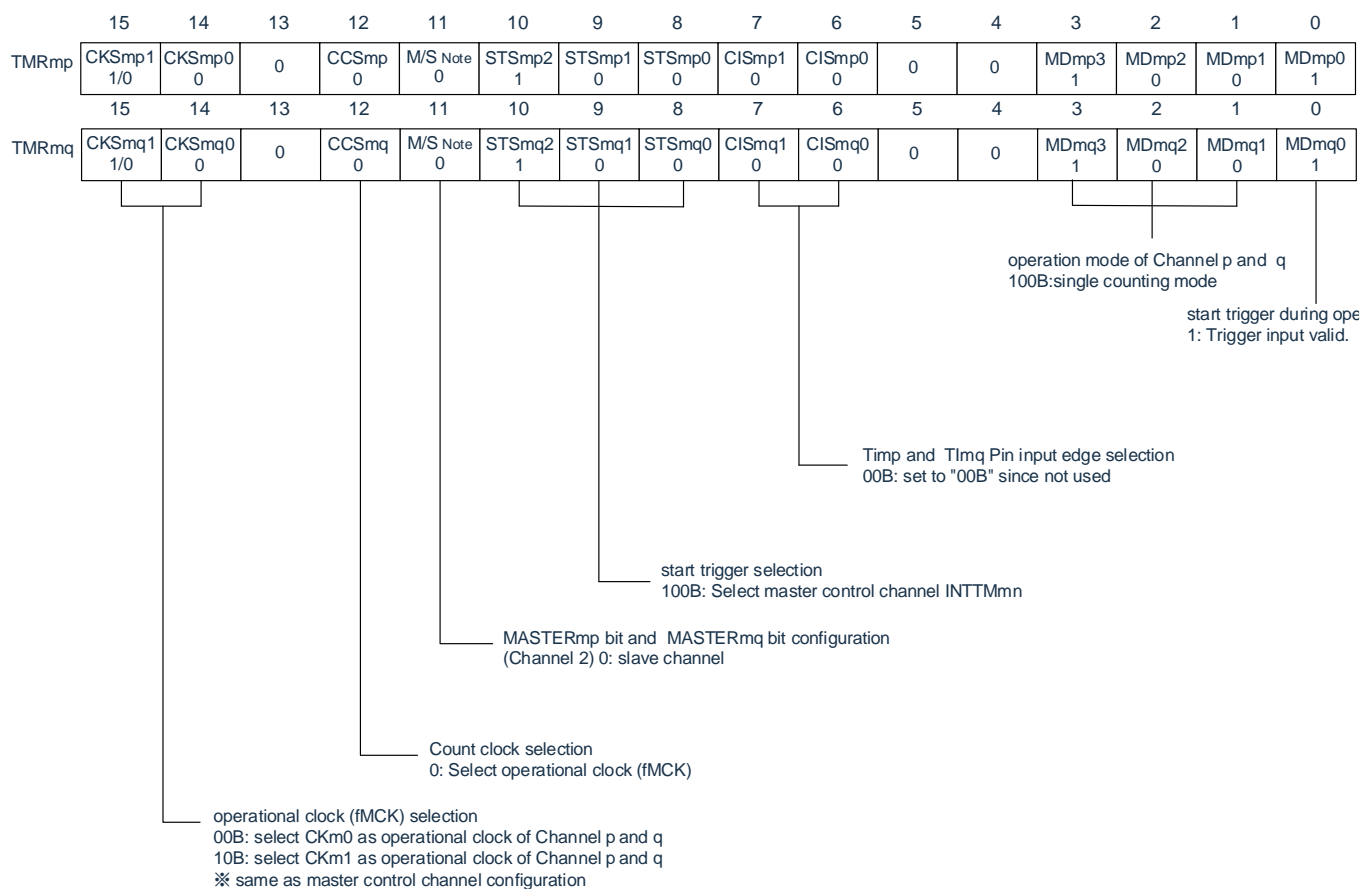
Note: TMRm2, TMRm4: MASTERmn= 1

TMRm0, TMRm5, TMRm7: Fixed to "0".

Remark: m: unit number (m=0) n: master channel number (n= 2, 4, 6)

Figure5-76: 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: Output "0" by by TOMP and TOMq. 1: Output "1" by by TOMP and TOMq.

(c) Timer output enable register m (TOEm)

	bit q	bit p	
TOEm	TOEmq 1/0	TOEmp 1/0	0: Stop the TOMP and Tomq outputs performed by the counting operation. 1: Enable the TOMP and Tomq outputs performed by the 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 level) 1: Negative logic output (active low level)

(e) Timer output mode register m (TOMm)

	bit q	bit p	
TOMm	TOMLq 1	TOMLp 1	1: Set the slave channel output mode.



Note: TMRm2, TMRm4: MASTERmp bit, MASTERmq bit

Remark: m: unit number (m=0) n: master channel number (n= 2, 4, 6)

p: slave channel number q: slave channel number m = 0 when:  $n < p < q \leq 7$  (p and q are integers greater than n)

Figure5-77: Procedure for the multiple PWM output function (output two types of PWMs) (1/2)

	Software operation	Hardware status
Timer8 initial settings		The input clock of timer unit m is in the stop-providing state. (Stop providing clock, cannot write to each register)
	Set the TM80EN bit of the peripheral enable register 0(PER0) to "1".	The input clock of timer unit m is in the providing state and the channels are in the stop state. (Start providing clock, can write to each register)
	Set the timer clock selection register m (TPSm). Determine the clock frequency of CKm0 and CKm1.	
Initial setting of the 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 value for the slave channel's TDRmp register and TDRmq register.	The channel is in the stop state. (Provides clock, and consumes some power)
	Slave channel setting Set TOMmp and TOMmq bits of the timer output mode register m (TOMm) to "1" (slave channel output mode). Set the TOLmp and TOLmq bits to "0". Set the TOMP and TOMq bits and determine the initial output level of the TOMP and TOMq bits. Set the TOEmp and TOEmq bits to "1" and enable TOMP and TOMq output. Set the Port Register and Port Mode Register to "0".	The TOMP pin is in Hi-Z output state. When the port mode register is in output mode and the port register is "0", The TOMP and TOMq initial set levels are output. The TOMP and TOMq remains unchanged because the channel is in the stop state. The TOMP pin and TOMq pin output the levels set by the TOMP and TOMq.

Figure5-78: Procedure for the multiple PWM output function (output two types of PWMs) (2/2)

	Software operation	Hardware status
Start operation n	(Set the TOEmp bit and TOEmq bit only when restarting operation. Set the TSmn bit (master), TSmp bit and TSmq bit (slave) of the timer channel start register m (TSM) to "1" at the same time. Since the TSmn bit, TSmp bit and TSmq bit are trigger bits, they are automatically returned to "0".	The TEMn, TEmq and TEMq bits become "1". The master channel starts counting and generates INTTMmn. With this as a trigger, the slave channel also starts counting.
In operation n	The setting values of the TMRmn, TMRmp, TMRmq registers and the TOMmn bit, TOMmp bit, TOMmq bit, TOLmn bit, TOLmp, TOLmq bit cannot be changed. The setting values of the TDRmn, TDRmp, and TDRmq registers can be changed after INTTMmn is generated by the master channel. The TCRmn, TCRmp and TCRmq registers can be read at any time. The TSRmn, TSRmp, and TSRmq registers are not used.	The master channel loads the value of the TDRmn register into the timer count register mn (TCRmn) and perform decremental counting. If TCRmn counts till "0000H", then generating INTTMmn. At the same time, load the TDRmn register value into the TCRmn register and restart decremental counting. The slave channel 1 uses the master channel's INTTMmn as a trigger, loads the value of the TDRmp register into the TCRmp register and the counter starts decrementing 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 counts to "0000H", it stops counting after setting the output level of the TOmp to an invalid level. The slave channel 2 uses the master channel's INTTMmn as a trigger, loads the value of the TDRmq register into the TCRmq register and the counter starts decrementing counting. After INTTMmn is output from the master channel and one count clock has elapsed, the output level of TOmq is set to an active level. Then, if counts to "0000H", it stops counting after setting the output level of the TOmq to an invalid level. Thereafter, repeat this operation.
Stop Operation n	Set the TTmn bit (master), TTmp bit and TTmq bit (slave) to "1" at the same time. Since the TTmn bit, TTmp bit and TTmq bit are trigger bits, they automatically return to "0".	The TEMn bit, TEmq bit, and TEMq bit all become "0" and stop counting. The TCRmn, TCRmp and TCRmq registers hold the count value and stop counting. The TOmp and TOmq outputs are not initialized and remain in state.
	Set the TOEmp and TOEmq bits of the slave channel to "0" and sets the values for the TOmp and TOmq bits.	The TOmp pin and TOmq pin output the levels set by TOmp and TOmq.
Timer8 Stop	To maintain the output levels of TOmp pin and TOmq pin: Set the TOmp bit and TOmq bits to "0" after setting the value to be held in the port register. No need to maintain the output levels of the TOmp pin and TOmq pin: No need to set.	The output level of the TOmp pin is maintained by the port function.
	Set the TM80EN 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. (TOmp bit and TOmq bit become "0" and TOmp pin and TOmq pin become port function)

Restart the operation

Remark: m: unit number (m=0) n: master channel number (n= 2, 4, 6)

p: slave channel number q: slave channel number  $n < p < q \leq 7$  (p and q are integers greater than n)

## 5.10 Cautions when using the general-purpose timer unit

### 5.10.1 Cautions when using timer output

Depending on the product, the pins assigned with timer output functions may also be assigned with outputs of other multiplexed functions. When using the timer output in this case, it is necessary to set the output of the other multiplexed function to its initial value.

For details, please refer to Chapter 2 Port Function".

# Chapter 6 TimerA

## 6.1 TimerA function

Timer A is a 16-bit timer capable of pulse output, pulse width and period measurement of external inputs, and counting of external events.

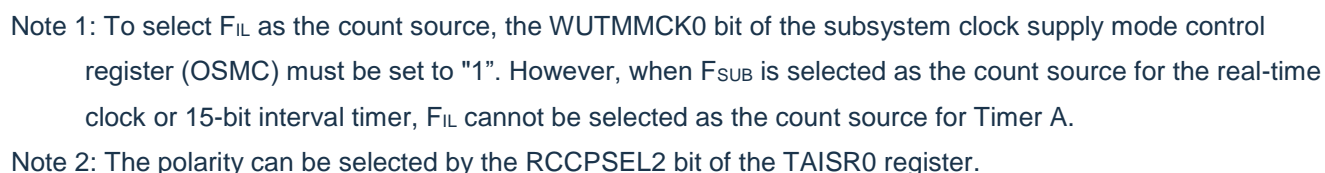
The 16-bit timer consists of a reload register and a decrement counter. The reload register and the decrement counter are assigned at the same address. If the TA0 register is accessed, the reload register and the counter can be accessed.

The specification and block diagram of timer A are as follows Figure6-1 and Table6-1 respectively.

Table6-1: Timer A specifications

Item		Content
Operation mode	Timer mode	Counts the count source
	Pulse output mode	Counts the count source and outputs pulses of opposite polarity when the timer underflow occurs.
	Event counter mode	Counts external events. It can also operate in deep sleep mode.
	Pulse width measurement mode	Measures the pulse width of the external input.
	Pulse period measurement mode	Measures the pulse period of the external input.
Counting source (operating clock)		Events can be selected from $F_{CLK}$ , $F_{CLK}/2$ , $F_{CLK}/8$ , $F_{IL}$ , $F_{SUB}$ or EVENTC input.
Interrupt		<ul style="list-style-type: none"> <li>•When the counter overflows</li> <li>•At the end of the effective width measurement of the external input (TAIO) in pulse width measurement mode</li> <li>•When the set edge of the external input (TAIO) is entered in pulse period measurement mode</li> </ul>
Select function		•Collaboration with EVENTC: Events entered by EVENTC can be selected as the count source.

The block diagram and pin structure of Timer A are as follows Figure6-1 and Table 6-2 respectively.



Pin name	I/O	Function
INTP1	Input	Event counter mode control for timer A
TAIO <sup>Note</sup>	I/O	External event input and pulse output of timer A
TAO <sup>Note</sup>	Output	Pulse output of timer A

[www.mcu.com.cn](http://www.mcu.com.cn)

## 6.3 Registers for controlling Timer A

The registers controlling Timer A are shown in Table 6-3.

Table 6-3: Registers for controlling Timer A

Register Name	Symbol
Port multiplexing function configuration register	PxxCFG
Peripheral enable register 0	PER0
Subsystem clock supply mode control register (OSMC)	OSMC
Register 0 controlling timer A <sup>Note</sup>	TA0
Register 0 controlling timer A	TACR0
Timer A/O control register 0	TAIOC0
Timer A mode register 0	TAMR0
Timer A event pin selection register 0	TAISR0
Port register x	Px
Port mode register x	PMx

Note: When the TA0 register is accessed, the CPU does not enter the processing of the next instruction but is in the wait state for CPU processing. Therefore, when this wait occurs, the number of clocks for instruction execution increases the number of clocks for waiting. The number of clocks waiting for both reading and writing when accessing the TA0 register is 1 clock.

### 6.3.1 Peripheral enable register 0(PER0)

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

To use the general timer A, bit0 (TMA) 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 6-2: Format of peripheral enabled register 0 (PER0)

Address: 40020420H	After reset: 00H		R/W					
Symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	IICAEN	IRDAEN	SCI2EN	SCI1EN	SCI0EN	TMAEN	TM80EN

TMAEN	Provides control of the input clock of Timer A
0	Stop providing input clock. • The SFR used by Timer A cannot be written. • Timer A in the reset state.
1	Provides input clock. • The SFR used by Timer A can be read and written.

Notice: To set Timer A, the TMAEN bit must be set to "1" first. When the TMAEN bit is "0", the write operation of the control register of Timer A is ignored, and the read value is the initial value (except for Port Mode Register PMx and Port Register Px).



## 6.3.2 Subsystem Clock Supply Mode Control Register (OSMC)

The operating clock of timer A can be selected by the WUTMMCK0 bits.

The RTCLPC bit is a bit that reduces power consumption by stopping the unwanted clock function. For the setting of RTCLPC bit, please refer to "Chapter 4 Clock Generation Circuit".

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

Figure 6-3: Format of the system clock supply mode control register (OSMC)

Symbol	7	6	5	4	3	2	1	0
OSMC	RTCLPC	0	0	WUTMMCK0	0	0	0	0

WUTMMCK0	Selection of the real time clock, the operation clock for the 15-bit interval timer (F <sub>RTC</sub> ) and Timer A
0	Sub-system Clock (F <sub>SUB</sub> ) <ul style="list-style-type: none"> <li>The subsystem clock is the running clock for the real-time clock and the 15-bit interval timer.</li> <li>The low-speed internal oscillator cannot be selected as the counting source for Timer A.</li> </ul>
1	Low-speed internal oscillator clock (F <sub>IL</sub> ) <ul style="list-style-type: none"> <li>The low-speed internal oscillator clock is the running clock for the real-time clock and the 15-bit interval timer.</li> <li>The low-speed internal oscillator or subsystem clock can be selected as the counting source for Timer A.</li> </ul>

### 6.3.3 Timer A count register 0 (TA0)

This is a 16-bit register. If this register is written, the data is written to the reload register. If this register is read, the count value is read. The status of the reload register and counter changes depending on the value of the TSTART bit of the TACR0 register. For details, refer to "6.4.1 Rewriting the reload register and counter".

The TA0 register is set by a 16-bit memory manipulation instruction. After the reset signal is generated, the value of TA0 register changes to "FFFFH".

Figure 6-4: Format of timer A count register 0 (TA0)

Symbol	Address: 40044004H				After reset: FFFFH				R/W							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TA0																

-	Function	Setting range
bit15~0	16-bit counter <sup>Note 1.2</sup>	0000H~FFFFH

Note 1: If you write "1" to the TSTOP bit of TACR0 register, it will force to stop the count of 16-bit counter and the count value will change to "FFFFH".

Note 2: If the value of bits TCK2~TCK0 of TAMR0 register is not "001B" ( $F_{CLK}/8$ ) and "011B" ( $F_{CLK}/2$ ) and the value of TA0 register is "0000H", only 1 request signal is generated to DMA and EVENTC immediately after counting starts. However, TAO and TAO perform alternate outputs.

In the event counter mode, if the value of TCK2 to TCK0 bits is "0000H", only one request signal is generated to DMA and EVENTC immediately after counting starts, and TAO is output alternately even if it is not counting the specified period.

If the value of TA0 register is greater than or equal to "0001H", a request signal is generated each time TA underflow occurs.

Note: When the TA0 register is accessed, the CPU does not enter the processing of the next instruction but is in the wait state for CPU processing. Therefore, when this wait occurs, the number of clocks for instruction execution increases the number of clocks for waiting. The number of clocks waiting for both reading and writing when accessing the TA0 register is 1 clock.

## 6.3.4 Timer A control register 0 (TACR0)

The TACR0 register is the register that controls the count and stop of register A and indicates the status of timer A.

The TACR0 register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of the TACR0 register changes to "00H".

Figure 6-5: Format of timer A control register 0 (TACR0)

Address: 40044000H		After reset: 00H		R/W				
Symbol	7	6	5	4	3	2	1	0
TACR0	0	0	TUNDF	TEDGF	0	TSTOP	TCSTF	TSTART

TUNDF	TimerA underflow flag
0	No underflow occurred.
1	Underflow occurs.
[condition of "0"] <ul style="list-style-type: none"> <li>When writing "0" to this bit by the program</li> </ul> [condition of "1"] <ul style="list-style-type: none"> <li>When the counter underflow occurs</li> </ul>	

TEDGF	Flags for active edges
0	No active edges
1	Have active edges
[condition of "0"] <ul style="list-style-type: none"> <li>When writing "0" to this bit by the program</li> </ul> [condition of "1"] <ul style="list-style-type: none"> <li>At the end of the active width measurement of the external input (TAIO) in pulse width measurement mode</li> <li>When the set edge of the external input (TAIO) is entered in pulse period measurement mode</li> </ul>	

TSTOP	The count of timer A is forced to stop <sup>Note 1</sup>
If a "1" is written to this bit, it forces the count to stop. The read value is "0".	

TCSTF	The count status flag for timer A <sup>Note 2</sup>
0	Stop counting
1	Counting
[condition of "0"] <ul style="list-style-type: none"> <li>When writing "0" to the TSTART bit (which becomes "0" synchronously with the count source).</li> <li>When writing "1" to the TSTOP bit</li> </ul> [condition of "1"] <ul style="list-style-type: none"> <li>When writing "1" to the TSTART bit (which becomes "1" synchronously with the count source).</li> </ul>	

TSTART	Timer A starts counting <sup>Note 2</sup>
0	Stop counting
1	Start counting
Start counting by writing "1" to the TSTART bit. Stop counting by writing "0" to the TSTART bit. If you set the TSTART bit to "1" (start counting), the TCSTF bit changes to "1" (counting) synchronously with the counting source. Also, after writing "0" to the TSTART bit, the TCSTF bit changes to "0" (stop counting) synchronously with the counting source. For details, please refer to "6.5.1 Start and stop control of counting".	

Note 1: If you write "1" to the TSTOP bit (force stop counting), the TSTART bit and TCSTF bit are initialized at the same time, and the pulse output levels are also initialized.

Note 2: Refer to "6.5.1 Start and stop control of counting" for cautions on using the TSTART and TCSTF bits.

### 6.3.5 Timer AI/O control register 0 (TAIOC0)

The TAIOC0 register is a register that sets the input/output of timer A. The TAIOC0 register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of the TAIOC0 register changes to "00H".

Figure 6-6: Format of timer AI/O control register 0 (TAIOC0)

Address: 40044001H

After reset: 00H

R/W

Symbol	7	6	5	4	3	2	1	0
TAIOC0	TIOGT1	TIOGT0	TIPF1	TIPF0	0	TOENA	0	TEDGSEL

TIOGT1	TIOGT0	TAIO Counting Control <small>Note 1.2</small>
0	0	Always count the events.
0	1	Events are counted during the polarity specified by INTP4.
1	0	Events are counted during the specified polarity of the timer output signal.
Others:		Settings are disabled.

TIPF1	TIPF0	Selection of TAIO input filters
0	0	No filter.
0	1	There are filters that are sampled via $F_{CLK}$ .
1	0	There are filters that are sampled via $F_{CLK}/8$ .
1	1	There are filters that are sampled via $F_{CLK}/32$ .
These bits specify the sampling frequency of the TAIO input filter. The input to the TAIO pin is sampled and if the sampled value is the same 3 times in a row, this value is determined to be the input value.		

TOENA	TAIO output enable
0	Disable TAO output (port).
1	Enable TAO output.

TEDGSEL	Polarity switching of I/O
Functions vary depending on the mode of operation (see Table 6-4 and Table 6-5).	

Note 1: When using INTP4 or a timer output signal, the count polarity of the event can be selected by the RCCPSEL2 bit of the TAISR0 register.

Note 2: TIOGT0 bit and TIOGT1 bit are valid only in event counter mode.

Table 6-4: Edge and polarity switching of TAIO input/output

Operation mode	Function
Timer mode	Not used (IO Ports).
Pulse output mode	0: Output from the "H" level (initial level: "H"). 1: Output from the "L" level (initial level: "L").
Event counter mode	0: Count on the rising edge 1: Count on the falling edge
Pulse width measurement mode	0: Measure the "L" level width 1: Measure the "H" level width
Pulse period measurement mode	0: Measure between the rising edge of the measurement pulse and the next rising edge 1: Measure between the falling edge of the measurement pulse and the next falling edge

Table 6-5: Polarity switching of TAO output

Operation mode	Function
All modes	0: Output from the "L" level (initial level: "L"). 1: Output from the "H" level (initial level: "H").

## 6.3.6 Timer A control register 0 (TAMR0)

The TAMR0 register is a register that sets the operating mode of register A. The TAMR0 register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of the TAMR0 register changes to "00H".

Figure 6-7: Format of timer A control register 0 (TAMR0)

Address: 40044002H

After reset: 00H

R/W

Symbol	7	6	5	4	3	2	1	0
TAMR0	0	TCK2	TCK1	TCK0	TEDGPL	TMOD2	TMOD1	TMOD0

TCK2	TCK1	TCK0	Timer A count source selection <sup>Note 1,2</sup>
0	0	0	F <sub>CLK</sub>
0	0	1	F <sub>CLK</sub> /8
0	1	1	F <sub>CLK</sub> /2
1	0	0	F <sub>IL</sub>
1	0	1	EVENTC input event
1	1	0	F <sub>SUB</sub>
Others:			Settings are disabled.

TEDGPL	TAIO edge polarity selection <sup>Note5</sup>
0	Single edge
1	Double edges

TMOD2	TMOD1	TMOD0	Timer A operation mode selection <sup>Note 3</sup>
0	0	0	Timer mode
0	0	1	Pulse output mode
0	1	0	Event counter mode
0	1	1	Pulse width measurement mode
1	0	0	Pulse period measurement mode
Others:			Settings are disabled.

Note 1: If you select the event counter mode, the external input (TAIO) is selected as the counting source, regardless of the setting of TCK0~TCK2 bits.

Note 2: You cannot switch the count source during the counting process. If you want to switch the counting source, you must do so when both the TSTART bit and TCSTF bit of the TACR0 register are "0" (stop counting).

Note 3: The operation mode can be changed only when the counting is stopped (TSTART bit and TCSTF bit of TACR0 register are both "0" (stop counting)), and cannot be changed during the counting process.

Note 1: To select F<sub>IL</sub> as the count source, the WUTMMCK0 bit of the subsystem clock supply mode control register (OSMC) must be set to "1". However, when F<sub>SUB</sub> is selected as the count source for the real-time clock or 12-bit interval timer, F<sub>IL</sub> cannot be selected as the count source for Timer A.

Note 5: The TEDGPL bit is only valid in event counter mode.

Note 6: Initialize the output of TAO pin and TAIO pin of Timer A by writing TAMR0 register. For the output level during initialization, please refer to "Figure 6-6: Format of timer A/I/O control register 0 (TAIOC0)".

### 6.3.7 Timer A event pin selection register 0 (TAISR0)

The TAISR0 register is a register that selects the timer that controls the event count period in the event counter mode and sets the polarity. The TAISR0 register is set by an 8-bit memory manipulation instruction.

After a reset signal is generated, the value of the TAISR0 register changes to "00H".

Figure 6-8: Format of timer A event pin select register 0 (TAISR0)

Address: 40044003H

After reset: 00H

R/W

Symbol	7	6	5	4	3	2	1	0
TAISR0	0	0	0	0	0	RCCPSEL2 Note	RCCPSEL1 Note	RCCPSEL0 Note

RCCPSEL2 <sup>Note</sup>	Selection of timer output signal and INTP4 polarity
0	Events are counted during the "L" level.
1	Events are counted during the "H" level.

RCCPSEL1 Note	RCCPSEL0 Note	Selection of timer output signals
0	0	TMIOD1
0	1	TMIOC1
1	0	TO02
1	1	TO03

Note: The RCCPSEL0~RCCPSEL2 bits are only valid in event counter mode.

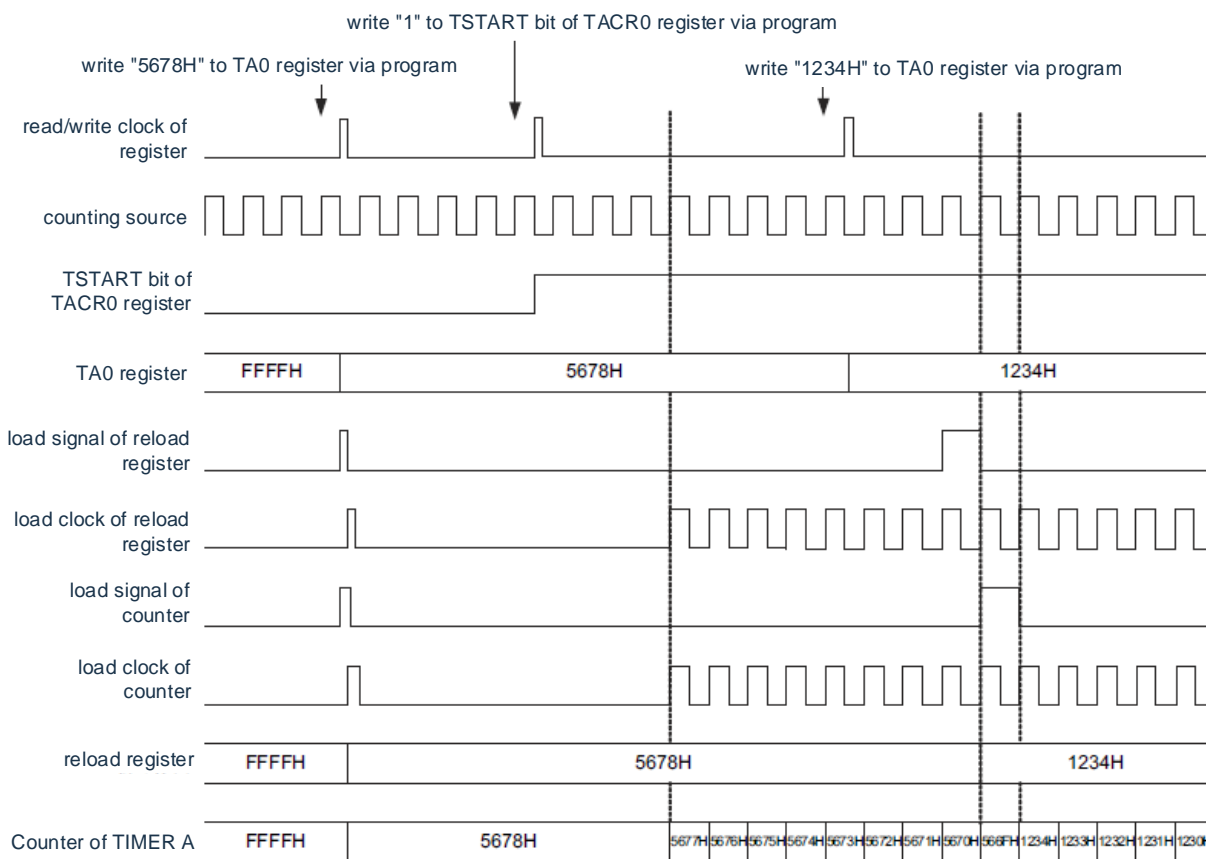
## 6.4 Timer A operation

### 6.4.1 Rewriting the reload register and counter

Independent of the operation mode, the rewrite timing of the reload registers and counters varies depending on the value of the TSTART bit in the TACR0 register. When the TSTART bit is "0" (stop counting), the reload register and counter are written directly. When the TSTART bit is "1" (start counting), after writing the reload register synchronously with the counting source, the reload register is written synchronously with the next counting source.

The rewrite timing diagram, determined by the value of the TSTART bit is shown in Figure 6-9.

Figure 6-9: Rewritten timing diagram determined by the value of the TSTART bit

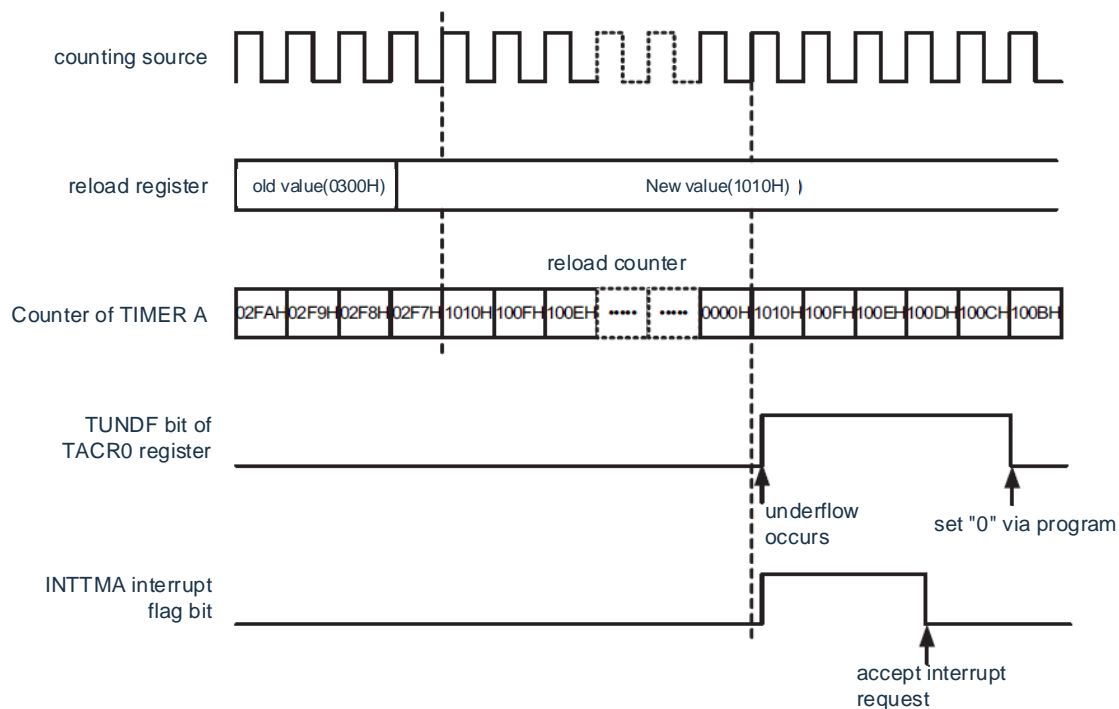




## 6.4.2 Timer mode

This is a mode of decreasing count by the count source selected by the TCK0~TCK2 bits of TAMR0 register. In the timer mode, the count value is decremented by 1 whenever the count source is input, and if the count value becomes "0000H" and the next count source is input, underflow occurs and an interrupt request is generated. An example of the timer mode is shown in Figure 6-10.

Figure 6-10: Example of timer mode operation



### 6.4.3 Pulse output mode

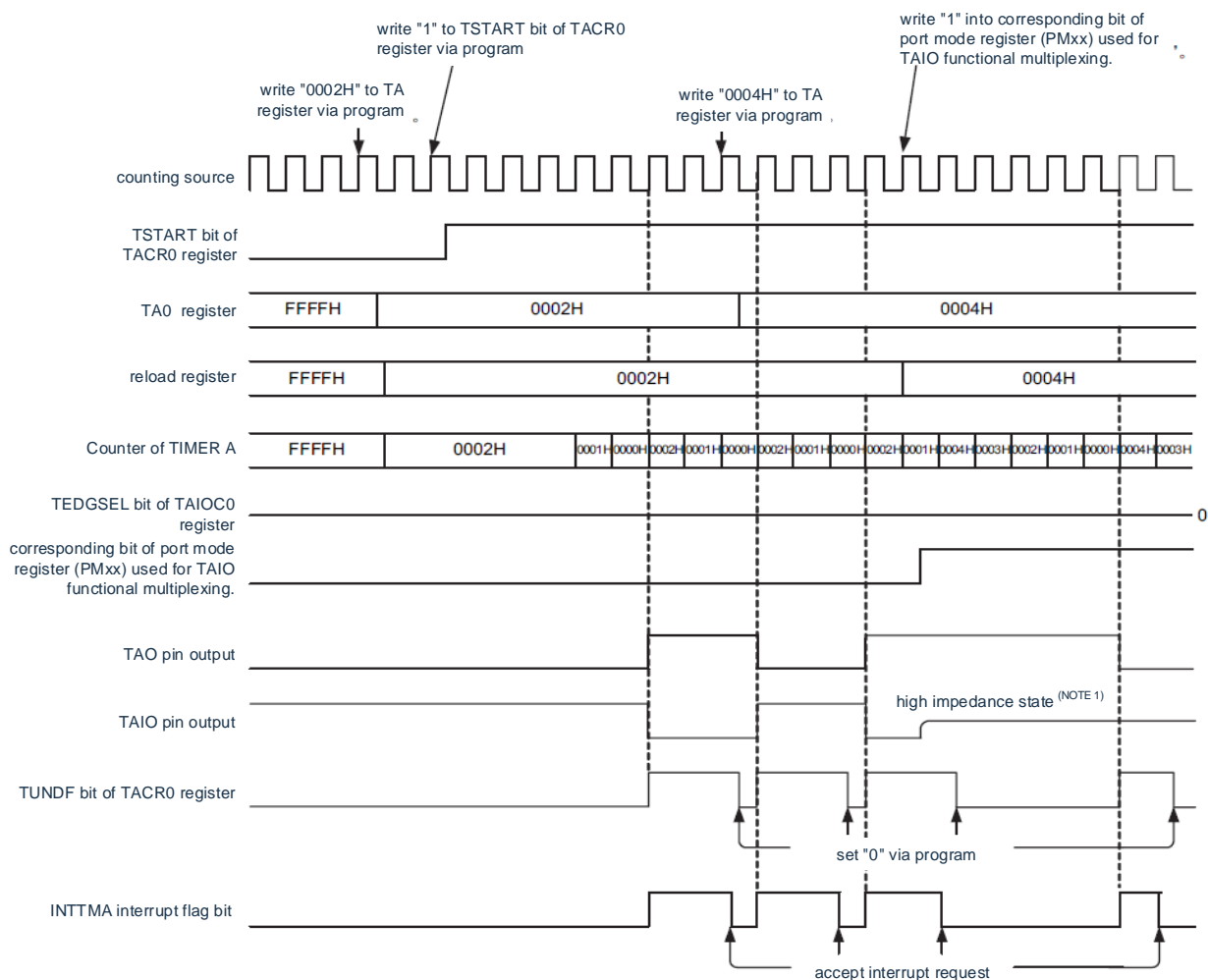
In this mode, the count source selected by bits TCK0 to TCK2 of the TAMR0 register is decremented to count, and the output levels of the TAIO pin and TAO pin are inverted and output whenever an underflow occurs.

In pulse output mode, the count value is decremented by 1 whenever a count source is input, and if the count value becomes "0000H" and the next count source is input, an underflow occurs and an interrupt request is generated.

Pulses can be output from the TAIO pin and TAO pin, and the output level is inverted whenever underflow occurs. The pulse output of the TAO pin can be stopped through the TOENA bit of the TAIOC0 register.

In addition, the output level can be selected by the TEDGSEL bit of the TAIOC0 register. An example of the pulse output mode is shown in Figure 6-11.

Figure 6-11: Example of operation in pulse output mode



NOTE 1: configure to high impedance state via port output enable control of the selected TAIO function.

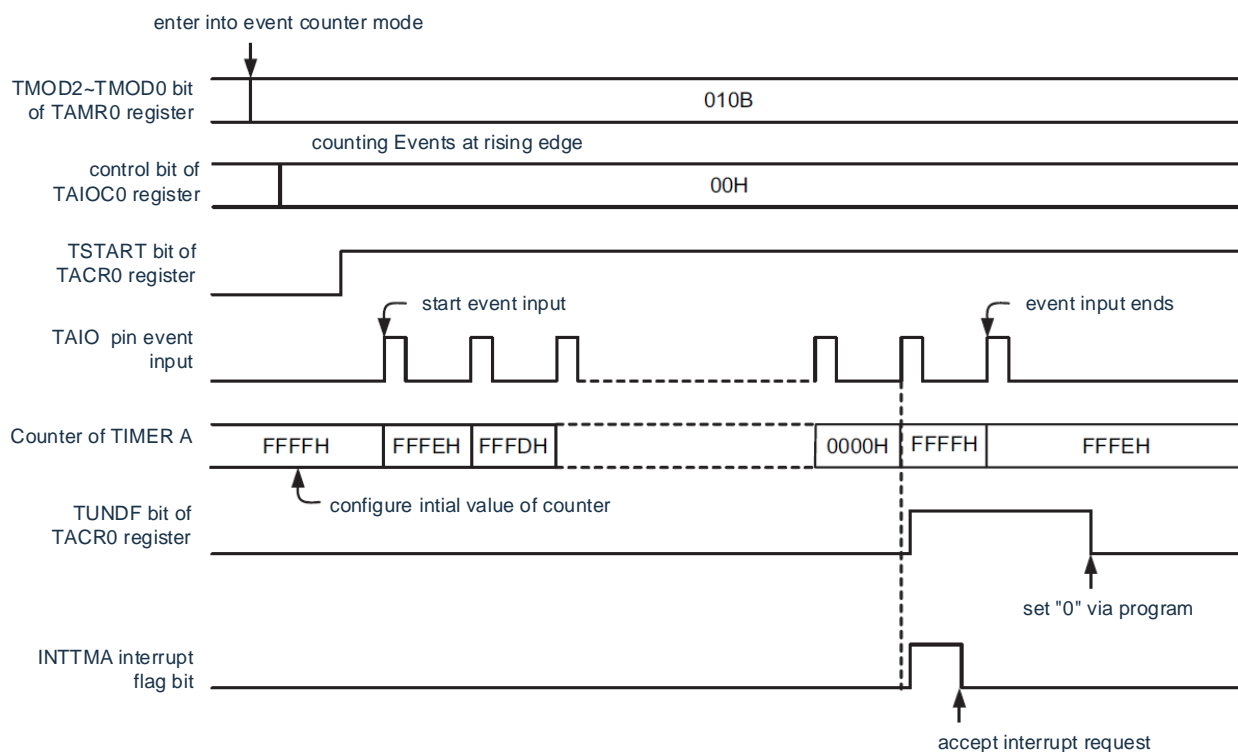
## 6.4.4 Event counter mode

This is a mode of decrementing counting via an external event signal (count source) input from the TAIO pin.

Various settings during event counting can be made via the TIOGT0~TIOGT1 bits of TAIOC0 register and TAISR0 register, and the filter function of TAIO input can be specified via the TIPF0~TIPF1 bits of TAIOC0 register.

The TAO pin is capable of alternate output even in the event counter mode. To use the event counter mode, refer to "6.5.5 Setting procedure for TAO pin and TAIO pin". An example of event counter mode operation<sup>1</sup> is shown in Figure 6-12.

Figure 6-12: Example of operating the event counter mode 1



An example of operating the specified time count in the event counter mode (TIOGT1 and TIOGT0 bits of TAIOC0 register are "01B" or "10B") is shown in Figure 6-13.

**Figure 6-13: Example of operating the event counter mode 2**

■ example of timing sequence to configure operational mode to following scenario.

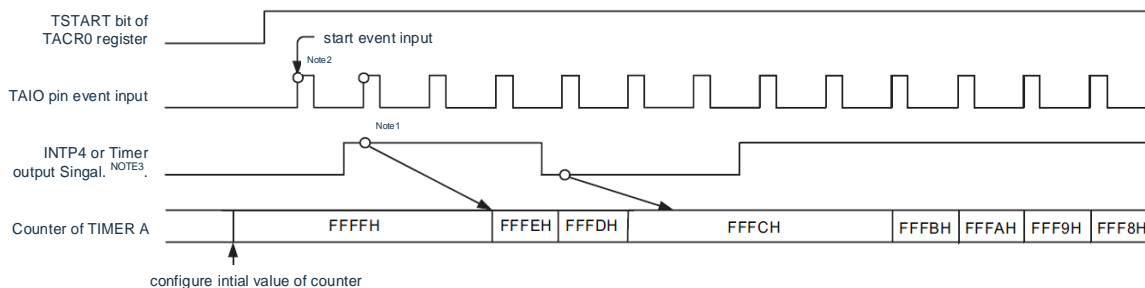
TAMR0 register: TMOD2, 1, 0=010B (Event counter mode)

TAIOC0 register: TIOGT1, 0=01B(event count during external interrupt pin defined period)

TIPF1, 0=00B (no filter)

TEDGSEL=0 (counting at rising edge)

TAISR0 register: RCCPSEL2=1(counting during H period)



The below precaution note only is relevant to the event counting mode configuration while TIOGT1 and TIOGT0 bit of TAIOC0 register is configured as "01B" or "10B".

NOTE1. To have synchronization control, 2 cycles of counting source clock delay can be reflected before counting execution starts.

2. the 2 counting source clock can start counting based on the state of previous counting stop, initialization shall be done towards internal circuit and start counting after operational configuration. In order to invalid the change of 2 counting source clock after counting starts, TSTOP bit of TACR0 register shall be set to '1'.

3. To timer output pin selected by RCCPSEL1 and RCCPSEL0 bit of TAISR0 register, the pins which are allocated to the timer output pin can not be used as other multiplex function output.

## 6.4.5 Pulse width measurement mode

This is a mode to measure the pulse width of the external signal input to the TAIO pin.

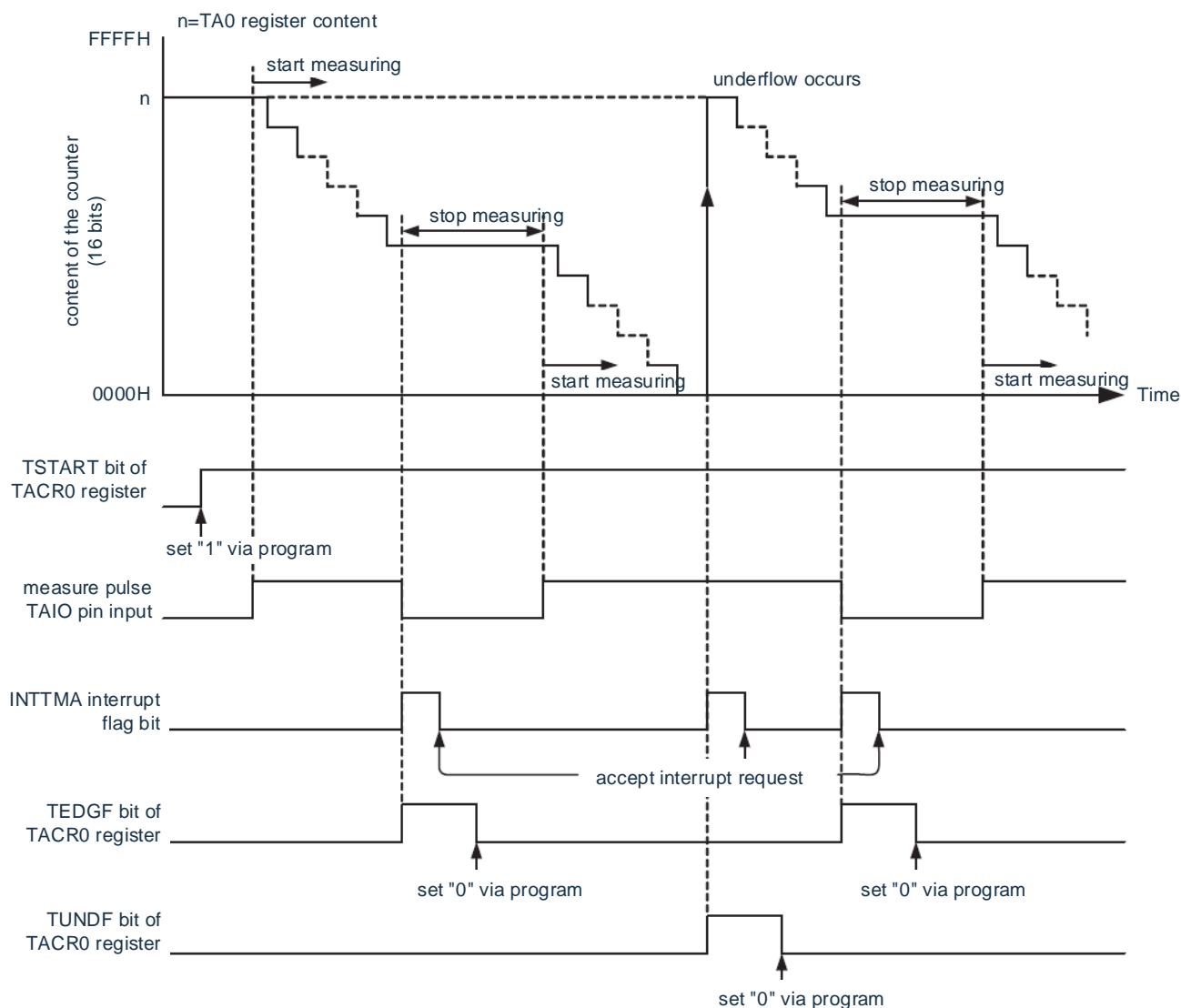
In the pulse width measurement mode, if the level specified by the TEDGSEL bit of the TAIOC0 register is input to the TAIO pin, counting starts decreasingly by the selected counting source. If the specified level input to the TAIO pin ends, the counter stops counting, the TEDGF bit in the TACR0 register becomes "1" (with an active edge) and an interrupt request is generated. The pulse width data is measured by reading the count value when the counter stops counting. If the counter underflows during the measurement, the TUNDF bit of the TACR0 register becomes "1" (underflow occurs) and an interrupt request is generated.

An example of the pulse width measurement mode in operation is shown in Figure 6-14.

To access the TEDGF bits and TUNDF bits of the TACR0 register, refer to "6.5.2 Flag access (TEDGF and TUNDF bits of TACR0 register)".

Figure 6-14: Example of operation in pulse measurement mode

This is the case where the measurement is performed on the "H" level of the measurement pulse (TEDGSEL=1 in the TAIOC0 register).



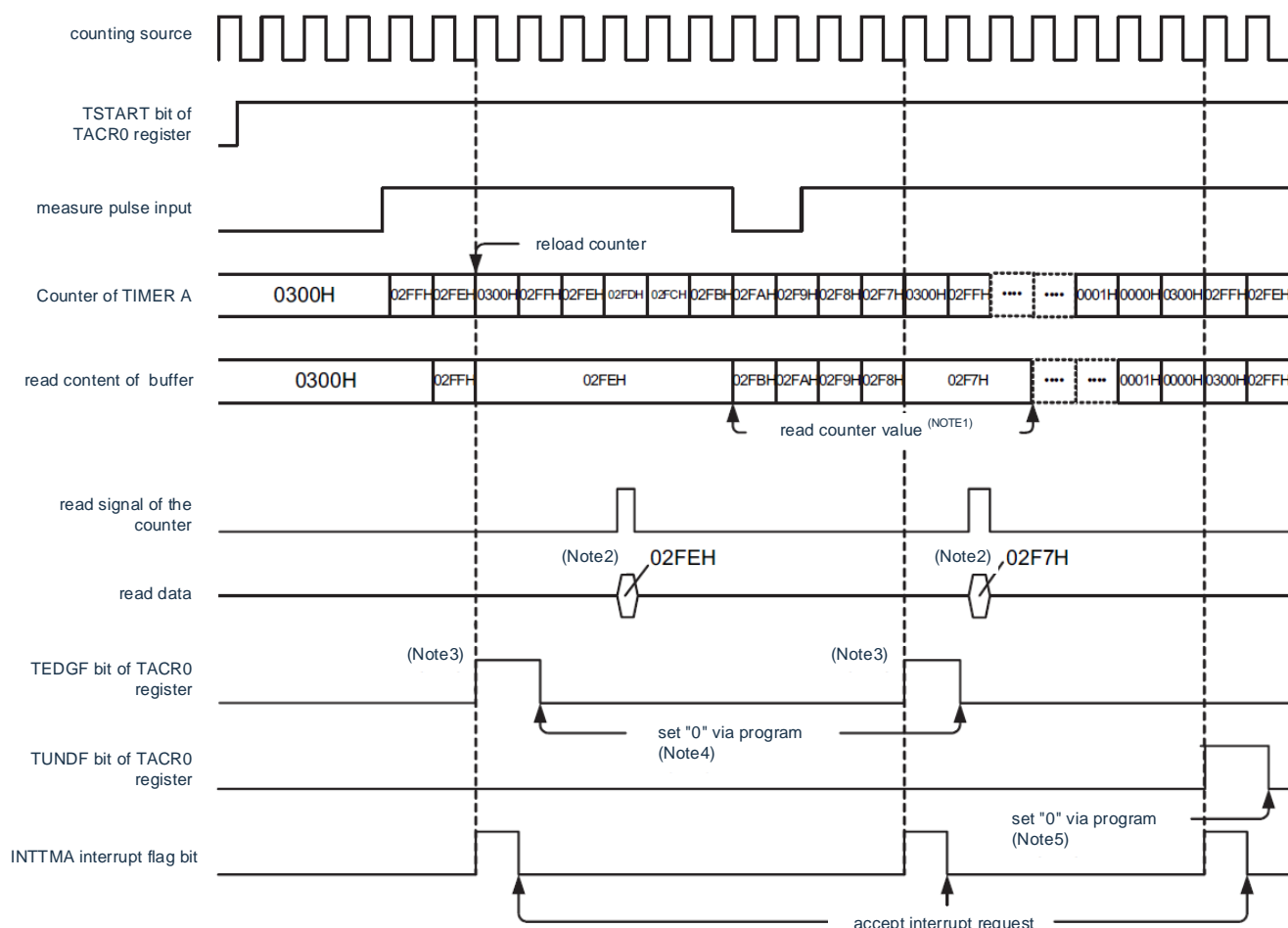
## 6.4.6 Pulse period measurement mode

This is a mode to measure the pulse period of the external signal input to the TAIO pin.

The counter counts decreasingly by the count source selected by bits TCK0 to TCK2 of the TAMR0 register. If a pulse for the period specified by the TEDGSEL bit of TAIOC0 register is input to the TIO pin, the count value is transferred to the read buffer on the rising edge of the count source and the value of the reload register is loaded to the counter on the next rising edge, while the TEDGF bit of TACR0 register becomes "1" (with an active edge) and an interrupt request is generated. At this point, the TA0 register (read buffer) is read, and the difference between the read and reload values is the cycle data of the input pulse. The cycle data is held until the read buffer is read. If the counter underflows, the TUNDF bit in the TACR0 register becomes "1" (underflow occurs) and an interrupt request is generated. An example of pulse period measurement mode operation is shown in Figure 6-15.

A pulse greater than two times the period of the count source must be input, and the width of the input "L" and "H" levels must be greater than the pulse period of the count source. If the input pulse period and width do not meet these conditions, the input pulse may be ignored.

Figure 6-15: Example of operation in pulse period measurement mode



This is the scenario done while TA0 register initial value as "0300H" and TEDGSEL bit of TAIOC0 register set to 0 and measurement done before pulse arises.

Note1. reading TA0 register must be done from the moment TEDGF bit changes to 1 till next valid edge input. Content of the read buffer will be preserved till reading TA0 register, thus, if the TA0 register is not read before the input valid edge, it will remain the measurement result of previous cycle.

2. if reading TA0 register in pulse period measurement mode, the read value is the content of read buffer.

3. TEDGF bit of the TACR0 register will change to 1 (valid edge), if specified edge of external pulse input occurs after the input measurement pulse valid edge.

4. TEDGF bit of the TACR0 register must be set to 0 via 8 bit operation instruction if program wants to set it to 0.

5. TUNDF bit of the TACR0 register must be set to 0 via 8 bit operation instruction if program wants to set it to 0.

## 6.4.7 Collaboration with EVENTC

By working with EVENTC, events entered by EVENTC can be set as the count source.

Counting is performed on the rising edge of the event of the ELC input via bits TCK0 to TCK2 of the TAMR0 register. However, the EVENTC input does not work in event counter mode.

The EVENTC setup procedure is shown below.

- Steps to start operation
  - (1) Set the event output target selection register (ELSELRn) of EVENTC.
  - (2) Set the operation mode of the event generation source.
  - (3) Set the mode of Timer A.
  - (4) Start the count of timer A.
  - (5) Start the operation of the event generation source.
- Steps to stop operation
  - (1) Stop the operation of the event generation source.
  - (2) Stop the count of Timer A.
  - (3) Set the event output target selection register (ELSELRn) of EVENTC to "0".

## 6.4.8 Output settings for each mode

The status of the TAO pin and TAIO pin in each mode are shown in Table 6-6 and Table 6-7.

Table 6-6: TAO pin settings

Operation mode	TAIOC0 register		Output of the TAO pin
	TOENA bit	TEDGSEL bit	
All modes	1	1	Inverted output
		0	Non-inverted output
	0	0/1	Disable output

Table 6-7: TAIO pin settings

Operation mode	TAIOC0 register		I/O of the TAIO pin
	PMXX bit <sup>Note</sup>	TEDGSEL bit	
Timer mode	0/1	0/1	Input (not used)
Pulse output mode	1	0/1	Disable output (Hi-Z output)
	0	1	Non-inverted output
		0	Inverted output
Event counter mode	1	0/1	Input
Pulse width measurement mode			
Pulse period measurement mode			

Note: This is the bit of the port mode register (PMxx) corresponding to the TAIO function multiplexing port.



## 6.5 Cautions when using timer A

### 6.5.1 Start and stop control of counting

- Event counting mode or when the counting source is set to non-EVENTC

If "1" is written to the TSTART bit of the TACR0 register during the counting stop, the TCSTF bit of the TACR0 register will be "0" (stop counting) within 3 counting source cycles. Except for the TCSTF bit, Timer A related register<sup>Note</sup> cannot be accessed before the TCSTF bit becomes "1" (counting).

If you write "0" (stop counting) to the TSTART bit during the counting process, the TCSTF bit is "1" within 3 counting source cycles. Stop counting when the TCSTF bit changes to "0". Except for the TCSTF bit, Timer A related register<sup>Note</sup> cannot be accessed before the TCSTF bit becomes "0" (counting). The interrupt register must be cleared before changing the TSTART bit from "0" to "1". For details, please refer to "Chapter 23 Interrupt function".

Note: Timer A's related registers: TA0, TACR0, TAIOC0, TAMR0, TAISR0

- • Event counting mode or when the count source is set to EVENTC

If "1" is written to the TSTART bit of the TACR0 register during the counting stop, the TCSTF bit of the TACR0 register will be "0" (stop counting) within 2 CPU clock cycles. Except for the TCSTF bit, Timer A related register<sup>Note</sup> cannot be accessed before the TCSTF bit becomes "1" (counting).

If you write "0" to the TSTART bit (stop counting) during counting, the TCSTF bit will be "1" within 2 CPU clock cycles. Stop counting when the TCSTF bit changes to "0". Except for the TCSTF bit, Timer A related register<sup>Note</sup> cannot be accessed before the TCSTF bit becomes "0".

The interrupt register must be cleared before changing the TSTART bit from "0" to "1". For details, please refer to "Chapter 23 Interrupt function".

Note: Timer A's related registers: TA0, TACR0, TAIOC0, TAMR0, TAISR0

### 6.5.2 Flag access (TEDGF bit and TUNDF bit of TACR0 register)

If you programmatically write "0" to the TEDGF and TUNDF bits of the TACR0 register, these bits will become "0". However, even if a "1" is written, the value remains the same. If a read-modify-write instruction is used for the TACR0 register, even if the TEDGF bit becomes "1" (with a valid edge) and the TUNDF bit becomes "1" (underflow occurs) during instruction execution, the TEDGF bit and TUNDF bit may be mistakenly changed due to timing. The TACR0 register must be accessed via an 8-bit memory manipulation instruction.

### 6.5.3 Access to counting registers

If the TSTART bit and TCSTF bit of the TACR0 register are both "1" (counting), successive writes to the TAO register must be separated by at least 3 count source clock cycles between the respective write operations.

### 6.5.4 Change in mode

The operation mode related registers (TAIOC0, TAMR0, TAISR0) of Timer A can be changed only when the count is stopped (TSTART bit and TCSTF bit of TACR0 register are both "0" (stop counting)), and cannot be changed during the count.

When the operation mode related registers of Timer A are changed, the values of the TEDGF and TUNDF bits are indefinite. Counting must start after writing "0" to the TEDGF bit (no valid edge) and "0" to the TUNDF bit (no underflow occurs).

### 6.5.5 Setting procedure for TAO pin and TAIO pin

After a reset, the multiplexed I/O ports on the TAO pin and TAIO pin are the input ports. To output from the TAO pin and TAIO pin, the following steps must be followed to set them up.

Steps to change

- (1) Set the mode.
- (2) Set the initial value and enable the output.
- (3) Set the bits of the corresponding port registers of the TAO and TAIO pins to "0".
- (4) Set the bits of the corresponding port mode registers of the TAO pin and TAIO pin to output mode.  
(output from TAO pin and TAIO pin)
- (5) Start counting (TSTART of TACR0 register=1).

To input from the TAIO pin, the following steps must be followed to set it up.

- (1) Set the mode.
- (2) Set the initial value and select the edge.
- (3) Set the bit of the corresponding port mode register of the TAIO pin to input mode.  
(input from TAIO pin)
- (4) Start counting (TSTART of TAMR0 register=1).
- (5) Wait until the TCSTF bit of the TACR0 register becomes "1" (counting).  
(Event counter mode only)
- (6) External events are input from the TAIO pin.
- (7) Invalid treatment of the measured value must be performed at the end of the first measurement (the second and subsequent measurements are valid).  
(Pulse width measurement mode and pulse period measurement mode only)

## 6.5.6 When timer A is not used

When Timer A is not used, you must set the TMOD2~TMOD0 register to "000B" (Timer mode) and set the TAIOC0 register to "0" (TAO output is disabled).

## 6.5.7 Stop of timer A operation clock

It is possible to control the provision or stop of Timer A clock by the TMA bit of PER0 register. However, the following SFR cannot be accessed while the timer A clock is stopped, but must be accessed in the state where the timer A clock is provided.

TA0 register, TACR0 register, TAMR0 register, TAIOC0 register and TAISR0 register

## 6.5.8 Setting steps for deep sleep mode (event counter mode)

To make the event counter mode run in deep sleep mode, you must transfer to deep sleep mode after providing the clock for timer A by following these steps

Setting steps

- (1) Set the operation mode.
- (2) Start counting (TSTART=1, TCSTF=1).
- (3) Stop providing the clock for Timer A.

To stop the event counter mode in deep sleep mode, the following steps must be followed to run the stop process.

- (1) Provide the clock for Timer A.
- (2) Stop counting (TSTART=0, TCSTF=0)

### **6.5.9 Function limitations in deep sleep mode (event counter mode only)**

To make the event counter mode run in deep sleep mode, the digital filter function cannot be used.

### **6.5.10 Forced count stop via the TSTOP bit**

The following SFRs cannot be taken 1 count source cycle after forcing the counter to stop counting via the TSTOP bit of the TACR0 register. TA0 register, TACR0 register and TAMR0 register

### **6.5.11 Digital Filters**

When using a digital filter, the timer cannot be started within 5 digital filter clock cycles after setting the TIPF1 bit and TIPF0 bit of TAIOC register.

In addition, in the state of using digital filter, even if the TEDGSEL bit of TAIOC register is changed, the timer operation cannot be started within 5 digital filter clock cycles as well.

## 6.5.12 Selecting $F_{IL}$ as the count source

To select  $F_{IL}$  as the count source, the WUTMMCK0 bit of the subsystem clock supply mode control register (OSMC) must be set to "1". However, when  $F_{SUB}$  is selected as the count source for the real-time clock or 15-bit interval timer,  $F_{IL}$  cannot be selected as the count source for Timer A.

# Chapter 7 Real time clock

## 7.1 Functions of real-time clock

The real-time clock has the following functions.

- Having counters of year, month, week, day, hour, minute, and second, and can count up to 99 years.
- Constant-period interrupt function (period: 0.5 seconds, 1 second, 1 minute, 1 hour, 1 day, 1 month)
- Alarm interrupt function (alarm: week, hour, minute)
- Pin output function of 1 Hz

## 7.2 Structure of real-time clock

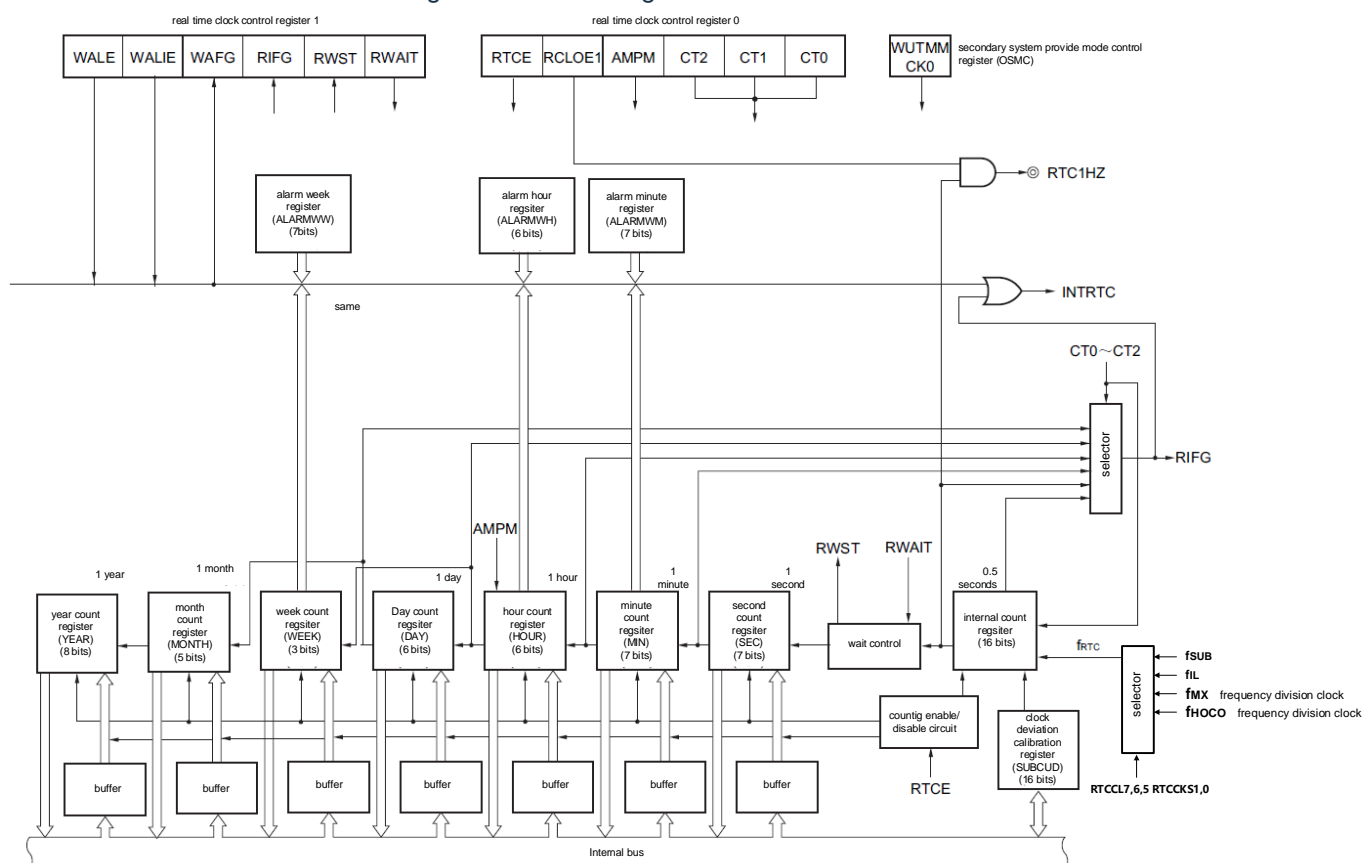
The real-time clock includes the following hardware.

Table7-1: 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 (RTCCL)
	Real-time clock control register 0 (RTCC0)
	Real-time clock control register 1 (RTCC1)
	Second count register (SEC)
	Minute count register (MIN)
	Hour count register (HOUR)
	Day count register (DAY)
	Week count register (WEEK)
	Month count register (MONTH)
	Year count register (YEAR)
	Watch error correction register (SUBCUD)
	Alarm minute register (ALARMWM)
	Alarm hour register (ALARMWH)
	Alarm week register (ALARMWW)

Note: The reset of the above RTC control registers is controlled by POR reset only.

Figure7-1: Block diagram of real-time clock



Notice: The count of year, month, week, day, hour, minutes and second can only be performed when the  $F_{MX}/F_{HOCO}$  week divider clock (after the divider  $\approx 32,768$  KHz) or a subsystem clock ( $F_{SUB} = 32.768$  kHz) is selected as the operation clock of the real-time clock. When the low-speed internal oscillator clock ( $F_{IL} = 32.768$  KHz) is selected, only the fixed-cycle interrupt function can be used.

The fixed-cycle interrupt interval when  $F_{IL}$  is selected is calculated by using the following equation:

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

## 7.3 Registers for controlling real-time clock

The real-time clock is controlled by the following registers.

- Peripheral enable register 0 (PER0)
- Real-time clock selection register (RTCCL)
- Real-time clock control register 0 (RTCC0)
- Real-time clock control register 1 (RTCC1)
- Second count register (SEC)
- Minute count register (MIN)
- Hour count register (HOUR)
- Day count register (DAY)
- Week count register (WEEK)
- Month count register (MONTH)
- Year count register (YEAR)
- Watch error correction register (SUBCUD)
- Alarm minute register (ALARMWM)
- Alarm hour register (ALARMWH)
- Alarm week register (ALARMWW)

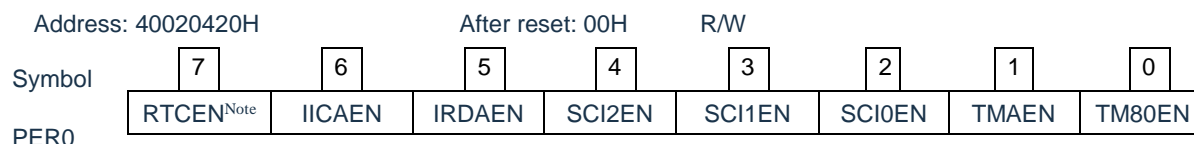


### 7.3.1 Peripheral enable register 0 (PER0)

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.

When the real-time clock is used, be sure to set bit 7 (RTCEN) of this register 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".

Figure7-2: Format of peripheral enabled register 0 (PER0)



RTCEN	Provides control of the input clock for the real-time clock (RTC) and 15-bit interval timer
0	Stop providing input clock. <ul style="list-style-type: none"> <li>The SFR used by the Real Time Clock (RTC) and the 15-bit interval timer cannot be written.</li> <li>The real-time clock (RTC) and 15-bit interval timer are in the reset state.</li> </ul>
1	Provides input clock. <ul style="list-style-type: none"> <li>The SFR used by the Real Time Clock (RTC) and the 15-bit interval timer can be read and written.</li> </ul>

#### Notice:

- To use the real time clock, you must first set the RTCEN bit to "1" while oscillation of the count clock ( $F_{RTC}$ ) 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 value is the initial value (except for the real time clock selection register (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)
  - Watch error correction register (SUBCUD)
  - Alarm minute register (ALARMWM)
  - Alarm hour register (ALARMWH)
  - Alarm week register (ALARMWW)
- It is possible to stop providing subsystem clock to peripheral functions other than real-time clock and 15-bit interval timer in deep sleep mode or sleep mode running by setting RTCLPC bit of subsystem clock supply mode control register(OSMC) to "1".

## 7.3.2 Real-time clock selection register (RTCCL)

The count clock ( $F_{RTC}$ ) of the real-time clock and the 15-bit interval timer can be selected via RTCCL.

Figure7-3: Format of real-time clock selection register (RTCCL)

Address: 0x40040C0C			After reset: 00H			R/W			
Symbol	7	6	5	4	3	2	1	0	
RTCCL	RTCCL7	RTCCL6	RTCCL5	0	0	0	RTCCKS1	RTCCKS0	0

RTCCL7	Selection of clock source for real time clock, counting clock for 15-bit interval timer
0	Select high-speed system clock ( $F_{MX}$ )
1	Select high-speed internal oscillator ( $F_{HOCO}$ )

RTCCKS1	RTCCKS0	RTCCL6	RTCCL5	Selection of operating clocks for real-time clocks, counting clocks for 15-bit interval timer
0	0	x	x	Sub-system Clock ( $F_{SUB}$ )
0	1			Low-speed internal oscillator clock ( $F_{IL}$ ) (WUTMMCK0 must be set to 1).
1	0	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

### 7.3.3 Real-time clock control register 0 (RTCC0)

This is an 8-bit register that sets the start or stop of the real-time clock, the control of the RTC1HZ pin, the 12/24-hour system, and the fixed cycle interrupt function.

The RTCC0 register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

Figure7-4: Format of real-time clock control register 0 (RTCC0)

Address: 0x4004240D		After reset: 00H		R/W				
Symbol	7	6	5	4	3	2	1	0
RTCC0	RTCE	0	RCLOE1 <small>Note</small>	0	AMPM	CT2	CT1	CT0

RTCE	Operation control of the real-time clock
0	Stop operation of counters.
1	Start operation of counters.

RCLOE1	Output control of the RTC1HZ pin
0	Disable the output of the RTC1HZ pin (1Hz).
1	Enable the output of the RTC1HZ pin (1Hz).

AMPM	Selection of 12-hour system/24-hour system
0	12-hour system (indicates morning or afternoon).
1	24-hour system
<ul style="list-style-type: none"> <li>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.</li> <li>The time bits are represented as shown in Table 8-2.</li> </ul>	

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.			

Notice:

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

Figure7-5: Format of real-time clock control register 1 (RTCC1)(1/ 2)

Address: 0x4004240E			After reset: 00H		R/W				
Symbol	7	6	5	4	3	2	1	0	
RTCC1	WALE	WALIE	0	WAFG	RIFG	0	RWST	RWAIT	

WALE	Operation control of the alarm clock
0	Consistent operation is invalid.
1	Consistent operation is valid.
To set the WALE bit when the counter is running (RTCE=1) and the WALIE bit is "1", you must set INTRTC to disable interrupt processing through the interrupt mask flag register before the rewrite and clear the WAFG flag and RTCIF flag after the rewrite. To set each alarm register (WALIE flag of RTCC1 register, alarm minute register (ALARMWM), alarm hour register (ALARMWH) and the alarm week register (ALARMWW)), the WALE bit must be set to "0" (invalid for consistent operation).	

WALIE	Operation control of the alarm clock interrupt (INTRTC) function
0	No consistent alarm interruptions.
1	Generate consistent alarm interruptions.

WAFG	Alarm clock detection status flag
0	The alarm clock is inconsistent.
1	Consistent alarms detected.
This is a status flag that indicates that an alarm clock has been detected consistently. It is only valid when the WALE bit is "1" and changes to "1" after one F <sub>RTC</sub> clock has elapsed and the alarm is detected. Clear this flag by writing "0" to it. Writing a "1" is not valid.	

Figure7-5: 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 up to 1 F <sub>RTC</sub> 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).



### 7.3.6 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".

Figure7-7: Format of second count register (SEC)

Address: 0x40042402		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 "7.3.18 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".

Figure7-8: Format of minute count register (MIN)

Address: 0x40042403		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 "7.3.18 Reading/writing real-time clock".



### 7.3.8 Hour count register (HOUR)

The HOUR register is an 8-bit register that takes a value of 00 to 23 or 01 to 12 and 21 to 32 (decimal) and indicates the count value of hours. It counts up when the minute counter overflows.

When data is written to this register, it is written to a buffer and then to the counter up to two cycles of  $F_{RTC}$  later. Even if the minute count register overflows while this register is being written, this register ignores the overflow and is set to the value written.

Specify a decimal value of 00 to 23, 01 to 12, or 21 to 32 by using BCD code according to the time system specified using bit 3 (AMPM) of real-time clock control register 0 (RTCC0).

If the AMPM bit value is changed, the values of the HOUR register change according to the specified time system. The HOUR register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "12H".

However, the value of this register is 00H if the AMPM bit is set to 1 after reset.

Figure7-9: Format of hour count register (HOUR)

Address: 0x40042404		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 "7.3.18Reading/writing real-time clock".

Table7-2 shows the relationship between the setting value of the AMPM bit, the hour count register (HOUR) value, and time.

Table7-2: 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".

Figure7-10: Format of day count register (DAY)

Address: 0x40042406	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 "7.3.18 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".

Figure7-11: Format of week count register (WEEK)

Address: 0x40042405			After reset: 00H		R/W			
Symbol	7	6	5	4	3	2	1	0
WEEK	0	0	0	0	0	WEEK4	WEEK2	WEEK1

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

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

Notice: When it reads or writes from/to the register while the counter is in operation ( $RTCE = 1$ ), follow the procedures described in "7.3.18 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".

Figure7-12: Format of month count register (MONTH)

Address: 0x40042407			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 "7.3.18 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".

Figure7-13: Format of year count register (YEAR)

Address: 0x40042408		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 "7.3.18 Reading/writing real-time clock".

### 7.3.13 Alarm minute register (ALARMWM)

This register is used to set minutes of alarm.

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

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.

Figure7-14: Format of alarm minute register (ALARMWM)

Address: 0x4004240A		After reset: 00H		R/W				
Symbol	7	6	5	4	3	2	1	0
AI ARMWM	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.

Figure7-15: Format of Alarm Hour Register (ALARMWH)

Address: 0x4004240B	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 manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

Figure7-16: Format of alarm week register (ALARMWW)

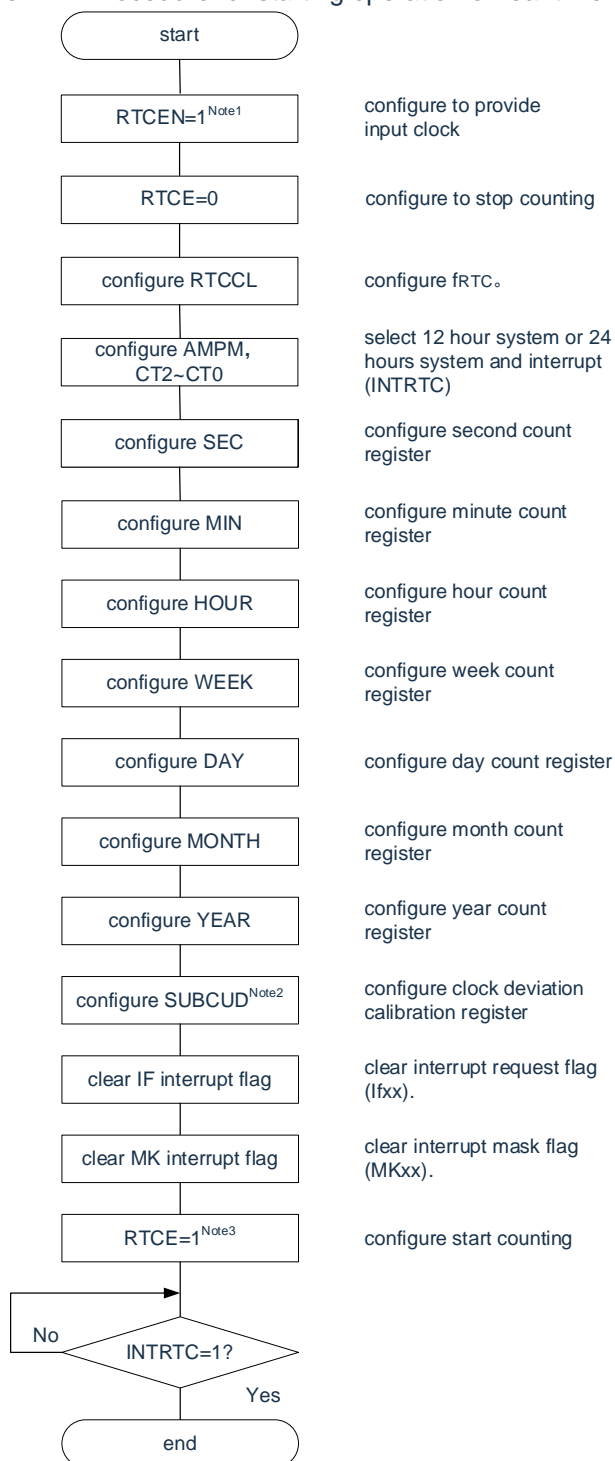
Address: 0x4004240C		After reset: 00H			R/W			
Symbol	7	6	5	4	3	2	1	0
ALARMWW	0	WW6	WW5	WW4	WW3	WW2	WW1	WW0

Here is an example of setting the alarm.

Time of Alarm	Day							12-Hour Display				24-Hour Display			
	Sund ay	Mond ay	Tues day	Wedn esda y	Thurs day	Frida y	Satur day	Hour 10	Hour 1	Minut e 10	Minut e 1	Hour 10	Hour 1	Minut e 10	Minut e 1
	W	W	W	W	W	W	W								
	0	1	2	3	4	5	6								
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 Starting operation of real-time clock

Figure7-17: Procedure for starting operation of real-time clock



Note 1: First set the RTCEN bit to 1, while oscillation of the count clock ( $F_{RTC}$ ) is stable.

Note 2: Set up the register only if the watch error must be corrected. For details about how to calculate the correction value, see “7.3.21 Example of watch error correction of real-time clock”.

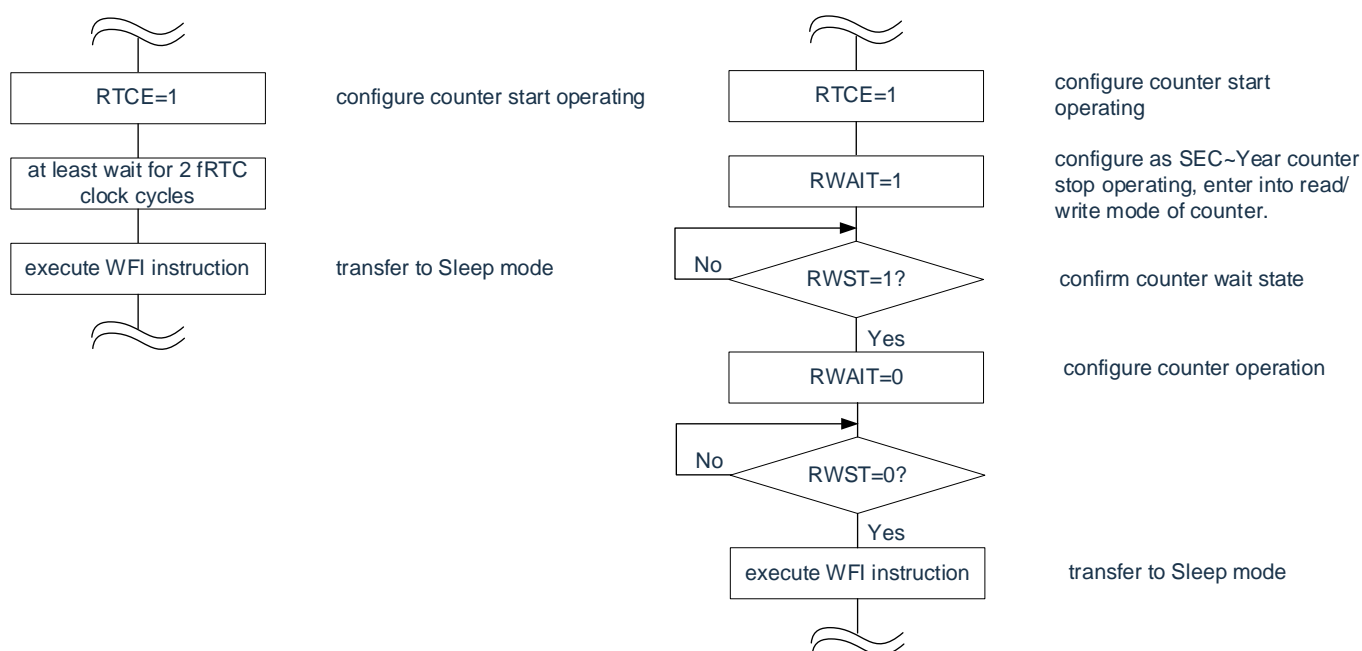
Confirm the procedure described in “7.3.17 Shifting to sleep mode after starting operation” when shifting to sleep mode without waiting for  $INTRTC = 1$  after  $RTCE = 1$ .

## 7.3.17 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).

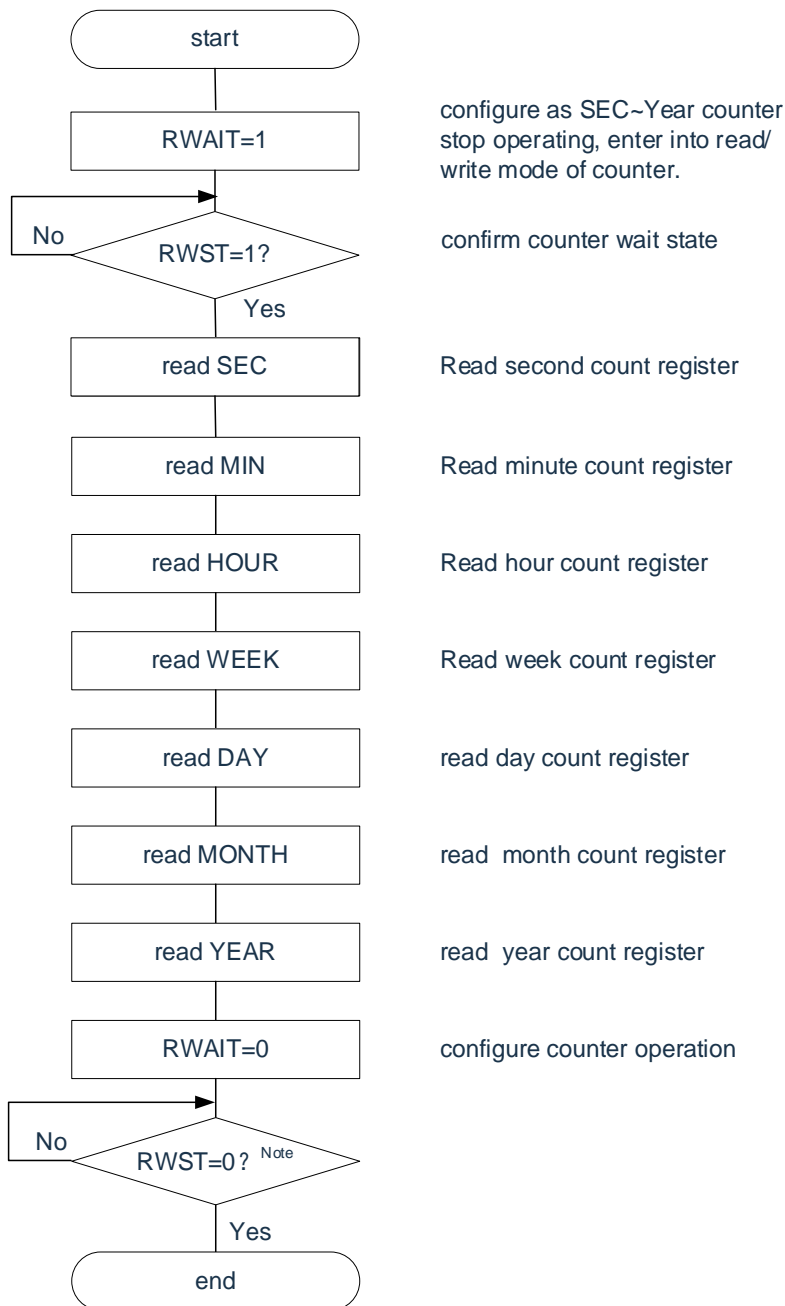
Figure 7-18: Procedure for shifting to sleep/deep sleep mode after setting RTCE bit to 1  
example1 Example 2



### 7.3.18 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.

Figure7-19: Procedure for reading real-time clock

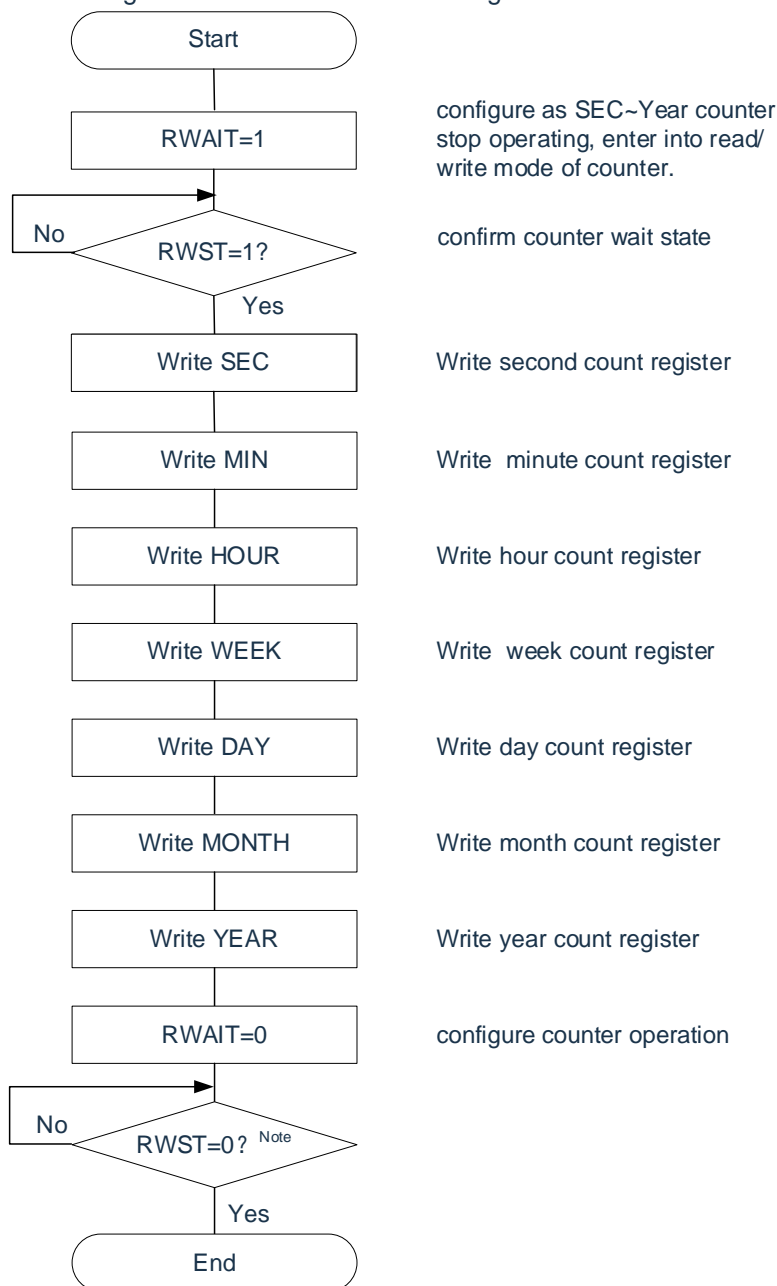


Note: Be sure to confirm that RWST = 0 before setting sleep mode.

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.

Figure7-20: Procedure for reading real-time clock



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

Notice:

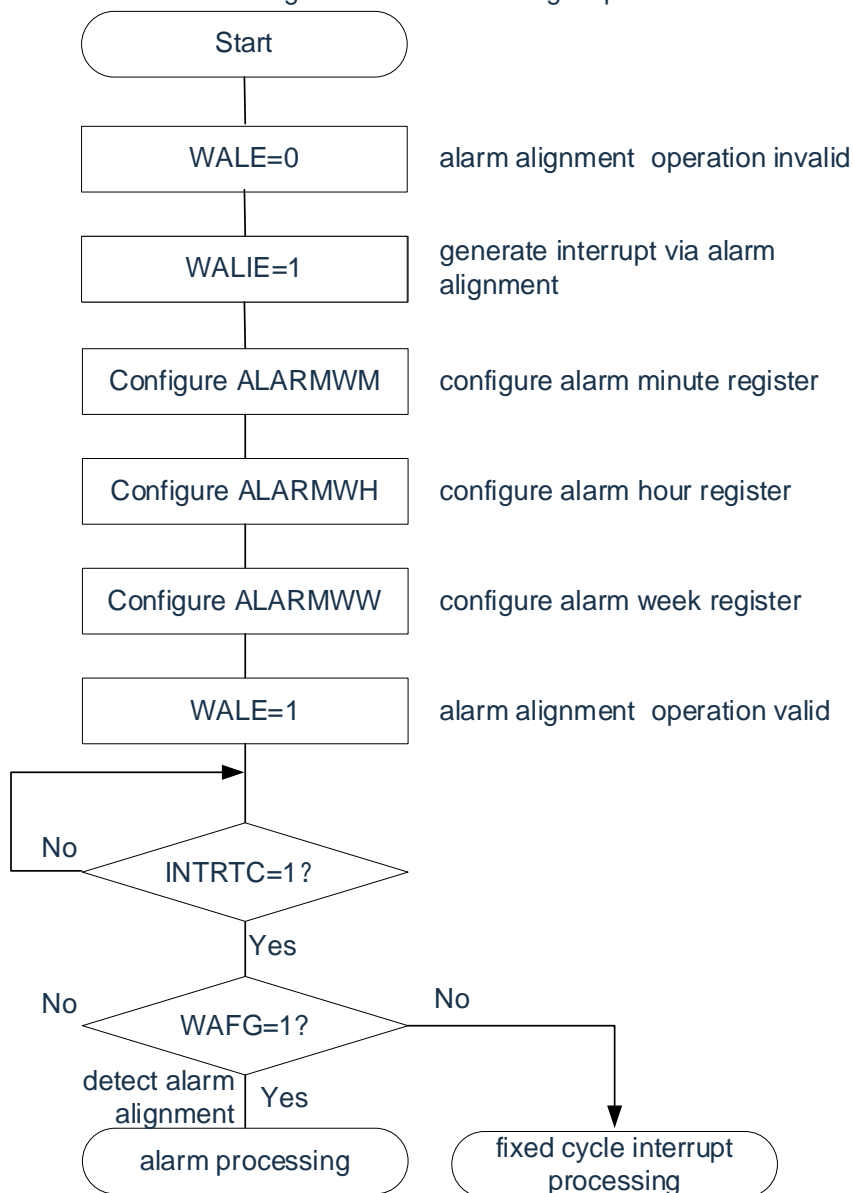
1. Notice: Complete the series of operations of setting the RWAIT bit to 1 to clearing the RWAIT bit to 0 within 1 second.
2. When changing the values of the SEC, MIN, HOUR, WEEK, DAY, MONTH, and YEAR register while the counter operates (RTCE = 1), rewrite the values after disabling interrupt servicing 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.

### 7.3.19 Setting alarm of real-time clock

Set alarm time after setting 0 to WALE (alarm operation invalid.) first.

Figure7-21: Alarm setting steps

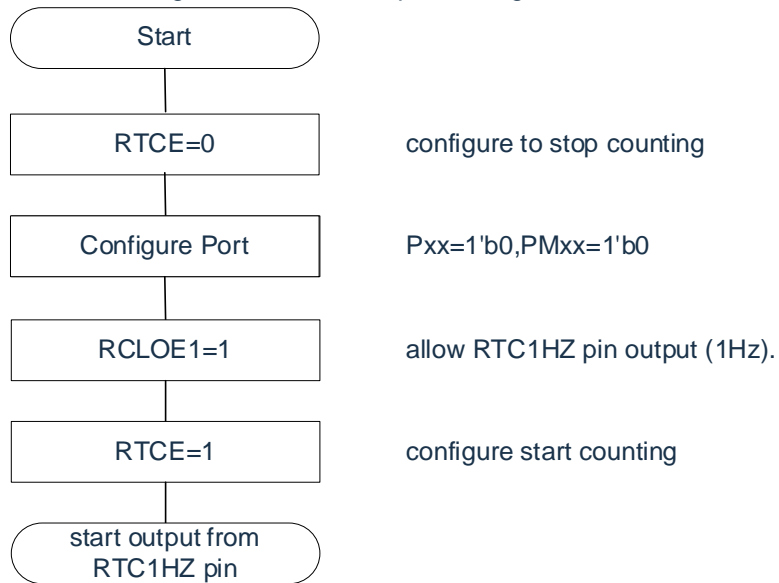


Remark:

1. There is no limit to the order of write operations for alarm minute registers (ALARMWM), alarm hour registers (ALARMWH), and alarm week registers (ALARMWW).
2. Fixed-cycle interrupts and alarm-consistent interrupts use the same interrupt source (INTRTC). When using these two types of interrupts at the same time, which interrupt occurred can be judged by checking the fixed-cycle interrupt status flag (RIFG) and the alarm detection status flag (WAFG) upon INTRTC occurrence.

### 7.3.20 1 Hz output of real-time clock

Figure7-22: 1 Hz Output Setting Procedure



Notice: First set the RTCEN bit to 1, while oscillation of the count clock ( $F_{SUB}$ ) is stable.

### 7.3.21 Example of watch error correction of real-time clock

The watch can be corrected with high accuracy when it is slow or fast, by setting a value to the watch error

Example of calculating the correction value

correction register.

The correction value used when correcting the count value of the internal counter (16-bit) is calculated by using the following equation: If a correctable range is -4165.6 ppm or lower and 4165.6 ppm or higher, set 0 to DEV.

(When DEV=0)

Correction value<sup>Note</sup>=Number of correction counts in 1 minute÷3=(Oscillation frequency÷Target frequency-1)×32768×60÷3

(When DEV=1)

Correction value<sup>Note</sup>=Number of correction counts in 1 minute=(Oscillation frequency÷Target frequency-1) ×32768×60

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

(When F12=0) Correction value ={(F11,F10,F9,F8,F7,F6,F5,F4,F3,F2,F1,F0)-1}×2

(When F12=1) Correction value =-{(/ F11,/ F10,/ F9,/ F8,/ F7,/ F6,/ F5,/ F4,/ F3,/ F2,/ F1,/ F0)+1}×2

When (F12~F0)=(\*,0,0,0,0,0,0,0,0,0,0,\*), watch 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 (F<sub>RTC</sub>), and can be calculated by the following equation:  
The output frequency of the RTC1HZ pin×32768 when the watch error correction register is set to its initial value (00H).
3. The target frequency is the frequency resulting after correction performed by using the watch 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 pinNote.

Note: For the setting of RTC1Hz output, please refer to "7.3. 20 1 Hz output of real-time clock".

[Calculating the correction value]

(When the output frequency of the RTC1HZ pin is 0.9999817Hz)

Oscillation frequency = 32768×0.9999817≈32767.4Hz

Suppose the target frequency is 32768Hz (32767.4Hz+18.3ppm) and DEV=1.

A equation for calculating the correction value when the DEV bit is "1" is applied.

$$\begin{aligned} \text{Correction value} &= \text{Number of correction counts in 1 minute} = (\text{Oscillation frequency} \div \text{Target frequency} - 1) \times 32768 \times 60 \\ &= (32767.4 \div 32768 - 1) \times 32768 \times 60 \\ &= -36 \end{aligned}$$

[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).

$$-((F11 \sim F0) - 1) \times 2 = -36$$

$$(F11 \sim F0) = 17$$

$$(F11 \sim F0) = (0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1)$$

$$(F11 \sim F0) = (1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0)$$

Therefore, the correction from 32767.4Hz to 32768Hz (32767.4Hz +18.3ppm) is as follows:

If by DEV=1 and correction value=-36 (bit12~0 of the SUBCUD register:1,1,1,1,1,1,1,1,0,1,1,1,0) to set the correction register, you can correct it to 32768Hz (0ppm).

# Chapter 8 15-bit interval timer

## 8.1 Functions 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.

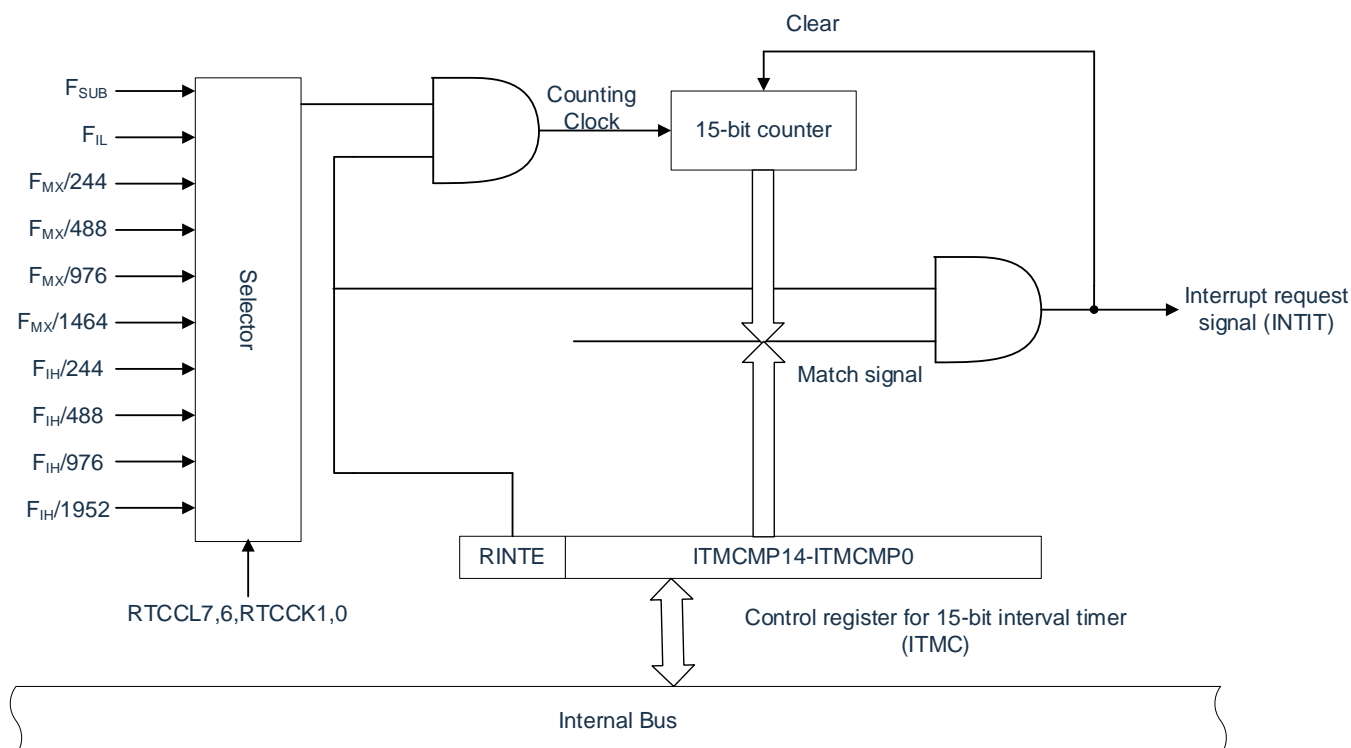
## 8.2 Structure of 15-bit Interval Timer

The 15-bit interval timer includes the following hardware.

Table8-1: Structure of 15-bit Interval Timer

Item	Structure
Counter	15-bit counter
Control register	Peripheral enable register 0 (PER0)
	Real-time clock selection register (RTCCL)
	15-bit interval timer control register (ITMC)

Figure8-1: Block Diagram of 15-bit Interval Timer



## 8.3 Registers controlling 15-bit Interval Timer

The 15-bit interval timer is controlled by the following registers.

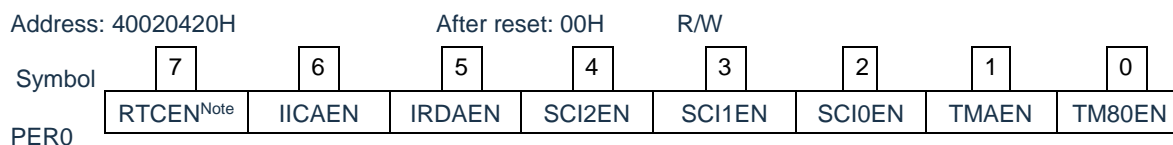
- Peripheral enable register 0 (PER0)
- Real-time clock selection register (RTCCL)
- 15-bit interval timer control register (ITMC)

### 8.3.1 Peripheral enable register 0 (PER0)

The PER0 register is 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.

When the 15-bit interval timer is used, be sure to set bit 7 (RTCEN) of this register 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 "00000000H".

Figure9-2: Format of peripheral enable register 0 (PER0)



RTCEN	Provides control of the input clock for the real-time clock (RTC) and 15-bit interval timer
0	Stop providing input clock. <ul style="list-style-type: none"> <li>• The SFR used by the Real Time Clock (RTC) and the 15-bit interval timer cannot be written.</li> <li>• The real-time clock (RTC) and 15-bit interval timer are in the reset state.</li> </ul>
1	Provides input clock. <ul style="list-style-type: none"> <li>• The SFR used by the Real Time Clock (RTC) and the 15-bit interval timer can be read and written.</li> </ul>

## 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.

Figure8-3: Format of real-time clock selection register (RTCCL)

Address: 0x40040C0C After reset: 00H R/W

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	Select high-speed system clock ( $F_{MX}$ )
1	Select high-speed internal 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	Sub-system Clock ( $F_{SUB}$ )
0	1			Low-speed internal oscillator clock ( $F_{IL}$ ) (WUTMMCK0 must be set to 1).
1	0	0	1	Main clock $F_{MAX}/F_{HOCO}$ (selected via RTCCL7)/1952
1	0	0	0	Main clock $F_{MAX}/F_{HOCO}$ (selected via RTCCL7)/1464
1	0	1	0	Main clock $F_{MAX}/F_{HOCO}$ (selected via RTCCL7)/976
1	1	0	0	Main clock $F_{MAX}/F_{HOCO}$ (selected via RTCCL7)/488
1	1	1	0	Main clock $F_{MAX}/F_{HOCO}$ (selected via RTCCL7)/244

### 8.3.3 15-bit interval timer control register (ITMC)

This register is used to set up the starting and stopping of the 15-bit interval timer operation and to specify the timer compare value.

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

After a reset signal is generated, the value of this register becomes "7FFFH".

Figure8-4: Format of 15-bit interval timer control register (ITMC)

Address: 0x40042400 After reset: 7FFFH R/W

Symbol 15 14~0

ITMC	RINTE	ITCMP14~ITCMP0
------	-------	----------------

RINTE	15-bit interval timer operation control
0	Count operation stopped (count clear)
1	Start operation of counters.

ITCMP14~ITCMP0	Specification of the 15-bit interval timer compare value
0001H	These bits generate “an interrupt at the fixed cycle, count clock cycles×(ITCMP set value+1)”.
• • •	
7FFFH	
0000H	Settings are disabled.
Example interrupt cycles when 001H or 7FFFH is specified for ITCMP14~ITCMP0	
<ul style="list-style-type: none"><li>ITCMP14~ITCMP0=0001H, count clock: F<sub>SUB</sub>=32.768KHz 1/32.768[KHz]×(1+1)=0.06103515625[ms]≈61.03[us]</li><li>ITCMP14~ITCMP0=7FFFH, count clock: F<sub>SUB</sub>=32.768KHz1/32.768[KHz]×(32767+1)=1000[ms]</li></ul>	

Notice:

- 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.
- The value read from the RINTE bit is applied one count clock cycle after setting the RINTE bit.
- 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.

- 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 15-bit interval timer operation

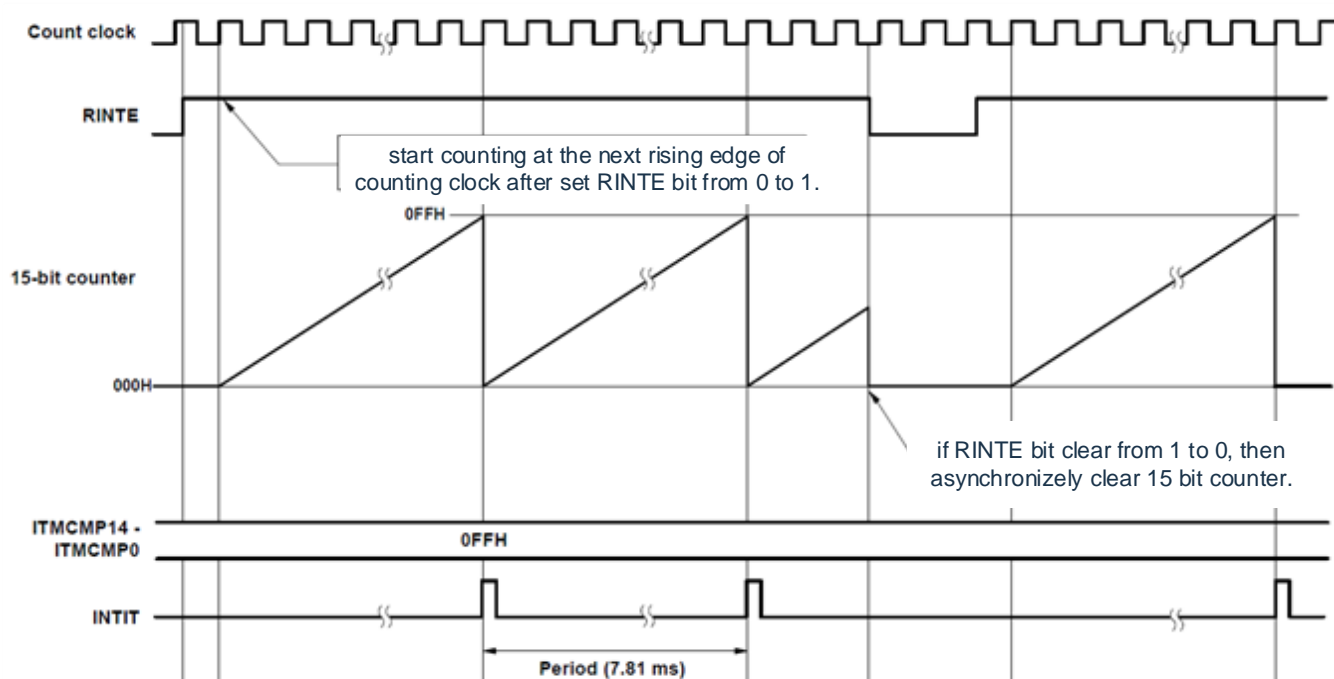
### 8.4.1 15-bit interval timer operation timing

The count value specified for the ITCMP14 to ITCMP0 bits is used as an interval to operate an 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 Figure8-5.

Figure8-5: 15-bit interval timer operation timing  
(ITCMP14~ITCMP0=0FFH, count clock:  $F_{SUB}=32.768\text{KHz}$ )

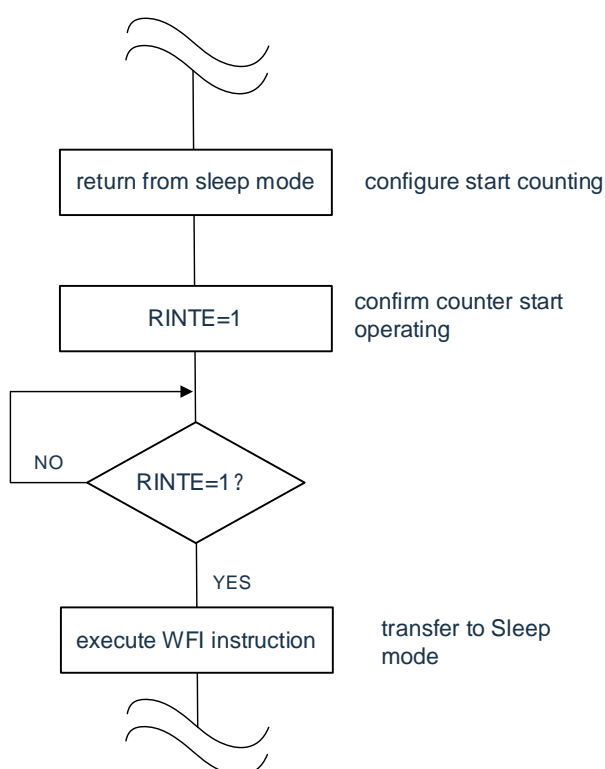


## 8.4.2 Start of count operation and re-enter to sleep mode after returned from sleep mode

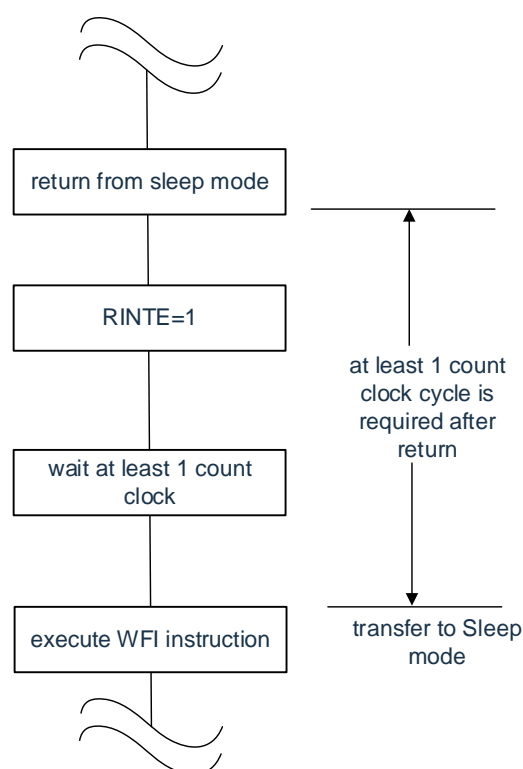
When setting the RINTE bit after returned from sleep mode and entering sleep mode again, write 1 to the RINTE bit, and confirm the written value of the RINTE bit is reflected or wait for at least one cycle of the count clock. Then, enter sleep mode.

- After setting RINTE to 1, confirm by polling that the RINTE bit has become 1, and then enter sleep mode (see Figure, Example 1).
- After setting RINTE to 1, wait for at least one cycle of the count clock and then enter sleep mode (see Figure, Example 2).

Example 1



Example 2



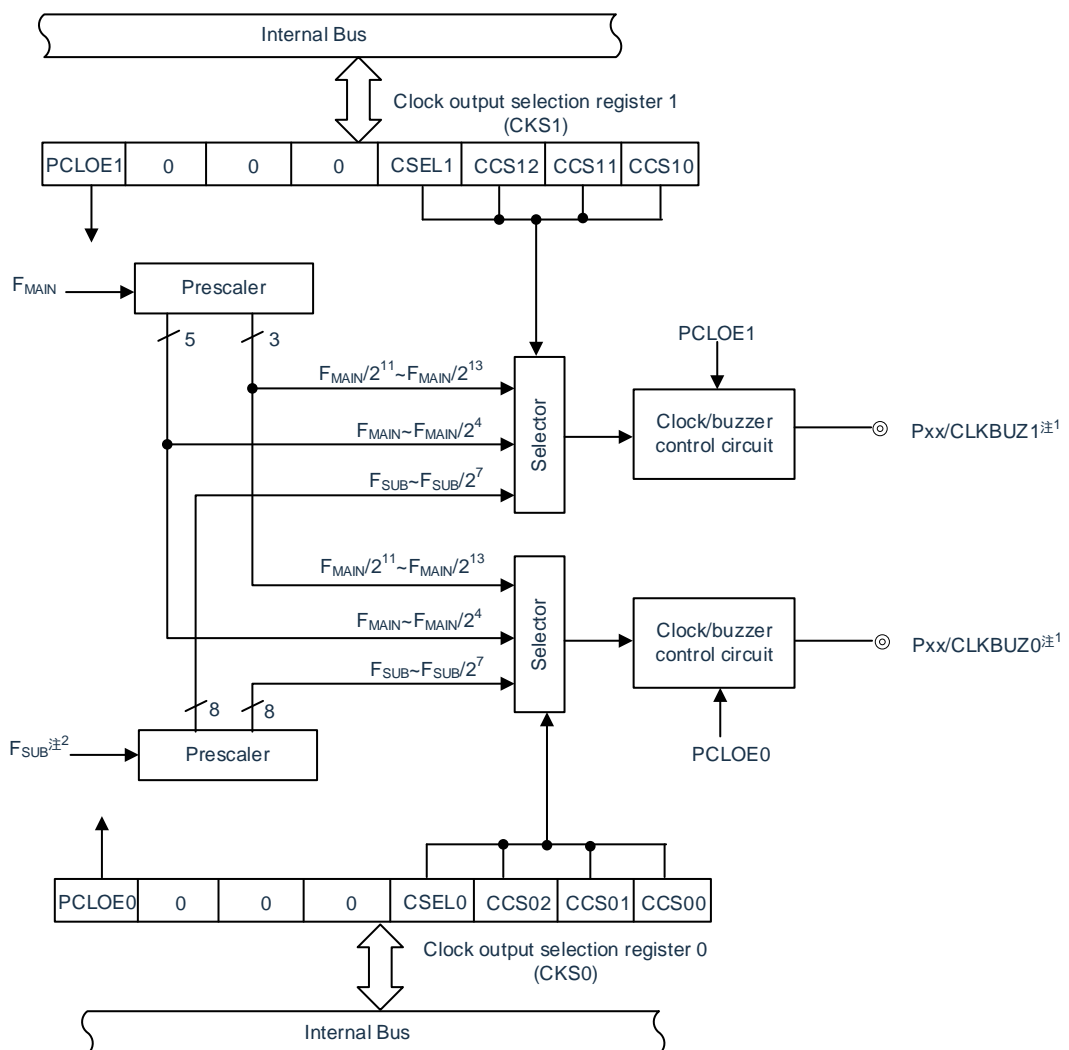
# Chapter 9 Clock Output/Buzzer Output Controller

## 9.1 Functions of clock output/buzzer output controller

The clock output controller is intended for clock output for supply to peripheral ICs. Buzzer output is a function to output a square wave of buzzer frequency.

This product has two clock output/buzzer output pins CLKBUZ0 and CLKBUZ1. 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. (n=0, 1)

Figure 9-1: Block diagram of the clock output/buzzer output controller





Note 1: For the frequencies that can be output from CLKBUZ0 and CLKBUZ1 pins, please refer to "AC Characteristics" in the data sheet.

Note 2: When the WUTMMCK0 bit of the OSMC register is set to "1", the actual output is  $F_{IL}$  when  $F_{SUB}$  is selected as the output clock for the clock output/buzzer output.

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 sleep mode where the CPU is operating with the subsystem clock ( $F_{SUB}$ ).

## 9.2 Registers for controlling clock output/buzzer output controller

Table9-1: Registers for controlling clock output/buzzer output controller

Item	Register list
Control register	Clock output select registers n (CKSn)

### 9.2.1 Clock output select registers 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".

Figure9-2: Format of clock output select register n (CKSn)

Address: 0x40041001 (CKS0), 0x40041000 (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: ( $n=0, 1$ )

$F_{MAIN}$  : Main system clock frequency

$F_{SUB}$  : Subsystem clock frequency

## 9.3 Operations 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).

- Operation as output pin

The CLKBUZn pin is output as the following procedure.

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).

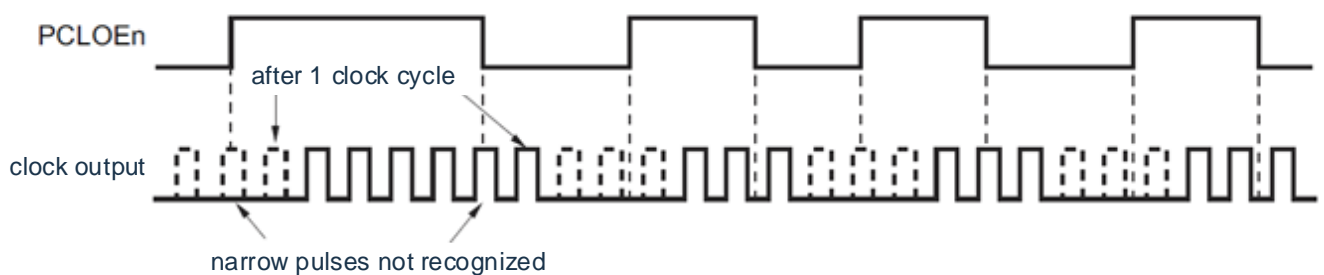
Select the output frequency with bits 0 to 3 (CCSn0 to CCSn2, CSELn) of the clock output select register (CKSn) of the CLKBUZn pin (output in disabled status).

Set bit 7 (PCLOEn) of the CKSn register to 1 to enable clock/buzzer output.

Remark:

1. The CLKBUZ0 is fixed multiplexed to PA00 port. When using the CLKBUZ0, it is not necessary to set the port multiplexing function configuration register (PxxCFG).
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. Figure9-3shows enabling or stopping output using the PCLOEn bit and the timing of outputting the clock.
3. n=0, 1

Figure9-3: CLKBUZn pin output clock timing



## 9.4 Cautions of 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 Functions of watchdog timer

The counting operation of the watchdog timer is set by the option byte (000C0H). The watchdog timer operates on the low-speed 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.

- If the watchdog timer counter overflows
- If a bit manipulation instruction is executed on the watchdog timer enable register (WDTE)
- If data other than “ACH” is written to the WDTE register
- If data is written to the WDTE register during a window close period

When a reset occurs due to the watchdog timer, bit 4 (WDTRF) of the reset control flag register (RESF) is set to “1”. For details of the RESF register, please refer to "Chapter 19 Reset Function". When  $75\%+1/2F_{IL}$  of the overflow time is reached, an interval interrupt can be generated.

## 10.2 Structure of watch dog timer

The watchdog timer includes the following hardware.

Table10-1: Structure of watch dog timer

Item	Structure
Counter	Internal counter (17-bit)
Control register	Watchdog timer enable register (WDTE)

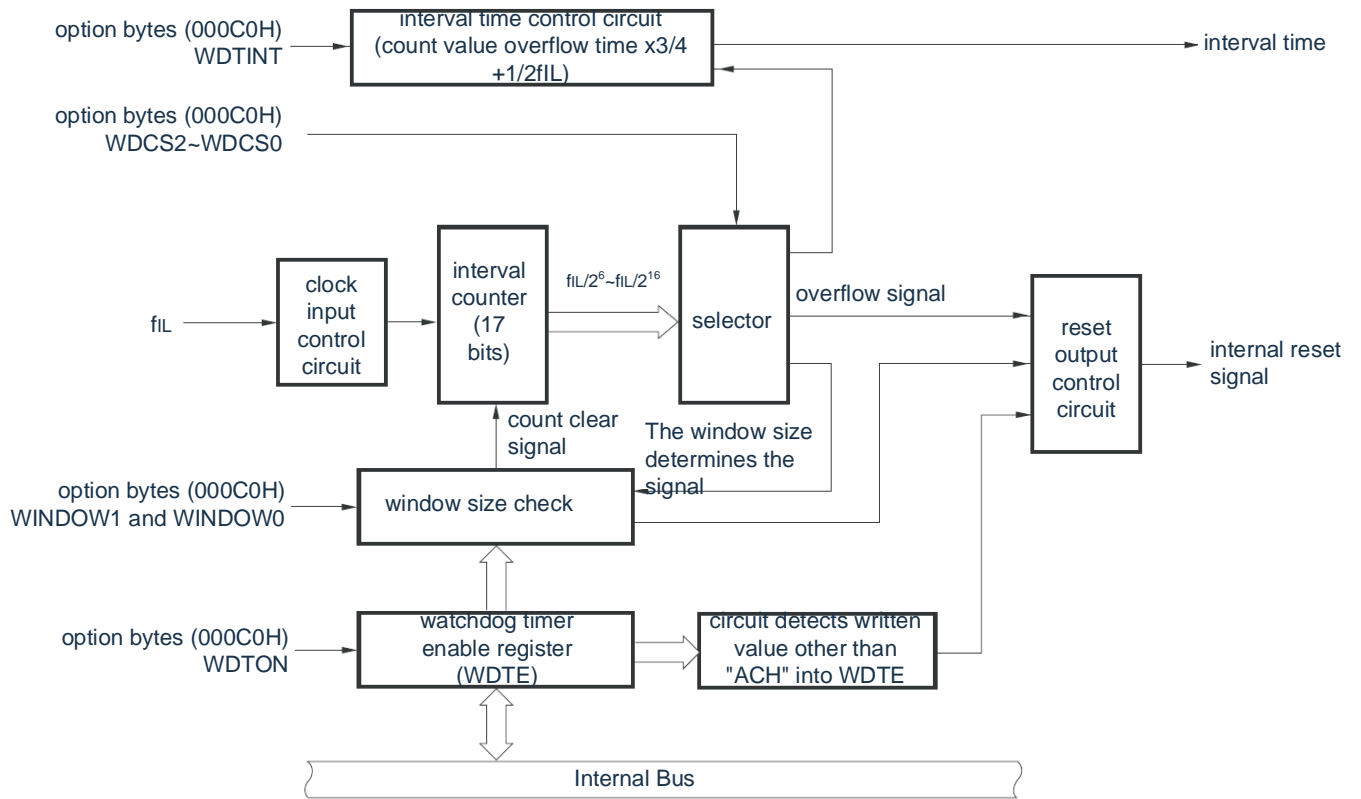
The operation of the counter is controlled by means of option bytes as well as setting the overflow time, the window opening period and the interval interruption.

Table10-2: 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	Bits 6 and 5 (WINDOW1, WINDOW0)
Controlling counter operation of watchdog timer	bit4 (WDTON)
Overflow time of watchdog timer	bit3~1(WDCS2~WDCS0)
Controlling counter operation of watchdog timer (in sleep mode)	bit0 (WDSTBYON)

Remark: For the option byte, see“Chapter 31 Option Byte”.

Figure10-1: Block diagram of watchdog timer



Remark:  $F_{IL}$  : Clock frequency of the low-speed internal oscillator

## 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”<sup>Note</sup>.

Figure10-2: Format of watchdog timer enable register (WDTE)

Address: 0x40021001 After reset: 9AH/1AH<sup>Note</sup> R/W

WDTE							
------	--	--	--	--	--	--	--

Note: The WDTE register reset value differs depending on the WDTON bit setting value of the option byte (000C0H). To operate watchdog timer, set the WDTON bit to 1.

WDTON bit setting value	WDTE register reset value
0 (watchdog timer count operation disabled)	1AH
1 (watchdog timer count operation enabled)	9AH

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) and its protection register (PRCR)

The LOCKCTL register is a configuration register for controlling the Cortex-M0+ LockUp function to operate the watchdog timer, and PRCR is its write-protect register.

The LOCKCTL, PRCR registers are set by an 8-bit memory manipulation instruction.

After generating a reset signal, the values of the LOCKCTL, PRCR register change to "00H".

Figure10-3: Format of LOCKUP control register (LOCKCTL) and its protection register (PRCR) (1/2)

Address: 40020405H After reset: 01H R/W

LOCKCTL	0	0	0	0	0	0	0	lockup_rst
---------	---	---	---	---	---	---	---	------------

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

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

PRCR	PRTKEY[7:1]	PRCR
------	-------------	------

PRCR	Write protection of LOCKUP controls register
0	• LOCKCTL register is not writable
1	• LOCKCTL register is writable

PRTKEY[7:	Write protection of PRCR
78H	• PRCR is writable
Others	• PRCR is not writable



### 10.3.3 WDTCFG configuration registers (WDTCFG0/1/2/3)

The WDTCFG configuration registers are the registers for whether to force the watchdog timer to operate.

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

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

Figure10-4: 10.3.3 WDTCFG configuration registers (WDTCFG0/1/2/3)

	Address: 40020408H	After reset: 00H	R/W
WDTCFG0	WDTCFG0		
	Address: 40020409H	After reset: 00H	R/W
WDTCFG1	WDTCFG1		
	Address: 4002040AH	After reset: 00H	R/W
WDTCFG2	WDTCFG2		
	Address: 4002040BH	After reset: 00H	R/W
WDTCFG3	WDTCFG3		

WDTCFG0	WDTCFG1	WDTCFG2	WDTCFG3	Configuration of watchdog timer
0x1A	0x2B	0x3C	0x4D	• Watchdog timer forced to stop
Others:				• Watchdog timer forced to start

## 10.4 Operation of watchdog timer

### 10.4.1 Operational control of watchdog timer

- 1) When using the watchdog timer, set the following items by option byte (000C0H):
  - 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 31 Option Byte for details).

WDTON	Counter of watchdog timer
0	Disable counting operation (stop counting after reset released)
1	Enable counting operation (start counting after release reset)

- The overflow time must be set by bit3~1 (WDCS2~WDCS0) of the option byte (000C0H) (for details, refer to 10.4.2 and Chapter 31 Option Byte).
  - The window opening period must be set by bit6 and bit5 (WINDOW1, WINDOW0) of the option byte (000C0H) (for details, please refer to 10.4.2 and Chapter 31 Option Byte).
- 2) After the reset is released, the watchdog timer starts counting.
  - 3) After starting counting and before the overflow time set by the option byte, writing "ACH" to the watchdog timer enable register (WDTE) clears the watchdog timer and starts counting again.
  - 4) Thereafter, writes to WDTE registers after the second time after the reset must be performed while the window is open. If you write the WDTE register while the window is closed, an internal reset signal is generated.
  - 5) If you do not write "ACH" to the WDTE register and exceed the overflow time, an internal reset signal is generated. An internal reset signal is generated if:
    - When a bit manipulation instruction is executed on a WDTE register
    - If data other than "ACH" is written to the WDTE register

Notice:

1. When data is written to the watchdog timer enable register (WDTE) for the first time after reset release, the watchdog timer is cleared in any timing regardless of the window open time, as long as the register is written before the overflow time, and the watchdog timer starts counting again.
2. After "ACH" is written to the WDTE register, an error of up to 2  $F_{IL}$  clocks may occur before the watchdog timer is cleared.
3. The watchdog timer can be cleared immediately before the count value overflows.
4. As shown below, the watchdog timer operates in sleep or deep sleep mode depending on the set value of bit0 (WDSTBYON) of the option byte (000C0H).

	WDSTBYON=0	WDSTBYON=1
Sleep mode	Stop operation of watchdog timer.	Continue operation of watchdog timer.
Deep sleep mode		

5. When the WDSTBYON bit is "0", restart the watchdog timer count after the sleep or deep sleep mode released. At this point, the counter is cleared to "0" and the count begins.

6. When operating with the X1 oscillation clock after releasing the deep sleep mode, the CPU starts operating after the oscillation stabilization time has elapsed.
7. Therefore, if the period between the deep sleep mode release and the watchdog timer overflow is short, an overflow occurs during the oscillation stabilization time, causing a reset. Consequently, set the overflow time in consideration of the oscillation stabilization time when operating with the X1 oscillation clock and when the watchdog timer is to be cleared after the deep sleep mode release by an interval interrupt.

## 10.4.2 Setting overflow time of watchdog timer

Set the overflow time of the watchdog timer by using bits 3 to 1 (WDCS2 to WDCS0) of the option byte (000C0H).

If an overflow occurs, an internal reset signal is generated. The present count is cleared and the watchdog timer starts counting again by writing “ACH” to the watchdog timer enable register (WDTE) during the window open period before the overflow time. The following overflow times can be set.

Table10-3: Setting of overflow time of watchdog timer

WDCS2	WDCS1	WDCS0	Overflow time of watchdog timer ( $F_{IL}=20\text{KHz}(\text{MAX.})$ )
0	0	0	$2^6/F_{IL}(3.2\text{ms})$
0	0	1	$2^7/F_{IL}(6.4\text{ms})$
0	1	0	$2^8/F_{IL}(12.8\text{ms})$
0	1	1	$2^9/F_{IL}(25.6\text{ms})$
1	0	0	$2^{11}/F_{IL}(102.4\text{ms})$
1	0	1	$2^{13}/F_{IL}(409.6\text{ms})$
1	1	0	$2^{14}/F_{IL}(819.2\text{ms})$
1	1	1	$2^{16}/F_{IL}(3276.8\text{ms})$

Remark:  $F_{IL}$  : Clock frequency of the low-speed internal oscillator.

### 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.

Figure10-4; 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:  
 $2^9/F_{IL}(\text{MAX.})=2^9/20\text{KHz}(\text{MAX.})=25.6\text{ms}$
- 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.

Table10-5: 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.

### 10.4.4.1 Operation of the 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~WDSC0) is set to 3'b010, that is, the overflow time is set to 12.8ms.

### 10.4.4.2 Operation of the watchdog timer without WDTCFG configured

When WDTCFG is not configured, the watchdog timer automatically starts operating, and the overflow time is determined by the overflow time control bit (WDSC2~WDSC0) of the option byte.

# Chapter 11 AD converter

## 11.1 Function of A/D converter

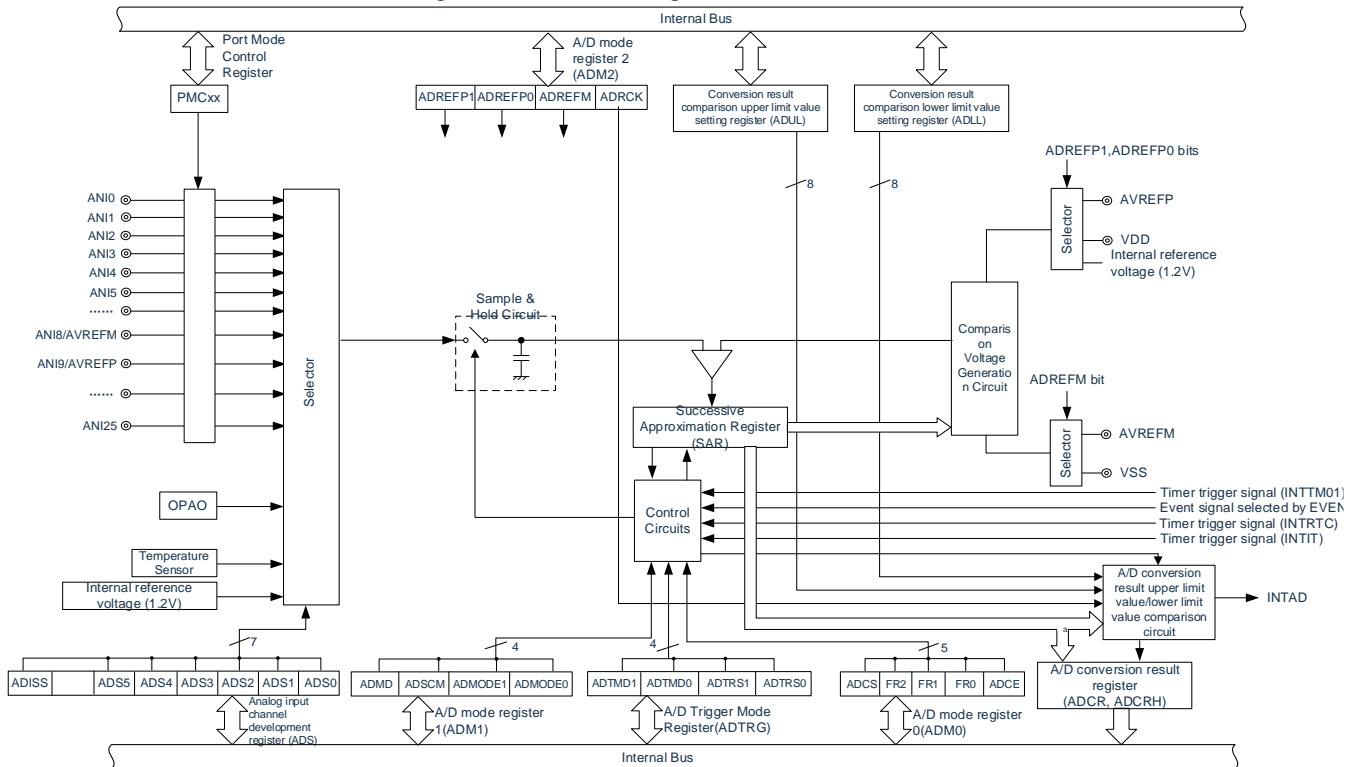
The A/D converter is used to convert analog input signals into digital values, including the following functions.

- 12-bit resolution A/D conversion
- Option to convert the analogue voltage of the external pins ANI0 to ANI25 input, the operational amplifier OPA output(OPAO), the internal reference voltage (1.45V) and the temperature sensor voltage
- An interrupt request (INTAD) is generated after each A/D conversion completion.

Various A/D conversion modes are set by the combination of modes as described below.

Trigger mode	Software triggered	Start the conversion with software operations.
	Hardware trigger no-wait mode	Conversion is started by detecting a hardware trigger.
	Hardware trigger wait mode	The power is turned on by detecting a hardware trigger while the system is off and in the conversion standby state, and conversion is then started automatically after the stabilization wait time passes.
Channel selection mode	Select mode	A/D conversion is performed on the analog input of one selected channel.
	Scan mode	A/D conversion is performed on the analog input of 4 channels in order. 4 consecutive channels can be selected from ANI0 to ANI14 as analog input channels.
		A/D conversion is performed on the analog input of 3 channels in order. 3 consecutive channels can be selected from ANI0 to ANI14 as analog input channels.
		A/D conversion is performed on the analog input of 2 channels in order. 2 consecutive channels can be selected from ANI0 to ANI14 as analog input channels.
Conversion mode	Single conversion mode	Perform 1 A/D conversion on the selected channel.
	Sequential conversion mode	Continuous A/D conversion of the selected channels until stopped by the software.
Sampling time	Sample clock 5.5 to 255 ADCLKs	The sampling time can be selected by the ADSMP register, which uses 13.5 conversion clocks ( $F_{AD}$ ) by default.

Figure11-1: Block Diagram of A/D converter



Remark: For the selection of analog input channel ANIx, please 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: PGCSC\_BASE=4002\_0800H;ADC\_BASE=4004\_5000H;PORT\_BASE=4004\_0000H

Register name	Register description	R/W	Reset value	Register address
PER1	Peripheral enable register 1	R/W	00H	PGCSC_BASE+1AH
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 converters 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
PMCn	Port mode control register	R/W	Note1	PORT_BASE+ <sup>Note1</sup>

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

Note 1: 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 (PER1)

The PER1 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, bit4 (ADCEN) must be set to "1".

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

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

Figure11-2: Table of peripheral enable register (PER1)

Address: 0x4002081AH	After reset: 00H				R/W			
Symbol	7	6	5	4	3	2	1	0
PER1	XX	XX	XX	ADCEN	XX	XX	XX	XX

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

Notice: 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 converters charge-discharge control register (ADNDIS)
- A/D converter sampling time extension control register (ADSMPWAIT)

## 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".

Figure11-3: Format of A/D converter mode register 0 (ADM0)

After reset: 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	Stop 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 <sup>Note 2</sup>
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 "Figure11-3".

Note 2: it takes 1  $\mu$ 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  $\mu$ 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.

### 11.2.3 A/D converter mode register 1 (ADM1)

This register sets the mode for A/D converter.

The ADM1 register is set by an 8-bit memory manipulation instruction.

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

Figure11-4: Format of A/D converter mode register 1 (ADM1)

After reset: 00H R

	7	6	5	4	3	2	1	0
ADM1	ADMD	SMODE1	SMODE0	0	ADSCM	0	ADMODE1	ADMODE0

ADMD	Setting of the A/D conversion channel selection mode
0	Select mode
1	Scan mode

SMODE1	SMODE0	Number of channels cycled during scan mode
0	0	4-channel scanning
0	1	3-channel scanning
1	0	2-channel scanning
Others:		Settings are disabled.

ADSCM	Setting of the A/D conversion mode
0	Sequential 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 27MHZ)
Others:		Settings are disabled.

Notice:

1. The bit 6~4, 2 must be set to "0".
2. Rewrite the value of the ADM1 register while conversion is stopped (ADCS = 0).
3. To complete A/D conversion, specify at least the following time as the hardware trigger interval:  
Hardware trigger no wait mode:  $2 F_{CLK}$  clock + A/D conversion time  
Hardware trigger wait mode:  $2 F_{CLK}$  clock + A/D power supply stabilization wait time + A/D conversion time
4. For A/D conversion, the number of 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. The fastest clock supported by ADCLK is 64MHz for high speed conversion mode and 27MHz for low current mode. For actual use, please configure the conversion mode and conversion clock frequency according to the "AC Characteristics" requirement in the data sheet.

Remark:  $F_{CLK}$ : Clock frequency of CPU/peripheral hardware

## 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".

Figure11-5: Format of A/D converter mode register 2 (ADM2) (1/3)

After reset: 00H R/W

	7	6	5	4	3	2	1	0
ADM2	ADREFP1	ADREFP0	ADREFM	0	ADRCK	0	CHRDE	0

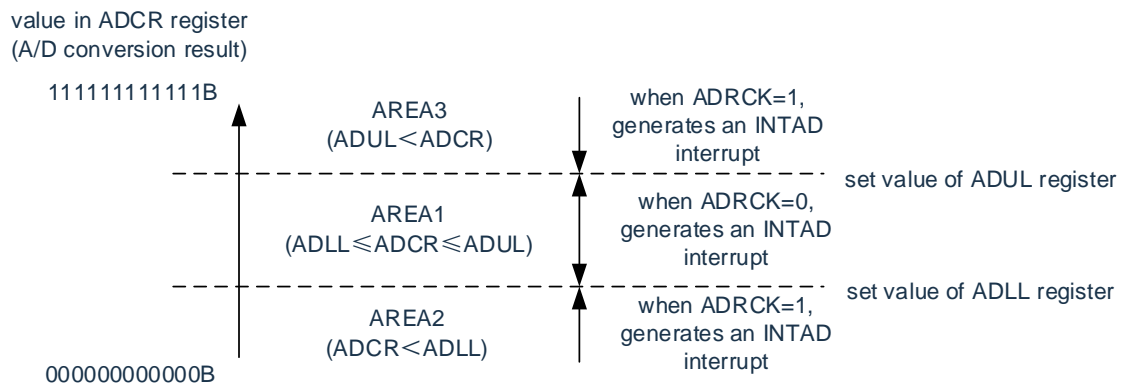
ADREFP1	ADREFP0	Selection of positive (+) reference voltage source of the A/D converter
0	0	Supplied from V <sub>DD</sub>
0	1	Supplied from AV <sub>REFP</sub> external terminals.
1	0	Supplied from the internal reference voltage (1.45 V)
1	1	Settings are disabled.

ADREFM	Selection of negative (-) reference voltage source of the A/D converter
0	Supplied from V <sub>SS</sub>
1	Supplied from AV <sub>REFM</sub> external terminals.

ADRCK	Checking the upper limit and lower limit conversion result values
0	When ADLL register ≤ ADCR register ≤ ADUL register (AREA1), the 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 interrupt signal (INTAD) generation range of AREA1~AREA3 is shown in Figure 14-8.	

CHRDE	Output enable for channel identification during A/D converter scan mode
0	In scan mode, the channel number is not identified in the conversion result
1	In scan mode, the upper four bits of the conversion result ([15:12] of the ADCR register) are the channel number of this result

Figure11-6: ADRCK bit interrupt signal generation range



Notice:

1. Rewrite the value of the ADM2 register while conversion is stopped (ADCS = 0).
2. When using AV<sub>REFP</sub> and AV<sub>REFM</sub>, the port must be set to analogue port mode(PMCxx=1).

Remark: When IND does not occur, the A/D conversion results are not saved to the ADCR registers and ADCRH registers.

## 11.2.5 A/D converter trigger mode register (ADTRG)

This is the register that sets the A/D conversion trigger mode and the hardware trigger signal.

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

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

Figure11-7: Format of A/D converter trigger mode register (ADTRG)

After reset: 00H R/W

	7	6	5	4	3	2	1	0
ADTRG	ADTMD1	ADTMD0	0	0	0	0	ADTRS1	ADTRS0

ADTMD1	ADTMD0	A/D conversion trigger mode selection
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	Timer channel 1 counts over or captures the end interrupt signal (INTTM01).
0	1	The event signal selected by the ELC
1	0	Real-time clock interrupt signal (INTRTC)
1	1	Interval timer interrupt signal (INTIT)

Notice:

1. To rewrite the ADTRG register, it must be done in conversion stopped state (ADCS=0, ADCE=0).
2. To complete A/D conversion, specify at least the following time as the hardware trigger interval:  
Hardware trigger no wait mode:  $2 F_{CLK}$  clock + A/D conversion time  
Hardware trigger wait mode:  $2 F_{CLK}$  clock + A/D power supply stabilization wait time + A/D conversion time

Remark:

1.  $F_{CLK}$ : Clock frequency of CPU/peripheral hardware
2. When using the output of the PLL as  $F_{CLK}$ , the hardware does not support the hardware-triggered wait mode because the frequency of  $F_{CLK}$  is uncertain and the hardware cannot calculate the 1us power wait time.

## 11.2.6 Analog input channel specification register (ADS)

This is the register that specifies the analog voltage input channel for A/D converter. The ADS register is set by an 8-bit memory manipulation instruction.

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

Figure11-8: Format of analog input channel specification register (ADS)

After reset: 00H R/W

	7	6	5	4	3	2	1	0
ADS	ADISS	0	ADS5	ADS4	ADS3	ADS2	ADS1	ADS0

- Selection mode (ADM1.ADMD=0)

THE SETTING VALUE OF ADS REGISTER		CH SELECTION	PIN NAME
ADISS	ADC[5:0]		
0	6'h00	ANI0	PA00
0	6'h01	ANI1	PA01
0	6'h02	ANI2	PA02
0	6'h03	ANI3	PA03
0	6'h04	ANI4	PA04
0	6'h05	ANI5	PA05
0	6'h06	ANI6	PA06
0	6'h07	ANI7	PA07
0	6'h08	ANI8	PB00
0	6'h09	ANI9	PB01
0	6'h0a	ANI10	PB02
0	6'h0b	ANI11	PB10
0	6'h0c	ANI12	PB11
0	6'h0d	ANI13	PB12
0	6'h0e	ANI14	PB13
0	6'h0f	ANI15	PB14
0	6'h10	ANI16	PB15
0	6'h11	ANI17	PC00
0	6'h12	ANI18	PC01
0	6'h13	ANI19	PC02
0	6'h14	ANI20	PC03
0	6'h15	ANI21	PC04
0	6'h16	ANI22	PC05
0	6'h17	ANI23	PC06
0	6'h18	ANI24	PD04
0	6'h19	ANI25	PD05
0	6'h23	OPAO	-
1	6'h00	BGR (temperature sensor0)	-
1	6'h01	BGR(1.45V)	-
Settings are disabled.			

Remark:

- 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.
- The analog input channels of the A/D converters vary by product. Please refer to the data sheet for detailed channel information.

- 4-channel scan mode (ADM1.ADMD=1)

ADISS	ADS[5:0]	Analog input channel			
		Scan 0	Scan 1	Scan 2	Scan 3
1'b0	6'h00	ANI0	ANI1	ANI2	ANI3
1'b0	6'h01	ANI1	ANI2	ANI3	ANI4
1'b0	6'h02	ANI2	ANI3	ANI4	ANI5
1'b0	6'h03	ANI3	ANI4	ANI5	ANI6
1'b0	6'h04	ANI4	ANI5	ANI6	ANI7
1'b0	6'h05	ANI5	ANI6	ANI7	ANI8
1'b0	6'h06	ANI6	ANI7	ANI8	ANI9
1'b0	6'h07	ANI7	ANI8	ANI9	ANI10
1'b0	6'h08	ANI8	ANI9	ANI10	ANI11
1'b0	6'h09	ANI9	ANI10	ANI11	ANI12
1'b0	6'h0A	ANI10	ANI11	ANI12	ANI13
1'b0	6'h0B	ANI11	ANI12	ANI13	ANI14
1'b0	6'h0C	ANI12	ANI13	ANI14	ANI15
Others:		Settings are disabled.			

- 3-channel scan mode (ADM1.ADMD=1)

ADISS	ADS[5:0]	Analog input channel		
		Scan 0	Scan 1	Scan 2
1'b0	6'h00	ANI0	ANI1	ANI2
1'b0	6'h01	ANI1	ANI2	ANI3
1'b0	6'h02	ANI2	ANI3	ANI4
1'b0	6'h03	ANI3	ANI4	ANI5
1'b0	6'h04	ANI4	ANI5	ANI6
1'b0	6'h05	ANI5	ANI6	ANI7
1'b0	6'h06	ANI6	ANI7	ANI8
1'b0	6'h07	ANI7	ANI8	ANI9
1'b0	6'h08	ANI8	ANI9	ANI10
1'b0	6'h09	ANI9	ANI10	ANI11
1'b0	6'h0A	ANI10	ANI11	ANI12
1'b0	6'h0B	ANI11	ANI12	ANI13
1'b0	6'h0C	ANI12	ANI13	ANI14
1'b0	6'h0D	ANI13	ANI14	ANI15
Others:		Settings are disabled.		



- 2-channel scan mode (ADM1.ADMD=1)

ADISS	ADS[5:0]	Analog input channel	
		Scan 0	Scan 1
1'b0	6'h00	ANI0	ANI1
1'b0	6'h01	ANI1	ANI2
1'b0	6'h02	ANI2	ANI3
1'b0	6'h03	ANI3	ANI4
1'b0	6'h04	ANI4	ANI5
1'b0	6'h05	ANI5	ANI6
1'b0	6'h06	ANI6	ANI7
1'b0	6'h07	ANI7	ANI8
1'b0	6'h08	ANI8	ANI9
1'b0	6'h09	ANI9	ANI10
1'b0	6'h0A	ANI10	ANI11
1'b0	6'h0B	ANI11	ANI12
1'b0	6'h0C	ANI12	ANI13
1'b0	6'h0D	ANI13	ANI14
1'b0	6'h0E	ANI14	ANI15
Others:		Settings are disabled.	

Notice:

- For the port set as analog input by PMCxx register, it can be designated as analog input for A/D conversion by ADS.
- To rewrite the ADISS bit, it must be done in the conversion stop state (ADCS=0, ADCE=0).
- When using AVREFP as the positive (+) reference voltage for the A/D converter, ANI9 cannot be selected as the A/D conversion channel.
- When using AVREFM as the negative (-) reference voltage for the A/D converter, ANI8 cannot be selected as the A/D conversion channel.
- After setting ADISS bit to "1", the first conversion result cannot be used. For the setting details, please refer to "11.5.4 Setup when temperature sensor output voltage/internal reference voltage is selected".
- The ADISS bit cannot be set to "1" when shifting to deep sleep mode or sleep mode while the CPU is operating on the subsystem 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)<sup>Note</sup>.

The high 4 bits of this register are fixed to "0" when the mode is selected, and the channel number of the conversion result can be configured by ADM2.CHRDE=1 in scan mode.

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

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

Note: If the value of the A/D conversion result is not within the set value range of the A/D conversion result comparison function (set by the ADRCK bit and the ADUL/ADLL register), the A/D conversion results are not saved.

Figure11-9: Format of 12-bit A/D conversion result register (ADCR)

After reset: 0000H R

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCR	ADCH3	ADCH2	ADCH1	ADCH0	ADCR[11:0]											

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.
- Selection mode (ADM1.ADMD=0)  
The readout value of ADCH0~3 is fixed at 4'b0000
  - Scan mode (ADM1.ADMD=1) and ADM2.CHRDE=1, the relationship between the readout values of ADCH0~3 and the converted channels is as follows

ADCH3	ADCH2	ADCH1	ADCH0	The 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<sup>Note</sup>.

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.

Figure11-10: Format of 8-bit A/D conversion result register (ADCRH)

After reset: 00H R

	7	6	5	4	3	2	1	0
ADCRH								

Notice: Read the conversion result following conversion completion before writing to the ADM0, ADS registers. Otherwise, you may not read the correct conversion results.

## 11.2.9 Conversion result comparison upper limit setting register (ADUL)

This register is used to specify the setting for checking the upper limit of the A/D conversion results.

The A/D conversion results and ADUL register value are compared, and interrupt signal (INTAD) generation is controlled in the range specified by the ADRCK bit of A/D converter mode register 2 (ADM2) (shown in

Figure11-6: ADRCK bit interrupt signal generation range). 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".

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.

Figure11-11: Format of conversion result comparison upper limit setting register (ADUL)

After reset: FFH R/W

	7	6	5	4	3	2	1	0
ADUL	ADUL7	ADUL6	ADUL5	ADUL4	ADUL3	ADUL2	ADUL1	ADUL0

## 11.2.10 Conversion result comparison lower limit setting register (ADLL)

This register is used to specify the setting for checking the lower limit of the A/D conversion results.

The A/D conversion results and ADLL register value are compared, and interrupt signal (INTAD) generation is controlled in the range specified by the ADRCK bit of A/D converter mode register 2 (ADM2) (shown in

Figure11-6: ADRCK bit interrupt signal generation range). 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".

Figure11-12: Format of conversion result comparison lower limit setting register (ADLL)

After reset: 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)

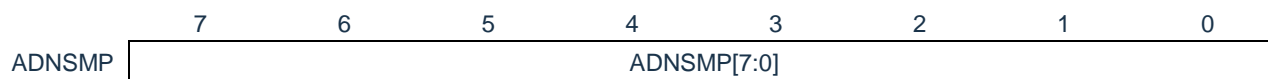
This register controls the A/D sampling time.

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

After a reset signal is generated, the value of this register becomes "0dH".

Figure11-13: Format of A/D converter sampling time control register (ADNSMP)

After reset: 0dH R/W



Number of sampling clocks set:

ADNSMP[7:0]	Sampling 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: Rewrite the value of the ADNSMP register while conversion is stopped (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<sub>AD</sub>

Low-current conversion mode: ADC conversion time = (number of sampling clocks + number of successive comparison clocks (40.5))/F<sub>AD</sub>

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.

The recommended sampling times for each conversion channel under different conditions are shown in the following table:

Sampling time calculation equations: Number of sampling clocks / F<sub>AD</sub> ≥ recommended sampling time

A/D conversion mode	AVDD[V]	AN1x[ns]	OPA/temperature sensors/internal reference voltage[ns]
High-speed conversion	4.5~5.5	211	633
	2.7~5.5	250	750
	2.4~5.5	422	1266
Low-current conversion	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 converter sampling time extension control register (ADSMPWAIT)

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

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

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

Figure11-14: Format of A/D converter sampling time extension control register (ADSMPWAIT)

After reset: 00H R/W

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

ADSMPWAIT	A/D conversion object
0	At "0", the A/D sampling time is set directly by the ADSMP register
1	The A/D sampling time is extended when "1" in any time, and when the sampling time is changed from "1" to "0", the sampling time is controlled by ADSMP.

Note: ADSMPWAIT=1 is set in the conversion stop state (ADCS=0), and ADSMPWAIT can be rewritten to "0" when (ADCS=1).

## 11.2.13 A/D test register (ADTES)

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

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

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

Figure 11-15: Format of A/D test register (ADTES)

After reset: 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	Usually converted
0	0	1	0-code self-diagnostic test
0	1	1	Half-code self-diagnostic test
1	0	1	Full-code self-diagnostic test
Others:			Settings are disabled.

Notice: The bit 7~3 must be set to "0".



## 11.2.14 A/D converters charge-discharge control register (ADNDIS)

This register is used to control the charge and discharge action and time of A/D converter

The ADNDIS register can be read and written by an 8-bit memory manipulation instruction.

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

Figure11-16: Format of A/D converters charge-discharge control register (ADNDIS)

After reset: 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	charge

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

Notice: It is forbidden to set the charge-discharge time to 1 ADCLK, that is, ADNDIS[3:0]=4'b0001

## 11.3 Input voltage and conversion result

The analogue input voltage at the analogue input pin (ANIX) and the theoretical A/D conversion result (12-bit A/D Conversion Result Register (ADCR)) are related by the following expressions.

$$ADCR = \text{INT}\left(\frac{V_{AIN}}{V_{REF}} \times 4096 + 0.5\right) \quad \text{or} \quad (ADCR - 0.5) \times \frac{V_{REF}}{4096} \leq V_{AIN} < (ADCR + 0.5) \times \frac{V_{REF}}{4096}$$

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

$V_{AIN}$ : Analog input voltage

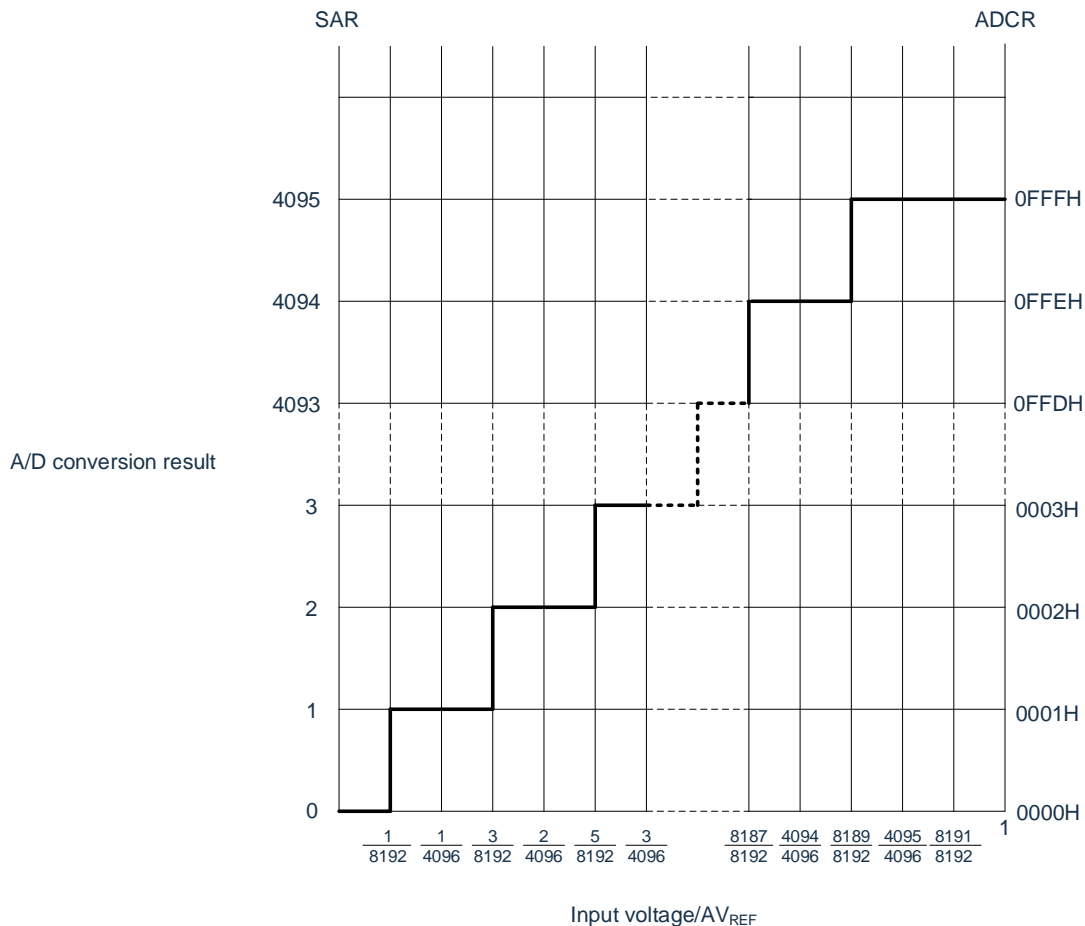
$V_{REF}$ :  $V_{REF}$  pin voltage

ADCR: The value of the A/D conversion result register (ADCR).

SAR: Successive approximation register

The analog input voltage and the A/D conversion result are shown in the following figure.

Figure11-17: Analog input voltage vs. A/D conversion result



Note:  $V_{REF}$  is the positive (+) reference voltage of the A/D converter, and  $V_{REFP}$  or  $V_{DD}$  can be selected.

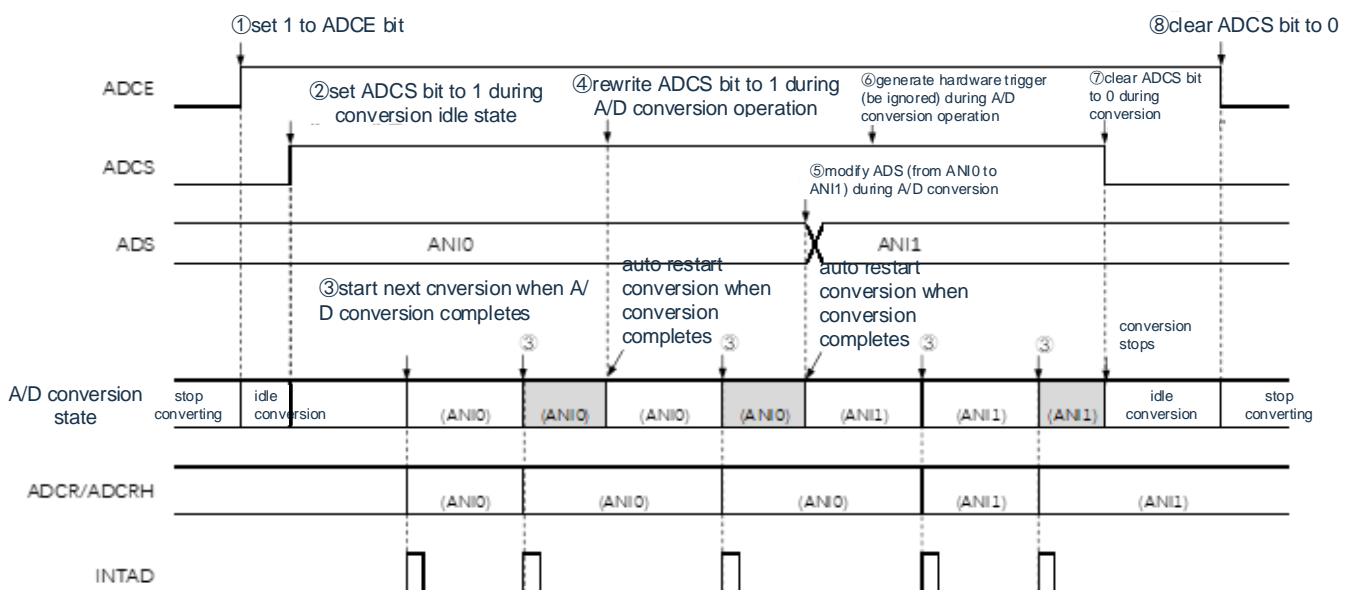
## 11.4 Operation mode of A/D converter

The A/D converter conversion operations are described below. For the setting of each mode, please refer to "11.5 A/D converter setup flowchart".

### 11.4.1 Software trigger mode (select mode, sequential conversion mode)

- ① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to "1".
- ② After the software counts up to the stabilization wait time (1  $\mu$ 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.

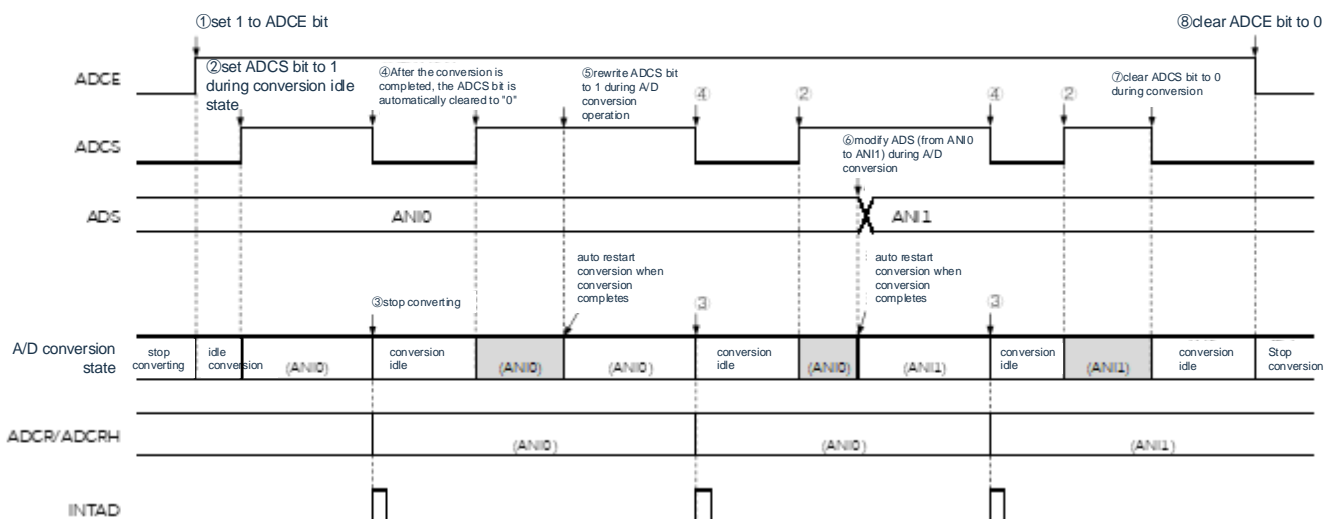
Figure11-18: Example of software trigger mode (select mode, sequential conversion mode) operation timing



## 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  $\mu$ 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.
- 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.

Figure11-19: Example of software trigger mode (select mode, single conversion mode) operation timing



- 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  $\mu$ s), 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.

The diagram illustrates the timing sequence for the A/D converter. The signals shown are ADCE, ADCS, ADS, A/D conversion state, ADCR/ADCRH, and INTAD.

**Sequence of Operations:**

- ① set 1 to ADCE bit
- ② set ADCS bit to 1 during conversion idle state
- ③ rewrite ADCS bit to 1 during A/D conversion operation
- ④ generate hardware trigger (be ignored) during A/D conversion operation
- ⑤ modify ADS (from ANI0 to ANI4) during A/D conversion
- ⑥ clear ADCS bit to 0 during conversion
- ⑦ clear ADCE bit to 0

**A/D conversion state:** stop converting, idle conversion, (ANI0), (ANI1), (ANI2), (ANI3), (ANI0), (ANI1), (ANI0), (ANI1), (ANI2), (ANI3), (ANI0), (ANI4), (ANI5), (ANI6), (ANI7), (ANI4), (ANI5), conversion idle, Stop conversion.

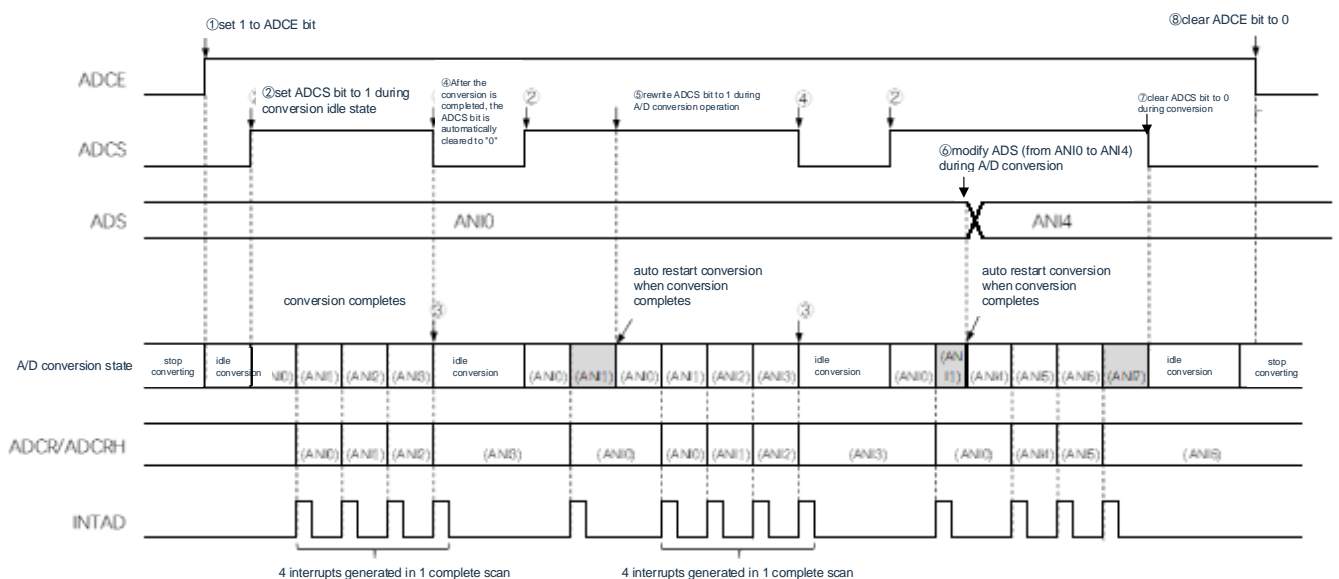
**ADCR/ADCRH:** (ANI0), (ANI1), (ANI2), (ANI3), (ANI0), (ANI0), (ANI1), (ANI2), (ANI3), (ANI0), (ANI4), (ANI5), (ANI6), (ANI7), (ANI4).

**INTAD:** Interrupts generated during conversion (4 interrupts generated in 1 complete scan).

## 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  $\mu$ s), 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.

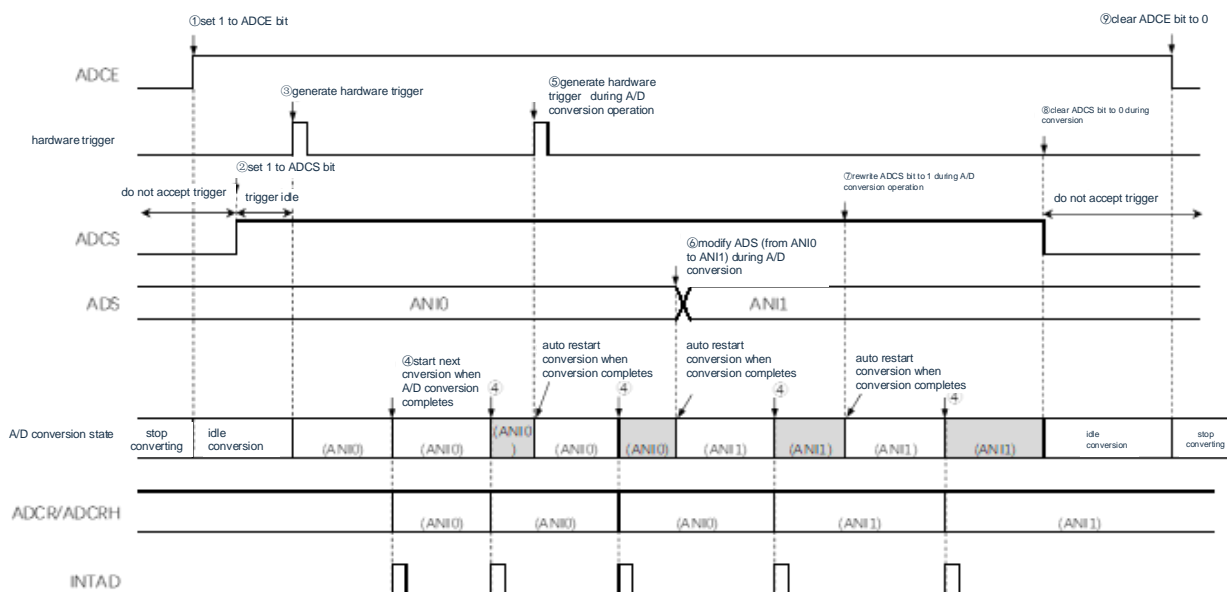
Figure11-21: Example of software trigger mode (scan mode, single conversion mode) operation timing



### 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  $\mu$ 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.

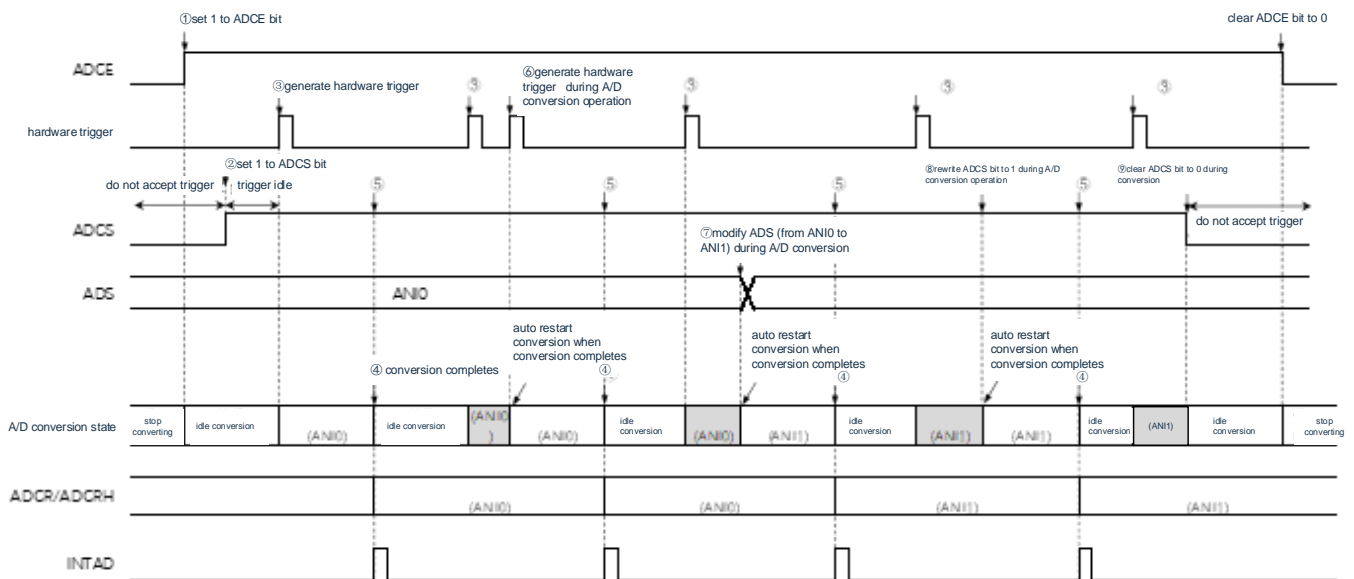
Figure11-22: Example of hardware trigger no-wait mode (select mode, sequential conversion mode) operation timing



## 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  $\mu$ 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.

Figure11-23: Example of hardware trigger no-wait mode (select mode, single conversion mode) operation timing

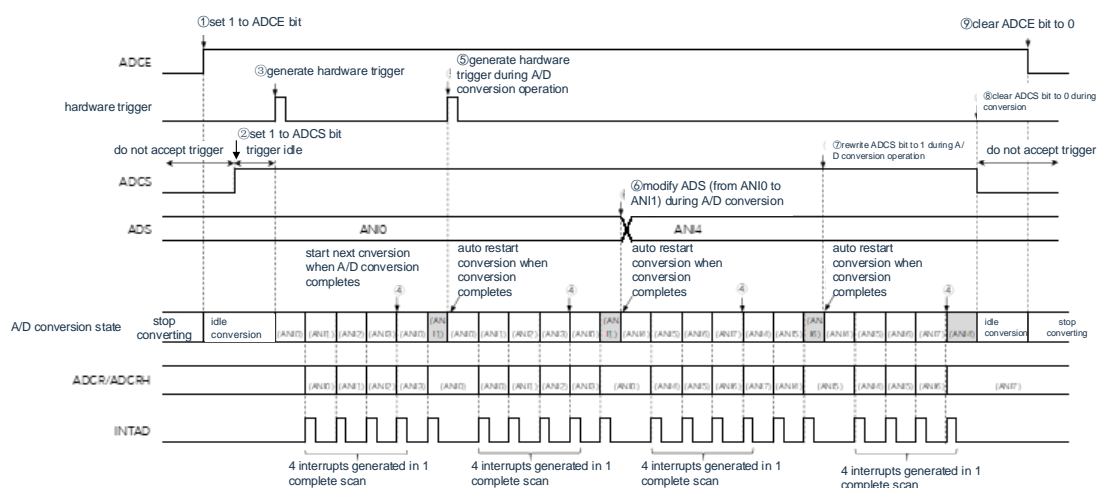




## 11.4.7 Hardware trigger no-wait mode (scan mode, sequential conversion mode)

- 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  $\mu$ 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.

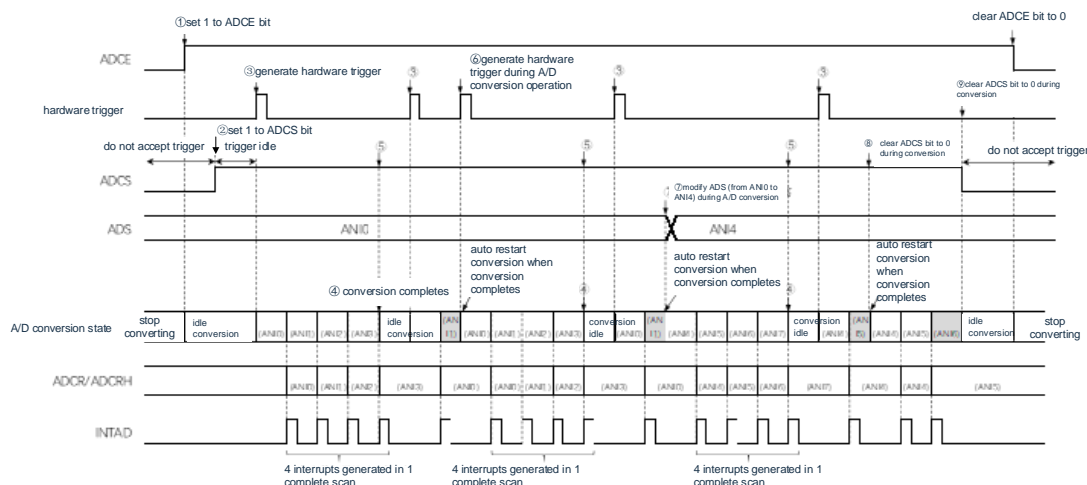
Figure11-24: Example of hardware trigger no-wait mode (scan mode, sequential conversion mode) operation timing



## 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  $\mu$ 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.

Figure11-25: Example of hardware trigger no-wait mode (scan mode, single conversion mode) operation timing

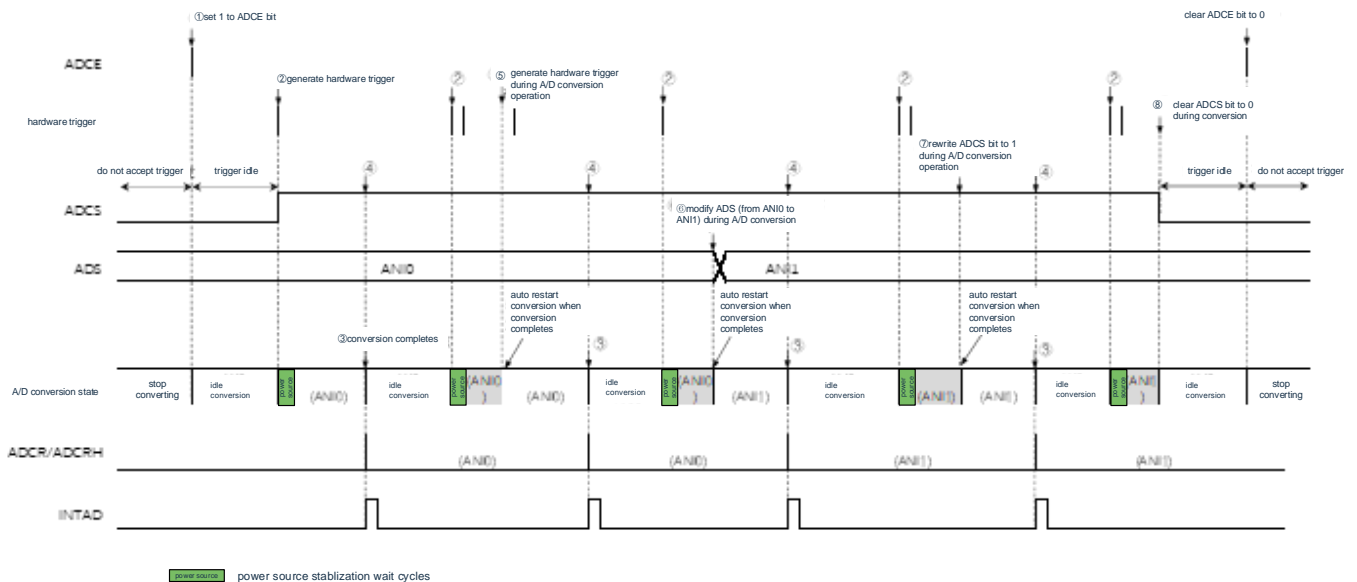





## 11.4.10 Hardware trigger wait mode (select mode, single conversion mode)

- 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.

Figure11-27: Example of hardware trigger wait mode (select mode, single conversion mode) operation timing

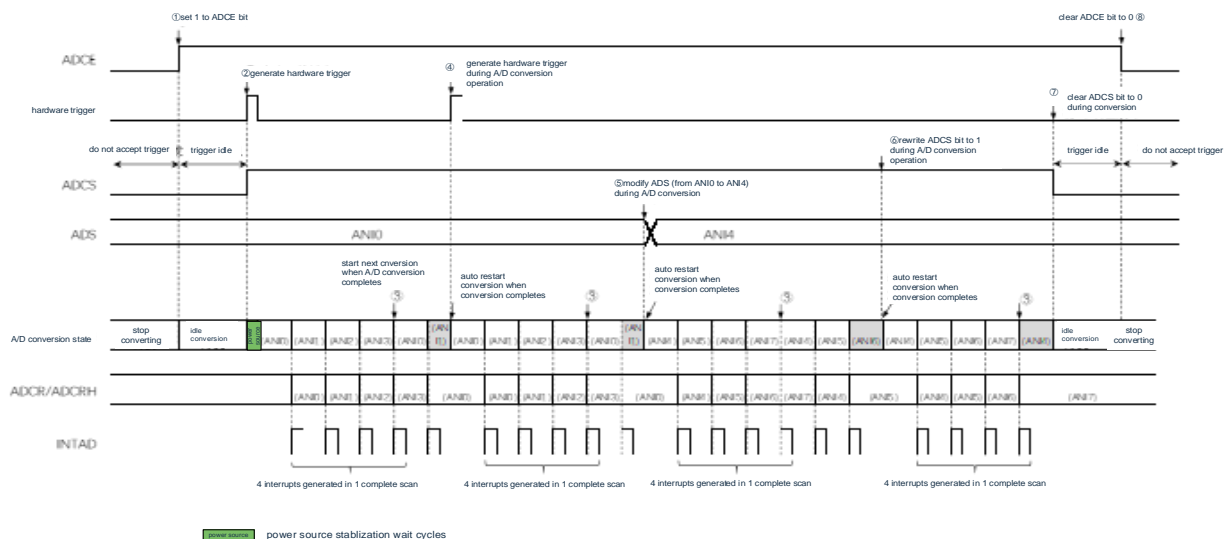


 电源安定等待时间

## 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.

Figure11-28: Example of hardware trigger wait mode (scan mode, sequential conversion mode) operation timing

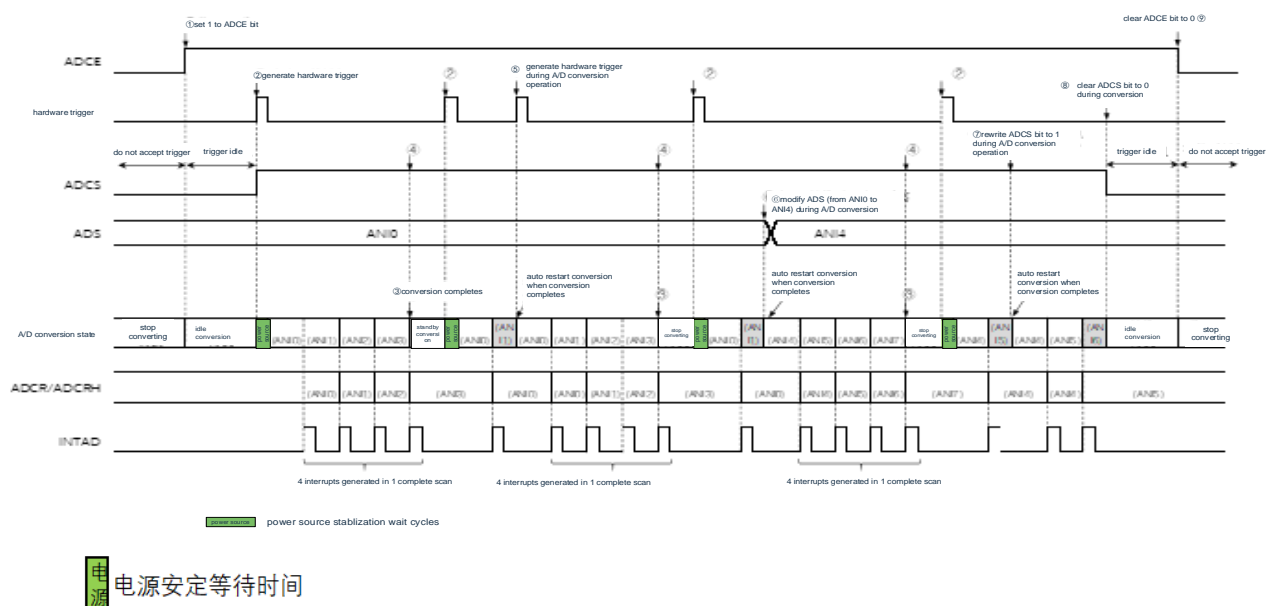


电源安定等待时间

## 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.

Figure11-29: Example of hardware trigger wait mode (scan mode, single conversion mode) operation timing

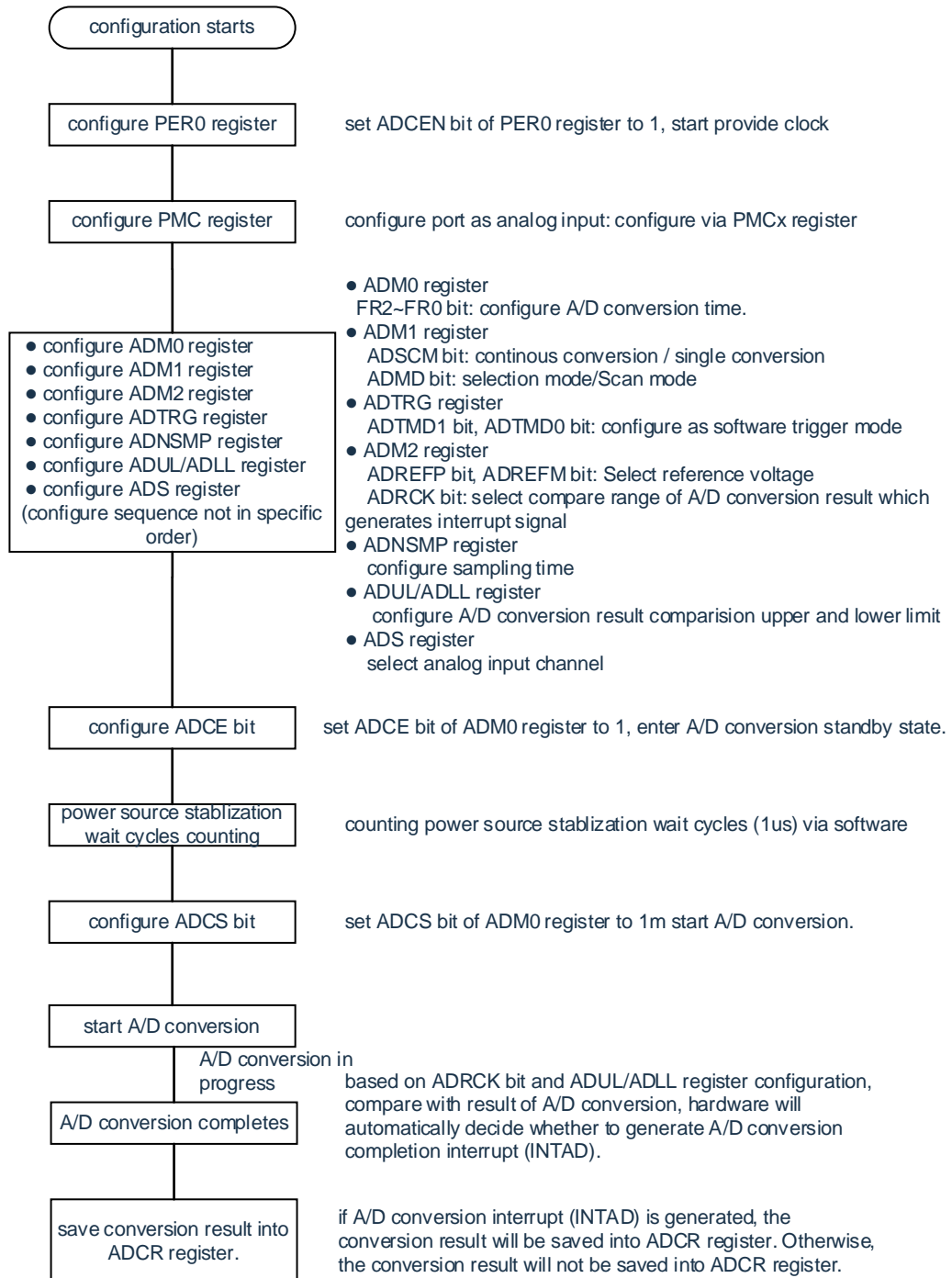


## 11.5 A/D converter setup flowchart

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

### 11.5.1 Setting up software trigger mode

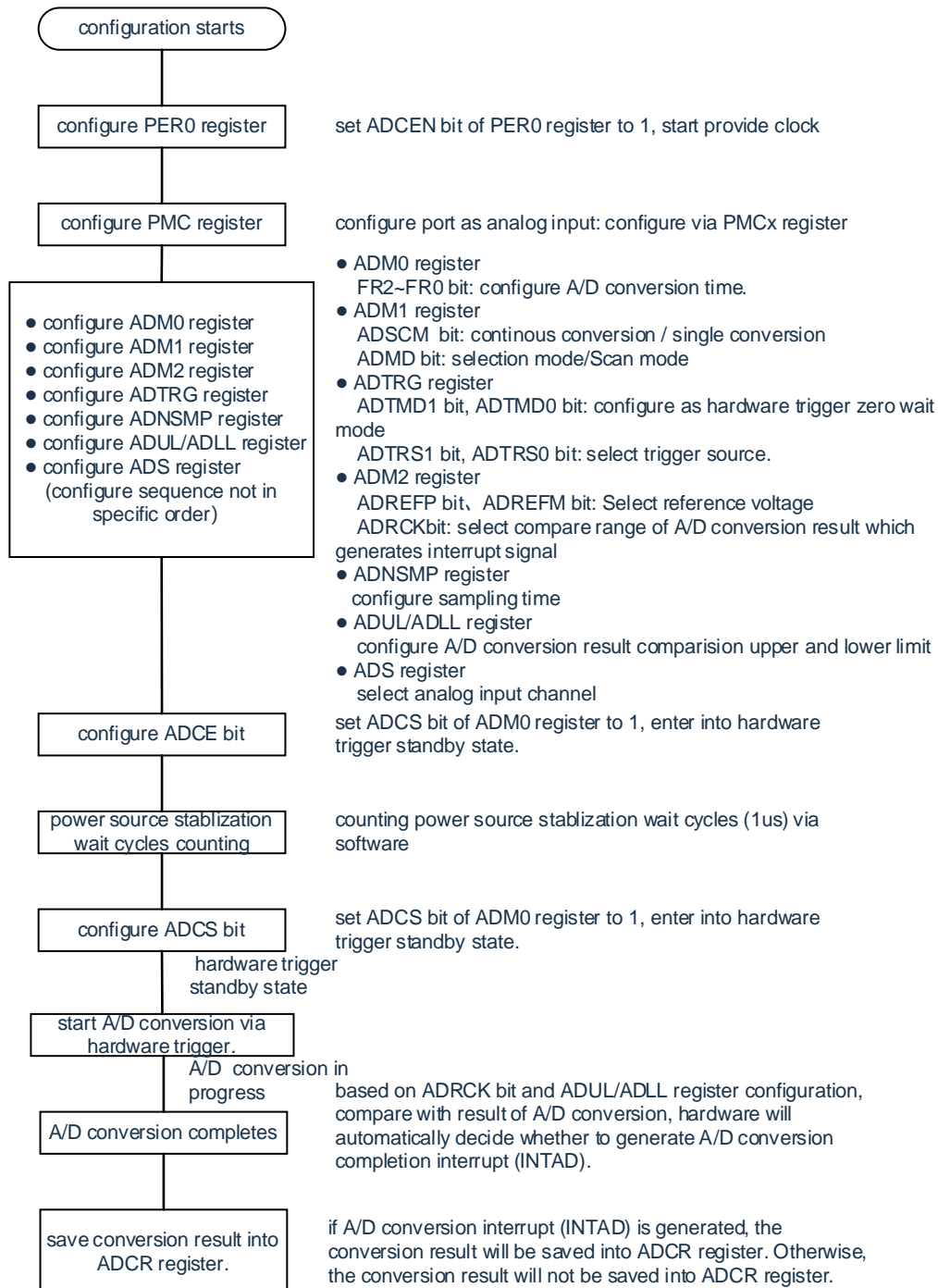
Figure11-30: Setting up software trigger mode





## 11.5.2 Setting up hardware trigger no-wait mode

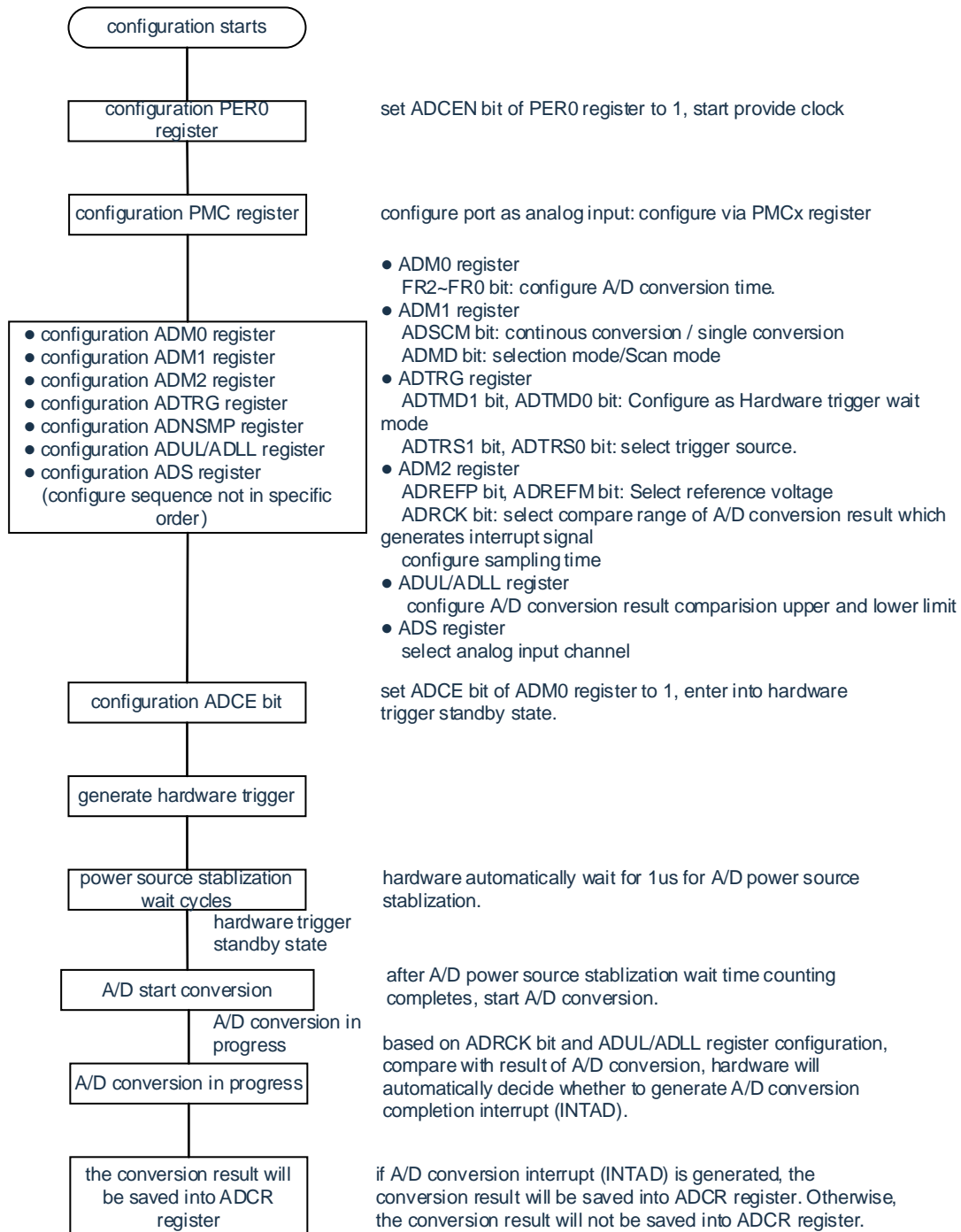
Figure11-31: Setting up hardware trigger no-wait mode





### 11.5.3 Setting up hardware trigger wait mode

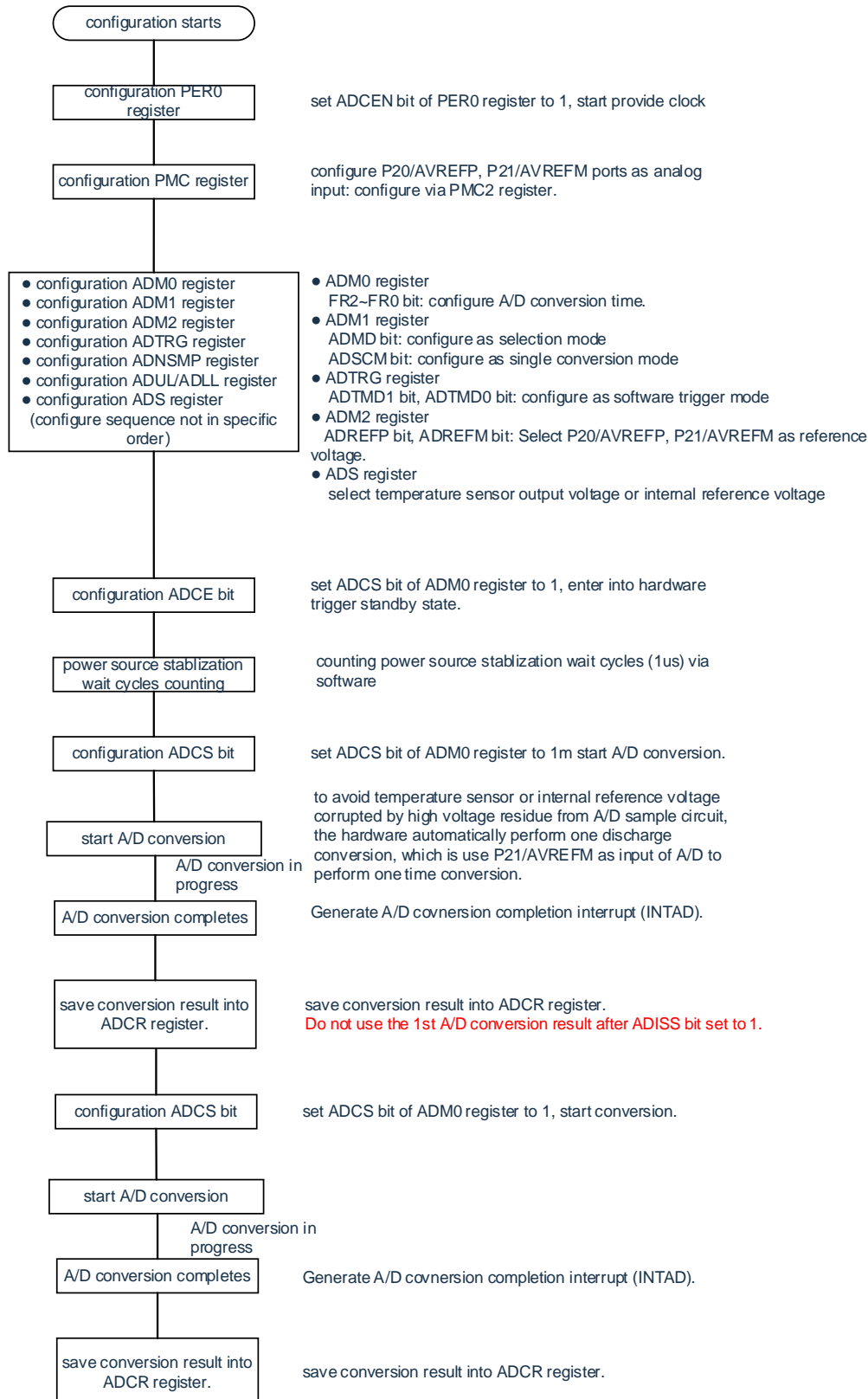
Figure11-32: Setting up hardware trigger wait mode



## 11.5.4 Setup when temperature sensor output voltage/internal reference voltage is selected

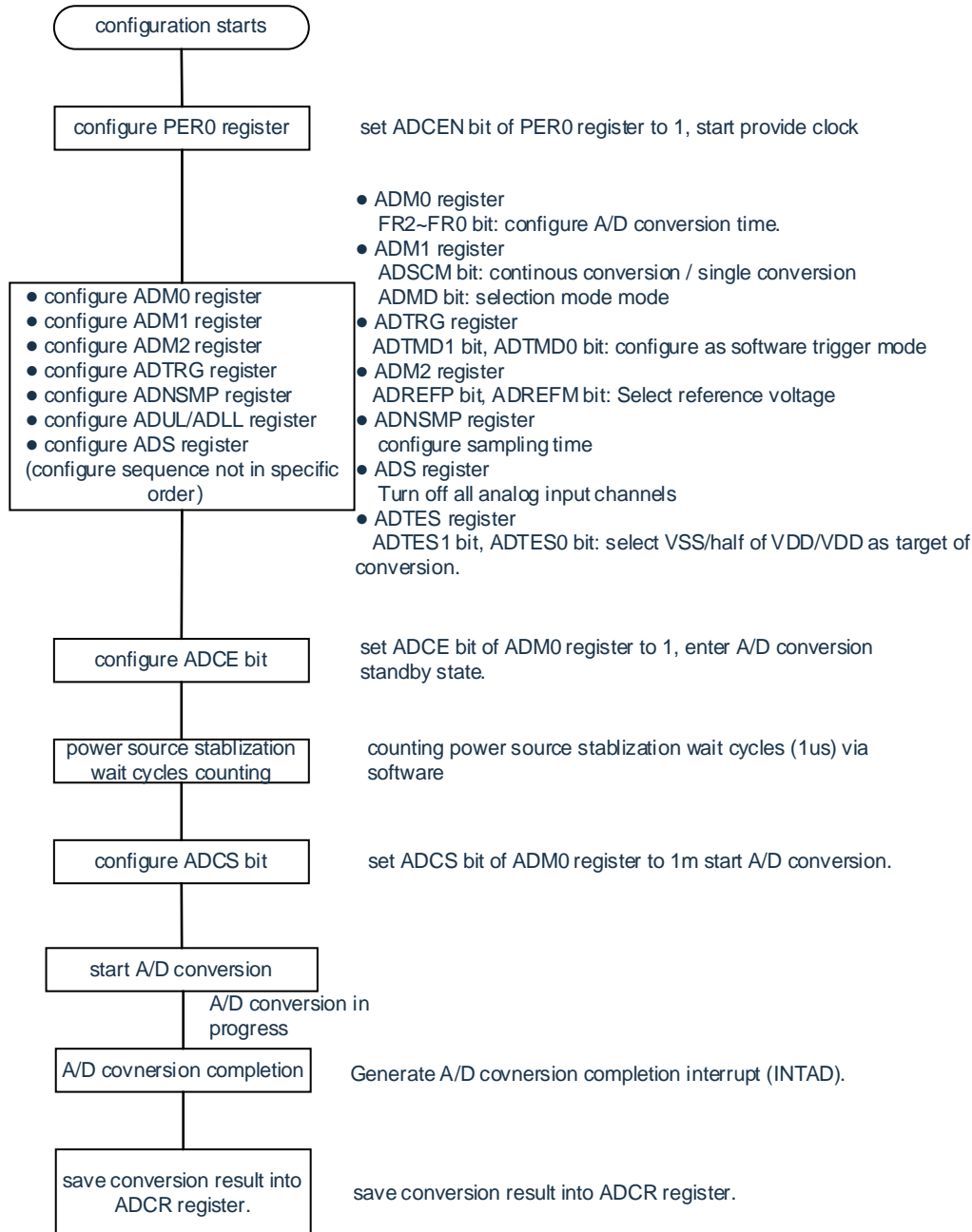
(example for software trigger mode and single conversion mode)

Figure11-33: Setup when temperature sensor output voltage/internal reference voltage is selected



## 11.5.5 Setting up test mode

Figure11-34: Setting up test mode ( $V_{SS}/half\_V_{DD}/V_{DD}$  as conversion object)



# Chapter 12 SIGMA-DELTA ADC

## 12.1 Overview

The 24-Bit Sigma-Delta ADC is a high-precision, low-power analog-to-digital conversion module. It supports one differential input channel, one built-in linear voltage regulator(LDO), temperature sensor and high-precision oscillator. The output capacity of LDO is 20 mA. The programmable gain amplifier (PGA) of Sigma-Delta ADC supports selectable gain options of 1, 2, 4, 8, 16, 32, 64, 128 and 256. The ADC output data rate in the normal mode of Sigma-Delta ADC is optional from 2.5Hz to 2.56KHz, and the default is 5Hz. The MCU internally communicates with the Sigma-Delta ADC via the 2-wire SPI interface SCLK, DRDYB/DOUT to configure it, e.g. channel selection, PGA gain settings, ADC ODR settings, etc.

## 12.2 Description of basic functions

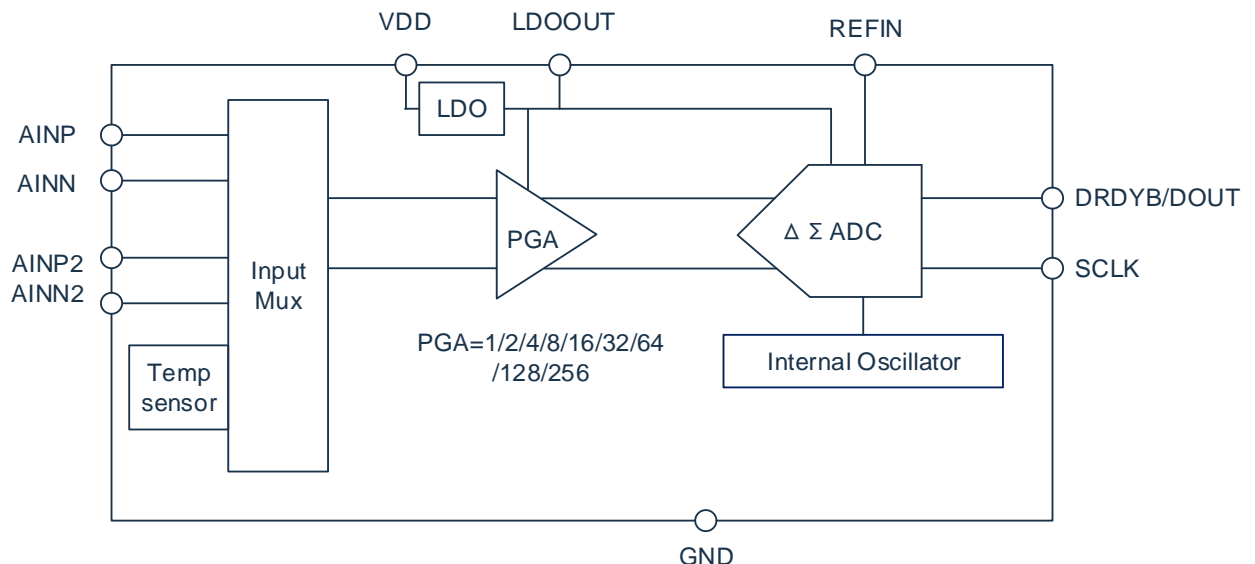
The ADC uses a Sigma-Delta modulator to achieve PGA amplification by amplifier structure for a low noise instrument with selectable gain options of 1, 2, 4, 8, 16, 32, 64, 128, 256. PGA = 128, ODR = 10Hz, SET\_LDO = 00, the effective resolution is 20.6-bit.

The Sigma-Delta ADC has a built-in oscillator and no need for external crystal oscillator.

The Sigma-Delta ADC can be configured in a variety of functional modes via DRDYB /DOUT and SCLK, such as for temperature detection, PGA gain selection, ADC ODR settings, etc.

The Sigma-Delta ADC supports sleep mode.

The block diagram of the Sigma-Delta ADC operation is as follows:



## 12.3 Description of ADC working principle

### 12.3.1 LDO

The built-in LDO of Sigma-Delta ADC can supply power to the on-chip analog module, at the same time, it can provide 20 mA current to the off-chip circuit. LDOOUT pin shall be connected externally with a capacitor at least 1uF. Different LDO output voltages can be configured by setting SEL\_LDO [1:0], such as 2.4V, 2.6V, 3V and 3.3V. LDO can also be configured to work in switch mode or linear voltage regulator mode by setting BYPASSLDO. When the chip enters sleep mode, LDO output drops to 0 V.

### 12.3.2 Analog inputs

Sigma-Delta ADC has one ADC integrated with one differential input channel. The analog input sources include differential inputs (AINP/AINN), and internal temperature sensor outputs. The switching of the input signal is controlled by registers CH\_SEL[3:0], which is located in register SDADCCON1, and CH\_SEL[2:0], which is located in register SDADCCON2.

### 12.3.3 Temperature sensor

Temperature measurement is provided internally in the module. It is recommended to use the configuration with PGA gain = 8 and ADC ODR = 640Hz. One-point temperature calibration is recommended to improve accuracy. Here is the calibration method: At a certain temperature point A, the temperature sensor is used to measure and get the temperature code Ya. Then other real temperature B can be calculated by  $B = Yb * (273.15 + A) / Ya - 273.15$ . The unit of temperature A is Celsius. Ya is the temperature code corresponding to temperature A. Yb is the temperature code corresponding to temperature B.

For applications where the reference voltage may change (such as ratio metric measurement, the absolute value of the reference voltage is not important), you can choose to indirectly measure the real-time reference voltage by measuring BG, and then calculate the real-time temperature.

### 12.3.4 Low noise programmable gain amplifier

Sigma-Delta ADC provides a low-noise, low-drift PGA amplifier with chopper technique, which is connected with the differential output of bridge sensor. PGA\_SEL[3:0] is used to configure different gain such as 2, 4, 8, 16, 32, 64, 128 and 256. When PGA=2, 4, 8, the first stage amplifier of the PGA is disabled for saving power. When using a low noise PGA amplifier, the input range is between GND+0.75V and VDD-1V. Exceeding this range will result in a decrease in actual performance. When the analog input skips the PGA and is input directly to the ADC, the gain is 1.

### 12.3.5 ADC clock, output data rate

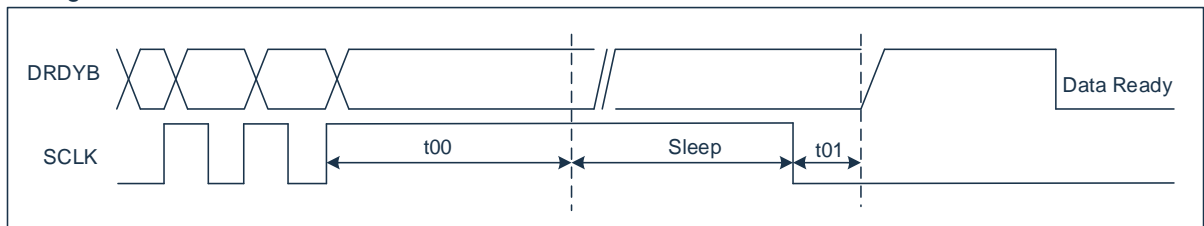
Sigma-Delta ADC has a built-in clock to provide the frequency required by the system. 328KHz or 656KHz can be selected by FADC. The ODR of ADC can be configured by FADC and OSR [2:0]. The clock, ODR, and chopping frequency are shown in the table below.

FADC	OSR[2:0]	ODR	ADC Clock
0	111	2.5Hz	328KHz
0	110	5Hz	328KHz
0	101	10Hz	328KHz
0	100	20Hz	328KHz
0	011	80Hz	328KHz
0	010	320Hz	328KHz
0	001	640Hz	328KHz
0	000	1.28KHz	328KHz
1	111	5Hz	656KHz
1	110	10Hz	656KHz
1	101	20Hz	656KHz
1	100	40Hz	656KHz
1	011	160Hz	656KHz
1	010	640Hz	656KHz
1	001	1.28KHz	656KHz
1	000	2.56KHz	656KHz

### 12.3.6 Reset and sleep mode

When the mode is powered on, the built-in power-on reset circuit will generate a reset signal to reset the mode automatically. When SCLK toggles from low to high level and keeps high state for more than 100us, the Sigma-Delta ADC enters sleep mode, and the power consumption is less than 50nA. When SCLK returns to low level, the mode returns to normal mode. When the system re-enters normal operation from sleep mode, all functions are configured to the state before sleep and no function configuration is required.

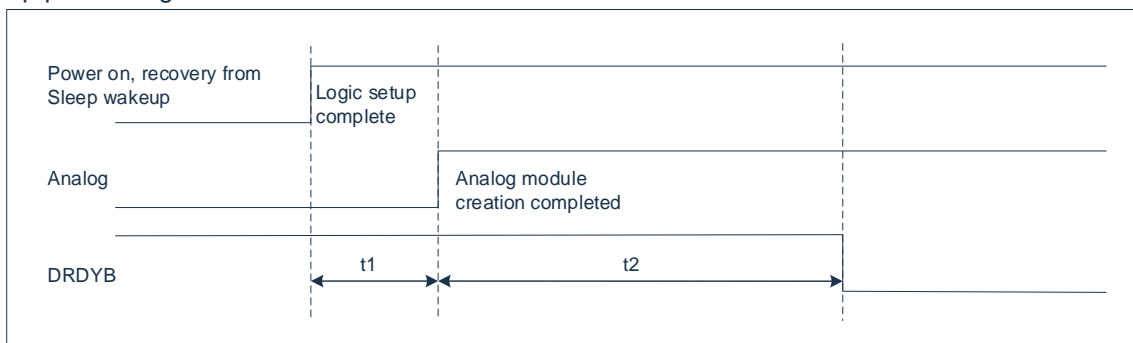
The schematic diagram of the sleep mode is shown in the figure below. Where t00 represents hold-on time of SCLK high level, which needs to be larger than 100us; t01 represents hold-on time of SCLK low level, which needs to be larger than 10us.



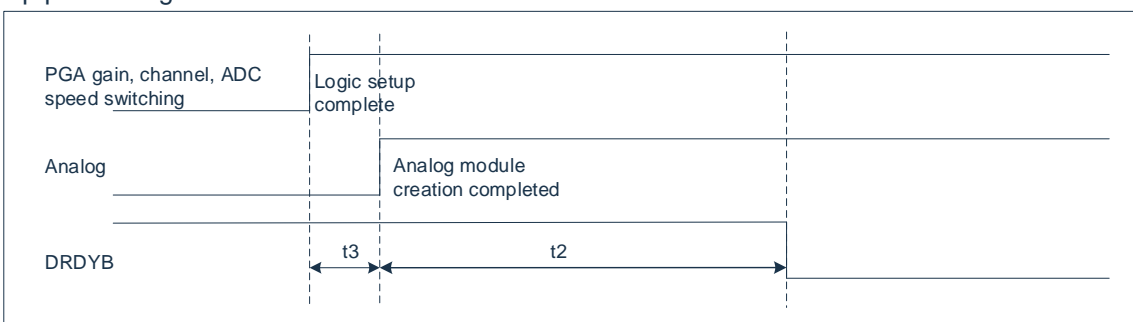
### 12.3.7 Setup time

The setup time of Sigma-Delta ADC is 3 conversion cycles.

Data setup processing 1:



Data setup processing 2:



Setup time:

Parameter	Description	Min	Typ	Max	Unit
Setup time					
t1	Power-on, recovery time from sleep mode	-	0.4	-	ms
t2	Data setup time	-	3	-	Conversion cycle
t3	Recovery time after PGA, channel, ADC speed switching	-	0.8	-	us

## 12.4 SPI Serial Interface

The Sigma-Delta ADC uses 2-wire SPI serial interface SCLK, DRDYB/DOUT for data reception and function configuration.

### 12.4.1 Digital output codes

The Sigma-Delta ADC outputs data as a digital output code of 24-bit binary complement, where B23 is the sign bit, 0 is positive and 1 is negative. The highest bit (MSB) is output first. The following table shows the ideal output codes corresponding to different analog input signals.

Analog input voltage	Digital output code																								Decimal Code	Hex Code
	B23	B22	B21	B20	B19	B18	B17	B16	B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0		B[23:0]
Vref-1LSB	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	8388607	7FFFFFFF
2LSB	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	000002
1LSB	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	000001
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	0	000000
-1LSB	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1	FFFFFF
-2LSB	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	-2	FFFFFE
-Vref	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-8388608	800000

### 12.4.2 Data ready/data input and output (DRDYB/DOUT)

The DRDYB/DOUT pin serves four purposes. Firstly, it indicates when a conversion is completed the output is low. Secondly, on the first rising edge of SCLK after data ready, the DRDYB/DOUT outputs the sign bit of the converted data. At the rising edge of each SCLK, the data will be automatically shifted by 1 bit. After 24 SCLKs all 24 bits of data will be read out, if the sending of SCLKs is paused at this time, the DRDYB/DOUT will hold the last bit of data until the next data is ready to be pulled high, thereafter when the DRDYB/DOUT is pulled low again, it means that the new data has been converted and the next data can be read. Thirdly, Output register status update flag at 25th and 26th SCLK. Fourthly, as a register data to write or read pin, when a register needs to be configured or a register value needs to be read, the SPI needs to send 46 SCLKs to determine whether it is a write register operation or a read register operation based on the instructions input by the DRDYB/DOUT.

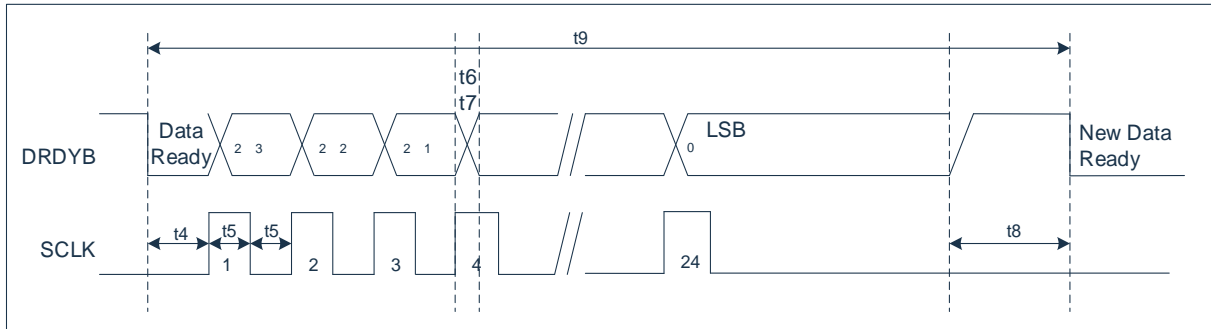
### 12.4.3 Serial input clock (SCLK)

The serial clock input SCLK is a digital pin. This signal should be guaranteed to be a clean signal, burrs or slow rising edges can cause incorrect data to be read or false states to be entered. Therefore, it should be ensured that both the rising and falling times of SCLK are less than 50ns.

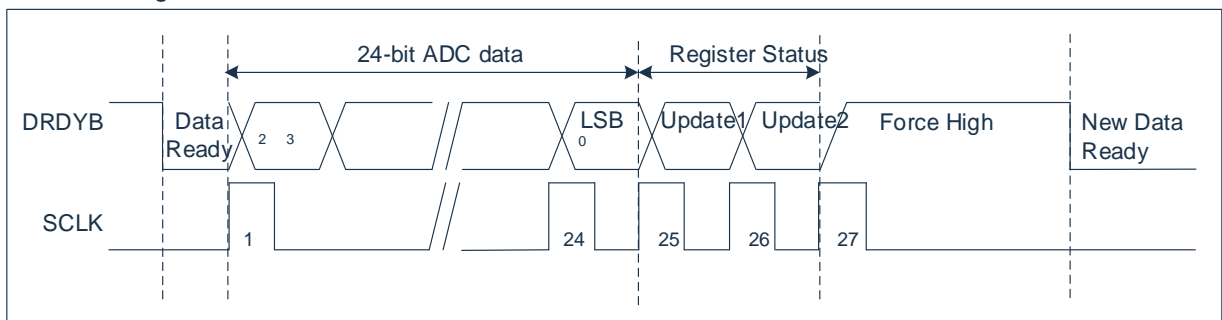


## 12.4.4 Serial data transmission

Read data timing chart 1:



Read data timing chart 2:



The Sigma-Delta ADC can continuously convert the analog input signal. When pulling the DRDYB/DOUT low, it indicates that the data is ready to be accepted, and the first SCLK of the input comes to read out the highest bit of the output, and after 24 SCLKs all 24 bits of data will be read out. If the SCLK sending is paused at this point, the DRDYB/DOUT will hold the last bit of data until it is pulled up, as in timing chart 1.

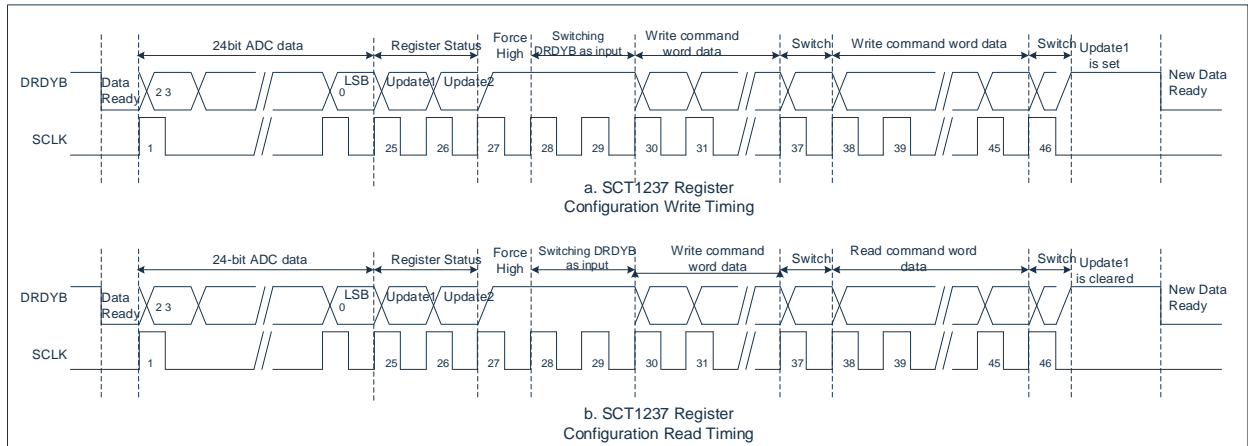
If the SCLK is continuously sent, the 25th and 26th SCLK output whether the configuration register has a write operation flag, the 25th SCLK corresponding to the DRDYB/DOUT is 1 when it indicates that the register configuration is written with a new value, and the DRDYB/DOUT, which is corresponding to the 26th SCLK, is the module extension reserved bit, and the output is always 0 at present. By the 27th SCLK can pull the DRDYB/DOUT high, after that when the DRDYB/DOUT is pulled low again, it means the new data is ready to accept for the next data conversion. The basic timing is shown in timing chart 2.

Read data timing chart 1:

Parameter	Description	Min	Typ	Max	Unit
t4	After DRDYB/DOUT goes low to the first SCLK rising edge	-	2	-	ns
t5	SCLK high or low pulse width	455	-	-	ns
t6	SCLK rising edge to new data valid (transmission delay)	455	-	-	ns
t7	SCLK rising edge to old data bit valid (hold time)	-	-	455	ns
t8	Data update, reading previous data is disabled	-	26	-	us
t9	Conversion time, 10Hz	-	100	-	ms
	Conversion time, 40Hz	-	25	-	ms
	Conversion time, 640Hz	-	1.5625	-	ms

## 12.4.5 Function configuration

The Sigma-Delta ADC can read and configure the registers through SCLK and DRDYB/DOUT, and the functional configuration timings are shown below:



A brief description of the function configuration process, which is determined after the DRDYB/DOUT has changed from high to low:

- 1) The 1st to 24th SCLK to read ADC data. If you do not need to configure registers or read registers, you can omit the following steps.
- 2) The 25th to 26th SCLK, read register writing operation status.
- 3) The 27th SCLK, the module pulls the DRDYB/DOUT output high.
- 4) The 28th to 29th SCLK, toggle DRDYB/DOUT as input.
- 5) The 30th to 36th SCLK, input register and write or read instruction data (high bit at first).
- 6) The 37th SCLK, toggles the direction of DRDYB/DOUT ( if it is write register, the DRDYB/DOUT is input; if it is read register, the DRDYB/DOUT is output).
- 7) The 38th to 45th SCLK, input register configuration data or output register configuration data (high bit to input/output at first).
- 8) The 46th SCLK toggles the DRDYB/DOUT to output and pulls DRDYB/DOUT high. update1/ update2 is set or cleared to zero.

## 12.4.6 Description of SPI opcode command

Sigma-Delta ADC has 8 opcode command word with 7-bit length, which are described as follows:

Name	Opcode command (controlled)	Function	Remark
SDADCCON1	0x65	Write SDADCCON1	SPI communication required for register configuration
	0x56	Read SDADCCON1	SPI communication required for register configuration
SDADCCON2	0x69	Write SDADCCON2	SPI communication required for register configuration
	0x5A	Read SDADCCON2	SPI communication required for register configuration
SDADCCON3	0x6D	Write SDADCCON3	SPI communication required for register configuration
	0x5E	Read SDADCCON3	SPI communication required for register configuration
SDADCCON4	0x61	Write SDADCCON4	SPI communication required for register configuration
	0x52	Read SDADCCON4	SPI communication required for register configuration

## 12.4.7 Cautions for SPI Communication

As described above, Sigma-Delta ADCs can have three types of communication timings, which are:

Timing 1: Read 24-bit ADC conversion data

Timing 2: Read 24-bit ADC conversion data + register status

Timing 3: Read 24-bit ADC conversion data + register status + read/write control word

Notice:

- 1) The read data is the result of the last ADC conversion completed before the transmit timing.
- 2) Need to determine that the DRDYB has generated a falling edge before sending the clock;
- 3) When sending any timing, the time from the falling edge of the DRDYB to the completion of all data sending is less than one conversion time (that is, the timing is sent between two falling edges), otherwise it will cause DRDYB abnormal and need to enter the sleep mode before recovery.

## 12.5 Related registers

### 12.5.1 Sigma-Delta ADC control register 1

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SDADCCON1	SET_LDO_1	SET_LDO_0	OSR_2	OSR_1	--	PGA_SEL2	PGA_SEL1	PGA_SEL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset value	0	0	1	1	0	1	1	0

Bit7~Bit6	SET_LDO<1:0>:	LDO output voltage control
	00=	3V;
	01=	2.4V;
	10=	2.6V;
	11=	3.3V.
Bit5~Bit4	OSR<2:1>:	OSR settings (conversion rate related)
	000=	64;
	001=	128;
	010=	256;
	011=	1024;
	100=	4096;
	101=	8192;
	110=	16384;
	111=	32768.
Bit3	CHSEL<3>:	Channel N selects high bits, with N = 1 or 2;
	1=	Channel 2;
	0=	Channel 1.
Bit2~Bit0	PGA_SEL<2:0>:	PGA gain;
	000=	2;
	001=	4;
	010=	8;
	011=	16;
	100=	32;
	101=	64;
	110=	128;
	111=	256.

## 12.5.2 Sigma-Delta ADC control register 2

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SDADCCON2	CHSEL2	CHSEL1	CHSEL0	LPWR	FADC	OSR_0	ENCHOPB	FCHOP_ADC
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset value	0	0	0	1	0	0	0	0

- Bit7~Bit5 CHSEL<2:0>: Channel N select low bits, N determined by CHSEL <3> Channel N selects high bits;  
000= Channel N;  
001= Channel N positive and negative switching (system chopper);  
010= Channel N temperature;  
011= Internal shortage of Channel N;  
100= Channel N connects directly to ADC and turns off PGA  
101= Channel N connects directly to ADC and turns off PGA  
110= BG of Channel N;  
111= Channel N is internal shortage and connects directly to ADC.
- Bit4 LPWR: Power consumption selection;  
0= Lowest power consumption;  
1= Normal power consumption.
- Bit3 FADC: Sigma-Delta ADC system clock;  
0= 328KHz;  
1= 656KHz (recommended).
- Bit2 OSR\_0: OSR LSB.
- Bit1 ENCHOPB: ADC chopper enable;  
0= Enable chopper function (recommended);  
1= Disable chopper function.
- Bit0 FCHOP\_ADC: ADC chopper frequency division control;  
0= Divided by 16;  
1= Divided by 32.

### 12.5.3 Sigma-Delta ADC control register 3

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SDADCCON3	--	--	--	--	--	BYPASSLDO	OCP_DIS_LDO	--
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset value	1	0	0	0	1	0	0	0

- Bit7~Bit4                   --: Reserved, and it must be 1000.
- Bit3                       --: Reserved, and it must be 1.
- Bit2           BYPASSLDO: LDO bypass control;  
                   0= LDO output;  
                   1= Bypass LDO and output VDD.
- Bit1           OCP\_DIS\_LDO: LDO output overcurrent protection enable control;  
                   0= Enable;  
                   1= Disable.
- Bit0                       --: Reserved, and it must be 0.

## 12.5.4 Sigma-Delta ADC control register 4

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SDADCCON4	--	--	--	--	--	--	FCHOP_1	FCHOP_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset value	1	0	0	0	0	0	0	1

Bit7~Bit4                      --: Reserved, and it must be 1000.  
 Bit3~Bit2                     --: Reserved, and it must be 00.  
 Bit1~Bit0            FCHOP<1:0>: PGA chopper frequency division control;  
                                  00= Divided by 4;  
                                  01= Divided by 8;  
                                  10= Divided by 16;  
                                  11= Divided by 32.

## Chapter 13 D/A Converter

### 13.1 Function of D/A converter

The D/A converter converts digital inputs to analog signals with 8-bit resolution and can control analog outputs.

The D/A converter has the following functions:

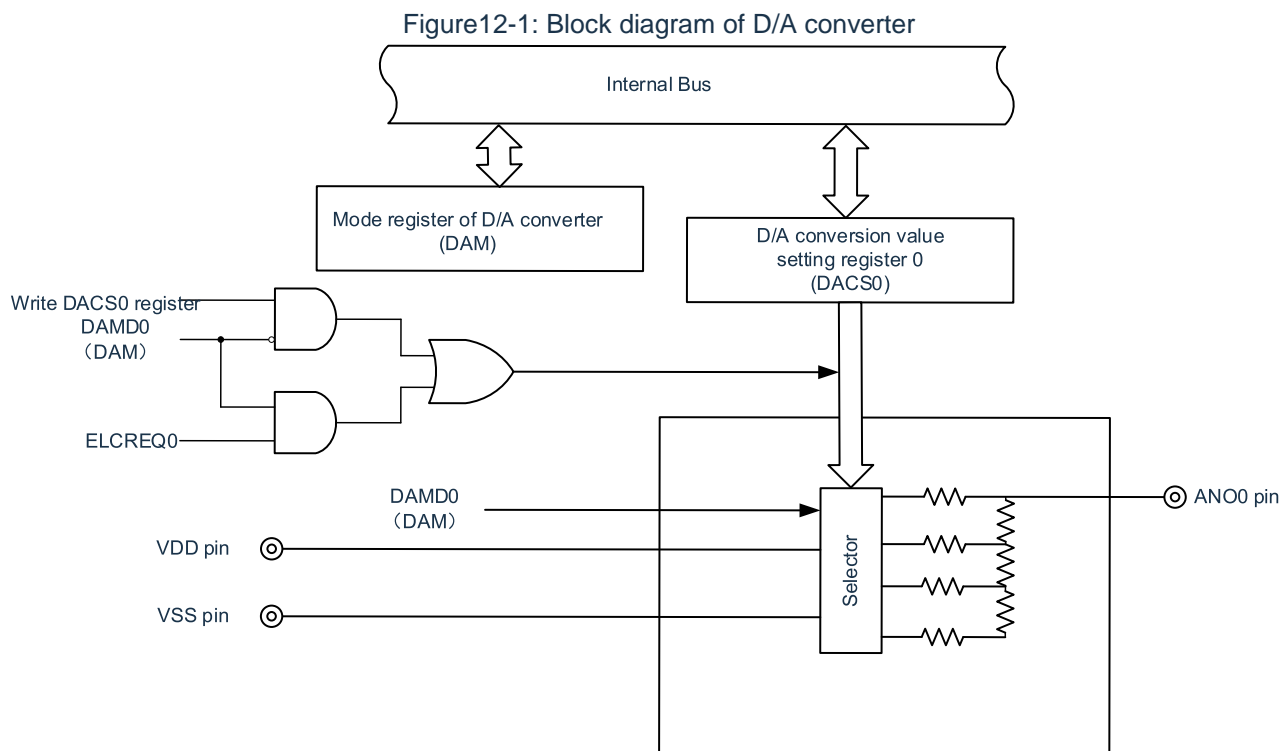
- 8-bit resolution
- R-2R ladder type
- Analog output voltage
  - 8-bit resolution:  $V_{DD} * m8 / 256$  (m8: Value set to DACS0 register)
- Operation mode
  - Normal mode
  - Real-time output mode

Remark: i=0



## 13.2 Structure of D/A converter

Figure12-1 shows the block diagram of the D/A converter.



Remark: ELCREQ0 is a trigger signal is used in the real-time output mode (EVENTC's event signal).

## 13.3 Registers for controlling D/A converter

The D/A converter is controlled by the following registers.

- Peripheral enable register 1 (PER1)
- D/A converter mode register (DAM)
- D/A conversion value setting register 0 (DACS0)
- Event output destination select register (ELSELRn), n= 00~15

### 13.3.1 Peripheral enable register 1 (PER1)

The PER1 register is a register that sets to enable or disable providing clocks to each peripheral hardware.

Reduce power consumption and noise by stopping clocks to hardware that is not in use.

To use the D/A converter, bit7 (DACEN) must be set to "1".

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

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

Figure 12-2: Format of peripheral enabled register 1 (PER1)

Address: 4002081AH	7	6	5	4	3	2	1	0
Symbol	OPAEN	CMPEN	DACEN	ADCEN	0	OSDCEN	DMAEN	SPIHSEN
PER1								

DACEN	Control of the input clock for the D/A converter
0	Stop providing input clock. <ul style="list-style-type: none"> <li>• The SFR used by the D/A converter cannot be written.</li> <li>• The D/A converter is in the reset state.</li> </ul>
1	Provides input clock. <ul style="list-style-type: none"> <li>• The SFR used by the D/A converter can be read and written.</li> </ul>

Notice: To set the D/A Converter, the DACEN bit must be set to "1" at first.

When the DACEN bit is "0", the write operation of the control register of the D/A Converter is ignored, and the read value is the initial value.

### 13.3.2 D/A converter mode register (DAM)

This register controls the operation of the D/A converter.

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

Figure12- 3: Format of D/A converter mode register (DAM)

Address: 40045402H	After reset: 00H		R/W					
Symbol	7	6	5	4	3	2	1	0
DAM	0	0	0	DACE0	0	0	0	DAMD0

DACEi	D/A conversion operation control
0	Stops D/A conversion operation.
1	Enables D/A conversion operation.

DAMDi	D/A converter operation mode selection
0	Normal mode
1	Real-time output mode

Remark: i=0

### 13.3.3 D/A conversion value setting register i (DACSi) (i=0)

This is the register that sets the analog voltage value output to the ANO0 pin and ANO1 pin when the D/A converter is used. The DACSi register is set by an 8-bit memory manipulation instruction.

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

Figure12- 4: Format of D/A conversion value setting register 0 (DACSi) (i=0)

Address: 40045400H (DACS0)	After reset: 00H		R/W					
Symbol	7	6	5	4	3	2	1	0
DACSi	DACSi7	DACSi6	DACSi5	DACSi4	DACSi3	DACSi2	DACSi1	DACSi0

Remark: The analog output voltage (VANOi) of the D/A converter is as follows:  $VANOi = V_{DD} * (DACSi) / 256$

When the D/A converter is not used, in order to reduce the unnecessary consumption current, it is necessary to set the DACEi bit to "0" (output disabled) and set the DACSi register to "00H" so that no current flows through the R-2R resistor.

### 13.3.4 Event output destination select register n (ELSELRn), n=00~15

When using the real-time output mode of the D/A converter, the event signal of the event link controller is used as the initiation trigger, and the D/A conversion is performed. For details, please refer to "21.3.1 Event output destination select register n(ELSELRn)(n=00~15)".

## 13.4 Operation of D/A Converter

### 13.4.1 Operation in Normal Mode

The D/A conversion is initiated using the write operation of the DACSi register as the initiation trigger.

The setup method is as follows:

- ① Set the DACEN bit of the peripheral enable register 1 (PER1) to 1 to start the supply of the input clock to the D/A converter.
- ② Use the port mode control register (PMC) to set the ports to analog pins.
- ③ Set the DAMDi bit of the DAM register (the mode register of the D/A converter) to "0" (normal mode).
- ④ Sets the analog voltage value of the ANOi pin output to the DACSi register (D/A conversion value setting register i).

The above (1) ~ (4) are the initial settings.

- ⑤ Set the DACEi bit of the DAM register to "1" (enable D/A conversion).

Start the D/A conversion and after the settling time elapses, the analog voltage set in step ④ is output to the ANOi pin.

- ⑥ To perform subsequent D/A conversions, write to the DACSi register.

The previous D/A conversion result is held until the next D/A conversion is performed

When the DACEi bit of the DAM register is set to "0" (D/A conversion operation stop), D/A conversion stops.

Notice:

1. Even if "1", "0", and then "1" is set to the DACEi bit, there is a wait after "1" is set for the last time.  
The analog voltage of the register setting is output to the ANOi pin.
2. If the DACS0 register is rewritten during the settling time, D/A conversion is aborted and reconversion by using the newly written values starts.

Remark: i=0

### 13.4.2 Operation in real-time output mode

Each channel of the D/A conversion is performed on using the signals from the EVENTC as triggers.

The setup method is as follows:

- ① Set the DACEN bit of the peripheral enable register 1 (PER1) to 1 to start the supply of the input clock to the D/A converter.
- ② Use the port mode control register (PMC) to set the ports to analog pins.
- ③ Set the DAMDi bit of the DAM register (the mode register of the D/A converter) to "0" (normal mode).
- ④ Sets the analog voltage value of the ANOi pin output to the DACSi register (D/A conversion value setting register i).
- ⑤ Set the DACEi bit of the DAM register to "1" (enable D/A conversion).
- ⑥ Start the D/A conversion and after the settling time elapses, the analog voltage set in step ③ is output to the ANOi pin.
- ⑦ Use the event output destination select register (ELSELRn) to set the real-time trigger signal.
- ⑧ Set the DAMDi bit of the DAM register to 1 (real-time output mode).
- ⑨ Start the operation of the event generation source.  
The above ①~⑧ are the initial settings.
- ⑩ Thereafter, by generating a trigger signal for the real-time output mode, the D/A conversion begins, and after the settling time has elapsed, the analog voltage set by ④ is output to the ANOi pin.

Before performing the next D/A conversion (which generates a trigger signal for real-time output mode), set the analog voltage value of the ANOi pin output to the DACSi registers.

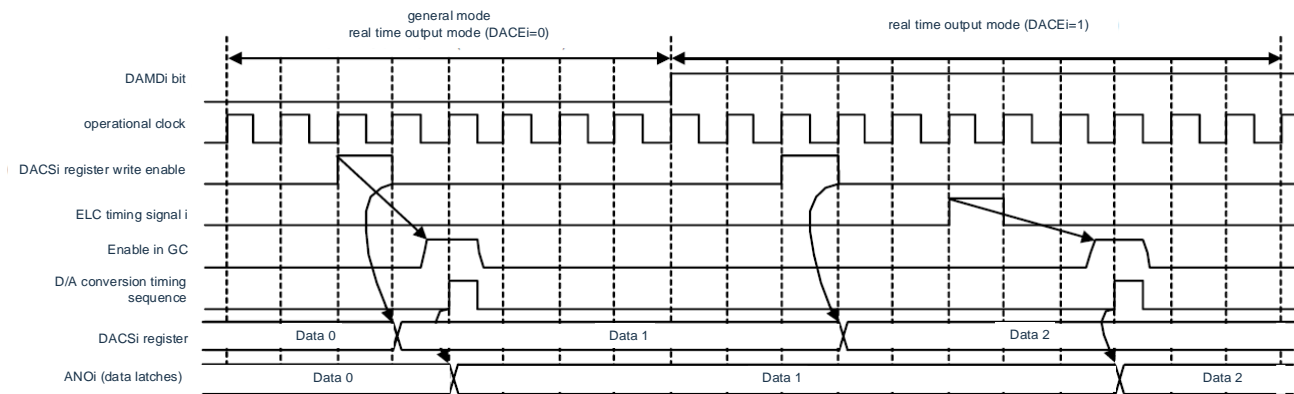
When the DACEi bit of the DAM register is set to "0" (D/A conversion operation stop), D/A conversion stops.

Notice:

1. Even if "1", "0", and then "1" is set to the DACEi bit, there is a wait after "1" is set for the last time. The analog voltage of the register setting is output to the ANOi pin.
2. Make the interval between each generation of the request trigger signal for the same channel real-time output mode longer than the settling time. If a trigger signal for the real-time output mode is generated during the settling time, the D/A conversion is aborted and the conversion restarts.
3. The generation interval of the trigger signal for the same channel real-time output mode must be greater than 3  $F_{CLK}$  clocks. If the start trigger is generated continuously at intervals of less than or equal to 3  $F_{CLK}$  clocks, the D/A conversion is performed only when the first trigger is generated.

### 13.4.3 D/A conversion output timing

Figure12-5shows the output timing of the D/A conversion.



Remark: i=0

- Normal operation mode and real-time output mode (when conversion is disabled).  
Write the data latch (output from the ANO<sub>i</sub> pin) after writing the DACSi register 1 cycle (operating clock).
- Real-time output mode (when conversion is enabled)  
Write the data latch (output from the ANO<sub>i</sub> pin) after 3 cycles after accepting the EVENTC event signal (operating clock).

## 13.5 Cautions for D/A Converter

Observe the following cautions when using the D/A converter.

- (1) The digital port I/O function, which is the alternate function of the ANO0 pin, does not operate if the ports are set to analog pins by using the port mode control register (PMC). When the P2 register is read while the ports are set to analog pins by using the ADPC register, "0" is read in the input mode and the set value of the P2 register is read in the output mode. If the digital output mode is set, no data is output to the pins.
- (2) In sleep mode and deep sleep mode, the D/A converter continues to operate. To reduce power consumption, the DACEi bit must be cleared to "0" and the WFI instructions must be executed after stopping the D/A conversion.

Remark: i=0

- (3) To stop the real-time output mode (including when changing to normal mode), the following procedures must be used:
  - Wait for at least three clocks after stopping the trigger output source and then set bits DACEi and DAMDi to 0.
  - After setting bits DACEi and DAMDi, set the DACEN bit of the PER1 register to 0 (DAC stop).

When the DACEN bit is set to "0", all the registers in the DAC are cleared.

So the settings of the SFRs are required to start the operation again.

- (4) When D/A conversion operation is enabled, do not perform A/D conversions from the analog input pin multiplexed with the ANO0 pin.
- (5) In the real-time output mode, set the value of the DACSi register before generating a trigger signal for real-time output mode.

Do not change the set value of the DACSi register while the trigger signal is output.

- (6) Since the output impedance of the D/A converter is high, no current can be taken out from the ANO0 pin and ANO1 pin. If the input impedance of the load is low, insert a follower amplifier between the load, the ANO0 pin and the ANO1 pin before use. In addition, the wiring length between the follower amplifier and the load must be as short as possible due to the high output impedance.

If the wiring length is long, take measures such as placing a ground pattern around the wiring area.

- (7) When entering deep sleep mode while the real-time output mode is enabled, disable linking of EVENTC events before entering deep sleep mode.

# Chapter 14      Comparator

This product has 2 built-in comparators CMP0 and CMP1.

## 14.1      Function of comparator

The comparators have the following functions:

- The positive terminal input of CMPn can be selected from one of six inputs, and the external port can be selected. CMP0 can select the OPA output.
- The negative terminal input of the CMPn is selected from one of four and can be selected from the external port, the internal reference voltage (1.45V), or the voltage at the 8-bit DAC output.
- The comparison results of Comparator 0 and Comparator 1 can be output by pin (VCOUT0, VCOUT1).

Table14-1: Comparator function summary

Unit	Function
COMP0/COMP1	2 channel comparators (CMP0 and CMP1)
	Selectable reference voltage for the negative terminal of the comparator: Both CMP0 and CMP1 can select external pin inputs (2), internal reference voltage (1.45V) or 8bit DAC output for the negative terminal
	Positive terminal of CMP0 can select external pin input (6) or OPA output
	Positive terminal of CMP1 can select external pin input (6)
	When the positive terminal input voltage > the negative terminal input voltage, the output is high
	When the positive terminal input voltage < the negative terminal input voltage, the output is low
	The filter width of the digital filter is selectable
	Output inversion function
	The comparison results can be output from pins (VCOUT0, VCOUT1).
	It detects the active edge of the comparator output and generates an interrupt signal
	Combine with Timer8 to output TIMER WINDOW
	Supports comparator positive hysteresis, negative hysteresis, and bilateral hysteresis with 20mV, 40mV, and 60mV delay voltages available



## 14.2 Structure of comparator

Figure14-1 shows the block diagram of comparator 0. Figure14-1 shows the block diagram of comparator 1.

Figure14-1: Block diagram of comparator 0

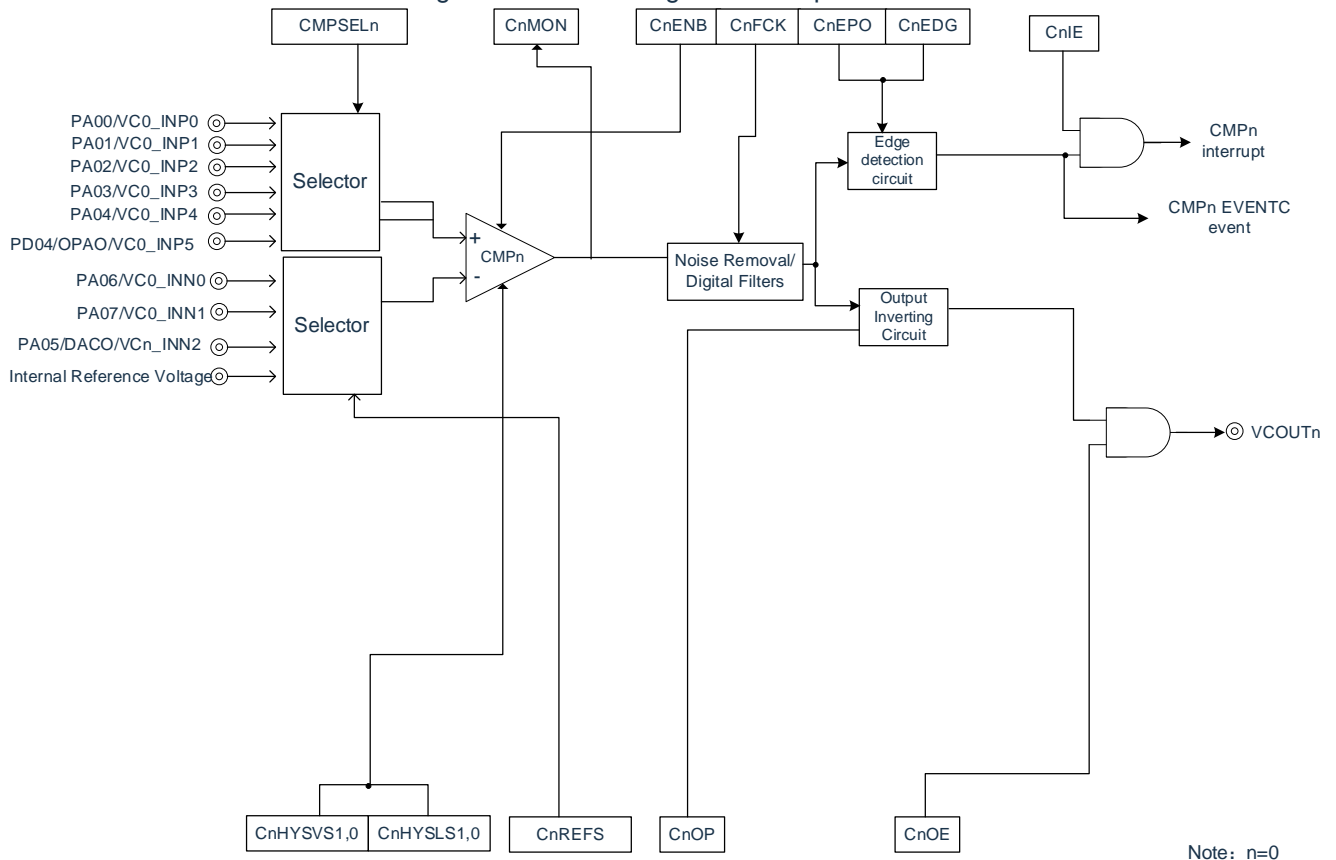
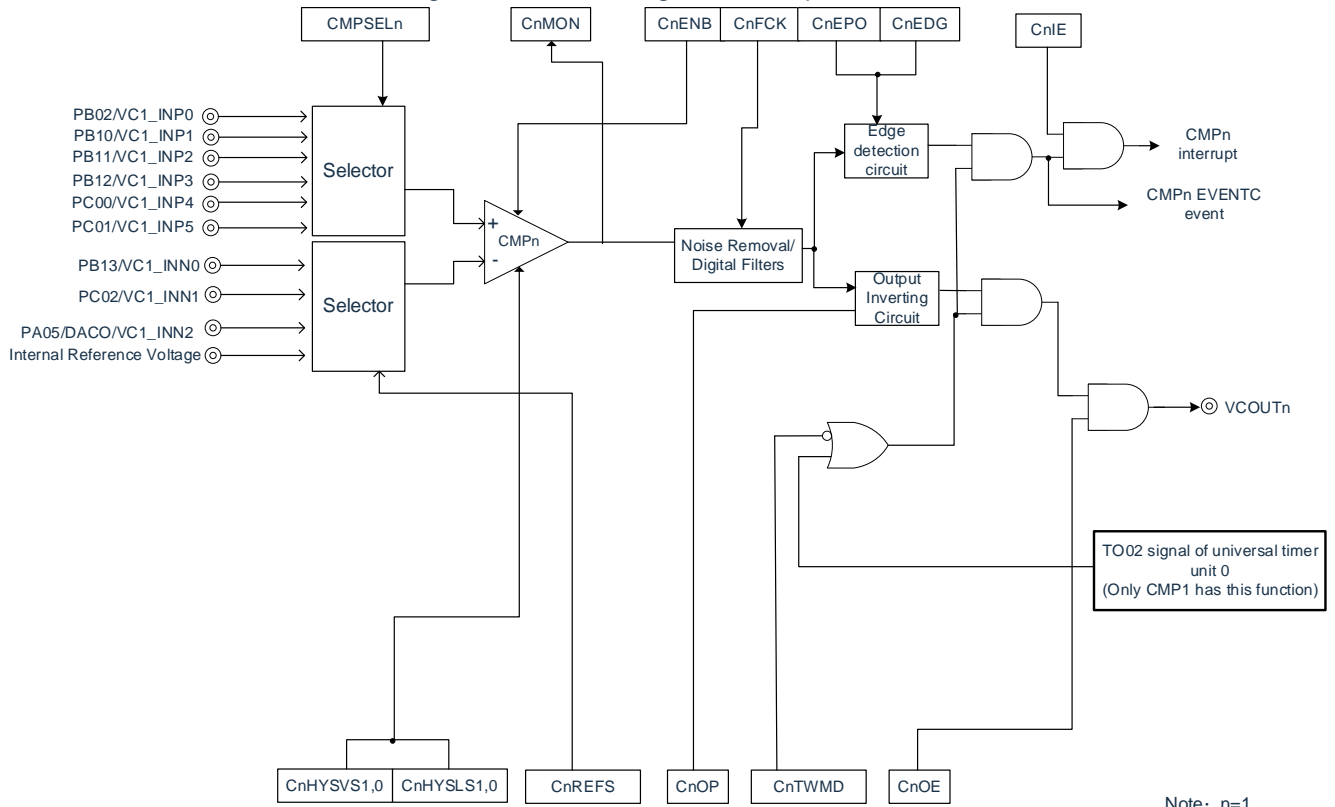


Figure14-2: Block diagram of comparator 1



## 14.3 Registers for controlling comparator

The registers controlling comparator are shown in Table14-2. The register addresses of the comparator are shown in Table14-3: List of registers for comparators.

Table14-2: Registers for controlling comparator

Register Name	Symbol
Peripheral enable register 1	PER1
Comparator mode setting register	COMPMDR
Comparator filter control register	COMPFIR
Comparator output control register	COMPOCR
Comparator 0 negative reference selection register	C0REFS
Comparator 1 negative reference selection register	C1REFS
Comparator 0 positive input selection register	CMPSEL0
Comparator 1 positive input selection register	CMPSEL1
Comparator 0 hysteresis control register	CMP0HY
Comparator 1 hysteresis control register	CMP1HY
Port mode control register	PMCxx
Port mode register	PMxx

Table14-3: List of registers for comparators

Base address	Offset address	Register name	R/W	Bit width	Reset value
0x40045800	0x000	COMPMDR	R/W	8	00H
	0x001	COMPFIR	R/W	8	00H
	0x002	COMPOCR	R/W	8	00H
	0x008	C0REFS	R/W	8	00H
	0x009	C1REFS	R/W	8	00H
	0x00A	CMPSEL0	R/W	8	00H
	0x00B	CMPSEL1	R/W	8	00H
	0x00E	CMP0HY	R/W	8	00H
	0x00F	CMP1HY	R/W	8	00H

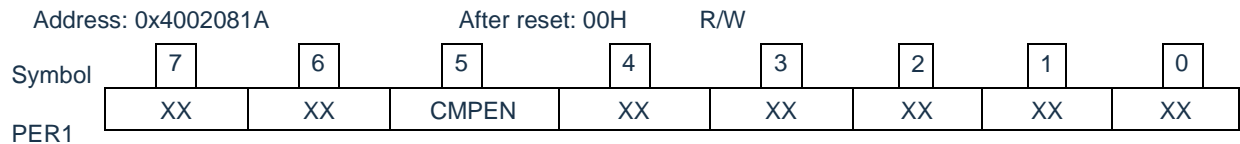
### 14.3.1 Peripheral enable register 1 (PER1)

The PER1 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 comparator, bit 6 (CMPEN) must be set to "1".

The PER1 register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

Figure14-3: Format of peripheral enabled register 1 (PER1)



CMPEN	Control of comparator input clock
0	Stop providing input clock. <ul style="list-style-type: none"> <li>The SFR used by the comparator cannot be written.</li> <li>The comparator is in the reset state.</li> </ul>
1	Provides input clock. <ul style="list-style-type: none"> <li>The SFR used by the comparator can be read and written.</li> </ul>

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

When the CMPEN bit is "0", writing to a control register of the comparator is ignored, and all read values are initial values (except for Port Mode Control Register and Port Mode Register (PMCxx, PMxx)).

## 14.3.2 Comparator mode setting register (COMPMDR)

The COMPMDR register is a register that enables/disables the comparator and detects the comparator output. Setting the CnENB bit to "0" is disabled when the comparator output is enabled (set the CnOE bit of the COMPOCR register to "1"). In the following cases, it is disabled to set the CnENB bit to "1" (n=0,1):

- When the CMPn selects the output of DAC as the reference voltage, and the DAC stops.
- When the input of CMP0 selects OPA output, and the OPA action stops.

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

Figure14-4: Format of comparator mode setting register (COMPMDR)

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

C1MON	Monitor flag for comparator 1 <sup>Note1,2</sup>
0	IVCMP1 < reference voltage of comparator 1, or
1	IVCMP1 > reference voltage of

C1ENB	Comparator 1 operation enable
0	Disables the operation of comparator 1.
1	Enables the operation of comparator 1.

C0MON	Monitor flag for comparator 0 <sup>Note1,2</sup>
0	IVCMP0 < reference voltage of comparator 0, or comparator 0 stops operating.
1	IVCMP0 > reference voltage of comparator 0

C0ENB	Comparator 0 operation enable
0	Disables the operation of comparator 0.
1	Enables the operation of comparator 0.

Note 1: It changed to "0" (initial value) immediately after the reset is released. If the C0ENB bit and C1ENB bit are both set to "0" after enabling the comparator to operate, the value will be indefinite.

Note 2: Ignore the write value of this bit.

### 14.3.3 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 manipulation instruction.

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

Figure14-5: Format of comparator filter control register (COMPFIR)

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

C1EDG	Comparator 1 edge detection selection <sup>Note 1</sup>
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 <sup>Note 1</sup>
0	An interrupt request is generated by the rising edge of the comparator 1.
1	An interrupt request is generated by the falling edge of the comparator 1.

C1FCK1	C1FCK0	Comparator 1 filter selection <sup>Note 1</sup>
0	0	Comparator 1 has no filter.
0	1	Comparator 1 has a filter and samples it by 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 it by $F_{CLK}/32$ .

C0EDG	Comparator 0 edge detection selection <sup>Note 2</sup>
0	An interrupt request is generated by single edge detection of the comparator 0.
1	An interrupt request is generated by bilateral edge detection of the comparator 0.

C0EPO	Comparator 0 edge polarity switching <sup>Note 2</sup>
0	An interrupt request is generated by the rising edge of the comparator 0.
1	An interrupt request is generated by the falling edge of the comparator 0.

C0FCK1	C0FCK0	Comparator 0 filter selection <sup>Note 2</sup>
0	0	Comparator 0 has no filter.
0	1	Comparator 0 has a filter and samples it by the $F_{CLK}$ .
1	0	Comparator 0 has a filter and samples it by $F_{CLK}/8$ .
1	1	Comparator 0 has a filter and samples it by $F_{CLK}/32$ .

Note 1: If C1FCK1~C1FCK0 bits, C1EPO bit and C1EDG bit are changed, interrupt requests of comparator 1 and event signals output to EVENTC may occur. These bits must be changed when the linkage controller is not linking the outputs of comparator 1. In addition, the IF of the interrupt request flag register must be cleared to "0" after the change. If bits C1FCK1 to C1FCK0 are changed from "00B" (comparator 1 without filter) to other values (comparator 1 with filter), the interrupt request of comparator 1 or the event signal output to EVENTC must be used after 4-beat sampling and waiting for the filter output to be updated.

Note 1: If C0FCK1~C0FCK0 bits, C0EPO bit and C0EDG bit are changed, interrupt requests of comparator 0 and event signals output to EVENTC may occur. These bits must be changed when the linkage

controller is not linking the outputs of comparator 0. In addition, the IF of the interrupt request flag register must be cleared to "0" after the change. If bits C0FCK1 to C0FCK0 are changed from "00B" (comparator 0 without filter) to other values (comparator 0 with filter), the interrupt request of comparator 0 or the event signal output to EVENTC must be used after 4-beat sampling and waiting for the filter output to be updated.

### 14.3.4 Comparator output control register (COMPOCR)

The COMPOCR register is a control register that sets the polarity of the comparator output, enables/disables the output and the interrupt output.

In the following cases, the CnOE bit of the COMPOCR register "1" is disabled (output enable). (n= 0, 1)

- When the comparator stops (the CnENB bit of the COMPMDR register is "0")
- When the CMPn selects the output of DAC as the reference voltage, and the DAC stops.
- When the input of CMP0 selects OPA output, and the OPA action stops.

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

Figure14-6Format of comparator output control register (COMPOCR)

Symbol	7	6	5	4	3	2	1	0
COMPOCR	C1OTWMD	C1OP	C1OE	C1IE	0	C0OP	C0OE	C0IE

C1OTWMD	TIMER WINDOW output mode control bit of comparator 1 <sup>Note 1</sup>
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 output polarity of VCOUT1
0	VCOUT1 is the output of comparator 1.
1	VCOUT1 is the inverted output of comparator 1.

C1OE	VCOUT1 pin output enable <sup>Note 2</sup>
0	Disable VCOUT1 output to the pin.
1	Enable VCOUT1 output to the pin.

C1IE	Comparator 1 interrupt request enable <sup>Note 3</sup>
0	Disable comparator 1 interrupt requests.
1	Enable comparator 1 interrupt requests.

C0OP	Selection of output polarity of VCOUT0
0	VCOUT0 is the output of comparator 0.
1	VCOUT0 is the inverted output of comparator 0.

C0OE	VCOUT0 pin output enable <sup>Note 4</sup>
0	Disable VCOUT0 output to the pin.
1	Enable VCOUT0 output to the pin <sup>Note 4, 8</sup> .

C0IE	Comparator 0 interrupt request enable <sup>Note 5</sup>
0	Disable comparator 0 interrupt requests.
1	Enable comparator 0 interrupt requests.



Note 1: When comparator 1 uses 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, and the C1OE bit must be set to "1" after the C1OTWMD bit is set.

Note 2: When the result of comparator 1 is output to the pin, the pins of Pxx, PMxx, PMCxx must be set to 0.

Note 3: If the C1IE is changed from "0" (interrupt request disable) to "1" (interrupt request enable), the IF of the interrupt request flag register may become "1" (interrupt request enable). Therefore, the interrupt must be used after clearing the IF of the interrupt request flag register to "0".

Note 4: When the result of comparator 0 is output to the pin, the pins of Pxx, PMxx, PMCxx must be set to 0.

Note 5: If the C0IE is changed from "0" (interrupt request disable) to "1" (interrupt request enable), the IF of the interrupt request flag register may become "1" (interrupt request enable). Therefore, the interrupt must be used after clearing the IF of the interrupt request flag register to "0".

### 14.3.5 Comparator negative reference selection register (CnREFS)

The CnREFS register is a register to set the negative terminal reference of the comparator. The CnREFS register can be rewritten when the comparator stops operating (CnENB=0). (n= 0, 1)

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

Figure14-7: Format of comparator negative reference selection register (CnREFS)

Symbol	7	6	5	4	3	2	1	0
CnREFS	0	0	0	0	0	0	CnREFS[1:0]	

CnREFS	Comparator negative reference selection
2'b00	External pin VCn_INN0
2'b01	External pin VCn_INN1
2'b10	DAC0 output/ external pin VCn_INN2
2'b11	1.45V reference voltage

Note 1: If the DAC is inactive, the comparator negative reference can also be input from an external pin.

For the configuration of the pin multiplexing function of VCn\_INN0~2 on the negative terminal of the comparator, please refer to "Chapter 2 Port Function".

### 14.3.6 Comparator positive input selection register (CMPSELn)

The CMPSELn register is a positive input selection register for the comparator. The CMPSELn register can be rewritten when the comparator stops operating (CnENB=0). (n= 0, 1)

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

Figure14-8: Format of comparator positive input selection register (CMPSELn)

Symbol	7	6	5	4	3	2	1	0
CMPSELn	0	0	0	0	0	CnVIPSEL		

C0INPSEL	CMP0 positive input selection
3'h0	External pin VC0_INP0
3'h1	External pin VC0_INP1
3'h2	External pin VC0_INP2
3'h3	External pin VC0_INP3
3'h4	External pin VC0_INP4
3'h5	OPA output (OPAO) / external pin VC0_INP5* Note 1
Others:	Settings are disabled.

Note 1: If the OPA is inactive, the comparator VC0\_INP5 can also be input from an external pin.

C1INPSEL	CMP1 positive input selection
3'h0	External pin VC1_INP0
3'h1	External pin VC1_INP1
3'h2	External pin VC1_INP2
3'h3	External pin VC1_INP3
3'h4	External pin VC1_INP4
3'h5	External pin VC1_INP5
Others:	Settings are disabled.

For the configuration of the pin multiplexing function of VCn\_INP0~5 on the positive terminal of the comparator, please refer to "Chapter 2 Port Function".

### 14.3.7 Comparator hysteresis control register (CMPnHY)

The CMPnHY register is a hysteresis function control register of the comparator. The CMPnHY register can be rewritten when the comparator stops operating (CnENB=0). The CMPnHY register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H". (n=0, 1)

Figure14-9: Format of comparator hysteresis control register (CMPnHY)

Symbol	7	6	5	4	3	2	1	0
CMPnHY	0	0	CnHYSVS1	CnHYSVS0	0	0	CnHYSLS1	CnHYSLS0

CnHYSVS1	CnHYSVS0	Hysteresis voltage selection bit of comparator n
0	0	No hysteresis
0	1	20mV
1	0	40mV
1	1	60mV

CnHYSLS1	CnHYSLS0	Hysteresis mode selection bit of comparator n
0	0	No hysteresis
0	1	Positive hysteresis
1	0	Negative hysteresis
1	1	Bilateral hysteresis

## 14.4 Operation

Comparator 0 and comparator 1 can operate independently. CMP0 and OPA can be combined and linked. The setting steps of independent operation and linkage of the comparator are shown in Table14-4.

Table14-4: Setup steps for comparator-related registers

Step	Register	Bit	Set value
1	OPACTL	INPSEL, MODE_OPA, MODE_RPD	OPA mode setting <sup>Note 3</sup>
2	OPADAC	OPA_DAO[4:0], OPA_DAP	OPA built-in 5-bit DAC control <sup>Note 3</sup>
3	OPACTL	OPAEN	1 (OPA operation enable) <sup>Note 3</sup>
4	Settling time for OPA (Min. 1us)		
5	DAM	DAMD	8-bit DAC action mode selection <sup>Note 4</sup>
6	DACS	DACS[7:0]	8-bit DAC voltage range selection <sup>Note 4</sup>
7	DAM	DACE	1 (DAC operation enable) <sup>Note 4</sup>
8	Settling time for DAC (Min. 6us)		
9	CMPSELn	CnVIPSEL/CnVIPSEL	Comparator n positive input selection
10	CnREFS	CnREFS	Comparator n negative input selection
11	Set VC_INP, VC_INN pin (input), OPA (input) <sup>Note 3</sup> to analog input function. PMCxx The VCIN0 pin, VCn and IVREFn pins are functionally selected to set PMCxx bit to "1" (analog input). Set PMxx bit to "1" (input mode).		
12	COMPMDR	CnENB	1 (enable operation)
13	Settling time for comparator (Min. 3us)		
14	COMPFIR	CnFCK	With or without the digital filter, select the sample clock.
		CnEOP, CnEDG	Select the edge detection condition (rising edge, falling edge, or double edges) for the interrupt request.
15	COMPOCR	CnOP, CnOE	Set the output of the VCOUTn (select polarity, enable set or disable output). Please refer to "14.4.4 Output of comparator n (n=0, 1)".
		CnIE	Sets the output that enables or disables interrupt requests. Please refer to "14.4.2 Comparator n interrupt (n=0, 1)".
		C1OTWMD	Set the TIMER WINDOW output enable/disable for comparator 1
16	MKxx <sup>Note1</sup>	MKL	When using interrupts: Select to mask interrupts.
17	IFxx <sup>Note1</sup>	IFL	When using interrupts: 0 (no interrupt request: initialization) <sup>Note 2</sup>

MKxx and IFxx are the interrupt control registers of the comparator, please refer to "Chapter 23 Interrupt Function" for details.

Note 2: After the comparator is set, until the period of stable operation, unwanted interrupt requests may be generated and the interrupt request flag bit must be initialized.

Note 3: When comparator 0 and OPA are linked, set according to the desired mode.

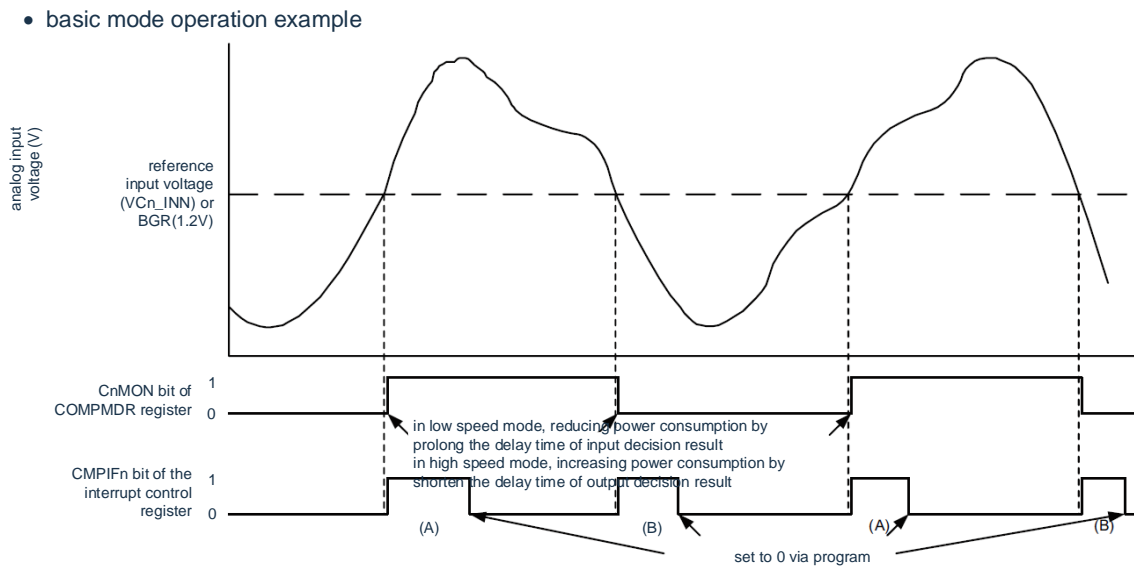
Note 4: Setting is required when using the output of the 8-bit DAC as the reference voltage for the negative terminal of comparator n.

Remark: (n=0, 1)

Figure14-10 shows the operation of comparator n ( $n=0, 1$ ). In the basic mode, when the analog input voltage is higher than the reference input voltage, the CnMON bit in the COMPMDR register is "1". When the analog input voltage is lower than the reference input voltage, the CnMON bit is "0".

To use comparator n interrupt, you must set CnIE bit of COMPOCR register to "1" (enable interrupt request). In this case, if the comparison result changes, an interrupt request for comparator n is generated. For details of the interrupt request, refer to "14.4.2 Comparator n interrupt ( $n=0, 1$ )".

Figure14-10: Example of comparator n operation ( $n=0, 1$ ) (basic mode)



Notice: The above figure shows the case when the CnFCK1 to CnFCK0 bits of COMPFIR register are "00B" (no filter) and the CnEDG bit is "1" (double edges) (CnEDG bit is "0" and CnEPO bit is "0" (rising edge)), and the CnEPO bit is "0" (rising edge), the CMPIFn is limited to the change of (A), the CnEDG bit is "0" and the CnEPO bit is "1" (falling edge) CMPIFn is limited to the change of (B) only).

### 14.4.1 Digital filter of comparator n (n=0, 1)

The comparator n has a built-in digital filter that can select the sampling clock by the CnFCK1 to CnFCK0 bits of the COMPFIR register. The output signal of comparator n is sampled by each sample clock, and the digital filter outputs this sample value at the next sampling clock cycle after if the same level is sampled three times

Figure14-11shows the result of the digital filter for comparator n, Figure14-12shows an example of a digital filter of digital comparator n (n=0, 1) and interrupt operation

Figure14-11: Digital filter of comparator n (n=0, 1) and edge detection structure

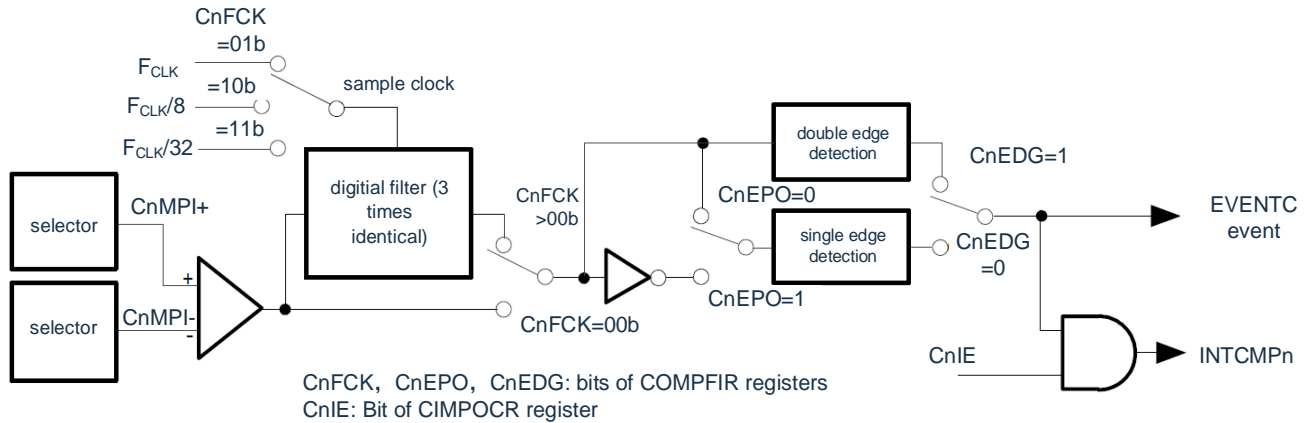
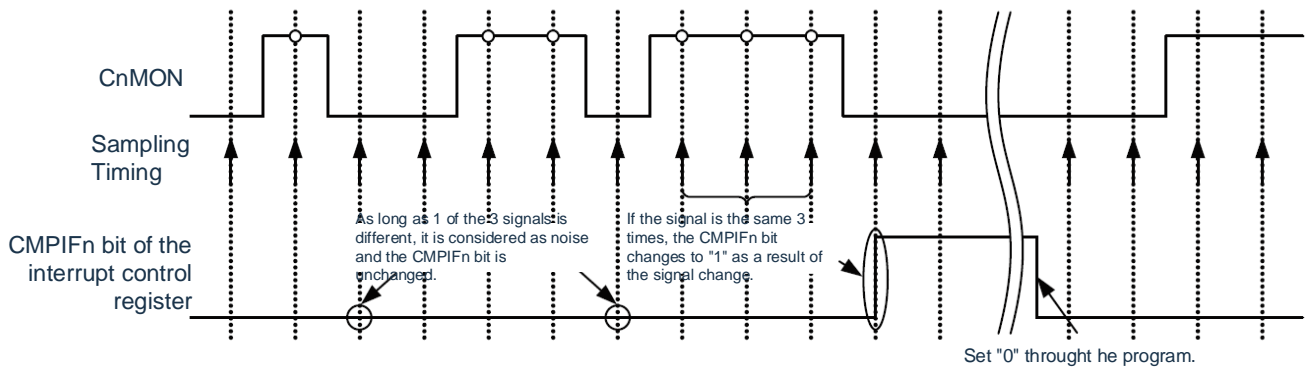


Figure14-12: Example of digital filter and interrupt operation of comparator n (n=0, 1)



Note: The above figure shows the operation when the CnFCK1~CnFCK0 bits of COMPFIR register are "01B", "10B" or "11B" (with digital filter).

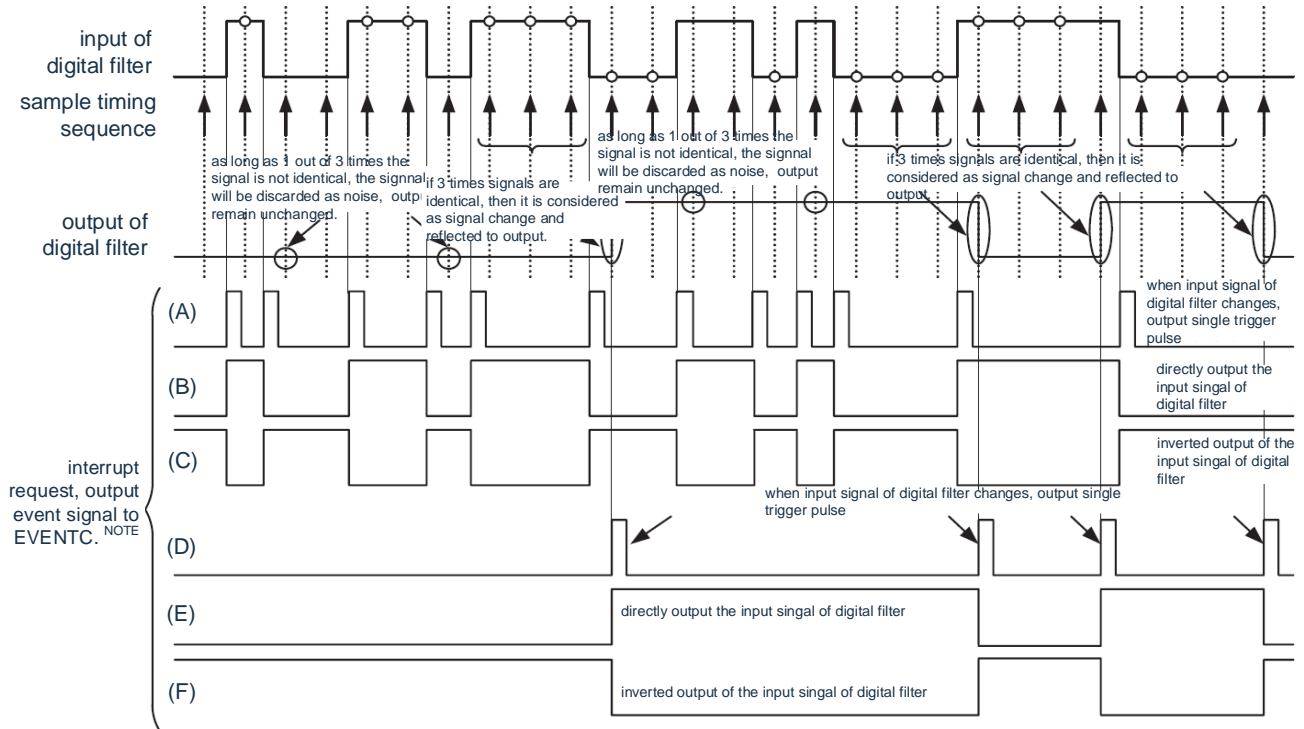
### 14.4.2 Comparator n interrupt (n=0, 1)

The comparator can generate a total of 2 interrupt requests for comparator 0 and comparator 1. To use the comparator n interrupt, it is necessary to set the CnIE bit of the COMPOCR register to "1" (enable interrupt request output). By setting the interrupt request generation conditions via the COMPFIR register, it is also possible to attach a digital filter to the comparator output. Three sample clocks can be selected for the digital filter. For the correspondence of register setting and interrupt request generation, refer to "14.3.3Comparator filter control register (COMPFIR)" and "14.3.4Comparator output control register (COMPOCR)".

### 14.4.3 Event signal output to the coordination controller (EVENTC)

The event signal to EVENTC is generated by detecting the output edge of the digital filter set by the COMPFIR register under the same conditions as the interrupt request. However, unlike the interrupt request, the event signal is always output to EVENTC regardless of the CnIE bit of the COMPOCR register. The selection of the event output target and the stop of the event link must be set by the ELSELR13 register and ELSELR14 register of EVENTC.

Figure14-13: Operation of digital filters, interrupt requests and output event signals to EVENTC



Note: When the CnIE bit (n=0, 1) is "1", the interrupt request and the event signal output to EVENTC are the same waveform. When the CnIE bit (n=0, 1) is "0", only the interrupt request is fixed to "0".

The waveforms in (A), (B), and (C) are the cases where the CnFCK bits (n=0, 1) of the COMPFIR register are "00B" (without digital filter), and the waveforms in (D), (E), and (F) are the cases where the CnFCK bits (n=0, 1) of the COMPFIR register are "01B", "10B" or "11B" (with digital filter). (A) and (D) are the cases where the CnEDG bit is "1" (double edges), (B) and (E) are the cases where the CnEDG bit is "0" and the CnEPO bit is "0" (rising edge), (B) and (E) are the cases where the CnFCK bit is "0" and the CnEPO bit is "0" (rising edge). (C) and (F) are the cases where the CnEDG bit is "0" and the CnEPO bit is "1" (falling edge).



#### 14.4.4 Output of comparator n (n=0, 1)

The CnOE 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 CnOP bit of the COMPOCR register can be used to set the output polarity (positive or negative output). For the correspondence between the register setting and the comparator output, please refer to "14.3.4 Comparator output control register (COMPOCR)".

To output the comparison result of the comparator to the VCOUTn pin, the port must be set as follows (after a reset, the port defaults to the input state), as detailed in "Chapter 2 Port Function":

- ① Set the mode of the comparator (Step 2~5 of "Table14-4: Setup steps for comparator-related registers").
- ② Set the VCOUTn output of the comparator (sets the COMPOCR register, and selects the polarity and enables the output).
- ③ Set the PxxCFG register to select which port to concurrently use the output of VCOUTn.
- ④ Set the output pin of VCOUTn corresponding to the port mode control register PMCxx bit to "0".
- ⑤ Set the output pin of VCOUTn corresponding to the port output latch register Pxx bit to "0".
- ⑥ Set the output pin of VCOUTn corresponding to the port direction register PMxx to output (output from the pin).

### 14.4.5 Stopping or Supplying comparator clock

To stop the comparator by setting peripheral enable register 1 (PER1), use the following procedure:

- ① Set the CnENB bit in the COMPMDR register to "0" (stop the comparator).
- ② Set the IF bit of the interrupt request flag register to "0" (clear any unnecessary interrupts before stopping the comparator).
- ③ Set the CMPEN bit of the PER1 register to "0".

When the clock is stopped by setting PER1, all the internal registers in the comparator are initialized. To use the comparator again, follow the procedure in Table14-4 to set the registers.

Notice:

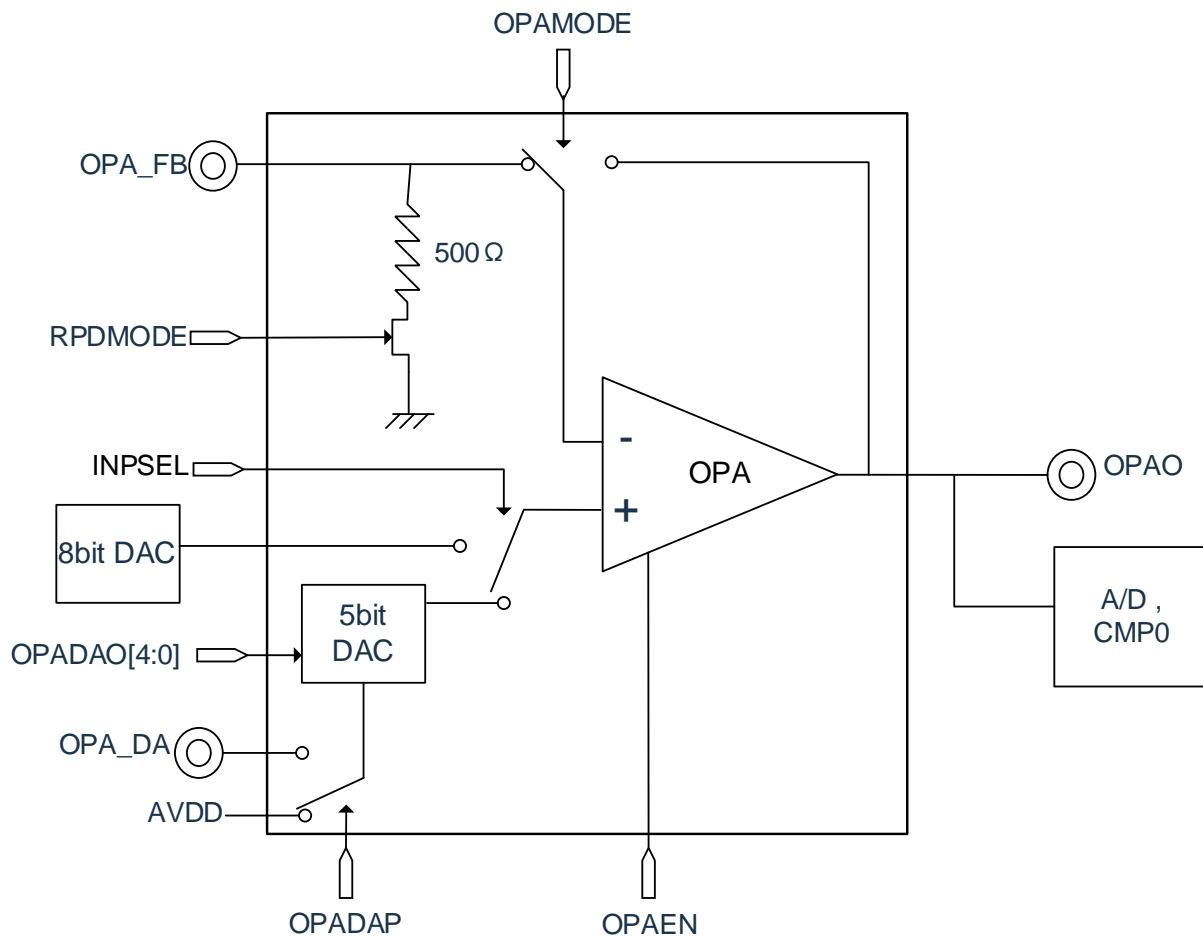
1. When the internal reference voltage (1.45V) is selected for the negative reference voltage the comparator, the A/D converter cannot convert the temperature sensor voltage.
2. DMA transmission can be initiated in any of the following cases and an interrupt generated at the end of the transmission. Before enabling DMA, you must confirm the monitor flag (CnMON) of the comparator as needed, and then enable the DMA to start.
  - Set to generate an interrupt request by single edge detection of the comparator (CnEDG=0) and generate an interrupt request by rising edge of the comparator (CnEPO=0) and  $V_{Cn\_INP} > V_{Cn\_INN}$  (or internal reference voltage is 1.45V).
  - Set to generate an interrupt request by single edge detection of the comparator (CnEDG=0) and generate an interrupt request by falling edge of the comparator (CnEPO=1) and  $V_{Cn\_INP} < V_{Cn\_INN}$  (or internal reference voltage is 1.45V).(n=0, 1)

# Chapter 15 Operational Amplifier (OPA)

## 15.1 Function of operational amplifier

This product has a built-in one channel operational amplifier (OPA) with the following functions:

- Support rail-to-rail
- The OPA works in constant current mode and Buffer mode. The Buffer mode can be used to test OPA offset
- The OPA negative feedback terminal supports the use of internal constant current resistors or the external pin OPA\_FB
- The OPA input can be selected from the 5-bit DAC built into the op-amp module or the 8-bit DAC that comes with the product
- The output of OPA can be used for the analog input of the A/D converter or the positive analog input of comparator 0 (CMP0)



## 15.2 Register of operational amplifier

Table15-1: Registers for controlling operational amplifier

Register name	Symbol
Peripheral enable register 1	PER1
Operational amplifier control register	OPACTL
Operational amplifier digital-to-analog control register	OPADAC
Port mode control register	PMCxx
Port mode register	PMxx

### 15.2.1 Peripheral enable register 1 (PER1)

The PER1 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 operational amplifier, bit 7 (OPAEN) of this PER1 register must be set to "1".

The PER1 register is set by a 1-bit or an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

Figure15-1: Peripheral enable register (PER1)

Address: 0x4002081A	After reset: 00H	R/W						
Symbol	7	6	5	4	3	2	1	0
PER1	OPAEN	XX 0	XX	XX	XX	XX	XX	XX

OPAEN	Control of input clock of operational amplifier
0	Stop providing input clock. The op-amp registers are not writable The op-amp in the reset state
1	Provides input clock. The op-amp registers can be read and write

Notice:

- Before configuring the comparator or op-amp registers, make sure that the OPAEN bit is set to 1 first.
- If OPAEN=0, the write operation to the control registers of the op amp is invalid and all read values are default. (Except for the Port Mode Register (PMXX) and Port Register PXX)

## 15.2.2 Operational amplifier control register (OPACTL)

The OPACTL register is used to control the operational amplifier to start, stop and select the operating mode.

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

Figure15-2: Format of operational amplifier control register (OPACTL)

Address:  
0x40045C00

After reset: 00H R/W

	7	6	5	4	3	2	1	0
OPACTL	OPAEN	0	RPDMODE	OPAMODE	0	INPSEL	0	0

OPAEN	Amplifier OPA enable signal
0	Amplifier stops operating
1	Enable to amplify

RPDMODE	Feedback selection
0	Not support internal constant current resistor
1	Support internal constant current resistor

OPAMODE	OPA operating mode selection
0	Constant current mode
1	Buffer mode

INPSEL	OPA positive input selection
0	Selects the output of the 5bit-DAC
1	Selects the output of the 8bit-DAC

### 15.2.3 Operational amplifier digital-to-analog control register (OPADAC)

The OPADAC register is used to control the op-amp's built-in 5-bit DAC. When the OPA has a built-in 5bit DAC for input selection, this register can be set to control the power selection of the 5bit DAC and the range of the DAC output voltage.

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

Figure15-3: Format of operational amplifier digital-to-analog control register (OPADAC)

Address: 0x40045C04      After reset: 00H    R/W

	7	6	5	4	3	2	1	0
OPADAC	OPADAP	0	0	OPADAO[4:0] <sup>Note1</sup>				

OPADAP	OPA built-in 5bit DAC power select signal
0	Select OPA power supply for 5-bit DAC
1	Select external pin OPA_DA as power supply for 5-bit DAC

Note 1: Analog output voltage of 5bit DAC converter:  $V_{out} = \frac{19+3*n}{560} * V_{ref}$ , and n=0,1,2...31.

## Chapter 16 General-Purpose Serial Communication Unit

This product is equipped with 3 general-purpose serial communication units, each unit has 2 serial channels, each channel can realize 3-wire serial (SSPI), UART and simplified I<sup>2</sup>C communication.

Function assignment of each channel supported by this product is as shown below.

Unit	Channel	Used asSSPI	Used asUART	Used as simplifiedI <sup>2</sup> C
0	0	SSPI00	UART0 (supportingLIN-bus)	IIC00
	1	SSPI01		IIC01
1	0	SSPI10	UART1	IIC10
	1	SSPI11		IIC11
2	0	SSPI20	UART2	IIC20
	1	SSPI21		IIC21

When “UART0” is used for channels 0 and 1 of the unit 0, SSPI00, SSPI01, IIC00 and IIC01 cannot be used.

When “UART1” is used for channels 0 and 1 of the unit 1, SSPI10, SSPI11, IIC10 and IIC11 cannot be used.

When “UART2” is used for channels 0 and 1 of the unit 2, SSPI20, SSPI21, IIC20 and IIC21 cannot be used.

## 16.1 Function of general-purpose serial communication unit

Each serial interface supported by this product has the following features.

### 16.1.1 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)

Synchronizes with the serial clock (SCLK) output from the master device for data transmission and reception.

This is a clock-synchronous communication function that uses a serial clock (SCLK), a transmit serial data (SDO), and a receive serial data (SDI) for communication on a total of three communication lines.

For specific setting examples, please refer to “16.5 Operation of 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) communication”.

[Data transmission and reception]

- 7~16 bit data length
- Phase control of transmit/receive data
- MSB/LSB preferred option
- Level setting for transmit/receive data

[Clock Control]

- Master/slave selection
- Phase control of I/O clock
- Setting of transfer period by prescaler and internal counter
- Maximum transfer rate

During master communication: Max.  $F_{CLK}/2$

During slave communication: Max.  $F_{MCK}/6$

[Interrupt function]

- Transfer end interrupt, buffer empty interrupt

[Error detection flag]

- Overflow error

Remark: Use the clocks within a range satisfying the SCLK cycle time ( $T_{KCY}$ ) characteristics. For details, please refer to the data sheet.



## 16.1.2 UART (UART0~UART2)

This is an asynchronous function using two lines: serial data transmission (TxD) and serial data reception (RxD) lines. By using these two communication lines, data is sent and received asynchronously (using the internal baud rate) with other communicating parties by data frame (consisting of start bits, data, parity bits, and stop bits). Full-duplex UART communication can be performed by using a channel dedicated to transmission (even-numbered channel) and a channel dedicated to reception (odd-numbered channel).

For details about the settings, please refer to “16.7 Operation of UART (UART0~UART2) communication”.

[Data transmission and reception]

- Data length of 7, 8, or 9 bits
- MSB/LSB preferred option
- Level setting of transmit/receive data and select of reverse
- Parity bit appending and parity check functions
- Stop bit appending

[Interrupt function]

- Transfer end interrupt, buffer empty interrupt
- Error interrupt in case of framing error, parity error, or overflow error

[Error detection flag]

- Framing error, parity error, or overflow error

### 16.1.3 Simplified I<sup>2</sup>C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)

This is a function for clock synchronization communication with two or more devices by using two lines: serial clock (SCL) and serial data (SDA). This simplified I<sup>2</sup>C is designed for single communication with a device such as EEPROM, flash memory, or A/D converter, and therefore, it functions only as a master.

Make sure by using software, as well as operating the control registers, that the AC specifications of the start and stop conditions are observed. For details about the settings, please refer to “16.9 Operation of simple I<sup>2</sup>C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication”.

[Data transmission and reception]

- Master transmission, master reception (only master function with a single master)
- ACK output function <sup>Note</sup> and ACK detection function
- Data length of 8 bits (When an address is transmitted, the address is specified by the higher 7 bits, and the lowest bit is used for R/W control.)
- Manual generation of start condition and stop condition

[Interrupt function]

- Transfer end interrupt

[Error detection flag]

- ACK error, or overflow error

※[Functions not supported by simplified I<sup>2</sup>C]

- Slave transmission, slave reception
- Arbitration loss detection function
- Wait detection functions

Note: When receiving the last data, ACK will not be output if 0 is written to the SOEmn bit (serial output enable register m (SOEm)) and serial communication data output is stopped. For details, please refer to “16.9.3(2)Process flow”.

Remark: To use an I<sup>2</sup>C bus of full function, see “Chapter 18 Serial interface IICA”.

## 16.2 Structure of general-purpose serial communication unit

The general-purpose serial communication unit consists of the following hardware.

Figure16-1: Structure of general-purpose serial communication unit

Item	Structure
Shift register	16-bit
Buffer register	Serial data register mn (SDRmn) <sup>Note</sup>
Serial clock I/O	SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, SCLK21 pins (for 3-wire serial I/O) SCL00, SCL01, SCL10, SCL11, SCL20, SCL21 pins (for simple I <sup>2</sup> 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 simple I <sup>2</sup> C)
Slave select input	SS00 pin (for slave select input function)
Control register	<p>&lt; Register of Unit Setting Section &gt;</p> <ul style="list-style-type: none"> <li>• Peripheral enable register 0 (PER0)</li> <li>• Serial clock select register m (SPSm)</li> <li>• Serial channel enable status register m (SEm)</li> <li>• Serial channel start register m (SSm)</li> <li>• Serial channel stop register m (STm)</li> <li>• Serial output enable register m (SOEm)</li> <li>• Serial output register m (SOM)</li> <li>• Serial output level register m (SOLm)</li> <li>• Serial standby control register m (SSCm)</li> <li>• Serial slave select enable register m (SSEm)</li> <li>• Input switching control register (ISC)</li> <li>• Noise filter enable register 0 (NFEN0)</li> </ul>
	<p>&lt;Registers of each channel&gt;</p> <ul style="list-style-type: none"> <li>• Serial data register mn (SDRmn)</li> <li>• Serial mode register mn (SMRmn)</li> <li>• Serial communication run setting register mn (SCRmn)</li> <li>• Serial status register mn (SSRmn)</li> <li>• Serial flag clear trigger register mn (SIRmn)</li> </ul>
	<ul style="list-style-type: none"> <li>• Port multiplexing function configuration register (PxxCFG)</li> <li>• Port output mode register (POMxx)</li> <li>• Port mode register (PMxx)</li> <li>• Port register (Pxx)</li> </ul>

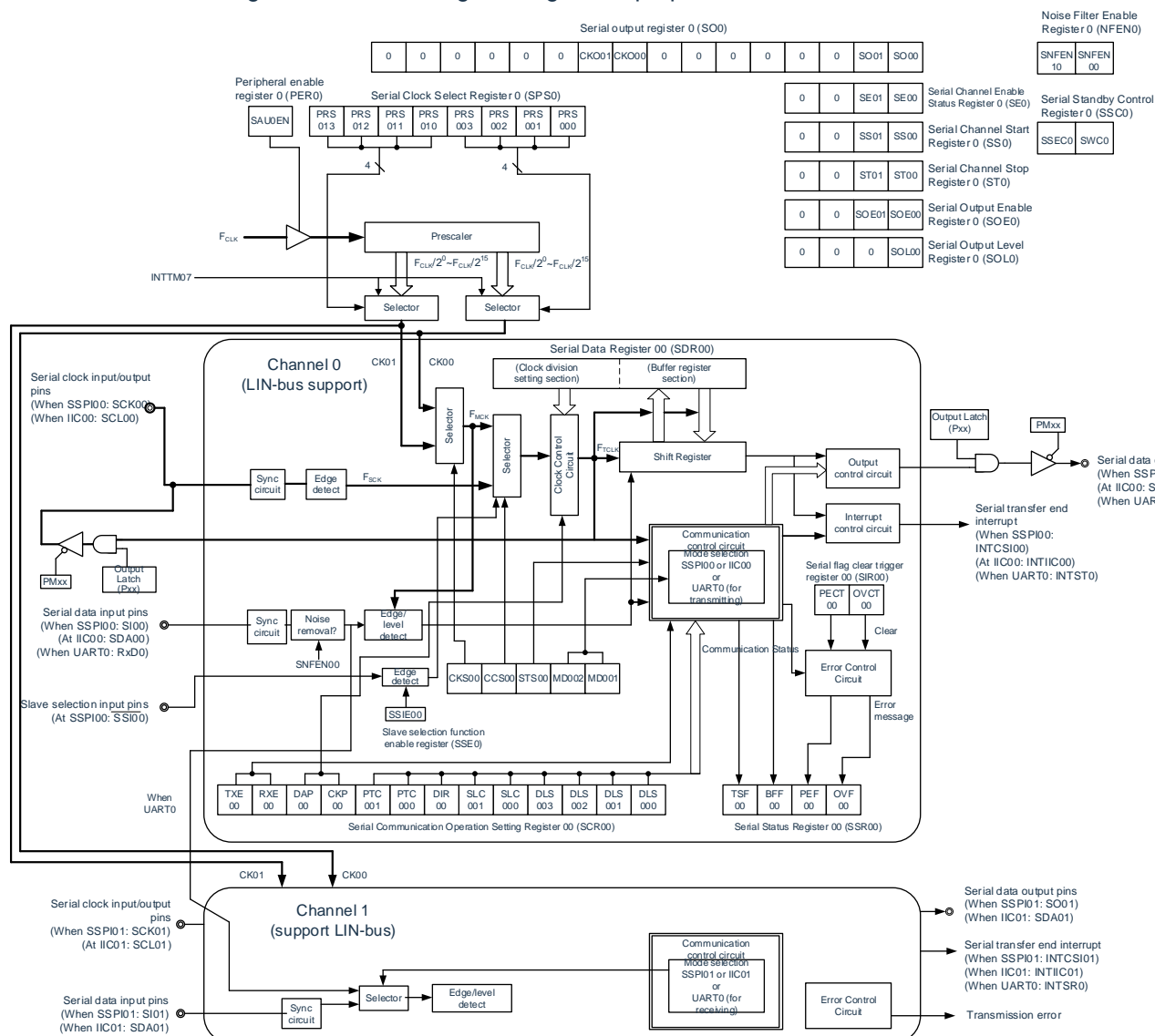
Note: When SE<sub>mn</sub> = 1.

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

(Unit 0 is used as an example)

Figure16-1: Block diagram of general-purpose serial communication unit



Note: Units 0, 1, and 2 have the same structure

## 16.2.1 Shift register

This is a 16-bit register that converts parallel data into serial data or vice versa.

During reception, it converts data input to the serial pin into parallel data. When data is transmitted, the value set to this register is output as serial data from the serial output pin. The shift register cannot be directly manipulated by program.

To read or write the shift register, use the serial data register mn (SDRmn) when operation is in progress (SEmn = 1).

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

## 16.2.2 Serial data register mn (SDRmn)

The SDRmn register is the transmit/receive data register (16 bits) of channel n.

If operation is stopped (SEmn = 0), bits 15 to 9 are used as a register that sets the division ratio of the operation clock( $F_{MCK}$ ). If operation is in progress (SEmn = 1), bit15~9 selected as a transmit/receive buffer register.

When receiving data, the parallel data converted by the shift register is saved to the serial data register SDRmn; when sending data, the send data that is transferred to the shift register is set to the serial data register SDRmn.

Regardless of the output order of the data, the data saved to the SDRmn register according to the setting of bit3 to bit0 (DLSmn3 to DLSmn0) of the serial communication run setting register mn (SCRmn) is shown below:

- 7-bit data length (stored in bit0~6 of SDRmn register)
- 8-bit data length (stored in bit0~7 of SDRmn register)
- 16-bit data length (stored in bit0~15 of SDRmn register)

The SDRmn register can be read or written in 16-bit units.

When SEmn=1, the lower 8 bits of the SDRmn register can be read and written in 8-bit increments as SDRmnL<sup>Note</sup>.

According to the communication mode, it can read and write SDRmnL registers with the following SFR names.

- SSPIp communication.....SDIOpL
- UARTq reception .....RXDq (UARTq receive data register)
- UARTq transmission .....TXDq (UARTq transmit data register)
- IICr communication .....SDIOr (IICr data register)

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

Note: At run stop (SEmn=0), it is forbidden to rewrite SDRmn[7:0] by 8-bit memory manipulation instruction (otherwise, all of SDRmn[15:9] will be cleared to "0").

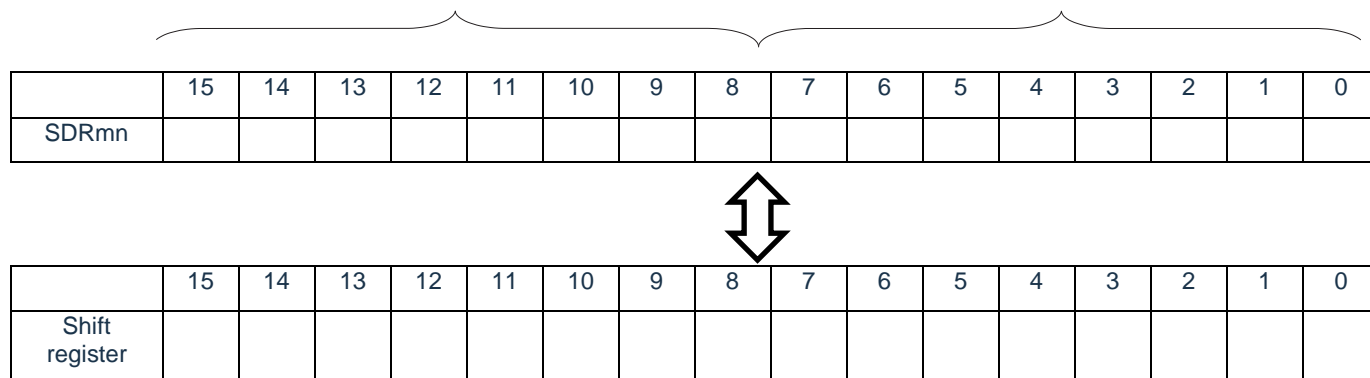
Remarks: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

q: UART number (q=0~2) r: IIC number (r=00, 01, 10, 11, 20, 21)

Figure16-2 : Format of the serial data register mn (SDRmn)

After reset: 0000H

R/W



Remark:

1. For the function of the higher 7 bits of the SDRmn register, please refer to "16.3 Registers for controlling general-purpose serial communication unit".
2. m: unit number (m=0, 1, 2) n: channel number (n=0, 1)

## 16.3 Registers for controlling general-purpose serial communication unit

The registers controlling general-purpose serial communication unit are shown below:

- Peripheral enable register 0 (PER0)
- Serial Clock Select Register m (SPSm)
- Serial Mode Registermn (SMRmn)
- Serial Communication Run Setting Registermn (SCRmn)
- Serial Data Registermn (SDRmn)
- Serial flag clear trigger register mn (SIRmn)
- Serial Status Registermn (SSRmn)
- Serial channel start register m (SSm)
- Serial channel stop register m (STm)
- Serial Channel Allowed Status Register m (SEm)
- Serial Output Allowance Register m (SOEm)
- Serial Output Level Register m (SOLm)
- Serial output register m (SOM)
- Serial Standby Control Register m (SSCm)
- Slave selection function enable register m (SSEm)
- Input switching control register (ISC)
- Noise Filter Allowance 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, 2) n: channel number (n=0, 1)

### Serial Communication Unit Register List

Unit 0 register base address: 0x40046000

Unit 1 register base address: 0x40046400

Unit 2 register base address: 0x40046800

Offset address	Register name	R/W	Reset value
0x000	SSRm0	R	0000H
0x002	SSRm1	R	0000H
0x004	SIRm0	R/W	0000H
0x006	SIRm1	R/W	0000H
0x008	SMRm0	R/W	0020H
0x00A	SMRm1	R/W	0020H
0x00C	SCRm0	R/W	0087H
0x00E	SCRm1	R/W	0087H
0x010	SEm	R	0000H
0x012	SSm	R/W	0000H
0x014	STm	R/W	0000H
0x016	SPSm	R/W	0000H
0x018	SOM	R/W	0303H
0x01A	SOEm	R/W	0000H
0x020	SOLm	R/W	0000H
0x022	SSEm	R/W	0000H
0x024	SSCm	R/W	0000H
0x040	SDRm0	R/W	0000H
0x040	SIOm0	R/W	00H
0x040	TXDm	R/W	00H
0x042	SDRm1	R/W	0000H
0x042	RXDm	R/W	00H
0x042	SIOm1	R/W	00H

Note: Unit number m=0, 1, 2



### 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 general-purpose serial communication unit 0, bit 2 (SCI0EN) must be set to "1".

To use general-purpose serial communication unit 1, bit 3 (SCI1EN) must be set to "1".

To use general-purpose serial communication unit 2, bit 4 (SCI2EN) 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".

Figure16-3: Format of peripheral enabled register 0 (PER0)

Address: 40020420H		After reset: 00000000H				R/W		
Symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	IICA0EN	IRDAEN	SCI2EN	SCI1EN	SCI0EN	TMAEN	TM80EN

SCI <sub>m</sub> EN	Provides control of the input clock of general-purpose serial communication unit m
0	Stop providing input clock. • The SFR used by the general-purpose serial communication unit m cannot be written. • The general-purpose serial communication unit m is in the reset state.
1	Enable the input clock to be provided. • Can read and write the SFR used by the general-purpose serial communication unit m.

Notice: To set the Universal Serial Communication Unit m, the following registers must be set in the state of SCI<sub>m</sub>EN bit "1" first. When the SCI<sub>m</sub>EN bit is "0", the write operation of the control registers of the Universal Serial Communication Unit m is ignored, and the read values are all initial values (input switching control register (ISC), noise filter allow register 0 (NFEN0), port multiplex function configuration register (PxxCFG), port output mode register (POMx), port mode register (PMx), and port mode register (PMx). Mode Register (POMx), Port Mode Register (PMx), Port Mode Control Register (PMCx), and Port Register (Px) are excluded).

- Serial Clock Select Register m (SPSm)
- Serial Mode Registermn (SMRmn)
- Serial Communication Run Setting Registermn (SCRmn)
- Serial Data Registermn (SDRmn)
- Serial flag clear trigger register mn (SIRmn)
- Serial Status Registermn (SSRmn)
- Serial channel start register m (SSm)
- Serial channel stop register m (STm)
- Serial Channel Allowed Status Register m (SEm)
- Serial Output Allowance Register m (SOEm)
- Serial Output Level Register m (SOLm)
- Serial output register m (SOM)

## 16.3.2 Serial clock selection register m (SPSm)

The SPSm register is a 16-bit register that selects two common operating clocks (CKm0, CKm1) available to each channel. Select CKm1 by bit7 to 4 of the SPSm register and select from bit3 to 0 CKm0.

It is forbidden to overwrite the SPSm register during operation (SEmn=1).

The SPSm register is set by means of a 16-bit memory manipulation instruction.

I can set the low 8 bits of the SPSm register with SPSmL and through the 8-bit memory manipulation instruction.

After generating a reset signal, the value of the SPSm register changes to "0000H".

Figure 16-4: Format of the serial clock selection register m (SPSm)

After reset: 0000H								R/W								
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPSm	0	0	0	0	0	0	0	0	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00

PRSmk3	PRSmk2	PRSmk1	PRSmk0	Select <sup>note</sup> for the running clock (CKmk).
0	0	0	0	F <sub>CLK</sub>
0	0	0	1	F <sub>CLK</sub> /2
0	0	1	0	F <sub>CLK</sub> /2 <sup>2</sup>
0	0	1	1	F <sub>CLK</sub> /2 <sup>3</sup>
0	1	0	0	F <sub>CLK</sub> /2 <sup>4</sup>
0	1	0	1	F <sub>CLK</sub> /2 <sup>5</sup>
0	1	1	0	F <sub>CLK</sub> /2 <sup>6</sup>
0	1	1	1	F <sub>CLK</sub> /2 <sup>7</sup>
1	0	0	0	F <sub>CLK</sub> /2 <sup>8</sup>
1	0	0	1	F <sub>CLK</sub> /2 <sup>9</sup>
1	0	1	0	F <sub>CLK</sub> /2 <sup>10</sup>
1	0	1	1	F <sub>CLK</sub> /2 <sup>11</sup>
1	1	0	0	F <sub>CLK</sub> /2 <sup>12</sup>
1	1	0	1	F <sub>CLK</sub> /2 <sup>13</sup>
1	1	1	0	F <sub>CLK</sub> /2 <sup>14</sup>
1	1	1	1	INTTM07

Note: When you change the clock selected as F<sub>CLK</sub> (change the value of the system clock control register (CKC)) during the operation of the general-purpose serial communication unit (SCI), you must stop the operation of the SCI (serial channel stop register m). (STm)=000FH) after making changes.

Notice: The bit 15~8 must be set to "0".

Remark:

1. F<sub>CLK</sub>: Clock frequency of CPU/peripheral hardware
2. m: Unit number (m=0~2).
3. k=0, 1

### 16.3.3 Serial mode register mn (SMRmn)

The SMRmn register is a register that sets the operating mode of channel n, selects the operating clock ( $F_{MCK}$ ), specifies whether the serial clock ( $F_{SCLK}$ ) input can be used, sets the start of triggering, and operates the mode settings (SSPI, UART, Simple I2C) and selection of interrupt sources. In addition, the inverting level of the received data is set only in UART mode.

It is forbidden to overwrite the SMRmn register during operation ( $SEmn=1$ ), but it is possible to overwrite the MDmn0 bit during operation.

The SMRmn register is set by means of a 16-bit memory manipulation instruction.

After generating a reset signal, the value of the SMRmn register changes to "0020H".

Figure 16-5: Serial mode register mn (SMRmn) (1/2)

After reset: 0020H										R/W						
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn	CCSmn	0	0	0	0	0	STSmnNote1	0	SISmn0Note1	1	0	0	MDmn2	MDmn1	MDmn0

CKSmn	Selection of channel n operating clock ( $F_{MCK}$ )
0	The SPSm register sets the operating clock CKm0
1	The SPSm register sets the operating clock CKm1
The operating clock ( $F_{MCK}$ ) is used for edge detection circuitry. By setting the CCSmn bit and the SDRmn register higher 7 bits, a transmit clock ( $F_{TCLK}$ ) is generated.	

CCSmn	Selection of channel n transmission clock ( $F_{TCLK}$ )
0	The CKSmn bit specifies the running clock $F_{MCK}$ divider clock
1	Input clock $F_{SCLK}$ from the SCLKp pin (slave transfer in SSPI mode).
The transmit clock $F_{TCLK}$ is used for shift registers, communication control circuits, output controllers, interrupt control circuits, and error control circuits. When the CCSmn bit is at "0", the operating clock ( $F_{MCK}$ ) is set the dividing ratio of the operating clock ( $F_{MCK}$ ) by the higher 7 bits of the SDRmn register.	

STSmn <sup>Note1</sup>	Selection of start trigger sources
0	Only software triggers are valid (selected in SSPI, UART transmit, simplified I2C).
1	The effective edge of the RxDq pin (selected when received by the UART).
When the above conditions are met after setting the SSm register to "1", the transfer starts.	

Note 1: Limited to SMR01, SMR11, AND SMR21 registers only.

Notice: Bit13~9, 7, 4, 3 (SMR00, SMR10, SMR20 registers are bit13~6, 4, 3) are set "0" and set bit 5 to "1".

Remark: m: unit number(m=0, 1, 2) n: channel number(n=0, 1) 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 16-5: Format of serial mode register mn (SMRmn) (2/2)

After reset: 0020H

R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKS mn	CCS mn	0	0	0	0	0	STS mn <small>Note1</small>	0	SISmn 0 <small>Note1</small>	1	0	0	MD mn2	MD mn1	MD mn0

SISmn0	Level inversion control of channel n receives data in UART mode
0	Detect the falling edge as the starting bit. The input communication data is not inverted.
1	Detects the rising edge as the starting bit. Invert the input communication data.

MDmn2	MDmn1	Setting of channel n operation mode
0	0	SSPI mode
0	1	UART mode
1	0	Simple I <sup>2</sup> C mode
1	1	Settings are disabled.

MDmn0	Channel n interrupt source selection
0	Transfer end interrupt
1	Buffer null interrupt (Occurs when data is transferred from the SDRmn register to the shift register).

On consecutive sends, if the MDmn0 bit is "1" and the data for SDRmn is empty, write down the next send data.

Note 1: Limited to SMR01, SMR11, SMR21 registers.

Notice: Bit13~9, 7, 4, 3 (SMR00, SMR10, SMR20 registers are bit13~6, 4, 3) are set to "0" and set bit 5 to "1".

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

## 16.3.4 Serial communication run setting register mn (SCRmn)

The SCRmn register is the communication operation setting register of channel n, which sets the data transmission and reception modes, data and clock phases, whether to mask the error signal, parity test bits, start bits, stop bits, and data length.

It is forbidden to overwrite the SCRmn register during operation (SEmn=1).

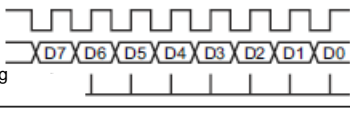
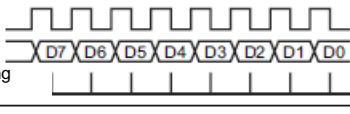
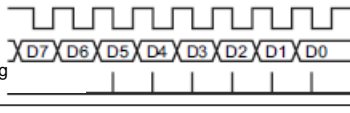
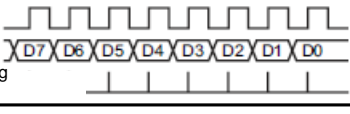
The SCRmn register is set by means of a 16-bit memory manipulation instruction.

After generating a reset signal, the value of the SCRmn register changes to "0087H".

Figure 16-6: Format of serial communication run setting register mn (SCRmn) (1/3)

After reset: 0087H																R/W	
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
SCRmn	TXEmn	RXEmn	DAPmn	CKPmn	0	0	PTCmn1	PTCmn0	DIRmn	0	SLCmn1	SLCmn0	DLSmn3	DLSmn2	DLSmn1	DLSmn0	

TXEmn	RXEmn	Setting of channel n operation mode
0	0	Prohibited communication.
0	1	Receive only.
1	0	Send only.
1	1	Enable sending and receiving.

DAPmn	CKPmn	data and clock phase selection in SSPI mode	Type
0	0		1
0	1		2
1	0		3
1	1		4

in UART mode and simple I2C mode, must set DAPmn bit and CKPmn bit both to 0.

Note 1: Limited to SCR00, SCR10, SCR20 registers.

Notice: bit6, 10, 11 must be set to "0" (also bit5 of SCR01, SCR11, SCR21 registers must be set to "0").

Remark: m: unit number(m=0, 1, 2) n: channel number(n=0, 1) p: SSPI number(p=00, 01, 10, 11, 20, 21)

Figure16-6: Format of serial communication run setting register mn (SCRmn) (2/3)  
After reset: 0087H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXE mn	RXE mn	DAP mn	CKP mn	0	0	PTC mn1	PTC mn0	DIR mn	0	SLCm n1 <sup>Note 1</sup>	SLC mn0	DLS mn3	DLS mn2	DLS mn1	DLS mn0

PTCmn1	PTCmn0	Setting of parity bits in UART mode	
		Send	Receiving
0	0	No parity bits are output	No parity check on reception
0	1	Output parity <sup>Note 3</sup>	Does not determine parity
1	0	Output even-check	Determine the even parity
1	1	Output odd checksum	Determine the odd checksum
In SSPI mode and Simple I <sup>2</sup> C mode, both PTCmn1 and PTCmn0 bits must be set to "0".			

DIRmn	Selection of data transfer sequence in SSPI and UART modes
0	Perform MSB-first input/output.
1	Perform LSB-first input/output.
In the simple I <sup>2</sup> C mode, the DIRmn bit must be set to "0".	

SLCmn1 <sup>Note 1</sup>	SLCmn0	Setting of the stop bit in UART mode
0	0	No stop bit
0	1	Stop bit length = 1 bit
1	0	Stop bit length = 2 bits (mn=00, 10, 20 only)
1	1	Prohibit setting.
If the transfer end interrupt is selected, the interrupt is generated after all stop bits have been transferred. It must be set to 1 stop bit (SLCmn1, SLCmn0=0, 1) during UART receive or in simple I <sup>2</sup> C mode. In SSPI mode, it must be set to no stop bit (SLCmn1, SLCmn0 = 0, 0). It must be set to 1 bit (SLCmn1, SLCmn0=0, 1) or 2 bits (SLCmn1, SLCmn0=1, 0) when UART is sent.		

Note 1: SCR00, SCR10, SCR20 registers only.

Note 2: It is always appended with "0", regardless of the content of the data.

Notice: bits 6, 10 and 11 must be set to "0".

Remark: m: unit number(m=0, 1, 2) n: channel number(n=0, 1) p: SSPI number(p=00, 01, 10, 11, 20, 21)

Figure16-6: Format of serial communication run setting register mn (SCRmn) (3/3)  
After reset: 0087H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXE mn	RXE mn	DAP mn	CKP mn	0	0	PTC mn1	PTC mn0	DIR mn	0	SLCm n1 <sup>Note1</sup>	SLC mn0	DLS mn3	DLS mn2	DLSm n1 <sup>Note2</sup>	DLS mn0

DLS mn3	DLS mn2	DLS mn1	DLS mn0	The setting of the data length	Serial function correspondence		
					SSPI	UART	IIC
0	1	1	0	7 bits of data length (bit0 to 6 stored in the SDRmn register).	○	○	×
0	1	1	1	8 bits of data length (bit0 to 7 saved in the SDRmn register).	○	○	○
1	0	0	0	9 bits of data length (bit0 to 8 saved in the SDRmn register).	○	○	×
1	0	0	1	10 bits of data length (bit0 to 9 saved in the SDRmn register).	○	×	×
1	0	1	0	11 bits of data length (bit0 to 10 stored in the SDRmn register).	○	×	×
1	0	1	1	12-bit data length (bit0 to 11 stored in the SDRmn register).	○	×	×
1	1	0	0	13 bits of data length (bit0 to 12 saved in the SDRmn register).	○	×	×
1	1	0	1	14-bit data length (bit0 to 13 stored in the SDRmn register).	○	×	×
1	1	1	0	15 bits of data length (bit0 to 14 saved in the SDRmn register).	○	×	×
1	1	1	1	16-bit data length (bit0 to 15 stored in the SDRmn register).	○	○	×
Others:				Settings are disabled.			
In the simple I <sup>2</sup> C mode, DLSmn3~ DLSmn0=0111B must be set.							

Note 1: SCR00, SCR10, SCR20 registers only.

Note: bits 6, 10 and 11 must be set to "0".

Remark: m: unit number(m=0, 1, 2) n: channel number(n=0, 1) p: SSPI number(p=00, 01, 10, 11, 20, 21)

### 16.3.5 Serial data register mn (SDRmn)

The SDRmn register is the data register (16-bit) that channel n sends and receives.

When the operation stops (SEmn=0), bit15~9 is used as a crossover setting register for the operating clock (F<sub>MCK</sub>). During operation (SEmn=1) bit15~9 is used as a transmit and receive buffer register.

If the CCSmn bit of the serial mode register mn (SMRmn) is "0", the bit15 to 9 of the SDRmn register is used as the divider clock of the operating clock (higher 7 bits) is used as the transmission clock.

The SIRmn register is set by means of a 16-bit memory manipulation instruction.

After generating a reset signal, the value of the SDRmn register changes to "0000H".

Figure 16-7: Serial data register mn (SDRmn)

After reset: 0000H								R/W								
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn																

SDRmn[15:9]							Transmission clock setting for running clock division
0	0	0	0	0	0	0	F <sub>MCK</sub>
0	0	0	0	0	0	1	F <sub>MCK</sub> /2
0	0	0	0	0	1	0	F <sub>MCK</sub> /3
0	0	0	0	0	1	1	F <sub>MCK</sub> /4
•	•	•	•	•	•	•	•
1	1	1	1	1	1	0	F <sub>MCK</sub> /127
1	1	1	1	1	1	1	F <sub>MCK</sub> /128

Notice:

- When operation is stopped (SEmn=0), bit8~0 must be cleared to zero.
- When using UART, it is prohibited to set SDRmn[15:9] to "00000000B" and "000000001B".
- When using Simple I<sup>2</sup>C, it is prohibited to set SDRmn[15:9] to "0000000B", and the setting value of SDRmn[15:9] must be greater than or equal to "0000001B".
- When operation is stopped (SEmn=0), it is prohibited to rewrite SDRmn[7:0] by 8-bit memory manipulation instruction (otherwise, all of SDRmn[15:9] is cleared to "0").

Remark:

- For the function of the SDRmn register during operation, please refer to "16.2 Structure of general-purpose serial communication unit".
- m: unit number(m=0, 1, 2) n: channel number(n=0, 1)



### 16.3.6 Serial flag clear trigger register mn(SIRmn)

This is a trigger register used to clear each error flag for channel n.

If each bit (FECTmn, PECTmn, OVCTmn) is set to "1", the corresponding bits (FEFmn, PEFmn, OVFmn) of the serial status register mn (SSRmn) are cleared to "0". Since SDIRmn register is a trigger register, if the corresponding bit of SSRmn register is cleared, SDIRmn register will also be cleared immediately.

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

It is possible to set the lower 8 bits of the SIRmn register with SIRmnL and by an 8-bit memory manipulation instruction.

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

Figure 16-8: Format of serial flag clear trigger register mn (SIRmn)

After reset: 0000H

R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIRmn	0	0	0	0	0	0	0	0	0	0	0	0	0	FECTmn <sup>Note 1</sup>	PEC Tmn	OVC Tmn

FECTmn <sup>Note 1</sup>	Channel n frame error flag clear trigger
0	No clearance.
1	Clear the FEFmn bit of the SSRmn register to "0".

PECTmn	Channel n parity error flag clear trigger
0	No clearance.
1	Clear the PEFmn bit of the SSRmn register to "0".

OVCTmn	Channel n overflow error flag clear trigger
0	No clearance.
1	Clear the OVFmn bit of the SSRmn register to "0".

Note 1: Limited to SIR01, SIR11, SIR21 registers.

Notice: bit15~3 (SIR00, SIR10, SIR20 registers are bit15~2) must be set to "0"

Remark:

1. m: unit number(m=0, 1, 2) n: channel number(n=0, 1)
2. The read value of SIRmn register is always "0000H".

## 16.3.7 Serial status register mn (SSRmn)

The SSRmn register indicates the communication status of channel n and the condition in which an error occurred. Errors represented are frame errors, parity errors, and overflow errors. Read the SSRmn registers via a 16-bit memory manipulation instruction.

It can read the lower 8 bits of the SSRmn register with SSRmnL and read the SSRmn register through the 8-bit memory manipulation instruction.

After generating a reset signal, the value of the SSRmn register changes to "0000H".

Figure16-9: Serial status register mn (SSRmn) (1/2)

After reset: 0000H										R						
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSRmn	0	0	0	0	0	0	0	0	0	TSF mn	BFF mn	0	0	FEF Mn <sup>Note1</sup>	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 SSmn bit of SSm register is set to "1" (communication standby state) • When the communication ends [Set Condition]. • When communication begins	

BFFmn	Indication flag for channel n buffer register
0	The SDRmn register does not hold valid data.
1	The SDRmn register holds valid data.
[Clear Condition] • When the transmit data from the SDRmn register to the shift register is completed during the send process • When the received data is read from the SDRmn register during the receive process • When the STmn bit of STmn register is set to "1" (communication stopped state) or the SSm bit of the SSm register is set to "1" (Communication enable state) [Set Condition] • When writing and transmitting data to the SDRmn register in the state where the TXEmn bit of the SCRmn register is "1" (the transmit mode, transmit and receive mode in each communication mode). • When saving the receive data to the SDRmn register in the state where the RXEmn bit of the SCRmn register is "1" (receive mode, transmit and receive mode in each communication mode). • When a receive error occurs	

Note 1: Limited to SSR01, SSR11, SSR21 registers.

Notice: If the SDRmn register is written while the BFFmn bit is "1", the saved transmit or receive data is discarded and an overflow error is detected (OVEmn=1).

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

Figure16-9: Serial status register mn (SSRmn) (2/2)

After reset: 0000H

R

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSRmn	0	0	0	0	0	0	0	0	0	TSF mn	BFF mn	0	0	FEF mn <sup>Note 1</sup>	PEF mn	OVF mn

FEFmn <sup>Note 1</sup>	Detection flag for channel n frame errors
0	No errors occurred.
1	An error occurred (when the UART was received).
[Clear Condition]. • When writing "1" to the FECTmn bit of the SIRmn register [Set Condition]. • When no stop bit is detected at the end of UART reception	

PEFmn	Detection flag for channel n parity errors
0	No errors occurred.
1	An error occurred (when the UART was received) or the ACK was not detected (when the I <sup>2</sup> C was sent).
[Clear Condition]. • When the PECTmn bit of the SIRmn register is written to "1" [Set Condition]. • When the parity and parity bits of the sent data are different at the end of the UART reception (parity error). • When the I <sup>2</sup> C is sent and when the ACK receives the timing slave does not return an ACK signal (ACK not detected).	

OVFmn	Detection flag for channel n overflow error
0	No errors occurred.
1	An error occurred.
[Clear Condition]. • When writing "1" to the OVCTmn bit of the SIRmn register [Set Condition]. • In the state where the RXEmn bit of the SCRmn register is "1" (receive mode, transmit and receive mode in each communication mode), although the received data is saved in the SDRmn register, But when the received data is not read and the sending data is written or written down the next received data • When data is not ready to be sent during a slave send in SSPI mode or during a slave send and receive	

Note 1: Limited to SSR01, SSR11, SSR21 registers.

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

### 16.3.8 Serial channel start register m(SSm)

The SSm register is a trigger register to set the enable communication/start count for each channel.

If a "1" is written to each bit (SSmn), the corresponding bit (SEmn) in the serial channel enable status register m (SEm) is set to a "1" (operation enable status). Since the SSmn bit is a trigger bit, the SSmn bit is cleared immediately if the SEmn bit is a "1".

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

It is possible to use SSmL and set the lower 8 bits of SSm register by an 8-bit memory manipulation instruction.

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

Figure 16-10: Format of the serial channel start register m (SSm)

After reset: 0000H															R/W	
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SSm1	SSm0

SSmn	Trigger at the beginning of channel n operation
0	No triggering.
1	Set the SEmn bit "1" and shift to communication standby state <sup>Note</sup> .

Note: If the SSmn bit is set to "1" during communication, communication will be stopped and enter standby state. At this time, the values of control register and shift register, SCLKmn pin and SDOmn pin, FEFmn flag, PEFmn flag and OVFn flag remain in state.

Notice: The bit15~2 of SSm register must be set to "0".

Remark:

1. m: unit number(m=0, 1, 2) n: channel number(n=0, 1)
2. The read value of SSm register is always "0000H".

### 16.3.9 Serial channel stop register m(STm)

The STm register is a trigger register for setting to enable communication/stop count for each channel.

If a "1" is written to each bit (STmn), the corresponding bit (SEmn) in the serial channel enable status register m (SEm) is cleared to "0" (stop status). Since the STmn bit is a trigger bit, if the SEmn bit is "0", the STmn bit is cleared immediately.

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

The lower 8 bits of the STm register can be set using STmL and by an 8-bit memory manipulation instruction.

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

Figure 16-11: Format of the serial channel stop register m (STm)

After reset: 0000H								R/W							
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
STm														STm1	STm0

STmn	Stop trigger for channel n operation
0	No triggering.
1	Clear the SEmn bit to "0" to stop the communication operation <sup>Note</sup> .

Note: The control register and shift register values, the SCLKmn pin and SDOmn pin, and the FEFmn flag, PEFmn flag, and OVFmn flag hold status.

Notice: The bit15~2 of STm register must be set to "0".

Remark:

1. m: unit number(m=0, 1, 2) n: channel number(n=0, 1)
2. The read value of STm register is always "0000H".

## 16.3.10 Serial channel enable status register m (SEm)

The SEm register is used to confirm the allow or stop status of serial transmit and receive for each channel.

If "1" is written to each of the serial start allow register m (SSm), the corresponding bit is set to "1". If you write "1" to each bit of the serial channel stop register m (STm), the corresponding bit is cleared to "0".

For channel n, which is allowed to run, the value of the CKOmn bit (serial clock output of channel n) of the serial output register m (SOM), described later, cannot be rewritten by software, and the value reflected by the communication run is output from the serial clock pin.

The value of the CKOmn bit in the SOM register can be set by software for channel n that is stopped, and the value is output from the serial clock pin. Thus, arbitrary waveforms such as start conditions or stop conditions can be generated by software.

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

The lower 8 bits of the SEm register can be read with SEmL and by 8-bit memory manipulation instructions.

After the reset signal is generated, the value of SEm register becomes "0000H".

Figure 16-12: Format of serial channel enable status register m (SEm)

After reset: 0000H														R		
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SEm1	SEm0

SEmn	Indication of the enable or stop state of channel n operation
0	Stop state
1	Operation enable state

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

## 16.3.11 Serial output enable register m (SOEm)

The SOEm register setting enable or stops the output of serial communication for each channel.

For channel n that enable serial output, the value of the SOMn bit of the serial output register m (SOM) described below cannot be rewritten by software, but the value reflected by the communication operation is output from the serial data output pin.

For channel n that stops the serial output, the value of the SOMn bit of the SOM register can be set by software and output from the serial data output pin. Thus, arbitrary waveforms such as start conditions or stop conditions can be generated by software.

The SOEm register is set by the 16-bit memory manipulation instruction.

I can set the low 8 bits of the SOEm register with SOEmL and through the 8-bit memory manipulation instruction.

After generating a reset signal, the value of the SOEm register changes to "0000H".

Figure16-13: Format of serial output enable register m (SOEm)

After reset: 0000H															R/W	
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE m1	SOE m0

SOEmn	Channel n serial output enable or stop
0	Stop the output of serial communication.
1	Enable the output of serial communication.

Notice: Bit15~2 of SOEm register is set to "0".

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

## 16.3.12 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 CKOm bit of this register is output from the serial clock output pin of channel n.

The SOMn bit of this register can be rewritten by software only when serial output is disabled (SOEmn=0).

When serial output is allowed (SOEmn=1), the value of the SOMn bit of this register can only be changed by serial communication, ignoring software rewriting.

The CKOm bit of this register can be rewritten by software only when the channel is stopped (SEmn=0).

When the channel is allowed to run (SEmn=1), software rewriting is ignored and the value of the CKOm bit in this register can only be changed via serial communication.

To use the pins of the serial interface for non-serial interface functions such as port functions, the corresponding CKOm bit and SOMn bit must be set to "1".

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

After the reset signal is generated, the value of SOM register changes to "0303H".

Figure 16-14: Format of serial output register m (SOM)

After reset: 0303H						R/W										
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S0m	0	0	0	0	0	0	CKOm1	CKOm0	0	0	0	0	0	0	S0m1	S0m0

CKOm <sub>n</sub>	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".

SOM <sub>n</sub>	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: The bit15~10 and bit7~2 of SOM register must be set to "0".

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



### 16.3.13 Serial output level register m (SOLm)

The SOLm register is a register for setting the data output level inversion for each channel.

This register can be set only in UART mode. In SSPI mode and simple I<sup>2</sup>C mode, the corresponding bits must be set to "0". Only when serial output is allowed (SOEmn=1), set the inverse of each channel n of this register to reflect to the pin output. When serial output is disabled (SOEmn=0), the value of the SOMn bit is output directly. It is prohibited to rewrite the SOLm register during operation (SEmn=1).

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

It is possible to set the lower 8 bits of SDOLm register with SOLmL and by 8-bit memory manipulation instruction.

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

Figure 16-15: Format of the serial output level register m (SOLm)

After reset: 0000H								R/W								
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOLm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOLm <sub>0</sub>

SOLmn	Selection of channel n transmit data level inversion in UART mode
0	Output the communication data directly.
1	Invert the communication data to output.

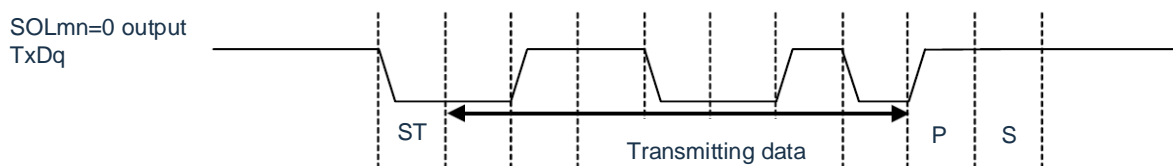
Notice: The bit15~1 of SOL0, SOL1 and SOL2 registers must be set to "0".

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

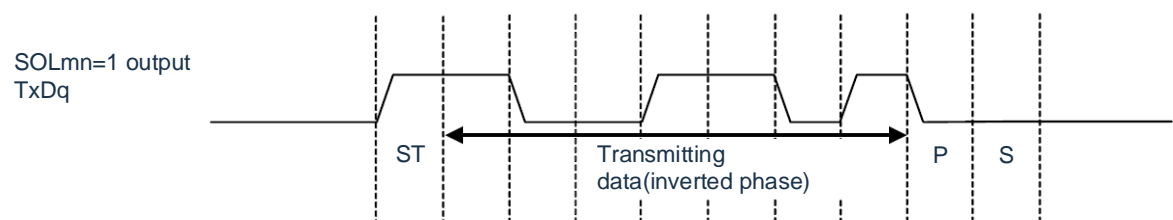
When UART transmission is performed, an example of level inversion of the transmitted data is shown inFigure16-16 shown in Figure16-16.

Figure16-16: Example of level inversion for sending data

(a) Normal phase output(SOLmn=0)



(b) Inverted output (SOLmn=1)



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

## 16.3.14 Serial standby control register m (SSCm)

The SSC0 register is the register that controls the start of receive operation from deep sleep mode during serial data reception at SSPI00 or UART0.

The SSC1 register is the register that controls the start of receive operation from deep sleep mode during serial data reception on SSPI10 or UART1.

The SSC2 register is the register that controls the start of receive operation from deep sleep mode during serial data reception on the SSPI20 or UART2.

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

The low 8 bits of the SSCm register can be set with SSCmL and by 8-bit memory manipulation instruction.

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

Figure 16-17: Format of serial standby control register m (SSCm)

	After reset: 0000H								R/W							
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSCm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SSEC <sub>m</sub>	SWCm

SSECm	Selection of enable or stop for communication error interrupt generation in deep sleep wakeup mode
0	Enable generation of error interrupts.
1	Disable error interrupt generation.
<ul style="list-style-type: none"> <li>In the case of UART reception in UART low power mode, the SSECm bit can only be set when the SWCm bit and EOCm bit are "1" or "0", otherwise the SSECm bit must be "0".</li> <li>Disable setting the SSECm bit and SWCm bit to "1" and "0" respectively.</li> </ul>	

SWCm	Setting of low power UART mode
0	Deep sleep wake up mode function is not used.
1	Use the deep sleep wake up mode function.
<ul style="list-style-type: none"> <li>In deep sleep mode, the deep sleep mode is released by a hardware trigger signal, and UART reception is performed without the CPU running. (Low-power UART mode). <ul style="list-style-type: none"> <li>The low-power UART mode function can be set only when the high-speed internal oscillator clock is selected as the CPU/peripheral hardware clock (F<sub>CLK</sub>), and when the setting is disabled in the case of other clocks.</li> <li>Even if you use the low-power UART mode, you must set SWCm bit to "0" in normal operation mode and before you are about to move to deep sleep mode. Set the SWCm bit to "1" Also, the SWCm bit must be set to "0" after returning from deep sleep mode to normal operation mode.</li> </ul> </li> </ul>	

## 16.3.15 Slave select function enable register m (SSEm)

The SSEm register controls whether the input to the SSImn terminal is valid or invalid when used as a slave function in SSPImn communication.

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

The lower 8 bits of the SSCm register can be set with SSCmL and by an 8-bit memory manipulation instruction.

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

Figure 16-18: Format of slave select enable register m (SSEm)

	After reset: 0000H								R/W							
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSEm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SSIEm 1	SSIEm 0

SSIEm	Setting of SSIm input of channel n in slave mode for SSPImn communication
0	SSImn pin input is invalid.
1	SSImn pin input valid.

## 16.3.16 Input switching control register (ISC)

When LIN-bus communication is implemented via UART0, the ISC1 bits and ISC0 bits of the ISC registers are used for coordination of external interrupts and timer array units. If bit0 is placed at "1", the input signal of the serial data input (RxD0) pin is selected as the input to the external interrupt (INTP0), so it can pass THE INTP0 interrupt detects the wake-up signal.

If bit1 is set to "1", the input signal of the serial data input (RxD0) pin is selected as the input to the timer, so the wake signal can be detected by the timer and the low width of the interval segment and the pulse width of the synchronization segment can be measured.

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

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

Figure16-19: Table of input switching control register (ISC)

Address: 40040C03H		After reset: 00H			R/W			
Symbol	7	6	5	4	3	2	1	0
ISC	0	0	0	0	0	0	ISC1	ISC0

ISC1	Input switching of channel 3 of Timer8
0	Use the input signal from the TI03 pin as an input to the timer (normally operating).
1	Use the input signal on the RxD0 pin as a timer input (detects the wake-up signal and measures the low level width of the break field and the pulse width of the sync field).

ISC0	Input switching of external interrupt (INTP0)
0	Use the input signal on the INTP0 pin as an external interrupt input (normally operating).
1	Use the input signal on the RxD0 pin as an external interrupt input (detecting a wake-up signal).

Notice: The bit 6~2 must be set to "0".

## 16.3.17 Noise filter enable register 0 (NFEN0)

The NFEN0 register sets whether the noise filter is used for the input signal of each channel's serial data input pin.

For pins used for SSPI or simple I<sup>2</sup>C communication, the corresponding bit must be "0" to invalidate the noise filter. For pins used for UART communication, the corresponding bit "1" must be set to make the noise filter effective.

When the noise filter is active, the 2 clocks are detected to be consistent after synchronization through the operating clock ( $F_{MCK}$ ) of the object channel; When the noise filter is invalid, synchronization is performed only through the operating clock ( $F_{MCK}$ ) of the object channel.

The NFEN0 register is set via the 8-bit memory manipulation instruction.

After generating a reset signal, the value of the NFEN0 register changes to "00H".

Figure 16-20: Format of noise filter enable register 0 (NFEN0)

Address: 40040C00H			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".	

Notice: bit7, 5, 3, 1 must be set to "0".

## 16.4 Operation stop mode

Each serial interface of a general-purpose serial communication unit has an operation stop mode. Serial communication is not possible in operation stop mode, so power consumption is reduced. In addition, pins for the serial interface can be used as port functions in operation stop mode.

### 16.4.1 Stopping the operation by units

The unit stop is set by peripheral enable register 0/2 (PER0/2).

Per0/2 registers are registers that are set to enable or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocks to unused hardware.

To stop general-purpose serial communication unit 0, the bit2 (SCI0EN) of PER0 must be set to "0"; To stop general-purpose serial communication unit 1, the bit3 (SCI1EN) of PER0 must be set to "0"; To stop general-purpose serial communication unit 2, bit4 (SCI2EN) of PER2 must be set to "0".

Peripheral enable register 0 (per 0) ... only the corresponding bit set to "0" of SCIm will be stopped.

Figure 16-21: Setting of peripheral enable register 0 (PER0) when stopping operation by unit

	7	6	5	4	3	2	1	0
PER0	XX	XX	XX	SCI2EN	SCI1EN	SCI0EN	XX	XX

Control of SCIm input clock

0: Stop providing input clock

1: Supply input clock

Notice: When the SCImEN bit is "0", the write operation of the control register of the general-purpose serial communication unit m is ignored, and the read values are initial values. However, the following registers are excluded:

- Input switching control register (ISC)
- Noise filter enable register (NFEN0)
- Port multiplexing function configuration register (PxxCFG)
- Port output mode register (POMx)
- Port output mode register (PMx)
- Port mode register (Px)

Remark: x: Bits are not used by the general-purpose serial communication units (depending on the setting of other peripheral functions).

0/1: Set "0" or "1" depending on the usage of the user.

## 16.4.2 Stopping the operation by channels

The stopping of the operation by channels is set using each of the following registers.

Figure16-22: Each register setting when stopping the operation by channels

(a) Serial channel stop register m (STm) ... This register is a trigger register that is used to enable stopping communication/count by each channel.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	STm1	STm0
															0/1	0/1

1: Clears the SE<sub>mn</sub> bit to 0 and stops the communication operation

※Because the ST<sub>mn</sub> bit is a trigger bit, it is cleared immediately when SE<sub>mn</sub> = 0.

(b) Serial channel enable status register m (SEm) ... This register indicates whether data transmission/reception operation of each channel is enabled or stopped.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SEm1	SEm0
															0/1	0/1

0: Operation stops

※The SEm register is a read-only status register, whose operation is stopped by using the STm register.  
With a channel whose operation is stopped, the value of the CKOm<sub>n</sub> bit of the SOM register can be set by software.

(c) Serial output enable register m (SOEm) ... This register is a register that is used to enable or stop output of the serial communication operation of each channel.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOEm1	SOEm0
															0/1	0/1

0: Stops output by serial communication operation

※For channel n, whose serial output is stopped, the SOM<sub>n</sub> bit value of the SOM register can be set by software.

(d) Serial output register m (SOM) ... This register is a buffer register for serial output of each channel.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOM	0	0	0	0	0	0	CKOm1	CKOm0	0	0	0	0	0	0	SOM1	SOM0
							0/1	0/1							0/1	0/1

1: Serial clock output value is "1"      1: Serial data output value is "1"

※When using pins corresponding to each channel as port function pins, set the corresponding CKOm<sub>n</sub>, SOM<sub>n</sub> bits to "1".

Note: Limited to general-purpose serial communication unit0.

Remark:

1. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)
2. ■: Setting disabled (set initial value). 0/1: Set "0" or "1" according to the user's purpose.

## 16.5 3-wire serial I/O(SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) communication

This is a clock synchronization communication function implemented by three lines of serial clock (SCLK) and serial data (SDI and SDO).

[Data transmission and reception]

- 7~16 bit data length
- Phase control of transmit/receive data
- MSB/LSB preferred option

[Clock Control]

- Master/slave selection
- Phase control of I/O clock
- Setting of transfer period by prescaler and internal counter
- Maximum transfer rate <sup>Note</sup>

During master communication: Max.  $F_{CLK}/2$

During slave communication: Max.  $F_{MCK}/6$

[Interrupt function]

- Transfer end interrupt, buffer empty interrupt

[Error detection flag]

- Overflow error

Note: Must be used within the range that satisfies the SCLK cycle time (tKCY) characteristic. For details, please refer to the data sheet.

Channel 0~1 of SCI0, Channel 0~1 of SCI1 and Channel 0~1 of SCI2 are channels that support 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21).

The 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) has the following 6 communication operations:

- Master transmission (refer to 16.5.1)
- Master reception (refer to 16.5.2)
- Master transmission and reception (refer to 16.5.3)
- Slave transmission (refer to 16.5.4)
- Slave reception (refer to 16.5.5)
- Slave transmission and reception (refer to 16.5.6)



## 16.5.1 Master transmission

Master transmission refers to the operation of this product output transmission clock and sending data to other devices.

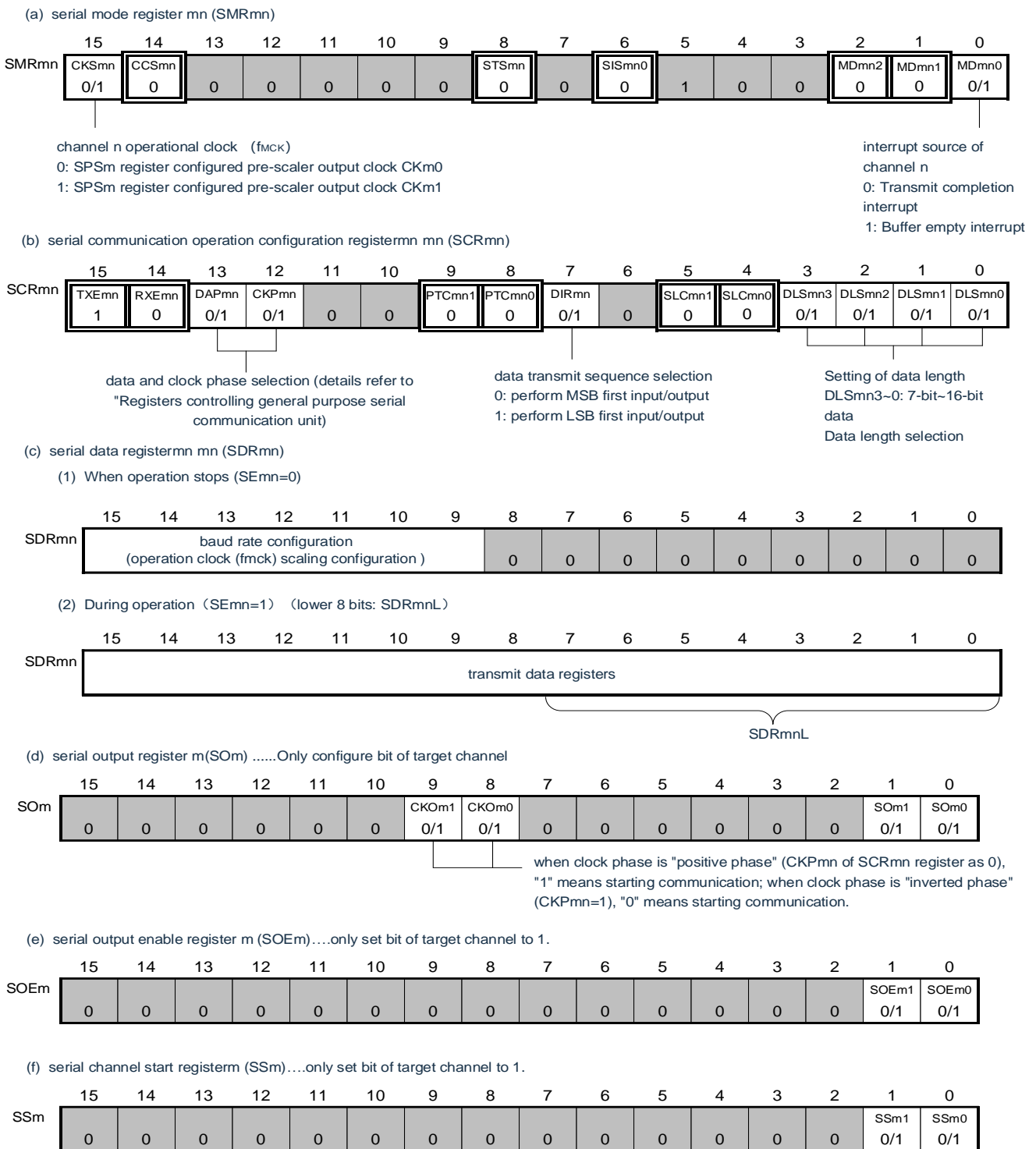
3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channels	Channel 0 of SCI0	Channel 1 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1	Channel 0 of SCI2	Channel 1 of SCI2
The pins used	SCLK00 SDO00	SCLK01 SDO01	SCLK10 SDO10	SCLK11 SDO11	SCLK20 SDO20	SCLK21 SDO21
Interrupt	INTCSI00	INTCSI01	INTCSI10	INTCSI11	INTCSI20	INTCSI21
	Selectable transmission end interrupt (single transmission mode) or buffer empty interrupt (continuous transmission mode).					
Error detection flags	None					
Transmitted data length	7~16 bits					
Transmission rate <sup>Note</sup>	Max.F <sub>CLK</sub> /2[Hz]					
	Min.F <sub>CLK</sub> /(2x2 <sup>11</sup> x128) [Hz] F <sub>CLK</sub> : System clock frequency					
Data Phase	It can be selected by the DAPmn bit of the SCRmn register. • DAPmn=0: Start data output when the serial clock starts running. • DAPmn=1: Start data output half a clock before the serial clock starts running.					
Clock Phases	It can be selected by the CKPmn bit of the SCRmn register. • CKPmn=0: positive phase • CKPmn=1: inverted phase					
Data Direction	MSB preferred or LSB preferred					

Note: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

Remark: m: unit number(m=0, 1, 2)n: channel number(n=0, 1)mn=00~01, 10~11, 20~21

## (1) Register setting

Figure16-23: 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)  
Example of register setting content when the master transmits



### Remark:

1. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)mn=00~01, 10~11, 20~21
2. ■: Setting disabled (set initial value). 0/1: Set "0" or "1" according to the user's purpose.

## (2) Procedure

Figure16-24: The initial setup steps of the master transmission

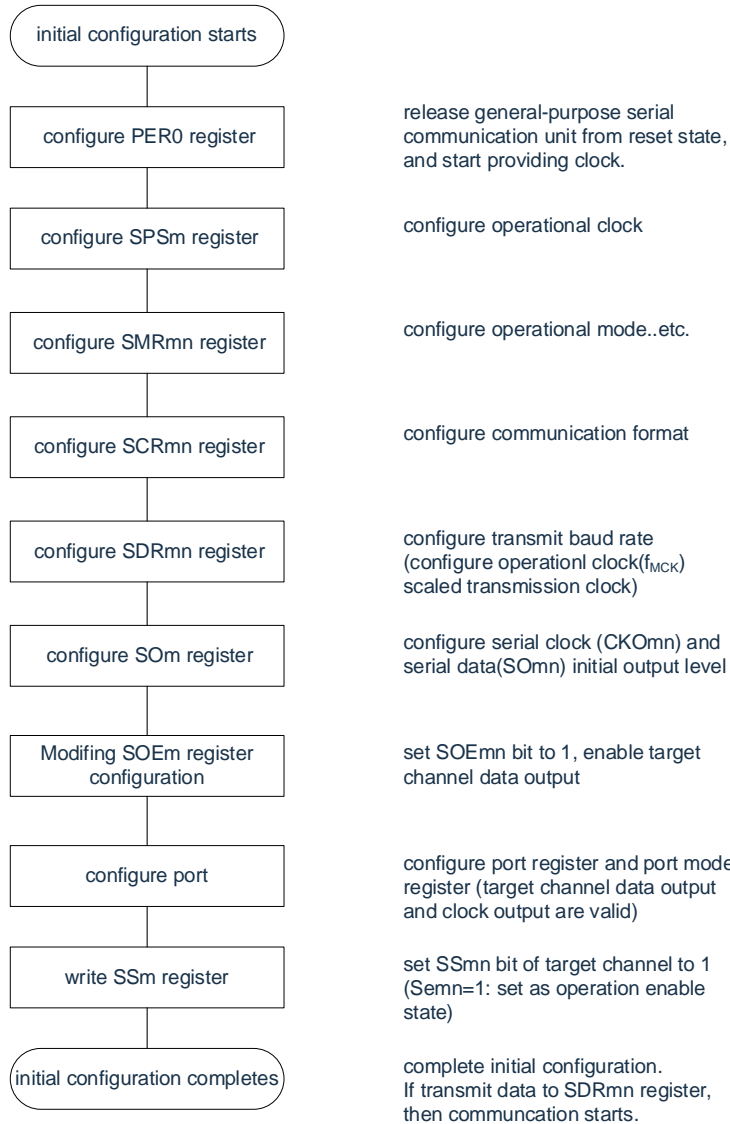


Figure 16-25: Setting steps of master transmission stop

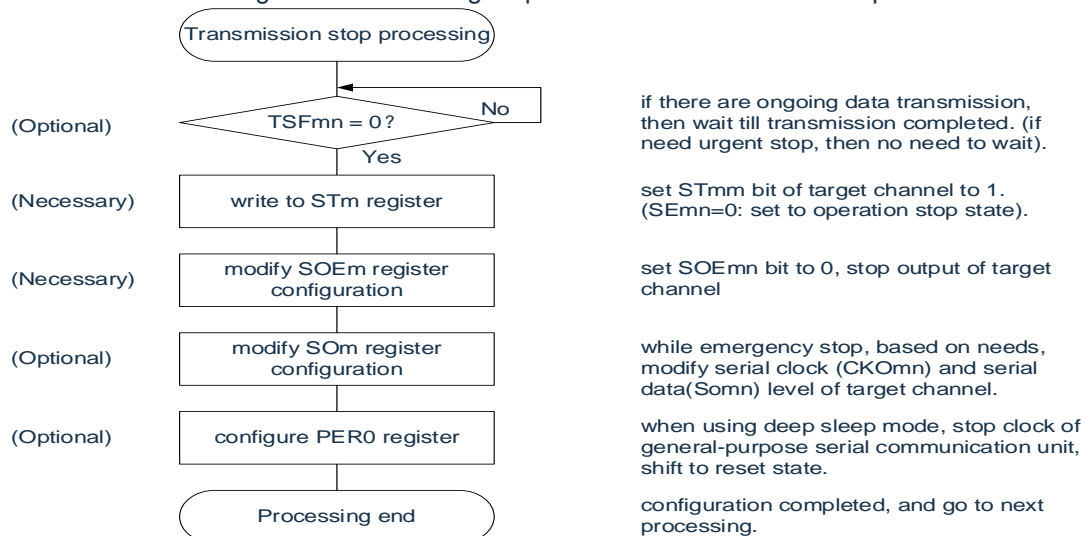
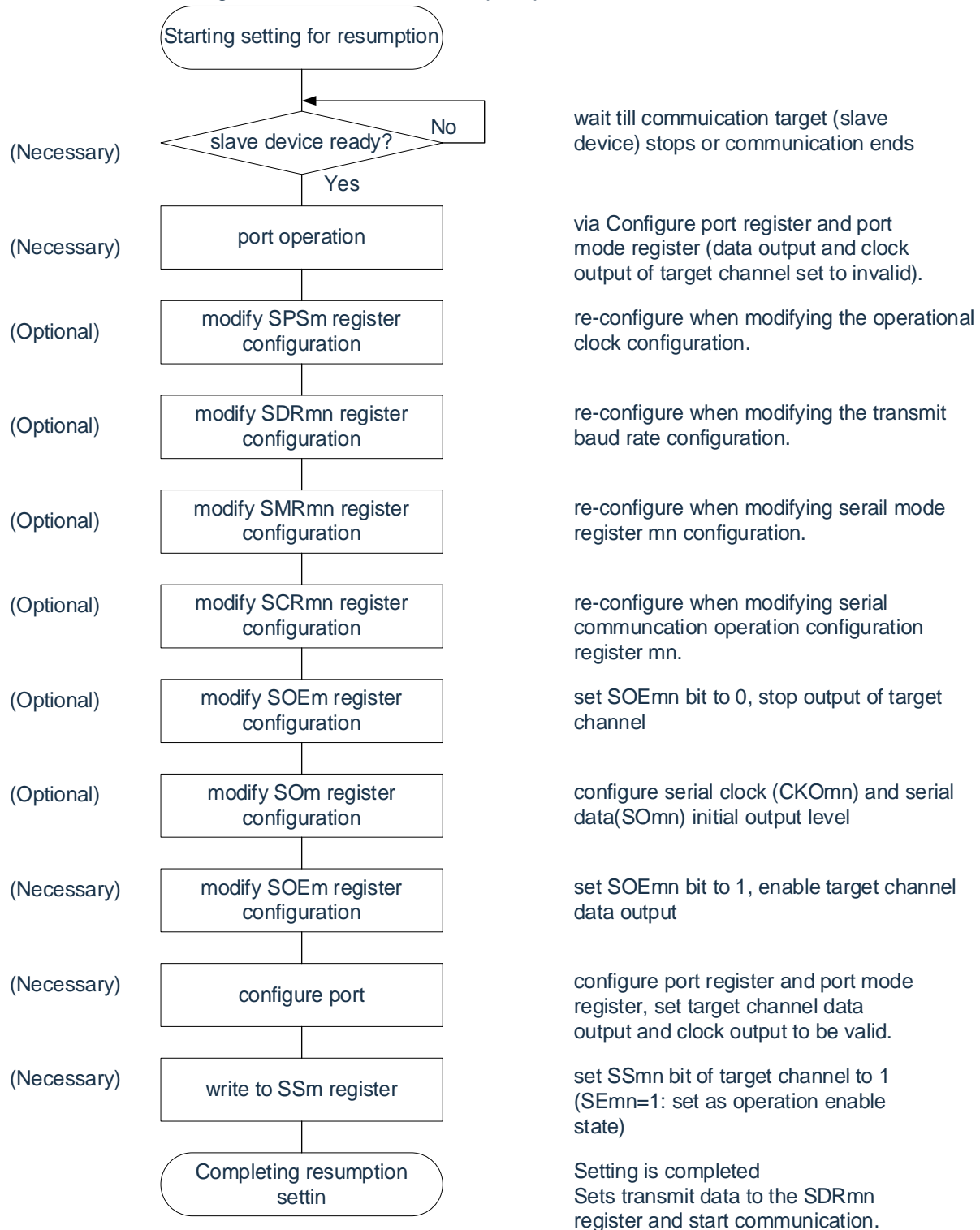


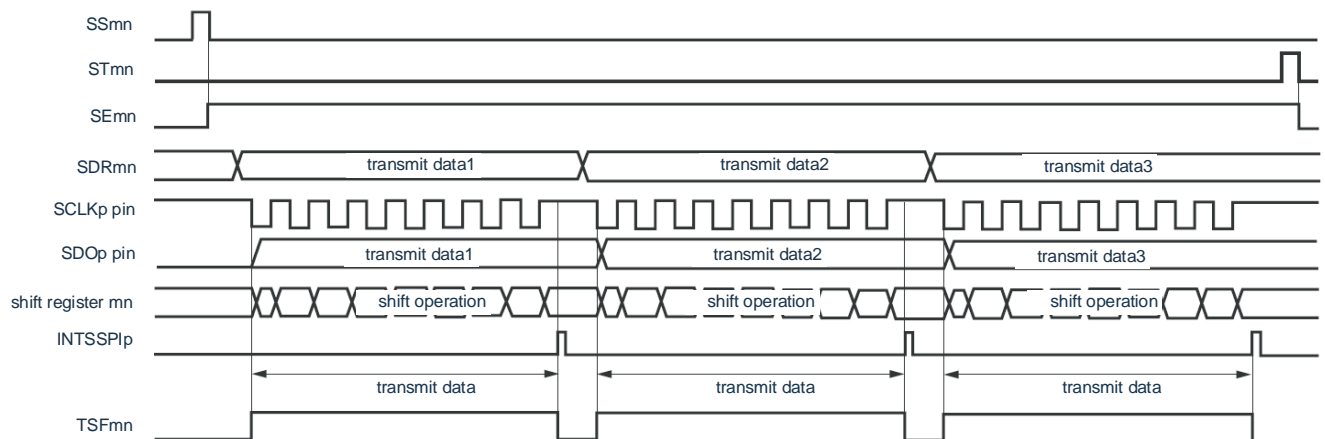
Figure 16-26: Restart the setup step of the master transmission



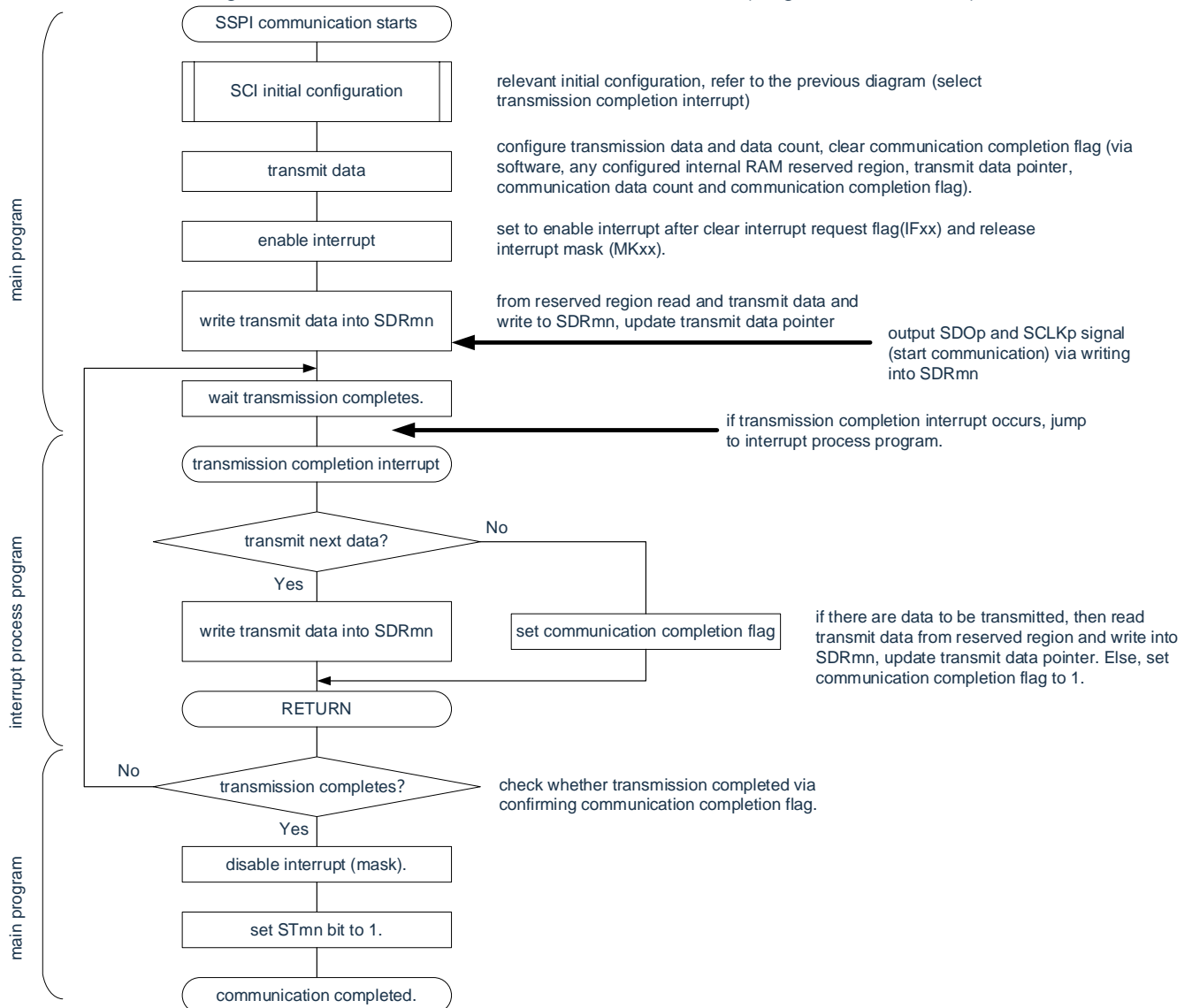
Remark: If you override PER0 in the abort settings to stop providing the clock, you must make the initial settings instead of restarting them when the communication object (slave) stops or when the communication ends.

### (3) Process flow (single send mode)

Figure 16-27: Timing diagram of the master transmission (single send mode) (type 1: DAPmn= 0, CKPmn = 0)



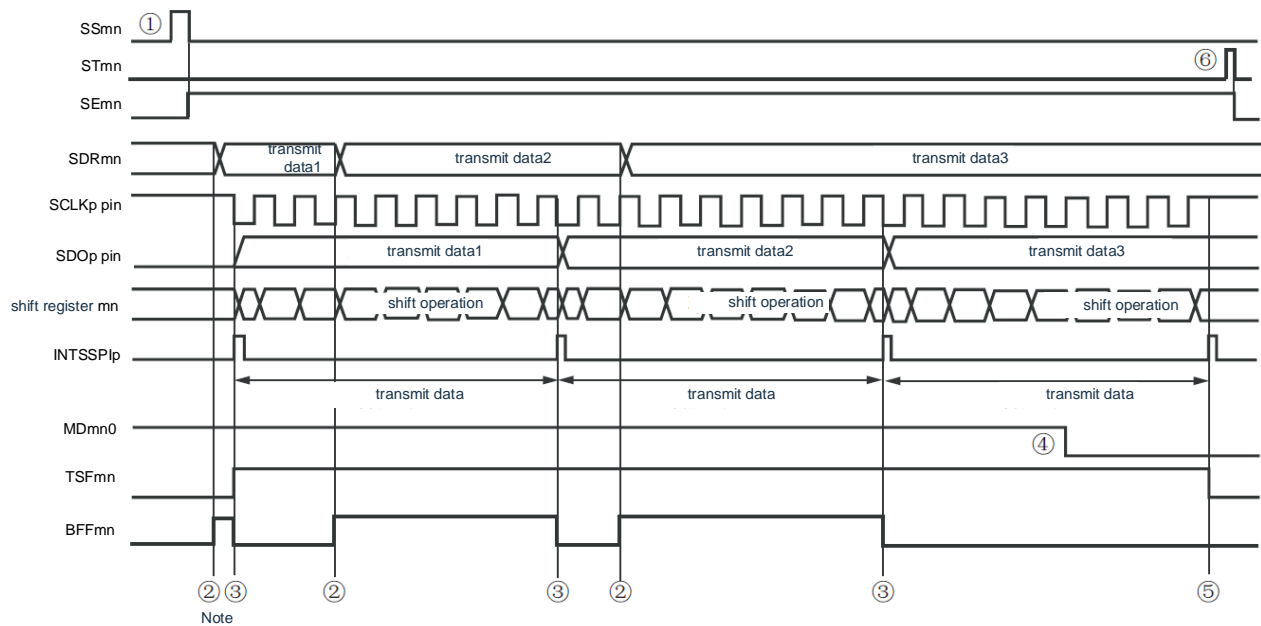
Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

**Figure 16-28: Flowchart of the master transmission (single transmit mode)**


#### (4) Process flow (continuous transmit mode)

Figure 16-29: Timing diagram of the master transmission (continuous transmit mode)

(type 1: DAPmn= 0, CKPmn = 0)

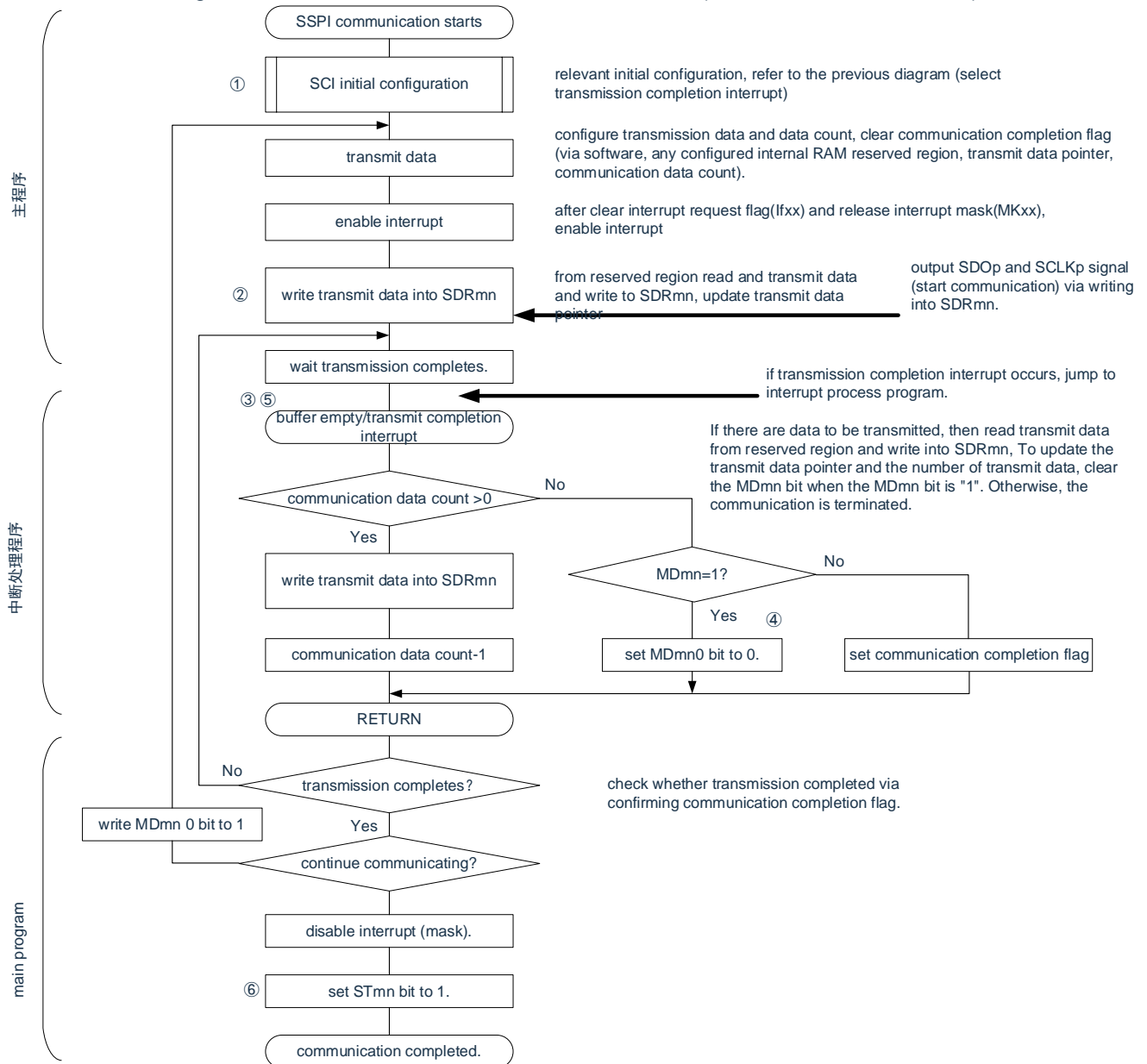


Note: If the BFFmn bit of the serial status register mn(SSRmn) is "1" (when valid data is saved in the serial data register mn(SDRmn)) is given When the SDRmn register writes the transmit data, it rewrites the send data.

Notice: MDmn0 bit of the serial mode register mn (SMRmn) can be rewritten even in operation. However, in order to catch the end-of-transmission interruption of the last sent data, it must be rewritten before the last bit of transmission begins.

Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

Figure 16-30: Flowchart of the master transmission (continuous transmit mode)



Remark: ①~⑥ in the figure corresponds to (1)~(6) in the "Figure 16-29: Timing diagram of the master transmission (continuous transmit mode)".



## 16.5.2 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 0 of SCI1	channel 1 of SCI1	channel 0 of SCI2	channel 1 of SCI2
Pin used	SCLK00 SDO00	SCLK01 SDO01	SCLK10 SDO10	SCLK11 SDO11	SCLK20 SDO20	SCLK21 SDO21
Interrupt	INTCSI00	INTCSI01	INTCSI10	INTCSI11	INTCSI20	INTCSI21
	Interrupt at that end of the transfer may be selecte (single transfer mode) or buffer empty interrupt (continuous transfer mode).					
Error detection flag	Only the overflow error detection flag (OVFmn).					
Transmitted data length	7~16 bits					
Transfer Rate <sup>Note</sup>	Max. $F_{CLK}/2$ [Hz]  Min. $F_{CLK}/(2 \times 2^{11} \times 128)$ [Hz] $F_{CLK}$ : system clock frequency					
Data phase	It can be selected by the DAPmn bit of the SCRmn register. • DAPmn=0: Start the data output when the serial clock starts running. • DAPmn=1: The data output is started half a clock before the serial clock starts running.					
Clock phase	It can be selected by the CKPmn bit of the SCRmn register. • CKPmn=0: positive phase • CKPmn=1: inverted phase					
Data orientation	MSB preferred or LSB preferred					

Note: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

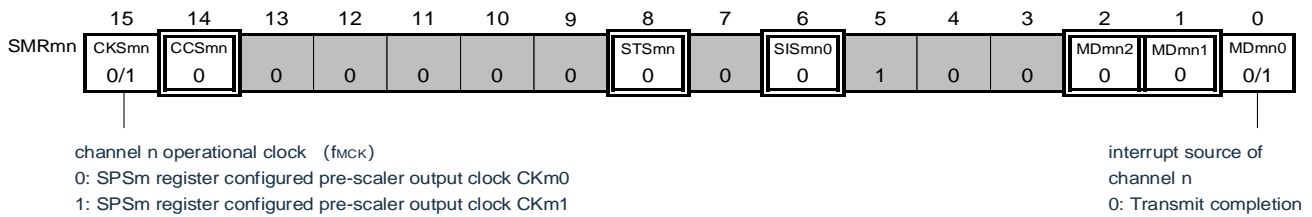
Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

## (1) Register setting

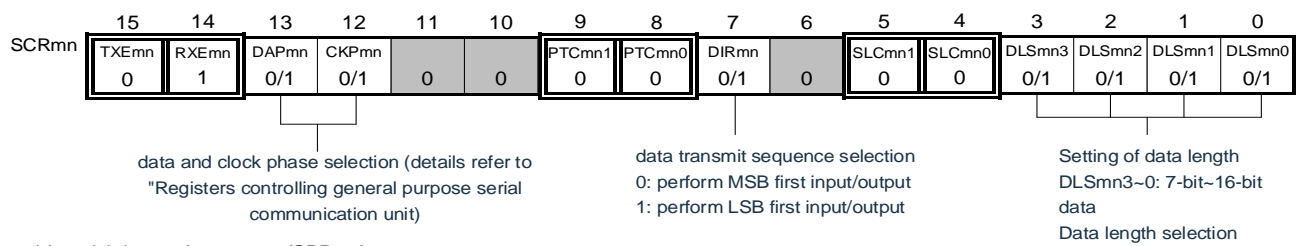
Figure 16-31: 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)

Example of register setting content when the master receives

(a) serial mode register mn (SMRmn)

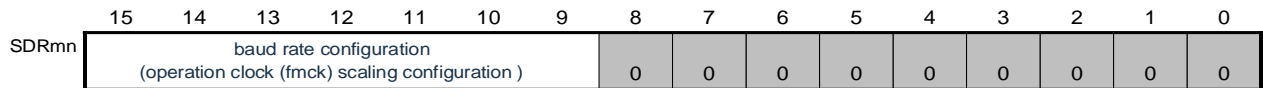


(b) serial communication operation configuration registermn mn (SCRmn)

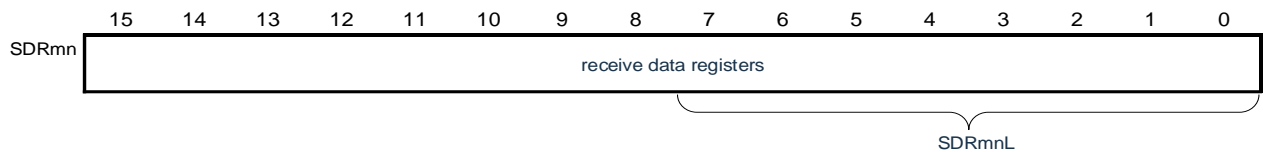


(c) serial data registermn mn (SDRmn)

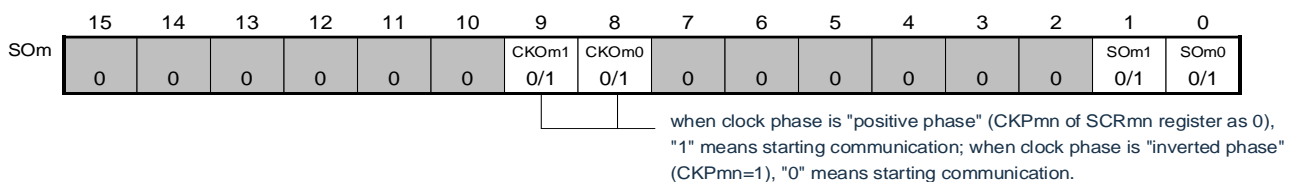
(1) When operation stops (SEmn=0)



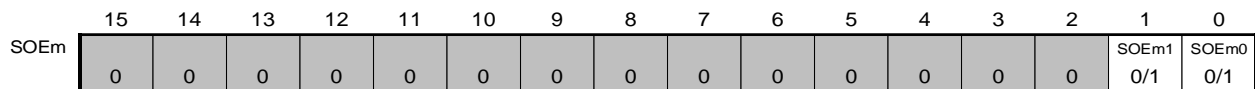
(2) During operation (SEmn=1) (lower 8 bits: SDRmnL)



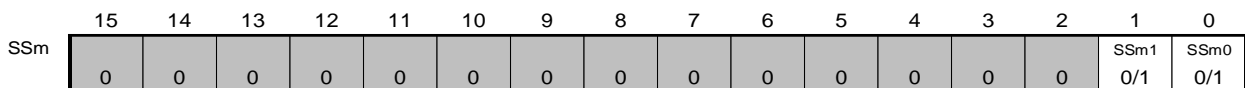
(d) serial output register m(SOm) .....Only configure bit of target channel



(e) serial output enable register m (SOEm)....not used in this mode.



(f) serial channel start registerm (SSm)....only set bit of target channel to 1.



Remark:

- m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)
- : Fixed setting in SSPI master receive mode.■: setting disabled(initial value).  
x: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).  
0/1: Set "0" or "1" according to the user's purpose.

## (2) Procedure

Figure 16-32: Initial setup steps for master reception

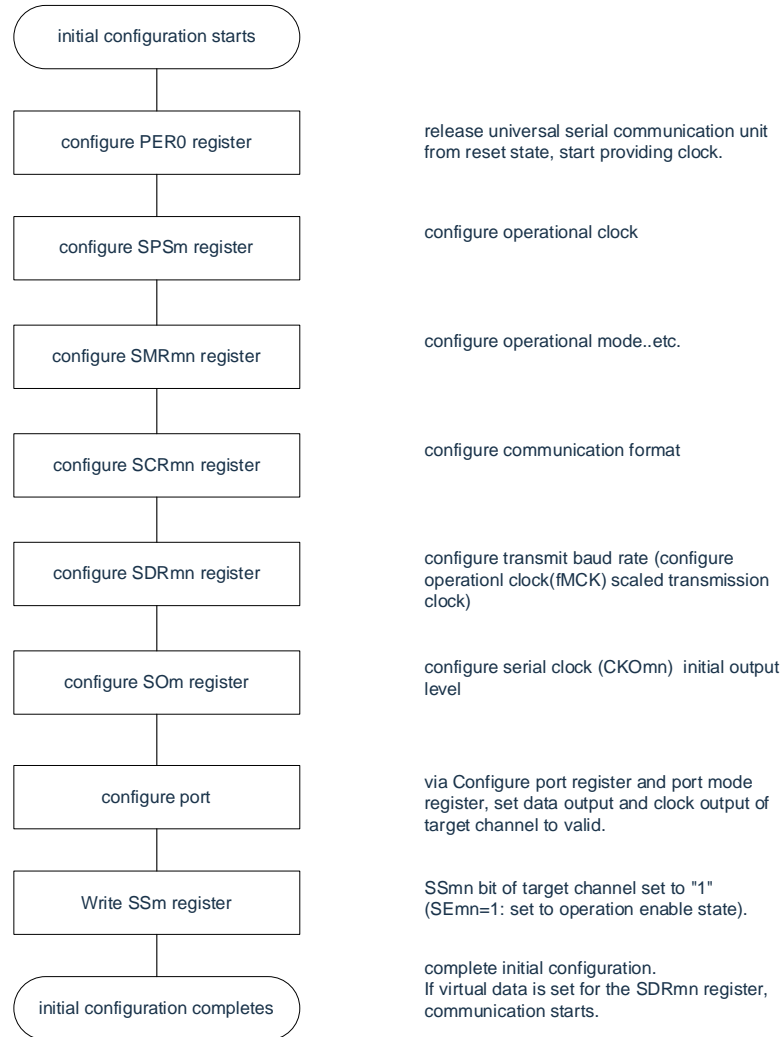


Figure 16-33: Stop steps for master reception

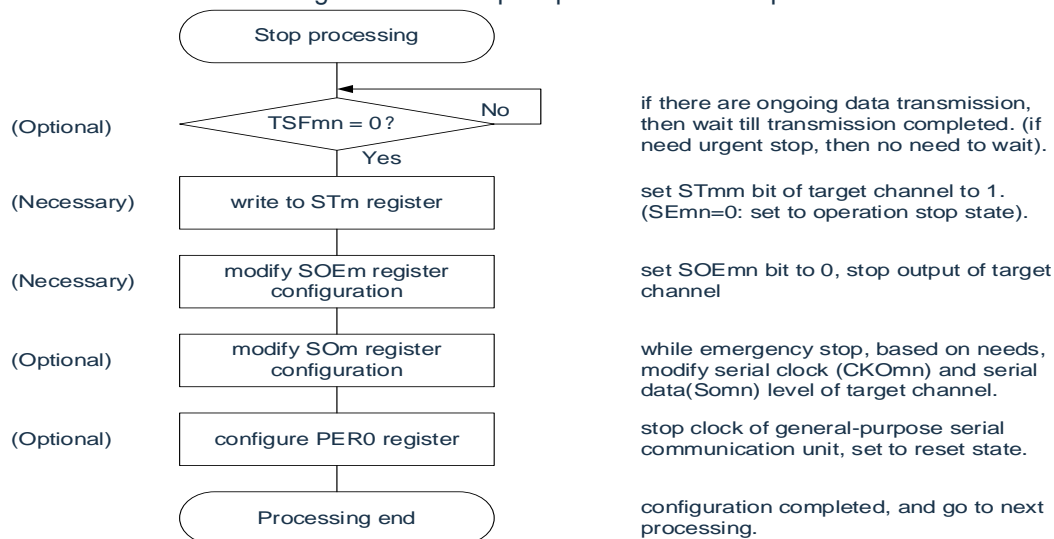
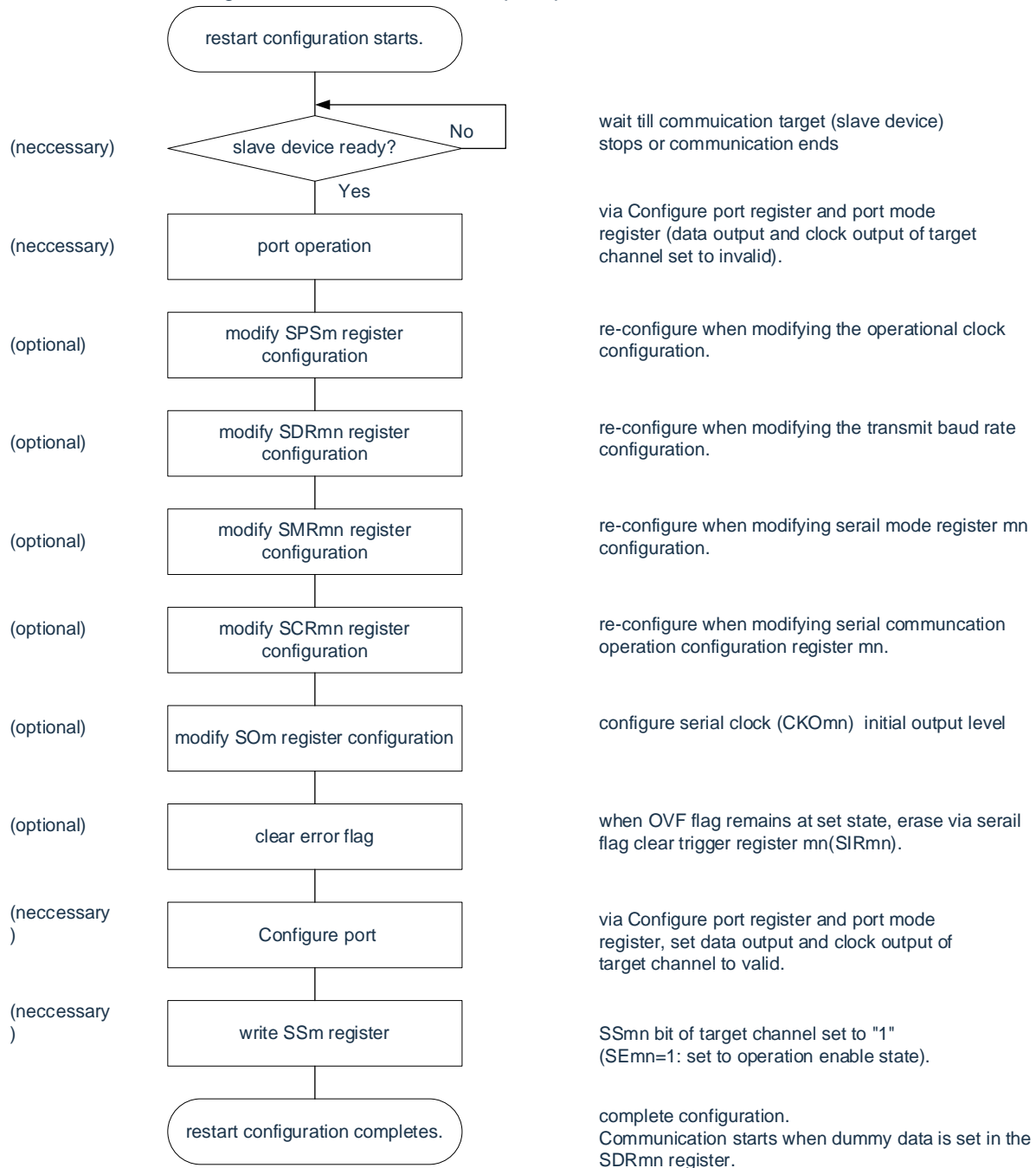


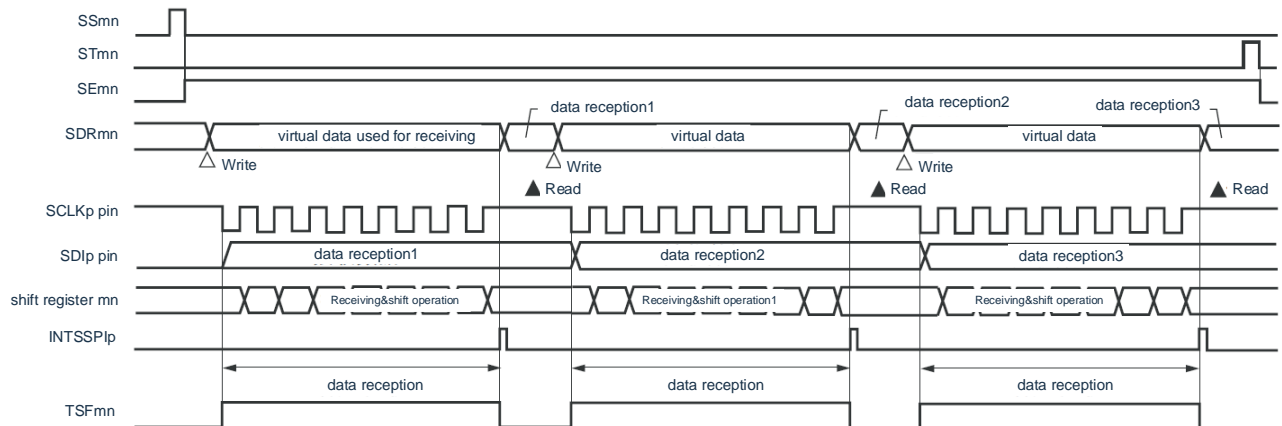
Figure16-34: Restart the setup step of the master transmission



Remark: If you override PER0 in the abort settings to stop providing the clock, you must make the initial settings instead of restarting them when the communication object (slave) stops or when the communication ends.

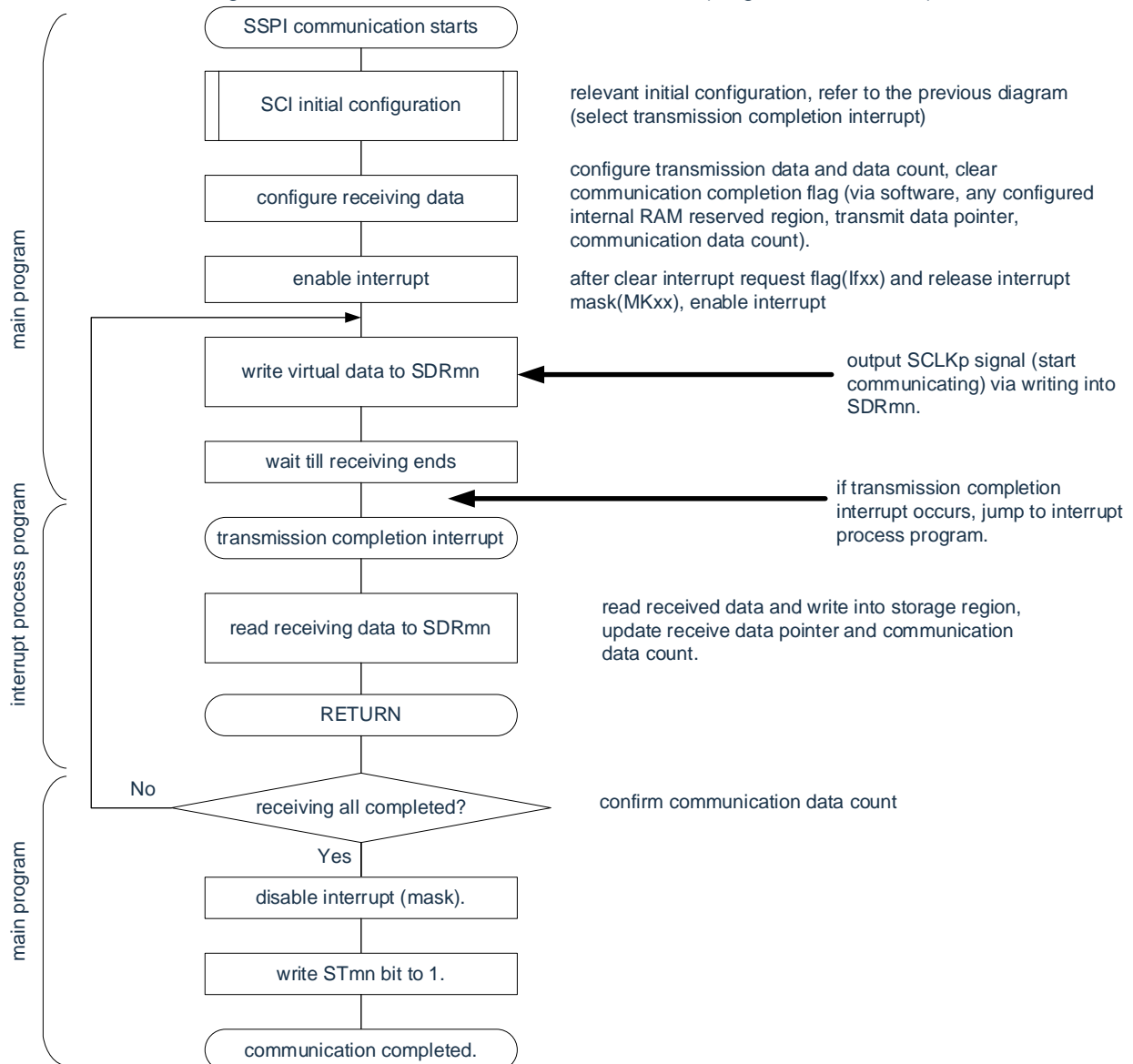
### (3) Process flow (single receive mode)

Figure 16-35: Timing diagram of the master receive (single receive mode) (type 1: DAPmn = 0, CKPmn = 0)



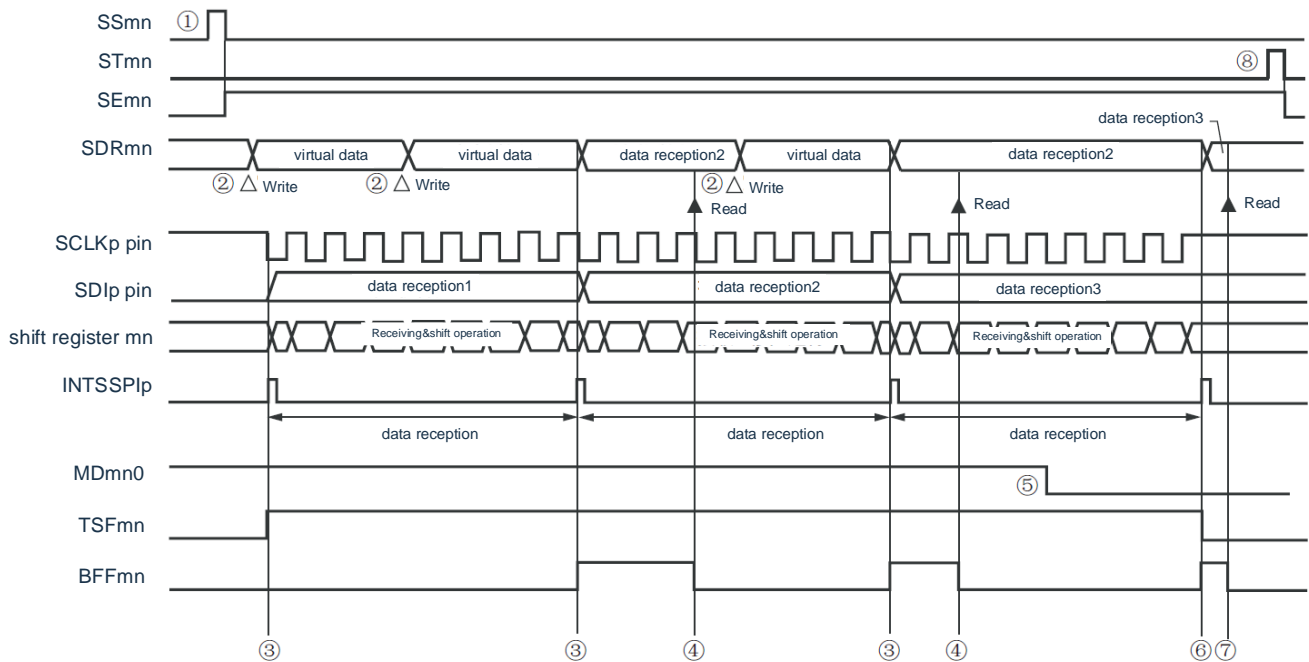
Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

Figure 16-36: Flowchart of the master receive (single receive mode)



#### (4) Process flow (continuous receive mode)

Figure 16-37: master receive (continuous receive mode) (type 1: DAPmn = 0, CKPmn = 0)

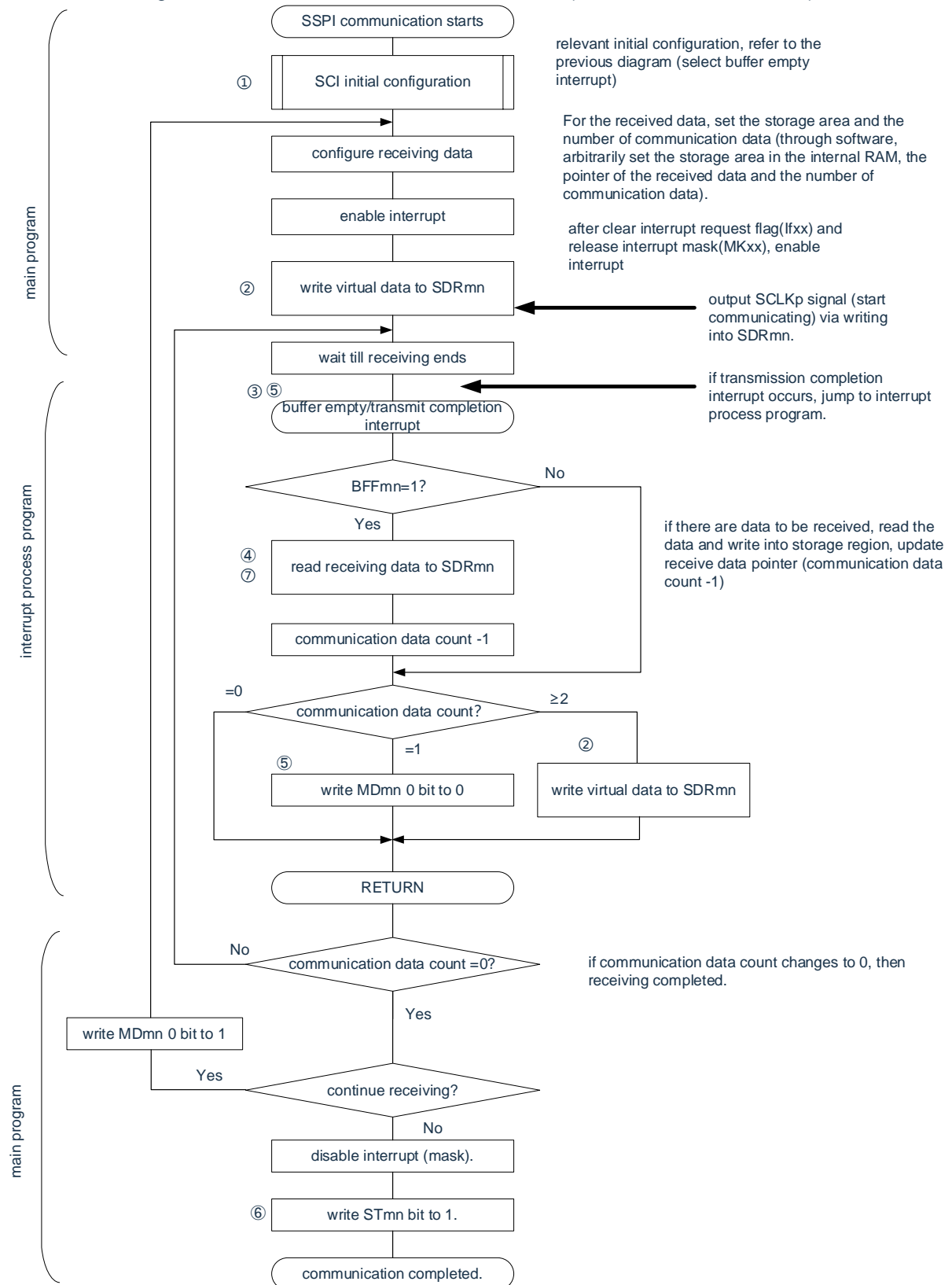


Notice: MDmn0 bit can be rewritten even during operation. However, in order to catch up with the end-of-transmission interruption of the last received data, it must be rewritten before the last bit can be received.

Remark:

1. 1) ~ (8) in the figure corresponds to (1) ~ (8) in “Figure16-38: Flowchart of the master receiver (continuous receive mode)”.
2. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)

Figure16-38: Flowchart of the master receiver (continuous receive mode)



Remark: 1) ~ (8) in the figure corresponds to (1) ~ (8) in the “Figure 16-37 master receive (continuous receive mode) (type 1: DAPmn= 0, CKPmn = 0)”.



### 16.5.3 Master transmission and reception

The transmission and reception of the master refers to the operation of this product output transmission clock and data transmission and reception with other devices.

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channels	Channel 0 of SCI0	Channel 1 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1	Channel 0 of SCI2	Channel 1 of SCI2
Pins used	SCLK00 SDI00 SDO00	SCLK01 SDI01 SDO01	SCLK10 SDI10 SDO10	SCLK11 SDI11 SDO11	SCLK20 SDI20 SDO20	SCLK21 SDI21 SDO21
Interrupt	INTCSI00	INTCSI01	INTCSI10	INTCSI11	INTCSI20	INTCSI21
	You can choose between an end-of-transmit interrupt (single transmit mode) or a buffer empty interrupt (continuous transfer mode).					
Error detection flag	There is only the overflow error detection flag (OVFmn)					
Transmitted data length	7-16 bits					
Transfer Rate <sup>Note</sup>	Max.F <sub>CLK</sub> /2[Hz]  Min.F <sub>CLK</sub> /(2x2 <sup>11</sup> x128) [Hz] F <sub>CLK</sub> : System clock frequency					
Data phase	It can be selected by the DAPmn bit of the SCRmn register. • DAPmn=0: Starts data output when the serial clock starts running. • DAPmn=1: Starts the data output half a clock before the serial clock starts running.					
Clock phase	It can be selected by the CKPmn bit of the SCRm n register. • CKPmn=0: positive phase • CKPmn=1: Inverted phase					
Data orientation	MSB preferred or LSB preferred					

**Note:** It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

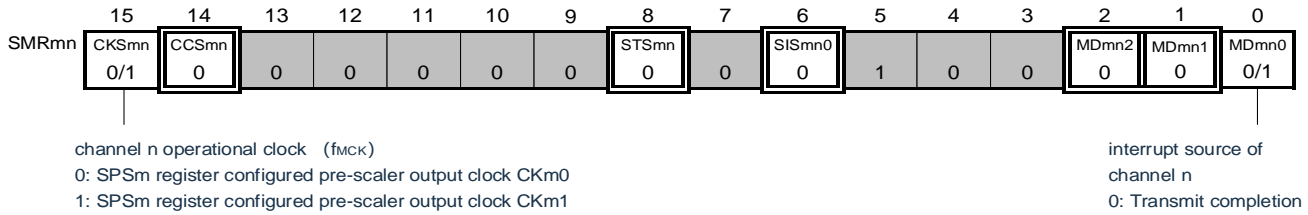
**Remark:** m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

## (1) Register setting

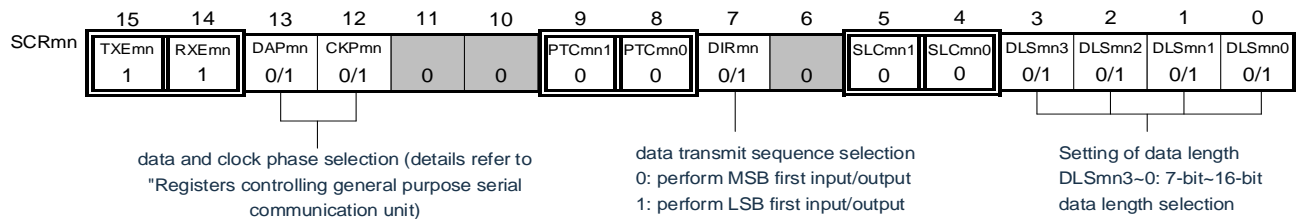
Figure16-39: 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)

Example of register settings for master sending and receiving

(a) serial mode register mn (SMRmn)

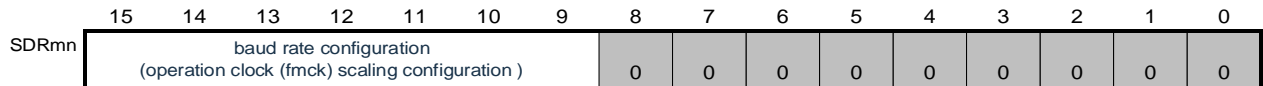


(b) serial communication operation configuration registermn mn (SCRmn)

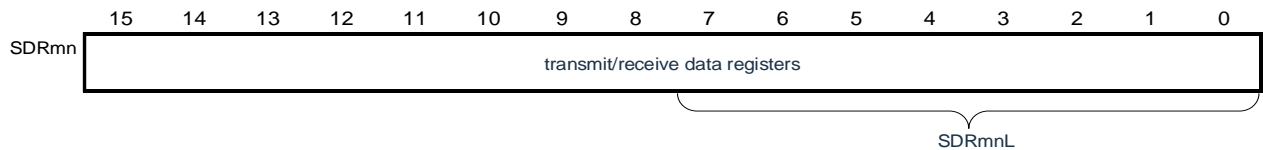


(c) serial data registermn mn (SDRmn)

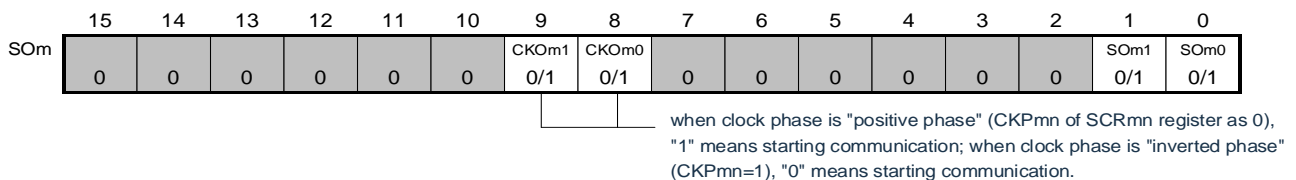
(1) When operation stops (SEmn=0)



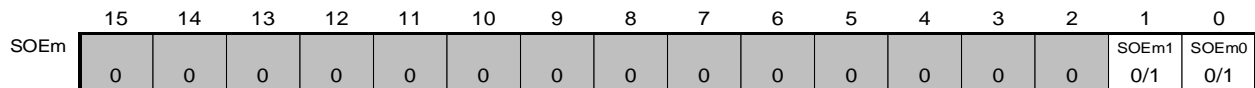
(2) During operation (SEmn=1) (lower 8 bits: SDRmnL)



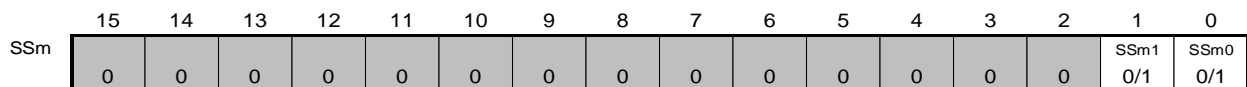
(d) serial output register m(SOm) .....Only configure bit of target channel



(e) serial output enable register m (SOEm).....Only set bit of target channel to 1.



(f) serial channel start registerm (SSm).....only set bit of target channel to 1.



Remark:

- m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)
- : Fixed setting in SSPI master send and receive mode.■: setting disabled(initial value).  
0/1: Set "0" or "1" according to the user's purpose.

## (2) Procedure

Figure16-40: initial setup steps for master sending and receiving

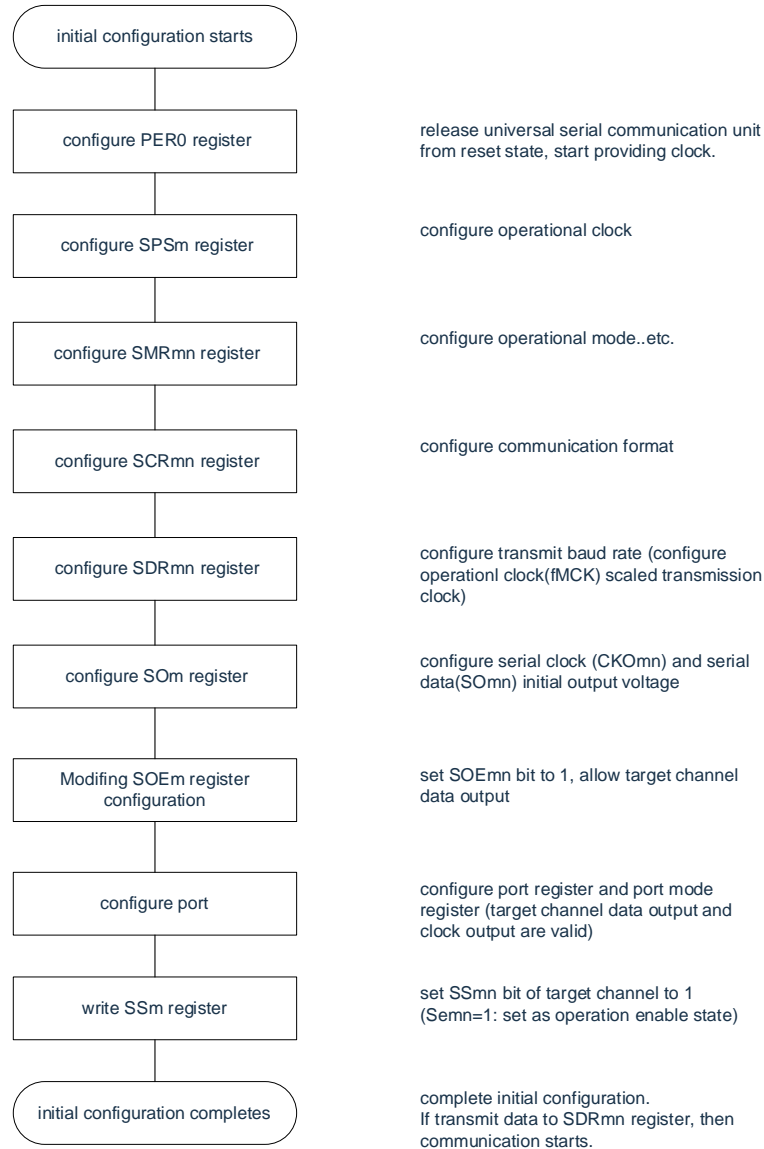


Figure16-41: Stop steps of the master transmit and receive

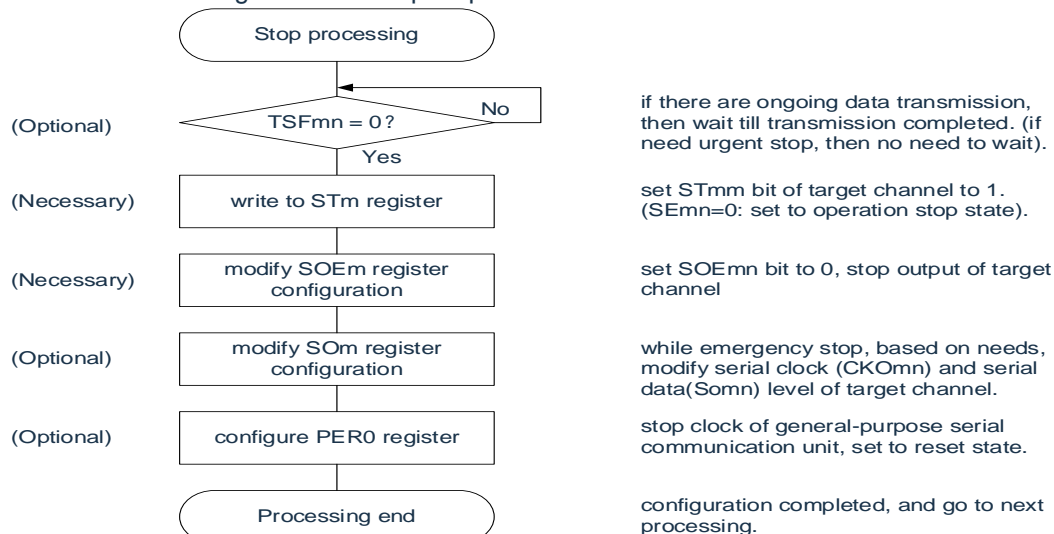
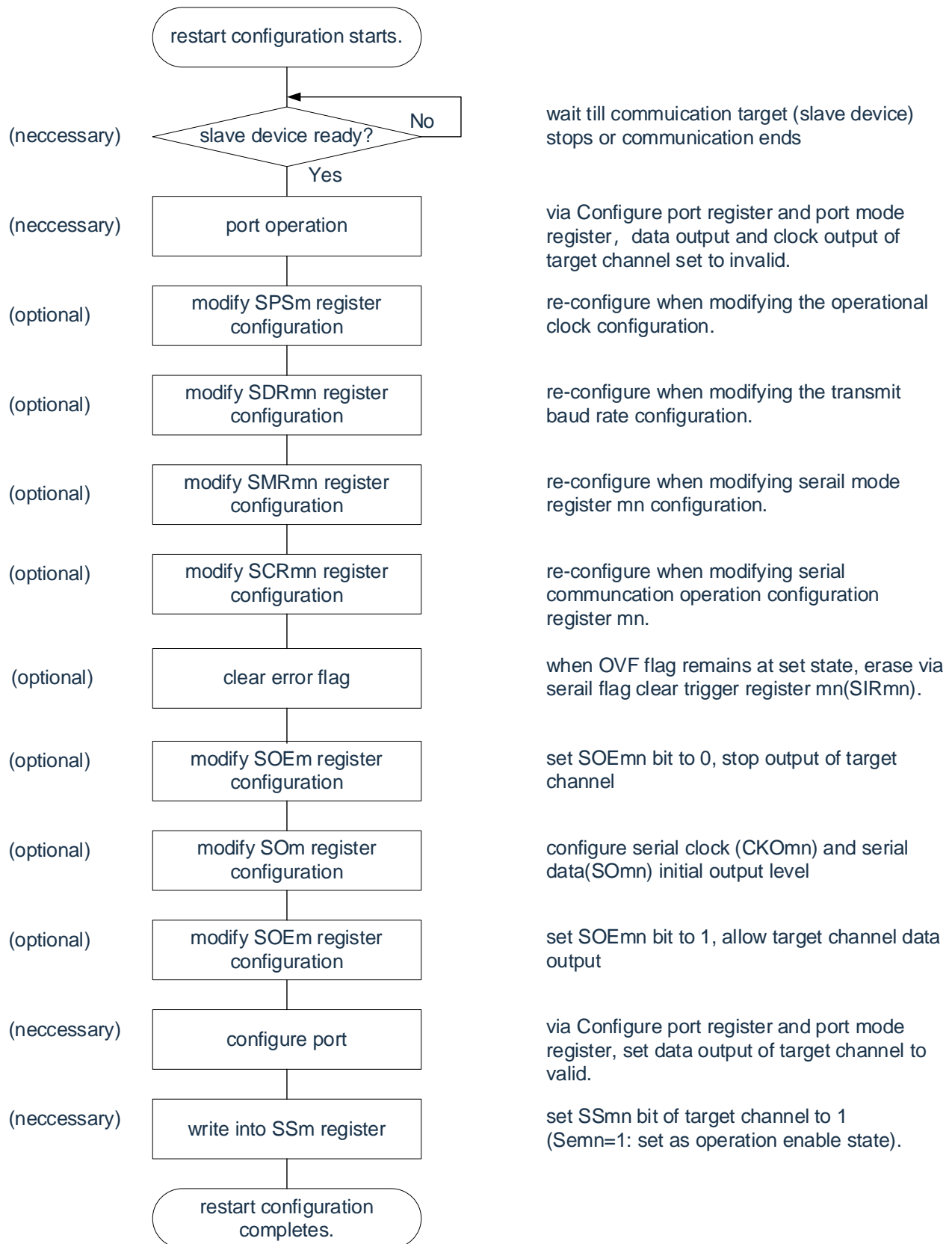
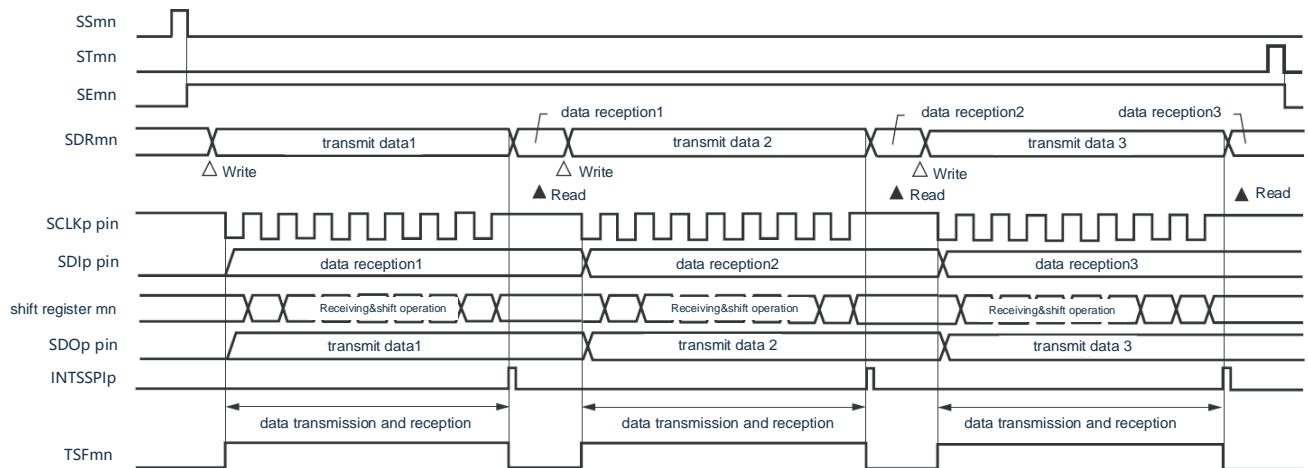


Figure16-42: Restart the setup steps of the master send and receive



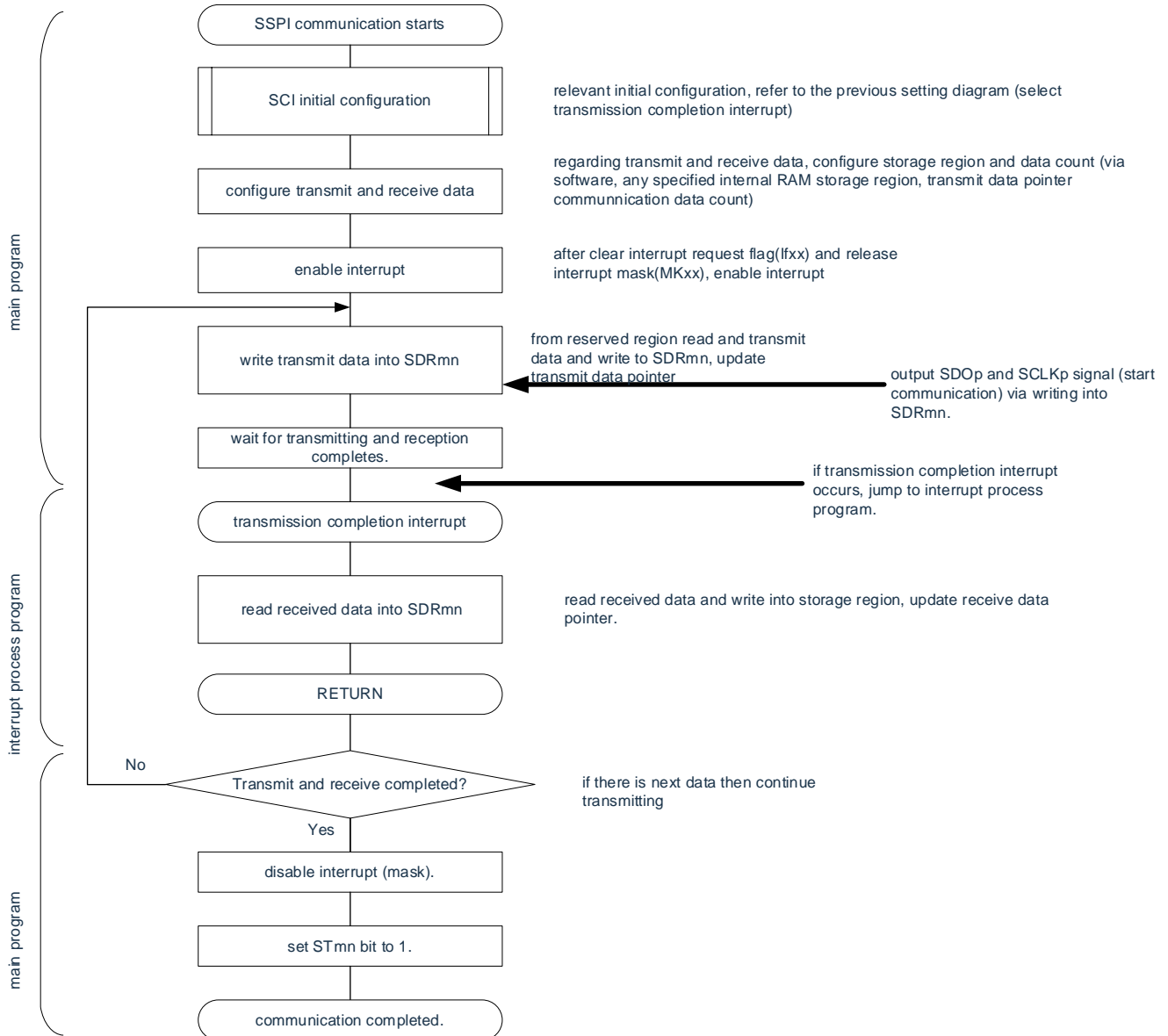
(3) Process flow (single send and receive mode)

Figure 16-43: Timing diagram of the master transmit and receive (single transmit and receive mode) (type 1: DAPmn=0, CKPmn=0)



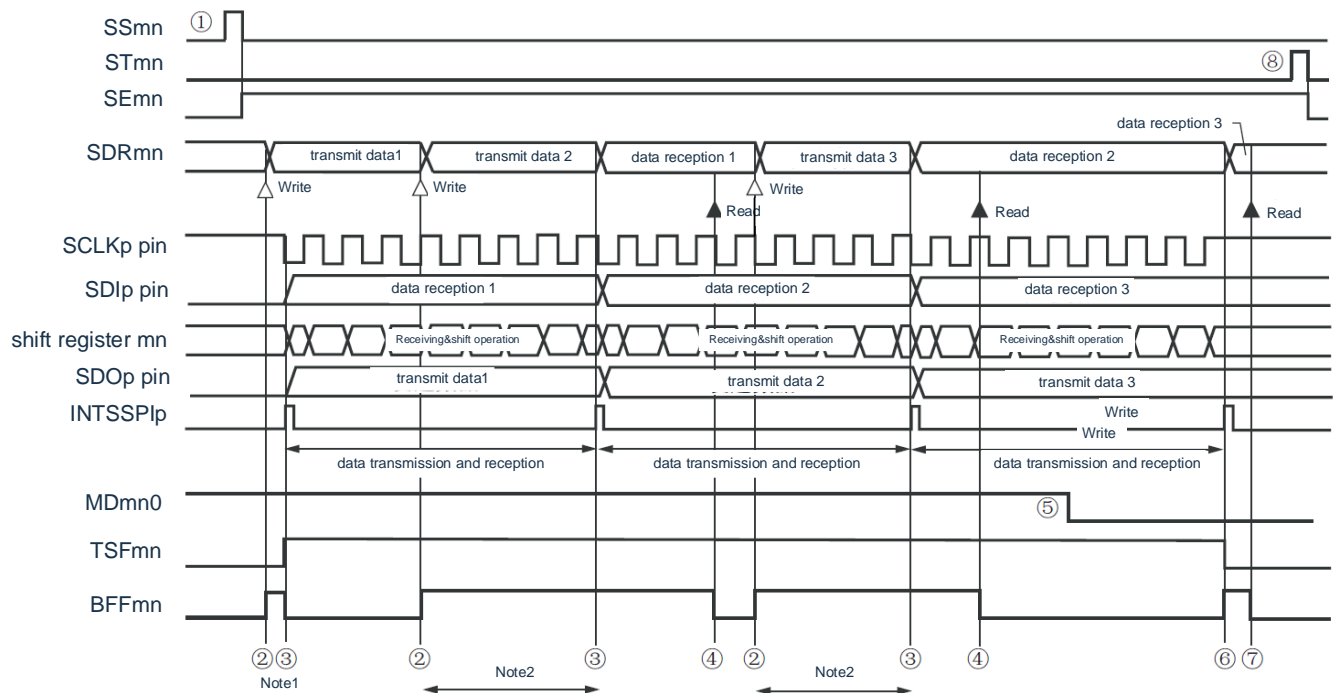
Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

Figure16-44: Flowchart of the master transmit and receive (single send and receive mode)



#### (4) Process flow (continuous send and receive mode)

Figure16-45: Timing diagram of the master transmit and receive (continuous transmit and receive mode) (type 1: DAPmn=0, CKPmn=0)



Note 1: If the BFFmn bit of serial status register mn (SSRmn) is "1" during the period (valid data is saved in serial data register mn (SDRmn) (when writing transmit data to the SDRmn register, rewrite the send data).

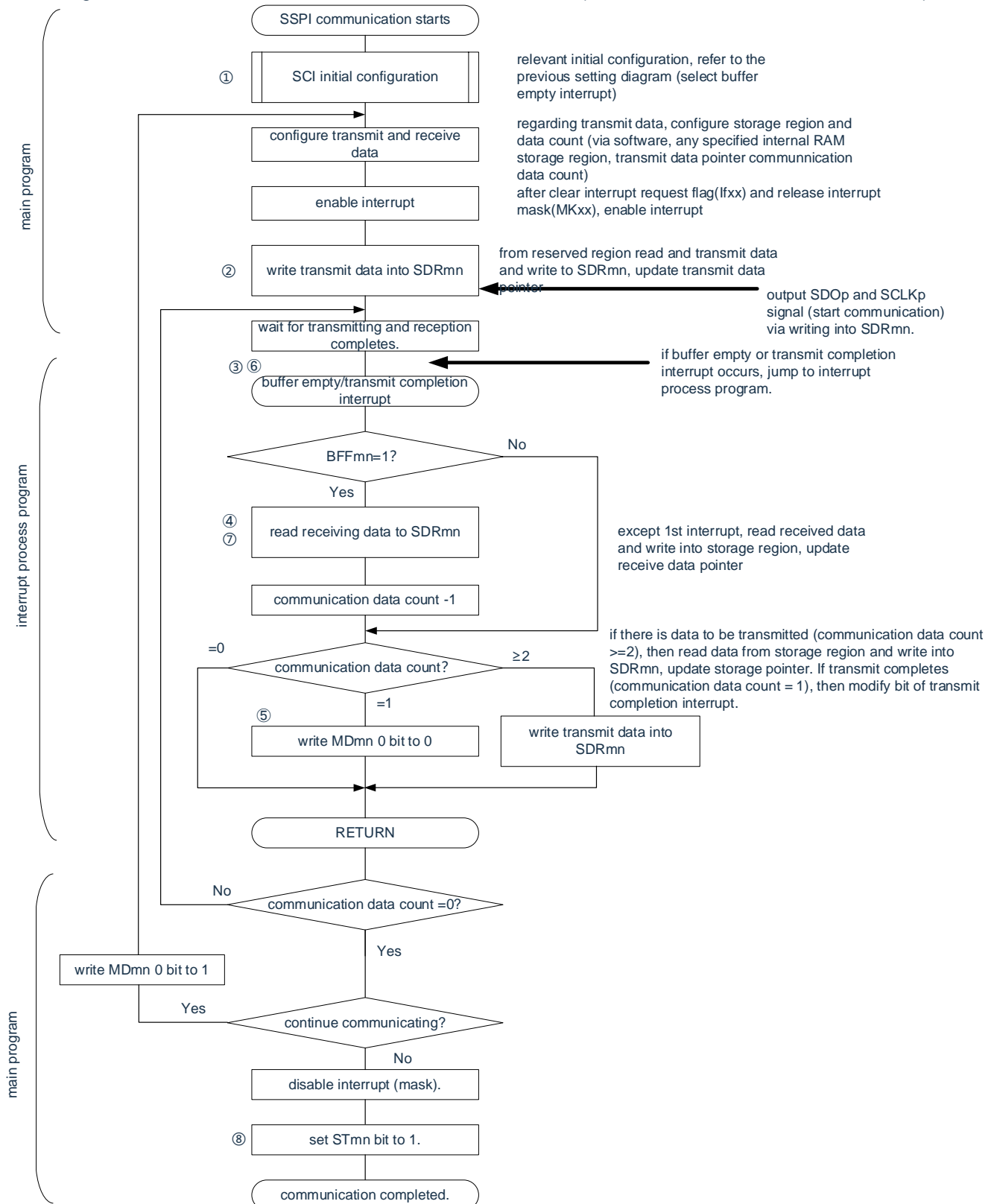
Note 2: If the SDRmn register is read during this period, the data can be read and sent. At this point, the transfer run is not affected.

Notice: MDmn0 bit of the serial mode register mn (SMRmn) can be rewritten even in operation. However, in order to catch the end-of-transmission interruption of the last sent data, it must be rewritten before the last bit of transmission begins.

Remark:

- (1) ~ (8) in the figure corresponds to "Figure16-46 Flowchart of the master transmit and receive (continuous transmit and receive mode)" in ①~⑧.
- m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

Figure16-46: Flowchart of the master transmit and receive (continuous transmit and receive mode)



Remark: (1) ~ (8) in the figure corresponds to (1) ~ (8) in the “Figure16-45 Timing diagram of the master transmit and receive (continuous transmit and receive mode)”.



## 16.5.4 Slave transmission

Slave transmission refers to the operation of the CMS32H6157 microcontroller to send data to other devices in the state of transmitting clocks from other device inputs.

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channels	Channel 0 of SCI0	Channel 1 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1	Channel 0 of SCI2	Channel 1 of SCI2
The pins used	SCLK00 SDO00	SCLK01 SDO01	SCLK10 SDO10	SCLK11 SDO11	SCLK20 SDO20	SCLK21 SDO21
interrupt	INTCSI00	INTCSI01	INTCSI10	INTCSI11	INTCSI20	INTCSI21
	Selectable end-of-transmit interrupt (single-pass mode) or buffer-empty interrupt (continuous transfer mode).					
Error detection flags	There are only overflow error detection flags (OVFmn).					
transferred data length	7-16 bits					
Transfer rate <sup>Note</sup>	Max.F <sub>CLK</sub> /2[Hz]					
	Min.F <sub>CLK</sub> /(2x2 <sup>11</sup> x128) [Hz] F <sub>CLK</sub> : system clock frequency					
Data phase	It can be selected by the DAPmn bit of the SCRmn register. <ul style="list-style-type: none"> <li>DAPmn=0: Starts data output when the serial clock starts running.</li> <li>DAPmn=1: Starts the data output half a clock before the serial clock starts running.</li> </ul>					
Clock phase	It can be selected by the CKPmn bit of the SCRmn register. <ul style="list-style-type: none"> <li>CKPmn=0: positive phase</li> <li>CKPmn=1: inverted phase</li> </ul>					
Data direction	MSB preferred or LSB preferred					

Note 1: Because the external serial clocks input from SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, SCLK21 pins are sampled internally and then used, the maximum transfer rate is F<sub>MCK</sub>/6[Hz].

Note 2: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

Remark:

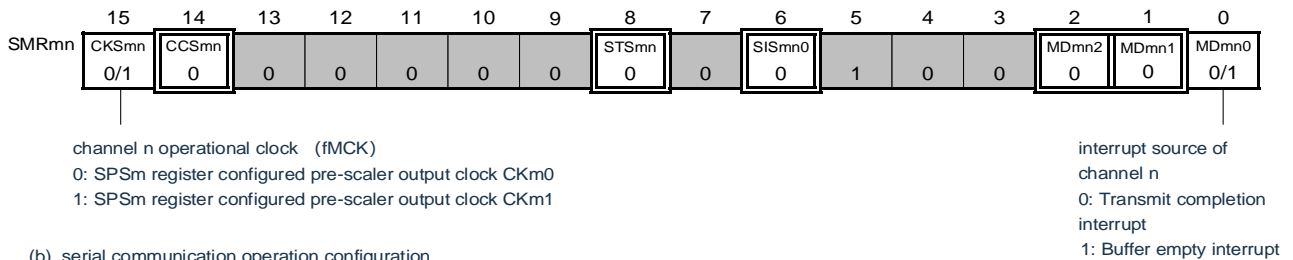
1. F<sub>MCK</sub>: The operating clock frequency of the object channel
2. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)

## (1) Register setting

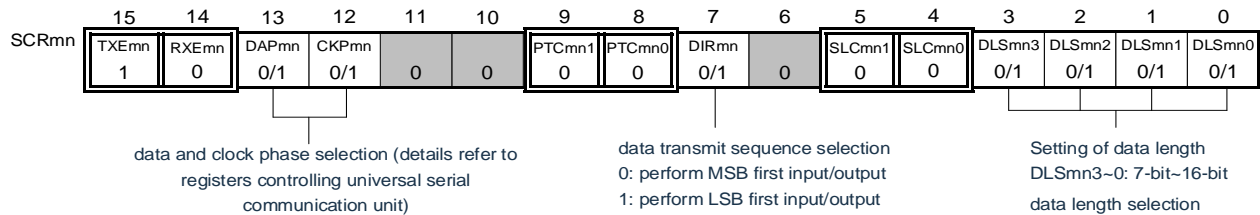
Figure16-47: 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)

Example of register settings at the time of slave transmission

(a) serial mode register mn (SMRmn)

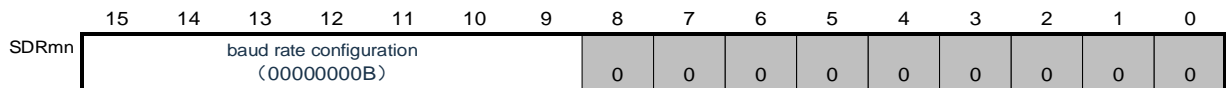


(b) serial communication operation configuration register mn (SCRmn)

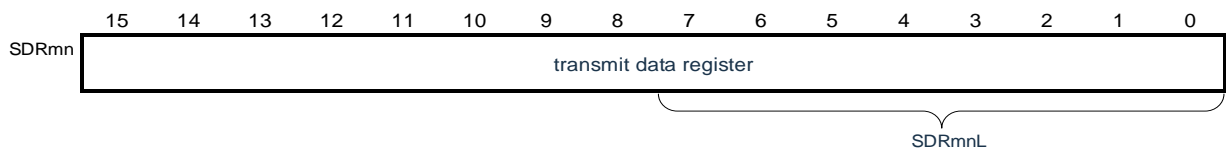


(c) serial data register mn (SDRmn)

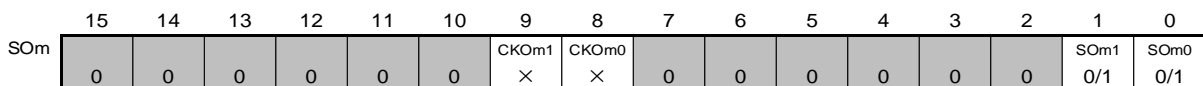
(1) When operation stops (SEmn=0)



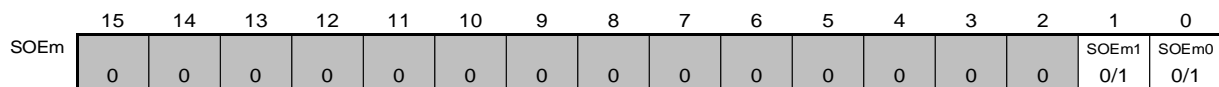
(2) During operation (SEmn=1) (lower 8 bits: SDRmnL)



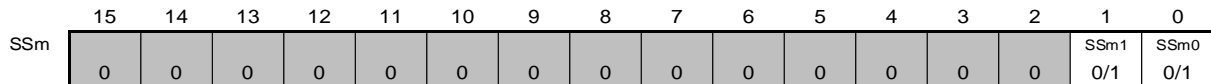
(d) serial output register m(SOm) .....Only configure bit of target channel



(e) serial output enable registerm (SOEm)....only set bit of target channel to 1.



(f) serial channel start register m (SSm) .... Only set bit of target channel to 1.



Remark:

- m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)
- : Fixed setting in SSPI slave send mode. ■: setting disabled(initial value).  
x: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).  
0/1: Set "0" or "1" according to the user's purpose.

## (2) Procedure

Figure 16-48: Initial setup steps for slave sending



Figure 16-49: Stop step of the slave send

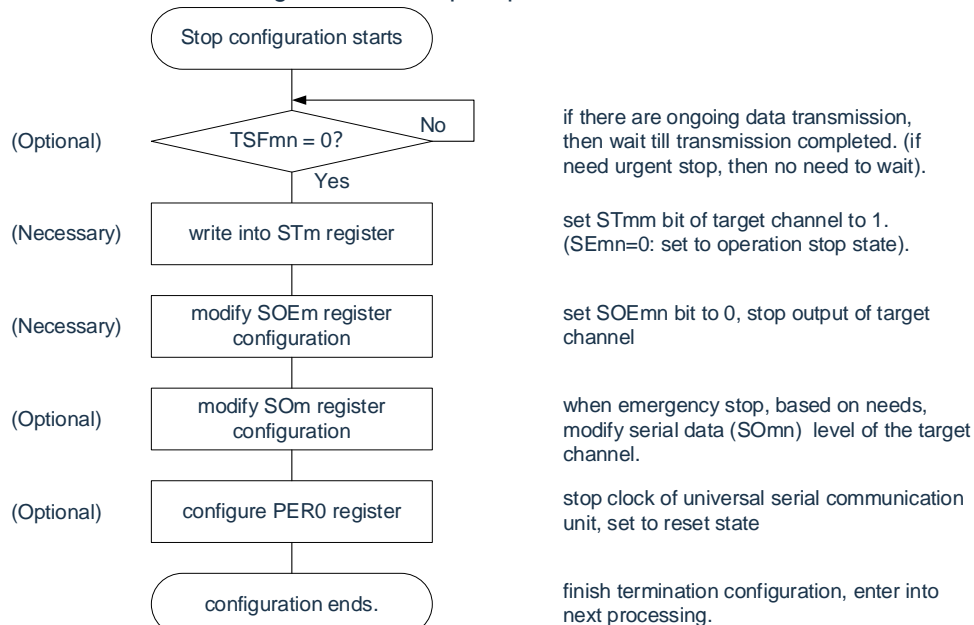
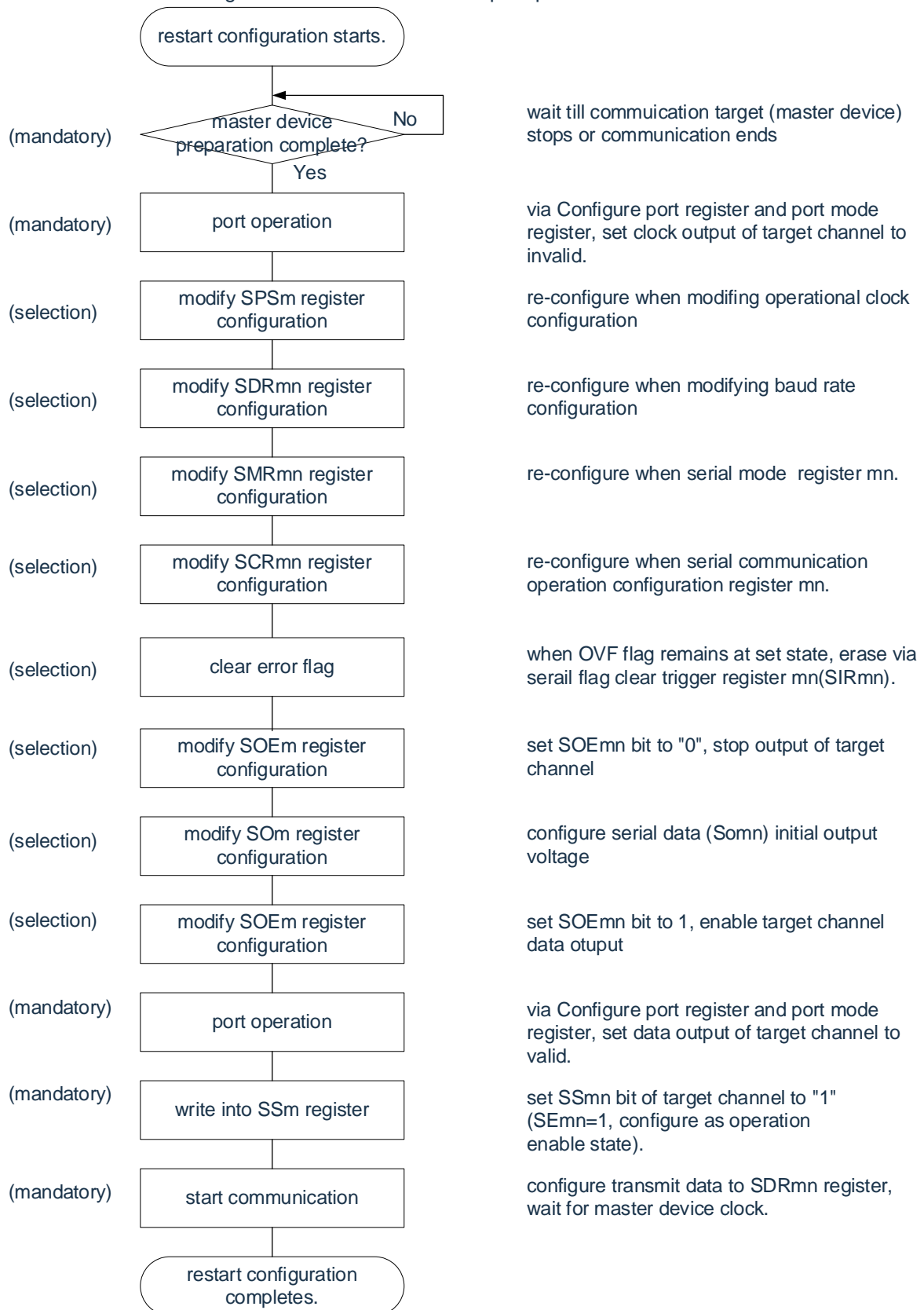


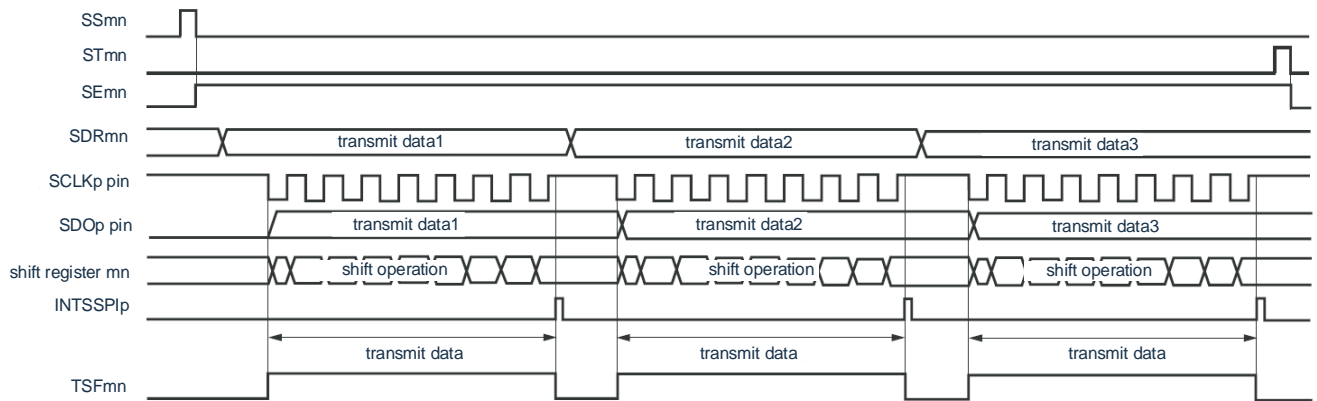
Figure16-50: Restarts the setup step of the slave send



Remark: If you override PER0 in the stop setting to stop providing the clock, you must make the initial setting instead of restarting the setting when the communication object (the master device) stops or the communication ends.

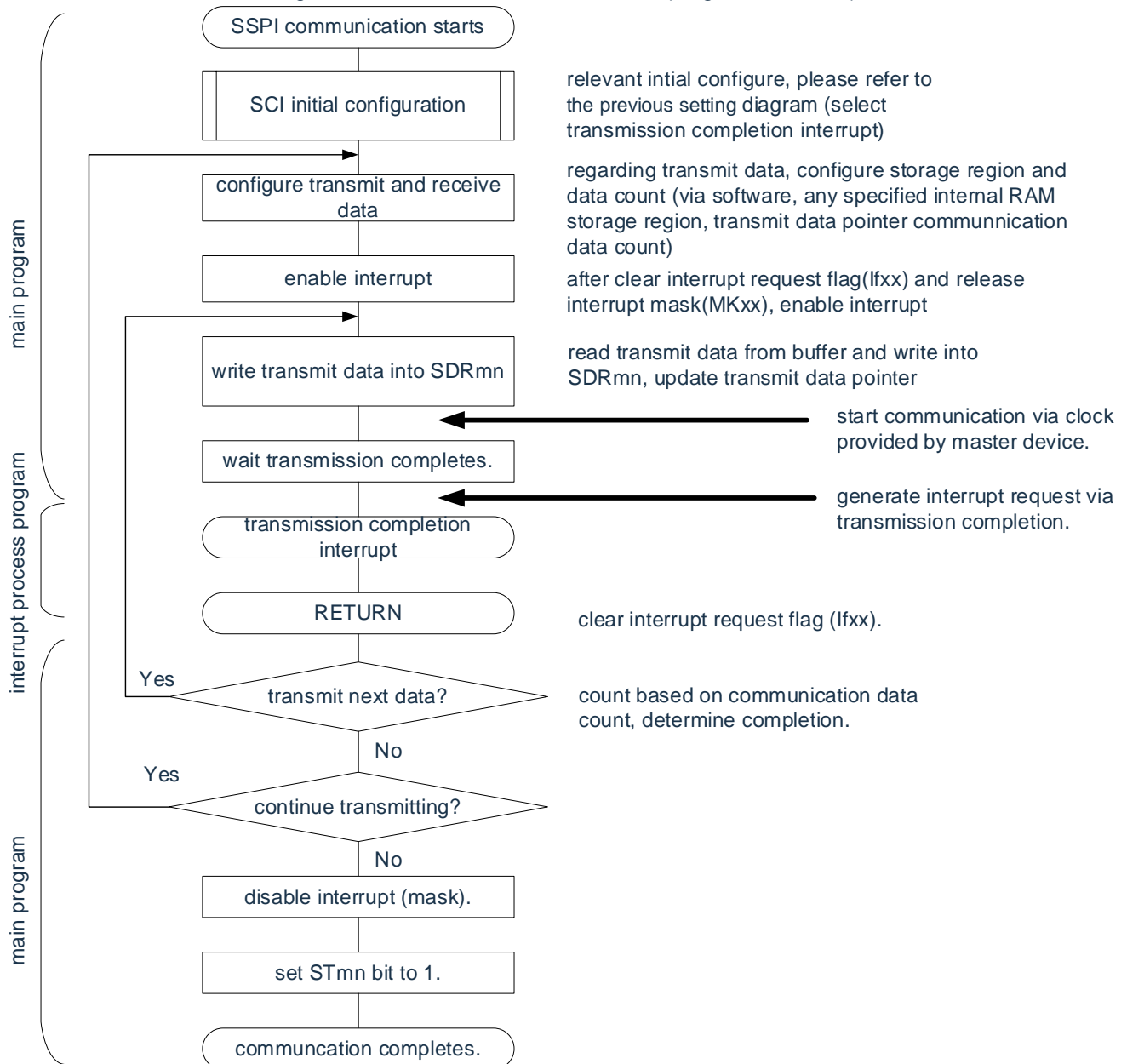
### (3) Process flow (single send mode)

Figure 16-51: Timing diagram of a Slave send (single send mode) (type 1: DAPmn=0, CKPmn=0)



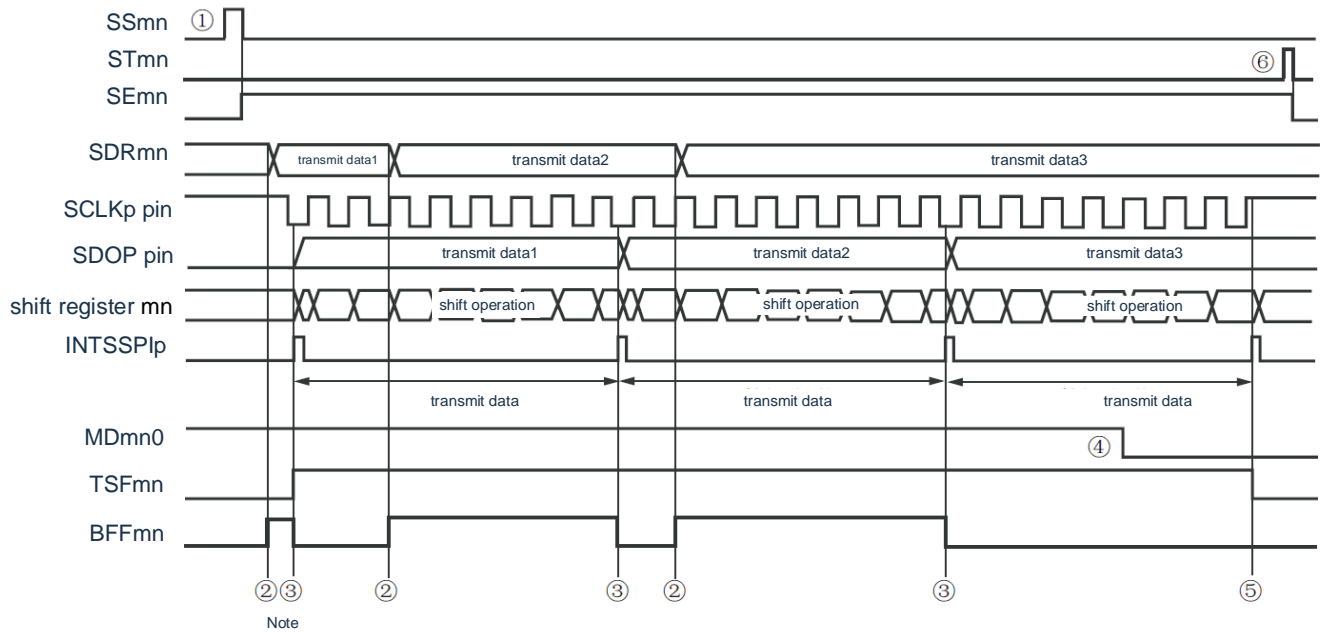
Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

Figure 16-52: Flowchart of Slave send (single send mode)



#### (4) Process flow (continuous send mode)

Figure 16-53: Timing diagram of Slave send (continuous send mode) (type 1: DAPmn=0, CKPmn=0)

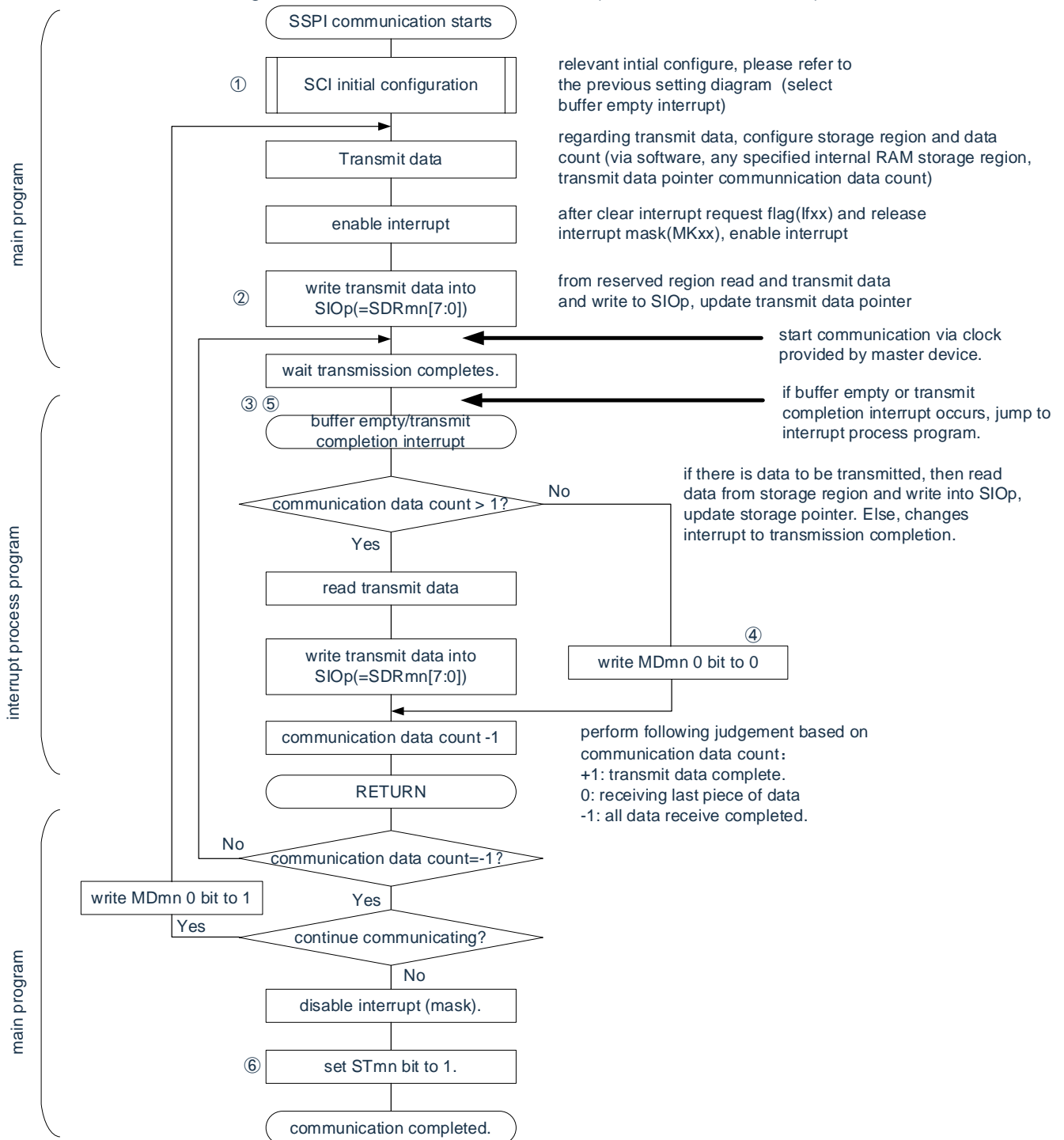


Note: If the BFFmn bit of the serial status register mn (SSRmn) is "1" (when valid data is saved in the serial data register mn (SDRmn)) is given. When the SDRmn register writes the transmit data, it rewrites the send data.

Notice: MDmn0 bit of the serial mode register mn (SMRmn) can be rewritten even in operation. However, it must be overwritten before the last bit can be transmitted.

Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

Figure 16-54: Flowchart of Slave send (continuous send mode)



Remark: (1)~(6) in the note figure corresponds to (1)~(6) in the “Figure 16-53: Timing diagram of Slave send (continuous send mode)”.



## 16.5.5 Slave reception

Slave reception refers to the operation of this product receiving data from other devices in the state of transmitting clocks from other devices.

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channels	SCI0	SCI0	SCI1	SCI1	SCI2	SCI2
	Channel 0	Channel 1	Channel 0	Channel 1	Channel 0	Channel 1
pins used	SCLK00 SDI00	SCLK01 SDI01	SCLK10 SDI10	SCLK11 SDI11	SCLK20 SDI20	SCLK21 SDI21
Interrupt	INTCSI00	INTCSI01	INTCSI10	INTCSI11	INTCSI20	INTCSI21
	Limited to end-of-transmit interrupts (buffer null interrupts are prohibited).					
Error detection flag	There are only overflow error detection flags (OVFmn).					
transferred data length	7-16 bits					
Transfer rate	Max. $F_{MCK}/6[Hz]$ <sup>Note 1,2</sup>					
Data phase	It can be selected by the DAPmn bit of the SCRmn register.					
	<ul style="list-style-type: none"> <li>DAPmn=0: Starts data output when the serial clock starts running.</li> <li>DAPmn=1: Starts the data output half a clock before the serial clock starts running.</li> </ul>					
Clock phase	It can be selected by the CKPmn bit of the SCRmn register					
	<ul style="list-style-type: none"> <li>CKPmn=0: positive phase</li> <li>CKPmn=1: inverted phase</li> </ul>					
Data direction	MSB preferred or LSB preferred					

Note 1: Because the external serial clocks input from SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, SCLK21 pins are sampled internally and then used, the maximum transfer rate is  $F_{MCK}/6[Hz]$ .

Note 2: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

Remark:

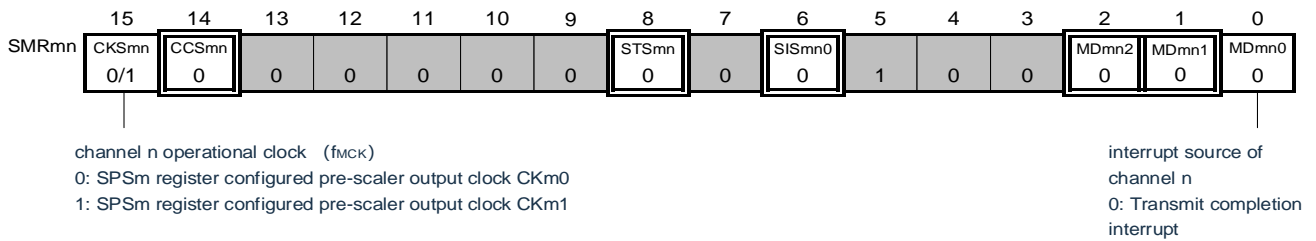
1.  $F_{MCK}$ : The operating clock frequency of the object channel
2. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)

## (1) Register setting

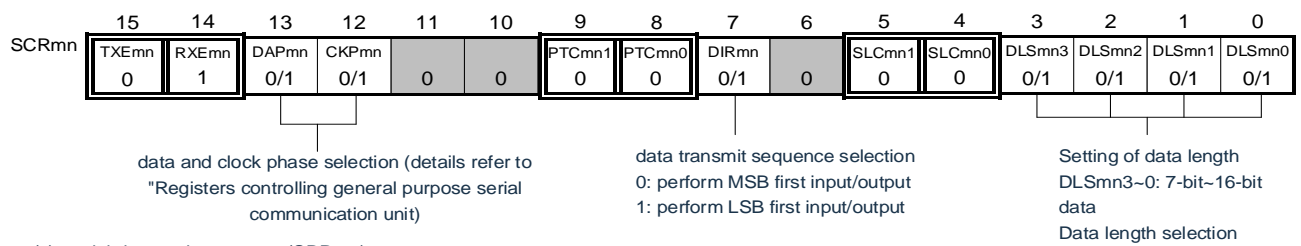
Figure16-55: 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)

Example of register setting content at slave receive time

(a) serial mode register mn (SMRmn)

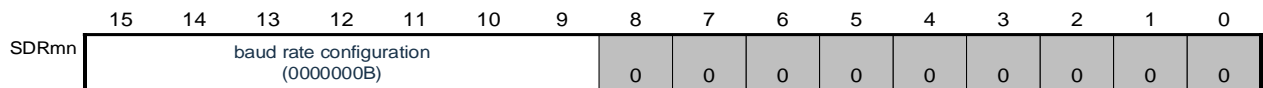


(b) serial communication operation configuration registermn mn (SCRmn)

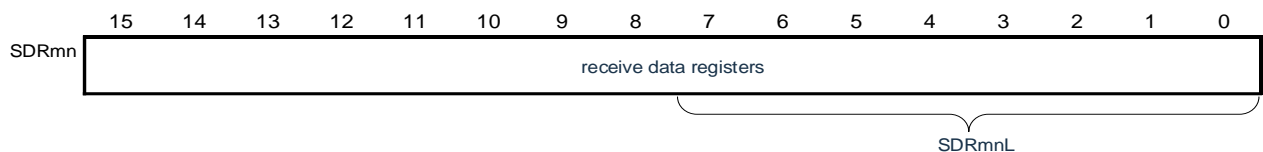


(c) serial data registermn mn (SDRmn)

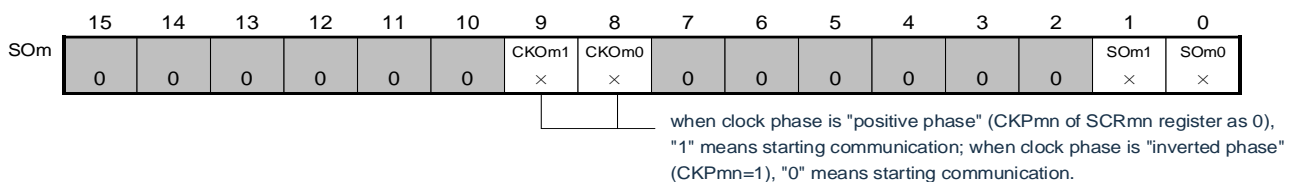
(1) When operation stops (SEmn=0)



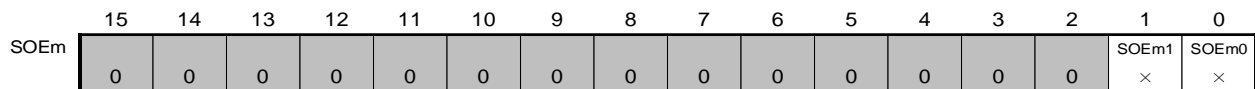
(2) During operation (SEmn=1) (lower 8 bits: SDRmnL)



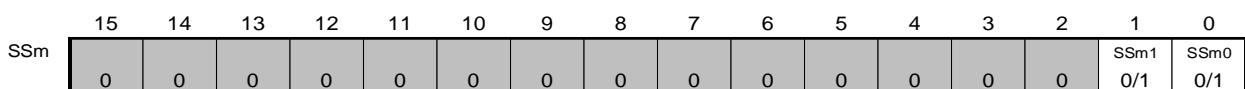
(d) serial output register m(SOm) .....not used in this mode.



(e) serial output enable register m(SOEm).....not used in this mode.



(f) serial channel start registerm (SSm)....only set bit of target channel to 1.



Remark:

- m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)
- : Fixed setting in slave receive mode.■: setting disabled(initial value).  
x: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).  
0/1: Set "0" or "1" according to the user's purpose.

## (2) Procedure

Figure16-56: Initial setup step of slave reception

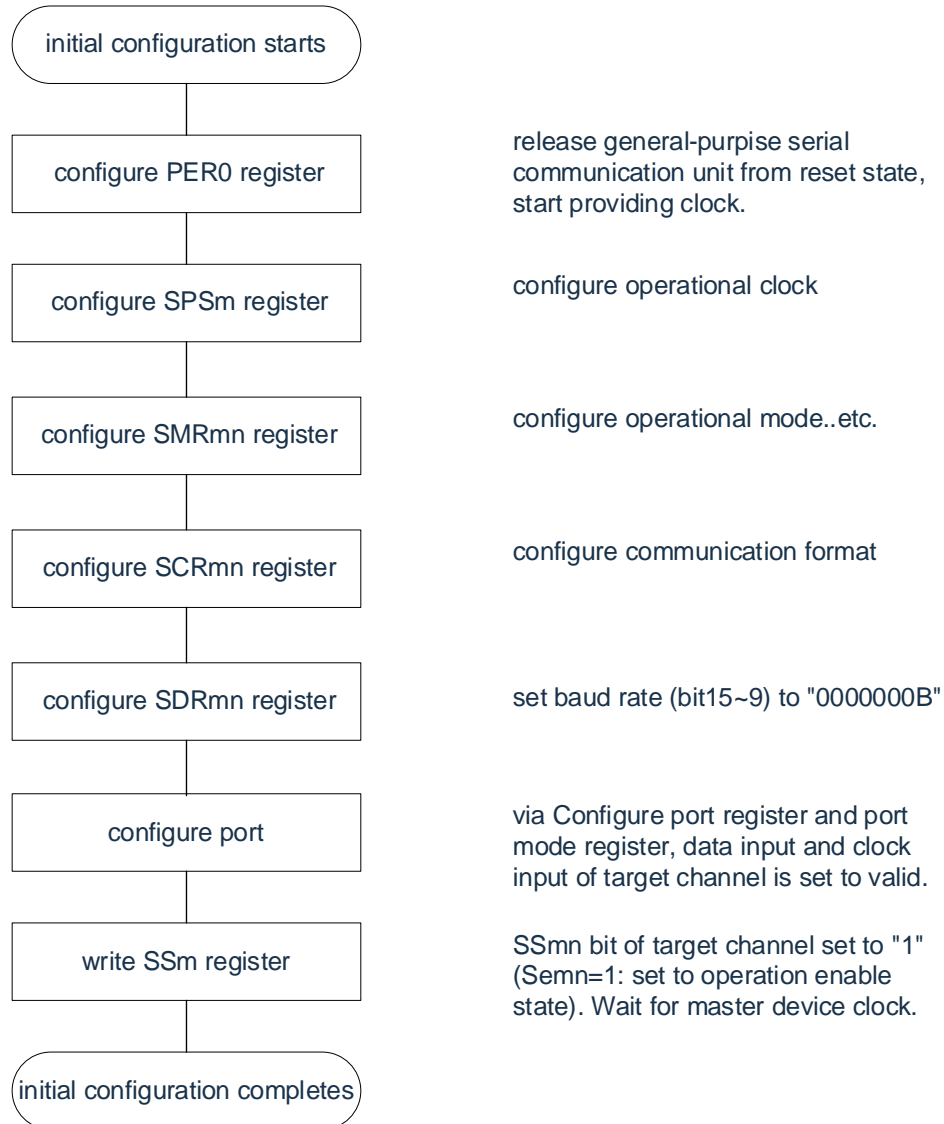


Figure16-57: Stop step of slave reception

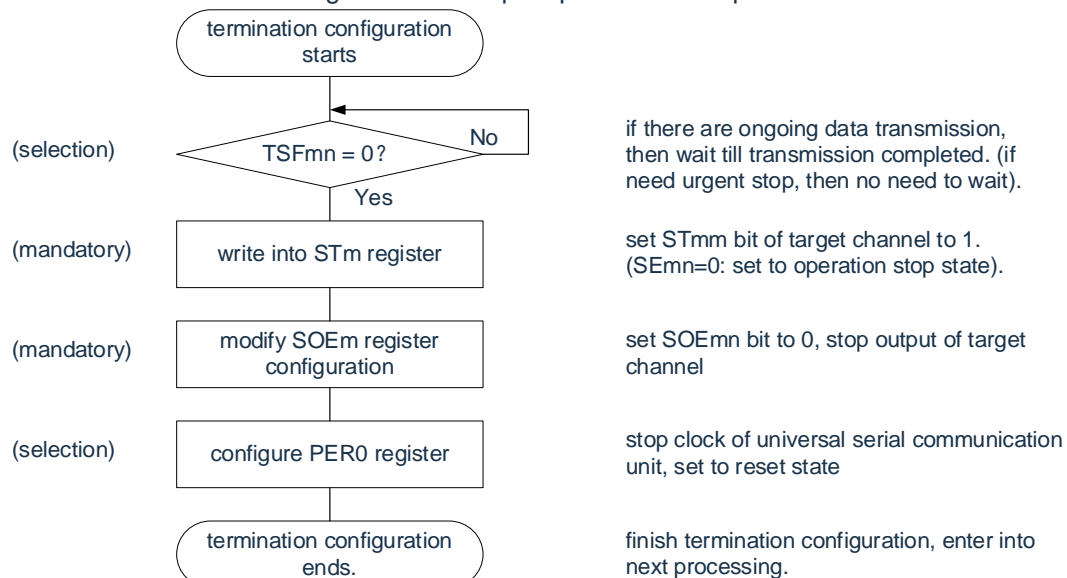
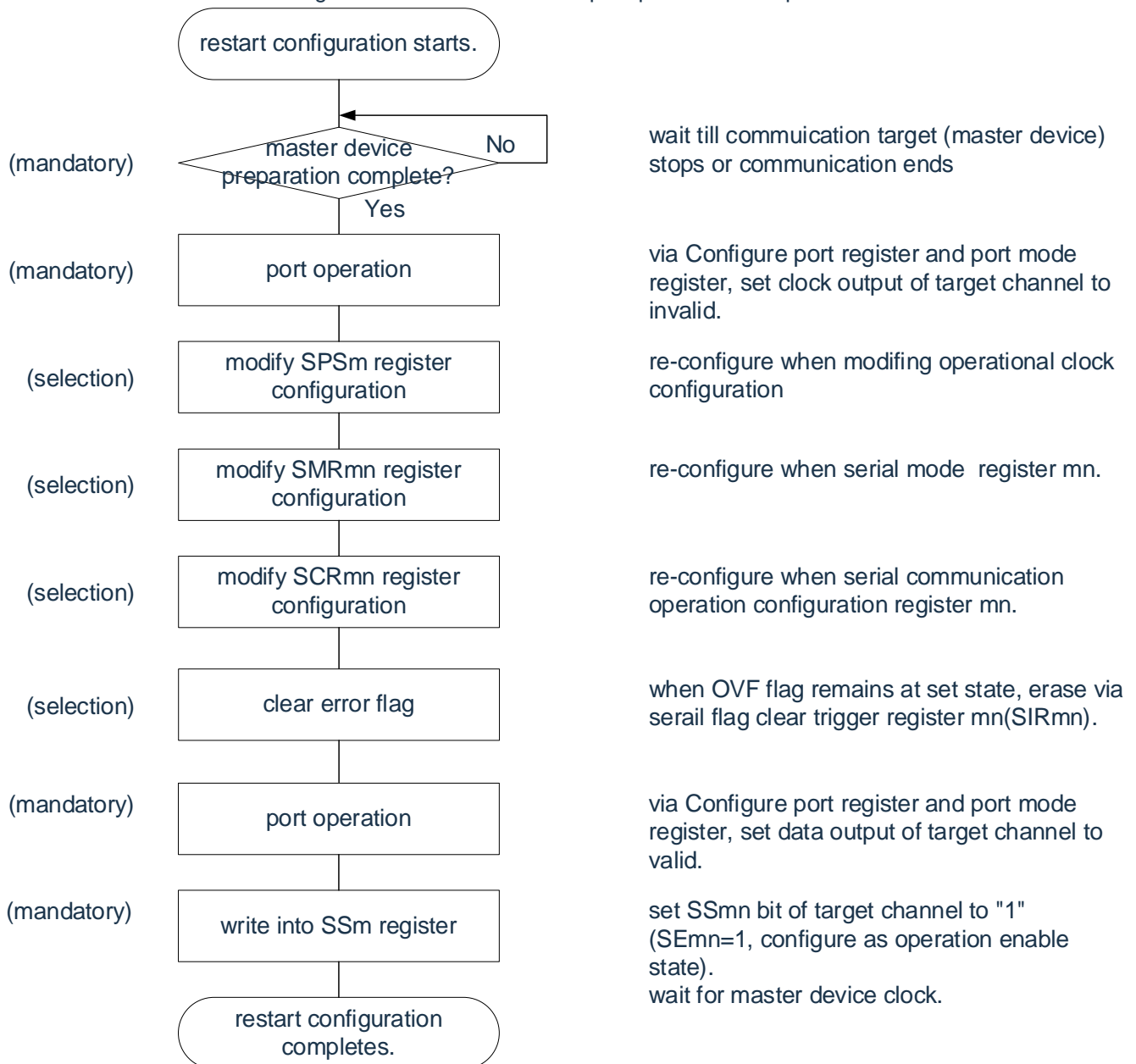


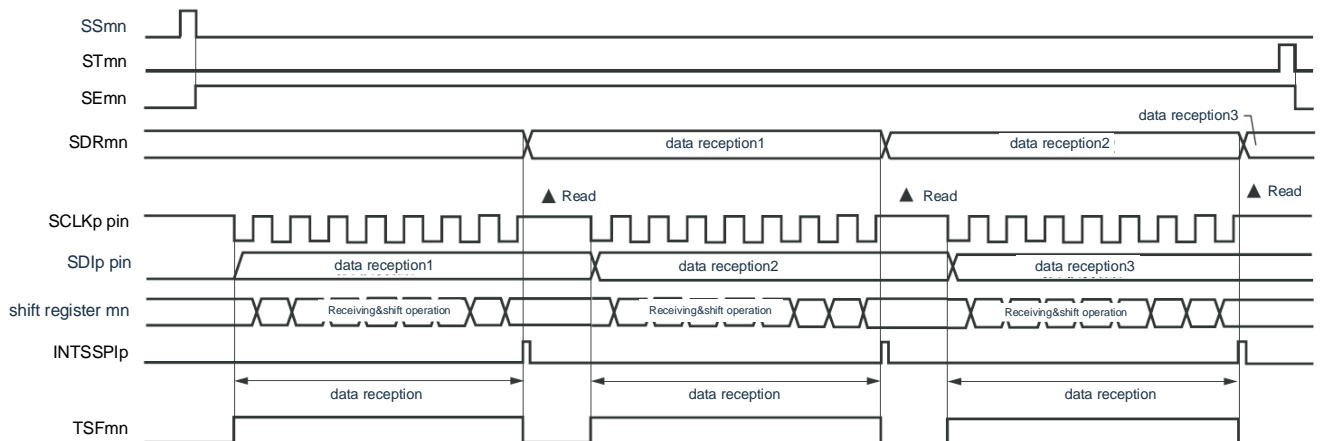
Figure16-58: Restart the setup step of slave reception



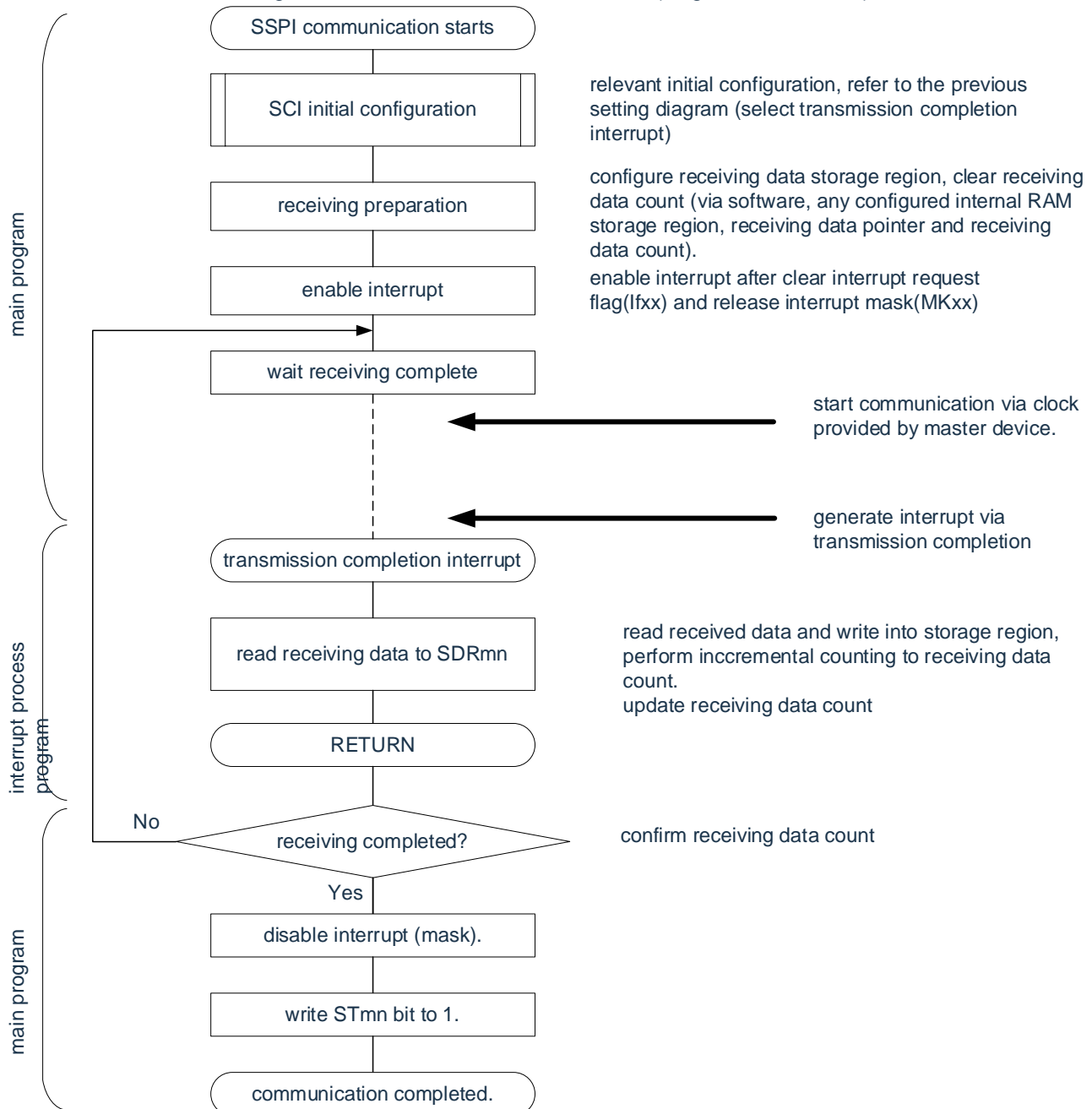
Remark: If PER0 is rewritten in the abort setting to stop providing the clock, the initial setting must be made after the communication object (master device) stops or communication ends instead of the restart setting.

### (3) Process flow (single receive mode)

Figure16-59: Timing diagram of slave receive (single receive mode) (type 1: DAPmn= 0, CKPmn = 0)



Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

**Figure16-60: Flowchart of slave receive (single receive mode)**


## 16.5.6 Slave transmission and reception

Slave transmission and reception refers to the operation of data transmission and reception by microcontrollers and other devices of this product in the state of transmitting clocks from other device inputs.

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channels	SCI0	SCI0	SCI1	SCI1	SCI2	SCI2
	Channel 0	Channel 1	Channel 0	Channel 1	Channel 0	Channel 1
Pin used	SCLK00	SCLK01	SCLK10	SCLK11	SCLK20	SCLK21
	SDI00	SDI01	SDI10	SDI11	SDI20	SDI21
	SDO00	SDO01	SDO10	SDO11	SDO20	SDO21
Interrupt	INTCSI00	INTCSI01	INTCSI10	INTCSI11	INTCSI20	INTCSI21
	End of transmission interrupt (single transmission mode) or buffer interrupt (continuous transmission mode) can be selected.					
Error detection flags	There are only overflow error detection flags(OVFmn).					
Transferred data length	7-16 bits					
Transfer rate	Max. $F_{MCK}/6[Hz]$ <sup>Note 1,2</sup>					
Data phase	It can be selected by the DAPmn bit of the SCRmn register.					
	<ul style="list-style-type: none"> <li>DAPmn=0: Starts data input/output when the serial clock starts running.</li> <li>DAPmn=1: Starts data input/output half a clock before the serial clock starts running.</li> </ul>					
Clock phase	It can be selected by the CKPmn bit of the SCRmn register.					
	<ul style="list-style-type: none"> <li>CKPmn=0: positive phase</li> <li>CKPmn=1: inverted phase</li> </ul>					
Data direction	MSB preferred or LSB preferred					

Note 1: Because the external serial clock input to SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, and SCLK21 pins are internally sampled and then used, the maximum transfer rate is  $F_{MCK}/6[Hz]$ .

Note 2: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

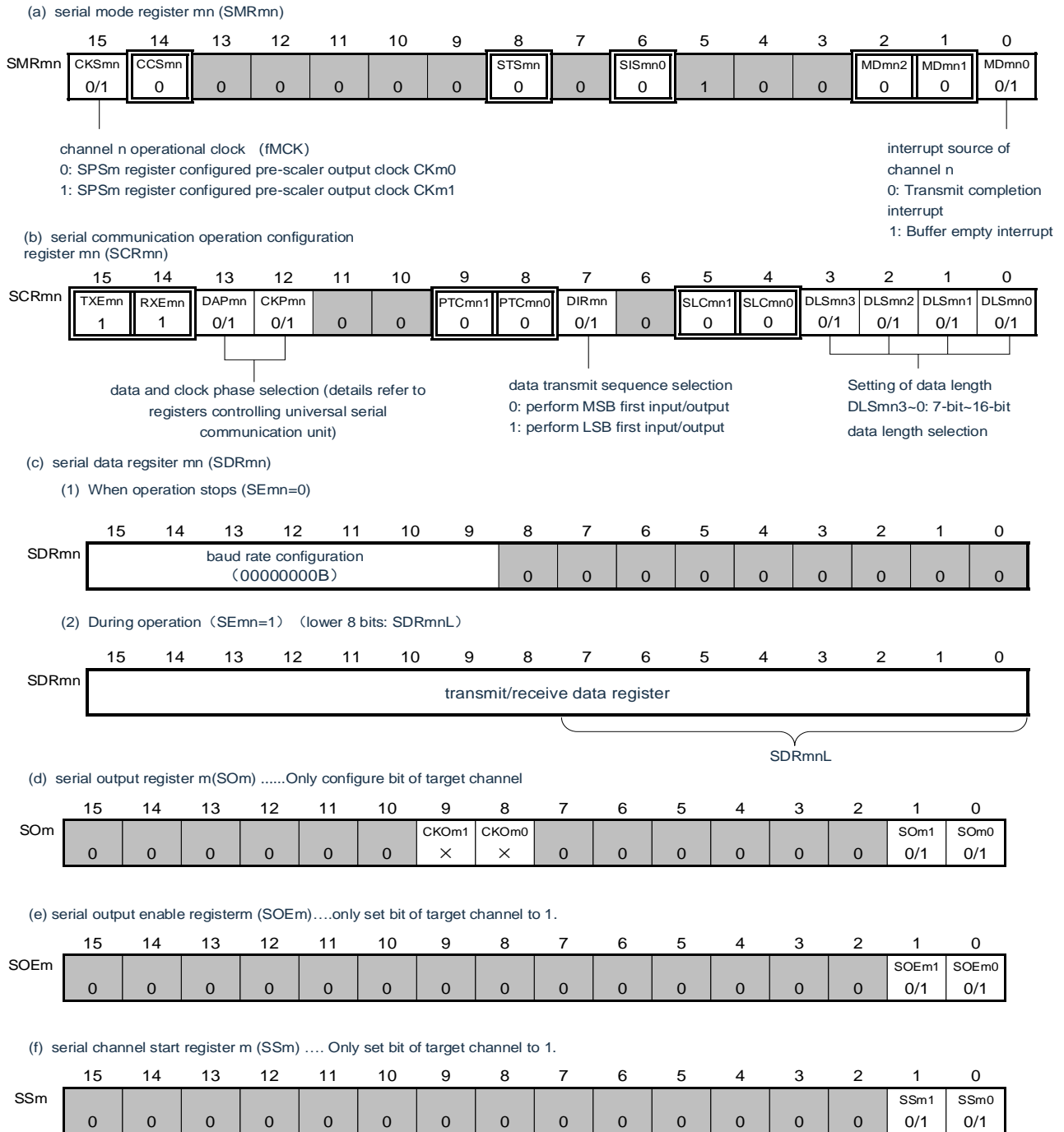
Remark:

1.  $F_{MCK}$ : The operating clock frequency of the object channel
2. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)

## 1) Register setting

Figure16-61: 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)

Example of register settings for Slave transmit and receive



Notice: The send data must be set to the SDRmn register before the master device starts outputting the clock.

Remark:

- m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)
- : Fixed setting in SSPI slave transmit and receive mode. ■: setting disabled(initial value).  
x: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).  
0/1: Set "0" or "1" according to the user's purpose.



## (2) Procedure

Figure16-62: Initial setup steps for slave sending and receiving



Notice: The send data must be set to the SDRmn register before the master device starts outputting the clock.

Figure 16-63: Stop steps for slave sending and receiving

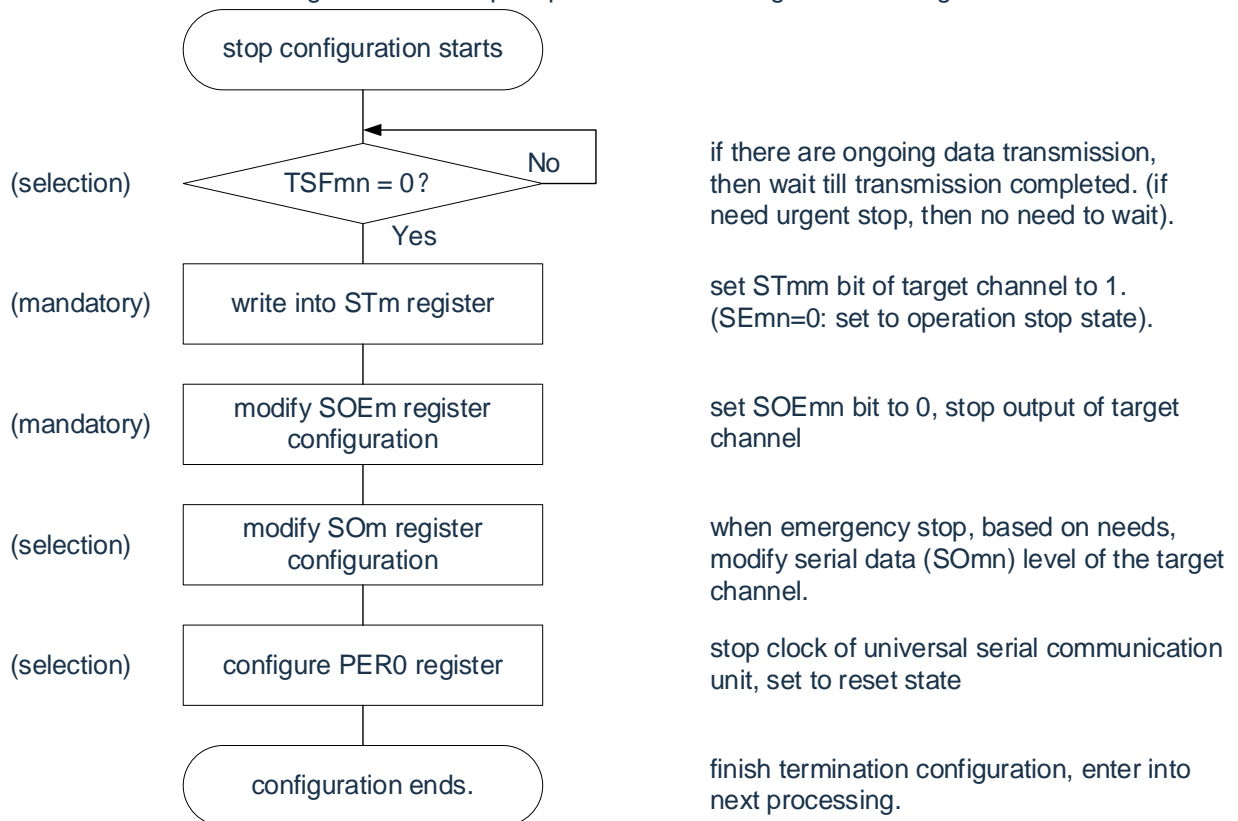
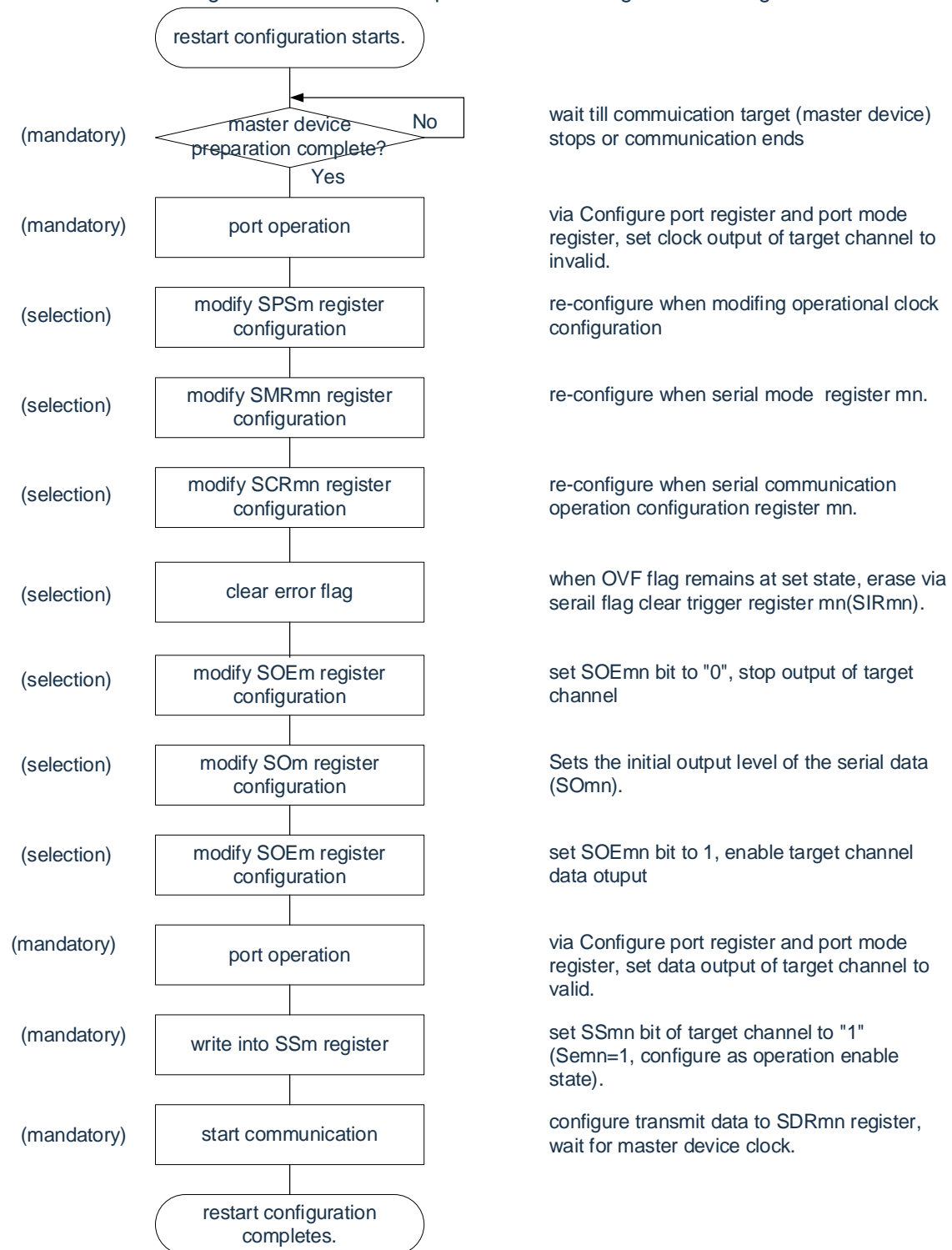


Figure 16-64: Restart steps for slave sending and receiving

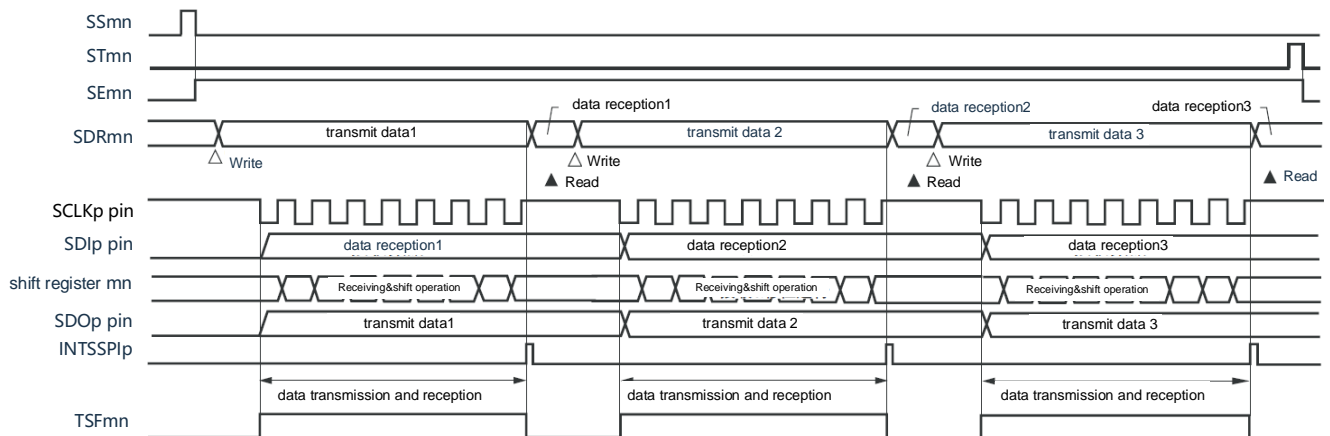


#### Notice:

1. The send data must be set to the SDRmn register before the master device starts outputting the clock.
2. If PER0 is rewritten in the abort setting to stop providing the clock, the initial setting must be made after the communication object (master device) stops or communication ends instead of the restart setting.

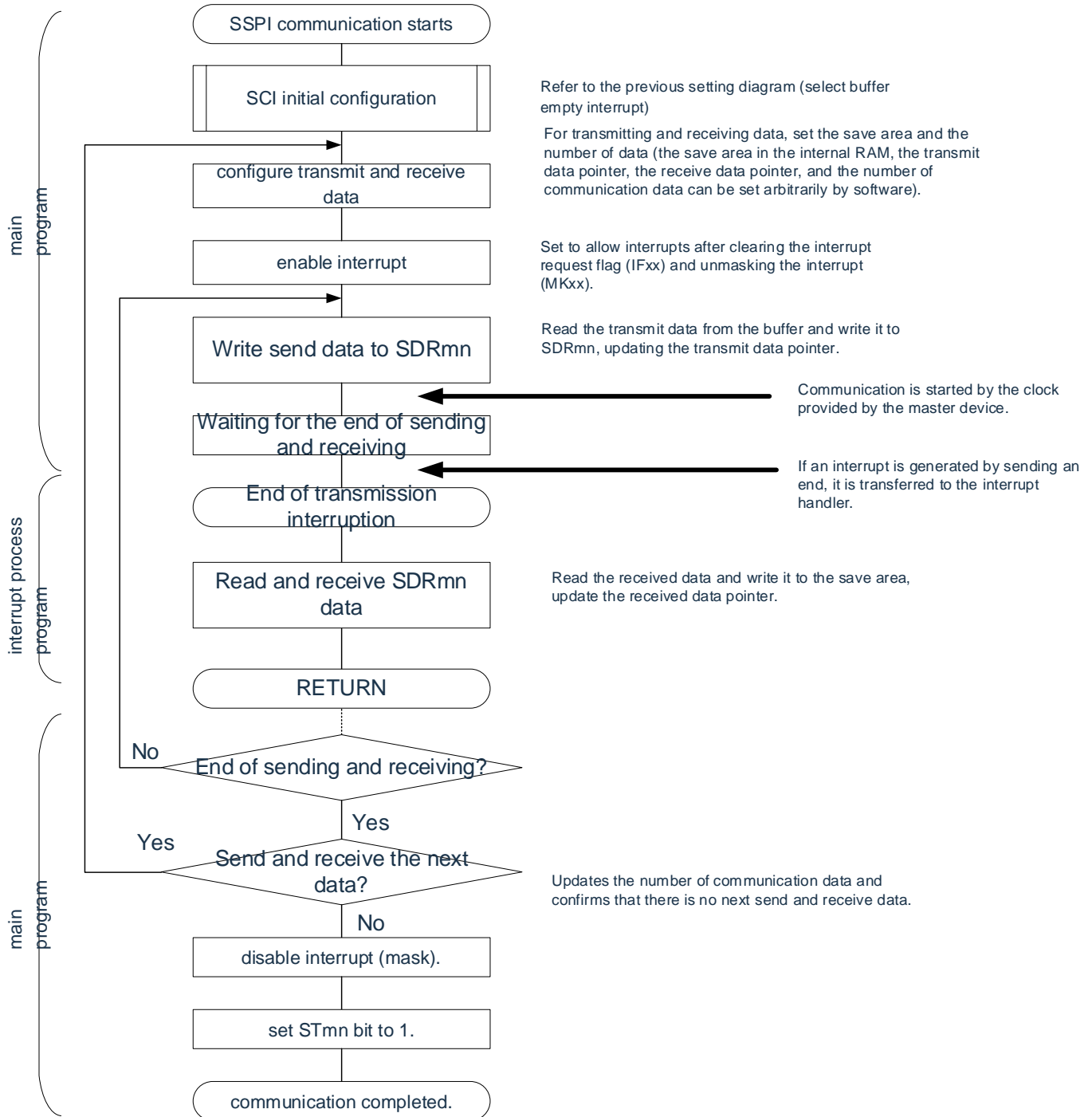
### (3) Process flow (single send and receive mode)

Figure16-65: Timing diagram of Slave transmit and receive (single transmit and receive mode)



Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

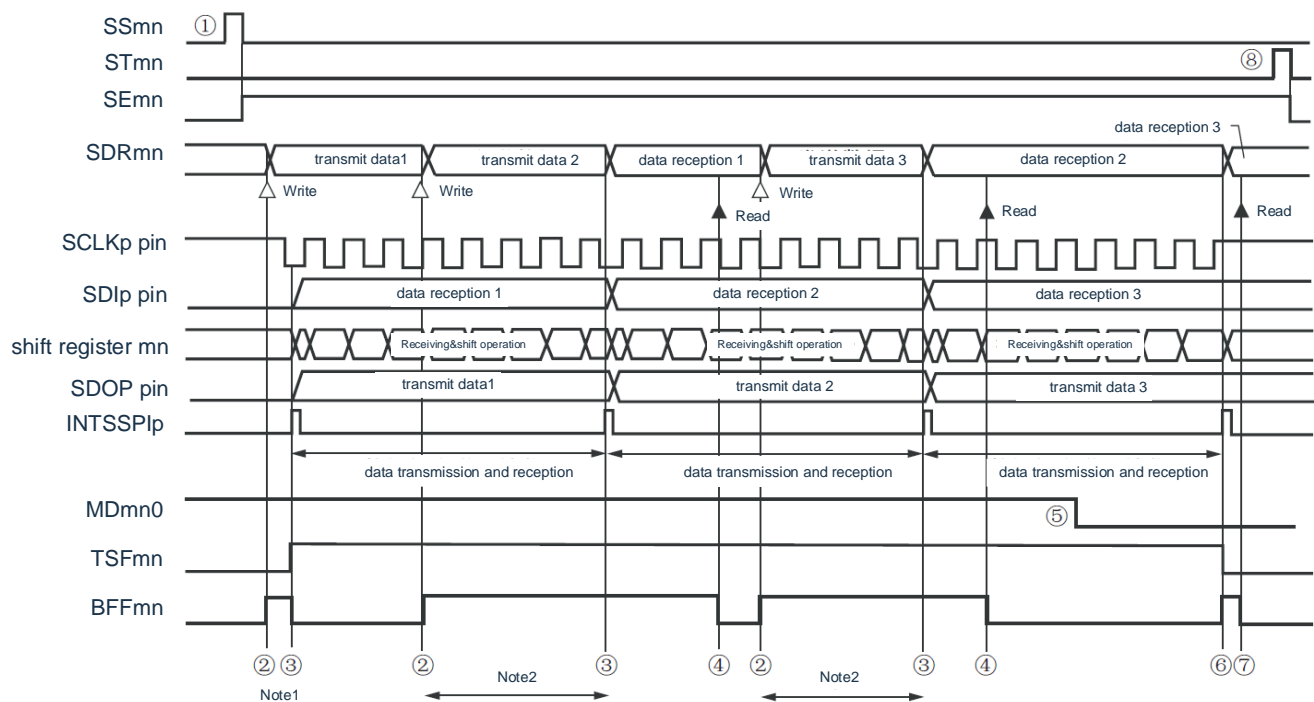
Figure 16-66: Flowchart of slave send and receive (single send and receive mode)



Notice: SIOp register must be set to send data before the master device starts to output the clock.

#### (4) Process flow (continuous send and receive mode)

Figure16-67: Timing diagram of Slave send and receive (continuous transmit and receive mode) (type 1: DAPmn=0, CKPmn=0)



Note 1: If the BFFmn bit of serial status register mn(SSRmn) is "1" during the period (valid data is saved in serial data register mn(SDRmn) (when writing and sending data to the SDRmn memory, rewrite the send data.

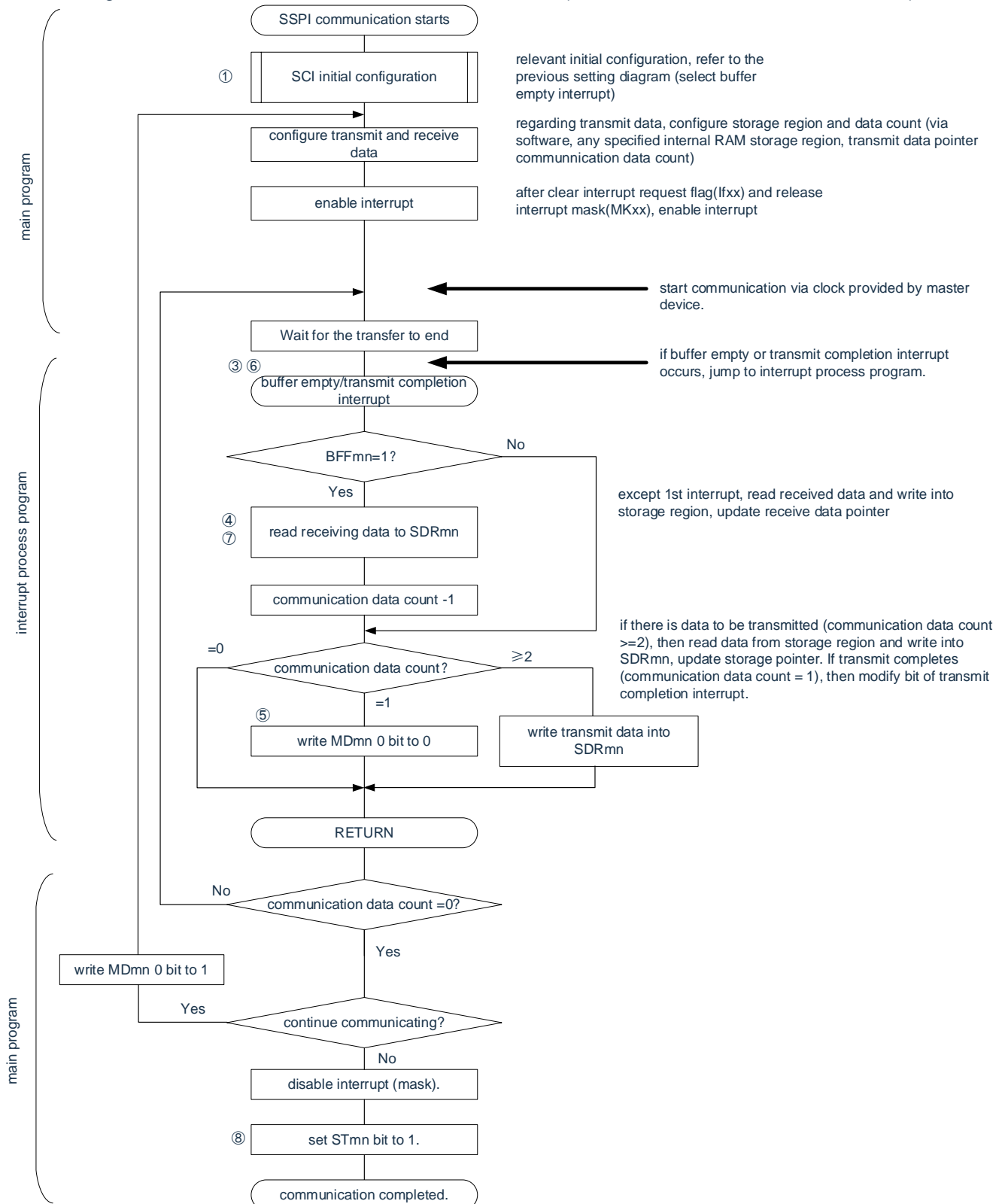
Note 2: If the SDRmn register is read during this period, the data can be read and sent. At this point, the transfer run is not affected.

Notice: MDmn0 bit of the serial mode register mn (SMRmn) can be rewritten even in operation. However, in order to catch the end-of-transmission interruption of the last sent data, it must be rewritten before the last bit of transmission begins.

Remark:

- (1) ~ (8) in the figure corresponds to (1) ~ (8) in "Figure16-68: Flowchart of Slave transmit and receive (continuous transmit and receive mode)"
- m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)

Figure16-68: Flowchart of Slave transmit and receive (continuous transmit and receive mode)



Notice: The send data must be set to the SDRmn register before the master device starts outputting the clock.

Remark: (1)~(8) in the note figure corresponds to (1)~(8) in “Figure16-67 Timing diagram of Slave send and receive (continuous transmit and receive mode)”.

## 16.5.7 Calculation of transmission 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

$$\text{(Transmit clock frequency)} = \{ \text{Object Channel's operating clock (F}_{\text{MCK}}\text{) frequency} \} \div (\text{SDRmn}[15:9] + 1) \div 2 [\text{Hz}]$$

$$\text{(Transmit clock frequency)} = \{ \text{Serial Clock (SCLK) Frequency Rate Provided by the Master Device (SCLK)} \} \text{Note} [\text{Hz}]$$

(2) Slave

Note: The maximum allowable transmit clock frequency is  $F_{\text{MCK}}/6$ .

Remark:

1. Because the value of SDRmn [15:9] is the value of bit15~9 of the serial data register mn(SDRmn). 1111111B), so it is 0~127.
2. The operating clock( $F_{\text{MCK}}$ ) depends on bit15 (CKSmn) of the serial clock selection register m (SPSm) and the serial mode register mn (SMRmn).



Table 16-2: 3-wire serial I/O operating clocks

SMRmn Register	SPSm register								Running Clock ( $f_{MCK}$ ) <sup>Note</sup>	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		When $F_{CLK}=32MHz$ is in operation
0	X	X	X	X	0	0	0	0	$F_{CLK}$	32MHz
	X	X	X	X	0	0	0	1	$F_{CLK}/2$	16MHz
	X	X	X	X	0	0	1	0	$F_{CLK}/2^2$	8MHz
	X	X	X	X	0	0	1	1	$F_{CLK}/2^3$	4MHz
	X	X	X	X	0	1	0	0	$F_{CLK}/2^4$	2MHz
	X	X	X	X	0	1	0	1	$F_{CLK}/2^5$	1MHz
	X	X	X	X	0	1	1	0	$F_{CLK}/2^6$	500KHz
	X	X	X	X	0	1	1	1	$F_{CLK}/2^7$	250KHz
	X	X	X	X	1	0	0	0	$F_{CLK}/2^8$	125KHz
	X	X	X	X	1	0	0	1	$F_{CLK}/2^9$	62.5KHz
	X	X	X	X	1	0	1	0	$F_{CLK}/2^{10}$	31.25KHz
	X	X	X	X	1	0	1	1	$F_{CLK}/2^{11}$	15.63KHz
	X	X	X	X	1	1	0	0	$F_{CLK}/2^{12}$	7.81KHz
	X	X	X	X	1	1	0	1	$F_{CLK}/2^{13}$	3.91KHz
	X	X	X	X	1	1	1	0	$F_{CLK}/2^{14}$	1.95KHz
	X	X	X	X	1	1	1	1	$F_{CLK}/2^{15}$	977Hz
1	0	0	0	0	X	X	X	X	$F_{CLK}$	32MHz
	0	0	0	1	X	X	X	X	$F_{CLK}/2$	16MHz
	0	0	1	0	X	X	X	X	$F_{CLK}/2^2$	8MHz
	0	0	1	1	X	X	X	X	$F_{CLK}/2^3$	4MHz
	0	1	0	0	X	X	X	X	$F_{CLK}/2^4$	2MHz
	0	1	0	1	X	X	X	X	$F_{CLK}/2^5$	1MHz
	0	1	1	0	X	X	X	X	$F_{CLK}/2^6$	500KHz
	0	1	1	1	X	X	X	X	$F_{CLK}/2^7$	250KHz
	1	0	0	0	X	X	X	X	$F_{CLK}/2^8$	125KHz
	1	0	0	1	X	X	X	X	$F_{CLK}/2^9$	62.5KHz
	1	0	1	0	X	X	X	X	$F_{CLK}/2^{10}$	31.25KHz
	1	0	1	1	X	X	X	X	$F_{CLK}/2^{11}$	15.63KHz
	1	1	0	0	X	X	X	X	$F_{CLK}/2^{12}$	7.81KHz
	1	1	0	1	X	X	X	X	$F_{CLK}/2^{13}$	3.91KHz
	1	1	1	0	X	X	X	X	$F_{CLK}/2^{14}$	1.95KHz
	1	1	1	1	X	X	X	X	$F_{CLK}/2^{15}$	977Hz

Note: When you change the clock selected as  $f_{CLK}$  (change the value of the system clock control register (CKC)), you must stop the operation of the general-purpose serial communication unit (SCI) (serial channel stop register m (STm)=000FH) after making changes.

Remark:

1. X: Ignore
2. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)

## 16.5.8 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, such as Figure 16-69 shows.

Figure 16-69: Processing steps when an overflow error occurs

Software operation	Hardware status	Remark
Read serial data register mn(SDRmn).	The BFFmn bit of the SSRmn register is "0" and the channel n is in a receiver state.	This is to prevent an overflow error from occurring to end the next receipt during error handling.
Read serial status register mn(SSRmn).		The type of error is determined and the read value is used to clear the error flag.
Clear the trigger register for the serial flag mn Set (SDIRmn) to "1".	Clear error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors in the read operation can only be cleared.\

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

## 16.6 Operation of clock-synchronous serial communication with slave selection input function

Each channel is a channel that supports clock-synchronous serial communication with a slave select input function.

[Data transmission and reception]

- 7~16 bit data length
- Phase control of transmit/receive data
- MSB/LSB preferred option
- Level setting for transmit/receive data

[Clock Control]

- Phase control of input/output clocks
- Set the transmission period generated by the prescaler and the channel internal counter.
- Maximum transmission rate<sup>Note</sup> Slave communication: Maximum value  $F_{MCK}/6$

[Interrupt function]

- Transmit end interrupt, buffer empty interrupt

[Error detection flag]

- Overflow error

Note: It must be used within the range that satisfies the SCLK cycle time ( $T_{KCY}$ ) characteristics. For details, please refer to the data sheet.

The slave selection input function has the following 3 types of communication operation:

- Slave transmission (refer to 16.6.1)
- Slave reception(refer to 16.6.2)
- Slave transmission and reception(refer to 16.6.3)

By using the slave selection input function, a master device can be connected to multiple slave devices for communication. The master device outputs the slave selection signal to the slave device (1) of the communication object, and each slave device determines whether it is selected as the communication object and controls the output of the SDO pin. When a slave device is selected as the communication object, the SDO pin can send data to the master device, and when a slave device is not selected as the communication object, the SDO pin becomes high resistance. In addition, the serial clock is not transmitted or received even if it is input to the master device.

Notice: The slave selection signal must be output by the operation of the port.

Figure16-70: Example of the structure of the slave selection input function

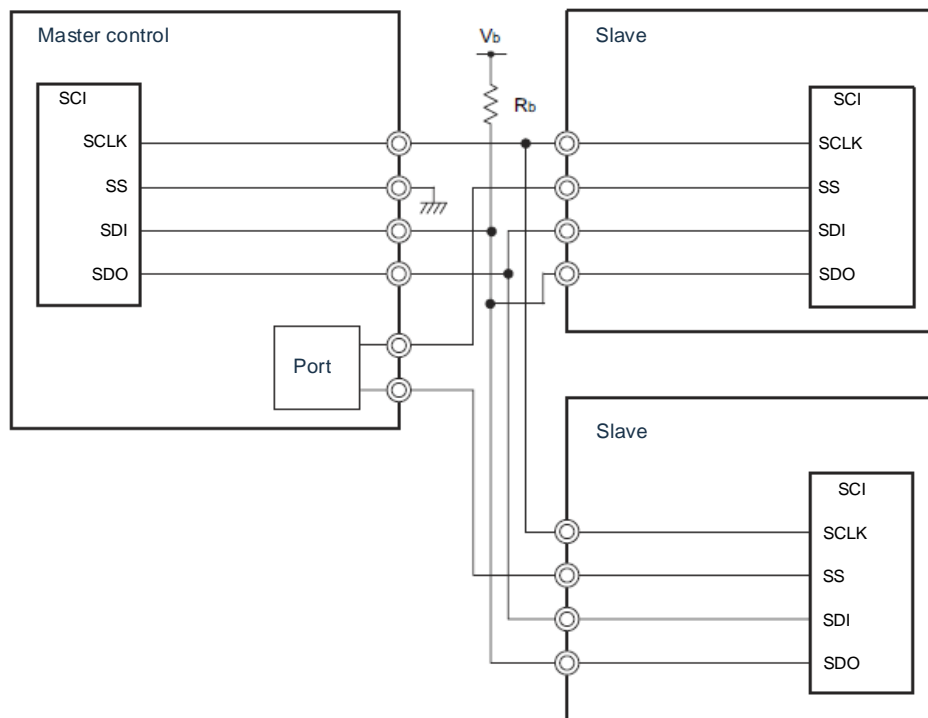
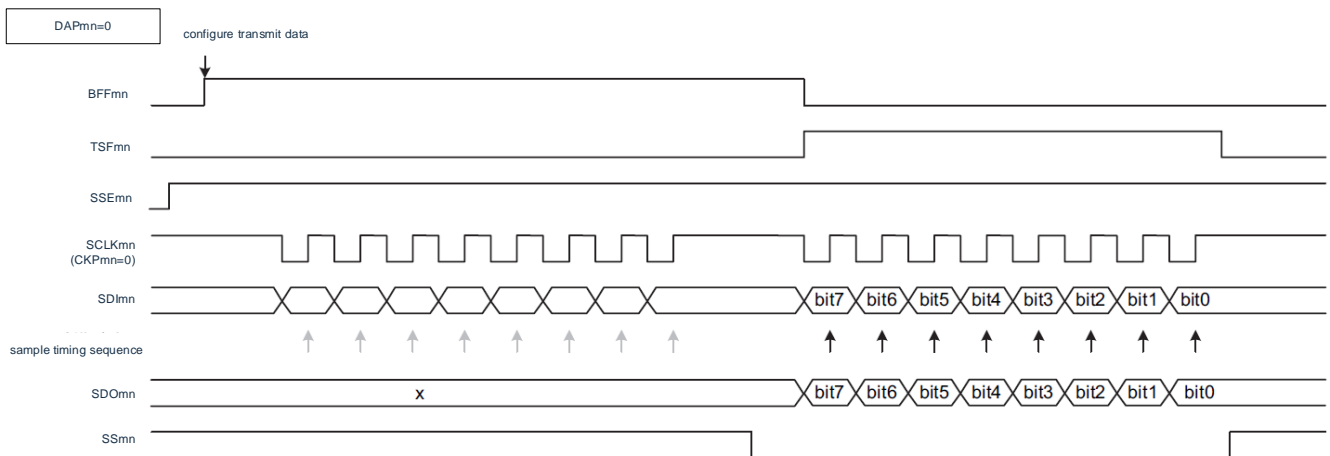
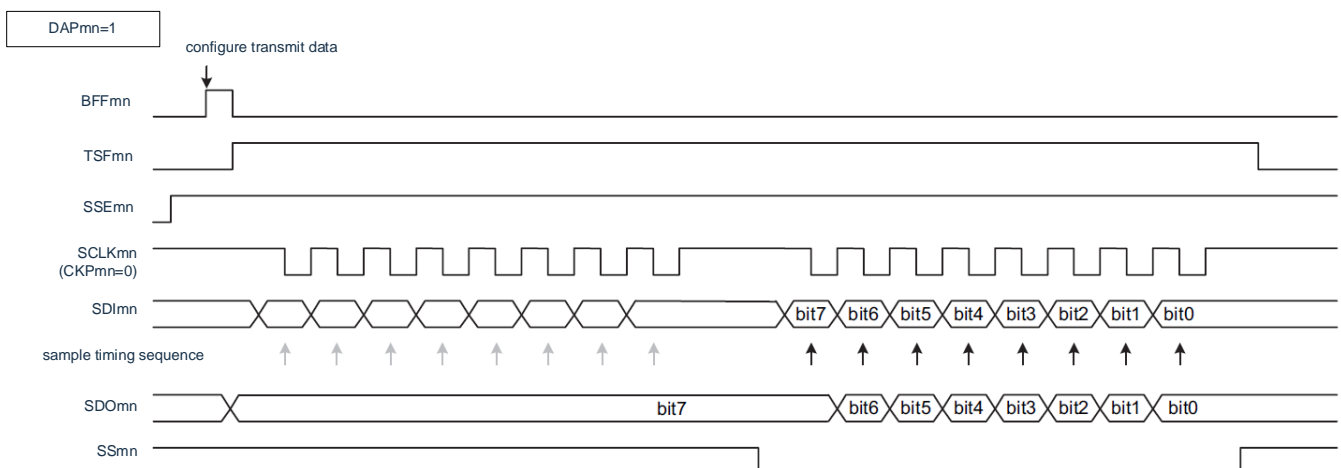


Figure16-71: Timing diagram of the Slave selection input function



During the period when  $SSImn$  is high, no transmission is performed even at the falling edge of  $SCKmn$  (serial clock), and no sampling of received data is performed synchronously with the rising edge.

When  $SSImn$  is low, data is output (shifted) synchronously with the falling edge of the serial clock and data is received synchronously with the rising edge.



When the  $DAPmn$  bit is "1", if the data is sent when  $SSmn$  is set to be high, the initial data (bit7) is provided to the data output. However, even the rising edge of the  $SCLKmn$  (serial clock) is not shifted, and the received data is not sampled in sync with the falling edge. If  $SSmn$  goes low, the output data is synchronized with the next rising edge (shift) and the data is received synchronously with the falling edge.

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

## 16.6.1 Slave transmission

Slave transmission refers to the operation of this product to send data to other devices in the state of transmitting clocks from other device inputs.

Slave selection input function	SSPImn
Object channels	Channel n of SCIm
Pin used	SCLKmn, SDOmn, SSImn
Interrupt	INTST0, INTSR0, INTST1, INTSR1, INTST2, INTSR2
	Selectable transmission end interrupt (single transmission mode) or buffer air interrupt (continuous transmission mode).
Error detection flags	There are only overflow error detection flags(OVFmn).
Transferred data length	7-16 bits
Transfer rate	Max.F <sub>MCK</sub> /6[Hz] <sup>Note 1,2</sup>
Data phase	It can be selected by the DAPmn bit of the SCRmn register.
	<ul style="list-style-type: none"> <li>DAPmn=0: Starts data output when the serial clock starts running.</li> <li>DAPmn=1: Starts the data output half a clock before the serial clock starts running.</li> </ul>
Clock phase	It can be selected by the CKPmn bit of the SCRmn register.
	<ul style="list-style-type: none"> <li>CKPmn=0: positive phase</li> <li>CKPmn=1: inverted phase</li> </ul>
Data direction	MSB preferred or LSB preferred
Slave selection input function	Optional operation of the slave selection function

Note 1: Because it is used after internal sampling of the external serial clock at the SCLK00 pin input, the maximum transfer rate is F<sub>MCK</sub>/6[Hz].

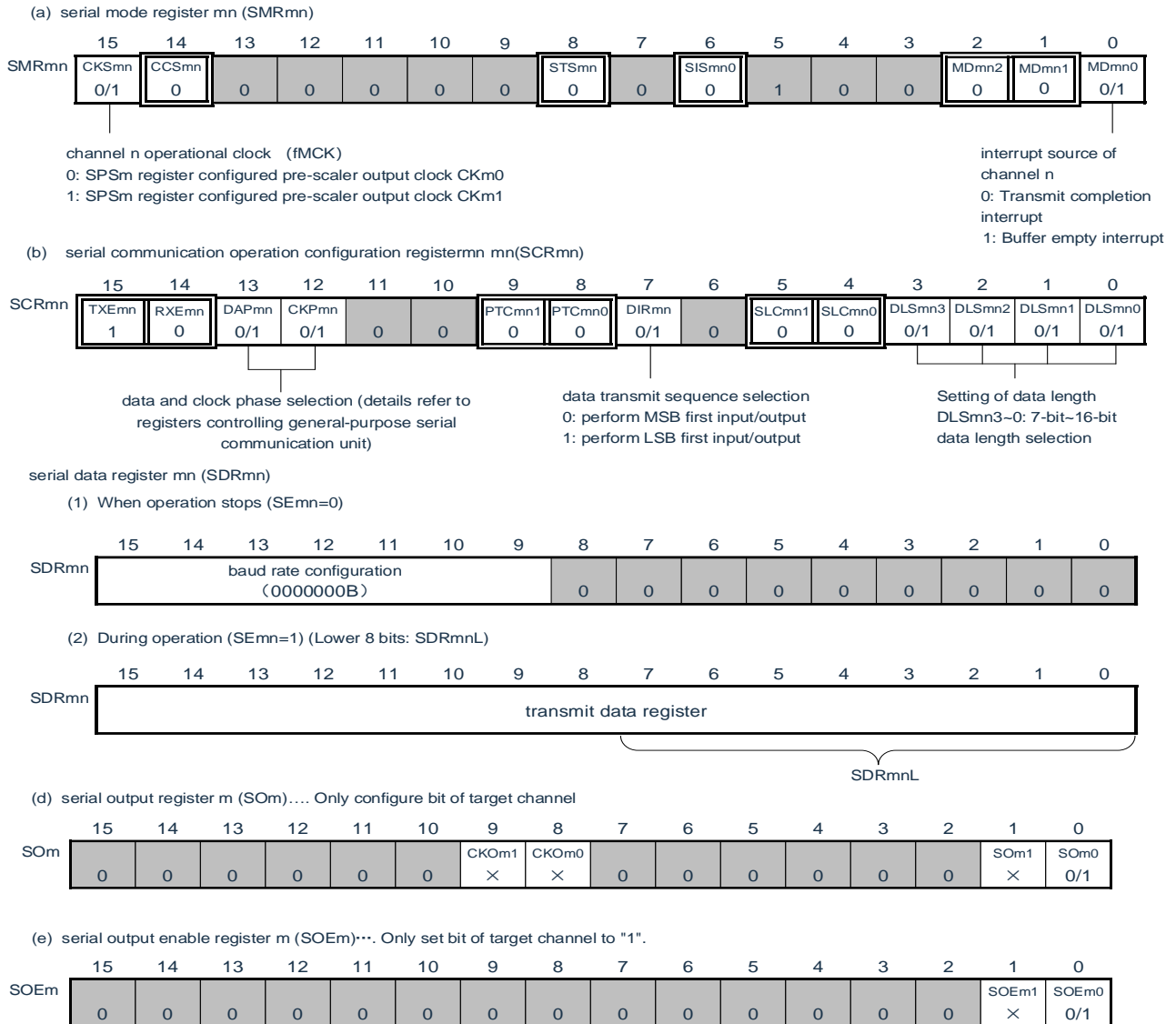
Note 2: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

Remark:

1. F<sub>MCK</sub>: The operating clock frequency of the object channel
2. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)

## (1) Register setting

Figure16-72: Example of register settings when slave select input function (SSPImn) slave transmits (1/2)



### Remark:

- m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)
- : Fixed setting in SSPI slave send mode. ■: setting disabled(initial value).  
x: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).  
0/1: Set "0" or "1" according to the user's purpose.

**Figure 16-72: Example of register settings when slave select input function (SSPImn) slave transmits (2/2)**

(f) serial channel start register m (SSm) .... Only set bit of target channel to 1.



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SSm1 X	SSm0 0/1

(g) Slave Select Function Enable Register (SSE) ..... This is the control of the SSImn pin of the SSPImn slave channel (channel n of unit m).

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSEm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SSIEm1 0/1	SSIEm0 0/1

0: SS00 pin input invalid  
1: SS00 pin input valid

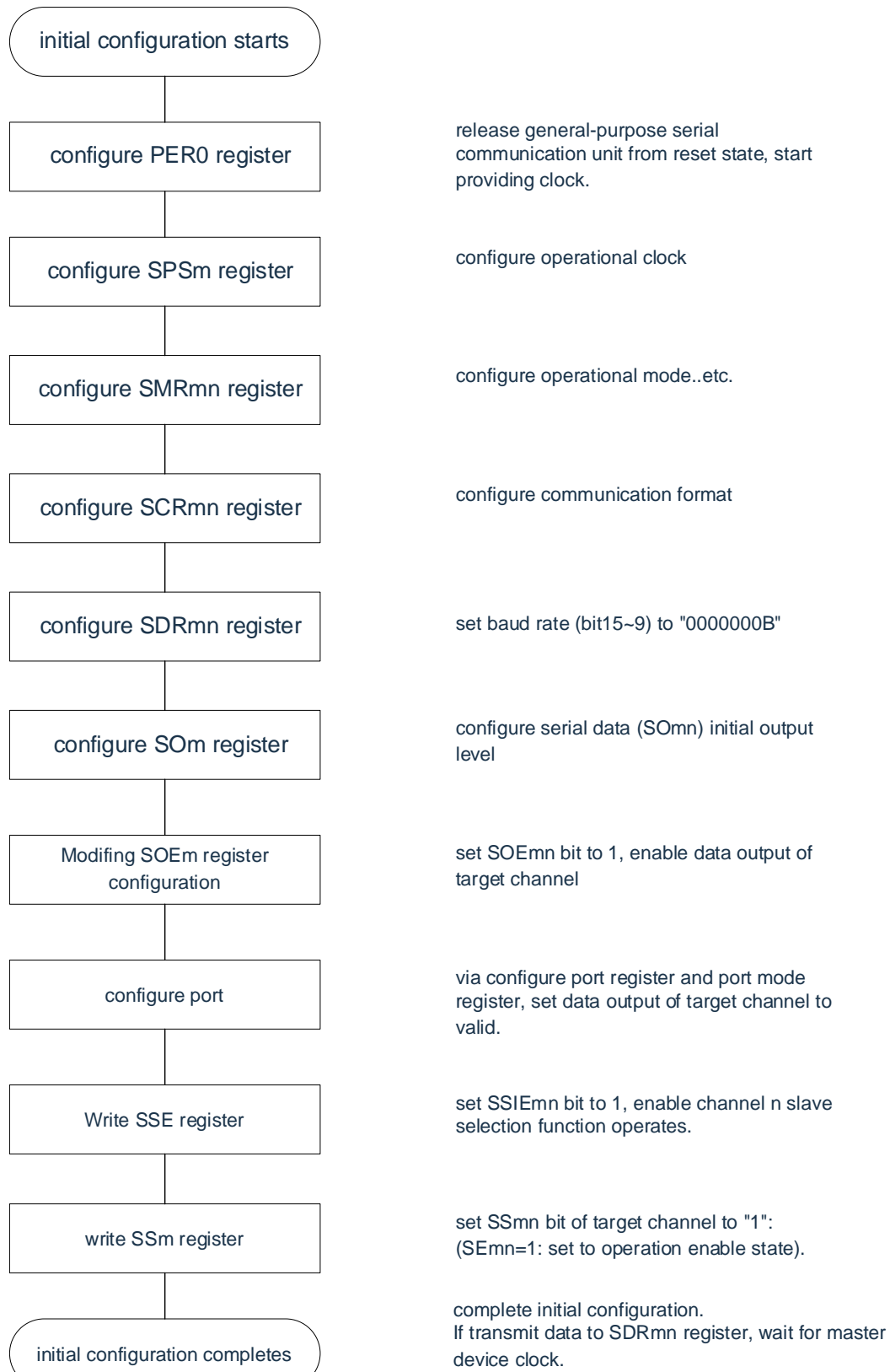
#### Remark:

1. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)
2. : Fixed setting in SSPI slave send mode. : setting disabled(initial value).  
x: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).  
0/1: Set "0" or "1" according to the user's purpose.



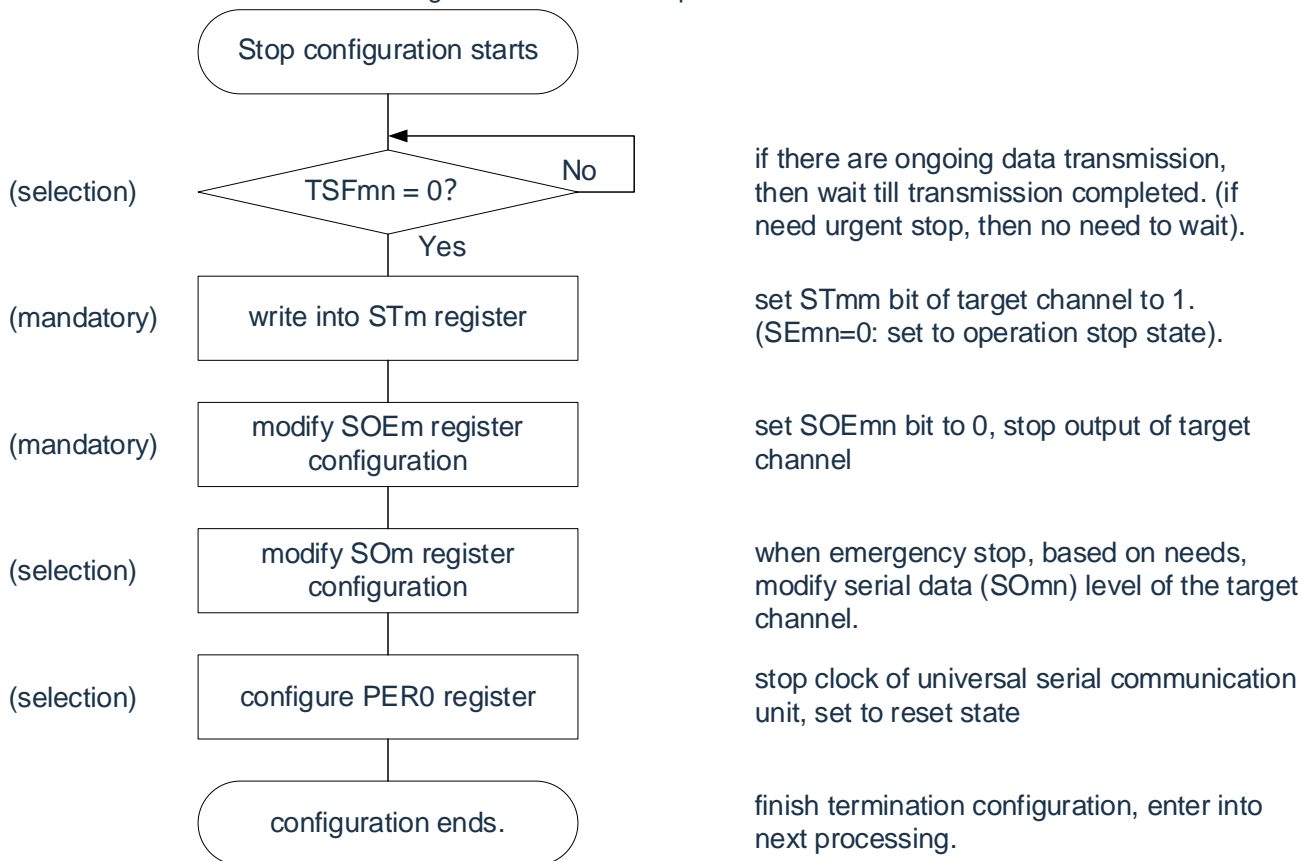
## (2) Procedure

Figure16-73: Initial setup steps for slave sending



Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

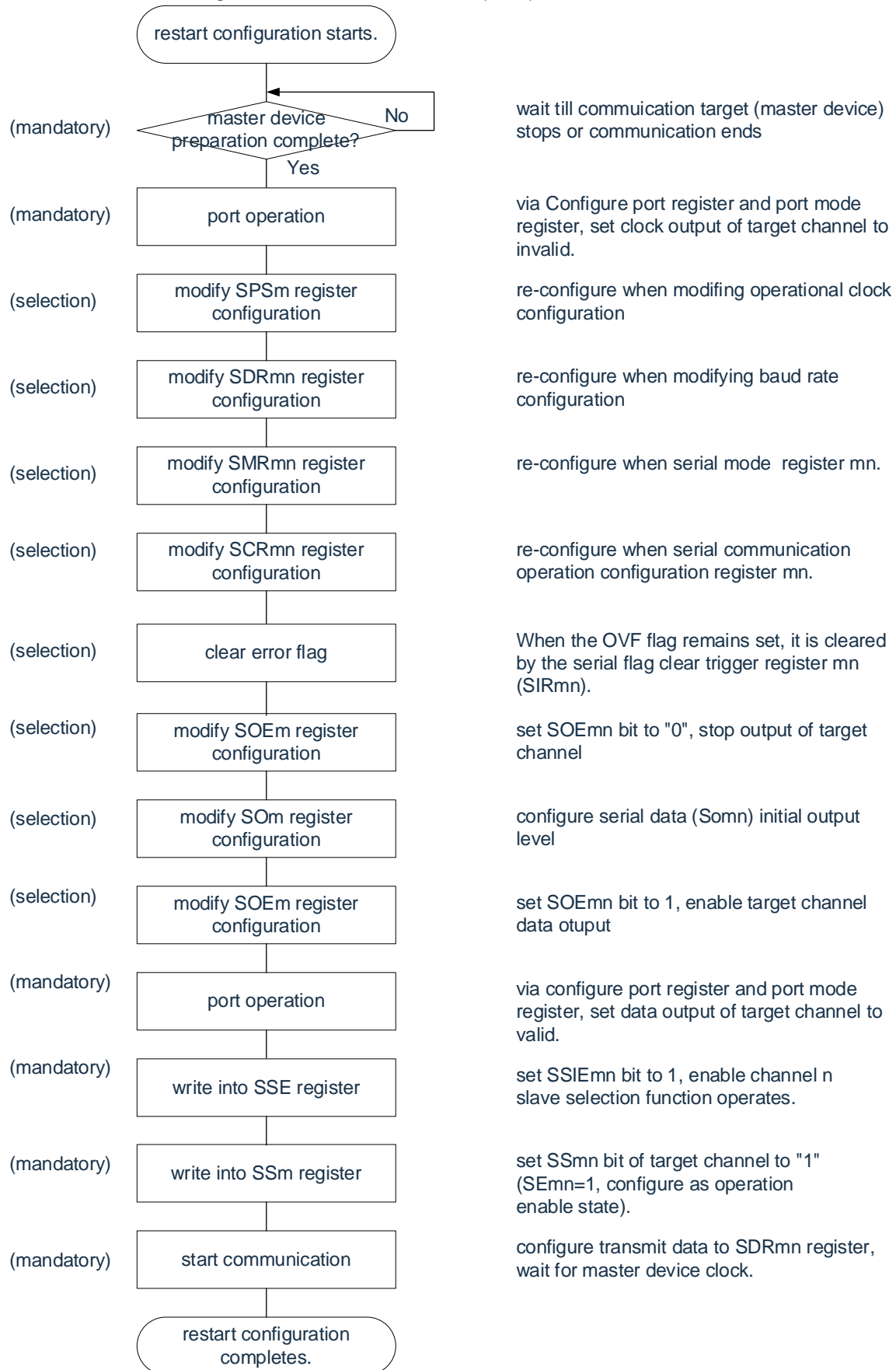
Figure 16-74: Abort step of the slave send



#### Remark:

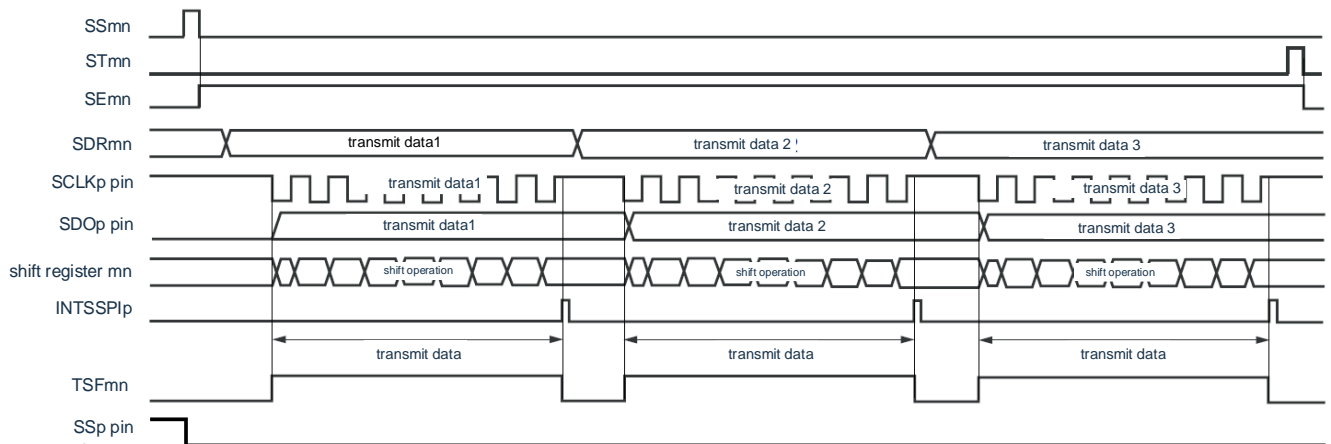
1. If PER0 is rewritten in the abort settings to stop providing the clock, the initial setting must not be restarted when the communication object (the master device) stops or when the communication ends.
2. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)

Figure 16-75: Restarts the setup step of the slave send



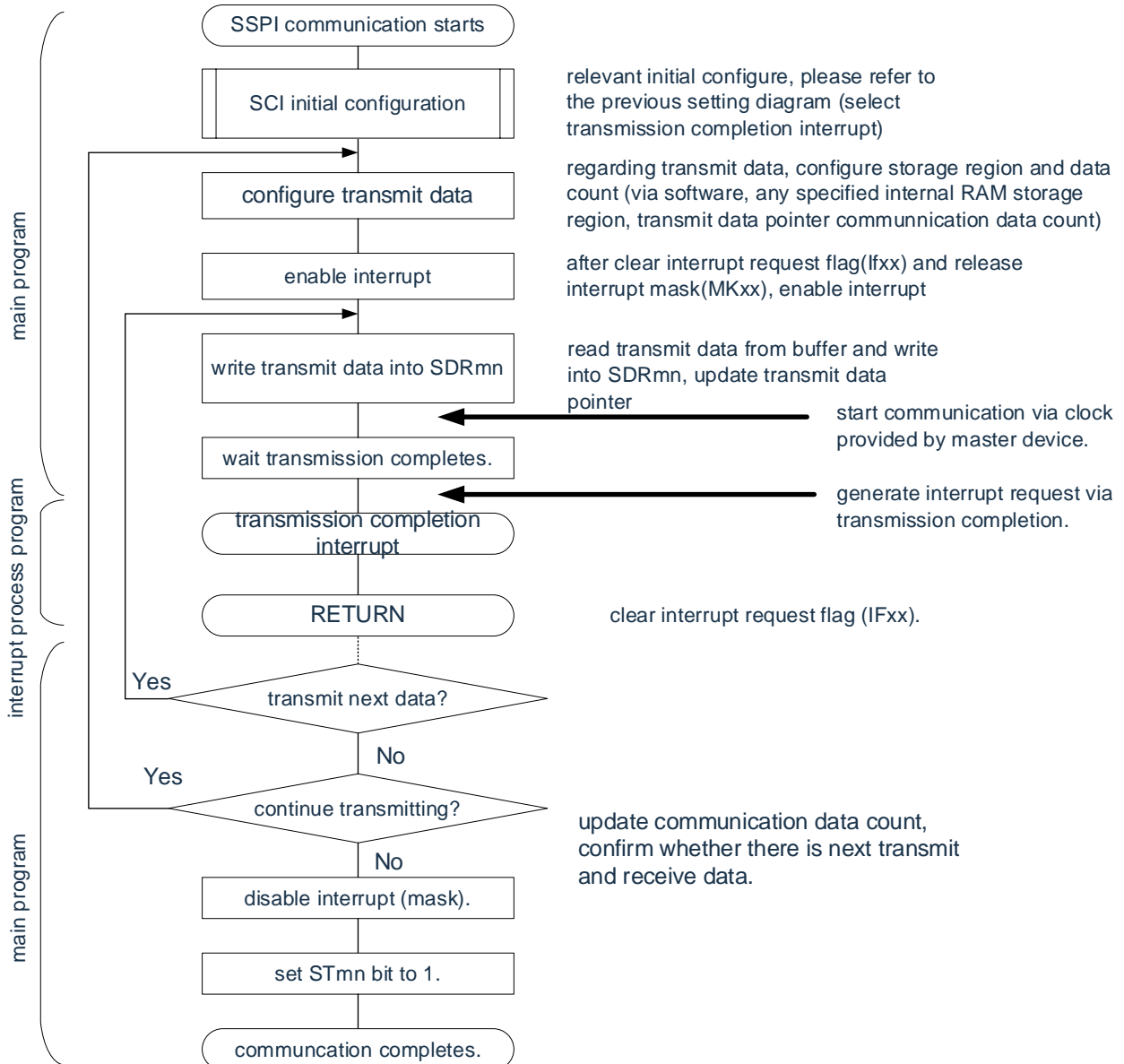
### (3) Process flow (single send mode)

Figure16-76: Timing diagram of a Slave send (single send mode) (type 1: DAPmn=0, CKPmn=0)



Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

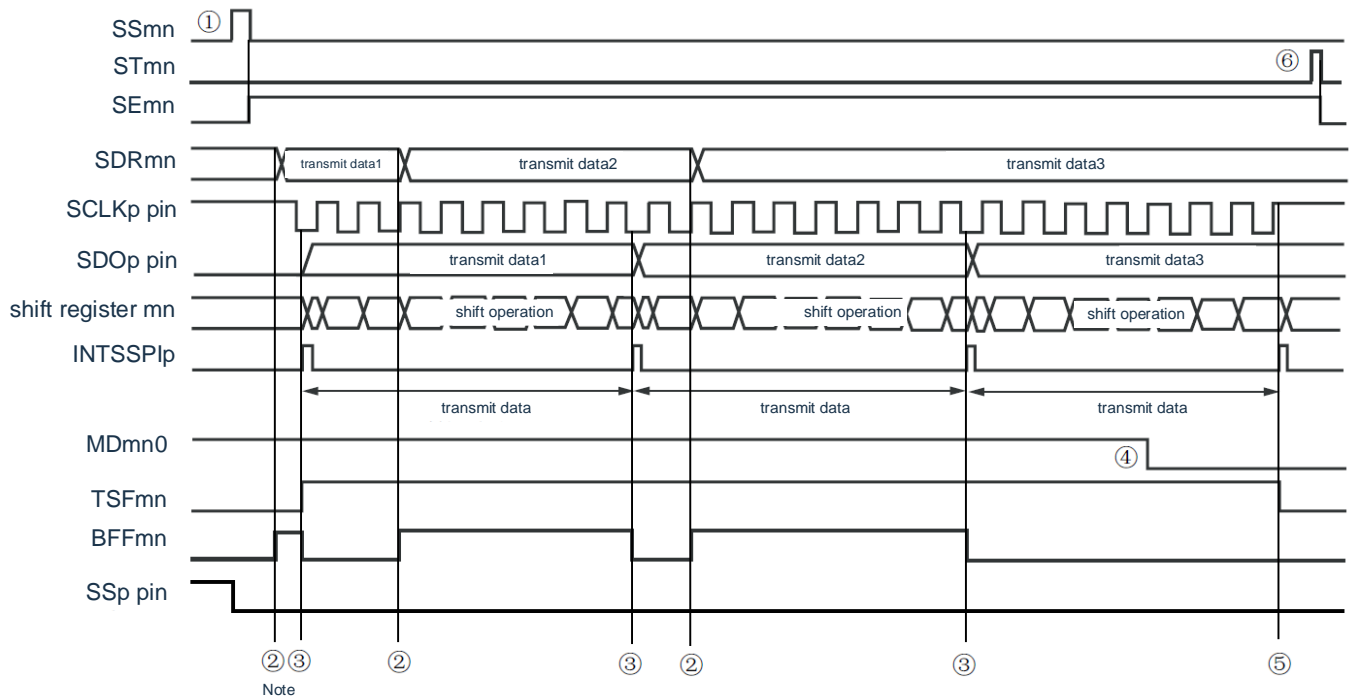
Figure 16-77: lowchart of slave send (single send mode)



Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

#### (4) Process flow (continuous send mode)

Figure 16-78: Timing diagram of Slave send (continuous send mode) (type 1: DAPmn= 0, CKPmn = 0)

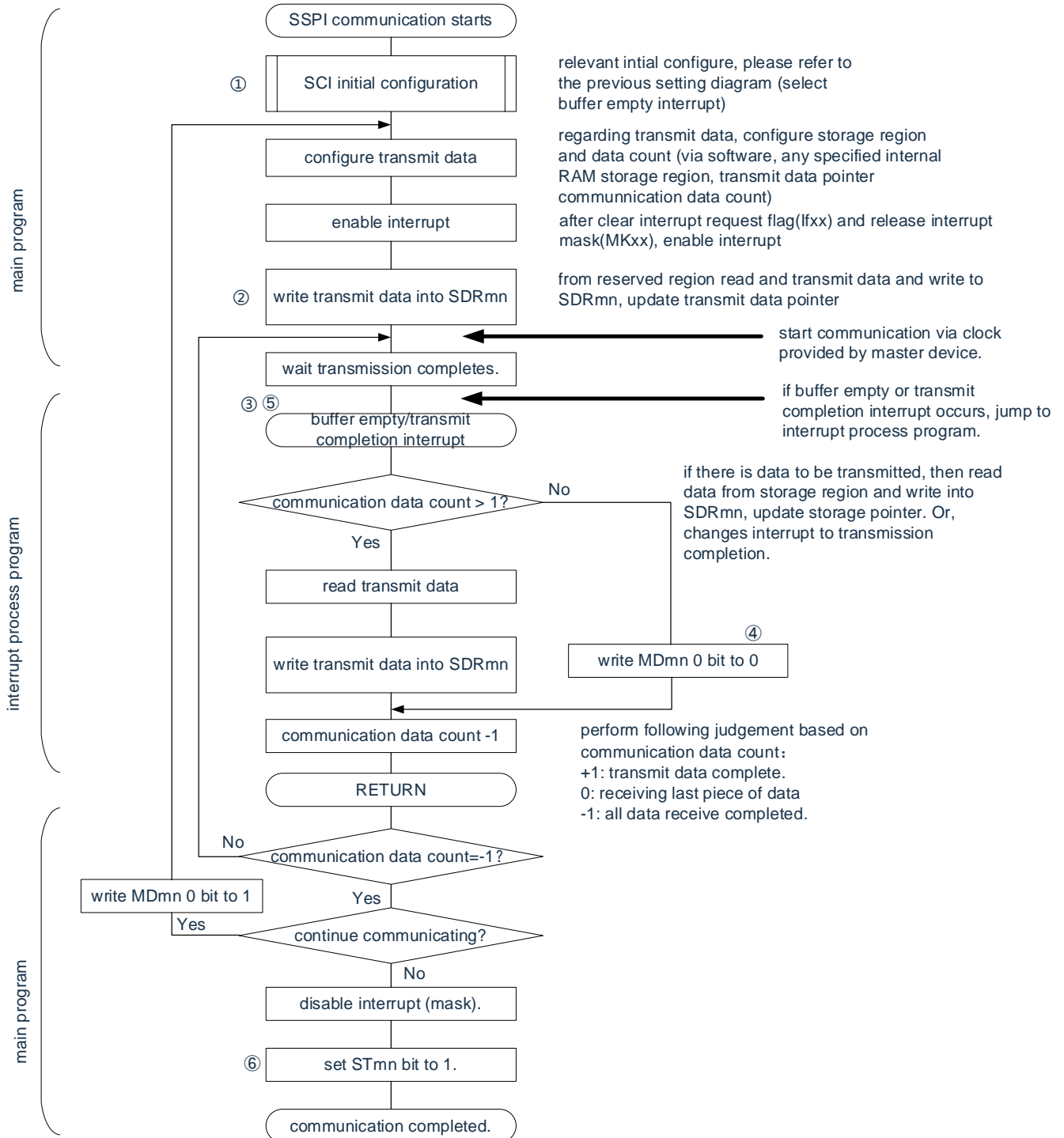


Note: If the BFFmn bit of the serial status register mn (SSRmn) is "1" (when valid data is saved in the serial data register mn (SDRmn)) is given When the SDRmn register writes the transmit data, it rewrites the send data.

Notice: MDmn0 bit of the serial mode register mn (SMRmn) can be rewritten even in operation. However, it must be overwritten before the last bit can be transmitted.

Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

Figure 16-79: Flowchart of slave send (continuous send mode)



#### Remark:

- (1) ~ (6) in the figure corresponds to (1) ~ (6) in "Figure 16-78 Timing diagram of Slave send (continuous send mode)".
- m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)

## 16.6.2 Slave reception

Slave reception refers to the operation of this product receiving data from other devices in the state of transmitting clocks from other devices.

Slave selection input function	SSPImn
Object channels	Channel n of SCIm
Pin used	SCLKmn, SDImn, SSImn
Interrupt	INTST0, INTSR0, INTST1, INTSR1, INTST2, INTSR2 Limited to end-of-transmit interrupts (buffer null interrupts are prohibited).
Error detection flags	There are only overflow error detection flags(OVFmn).
Transferred data length	7-16 bits
Transfer rate	Max.F <sub>MCK</sub> /6[Hz] <sup>Note 1,2</sup>
Data phase	It can be selected by the DAPmn bit of the SCRmn register. • DAPmn=0: Starts data output when the serial clock starts running. • DAPmn=1: Starts the data output half a clock before the serial clock starts running.
Clock phase	It can be selected by the CKPmn bit of the SCRmn register. • CKPmn=0: positive phase • CKPmn=1: inverted phase
Data direction	MSB preferred or LSB preferred
Slave selection input function	You can select the run of the Slave selection input function.

Note 1: Because it is used after internal sampling of the external serial clock at the SCLK00 pin input, the maximum transfer rate is F<sub>MCK</sub>/6[Hz].

Note 2: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

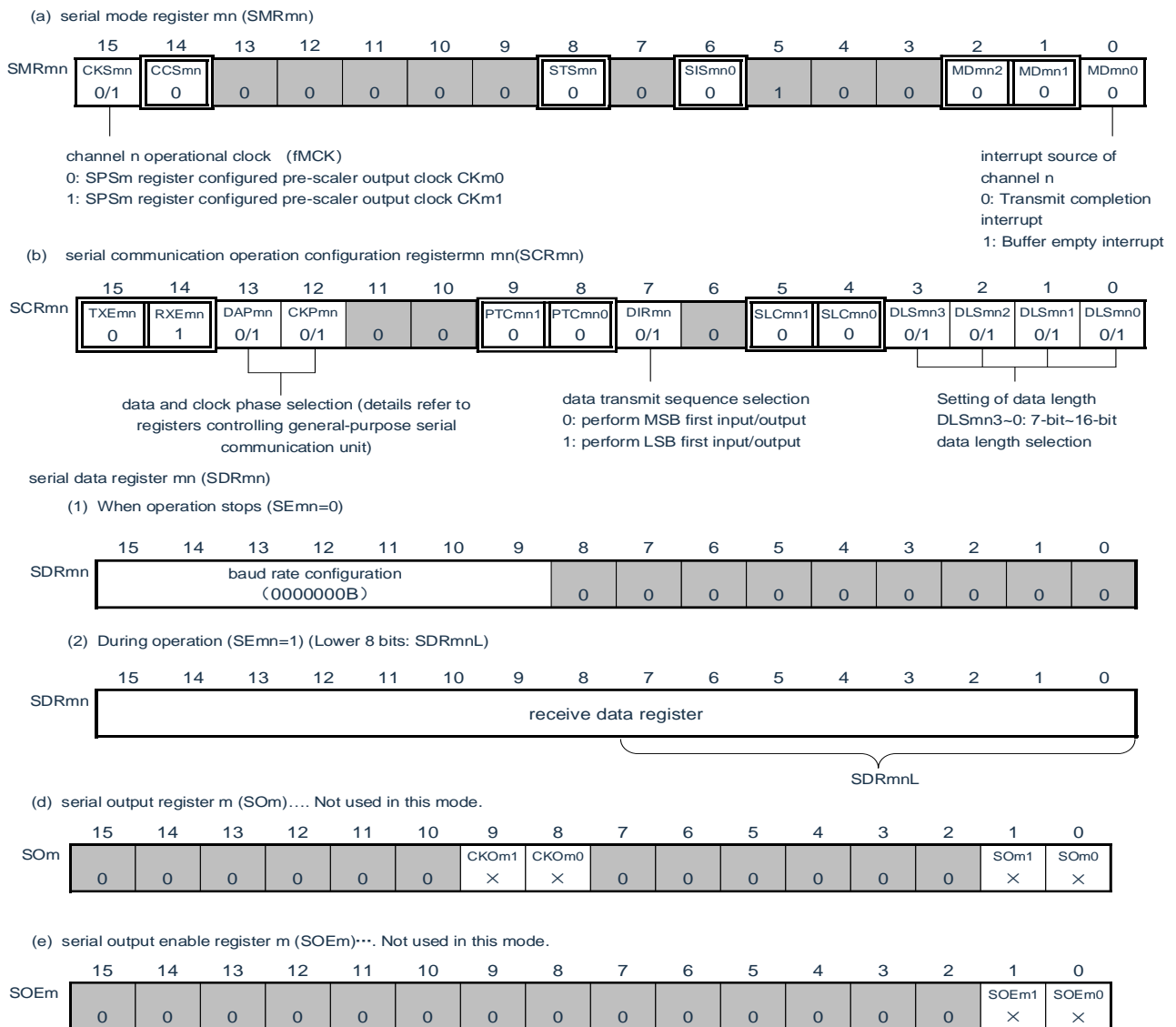
Remark:

1. F<sub>MCK</sub>: The operating clock frequency of the object channel
2. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)



## (1) Register setting

Figure16-80: Slave Selection Input Function (SSPImn) Example of register setting content when slave receives  
(1/2)



### Remark:

- m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)
- : Fixed setting in slave receive mode.■: setting disabled(initial value).  
x: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).  
0/1: Set "0" or "1" according to the user's purpose.

Figure16-81: Slave Selection Input Function (SSPlmn) Example of register setting content when slave receives  
(2/2)

(f) serial channel start register m (SSm) .... Only set bit of target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SSm1 X	SSm0 0/1

(g) Slave Select Function Enable Register (SSE) ..... This is the control of the SSPlmn pin of the SSPlmn slave channel (channel n of unit m).

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSEm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SSIEm1 0/1	SSIEm0 0/1

0: SS00 pin input invalid  
1: SS00 pin input valid

Remark:

1. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)
2. ☐ : Fixed setting in slave receive mode. ☐ : setting disabled(initial value).  
x: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).  
0/1: Set "0" or "1" according to the user's purpose.

## (2) Procedure

Figure16-82: Initial setup step of slave reception

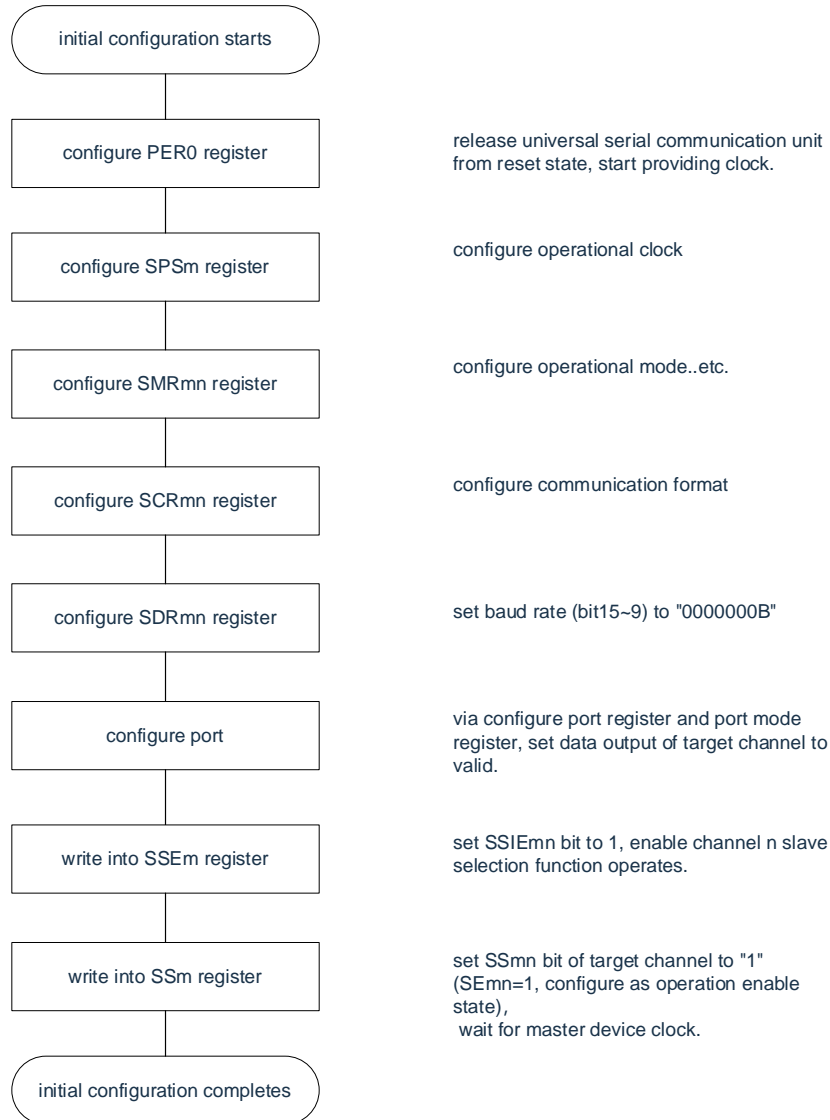
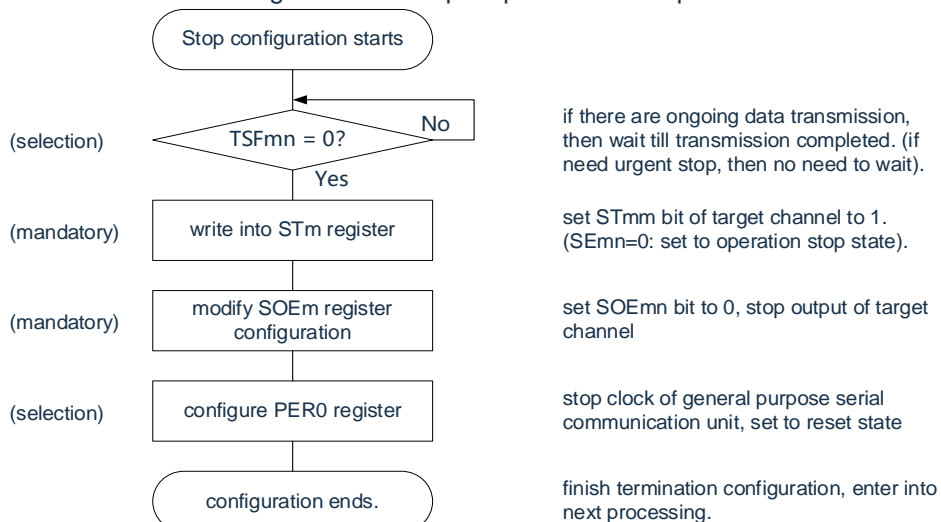
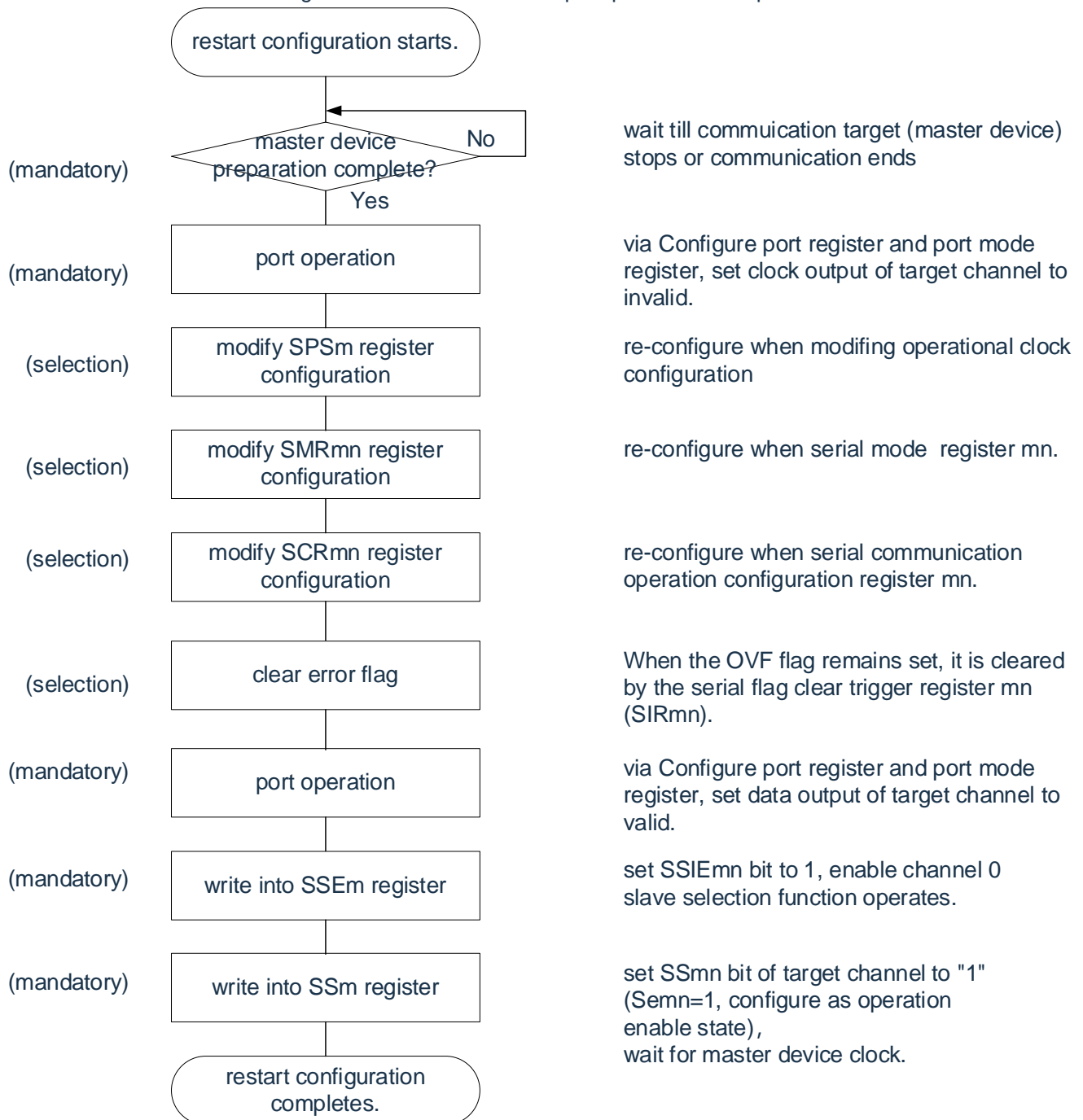


Figure16-83: Stop step of slave reception



Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

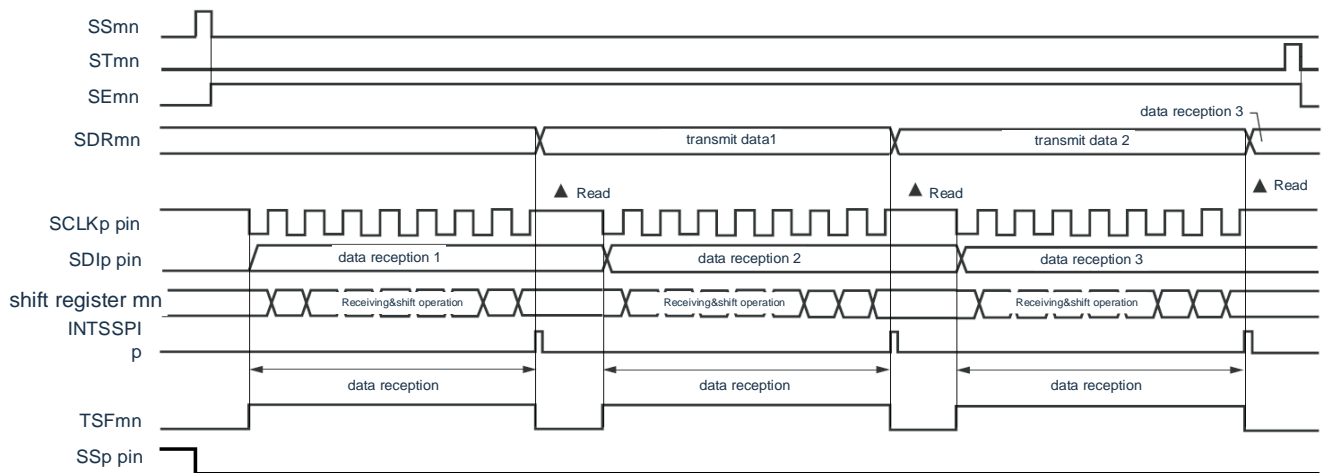
Figure16-84: Restart the setup step of Slave reception



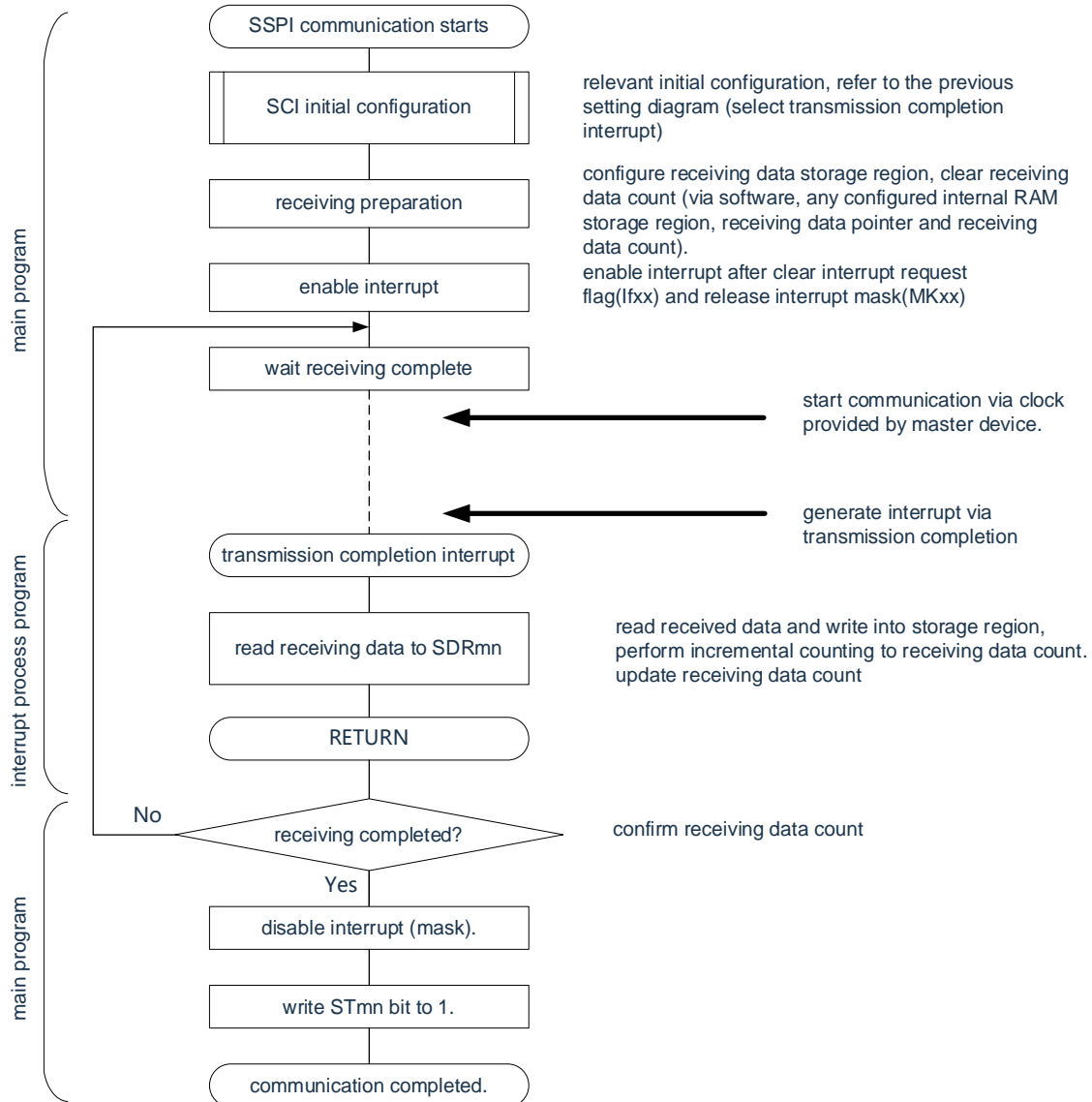
Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

### (3) Process flow (single receive mode)

Figure16-85: Timing diagram of slave receive (single receive mode) (type 1: DAPmn= 0, CKPmn = 0)



Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

**Figure16-86: Flowchart of slave receive (single receive mode)**


### 16.6.3 Slave transmission and reception

Slave transmission and reception refers to the operation of data transmission and reception of this product and other devices in the state of input transmission clock from other devices.

Slave selection input function	SSPImn
Object channels	Channel n of SCIm
Pin used	SCLKmn, SDOmn, SDImn, SSImn
Interrupt	INTST0, INTSR0, INTST1, INTSR1, INTST2, INTSR2 Selectable transmission end interrupt (single transmission mode) or buffer air interrupt (continuous transmission mode).
Error detection flags	There are only overflow error detection flags(OVFmn).
Transferred data length	7-16 bits
Transfer rate	Max.F <sub>MCK</sub> /6[Hz] <sup>Note 1,2</sup>
Data phase	It can be selected by the DAPmn bit of the SCRmn register. • DAPmn=0: Starts data output when the serial clock starts running. • DAPmn=1: Starts the data output half a clock before the serial clock starts running.
Clock phase	It can be selected by the CKPmn bit of the SCRmn register. • CKPmn=0: positive phase • CKPmn=1: inverted phase
Data direction	MSB preferred or LSB preferred
Slave selection input function	You can select the run of the Slave selection input function.

Note 1: Because it is used after internal sampling of the external serial clock at the SCLK00 pin input, the maximum transfer rate isF<sub>MCK</sub>/6[Hz].

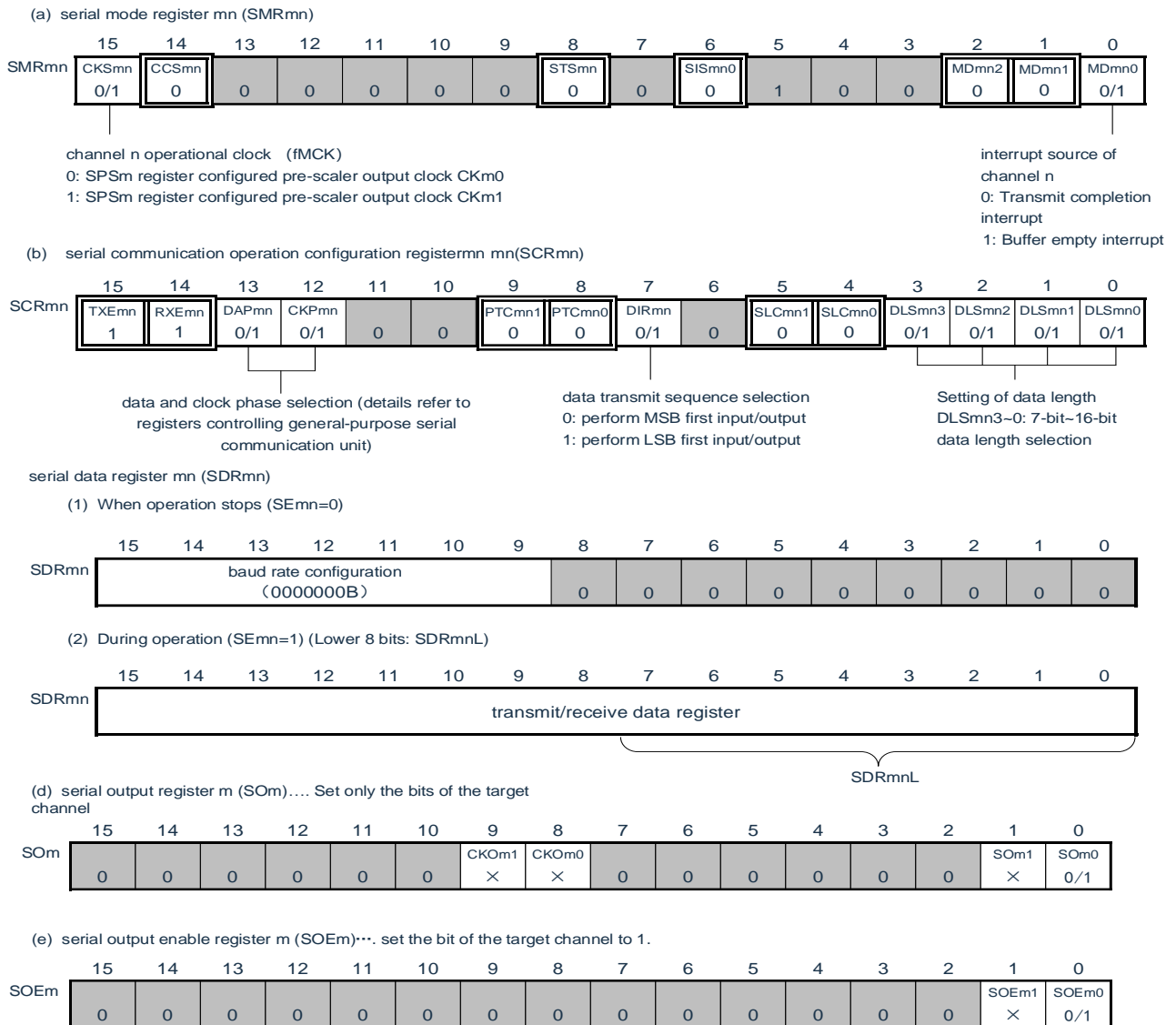
Note 2: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

Remark:

1. F<sub>MCK</sub>: The operating clock frequency of the object channel
2. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)

### (1) Register setting

Figure 16-87: Slave Selection Input Function (SSPImn) Example of register setting content when slave send and receive (1/2)



Notice: The send data must be set to the SDRmn register before the master device starts outputting the clock.

Remark:

1. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)
2.  : Fixed setting in slave receive mode.  : setting disabled(initial value).  
x: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).  
0/1: Set "0" or "1" according to the user's purpose.



Figure16-87: Slave Selection Input Function (SSPlmn) Example of register setting content when slave send and receive (2/2)

(f) serial channel start register m (SSm) .... Only set bit of target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SSm1 X	SSm0 0/1



(g) Slave Select Function Enable Register (SSE) ..... This is the control of the SSPlmn pin of the SSPlmn slave channel (channel n of unit m).

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSEm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SSIEm1 0/1	SSIEm0 0/1

0: SS00 pin input invalid  
1: SS00 pin input valid

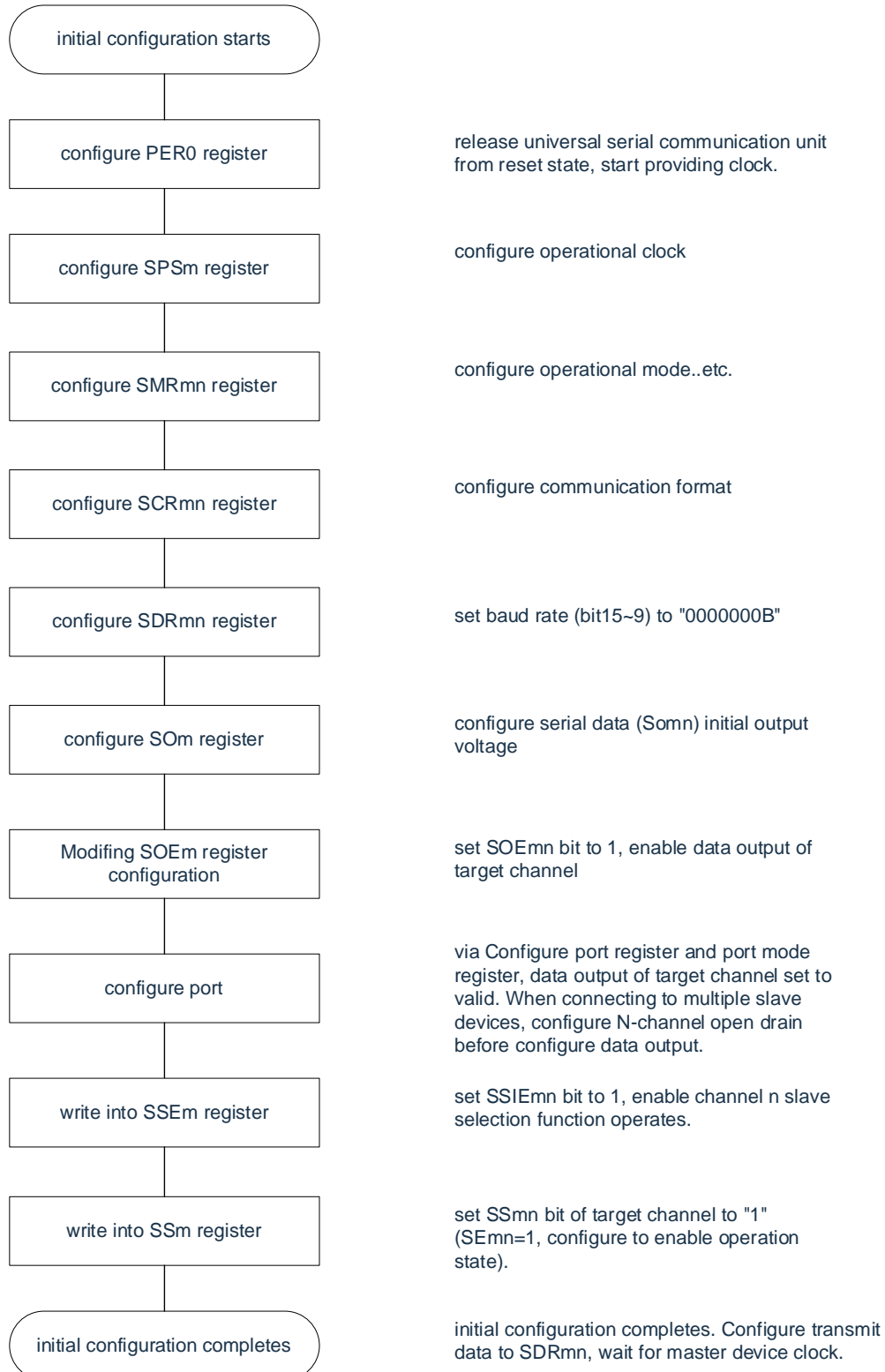
Notice: The send data must be set to the SDRmn register before the master device starts outputting the clock.

Remark:

1. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)
2.  : Fixed setting in slave receive mode.  : setting disabled(initial value).  
x: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).  
0/1: Set "0" or "1" according to the user's purpose.

## (2) Procedure

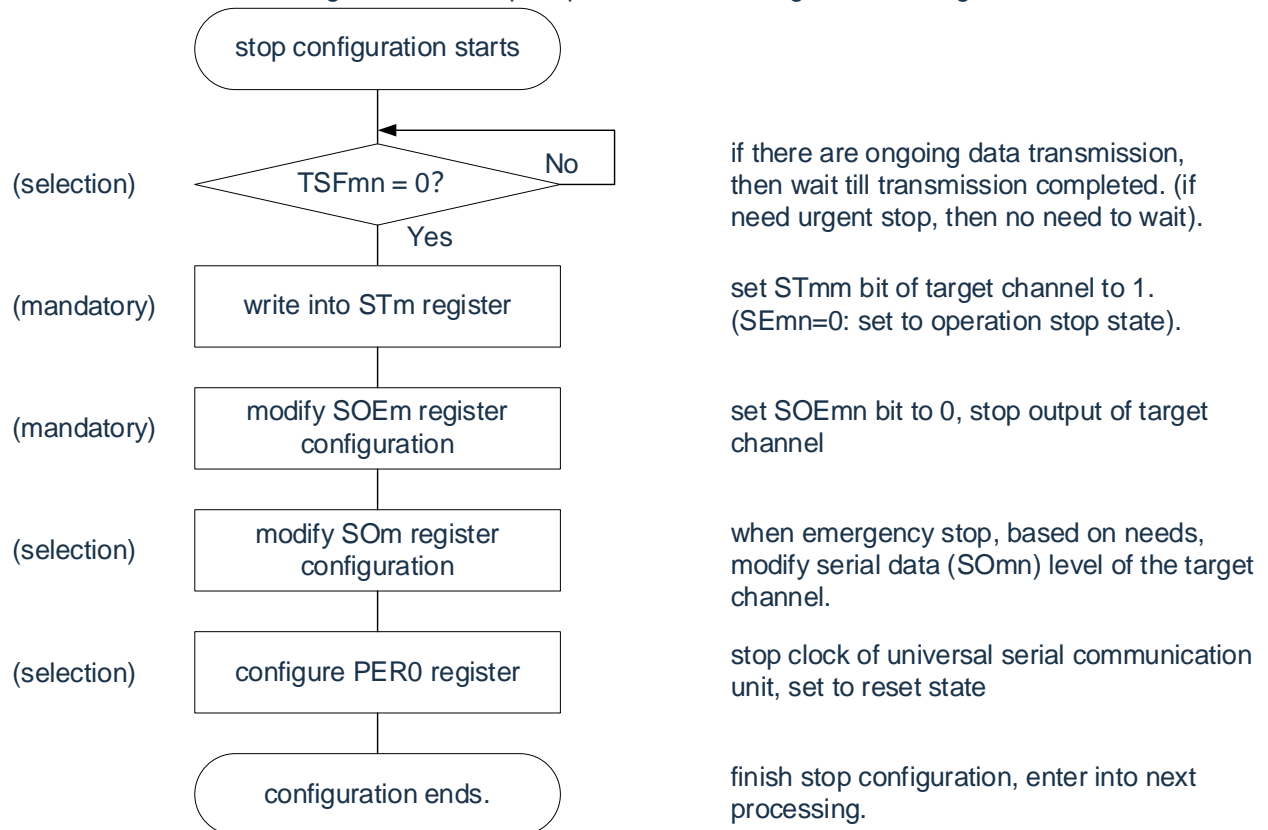
Figure16-88: Initial setup steps for slave sending and receiving



Notice: The send data must be set to the SDRmn register before the master device starts outputting the clock.

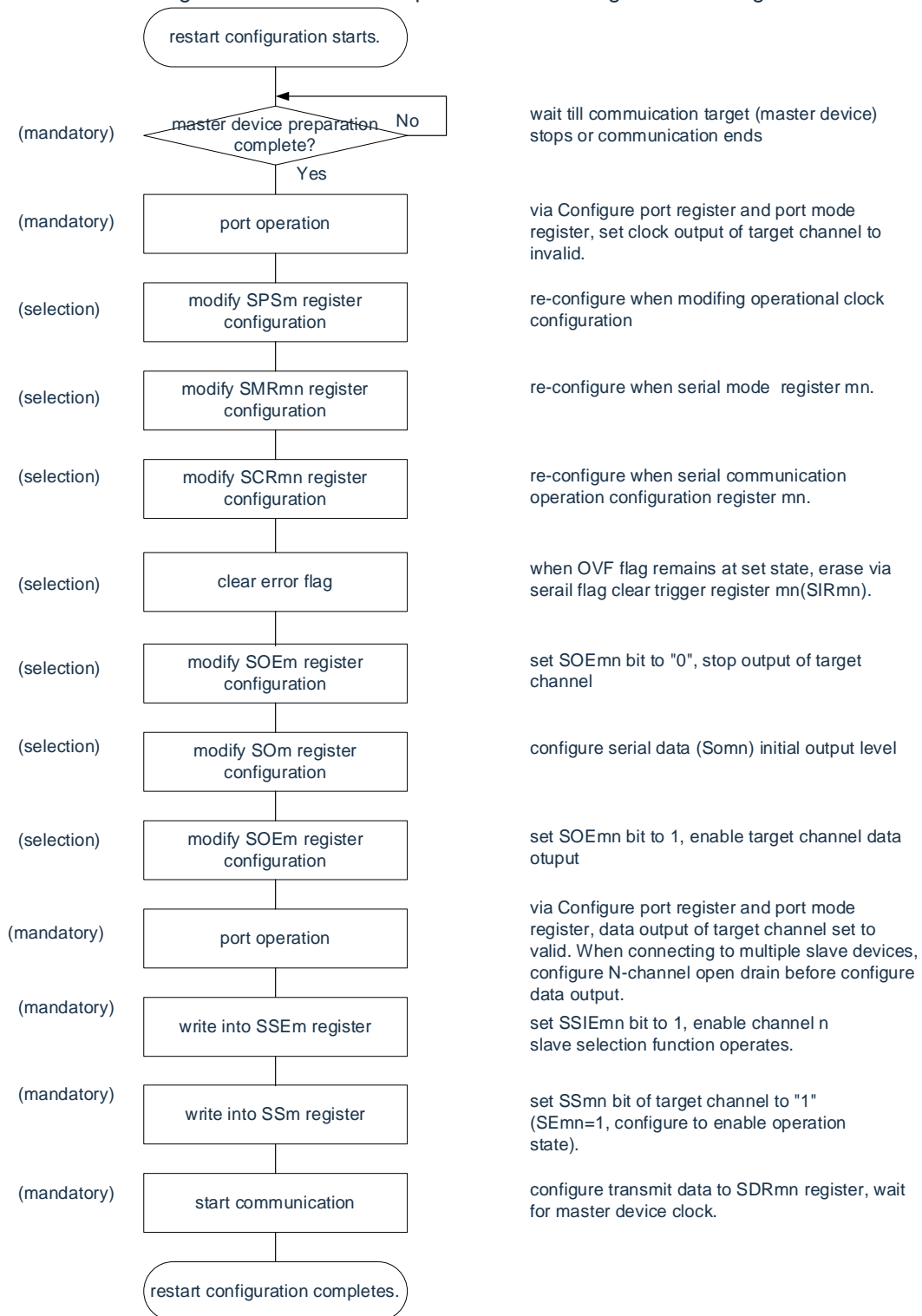
Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

Figure 16-89: Stop steps for slave sending and receiving



Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

Figure 16-90: Restart steps for slave sending and receiving

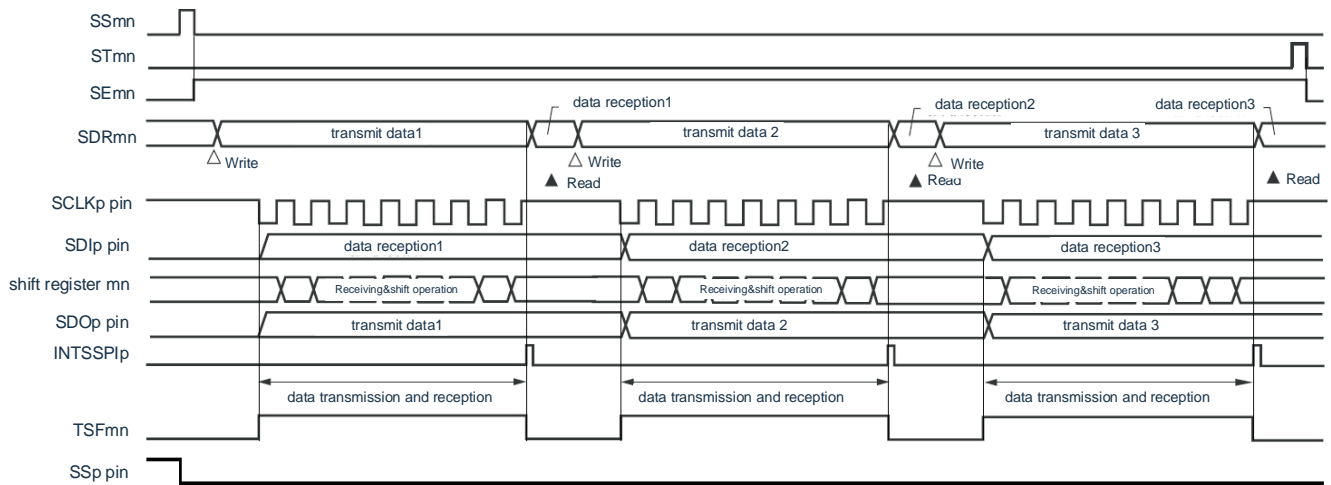


#### Notice:

1. The send data must be set to the SDRmn register before the master device starts outputting the clock.
2. If PER0 is rewritten in the abort settings to stop providing the clock, the initial setting must not be restarted when the communication object (the master device) stops or when the communication ends.

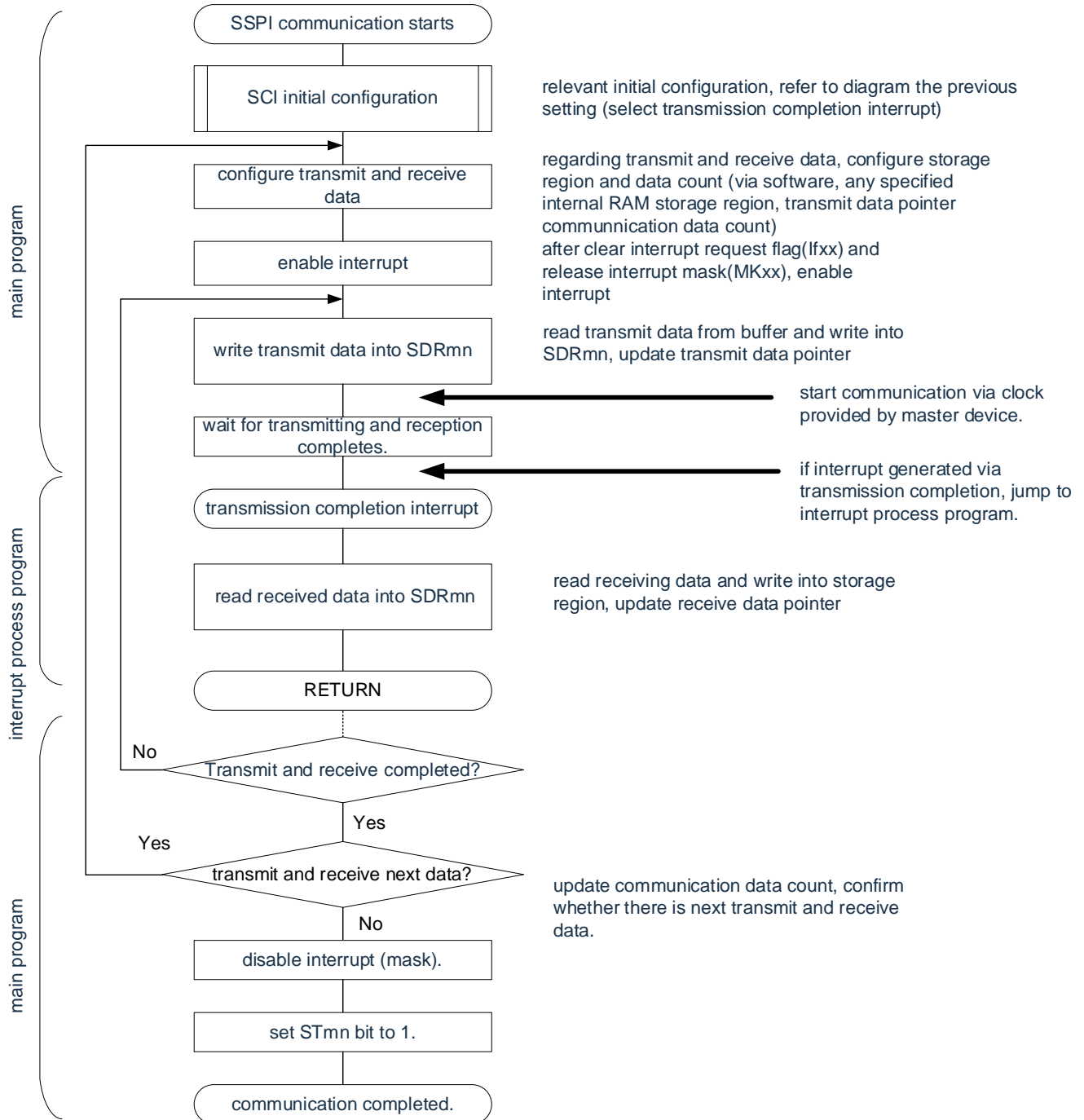
### (3) Process flow (single send and receive mode)

Figure16-91: Timing diagram of Slave transmit and receive (single transmit and receive mode)



Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

Figure16-92: Flowchart of slave send and receive (single send and receive mode)

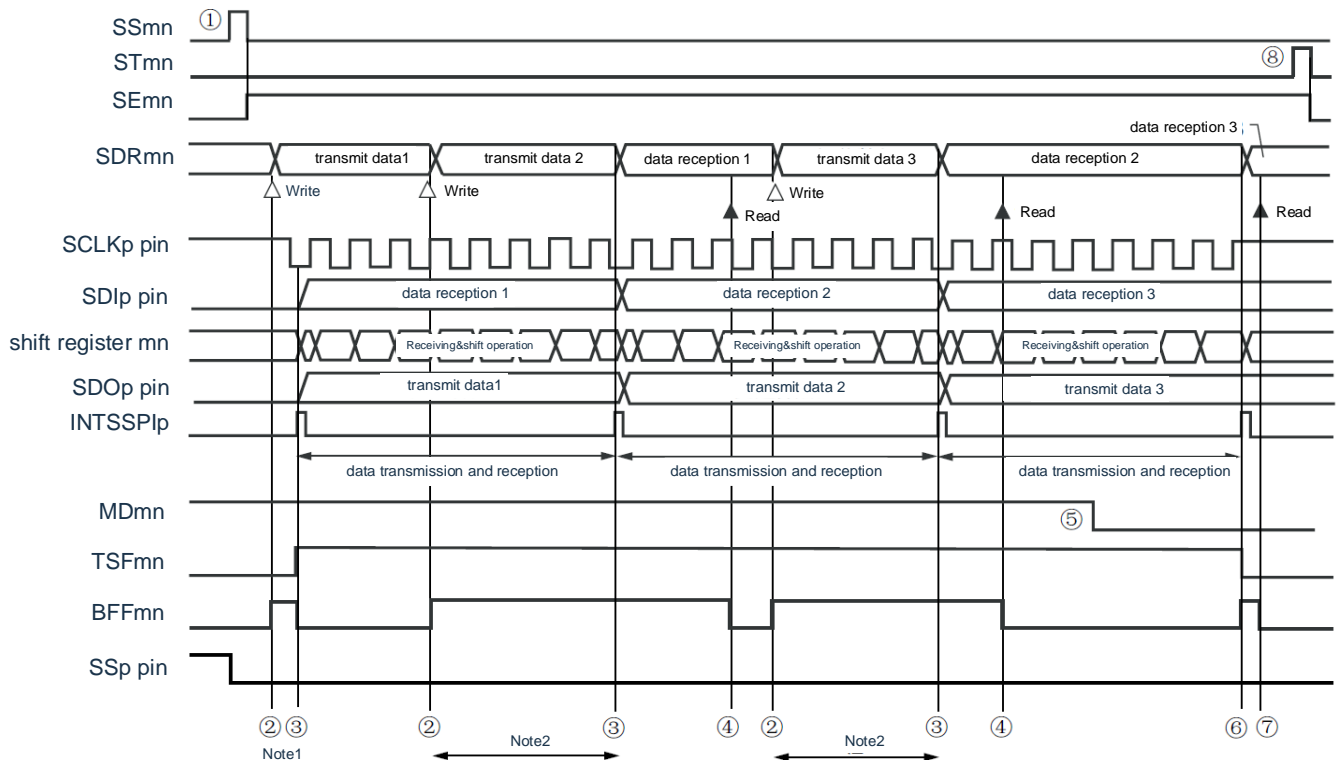


Notice: The send data must be set to the SDR<sub>mn</sub> register before the master device starts outputting the clock.

Remark: m: unit number (m=0, 1, 2) n: channel number (n=0, 1) p: SSPI number (p=00, 01, 10, 11, 20, 21)

#### (4) Process flow (continuous send and receive mode)

Figure16-93: Timing diagram of Slave send and receive (continuous transmit and receive mode) (type 1: DAPmn=0, CKPmn=0)



Note 1: If the BFFmn bit of serial status register mn(SSRmn) is "1" during the period (valid data is saved in serial data register mn(SDRmn(when writing transmit data to the SDRmn register, rewrite the send data)).

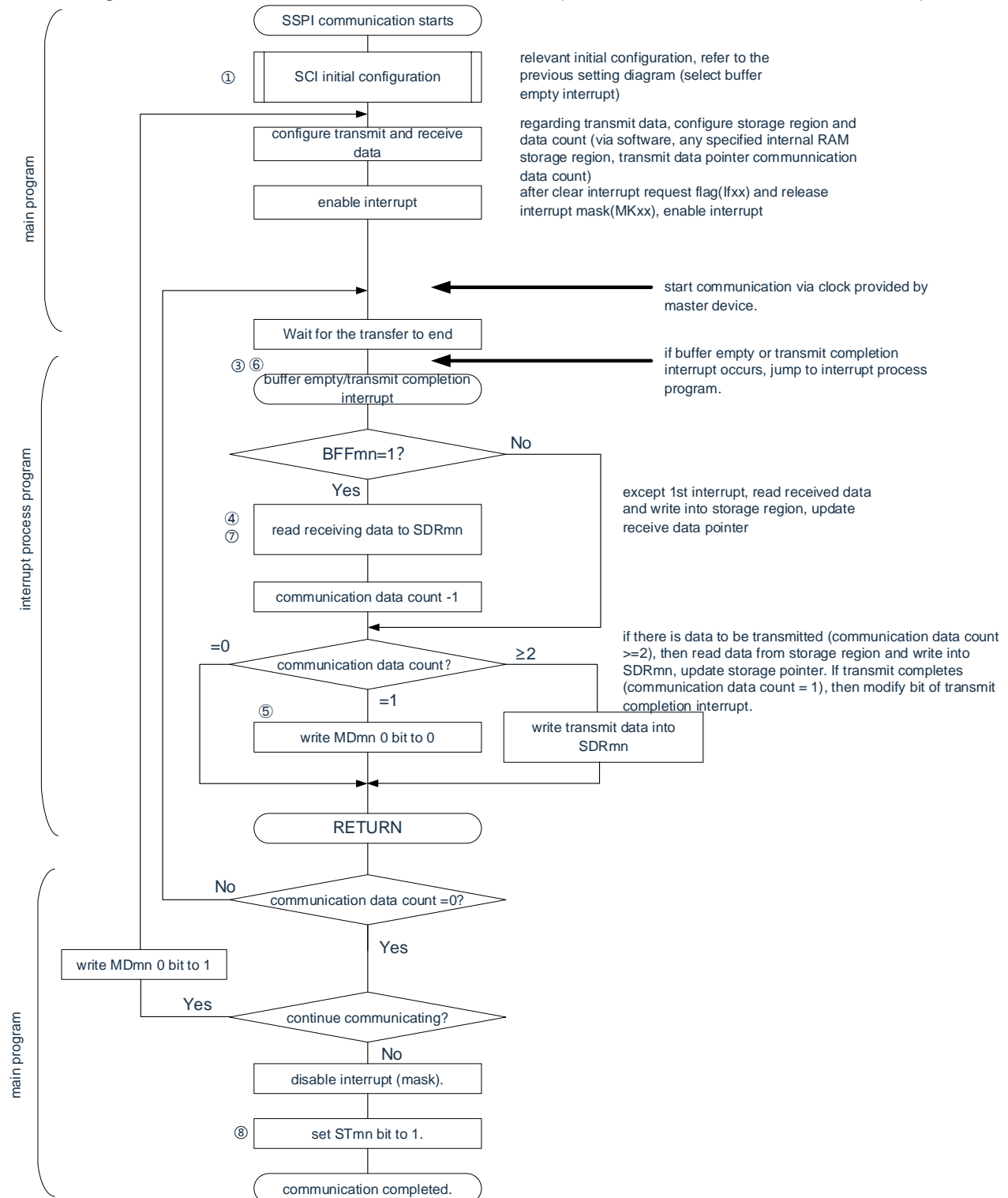
Note 2: If the SDRmn register is read during this period, the data can be read and sent. At this point, the transfer run is not affected.

Notice: MDmn0 bit of the serial mode register mn (SMRmn) can be rewritten even in operation. However, in order to catch the end-of-transmission interruption of the last sent data, it must be rewritten before the last bit of transmission begins.

Remark:

- (1) ~ (8) in the figure corresponds to "Figure16-94 Flowchart of Slave send and receive (continuous transmit and receive mode)".
- m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)

Figure16-94: Flowchart of Slave send and receive (continuous transmit and receive mode)



Notice: The send data must be set to the SDRmn register before the master device starts outputting the clock.

Remark:

- (1) to (8) in the figure corresponds to (1) to (8) in the “Figure16-93 Timing diagram of Slave send and receive (continuous transmit and receive mode)”.
- m: unit number(m=0, 1, 2)n: channel number(n=0, 1)p: SSPI number(p=00, 01, 10, 11, 20, 21)



## 16.6.4 Calculation of transmission clock frequency

The transmit clock frequency of the slave select input function (SSPImn) communication can be calculated using the following calculation equation.

(1) Slave

$$(\text{Transmit Clock Frequency}) = \{ \text{Serial Clock (SCLK) Frequency Rate Provided by the Master Device} \}^{\text{Note}} [\text{Hz}]$$

Note: The maximum allowable transmit clock frequency is  $F_{MCK}/6$ .

Table16-3: Slave Selection Input Function Running Clock Selection

SMRmn Register	SPSm register								Running Clock ( $F_{MCK}$ ) <sup>Note</sup>	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		$F_{CLK}=32\text{MHz}$ runtime
0	X	X	X	X	0	0	0	0	$F_{CLK}$	32MHz
	X	X	X	X	0	0	0	1	$F_{CLK}/2$	16MHz
	X	X	X	X	0	0	1	0	$F_{CLK}/2^2$	8MHz
	X	X	X	X	0	0	1	1	$F_{CLK}/2^3$	4MHz
	X	X	X	X	0	1	0	0	$F_{CLK}/2^4$	2MHz
	X	X	X	X	0	1	0	1	$F_{CLK}/2^5$	1MHz
	X	X	X	X	0	1	1	0	$F_{CLK}/2^6$	500KHz
	X	X	X	X	0	1	1	1	$F_{CLK}/2^7$	250KHz
	X	X	X	X	1	0	0	0	$F_{CLK}/2^8$	125KHz
	X	X	X	X	1	0	0	1	$F_{CLK}/2^9$	62.5KHz
	X	X	X	X	1	0	1	0	$F_{CLK}/2^{10}$	31.25KHz
	X	X	X	X	1	0	1	1	$F_{CLK}/2^{11}$	15.63KHz
	X	X	X	X	1	1	0	0	$F_{CLK}/2^{12}$	7.81KHz
	X	X	X	X	1	1	0	1	$F_{CLK}/2^{13}$	3.91KHz
	X	X	X	X	1	1	1	0	$F_{CLK}/2^{14}$	1.95KHz
	X	X	X	X	1	1	1	1	$F_{CLK}/2^{15}$	977Hz

Note: When you change the clock selected as FCLK (change the value of the system clock control register (CKC)), you must stop the operation of the general-purpos serial communication unit (SCI) (serial channel stop register m (STm)=000FH) after making changes.

Remark:

1. X: Ignore
2. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)

## 16.6.5 Procedure for handling errors during clock-synchronous serial communication with the slave selection input function

The processing steps for errors that occur during clock synchronization serial communication with the slave select input function are shown in Figure16-95.

Figure16-95: Processing steps when the overflow error occurs

Software operation	Hardware status	Remark
Read the serial data register mn mn(SDRmn).	→ The BFFmn bit of the SSRmn register is "0" and the channel n is in a receiver state.	This is to prevent an overflow error from occurring to end the next receipt during error handling. next receipt during error handling.
Read the serial status register mn (SSRmn).	-	The type of error is determined and the read value is used to clear the error flag.
Clear trigger register mn for serial flag (SDIRmn) Write "1".	► Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors in the read operation can only be cleared.

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

## 16.7 Operation of UART (UART0~UART2) communication

This is an asynchronous function using two lines: serial data transmission (TxD) and serial data reception (RxD) lines. By using these two communication lines, data is sent and received asynchronously (using the internal baud rate) with other communicating parties by data frame (consisting of start bits, data, parity bits, and stop bits). Full-duplex asynchronous UART communication can be achieved by using two channels dedicated for transmit (even channel) and receive dedicated (odd channel).

[Data transmission and reception]

- 7-bit, 8-bit, 9-bit or 16-bit data length<sup>Note</sup>
- MSB/LSB preferred option
- Level setting for sending and receiving data (select whether the level is reversed)
- Parity bit appending and parity check functions
- Stop bit attachment, stop bit detection function

[Interrupt function]

- Transfer end interrupt, buffer empty interrupt
- Error interrupts caused by frame errors, parity errors, and overflow errors

[Error detection flag]

- Framing error, parity error, or overrun error

UART0 uses channel 0 and channel 1 of SCI0.

UART1 uses channel 0 and channel 1 of SCI1.

UART2 uses channel 0 and channel 1 of SCI2.

Each channel arbitrarily selects one function to use, except for the selected function, other functions cannot be run.

For example, when using UART0 for channel 0 and channel 1 of unit m, SSPI00 and IIC01 cannot be used.

Note: When used as a UART, the sender (even channels) and receiver (odd channels) can only be used for UART.

UART has the following 2 kinds of communication operation:

- UART transmission (refer to 16.7.1)
- UART reception (refer to 16.7.2)

## 16.7.1 UART transmission

UART transmission is the operation of this product microcontroller to asynchronously send data to other devices.

An even number of the 2 channels used by UART is used for UART transmission.

UART	UART0	UART1	UART2
Object channels	Channel 0 for SCI0	Channel 0 for SCI1	Channel 0 for SCI2
Pin used	TxD0	TxD1	TxD2
Interrupt	INTST0	INTST1	INTST2
	Selectable transmission end interrupt (single transmission mode) or buffer air interrupt (continuous transmission mode).		
Error detection flags	None		
Transferred data length	7-bit, 8-bit, 9-bit or 16-bit		
Transfer rate	Max. $F_{MCK}/6$ [bps](SDRmn[15:9] $\geq 3$ ), Min. $F_{CLK}/(2 \times 2^{11} \times 128)$ [bps] <sup>Note</sup>		
Data phase	Positive-phase output (default: high). Inverting output (default: low).		
Parity bits	You can choose from the following: <ul style="list-style-type: none"> <li>• No parity bits.</li> <li>• Additional zero parity check.</li> <li>• Additional even parity check.</li> <li>• Additional odd parity check.</li> </ul>		
Stop bit	You can choose from the following: <ul style="list-style-type: none"> <li>• Add 1 bit.</li> <li>• Add 2 bits.</li> </ul>		
Data direction	MSB preferred or LSB preferred		

Note: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

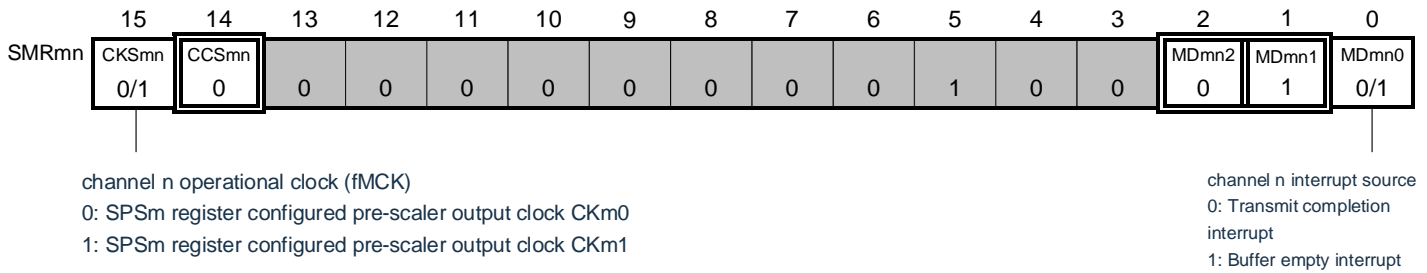
Remark:

1.  $F_{MCK}$ : The operating clock frequency of the object channel  
 $F_{CLK}$ : System clock frequency
2. m: unit number(m=0, 1, 2)n: channel number(n=0)

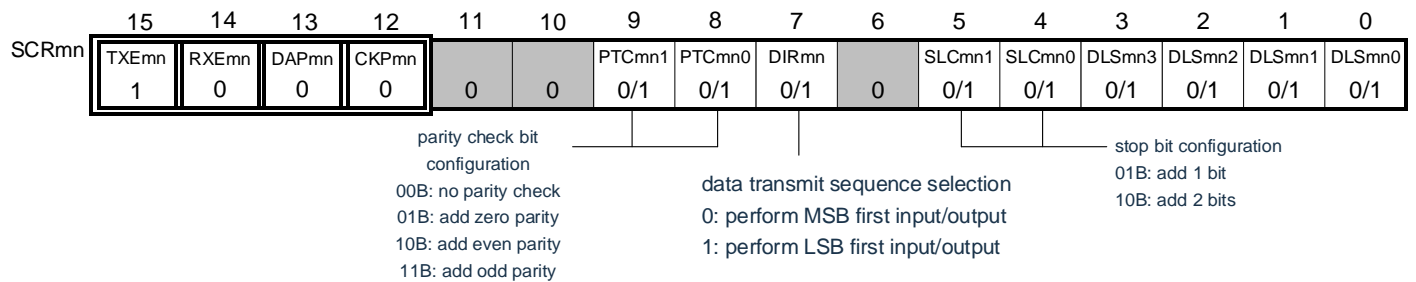
### (1) Register setting

Figure 16-96: Register setting content when UART is transmitted by UART (UART0~UART2) (1/2)

(a) serial mode register mn (SMRmn)

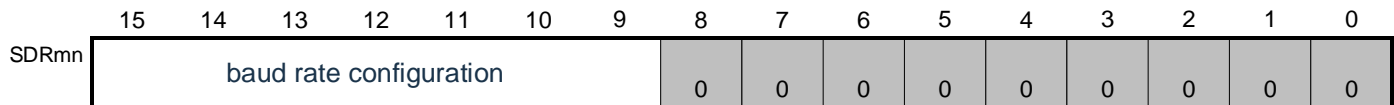


(b) serial communication operation configuration register mn (SCRmn)

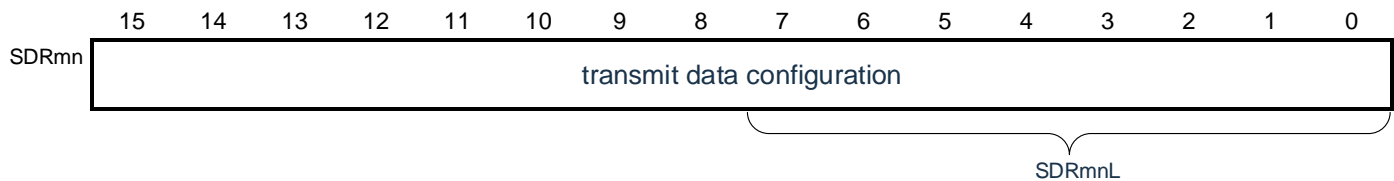


(c) Serial data register mn (SDRmn)

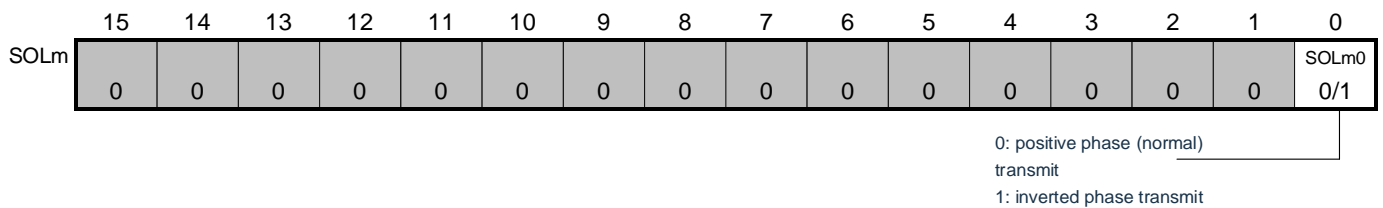
(1) When operation stops (SEmn=0)



(2) During operation (SEmn=1) (lower 8 bits: SDRmnL)



(d) serial output voltage register m (SOLm) .... Only configure bit of target channel.



#### Remark:

1. m: unit number(m=0, 1, 2)n: channel number(n=0)q: UART number(q=0~2)
2.  : Fixed setting in UART send mode.  : setting disabled(initial value).  
 x: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).  
 0/1: Set "0" or "1" according to the user's purpose.

**Figure16-96: Register setting content when UART is transmitted 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							CKOm1	CKOm0							SOM1	SOM0
	0	0	0	0	0	0	×	×	0	0	0	0	0	0	×	0/1 注

0: serial data output value as "0"  
 1: serial data output value as "1"

(f) serial output enable register m (SOEm).... Only set bit of target channel to "1".

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm															SOEm1	SOEm0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	×	0/1

(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															SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	×	0/1

**Note:** Before starting the transmission, when the SOLmn bit of the corresponding channel is "0", "1" must be set; When the SOLmn bit of the corresponding channel is "1", "0" must be set. During communication, the value changes depending on the communication data.

**Remark:**

- m: unit number(m=0, 1, 2)n: channel number(n=0)q: UART number(q=0~2)
- : Fixed setting in UART send mode.
  : setting disabled(initial value).  
 ×: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).  
 0/1: Set "0" or "1" according to the user's purpose.

## (2) Procedure

Figure 16-97: UART transmission

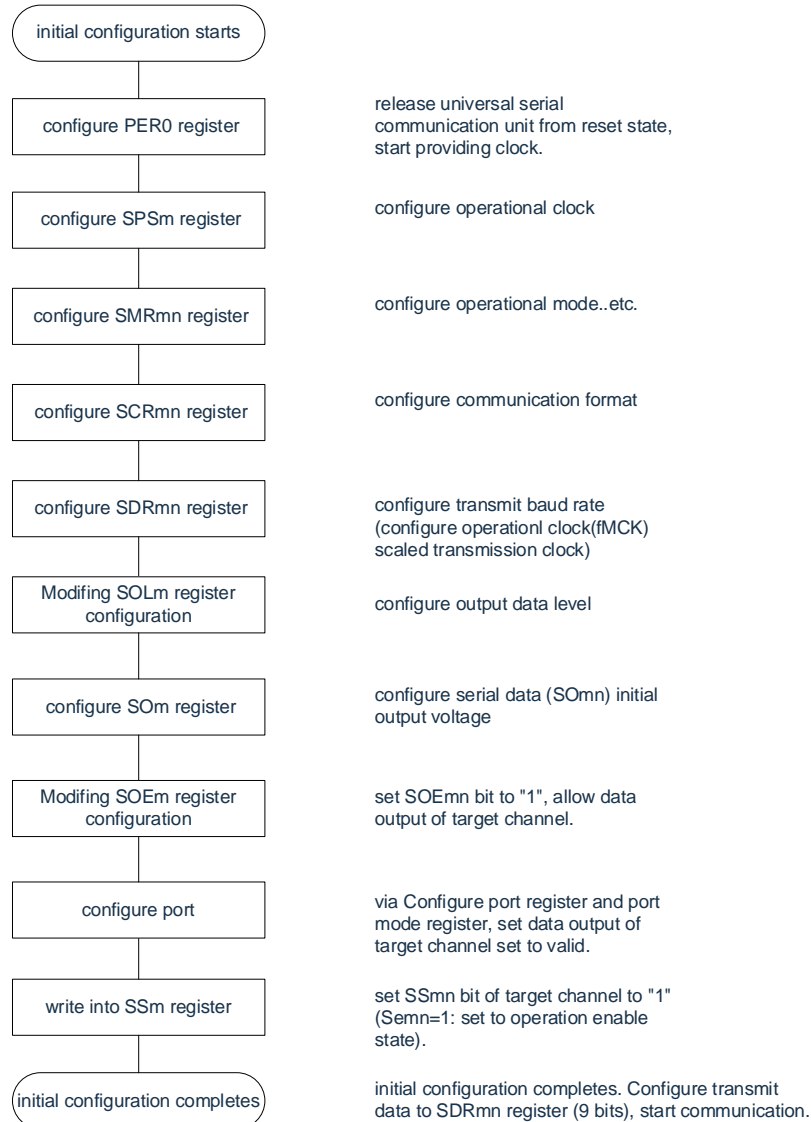


Figure 16-98: Stop steps sent by the UART

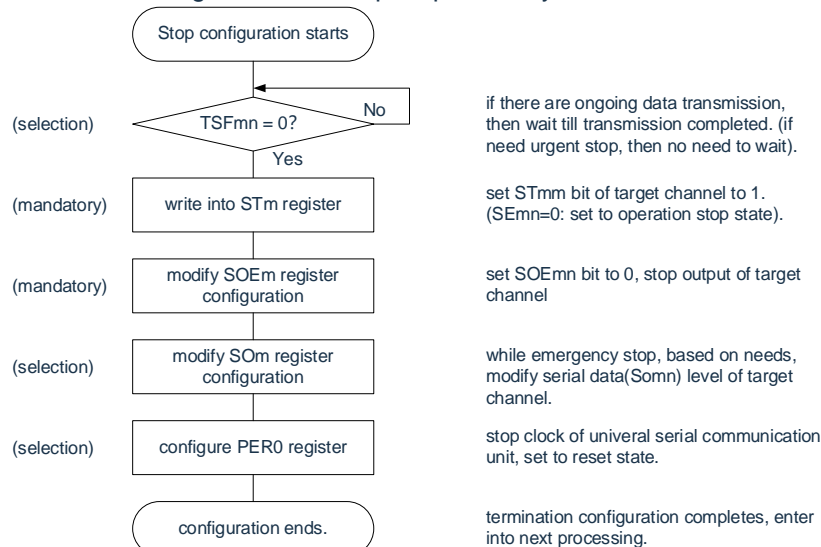
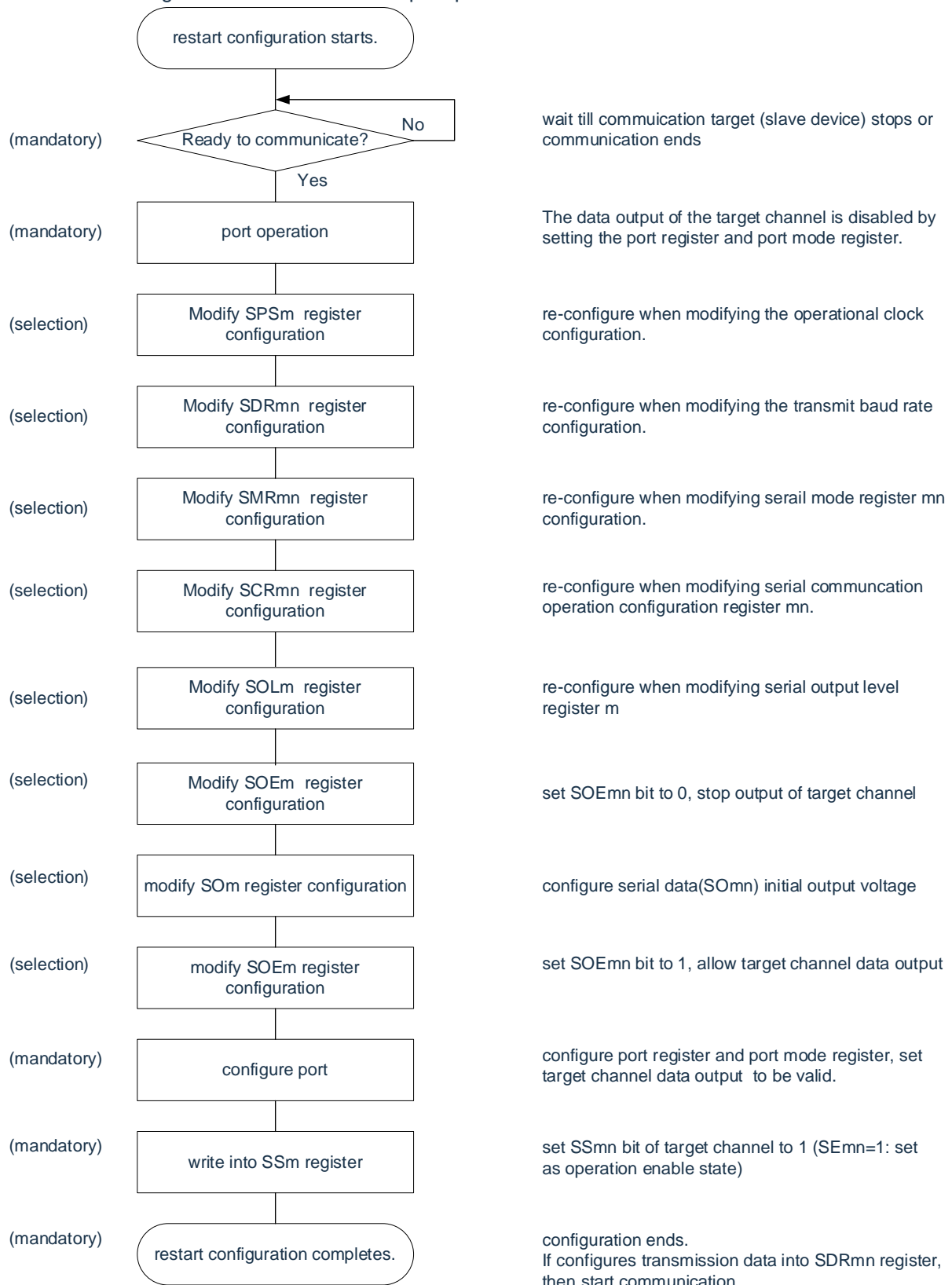


Figure 16-99: Reset the setup steps for the new start UART transmission

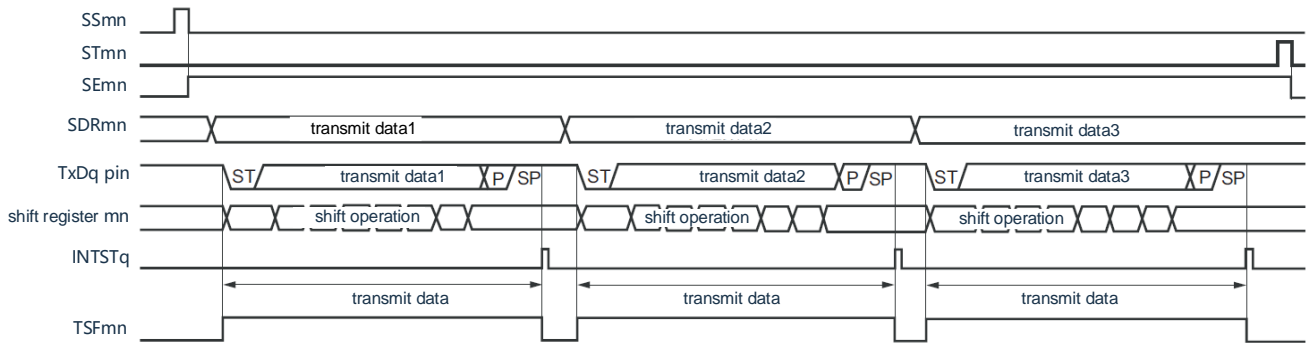


Remark: If you override PER0 in the abort settings to stop providing the clock, you must make the initial settings instead of restarting the settings when the communication object stops or the communication ends.



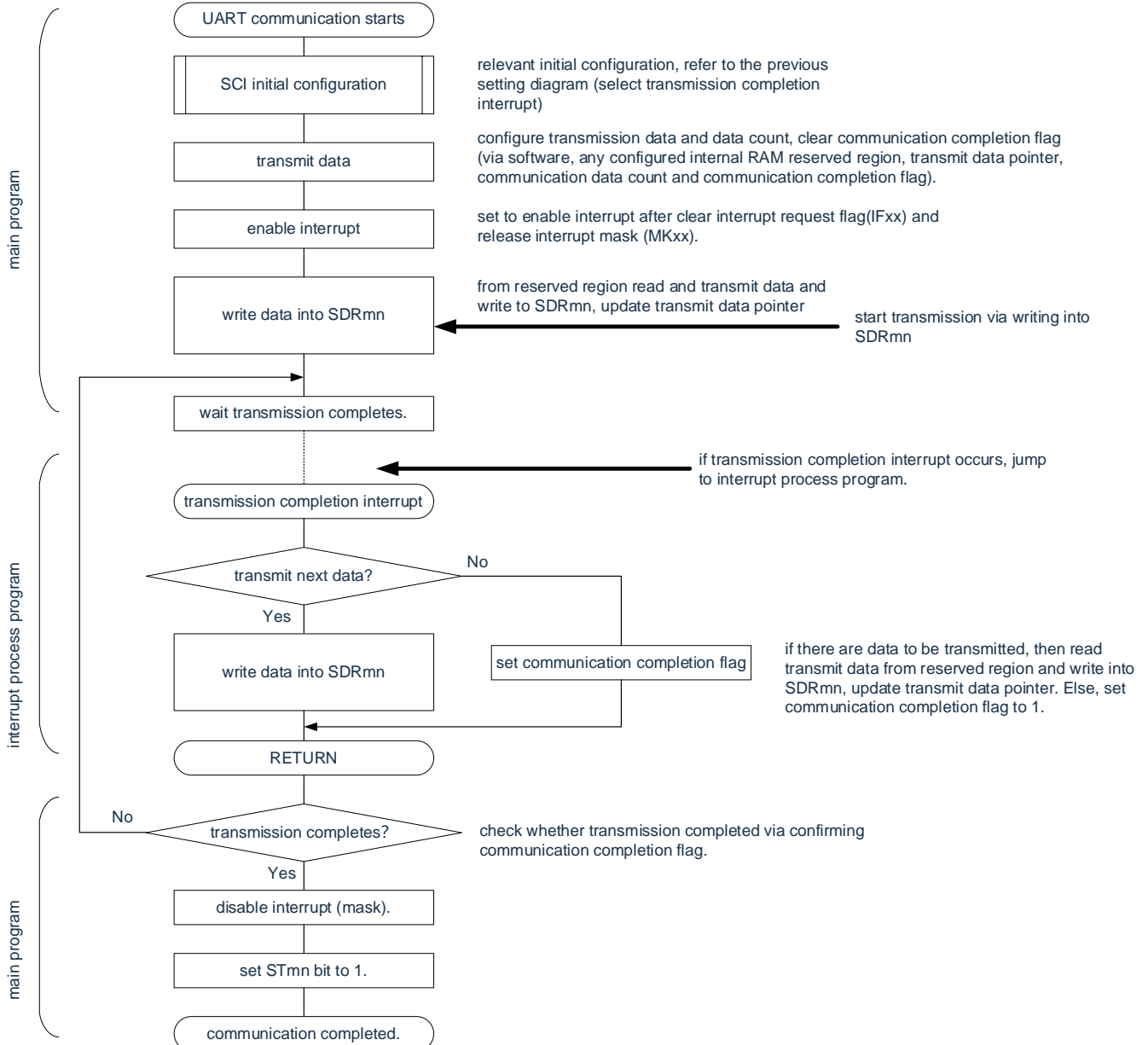
### (3) Process flow (single send mode)

Figure 16-100: UART send (single send mode)



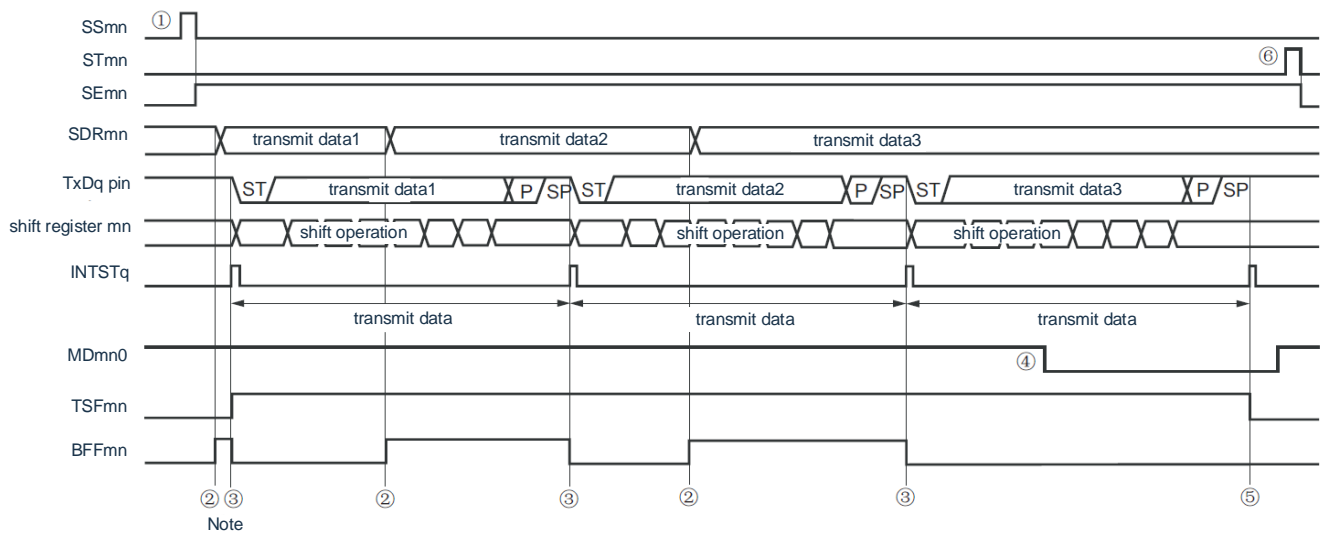
Remark: m: unit number(m=0, 1, 2)n: channel number(n=0)q: UART number(q=0~2)

Figure 16-101: UART transmit (single-pass mode)



#### (4) Process flow (continuous send mode)

Figure 16-102: UART transmission (continuous send mode)

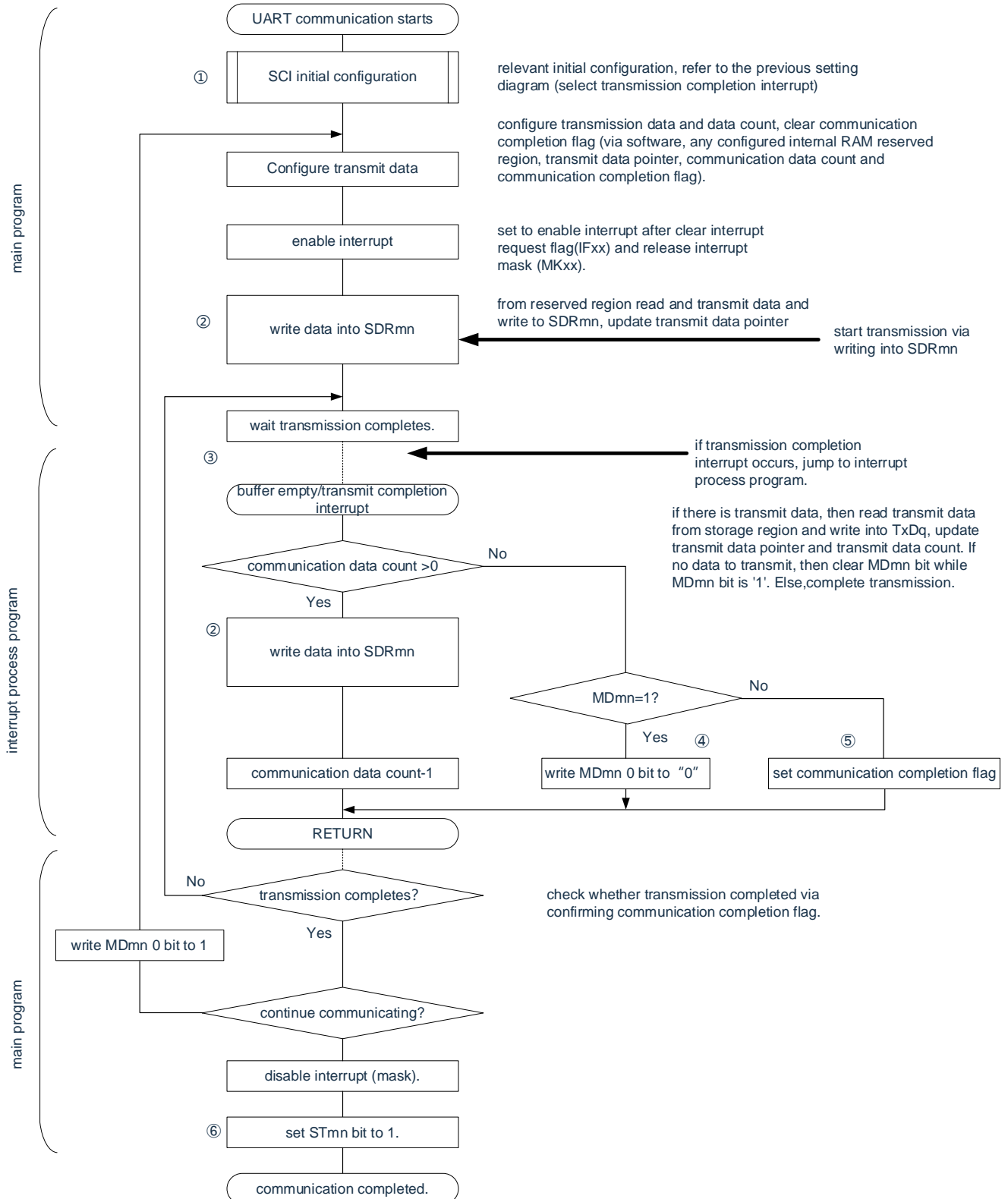


Note: If the BFFmn bit of the serial status register mn(SSRmn) is "1" (when valid data is saved in the serial data register mn(SDRmn)) is given When the SDRmn register writes the transmit data, it rewrites the send data.

Notice: MDmn0 bit of the serial mode register mn (SMRmn) can be rewritten even in operation. However, in order to catch the end-of-transmission interruption of the last sent data, it must be rewritten before the last bit of transmission begins.

Remark: m: unit number(m=0, 1, 2)n: channel number(n=0)q: UART number(q=0~2)

Figure 16-103: Flowchart of UART sending (continuous send mode)



Remark: (1) to (6) in the figure corresponds to (1) to (6) in "Figure 16-102 UART Transmission (Continuous Send Mode)".

## 16.7.2 UART reception

UART receive is the operation of this product microcontroller asynchronously receiving data from other devices.

The odd number of the 2 channels used by the UART is used for UART reception. However, the SMR registers for odd and even channels need to be set.

UART	UART0	UART1	UART2
Object channels	Channel 1 of SCI0	Channel 1 of SCI1	Channel 1 of SCI2
Pin used	RxD0	RxD1	RxD2
Interrupt	INTSR0	INTSR1	INTSR2
	Limited to end-of-transmit interrupts (buffer null interrupts are prohibited).		
Error detection flags	<ul style="list-style-type: none"> <li>• Frame Error Detection Flag (FEFmn)</li> <li>• Parity Error Detection Flag (PEFmn).</li> <li>• Overflow Error Detection Flag (OVFmn)</li> </ul>		
Transferred data length	7-bit, 8-bit, 9-bit or 16-bit		
Transfer rate	Max. $F_{MCK}/6[\text{bps}](\text{SDRmn}[15:9] \geq 2)$ , Min. $F_{CLK}/(2 \times 2^{15} \times 128)[\text{bps}]$		
Data phase	Positive-phase output (default: high). Inverting output (default: low).		
Parity bit	You can choose from the following: <ul style="list-style-type: none"> <li>• No parity bits(no parity).</li> <li>• Additional zero parity check(no parity).</li> <li>• Additional even parity check.</li> <li>• Additional odd parity check.</li> </ul>		
Stop bit	Add 1 bit.		
Data direction	MSB preferred or LSB preferred		

Note: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

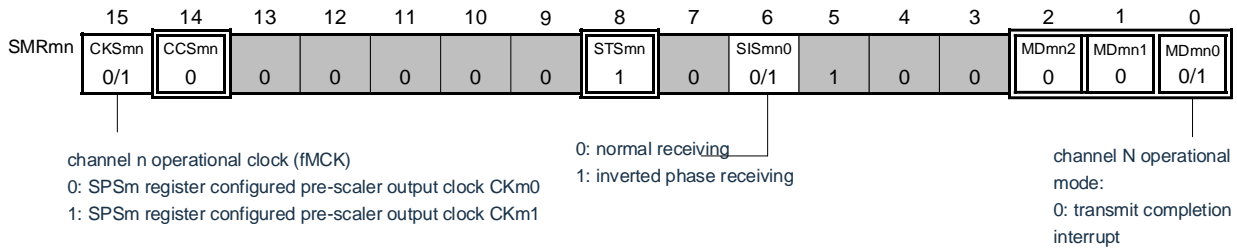
Remark:

1.  $F_{MCK}$ : The operating clock frequency of the object channel  
 $F_{CLK}$ : System clock frequency
2. m: unit number(m=0, 1, 2)n: channel number(n=1)

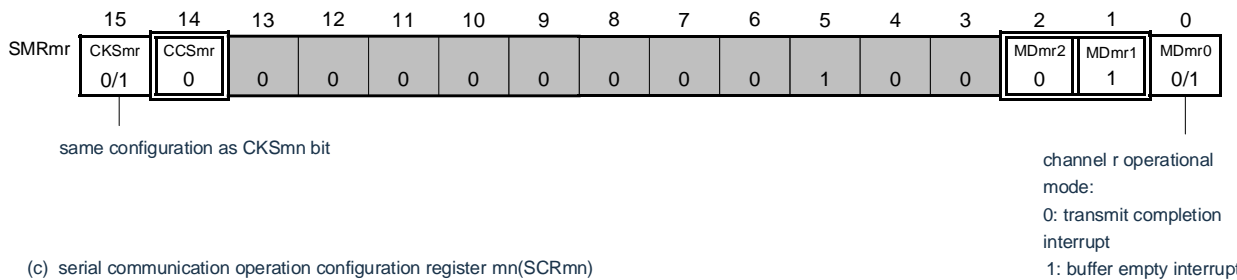
## (1) Register setting

Figure16-104: Example of register setting contents for UART reception of UART (UART0~UART2) (1/2)

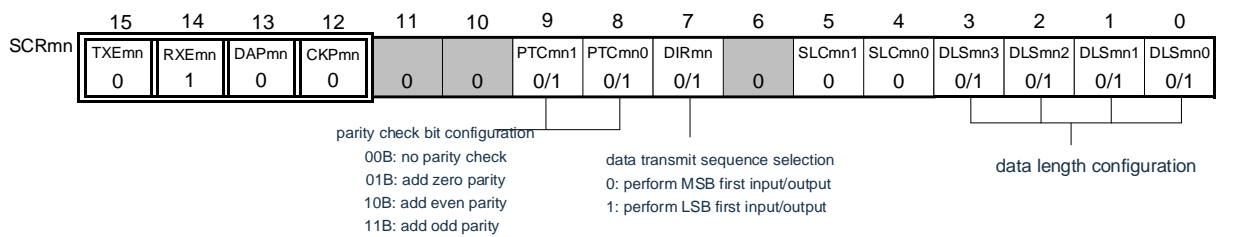
(a) serial mode register mn (SMRmn)



(b) serial mode register mr (SMRmr)



(c) serial communication operation configuration register mn (SCRmn)

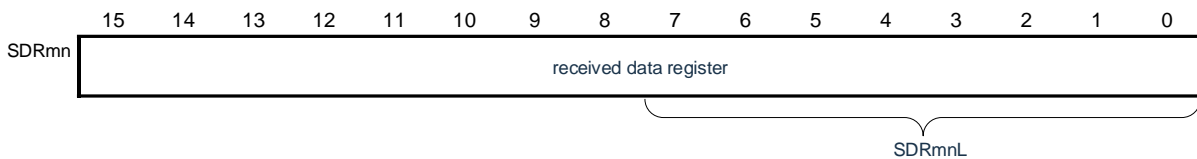


(d) serial data register mn (SDRmn)

(1) When operation stops (SEmn=0)



(2) During operation (SEmn=1) (lower 8 bits: SDRmnL)



Notice: For UART reception, the SMRmr register of channel r, which is paired with channel n, must also be set.

Remark:

- m: unit number(m=0, 1, 2)n: channel number(n=1)  
r: channel number(r=n-1)q: UART number(q=0~2)
- : Fixed setting in UART receive mode. ■: setting disabled(initial value).  
x: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).  
0/1: Set "0" or "1" according to the user's purpose.

**Figure 16-104: Example of register setting contents for UART reception of 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	0	0	CKOm1 ×	CKOm0 ×	0	0	0	0	0	0	SOM1 ×	SOM0 ×

(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	0	0	SOEm1 ×	SOEm0 ×

(g) serial channel start register m (SSm) .... Only set bit of target channel to "1".

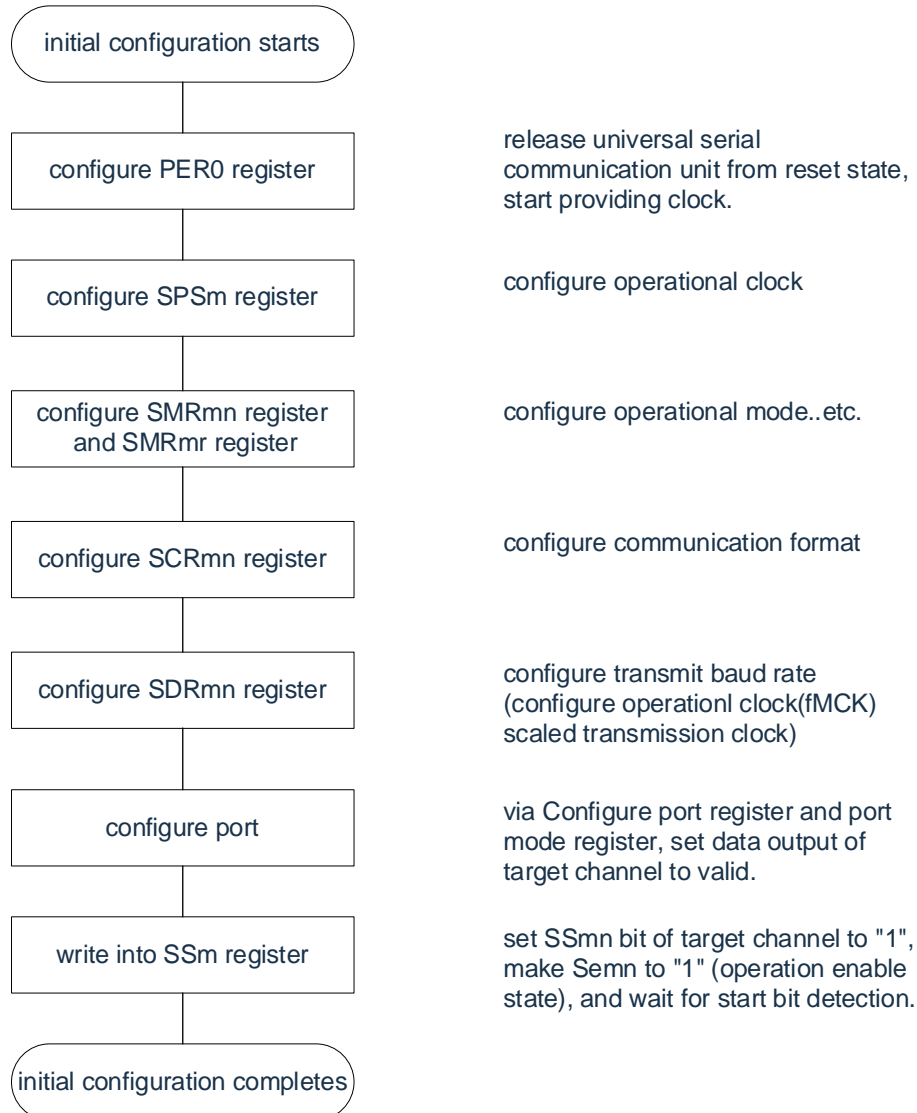
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SSm1 0/1	SSm0 ×

**Remark:**

1. m: unit number(m=0, 1, 2)
2. ☐ : Fixed setting in UART receive mode. ☐ : setting disabled(initial value).  
 ×: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).  
 0/1: Set "0" or "1" according to the user's purpose.

## (2) Procedure

Figure 16-105: 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 16-106: Stop steps for UART reception

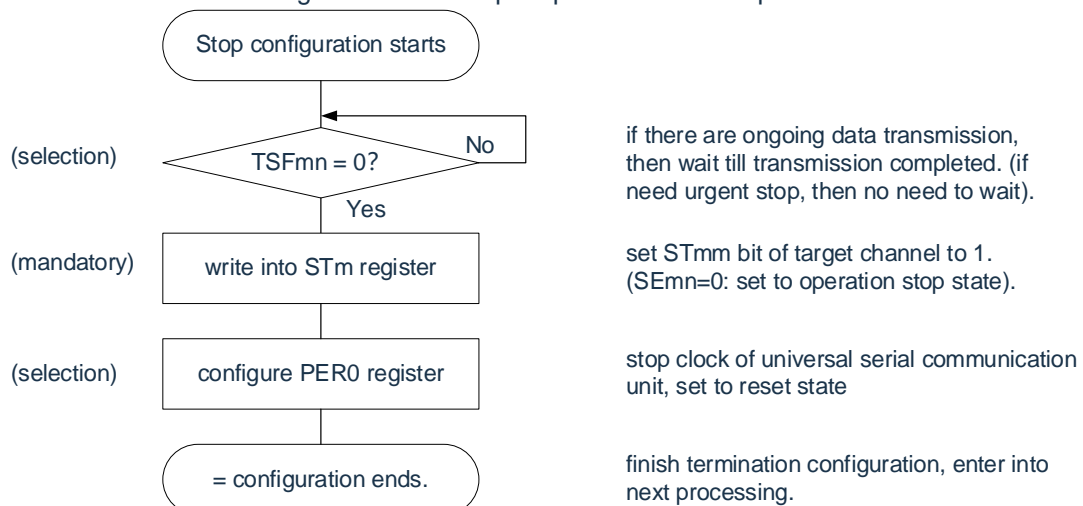
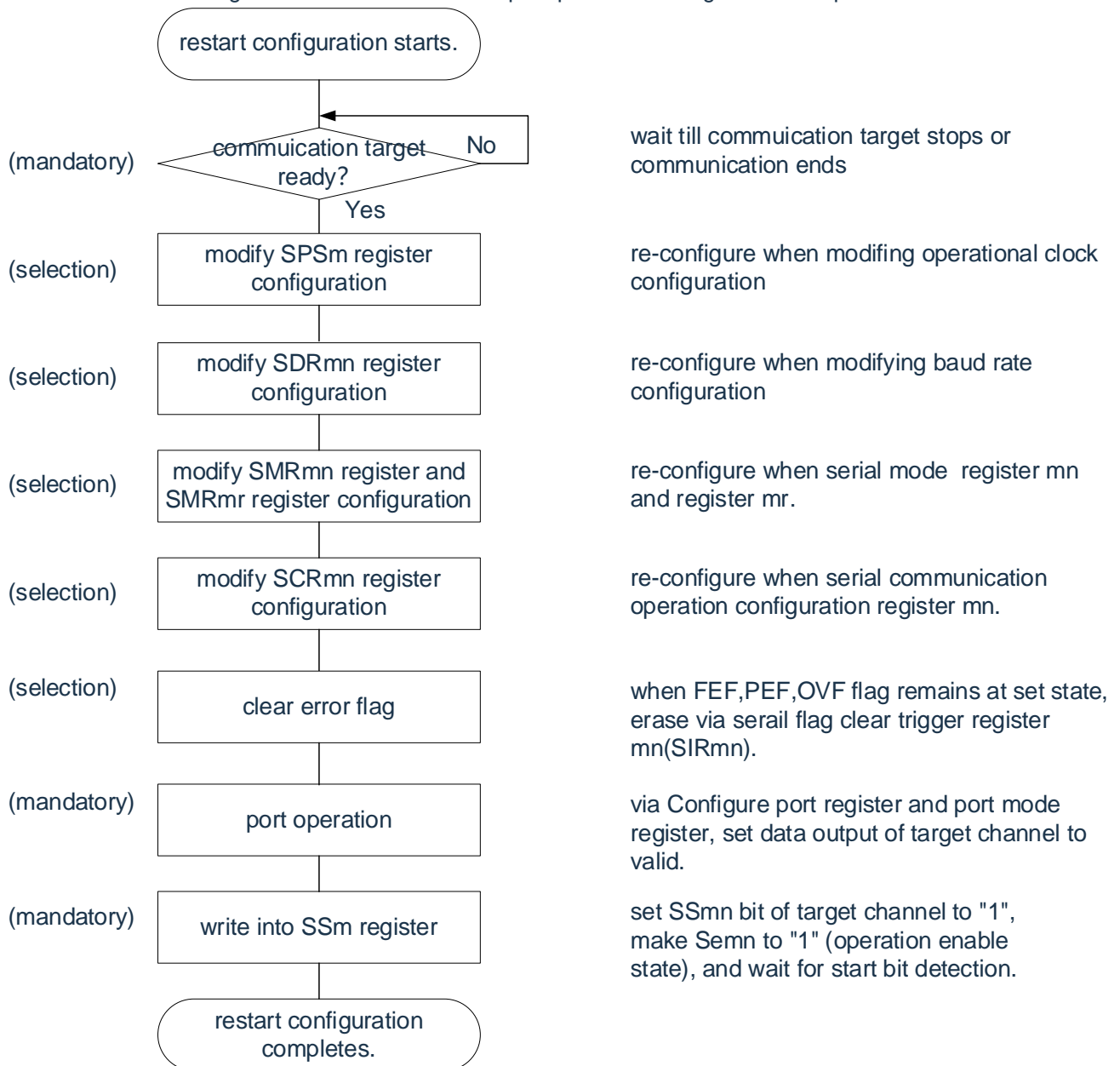


Figure 16-107: Reset the setup steps for restarting UART reception



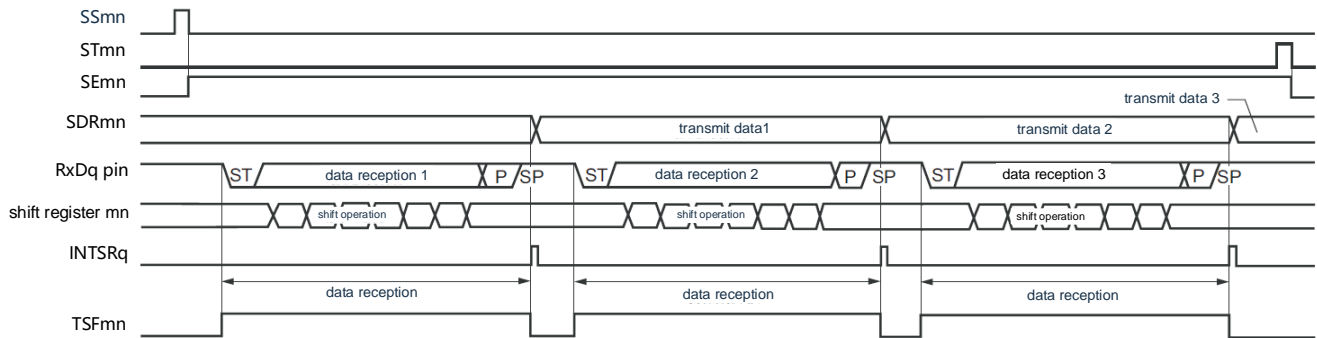
Notice: The SCRmn register must be set at least 4  $F_{MCK}$  clocks apart after setting RXEmn to "1" and then setting SSmn to "1".

Remark: If you override PER0 in the abort settings to stop providing the clock, you must make the initial settings instead of restarting the settings when the communication object stops or the communication ends.



(3) Process flow

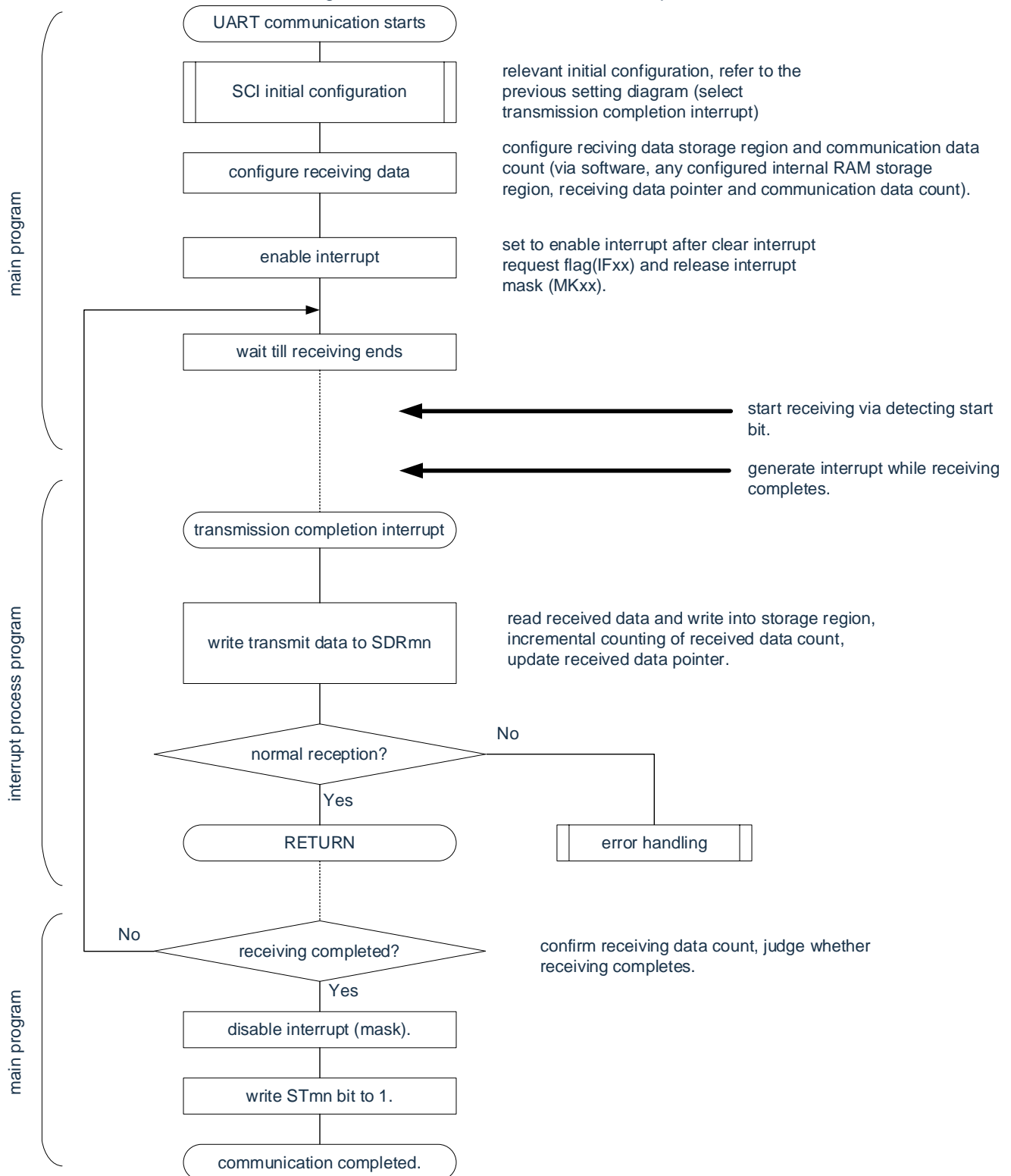
Figure 16-108: UART reception



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

R: channel number(r=n-1)q: UART number(q=0~2)

Figure 16-109: Flowchart of UART reception



### 16.7.3 Low-power UART mode function

This is a mode that enables the UART to receive by detecting the input of the RxDq pin in deep sleep mode. Normally, the UART stops communicating in Deep Sleep mode, but if you use Low Power UART mode, you can receive UART while the CPU is not running.

To use UARTq in low-power UART mode, the following settings must be made before moving to deep sleep mode (refer to “Figure 16-111 and Figure 16-113 Flow chart of low-power UART mode operation”).

- In low-power UART mode, you need to change the UART receive baud rate setting (to a different value from the normal operation). You must refer to Table 16-4, SPSm register and SDRmn register setting [15:9].
- Set the EOCmn bit and SSECmn bit. It can be set to allow or stop the generation of error interrupts in case of communication errors .
- The SWCm bit of the serial standby control register m (SSCm) must be set to "1" before shifting to deep sleep mode. After the initial setting, set the SSm1 bit of the serial channel start register m (SSm) to "1".
- If the start bit of the RxDq input is detected after shifting to deep sleep mode, UARTq starts receiving.

Notice:

1. Low-power UART mode can only be used when the high-speed internal oscillator clock ( $F_{IH}$ ) is selected as  $F_{CLK}$ .
2. The transmission rate in low-power UART mode is only 4800bps.
3. If SWCm is set to "1", UARTq can only be used when reception starts in deep sleep mode. When other low-power UART mode functions and interrupts are used at the same time and reception starts in the following states other than deep sleep mode, data may not be received properly and frame errors or parity errors may occur.
  - Receiving starts after setting SWCm bit to "1" and before moving to deep sleep mode
  - Start of reception in other low-power UART modes
  - Receiving starts after returning from deep sleep mode to normal operation via interrupts, etc. and before setting the SWCm bit to "0"
4. If the SSECm bit is set to "1", the PEFmn, FEFmn, OVFMn flags are not set in case of a parity error, frame error or overflow error. Therefore, when SSECm bit is "1", the PEFmn, FEFmn, OVFMn flags must be cleared and the SDRm1 register bit 7 ~ 0 (RxDq) must be read before setting the SWC0 bit to "1".
5. Shift to low-power UART mode by detecting the active edge of the RxDq pin.

If a short pulse is received that does not detect the input start bit, it may not start UART reception and remain in low-power UART mode. In this case, the data may not be received properly during the next UART reception and a frame error or a parity error may occur.

Table 16-4: Setting of UART receive baud rate in low power UART mode

High-speed internal oscillator (F <sub>IH</sub> )	UART receive baud rate in low power UART mode			
	Baud rate 4800bps			
	Operation clock(F <sub>MCK</sub> )	SDRmn[15:9]	Max tolerance value	Min tolerance value
32MHz±1.0% <sup>Note</sup>	F <sub>CLK</sub> /2 <sup>5</sup>	105	2.27%	-1.53%
24MHz±1.0% <sup>Note</sup>	F <sub>CLK</sub> /2 <sup>5</sup>	79	1.60%	-2.18%
16MHz±1.0% <sup>Note</sup>	F <sub>CLK</sub> /2 <sup>4</sup>	105	2.27%	-1.53%
12MHz±1.0% <sup>Note</sup>	F <sub>CLK</sub> /2 <sup>4</sup>	79	1.60%	-2.19%
8MHz±1.0% <sup>Note</sup>	F <sub>CLK</sub> /2 <sup>3</sup>	105	2.27%	-1.53%
6MHz±1.0% <sup>Note</sup>	F <sub>CLK</sub> /2 <sup>3</sup>	79	1.60%	-2.19%
4MHz±1.0% <sup>Note</sup>	F <sub>CLK</sub> /2 <sup>2</sup>	105	2.27%	-1.53%
3MHz±1.0% <sup>Note</sup>	F <sub>CLK</sub> /2 <sup>2</sup>	79	1.60%	-2.19%
2MHz±1.0% <sup>Note</sup>	F <sub>CLK</sub> /2	105	2.27%	-1.54%
1MHz±1.0% <sup>Note</sup>	F <sub>CLK</sub>	105	2.27%	-1.57%

Note: When the frequency accuracy of the high-speed internal oscillator is ±1.5%, ±2.0%, the following tolerance range becomes narrow.

- In the case of F<sub>IH</sub> ±1.5%, the maximum tolerance in the above table must be set to -0.5% and the minimum tolerance must be set to +0.5%.
- In the case of F<sub>IH</sub> ±2.0%, the maximum tolerance in the table above must be set to -1.0% and the minimum tolerance must be set to +1.0%.

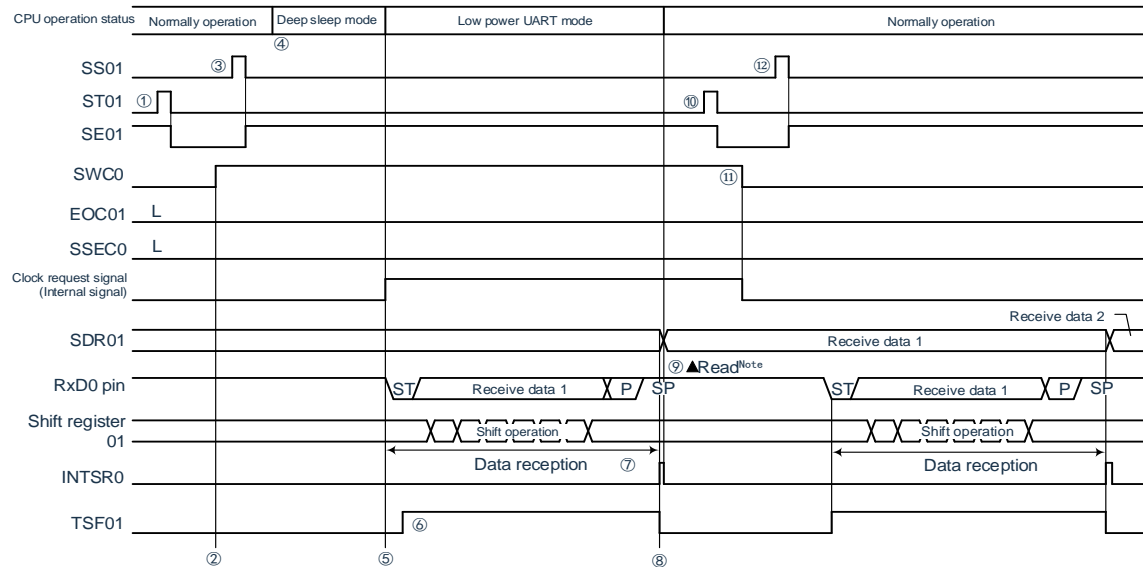
Remark: The maximum and minimum tolerances are the allowable baud rate for UART reception.

The baud rate of the sender must be set within this range

(1) Low power UART mode operation (EOCm1=0, SSECm=0/1)

Because the EOCm1 bit is "0", regardless of the SSECm bit setting, and no error interrupt is generated even if a communication error occurs. However, the end-of-transmission interrupt (INTSRq) is generated.

Figure 16-110: Timing diagram of low-power UART mode operation (EOCm1=0, SSECm=0/1)

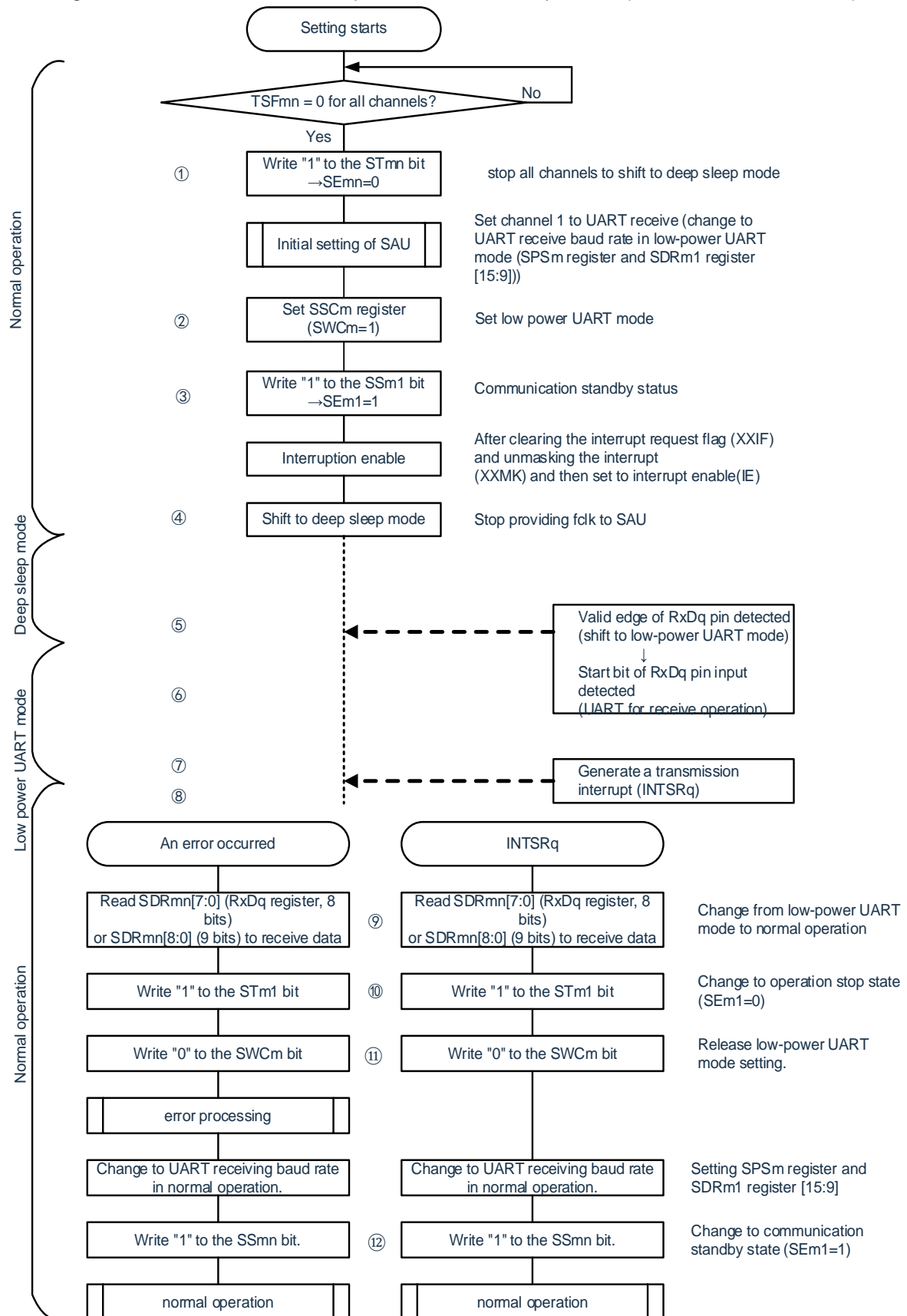


Note: The received data must be read when the SWCm bit is "1".

Notice: The STM1 bit must be set to "1" (to clear the SEm1 bit and stop operation) before moving to low-power UART mode or after receiving in low-power UART mode, and the SWCm bit must also be cleared after receiving is completed (to release low-power UART mode).

Remark: ① to ⑫ in the figure correspond to ① to ⑫ in “Figure 16-111: ”.

Figure 16-111: Flow chart of low-power UART mode operation (EOCm1=0, SSECm=0/1)

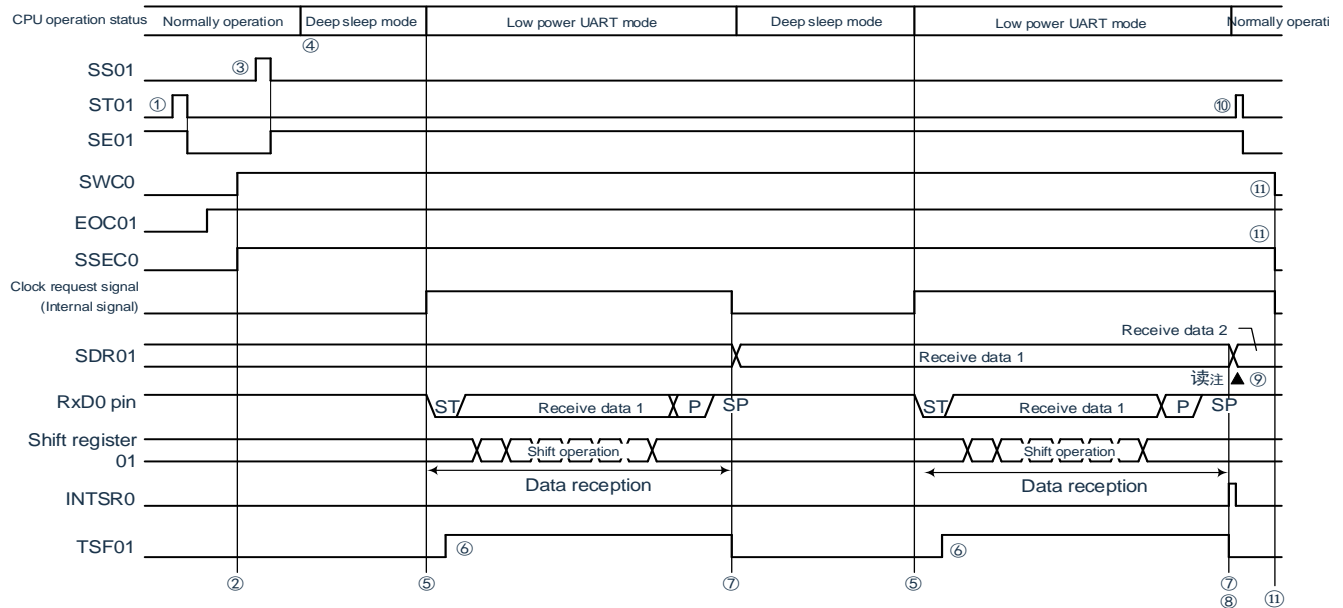


Remark: ① to ⑫ correspond to ① to ⑫ in "Figure 16-110: Timing diagram of low-power UART mode operation (EOCm1=0, SSECm=0/1)".

## (2) Low power UART mode operation (EOCm1=1, SSECM=1)

Because the EOCm1 bit is "1", a communication error does not generate an error interrupt.

Figure 16-112: Timing diagram for low-power UART mode operation (EOCm1=1, SSECM=1)



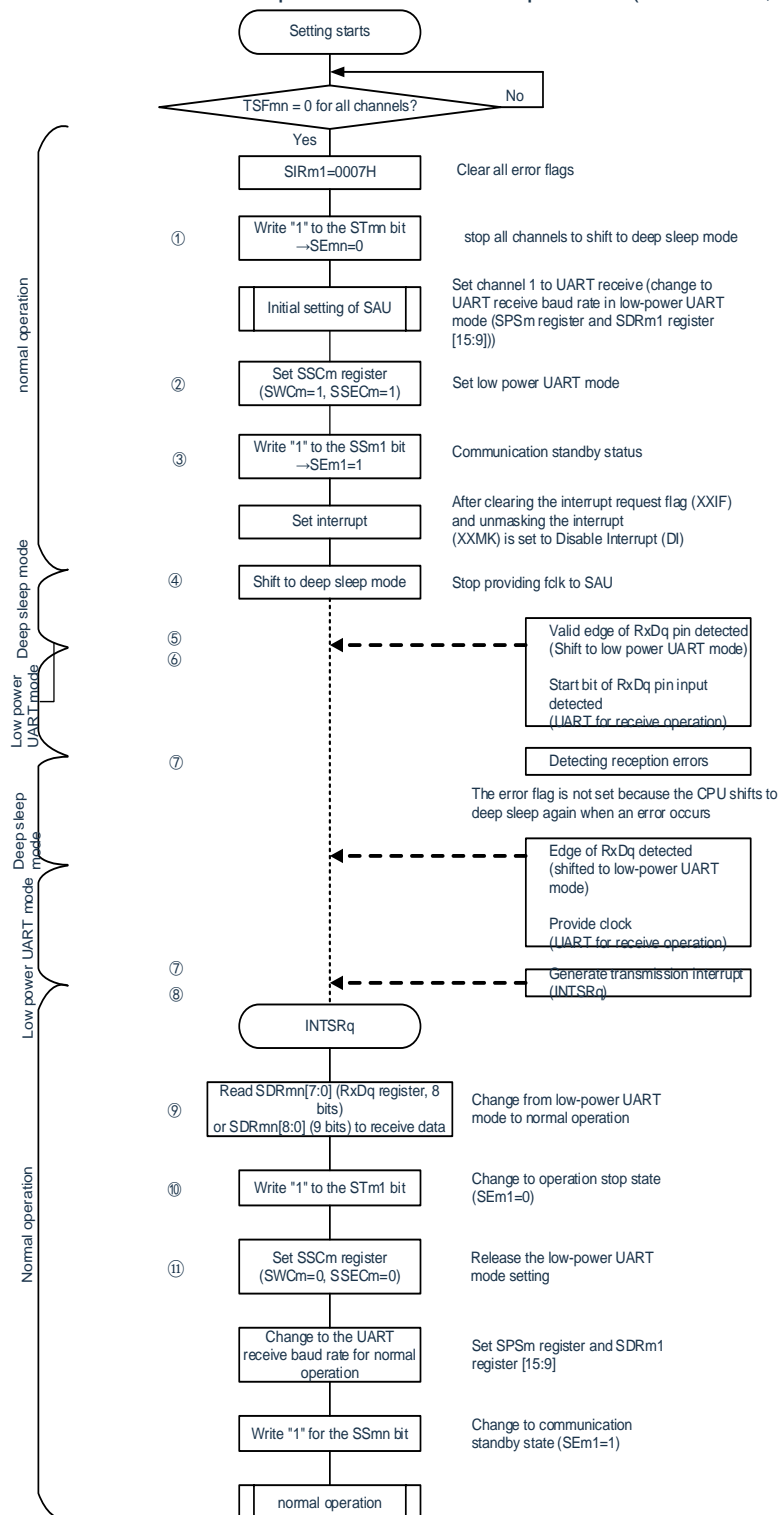
Note: The received data must be read when the SWCm bit is "1".

Notice:

1. The STm1 bit must be set to "1" (to clear the SEm1 bit and stop operation) before moving to low-power UART mode or after receiving in low-power UART mode, and the SWCm bit must also be cleared after receiving (to release low-power UART mode).
2. If the SSECM bit is "1", the PEFm1, FEFm1, OVFM1 flags are not set in case of a parity error, frame error or overflow error. Therefore, when SSECM bit is "1", the PEFm1, FEFm1, OVFM1 flags must be cleared and SDRm1[7:0] (RXDq register, 8 bits) or SDRm1[8:0] (9 bits) must be read before setting SWCm to "1".

Remark: ① to ⑪ correspond to ① to ⑪ in "Figure 16-113: ".

Figure 16-113: Flow chart of low-power UART mode operation (EOCm1=1, SSECm=1)



Notice: If the SSECm bit is "1", the PEFm1, FEFm1, OVFM1 flags are not set in case of a parity error, frame error or overflow error, so when SSECm bit is "1", you must clear the PEFm1, FEFm1, OVFM1 flags and read SDRm1[7:0] (RXDq register, 8 bits) or SDRm1[8:0] (9 bits) before setting SWC0 to "1".

Remark: ① to ⑪ correspond to ① to ⑪ in "Figure 16-112: ".



## 16.7.4 Calculation of baud rate

### (1) Equation for calculating baud rate

The baud rate of UART (UART0~UART2) communication can be calculated using the following

$$\text{(Baud rate)} = \{ \text{Operating clock (F}_{\text{MCK}}) \text{ frequency of the object channel} \} \div (\text{SDRmn}[15:9] + 1) \div 2 [\text{bps}]$$

calculation equation:

Notice: It is forbidden to set the serial data register mn (SDRmn) to SDRmn [15:9] to "0000000B" and "0000001B".

Remark:

1. Because when using UART, the value of SDRmn [15:9] is the value of bit15~9 of the SDRmn register (0000010B ~1111111B), so it is 2 to 127.
2. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)  
The operating clock (F<sub>MCK</sub>) depends on the bit15 (CKSmn bit) of the serial clock selection register m (SPSm) and the serial mode register mn (SMRmn).

Table 16-5: UART operating clocks

SMR <sub>m</sub> n Register	SPS <sub>m</sub> register								Running Clock (F <sub>MCK</sub> ) <sup>Note</sup>	
CKS <sub>m</sub> n	PRS <sub>m</sub> 13	PRS <sub>m</sub> 12	PRS <sub>m</sub> 11	PRS <sub>m</sub> 10	PRS <sub>m</sub> 03	PRS <sub>m</sub> 02	PRS <sub>m</sub> 01	PRS <sub>m</sub> 00		F <sub>CLK</sub> =32MHz runtime
0	X	X	X	X	0	0	0	0	F <sub>CLK</sub>	32MHz
	X	X	X	X	0	0	0	1	F <sub>CLK</sub> /2	16MHz
	X	X	X	X	0	0	1	0	F <sub>CLK</sub> /2 <sup>2</sup>	8MHz
	X	X	X	X	0	0	1	1	F <sub>CLK</sub> /2 <sup>3</sup>	4MHz
	X	X	X	X	0	1	0	0	F <sub>CLK</sub> /2 <sup>4</sup>	2MHz
	X	X	X	X	0	1	0	1	F <sub>CLK</sub> /2 <sup>5</sup>	1MHz
	X	X	X	X	0	1	1	0	F <sub>CLK</sub> /2 <sup>6</sup>	500KHz
	X	X	X	X	0	1	1	1	F <sub>CLK</sub> /2 <sup>7</sup>	250KHz
	X	X	X	X	1	0	0	0	F <sub>CLK</sub> /2 <sup>8</sup>	125KHz
	X	X	X	X	1	0	0	1	F <sub>CLK</sub> /2 <sup>9</sup>	62.5KHz
	X	X	X	X	1	0	1	0	F <sub>CLK</sub> /2 <sup>10</sup>	31.25KHz
	X	X	X	X	1	0	1	1	F <sub>CLK</sub> /2 <sup>11</sup>	15.63KHz
	X	X	X	X	1	1	0	0	F <sub>CLK</sub> /2 <sup>12</sup>	7.81KHz
	X	X	X	X	1	1	0	1	F <sub>CLK</sub> /2 <sup>13</sup>	3.91KHz
	X	X	X	X	1	1	1	0	F <sub>CLK</sub> /2 <sup>14</sup>	1.95KHz
	X	X	X	X	1	1	1	1	F <sub>CLK</sub> /2 <sup>15</sup>	977Hz
1	0	0	0	0	X	X	X	X	F <sub>CLK</sub>	32MHz
	0	0	0	1	X	X	X	X	F <sub>CLK</sub> /2	16MHz
	0	0	1	0	X	X	X	X	F <sub>CLK</sub> /2 <sup>2</sup>	8MHz
	0	0	1	1	X	X	X	X	F <sub>CLK</sub> /2 <sup>3</sup>	4MHz
	0	1	0	0	X	X	X	X	F <sub>CLK</sub> /2 <sup>4</sup>	2MHz
	0	1	0	1	X	X	X	X	F <sub>CLK</sub> /2 <sup>5</sup>	1MHz
	0	1	1	0	X	X	X	X	F <sub>CLK</sub> /2 <sup>6</sup>	500KHz
	0	1	1	1	X	X	X	X	F <sub>CLK</sub> /2 <sup>7</sup>	250KHz
	1	0	0	0	X	X	X	X	F <sub>CLK</sub> /2 <sup>8</sup>	125KHz
	1	0	0	1	X	X	X	X	F <sub>CLK</sub> /2 <sup>9</sup>	62.5KHz
	1	0	1	0	X	X	X	X	F <sub>CLK</sub> /2 <sup>10</sup>	31.25KHz
	1	0	1	1	X	X	X	X	F <sub>CLK</sub> /2 <sup>11</sup>	15.63KHz
	1	1	0	0	X	X	X	X	F <sub>CLK</sub> /2 <sup>12</sup>	7.81KHz
	1	1	0	1	X	X	X	X	F <sub>CLK</sub> /2 <sup>13</sup>	3.91KHz
	1	1	1	0	X	X	X	X	F <sub>CLK</sub> /2 <sup>14</sup>	1.95KHz
	1	1	1	1	X	X	X	X	F <sub>CLK</sub> /2 <sup>15</sup>	977Hz

Note: When you change the clock selected as FCLK (change the value of the system clock control register (CKC)), you must stop the operation of the general-purpose serial communication unit (SCI) (serial channel stop register m (ST<sub>m</sub>)=000FH) after making changes.

Remark:

1. X: Ignore
2. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)

## (2) Baud rate error when sending

The baud rate error during UART (UART0~UART2) communication transmission can be calculated using the following calculation equation, and the baud rate of the sender must be set within the allowable baud

$$(\text{Baud rate error}) = (\text{calculated value of baud rate}) \div (\text{value of the target baud rate}) \times 100 - 100[\%]$$

rate of the receiver.

An example of the UART baud rate at  $F_{CLK}=32\text{MHz}$  is shown below.

UART baud rate (Target baud rate)	$F_{CLK}=32\text{MHz}$			
	Running Clock ( $F_{MCK}$ )	SDRmn[15:9]	Calculated value of the baud rate	Error from the target baud rate
300bps	$F_{CLK}/2^9$	103	300.48bps	+0.16%
600bps	$F_{CLK}/2^8$	103	600.96bps	+0.16%
1200bps	$F_{CLK}/2^7$	103	1201.92bps	+0.16%
2400bps	$F_{CLK}/2^6$	103	2403.85bps	+0.16%
4800bps	$F_{CLK}/2^5$	103	4807.69bps	+0.16%
9600bps	$F_{CLK}/2^4$	103	9615.38bps	+0.16%
19200bps	$F_{CLK}/2^3$	103	19230.8bps	+0.16%
31250bps	$F_{CLK}/2^3$	63	31250.0bps	$\pm 0.0\%$
38400bps	$F_{CLK}/2^2$	103	38461.5bps	+0.16%
76800bps	$F_{CLK}/2$	103	76923.1bps	+0.16%
153600bps	$F_{CLK}$	103	153846bps	+0.16%
312500bps	$F_{CLK}$	50	313725bps	$\pm 0.39\%$

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

### (3) Allowable range of baud rate when receiving

The baud rate tolerance range for UART (UART0~UART2) communication reception can be calculated using the following equation, and the baud rate of the sender must be set within the baud

$$(\text{Maximum baud rate that can be received}) = \frac{2 \times k \times \text{Nfr}}{2 \times k \times \text{Nfr} - k + 2} \times \text{Brate}$$

$$(\text{Minimum baud rate that can be received}) = \frac{2 \times k \times (\text{Nfr} - 1)}{2 \times k \times \text{Nfr} - k - 2} \times \text{Brate}$$

rate tolerance range of the receiver.

Brate: Calculated value of the receiver's baud rate (refer to "Calculation of baud rate")

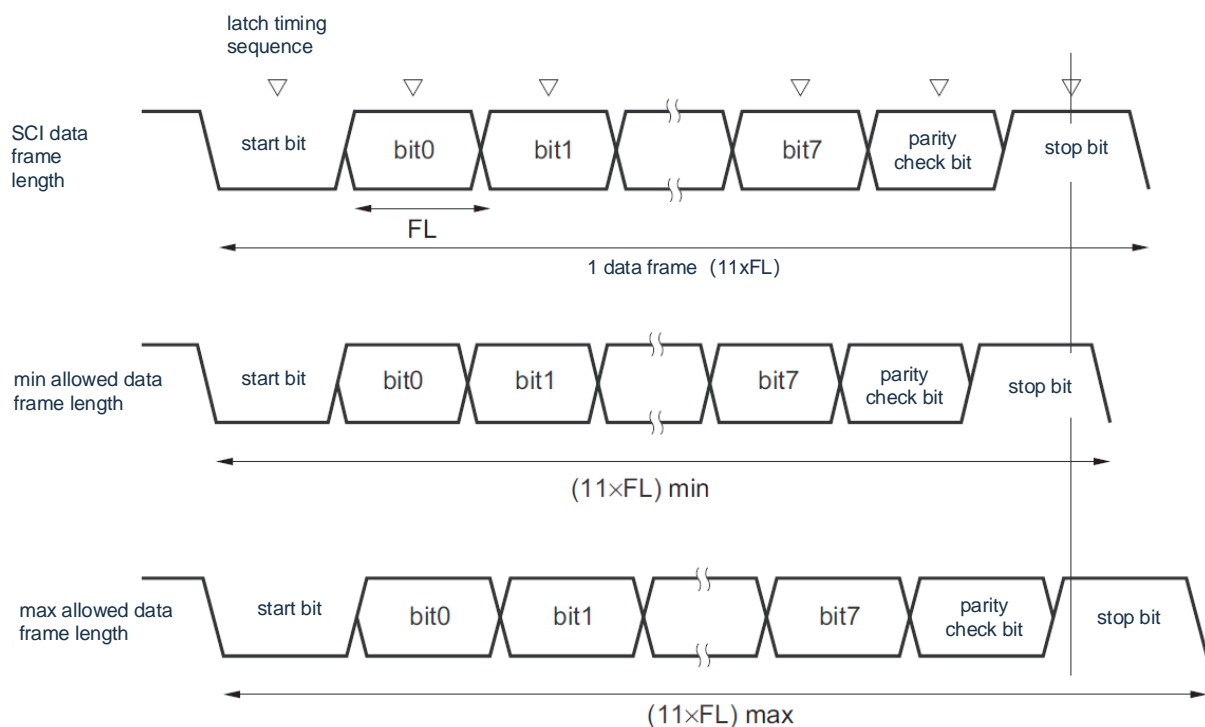
k: SDRmn[15:9]+1

Nfr: Frame length of 1 data [bit]

=(start bit) + (data length) + (parity bit) + (stop bit)

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

Figure16-114: Baud rate tolerance range at reception (1 data frame length = 11 bits)



As shown in Figure 16-114, after the start bit is detected, the latch timing of the received data depends on the division ratio set by bit15 to 9 of the serial data register mn (SDRmn). If the last data (stop bit) catches up with this latch timing, it can be received normally.

## 16.7.5 Handling steps when an error occurs during UART (UART0~UART2) communication

The processing steps for errors that occur during UART (UART0~UART2) communication are shown in Figure16-115andFigure 16-116.

Figure16-115: 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 in the receivable state.	This is to prevent an overflow error from occurring at the end of the next receive during error handling.
Read the serial status register mn (SSRmn).		The error type is determined, and the reading value is used to clear the error flag.
Clear the trigger register mn to the serial flag (SDIRmn) write "1".	→ Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, only errors during the read operation can be cleared.

Figure 16-116: Processing steps when a frame 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 receivable.	This is to prevent overflow errors from occurring at the end of the next receive during mishandling.
Read the serial status register mn (SSRmn).		The error class is judged, and the reading value is used to remove the error flag.
Write the serial flag to clear the trigger register mn (SIRmn).	→ Clear error flags.	By writing the read value of the SSRmn register directly to the SDIRmn register, only errors during the read operation can be cleared.
Set the STmn bit of the serial channel stop register m (STm) to "1".	→ The SEMn bit of the serial channel allow status register m (SEm) is "0" and channel n is in operation stop status.	
Synchronize processing with the communicating party.		Because the start bit is offset, a frame error can be considered. Therefore, it is necessary to re-synchronize with the communicating party and restart communication.
Set the SSmn bit of the serial channel start register m (SSm) to "1".	→ The SEMn bit of the Serial Channel Allowed Status Register m (SEm) is "1" and channel n is operational.	

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

## 16.8 Operation of LIN communication

### 16.8.1 LIN transmission

UART0 supports LIN communication in UART delivery.

LIN sends channel 0 of usage unit 0.

UART	UART0	UART1	UART2
LIN Communication Support	Yes	No	No
object channel	Channel 0 of SCI0	-	-
Pin Used	TxD0	-	-
Interrupt	INTST0	-	-
	Selectable transmission end interrupt (single transmission mode) or buffer air interrupt (continuous transmission mode).		
Error detection flag	None		
Length of transmit data	8-bit		
Transfer Rate <sup>Note</sup>	Max. $F_{MCK}/6$ [bps](SDR00[15:9]≥2), Min. $F_{CLK}/(2 \times 2^{15} \times 128)$ [bps]		
Data phase parity bit	Positive output (default: High level). Invert output (default: Low level).		
Stop bit	No parity bits.		
Data orientation	Add 1 bit.		
LIN Communication Support	LSB preferred		

Note: It must be used within the context of peripheral functionality (reference data manual) that meets this condition and electrical characteristics, and 2.4/9.6/19.2kbps.

Remark:  $F_{MCK}$ : The operating clock frequency of the object channel

$F_{CLK}$ : System clock frequency

LIN is the abbreviation of Local Interconnect Network, which is a low-speed (1~20kbps) serial communication protocol to reduce automobile network cost. LIN communications are single-master communications, with up to 15 slave devices connected to a single master device.

The LIN slave is used for the control of switches, transmission devices, sensors, etc., which are connected to the main control device through the LIN.

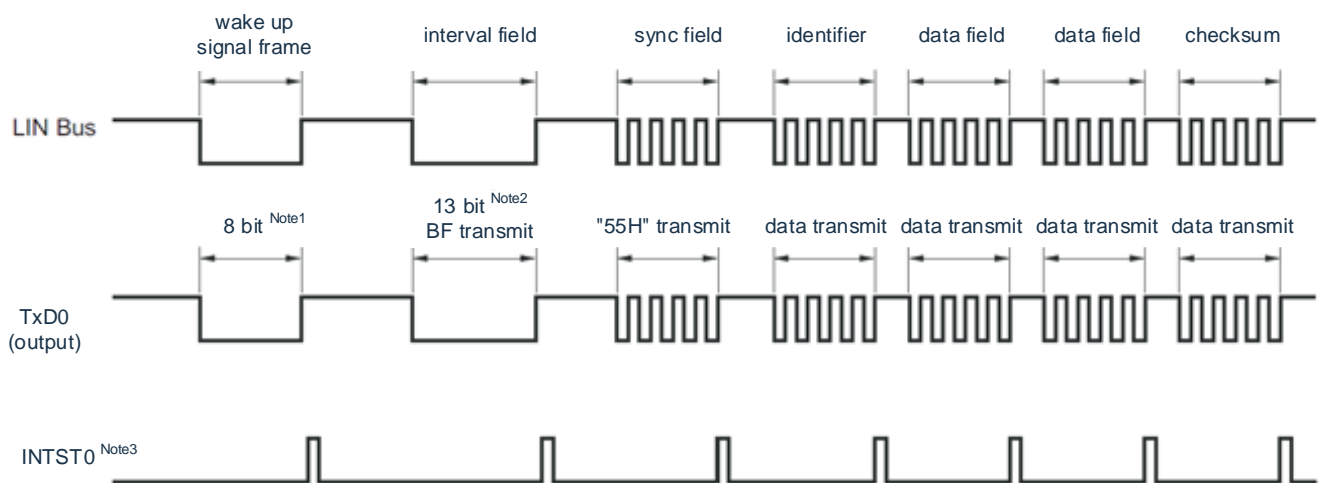
The LIN controls the network which connects CAN (Controller Area Network) and so on.

The LIN bus is a single-line bus, which connects nodes through ISDO9141-compliant transceivers.

According to the LIN protocol, the master device sends a frame with additional baud rate information, and the slave device receives the frame and corrects the baud rate error with the master device. Therefore, if the baud rate error of the slave device is not greater than  $\pm 15\%$ , communication can be performed.

A summary of the LIN's send operations is shown in Figure14-113.

Figure 16-117: LIN Send Operation



Note 1: In order to meet the requirements of wake-up signal, set baud rate and send "80H" data to correspond.

Note 2: The interval segment is specified as a 13-bit wide low-level output, so the baud rate used for the main transmission is  $N[\text{bps}]$ :

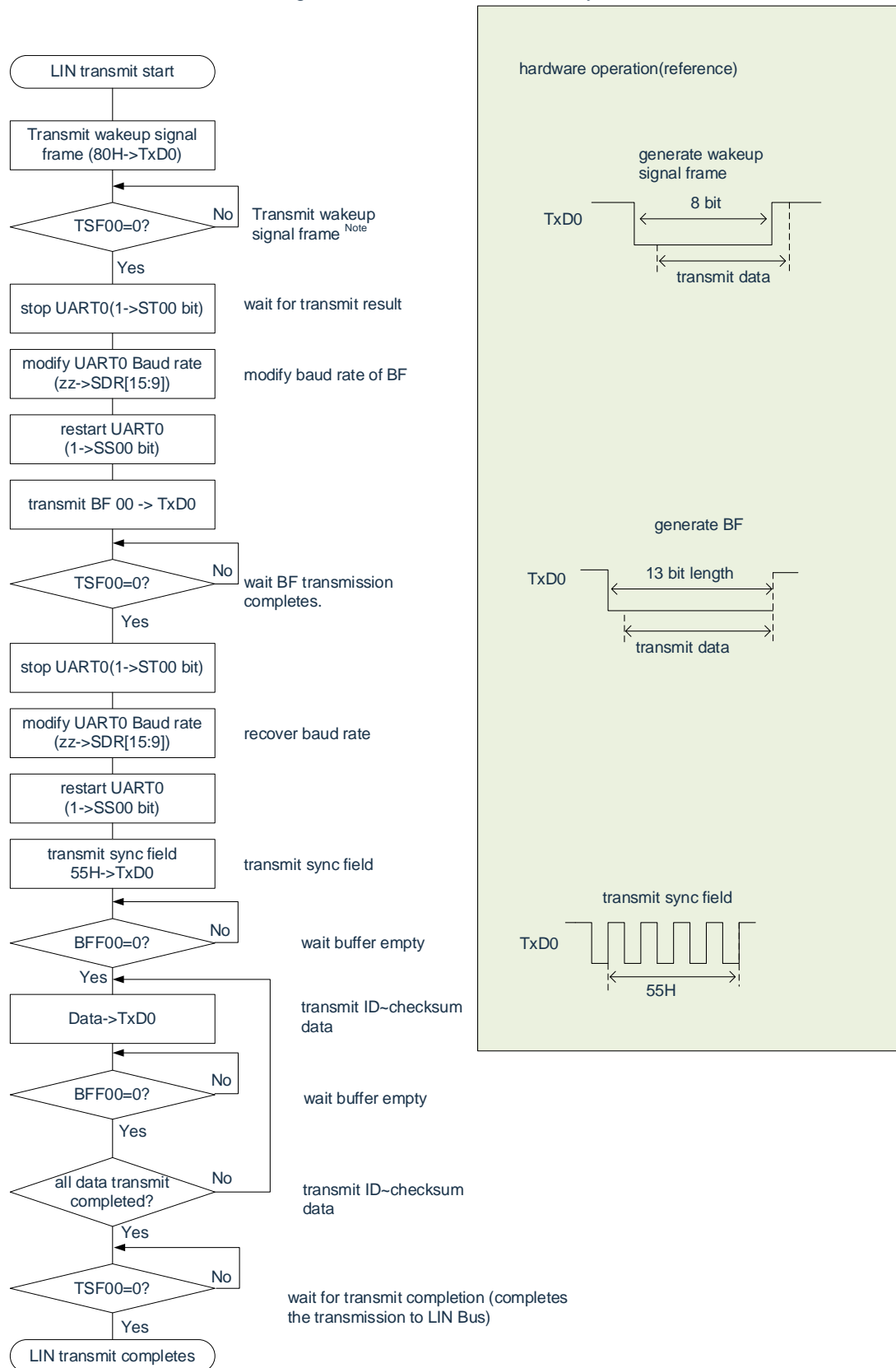
$$(\text{Baud Rate for Interval}) = 9/13 \times N$$

Send the data of "00H" through this baud rate to generate intervals.

Note 3: Output INTST0 at the end of each data transmission, and also output INTST0 at BF transmission.

Remark: The software controls the interval between segments.

Figure 16-118: Flowchart sent by LIN



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 sending.



## 16.8.2 LIN reception

In UART receiving, UART0 supports LIN communication.

The LIN receives the channel 1 of the Unit0.

UART	UART0	UART1	UART2
LIN Communication Support	Yes	No	No
Object channels	Channel 1 of SCI0	-	-
Pin used	RxD0	-	-
Interrupt	INTSR0	-	-
Error detection flag	Interrupt at that end of the transfer only (Disable from setting buffer air interrupt). <ul style="list-style-type: none"> <li>• Frame Error Detection Flag (FEF01)</li> <li>• Overflow Error Detection Flag (OVF01)</li> </ul>		
Transferred data length	8-bit		
Transfer Rate <sup>Note</sup>	Max. $F_{MCK}/6[\text{bps}]$ (SDR01[15:9] $\geq 2$ ), Min. $F_{CLK}/(2 \times 2^{15} \times 128)[\text{bps}]$		
Data phase	Positive-phase output (default: high). Inverting output (default: low).		
Parity bit	No parity bits (no parity).		
Stop bit	Add 1 bit.		
Data direction	LSB preferred		

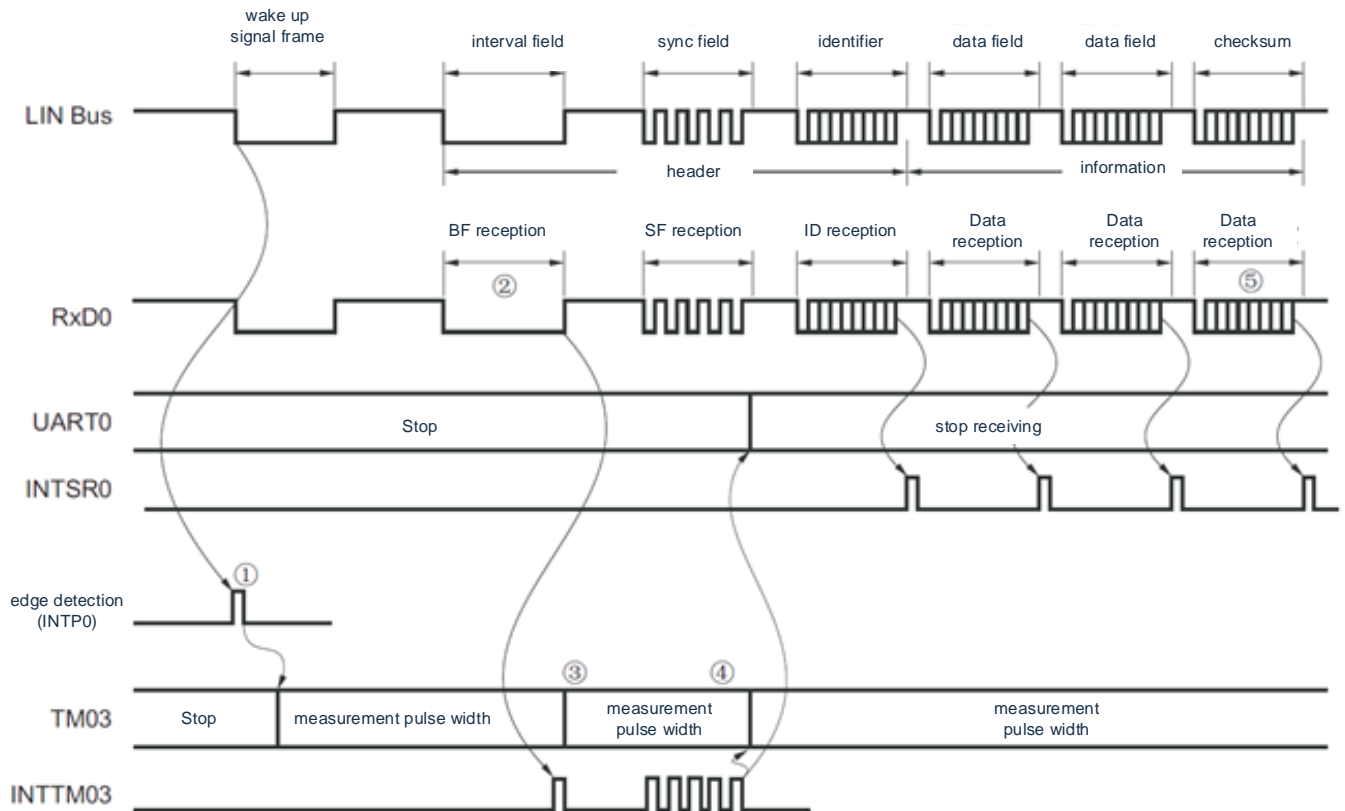
Note: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

Remark:  $F_{MCK}$ : The operating clock frequency of the object channel

$F_{CLK}$ : System clock frequency

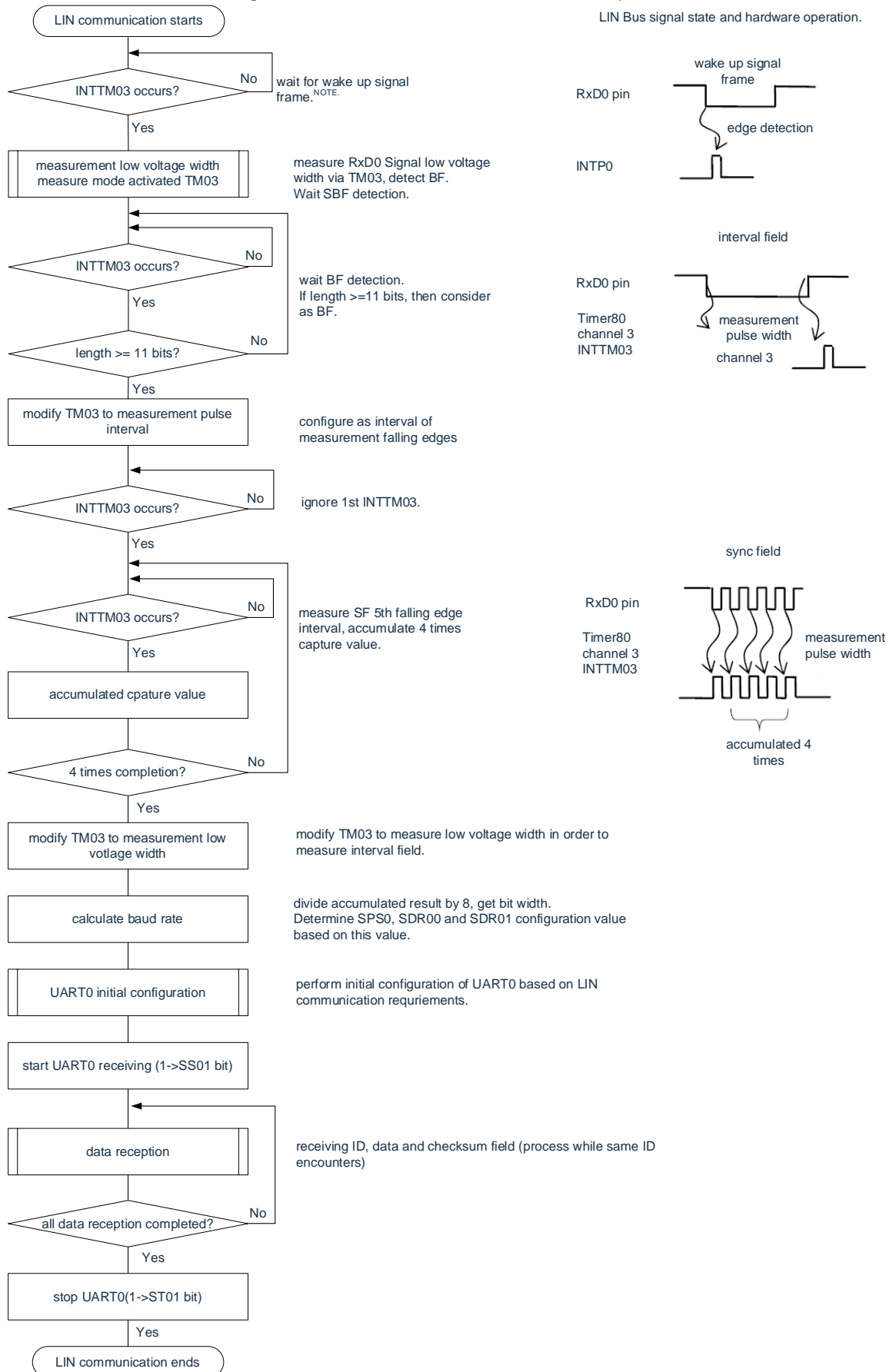
A summary of the receive operations for the LIN is shown in Figure 14-115.

Figure 16-119: Receive operation for LIN



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 16-120: LIN Flowchart for LIN Reception**


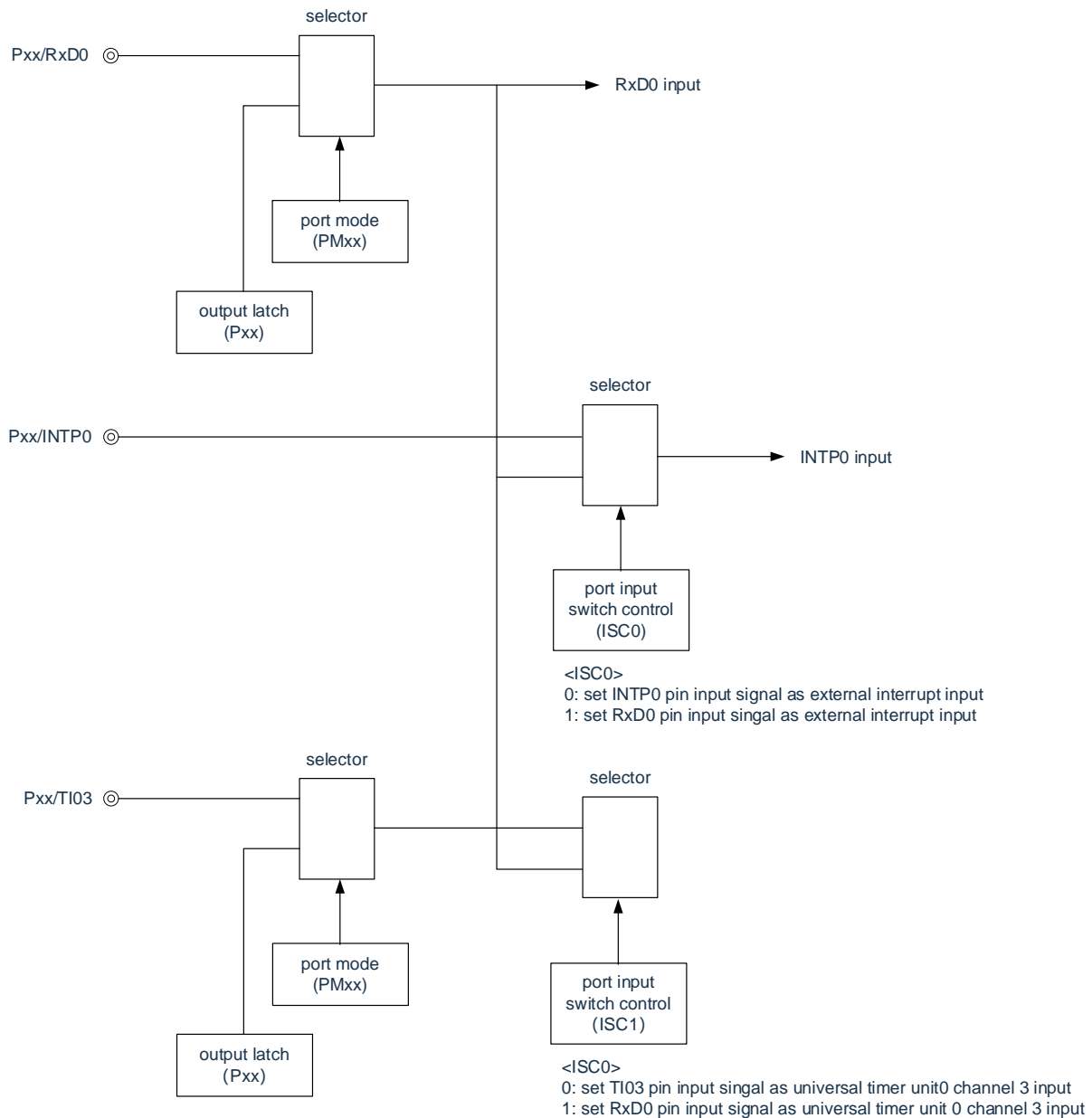
Note: Only during sleep.

The port structure diagram for LIN receive operations is shown in Figure Figure 16-121.

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 synchronization segment 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 16-121: Port Map for LIN Receive Operation



Remark: ISC0, ISC1: Enter the bit0 and bit1 for the Switch Control Register (ISC).

Peripheral features for LIN communication operations are summarized as follows:

<Peripheral Features Used>

- External interrupt (INTP0): Detection of wake-up signal  
Purposes: Detects edges of wake-up signals and the start of communication.
- Channel 3 of the general-purpose timer unit: Detection of Baud Rate Error and Detection of Interval (BF)
- Purposes of use: The length of the synchronization section (SF) is detected and the baud rate error is detected by dividing its length by bits (the interval of the RxD0 input edge is measured by capture mode). A low level width is measured to determine whether it is a spacer (BF).
- Channel 0 and channel 1 (UART0) of general-purpose serial communication unit 0 (SCI0).

## 16.9 Operation of simple I<sup>2</sup>C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication

This is a function for clock synchronization communication with two or more devices by using two lines: serial clock (SCL) and serial data (SDA). This simplified I<sup>2</sup>C is designed for single communication with a device such as EEPROM, flash memory, or A/D converter, and therefore, it functions only as a master.

Make sure by using software, as well as operating the control registers, that the AC specifications of the start and stop conditions are observed.

[Data transmission and reception]

- Master transmission, master reception (only master function with a single master)
- ACK output function <sup>Note</sup> and ACK detection function
- Data length of 8 bits (When an address is transmitted, the address is specified by the higher 7 bits, and the lowest bit is used for R/W control.)
- A start condition and a stop condition are generated by the software.

[Interrupt function]

- Transfer end interrupt

[Error detection flag]

- ACK error

※[Functions not supported by simplified I<sup>2</sup>C]

- Slave transmission, slave reception
- Multi-Master (Quorum Failure Detection)
- Wait detection functions

Note: When the last data is received, if "0" is written to the SDOEmn bit (SDOEm register) to stop the serial communication data output. Refer to "16.9.3(2) Process Flow".

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

The channels 0~1 of SCI0, 0~1 of SCI1 and 0~1of SCI2 are the channels supporting simple I<sup>2</sup>C(IIC00, IIC01, IIC10, IIC11, IIC20, IIC21).

Simple I<sup>2</sup>C(IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) has the following four kinds of communication running:

- address segment sending (See 16.9.1)
- data transmission (See 16.9.2)
- data receiving (See 16.9.3)
- generation of stop condition (See 16.9.4)

## 16.9.1 Address segment sending

Address segment sending is the first transmission operation in I<sup>2</sup>C communication to specify the transmission object (slave device) in particular. After the start condition is generated, the address (7 bits) and the transmission direction (1 bit) are sent as 1 frame.

Simple I <sup>2</sup> C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21
Object channels	Channel 0 of SCI0	Channel 1 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1	Channel 0 of SCI2	Channel 1 of SCI2
Pin used	SCL00 SDA00 <sup>Note 1</sup>	SCL01 SDA01 <sup>Note 1</sup>	SCL10 SDA10 <sup>Note 1</sup>	SCL11 SDA11 <sup>Note 1</sup>	SCL20 SDA20 <sup>Note 1</sup>	SCL21 SDA21 <sup>Note 1</sup>
Interrupt	INTIIC00	INTIIC01	INTIIC10	INTIIC11	INTIIC20	INTIIC21
Error detection flag	Only interrupt at that end of the transfer (no buffer interrupt can be selected).					
Transferred data length	ACK error detection flag(PEFmn)					
Transfer Rate <sup>Note 2</sup>	8-bit (High 7 bits as addresses and low 1 bits as R/W controls)					
	Max.F <sub>MCK</sub> /4[Hz](SDRmn[15:9]≥1)F <sub>MCK</sub> : The operating clock frequency of the object channel, however, must meet the following conditions in each mode of I <sup>2</sup> C: • Max.1MHz(Enhanced fast mode) • Max.400KHz(Quick Mode) • Max.100KHz(Standard Mode)					
Data level	Positive output (default: High level).					
Parity bit	No parity bits.					
Stop bit	Add 1 bit (for ACK reception).					
Data direction	MSB preferred					

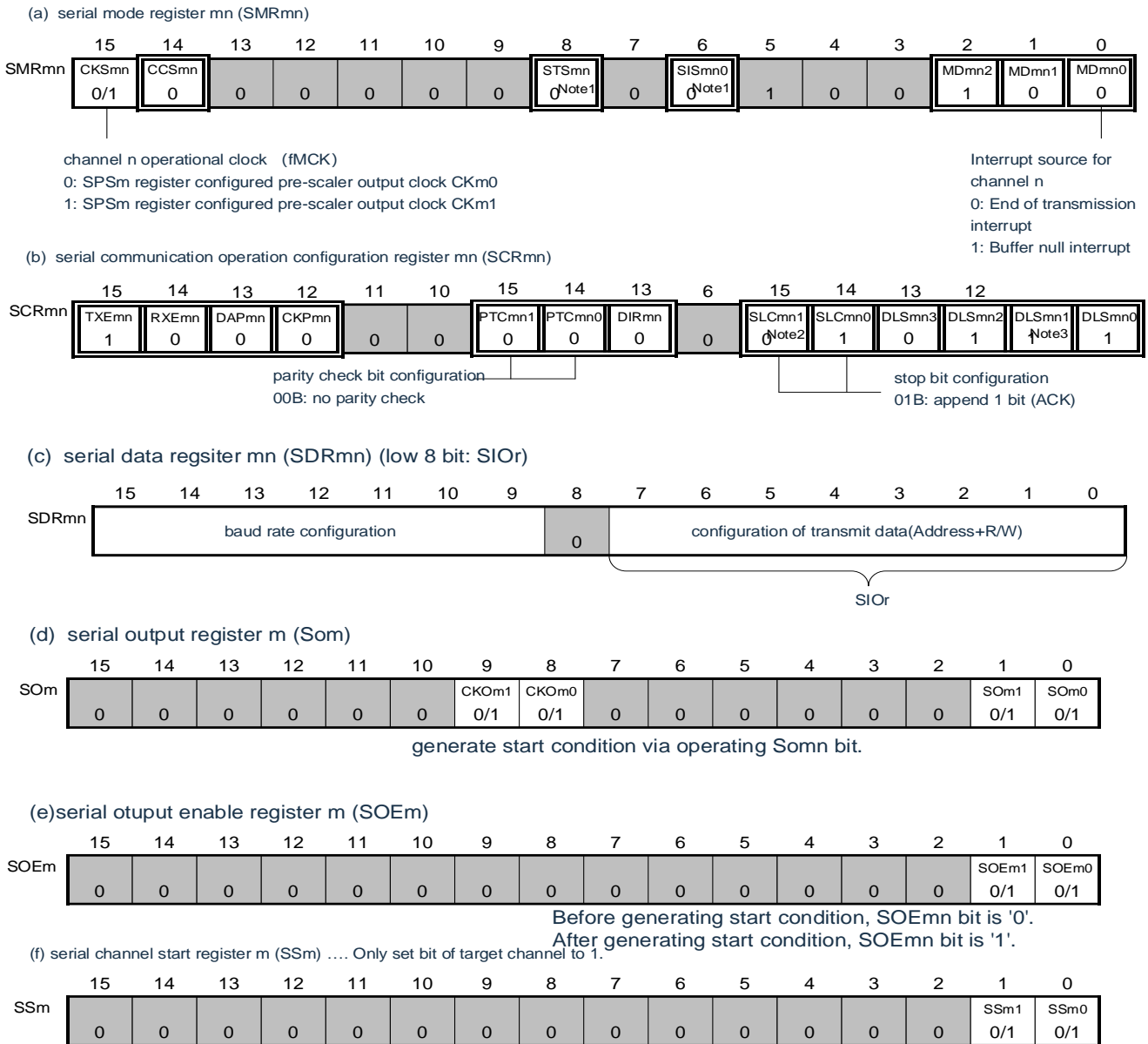
Note 1: To communicate through simple I<sup>2</sup>C, the N-channel drain open output mode (POMxx=1) must be set through the port output mode register (POMxx). For details, refer to “Chapter 2 Port Function”.

Note 2: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

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

### (1) Register setting

Figure 16-122: Example of register setting contents when sending address segments of Simple I<sup>2</sup>C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)



Note 1: Limited to SMR01, SMR11, SMR21.

Note 2: Limited to SCR00, SCR10, SCR20.

Note 3: Limited to SCR00 register and SCR01 register, others fixed to "1".

Remark:

1. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)r: IIC number(r=00, 01, 10, 11, 20, 21)

2.  : Fixed setting in IIC mode.  : setting disabled(initial value).

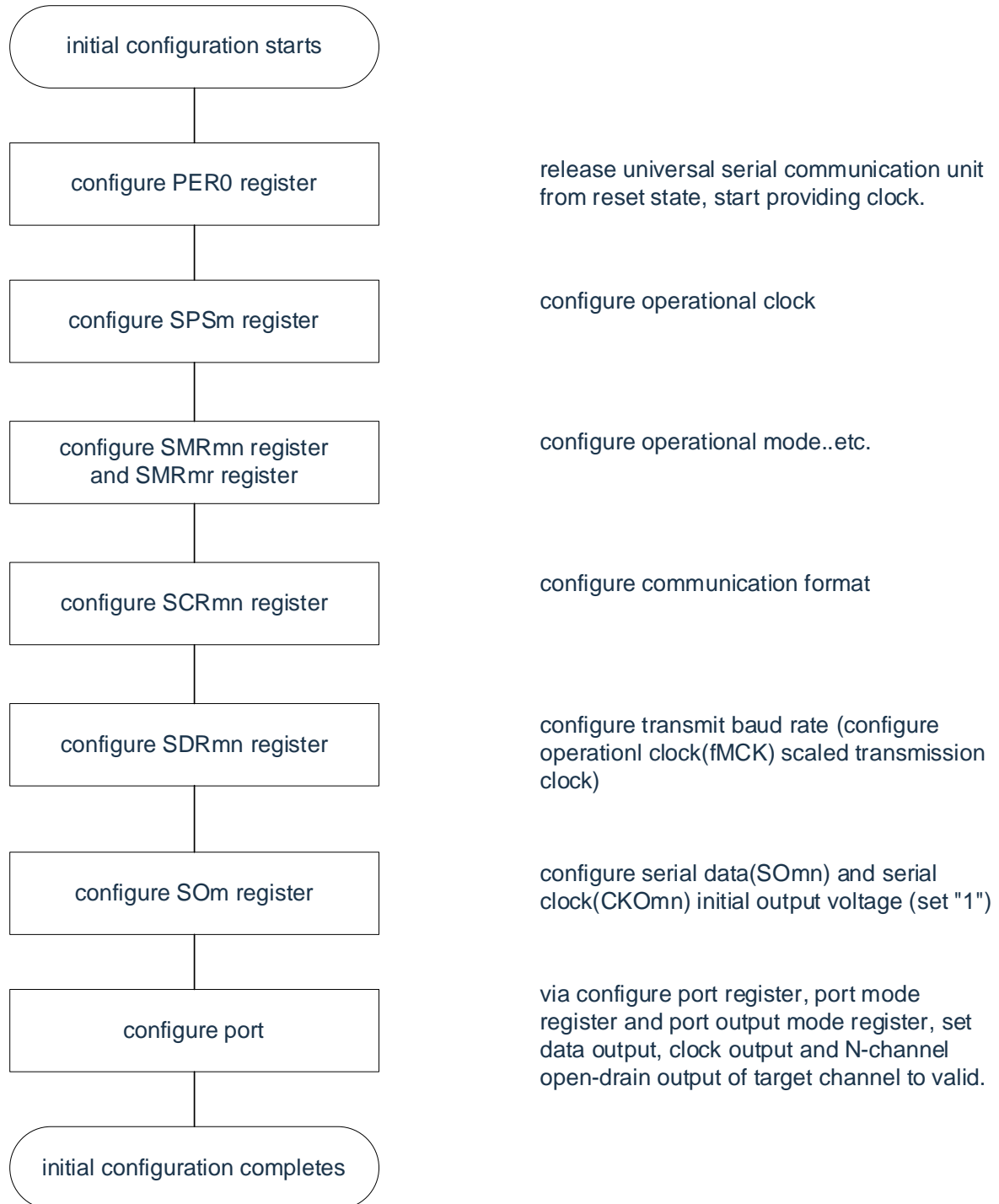
x: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).

0/1: Set "0" or "1" according to the user's purpose.



## (2) Procedure

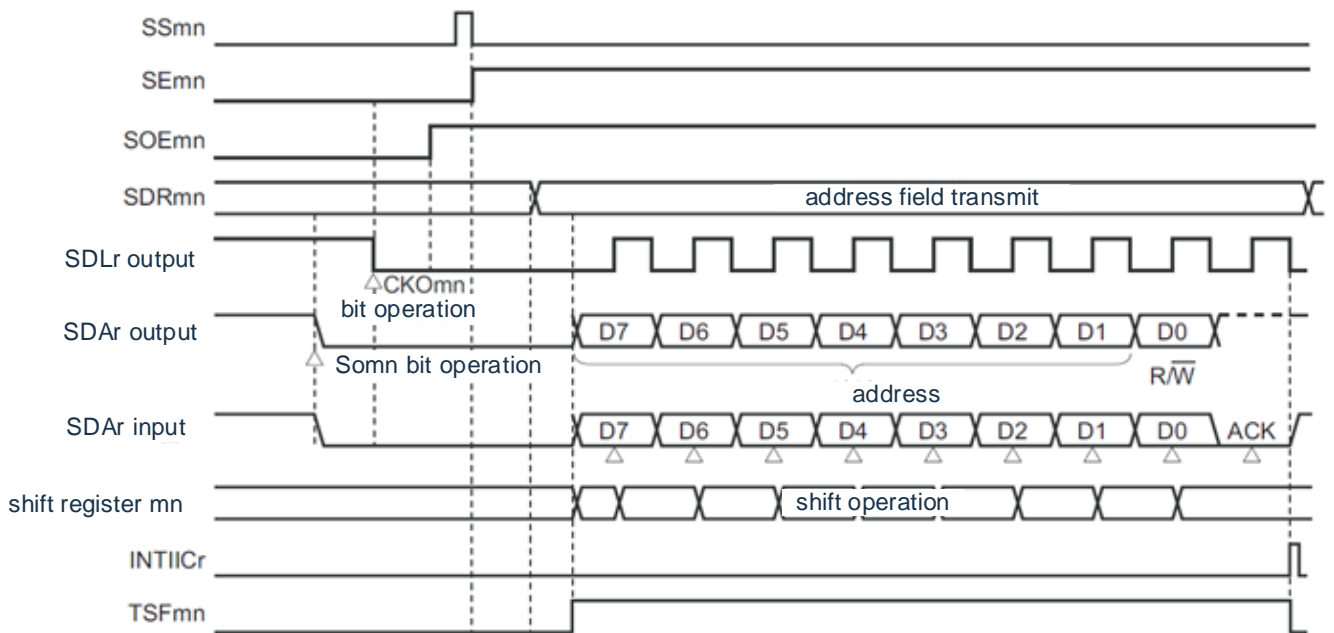
Figure 16-123: Initial set-up steps for address segment transmission



Remark: At the end of the initial set-up, Simple I<sup>2</sup>C (IIC00, IIC01, IIC10, IIC11, IIC20,IIC21) is output-disabled and in a run-stop state.

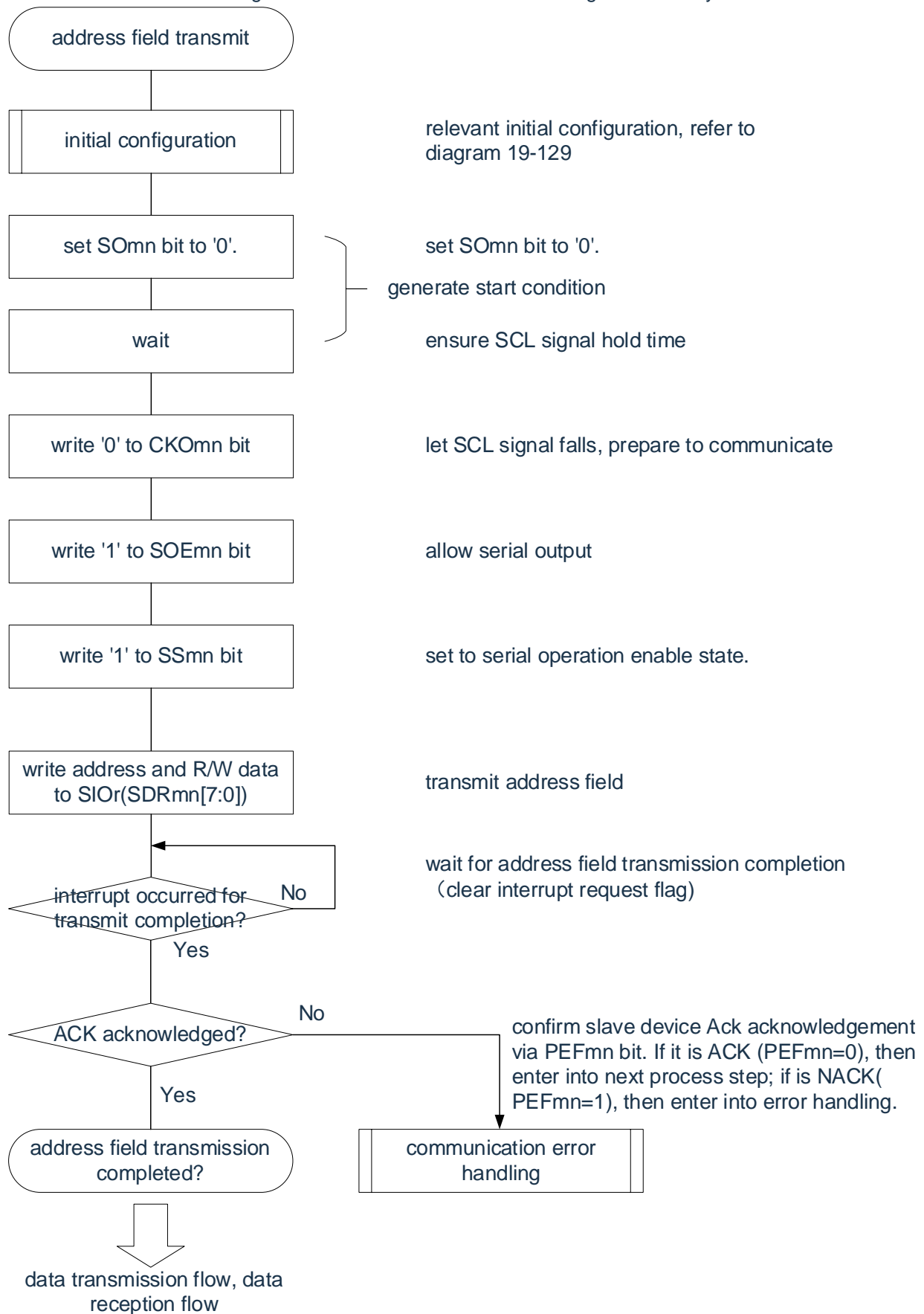
### (3) Process flow

Figure 16-124: Timing Diagram for Address Segment Transmission



Remark: m: unit number(m=0, 1, 2)n: channel number(n=0, 1)r: IIC number(r=00, 01, 10, 11, 20, 21)

Figure 16-125: Flowchart for address segment delivery



## 16.9.2 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.

Simple I <sup>2</sup> C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21
Object channels	Channel 0 of SCI0	Channel 1 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1	Channel 0 of SCI2	Channel 1 of SCI2
Pin used	SCL00 SDA00 <sup>Note 1</sup>	SCL01 SDA01 <sup>Note 1</sup>	SCL10 SDA10 <sup>Note 1</sup>	SCL11 SDA11 <sup>Note 1</sup>	SCL20 SDA20 <sup>Note 1</sup>	SCL21 SDA21 <sup>Note 1</sup>
Interrupt	INTIIC00	INTIIC01	INTIIC10	INTIIC11	INTIIC20	INTIIC21
	Only interrupt at that end of the transfer (no buffer interrupt can be selected).					
Error detection flag	ACK error flag (PEFmn)					
Transferred data length	8-bit					
Transfer Rate <sup>Note 2</sup>	Max. $F_{MCK}/4[Hz](SDRmn[15:9] \geq 1)F_{MCK}$ : The operating clock frequency of the object channel, however, must meet the following conditions in each mode of I <sup>2</sup> C: <ul style="list-style-type: none"> <li>• Max.1MHz(Enhanced fast mode)</li> <li>• Max.400KHz(Quick mode)</li> <li>• Max.100KHz(Standard mode)</li> </ul>					
Data level	Positive output (default: High level).					
Parity bit	No parity bits.					
Stop bit	Add 1 bit (for ACK reception).					
Data direction	MSB preferred					

Note 1: To communicate via Simple I<sup>2</sup>C, the N-channel open drain output mode (POMxx=1) must be set via the Port Output Mode Register (POMxx).

Note 2: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

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

### (1) Register setting

Figure 16-126: Example of register setting contents for simple I<sup>2</sup>C 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
SMRmn	CKSmn	CCSmn						STSmn		SISmn0				MDmn2	MDmn1	MDmn0
	0/1	0	0	0	0	0	0	0 <sup>Note1</sup>	0	0 <sup>Note1</sup>	1	0	0	1	0	0

(b) serial communication operation configuration register mn (SCRmn).....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
SCRmn	TXEmn	RXEmn	DAPmn	CKPmn				PTCmn1	PTCmn0	DIRmn		SLCmn1	SLCmn0	DLSmn3	DLSmn2	DLSmn1
	1	0	0	0	0	0	0	0	0	0	0	0 <sup>Note2</sup>	1	0	1	1 <sup>Note3</sup>

(c) serial data register mn (SDRmn) (low 8 bit: SIO<sub>r</sub>) .....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
SDRmn	baud rate configuration <sup>Note4</sup>								0	configuration of transmit data						
										SIO <sub>r</sub>						

(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							CKOm1	CKOm0							SOm1	SOm0
	0	0	0	0	0	0	0/1 <sup>Note5</sup>	0/1 <sup>Note5</sup>	0	0	0	0	0	0	0/1 <sup>Note5</sup>	0/1 <sup>Note5</sup>

(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															SOEm1	SOEm0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	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
SSm															SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0/1	0/1

Note 1: Limited to SMR01, SMR11, SMR21 registers only.

Note 2: Limited to SCR00, SCR10, SCR20 registers only.

Note 3: Limited to SCR00, SCR01, SCR10, SCR11, SCR20, SCR21 registers, other fixed to "1".

Note 4: No set-up is required because the address segment is already set when it is sent.

Note 5: During a communication run, the value changes due to communication data.

Remark:

1. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)r: IIC number(r=00, 01, 10, 11, 20, 21)

2.   : Fixed setting in IIC mode.   : setting disabled(initial value).

x: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).

0/1: Set "0" or "1" according to the user's purpose.

## (2) Process flow

Figure 16-127: Timing diagram of data transmission

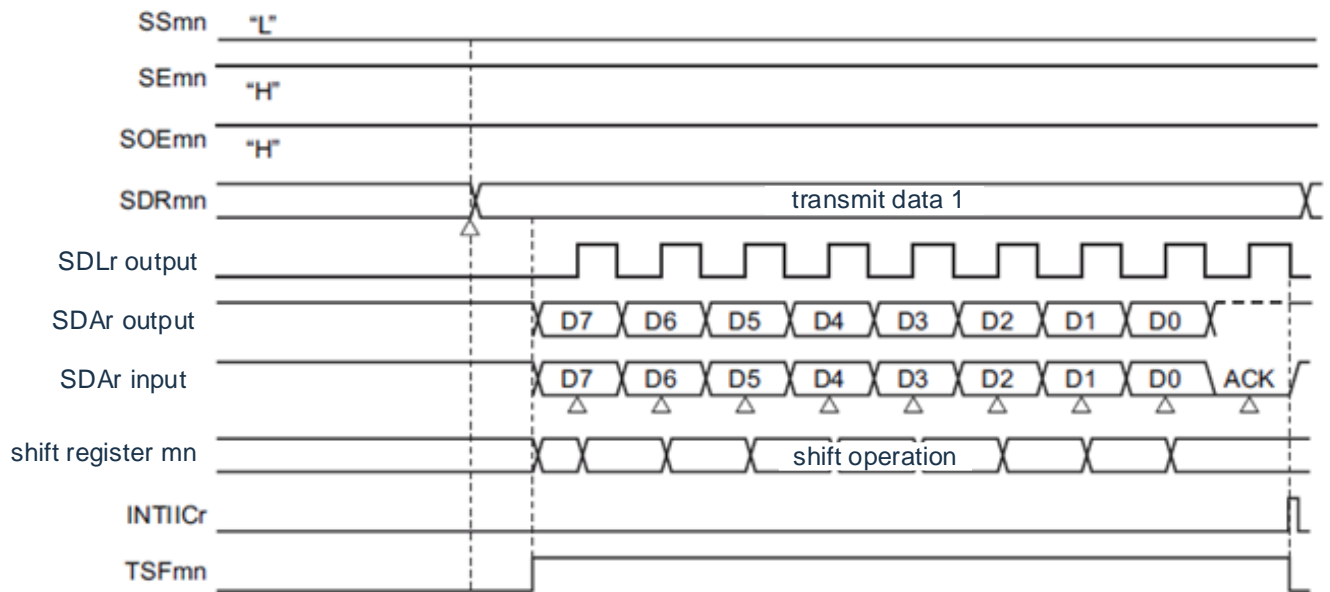
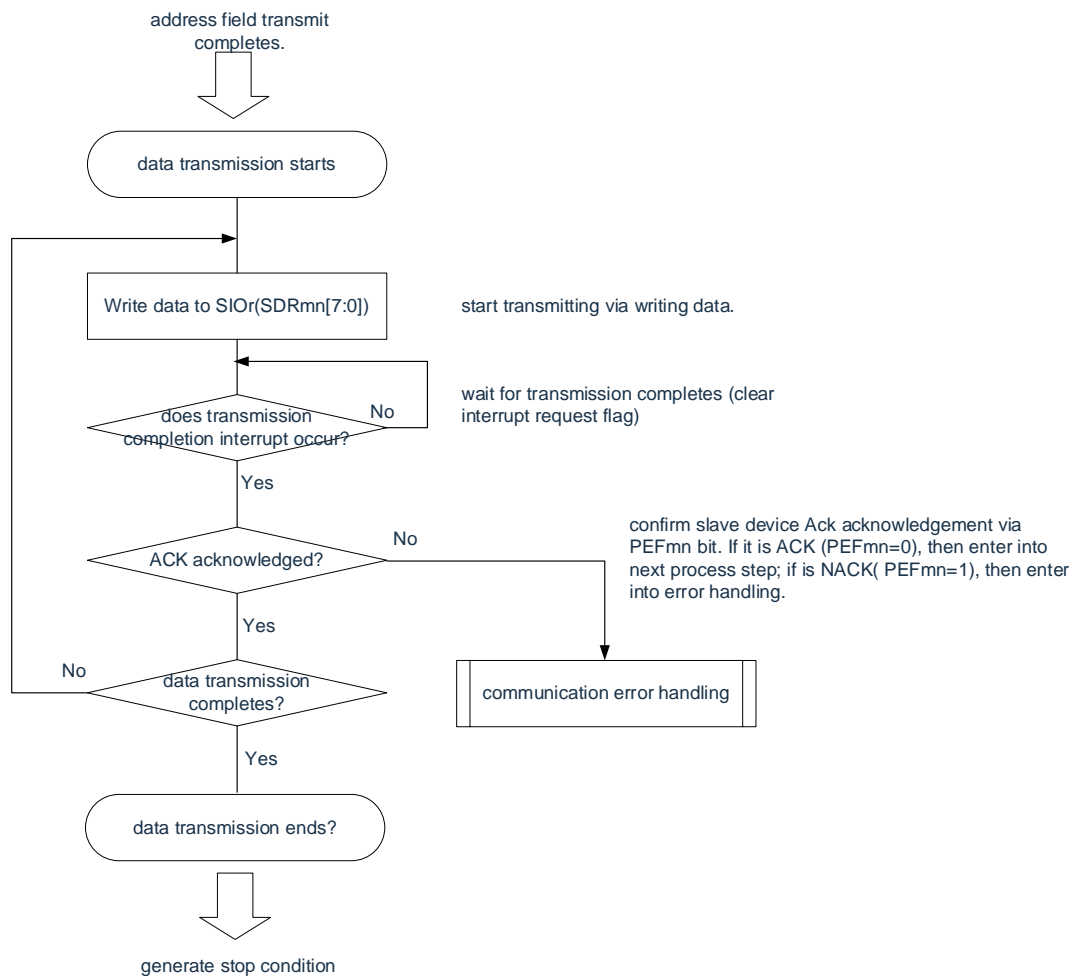


Figure 16-128: Flow chart for data delivery



### 16.9.3 Data reception

Data reception is a run that receives data from a transfer object (slave) after sending an address segment. A stop condition is generated and the bus is released after receiving all the data from the object slave.

Simple I <sup>2</sup> C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21
Object channels	Channel 0 of SCI0	Channel 1 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1	Channel 0 of SCI2	Channel 1 of SCI2
Pin used	SCL00 SDA00 <sup>Note 1</sup>	SCL01 SDA01 <sup>Note 1</sup>	SCL10 SDA10 <sup>Note 1</sup>	SCL11 SDA11 <sup>Note 1</sup>	SCL20 SDA20 <sup>Note 1</sup>	SCL21 SDA21 <sup>Note 1</sup>
Interrupt	INTIIC00	INTIIC01	INTIIC10	INTIIC11	INTIIC20	INTIIC21
	Only interrupt at that end of the transfer (no buffer interrupt can be selected).					
Error detection flag	There are only overflow error detection flags(OVFmn).					
Transferred data length	8-bit					
Transfer Rate <sup>Note 2</sup>	$\text{Max.F}_{\text{MCK}}/4[\text{Hz}](\text{SDRmn}[15:9] \geq 1)\text{F}_{\text{MCK}}$ : The operating clock frequency of the object channel, however, must meet the following conditions in each mode of I <sup>2</sup> C: <ul style="list-style-type: none"> <li>• Max.1MHz(Enhanced fast mode)</li> <li>• Max.400KHz(Quick mode)</li> <li>• Max.100KHz(Standard mode)</li> </ul>					
Data level	Positive output (default: High level).					
Parity bit	No parity bits.					
Stop bit	Add 1 bit (ACK send).					
Data direction	MSB preferred					

Note 1: To communicate via simple I<sup>2</sup>C, the N-channel open drain output mode (POMxx=1) must be set via the Port Output Mode Register (POMxx).

Note 2: It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

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

## (1) Register setting

Figure 16-129: Example of register setting contents for simple I<sup>2</sup>C 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 <sup>Note1</sup>	0	0 <sup>Note1</sup>	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	15	14	13	6	15	14	13	12		
SCRmn	TXEmn	RXEmn	DAPmn	CKPmn			PTCmn1	PTCmn0	DIRmn		SLCmn1	SLCmn0	DLSmn3	DLSmn2	DLSmn1	DLSmn0
	0	1	0	0	0	0	0	0	0	0	0 <sup>Note2</sup>	1	0	1	1 <sup>Note3</sup>	1

(c) serial data register mn (SDRmn) (low 8 bit: SIO<sub>r</sub>)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration <sup>Note4</sup>								0	virtual transmit data configuration (FFH)						
										SIO <sub>r</sub>						

(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							CKOm1	CKOm0							SOM1	SOM0
	0	0	0	0	0	0	0/1 <sup>Note5</sup>	0/1 <sup>Note5</sup>	0	0	0	0	0	0	0/1 <sup>Note5</sup>	0/1 <sup>Note5</sup>

(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															SOEm1	SOEm0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	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															SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0/1	0/1

Note 1: Limited to SMR01, SMR11, SMR21 registers only.

Note 2: Limited to SCR00, SCR10, SCR20 registers only.

Note 3: Limited to SCR00, SCR01, SCR10, SCR11, SCR20, SCR21 registers, other fixed to "1".

Note 4: No set-up is required because the address segment is already set when it is sent.

Note 5: During a communication run, the value changes due to communication data.

Remark:

1. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)r: IIC number(r=00, 01, 10, 11, 20, 21)

2.   : Fixed setting in IIC mode.   : setting disabled(initial value).

x: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).

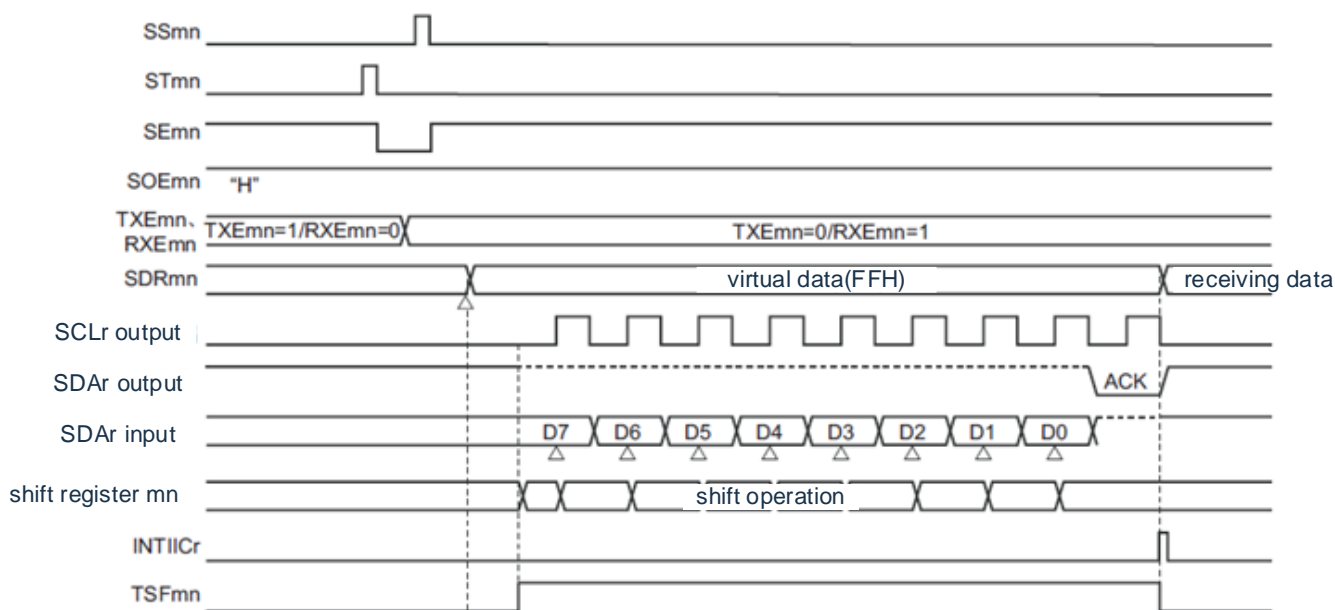
0/1: Set "0" or "1" according to the user's purpose.



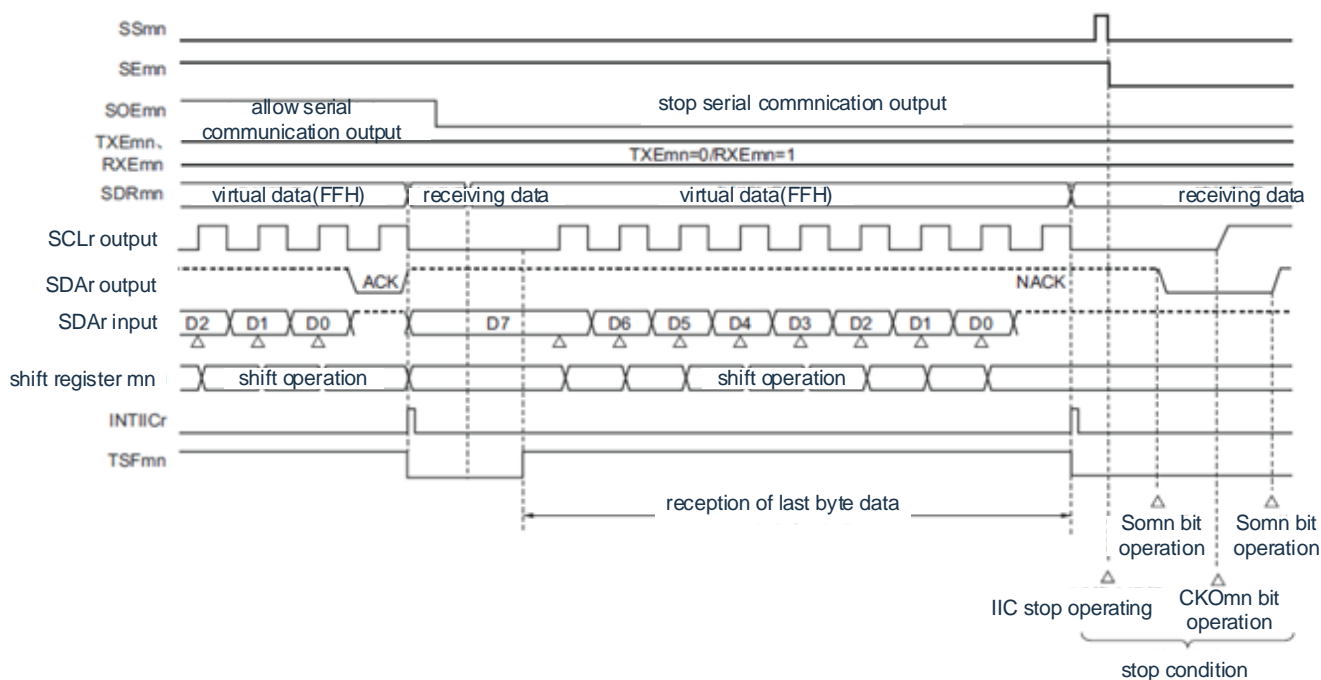
Figure16-130: Timing diagram for data reception

Figure16-130: Timing diagram for data reception

(a) Start of receiving data

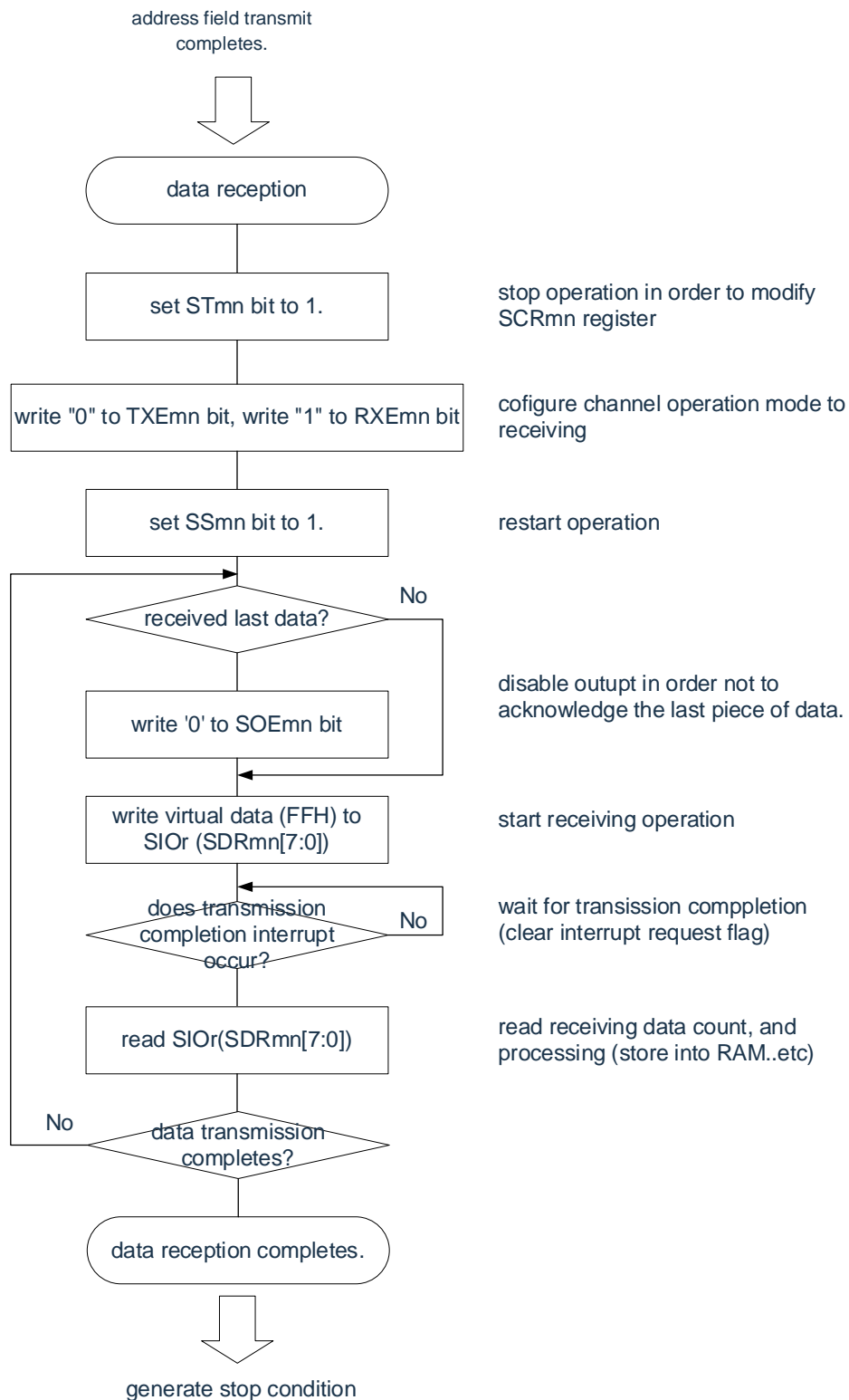


(b) Status of receipt of final data



Remark: m: unit number(m=0, 1, 2)n: channel number(n=0, 1)r: IIC number(r=00, 01, 10, 11, 20, 21)

Figure 16-131: Flow chart for data reception



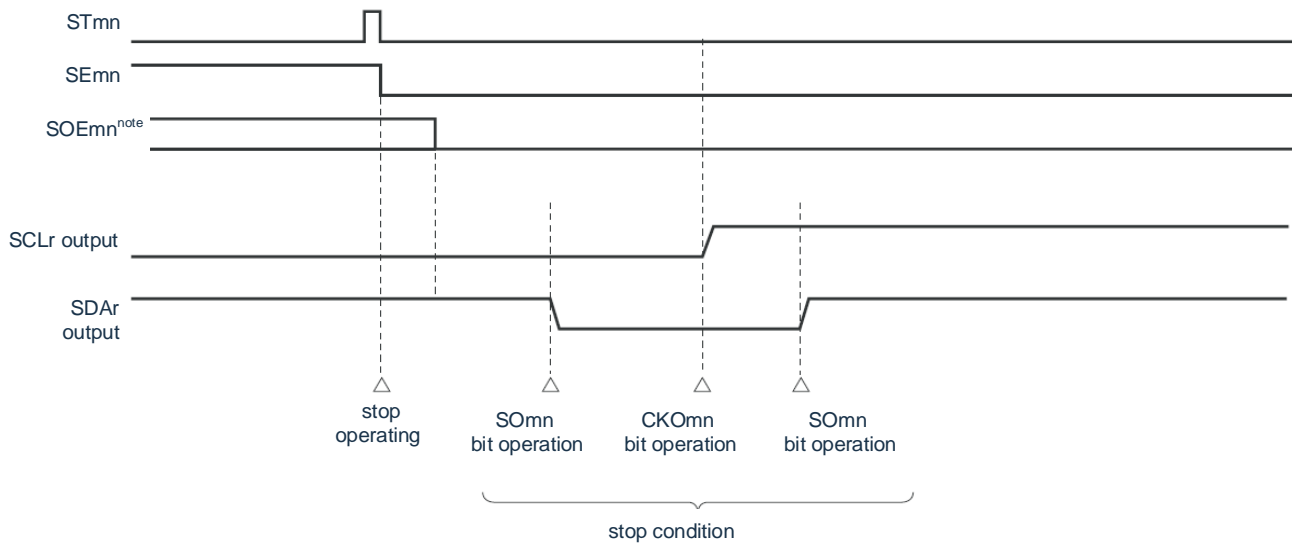
Notice: No ACK(NACK) is output when the last data is received. Thereafter, the communication is terminated by stopping the STmn bit of the serial channel stop register m (STm) to '1'.

## 16.9.4 Generation of stop condition

After sending and receiving all the data with the object slave, a stop condition is generated and the bus is released.

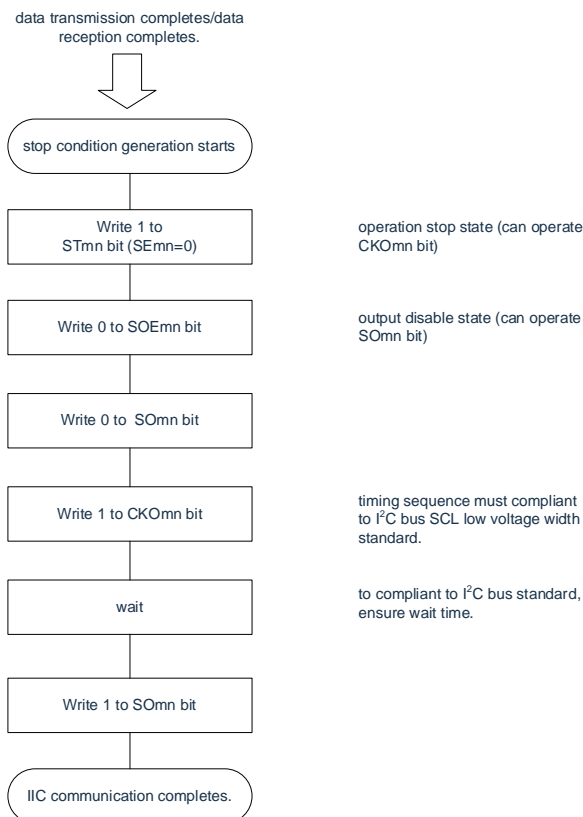
### (1) Process flow

Figure 16-132: Timing diagram for generating stop condition



Note: At the time of receiving, set the SOEmn bit of the serial output enable register m (SOEm) to "0".

Figure 16-133: Flow chart for generating stop condition



## 16.9.5 Calculation of transfer rate

The transfer rate for simple I<sup>2</sup>C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication can be

$$(\text{Transfer Rate}) = \{ \text{Runtime Clock } (F_{\text{MCK}}) \text{ Frequency} \} \div (\text{SDRmn}[15:9] + 1) \div 2$$

calculated using the following formula.

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 simple I<sup>2</sup>C is 50%. In I<sup>2</sup>C 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 I<sup>2</sup>C bus. You must set a value for SDRmn[15:9] that meets the I<sup>2</sup>C bus specification.

Remark:

1. Because the value of SDRmn[15:9] is the value of bit15~9 of serial data register (SDRmn) (0000001B~1111111B), it is 1~127.
2. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)

The runtime clock ( $f_{\text{MCK}}$ ) depends on the bit15 (CKSmn bit) of the serial clock selection register m (SPSm) and the serial mode register mn (SMRmn).

Table 16-6: Simple I<sup>2</sup>C Operating Clock Selection

SMRmn Register	SPSm register								Runtime Clock (F <sub>MCK</sub> ) <sup>Note</sup>	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		F <sub>CLK</sub> =32MHz in operation
0	X	X	X	X	0	0	0	0	F <sub>CLK</sub>	32MHz
	X	X	X	X	0	0	0	1	F <sub>CLK</sub> /2	16MHz
	X	X	X	X	0	0	1	0	F <sub>CLK</sub> /2 <sup>2</sup>	8MHz
	X	X	X	X	0	0	1	1	F <sub>CLK</sub> /2 <sup>3</sup>	4MHz
	X	X	X	X	0	1	0	0	F <sub>CLK</sub> /2 <sup>4</sup>	2MHz
	X	X	X	X	0	1	0	1	F <sub>CLK</sub> /2 <sup>5</sup>	1MHz
	X	X	X	X	0	1	1	0	F <sub>CLK</sub> /2 <sup>6</sup>	500KHz
	X	X	X	X	0	1	1	1	F <sub>CLK</sub> /2 <sup>7</sup>	250KHz
	X	X	X	X	1	0	0	0	F <sub>CLK</sub> /2 <sup>8</sup>	125KHz
	X	X	X	X	1	0	0	1	F <sub>CLK</sub> /2 <sup>9</sup>	62.5KHz
	X	X	X	X	1	0	1	0	F <sub>CLK</sub> /2 <sup>10</sup>	31.25KHz
	X	X	X	X	1	0	1	1	F <sub>CLK</sub> /2 <sup>11</sup>	15.63KHz
1	0	0	0	0	X	X	X	X	F <sub>CLK</sub>	32MHz
	0	0	0	1	X	X	X	X	F <sub>CLK</sub> /2	16MHz
	0	0	1	0	X	X	X	X	F <sub>CLK</sub> /2 <sup>2</sup>	8MHz
	0	0	1	1	X	X	X	X	F <sub>CLK</sub> /2 <sup>3</sup>	4MHz
	0	1	0	0	X	X	X	X	F <sub>CLK</sub> /2 <sup>4</sup>	2MHz
	0	1	0	1	X	X	X	X	F <sub>CLK</sub> /2 <sup>5</sup>	1MHz
	0	1	1	0	X	X	X	X	F <sub>CLK</sub> /2 <sup>6</sup>	500KHz
	0	1	1	1	X	X	X	X	F <sub>CLK</sub> /2 <sup>7</sup>	250KHz
	1	0	0	0	X	X	X	X	F <sub>CLK</sub> /2 <sup>8</sup>	125KHz
	1	0	0	1	X	X	X	X	F <sub>CLK</sub> /2 <sup>9</sup>	62.5KHz
	1	0	1	0	X	X	X	X	F <sub>CLK</sub> /2 <sup>10</sup>	31.25KHz
	1	0	1	1	X	X	X	X	F <sub>CLK</sub> /2 <sup>11</sup>	15.63KHz
Others:									Settings are disabled.	

Note: When you change the clock selected as FCLK (change the value of the system clock control register (CKC)), you must stop the operation of the general-purpos serial communication unit (SCI) (serial channel stop register m (STm)=000FH) after making changes.

Remark:

1. X: Ignore
2. m: unit number(m=0, 1, 2)n: channel number(n=0, 1)

An example of setting the I<sup>2</sup>C transfer rate at F<sub>MCK</sub>=F<sub>CLK</sub>=32MHz is shown below:

I <sup>2</sup> C transfer mode (Expected Transfer Rate)	F <sub>CLK</sub> =32MHz			
	Runtime Clock (F <sub>MCK</sub> )	SDRmn [15:9]	Calculated transfer rate	Error with expected transfer rate
100KHz	F <sub>CLK</sub> /2	79	100KHz	0.0%
400KHz	F <sub>CLK</sub>	41	380KHz	5.0% <sup>Note</sup>
1MHz	F <sub>CLK</sub>	18	0.84MHz	16.0% <sup>Note</sup>

Note: The error cannot be set to '0%' because the SCL signal has a 50% duty cycle.

## 16.9.6 Processing steps when an error occurs in a simple I<sup>2</sup>C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication process

The processing steps when an error occurs during a simple I<sup>2</sup>C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication are shown in Figures Figure 15-134 and Figure 16-135.

Figure 15-134: Handling steps when overflow errors occur

Software operation	Hardware status	Remark
Read the serial data register mn (SDRmn).	The BFFmn bit of the SSRmn register is "0" and the channel n is in a receiver state.	This is to prevent an overflow error from occurring to end the next receipt during error handling.
Read the serial status register mn (SSRmn).		The type of error is determined and the read value is used to clear the error flag.
Clear trigger register mn for serial flag.	Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors in the read operation can only be cleared.

Figure 16-135: Processing steps when an ACK error occurs in a simple I<sup>2</sup>C mode

Software operation	Hardware status	Remark
Read the serial status register mn (SSRmn).		Determine the type of error and the read value is used to clear the error flag
Write serial flag clear trigger register mn	Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors in the read operation can only be cleared.
Stop the Serial Channel from Register m (STm)	The serial channel allows the SEMn bit of the status register m (SEm) to be "0" and channel n to be running stopped.	The slave device is not ready to receive because no ACK is returned. Accordingly, a stop condition is generated and the bus is released, communication is started again from the start condition, or a restart condition can also be generated and restarted from the address transmission.
Generate a stop condition		
Generate a start condition.		
Set the SSmn bit of the serial channel start register m (SSm) to "1"	The serial channel allows the SEMn bit of the status register m (SEm) to be "1" and channel n to be operational.	

Note: m: unit number(m=0, 1, 2) n: channel number(n=0, 1)r: IIC number(r=00, 01, 10, 11, 20, 21)

# Chapter 17 Serial Interface SPI

## 17.1 Function of SPI

This product is equipped with a SPI, and has the following two modes.

(1) Run stop mode

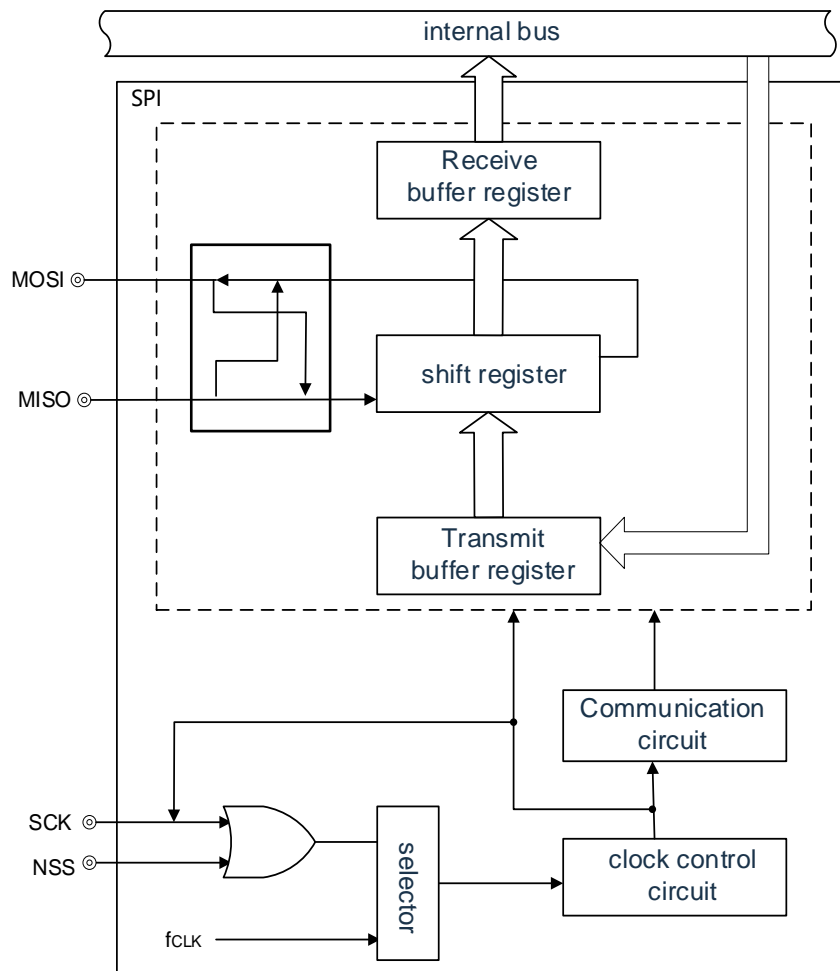
This is a mode used when serial transfer is not in progress and reduces power consumption.

(2) 3-wire serial I/O mode

This mode transmits 8-bit or 16-bit data to multiple devices via three lines of serial clock (SCK) and serial data bus (MISO and MOSI).

## 17.2 Structure of SPI

Figure 17-1: Diagram of SPI



## 17.3 Registers for controlling SPI

The SPI is controlled by the following registers.

- Peripheral enable register 1 (PER1)
- Serial operating mode register (SPIM)
- Serial clock selection register (SPIC)
- Transmit buffer register (SOTB)
- Receive Buffer Register (SIO)

### 17.3.1 Peripheral enable register 1 (PER1)

The PER1 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 SPI function, SPIHSOEN must be set to "1".

For details, see "4.3.6 Peripheral enable register 1 (PER1)"



## 17.3.2 SPI operating mode register (SPIM)

SPIM is used to select the mode of operation and control the permission or prohibition of the operation.

SPIM can be set by an 8-bit memory manipulation instruction.

A reset signal is generated to clear the register to 00H.

Figure 17-2: Format of SPI operating mode register (SPIM)

	Address: 0x40047800			After reset: 00H			R/W <sup>Note 1</sup>	
Symbol	7	6	5	4	3	2	1	0
SPIM	SPIE	TRMD	NSSE	DIR	INTMD	DLS	RECMD	-

SPIE	SPI running enable
0	Stop running.
1	Enable to run.

TRMD <sup>Note 3</sup>	Transmit/Receive mode control
0	Receive mode
1	Send/Receive mode

NSSE <sup>Note 4</sup>	NSS pin uses selection
0	The NSS pin is not used
1	Use the NSS pin

DIR	Data transfer order selection
0	Perform MSB-first input/output.
1	Perform LSB-first input/output.

INTMD	Interrupt source selection
0	The end of transfer is interrupted
1	The send buffer is empty interrupt

DLS	The setting of the data length
0	8 bits of data length
1	16-bit data length

RECMD	Mode selection for receive mode
0	Single receive
1	Continuous reception

Note 1: When SPTF=1 (during serial communication), rewriting of TRMD, DIR, NSSE is prohibited.

Note 2: The MO or SO output is fixed low when the TRMD is 0.

Note 3: Before setting the bit to 1, fix the NSS pin input level to 0 or 1.

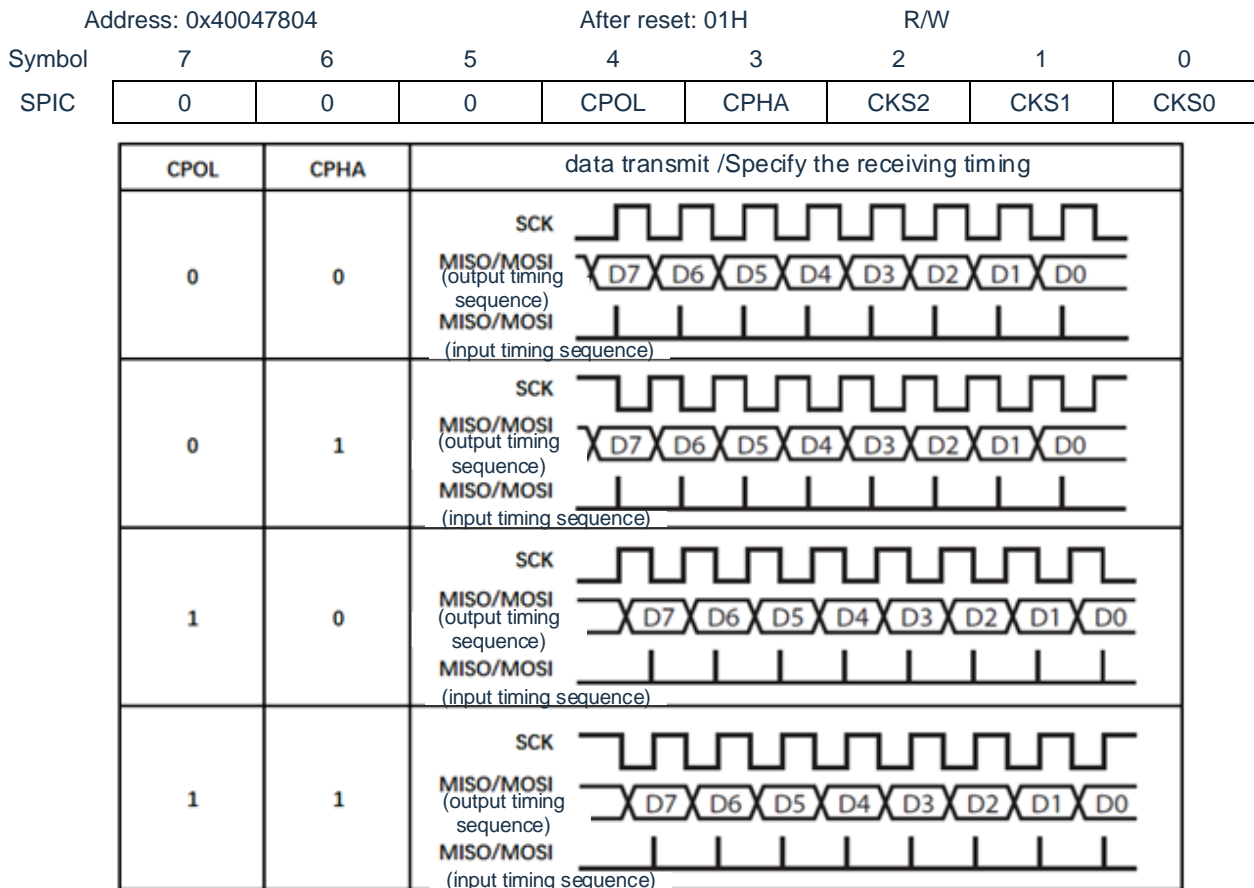
### 17.3.3 SPI clock selection register (SPIC)

This register specifies the timing of data send/receive and sets the serial clock.

can be set by an 8-bit storage operation instruction.

A reset signal is generated to clear the register to 01H.

Figure 17-3: Format of clock selection register (SPIC)



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	An external clock input from SCK	Slave mode

Notice:

1. Write TOPICn is prohibited when SPIE n=1 (operation enabled).
2. The phase type of the data clock after reset is Type 1.

### 17.3.4 SPI status register (SPIS)

The SPIS register is used to confirm the communication status of the SPI.

SPIS can be read by an 8-bit storage operation instructionsa.

A reset signal is generated to clear the register to 00H.

Figure 17-4: Format of SPI status register (SPIS)

	Address: 0x40047810			After reset: 00H			R	
Symbol	7	6	5	4	3	2	1	0
SPIS	-	-	-	-	-	-	SDRIF	SPTF

SDRIF	Receive buffer non-null flag bits
0	There is no new valid data in the receive cache
1	There is valid data received in the receive cache. When the register SDRIF is read, the bit is cleared to 0

SPTF <sup>Note 1</sup>	Communication status flag bits
0	Communication interrupt
1	Communication is in progress

Note 1: When SPTF=1 (during serial communication), rewriting of TRMD, DIR, NSSE is prohibited.

### 17.3.5 Transmit buffer register (SOTB)

The register is set to send data.

When setting bits 7 (SPIE) and bit 6 (TRMD) of the serial operating mode register (SPIM) to 1 When sending/receiving starts by writing data to SOTB.

Serial I/O shift registers convert data in SOTB from parallel to serial data and output to the serial output pin.

SOTB can be written to or read with 8-bit or 16-bit storage operation instructions.

A reset signal is generated to clear the register to 0000H.

Figure 17-5: Format of transmit buffer register (SOTB)

	Address: 0x40047808						After reset: 0000H					R/W				
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOTB	SOTB															

### 17.3.6 Receive buffer register (SIO)

This register stores the received data.

If bit 6 (TRMD) of the serial operating mode register (SPIM) is set to 0, the reception begins by reading data from the SDRI.

During reception, data is read from the serial input pin into SIO.

SIO can be read with 8-bit or 16-bit memory manipulation instructions.

A reset signal is generated to clear the register to 0000H.

Figure 17-6: Format of receive buffer register (SIO)

	Address: 0x4004780C						After reset: 0000H				R					
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIO	SIO															

## 17.4 Operation of SPI

In 3-wire serial I/O mode, data is sent or received in 8-bit or 16-bit units. The data is sent or received synchronously with the serial clock.

After communication begins, bit 0 (SPTF) of SPIT is set to 1. When the communication of data is complete, set the communication completion interrupt request flag (SPIIF) and clear SPTF to 0. Then enable the next communication.

Notice:

1. When SPTF=1 (during serial communication), access to control registers and data registers is prohibited.
2. It must be used within the range that satisfies the SCLK Cycle Time (tKCY) characteristics. For details, please refer to the data sheet.

## 17.4.1 Master transmission and reception

If the bit 6 (TRMDn) of the serial operating mode register (SPIMn) is 1, data can be sent or received. When a value is written to the transmit buffer register (SDR0n), send/receive starts.

### (1) Procedure

Figure 17-7: Initial setup steps for master send/receive

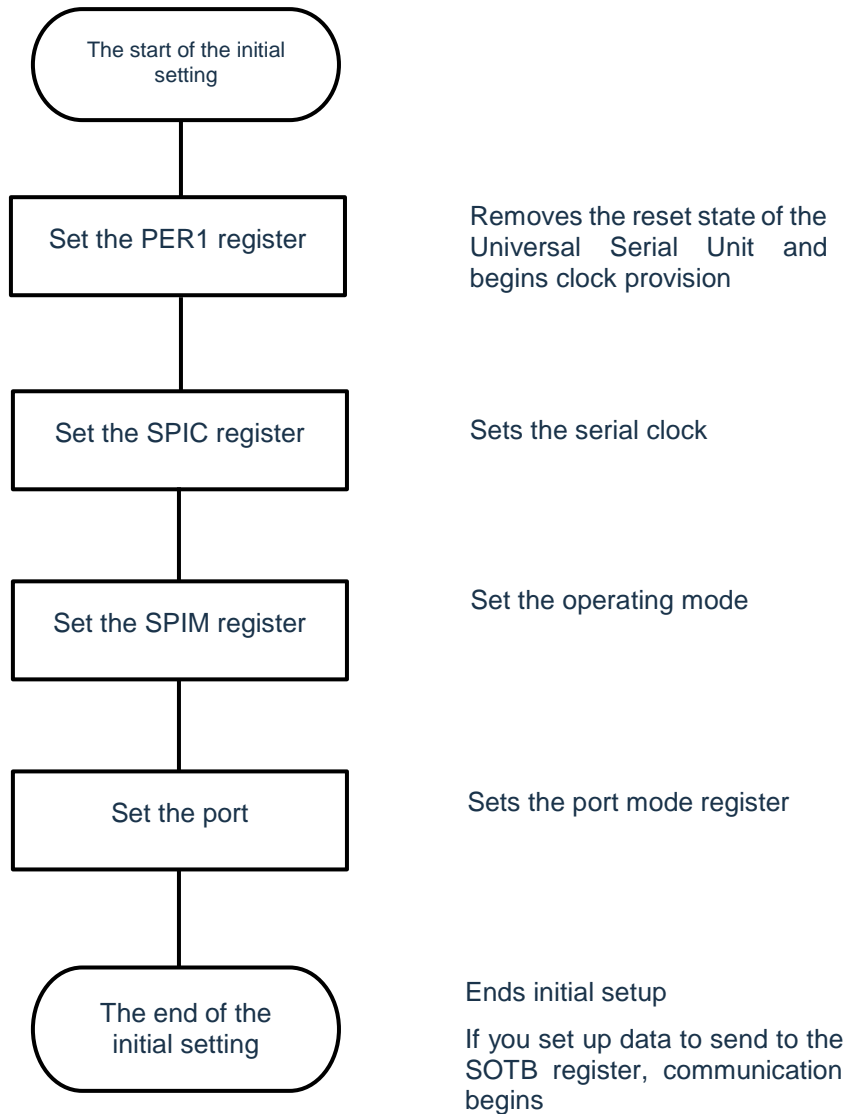
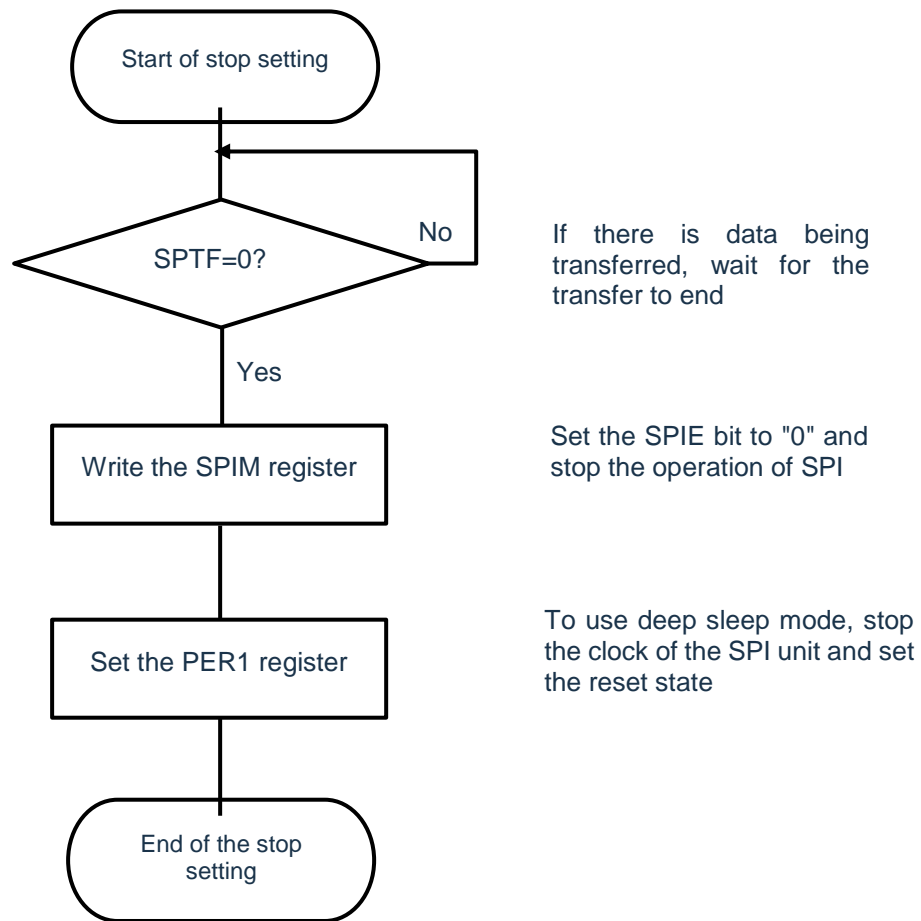


Figure 17-8: Stop step of the master transmit/receive



## (2) Processing

Figure 17-9: Timing diagram of transmit/receive (single transmit mode) (INTMD=0,CPHA=1, CPOL=1)

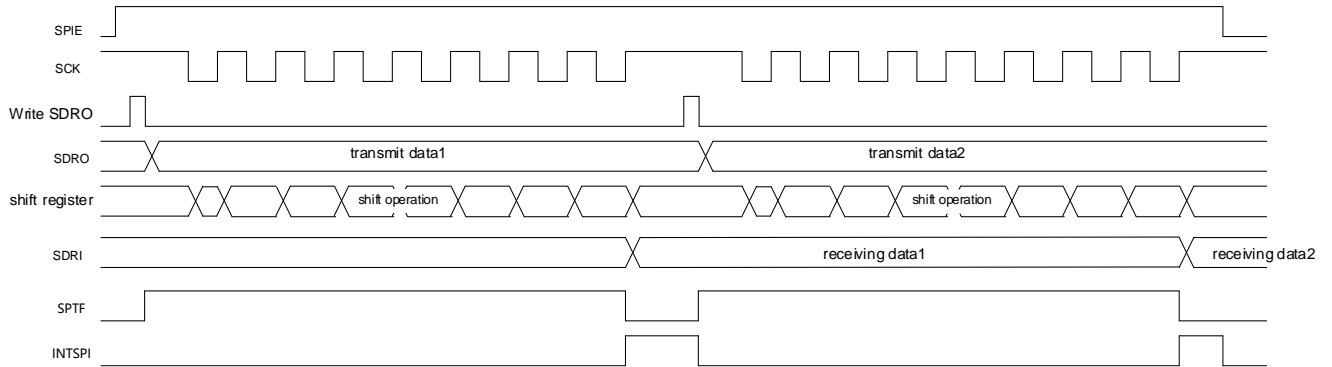
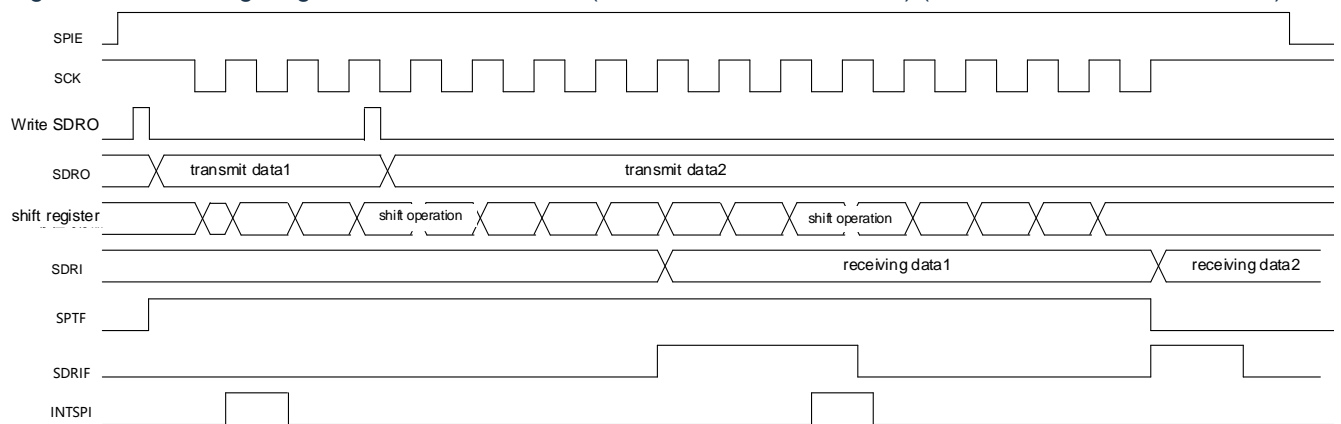


Figure 17-10: Timing diagram of transmit/receive (continuous transmit mode) (INTMD=1,CPHA=1, CPOL=1)





## 17.4.2 Master reception

If bit 6 (TRMD) of the serial operating mode register (SPIM) is 0, only data can be received. When data is read from the receive buffer register (SIO), the reception begins

### (1) Procedure

Figure 17-11: Initial setup steps for master reception

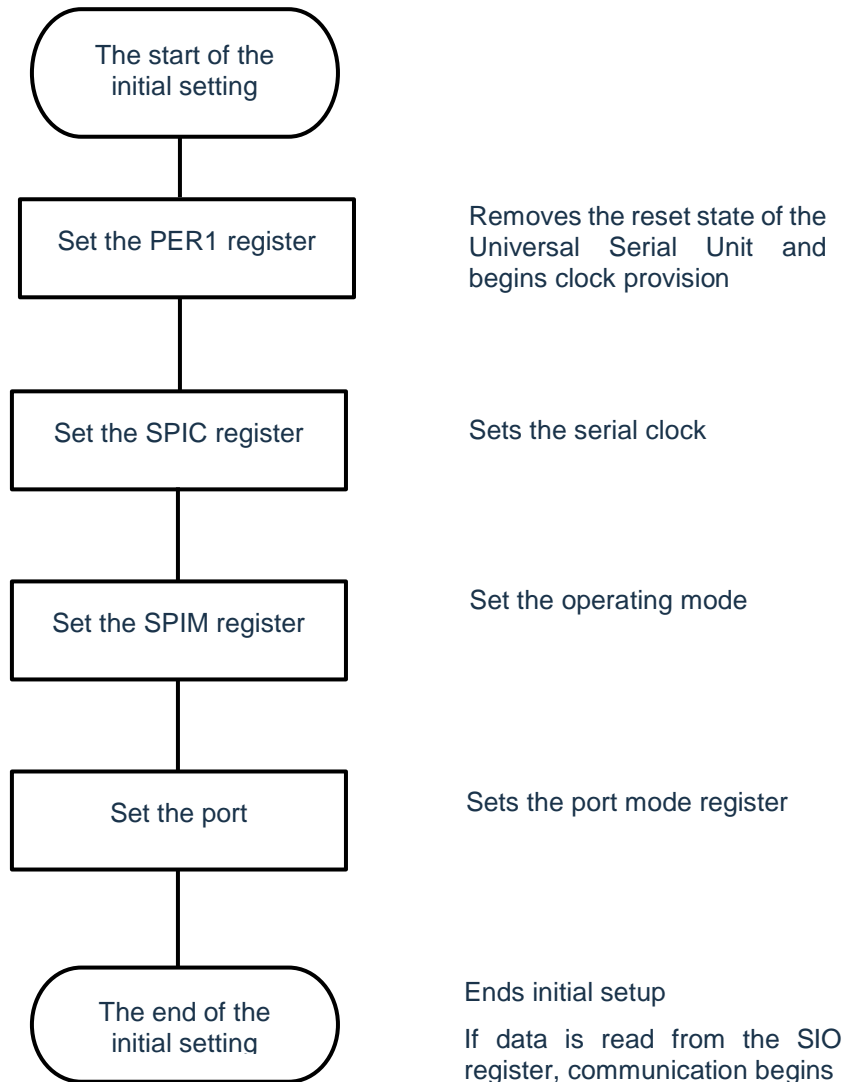
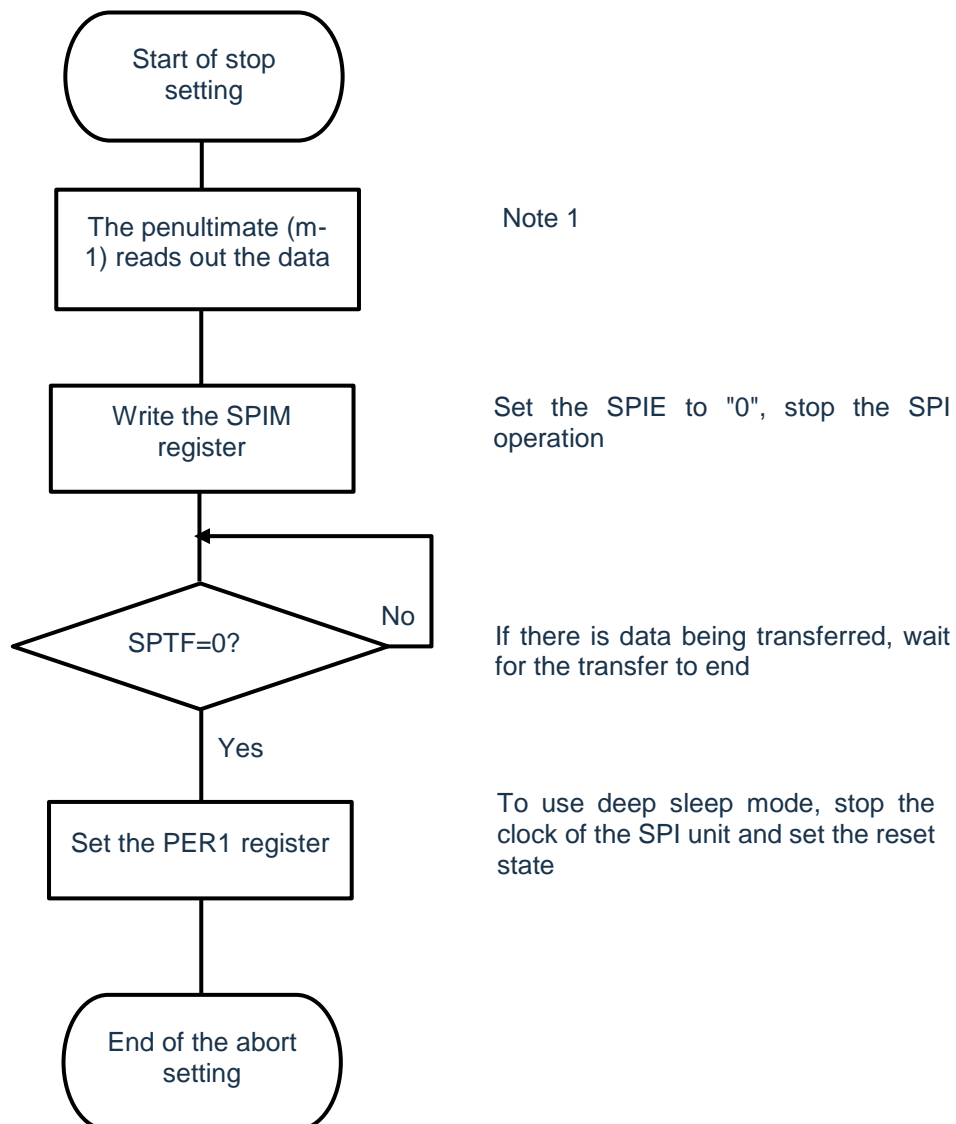


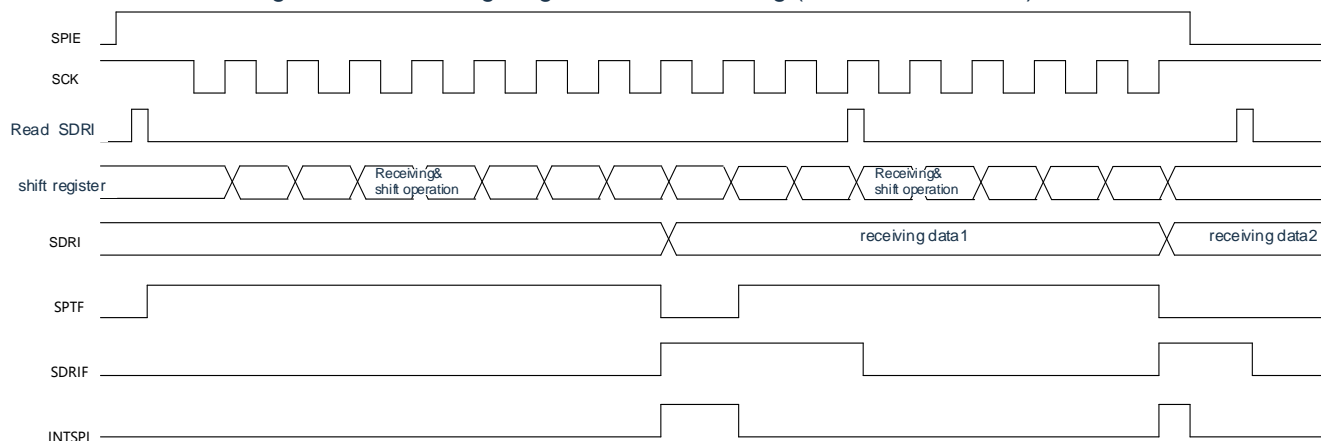
Figure 17-12: Stop step of master receive



Note 1: In receive-only mode, the SPI transmission is triggered by reading the value of the SIO register. If the SPI is not aborted in time, there may be a redundant transmission after the last read of SIO. If you want to avoid the last redundant transmission, you can turn off SPIE after waiting for an SCK cycle after the penultimate data readout. The transfer of the SPI will be aborted after the last data transfer is complete.

(2) Processing

Figure 17-13: Timing diagram of the receiving (CPHA=1, CPOL=1)



### 17.4.3 Slave transmission and reception

If bits CKS2-0 of the serial clock selection register (SPIC) select slave mode, bit 6 (TRMD) of the serial operation mode register (SPIM) is 1, you enter slave send/receive mode. When a value is written to the transmit buffer register (SOTB), wait for the clock of the master device to start sending/receiving.

(1) Procedure

Figure 17-14: Initial setup steps for slave send/receive

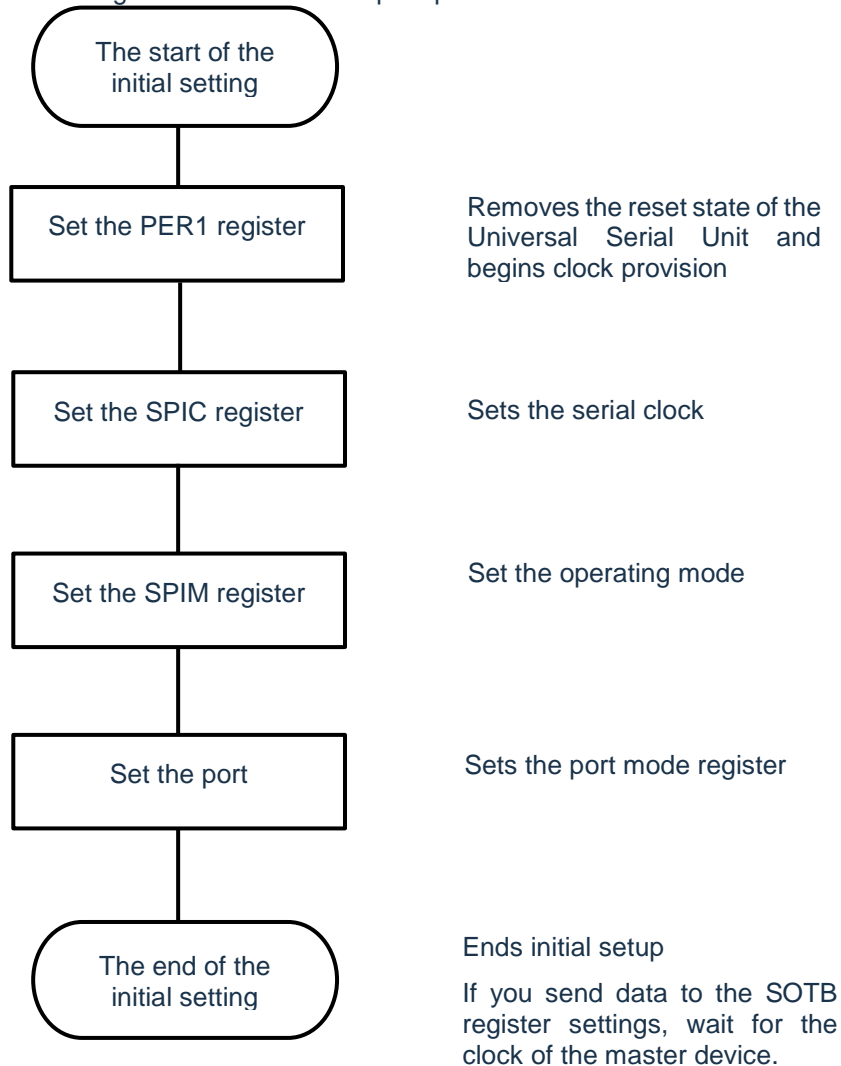
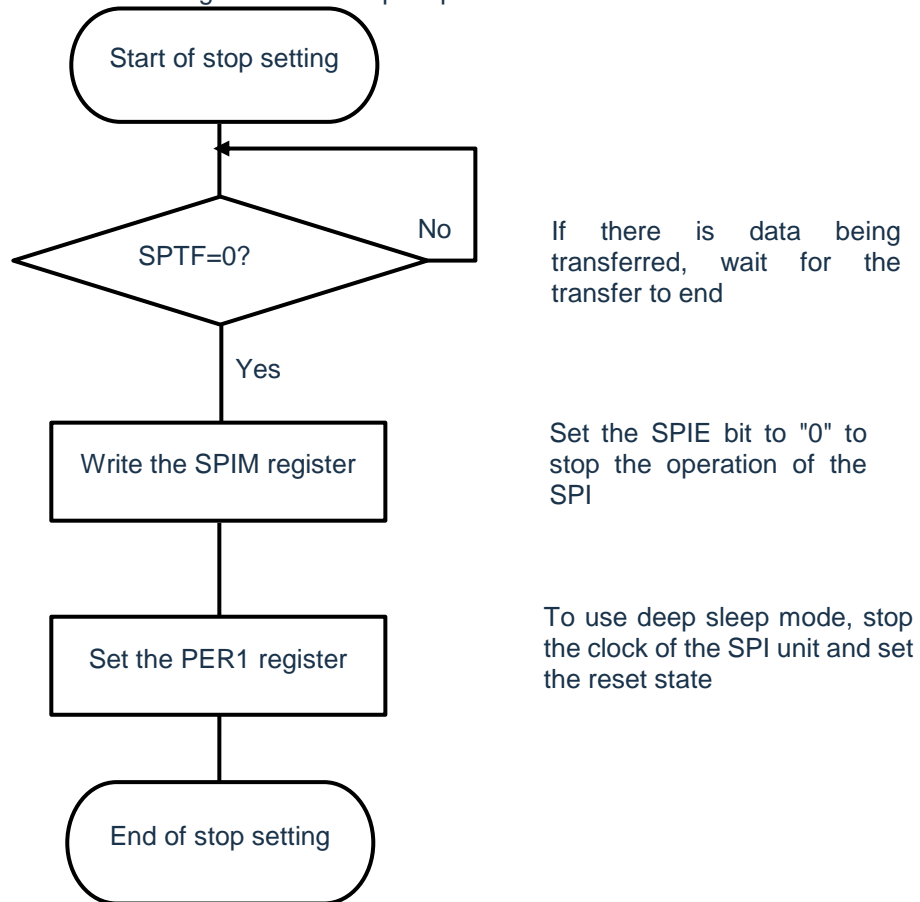


Figure 17-15: Stop step of slave send/receive



## (2) Processing

Figure 17-16: Timing diagram of send/receive (single-send mode) (INTMD=0,CPHA=1, CPOL=1)

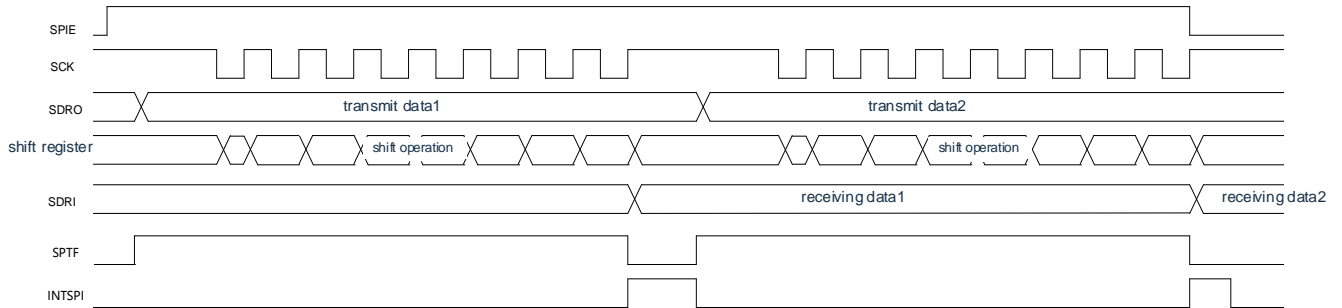
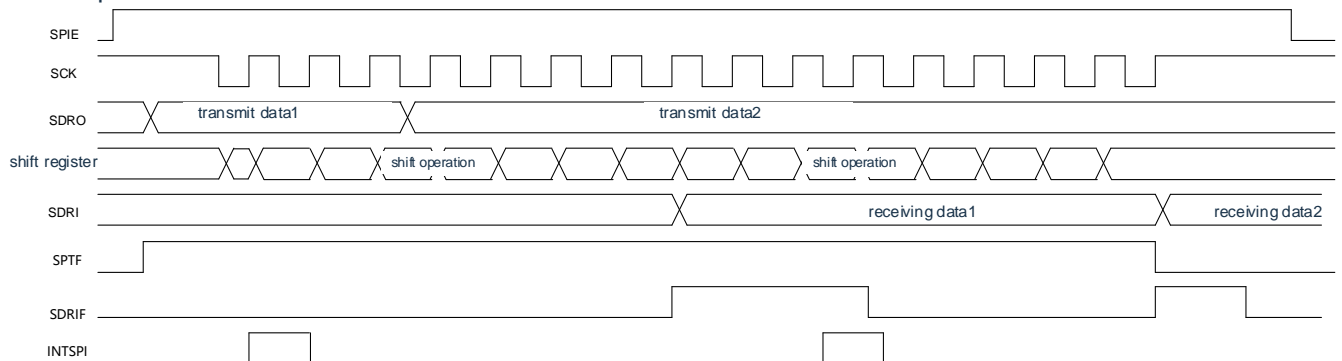


Figure 17-17: Timing diagram of transmit/receive (continuous transmission mode) (INTMD=1, CPHA=1, CPOL=1)  
Slave reception



## 17.4.4 Slave reception

If bits CKS2-0n of the serial clock selection register (SPIC) select slave mode and bit 6 (TRMD) of the serial operation mode register (SPIM) is 0, slave receive mode is entered. When reading data from the receive buffer register (SIO), wait for the clock of the master device to start receiving.

### (1) Procedure

Figure 17-18: Initial setup steps for slave reception

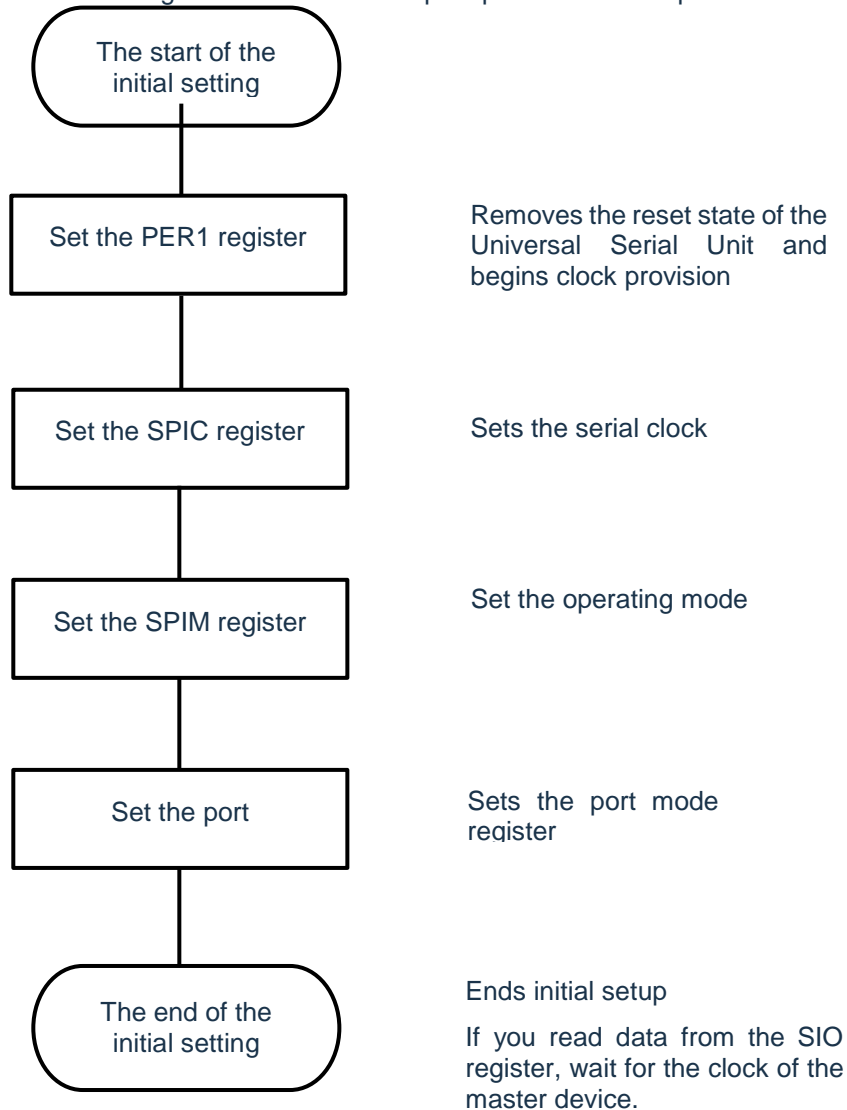
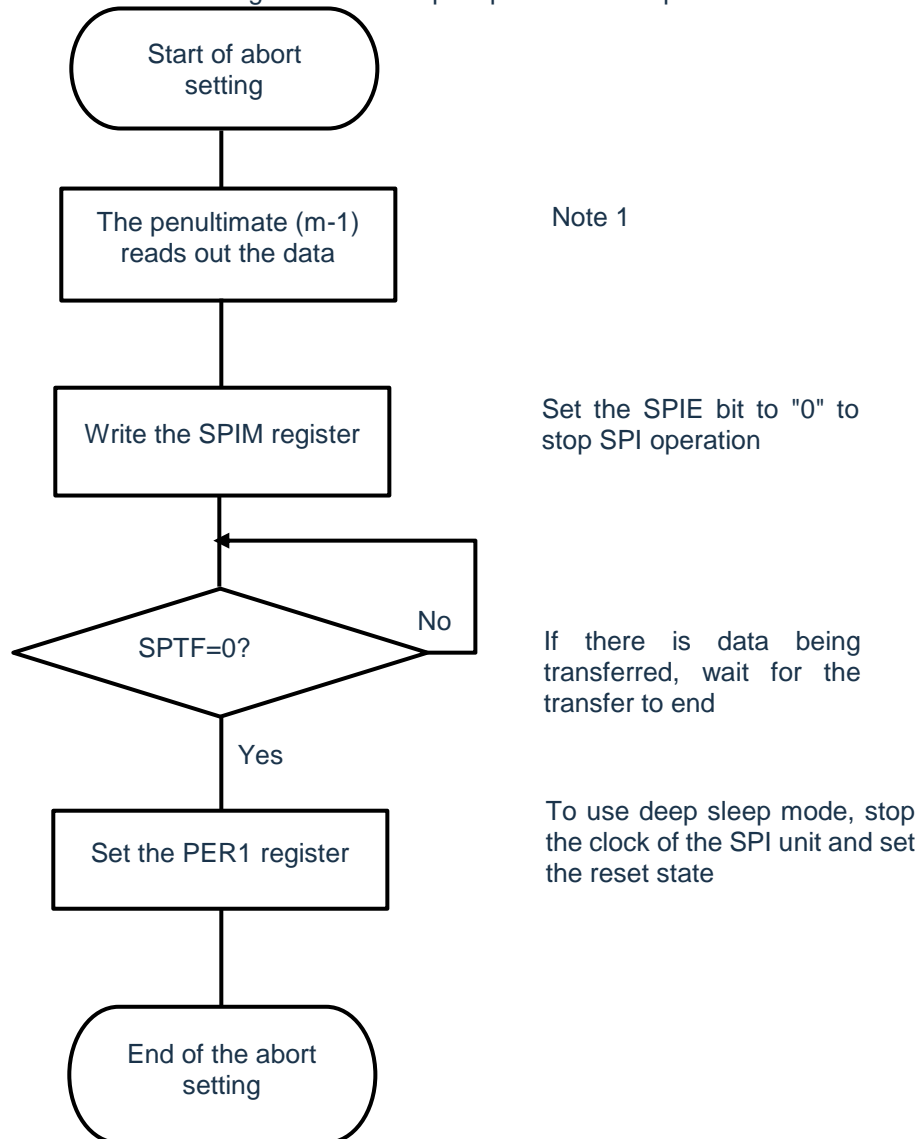


Figure 17-19: Stop step of slave reception

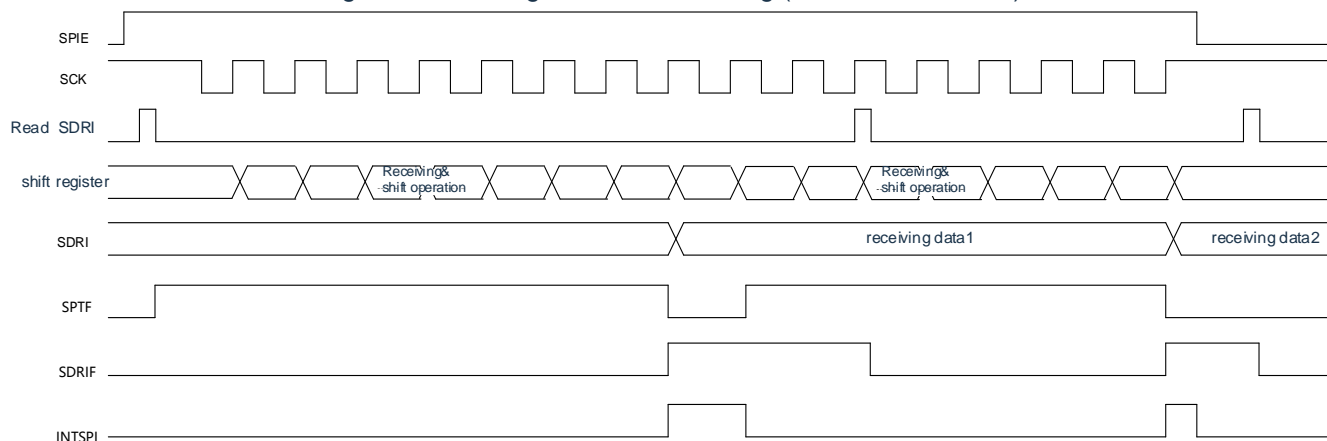


Note 1: In receive-only mode, the SPI transmission is triggered by reading the value of the SIO register. If the SPI is not aborted in time, there may be a redundant transmission after the last read of SIO. If you want to avoid the last redundant transmission, you can turn off SPIE after waiting for an SCK cycle after the penultimate data readout. The transfer of the SPI will be aborted after the last data transfer is complete.



(2) Processing

Figure 17-20: Diagram of the receiving (CPHA=1, CPOL=1)



# Chapter 18 Serial Interface IICA

## 18.1 Function of IICA

This product is equipped with a serial interfaces IICA0, and has the following three modes.

### 18.1.1 Run stop mode

This is a mode used when serial transfer is not in progress and reduces power consumption.

### 18.1.2 I<sup>2</sup>C-bus mode (supports multi-master)

This mode transmits 8-bit data to multiple devices via two lines of serial clock (SCLAn) and serial data bus (SDAAn). Conforming to the I<sup>2</sup>C-bus format, the master device can generate "start conditions" and "addresses" for the slave device on the serial data bus, Indication of Transfer Direction, Data, and Stop Condition. The slave automatically detects the received status and data through the hardware. This feature simplifies the I<sup>2</sup>C-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. In sleep mode, when the extension code of the autonomous control device or the address of the local station is received, the deep sleep mode can be released by generating an interrupt request signal (INTIICAn). This is set via the WUPn bit of the IICA control register n1 (IICCTLn1).

A block diagram of the serial interface IICA is shown in Figure 18-1.

Remark: n=0.

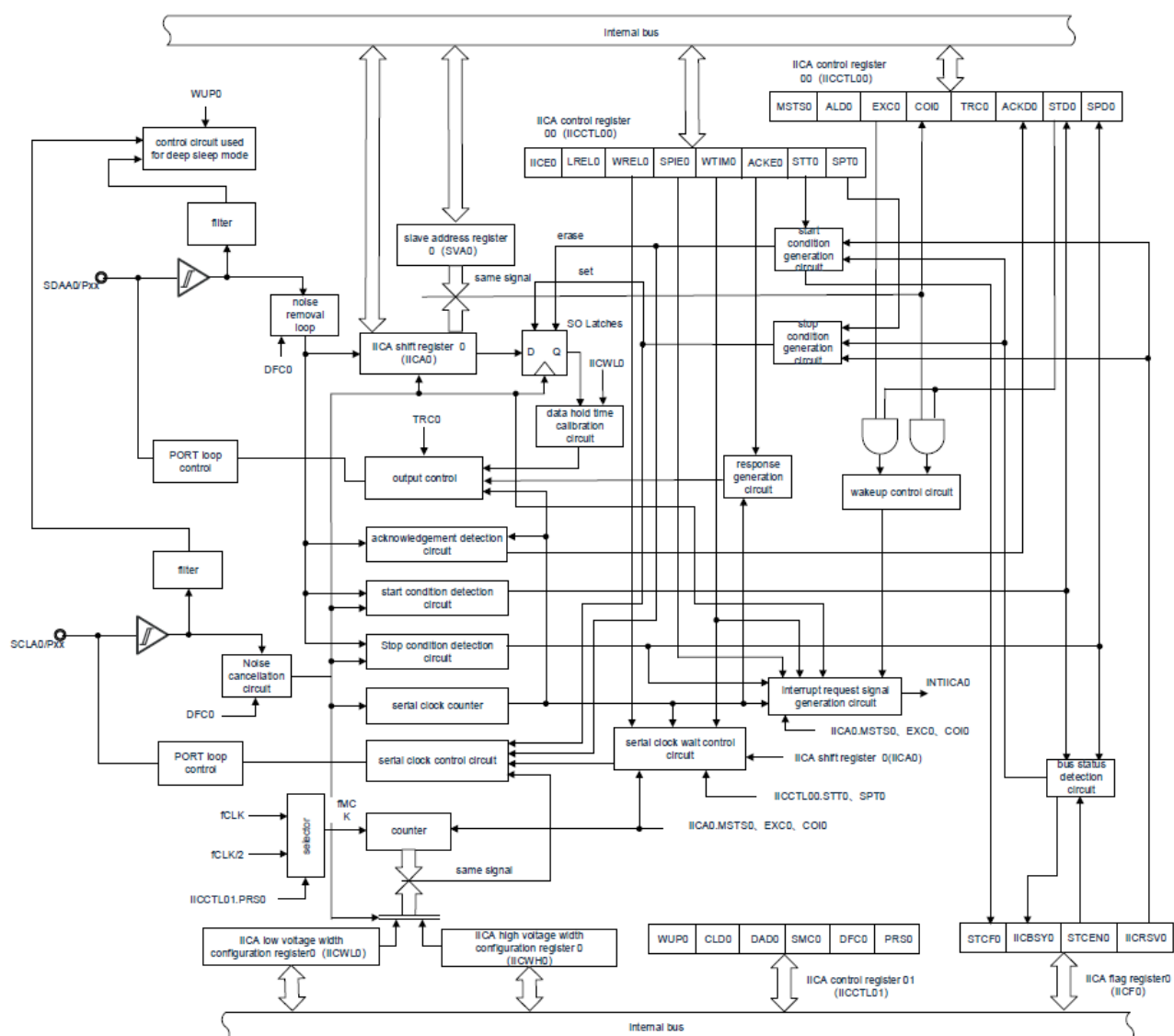
### 18.1.3 Wake-up mode

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 Figure 18-1.

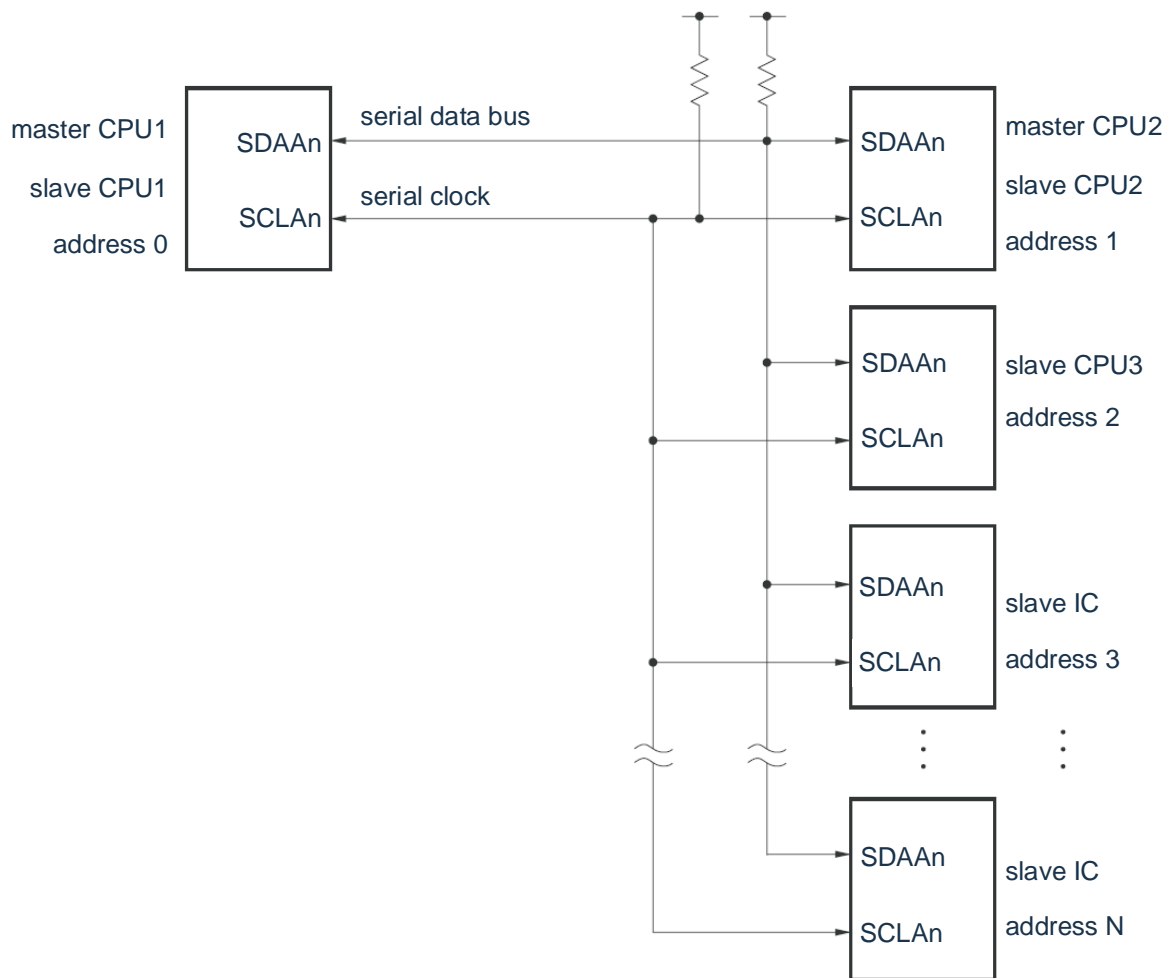
Remark: n=0

Figure 18-1: Diagram of the serial interface IICA



An example of the structure of a serial bus is shown in Figure 18-2.

Figure 18-2: Example of a serial bus structure for I<sup>2</sup>C-bus



Remark: n=0.

## 18.2 Structure of IICA

The serial interface IICA consists of the following hardware.

Table 18-1: Structure of the serial interface IICA

Item	Structure
Register	IICA shift register n (IICAn) slave address register n(SVAn)
Control register	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.

List of registers:

Base address	Offset address	Register name	R/W	Reset value
0x40047000	0x000	IICCTL00	R/W	00H
	0x001	IICCTL01	R/W	00H
	0x002	IICWL0	R/W	FFH
	0x003	IICWH0	R/W	FFH
	0x004	SVA0	R/W	00H
	0x005	IICSE0	R	00H
	0x008	IICA0	R/W	00H
	0x009	IICS0	R	00H
	0x00A	IICF0	R/W	00H

## 18.2.1 IICA shift register n (IICAn)

IICAn registers are registers that convert 8-bit serial data and 8-bit parallel data to and from a serial clock for sending and receiving. 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 the 8-bit memory manipulation instructions. After a reset signal is generated, the value of this register becomes "00H".

Figure 18-3: Format of IICAn shift register n (IICAn)

After reset: 00H, R/W

Symbol	7	6	5	4	3	2	1	0
IICAn								

Notice:

1. During data transfer, data cannot be written to the IICAn registers.
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.

## 18.2.2 Slave address register n(SVAn)

This is the register that holds the 7-bit local station address {A6,A5,A4,A3,A2,A1,A0} when used as a slave.

The SVAn register is set by the 8-bit memory manipulation instruction. However, when the STDn bit is "1" (start condition detected), it is forbidden to overwrite this register.

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

Figure 18-4: Format of slave address register n (SVAn)

After reset: 00H, R/W

Symbol	7	6	5	4	3	2	1	0
SVAn	A6	A5	A4	A3	A2	A1	A0	0 <sup>Note</sup>

Note: bit 0 is fixed to "0".

## 18.2.3 SO latch

The SO latch holds the output level of the SDAAn pin.

## 18.2.4 Wake-up control circuitry

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.

## 18.2.5 Serial clock counter

During the send or receive process, this counter counts the output or input serial clock to check whether 8 bits of data have been sent and received.

## 18.2.6 Interrupt request signal generation circuit

This circuit control generates an interrupt request signal (INTIICAn). An I<sup>2</sup>C 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: WTIMn bit: bit3 of IICA control register n0 (IICCTLn0)

SPIEn bit: bit4 of IICA control register n0 (IICCTLn0)

## 18.2.7 Serial clock control circuitry

In master mode, this circuit generates the output from the sample clock to the clock of the SCLAn pin.

## 18.2.8 Serial clock wait control circuit

This circuit controls the wait timing.

## 18.2.9 Ack generation circuit, stop condition detection circuit, start condition detection circuit, Ack detection circuit

These circuits generate and detect various states.

## 18.2.10 Data hold time correction circuit

This circuit generates a data hold time for the serial clock to drop.

## 18.2.11 Start conditional generation circuitry

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".

## 18.2.12 Stop condition generation circuitry

If the SPTn bit is "1", the circuit generates a stop condition

## 18.2.13 Bus status detection circuitry

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



## 18.3 Registers for controlling 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)

Remark: n=0

### 18.3.1 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 IICA0, PER0 Bit6 (IICA0EN) is 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 18-5: Format of peripheral enable register 0(PER0)

Symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	IICAEN	IRDAEN	SCI2EN	SCI1EN	SCI0EN	TMAEN	TM80EN
Address: 40020424H	After reset: 00000000H				R/W			

IICAEN	Provides control of the input clock of the serial interface IICA
0	Stop providing input clock. <ul style="list-style-type: none"> <li>• The SFR used by the serial interface IICA cannot be written</li> <li>• The serial interface IICA is in reset state.</li> </ul>
1	Enable providing input clock <ul style="list-style-type: none"> <li>• The SFRs used by the serial interface IICA can be read and written.</li> </ul>

Notice: To set the serial interface IICA, the following registers must first be set in the IICAnEN bit "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 (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)

## 18.3.2 IICA control register n0 (IICCTLn0)

This is a register that starts or stops I<sup>2</sup>C operation, sets the wait sequence, and sets other I<sup>2</sup>C operations.

The IICCTLn0 register is set by the 8-bit memory manipulation instruction. However, the SPIEn bits, WTIMn bits, and ACKEn bits must be set when the IICEn bit is "0" or during waiting, and the IICEn bits must be set Bits can be set simultaneously when the bits are set from "0" to "1".

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

Remark: n=0.

Figure 18-6: Format of IICA control register n0(IICCTLn0) (1/4)

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 <sup>2</sup> C operation enable
0	Stop running. Reset <sup>note 1</sup> to IICA status register n (IICSn) and stop internal operation.
1	Enable running.
This bit set to "1" in the state where the SCLAn and SDAAn lines are high.	
Clear condition (IICEn=0).	
Set condition (IICEn=1).	
• Clear by command.	
• When resetting	
• Set by command.	

LRELn <sup>Note2,3</sup>	Exit of communication
0	Normal operation
1	Exits the current communication and enters standby. 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•MSTS•EXCn•COIn•TRCn•ACKDn•STDn
Becomes a standby state to exit the communication until the following communication participation conditions are met.	
•Boot as master device after detecting a stop condition.	
•Addresses match or extended codes are received after the start condition is detected.	
Clear condition (LRELn=0).	
Set condition (LRELn=1).	
•Automatically clears after execution.	
• When resetting	
• Set by command.	

WRELn <sup>Note2,3</sup>	Pending release
0	Do not dismiss the wait.
1	Release the wait. Clears automatically after the wait is released.
If the WRELn bit (untapped) is set during the 9th clock wait in the transmit state (TRCn=1), the SDAAn line becomes High impedance state (TRCn=0).	
Clear condition (WRELn=0).	
Set condition (WRELn=1).	
•Automatically clears after execution.	
• When resetting	
• Set by command.	

Note 1: To IICA shift register n (IICAn), IICA flag register n (IICFn). STCFn bits and IICBSYn bits and CLDn of IICA control register n1 (IICCTLn1). Bits and DADn bits are reset.

Note 2: In the state where the IICEn bit is "0", the signal for this bit is invalid.

Note 3: The read values for LRELn bits and WRELn bits are always "0".

Notice: If the SCLAn line is high, the SDAAn line is low, and the digital filter is ON (DFCn=1 for the IICCTLn1 register), then enable I<sup>2</sup>C 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 allowing I<sup>2</sup>C to run (IICEn=1).

Remark: n=0

Figure 18-6: Format of IICA control register (IICCTLn0) (2/4)

SPIEn <sup>Note 1</sup>	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). • Clear by command. • When resetting	
Set condition (SPIEn=1). • Set by command.	

WTIMn <sup>Note 1</sup>	Control of waiting and interrupt requests
0	An interrupt request signal is generated on the falling edge of the 8th clock. Master device: After outputting 8 clocks, set the clock output low to wait. Slave: After entering 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 device: After outputting 9 clocks, set the clock output low to wait. Slave: After entering 9 clocks, set the clock low and wait for the master.
During address transmission, regardless of the setting of this bit, an interrupt is generated on the falling edge of the 9th clock; After the address transfer ends, the setting for this bit is there Effect. The 9th clock descent edge of the master device during address transfer enters a waiting state. The slave that receives the local station address is generating a reply The 9th clock drop edge after (ACK) enters the wait state, but the slave device that receives the extension enters the wait state on the 8th clock drop edge.	
Clear condition (WTIMn=0). • Clear by command. • When resetting	
Set condition (WTIMn=1). • Set by command. • When resetting	

ACKEn <sup>Note 1,2</sup>	ACK control
0	No Ack.
1	Allow Acks. Set the SDAAn line low during the 9th clock.
Clear condition (ACKEn=0). • Clear by command. • When resetting	
Set condition (ACKEn=1). • Set by command.	

Note 1: In the state where the IICEn bit is "0", the signal for this bit is invalid. This bit must be set during this time.

Note 2: When the address transmission process is not an extension code, the setpoint is invalid. When it is a slave device and the address matches, an Ack is generated independent of the setpoint.

Remark: n=0

Figure 18-6: Format of IICA control register n0 (IICCTLn0) (3/4)

STTn <sup>Note 1,2</sup>	The triggering of the start condition
0	No start conditions are generated.
1	<p>When the bus is released (standby, IICBSYn bit is "0"): If this bit is "1", a start condition (boot as the master device) is generated.</p> <p>When a third party is communicating:</p> <ul style="list-style-type: none"> <li>• Allow communication in the case of the reservation function (IICRSVn=0). Used as a start condition reservation sign. If you set this bit to "1", a start condition is automatically generated just after the bus is released.</li> <li>• The case where 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.</li> </ul>
<p>Notes on set timing:</p> <ul style="list-style-type: none"> <li>• Master Receive: Disables this bit to "1" during transmission. This bit "1" can only be placed during the waiting period when ACKEn is "0" and notifying the slave that receiving it has completed.</li> <li>• Master Send: During the Ack, the start condition may not be generated properly. This bit "1" must be placed during the wait period after the 9th clock is output.</li> <li>• Prohibit and Trigger of Stop Condition (SPTn) with "1" at the same time.</li> <li>• After placing stTn to "1", it is forbidden to use this bit "1" again until the purge condition is met.</li> </ul>	
Clear condition (STTn=0).	Set condition (STTn=1).
<ul style="list-style-type: none"> <li>• When arbitration fails</li> <li>• Master device generation start condition.</li> <li>• Cleared because the LRELn bit is "1" (Exit Communication).</li> <li>• When the IICEn bit is "0" (stop running).</li> <li>• When resetting</li> </ul>	<ul style="list-style-type: none"> <li>• Set by command.</li> </ul>

Note 1: In the state where the IICEn bit is "0", the signal for this bit is invalid.

Note 2: The read value of the STTn bit is always "0".

Remark:

1. If bit1 (STTn) is read after setting the data, this bit becomes "0".
2. IICRSVn: Bit0 of IICA flag register n (IICFn)  
STCFn: Bit7 of IICA flag register n (IICFn)
3. n=0

Figure 18-6: Format of IICA control register n0 (IICCTLn0) (4/4)

SPTn <sup>Note</sup>	The trigger of the stop condition
0	No stop conditions are generated.
1	Generate a stop condition (end of transfer as master).
<p>Notes on set timing:</p> <ul style="list-style-type: none"> <li>• Master Receive: Disables this bit setting 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 that receiving it has completed.</li> <li>• Master Send: During the Ack, the stop condition may not be generated properly. This bit must be set to "1" during the wait period after the 9th clock is output.</li> <li>• Prohibits placing a "1" simultaneously with the trigger of the start condition (STTn).</li> <li>• SPTn can only be placed in the case of the master device "1".</li> <li>• When the WTIMn bit is "0", it must be noted that if the SPTn bit is set to "1" during the wait after 8 clocks of output, the stop condition is generated during the high level of the 9th clock after the release of the wait. 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.</li> <li>• After setting the SPTn to "1", it is forbidden to set this bit "1" again until the purge condition is met.</li> </ul>	
Clear condition (SPTn=0).	Set condition (SPTn=1).
<ul style="list-style-type: none"> <li>• When arbitration fails</li> <li>• Clear automatically when a stop condition is detected.</li> <li>• Cleared because the LRELn bit is "1" (Exit Communication).</li> <li>• When the IICEn bit is "0" (stop running).</li> <li>• When resetting</li> </ul>	<ul style="list-style-type: none"> <li>• Set by command.</li> </ul>

Note: The read value for the SPTn bit is always "0".

Notice: When bit3 (TRCn) of IICA status register n (IICSn) is "1" (transmit state), if at 9 Clocks set bit5 (WRELn) of the IICCTLn0 register to "1" to relieve the wait, clearing the TRCn The bit (receive state) sets the SDAAn line to high impedance. Waiting for the TRCn bit of "1" (send state) must be released by writing the IICA shift register n.

Remark: n=0

### 18.3.3 IICA status register n(IICSn)

This is the register that represents the I<sup>2</sup>C state.

The IICSn register can only be read by the 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 run) 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 18-7: Format of IICA status register n (IICSn) (1/3)

After reset: 00H				R				
Symbol	7	6	5	4	3	2	1	0
IICS <sub>n</sub>	MSTS <sub>n</sub>	ALD <sub>n</sub>	EXC <sub>n</sub>	COI <sub>n</sub>	TRC <sub>n</sub>	ACKD <sub>n</sub>	STD <sub>n</sub>	SPD <sub>n</sub>

MSTS <sub>n</sub>	Confirmation flag for the master status
0	Slave state or communication standby
1	Master communication status
Clear condition (MSTS <sub>n</sub> =0).	
Set condition (MSTS <sub>n</sub> =1).	
<ul style="list-style-type: none"> <li>• When a stop condition is detected</li> <li>• When the ALD<sub>n</sub> bit is "1" (arbitration failed).</li> <li>• Cleared because the LREL<sub>n</sub> bit is "1" (Exit Communication).</li> <li>• When the IICEn bit changes from "1" to "0" (stops running).</li> </ul>	
• When the build starts condition	

ALD <sub>n</sub>	Detection of arbitration failures
0	Indicates that no arbitration occurred or was won.
1	Indicates that the arbitration failed. Clear the MSTS <sub>n</sub> bit.
Clear condition (ALD <sub>n</sub> =0).	
Set condition (ALD <sub>n</sub> =1).	
<ul style="list-style-type: none"> <li>• Automatically clear after reading the IICSn register</li> <li>• When the IICEn bit changes from "1" to "0" (stops running).</li> <li>• When resetting</li> </ul>	
• 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 aldn bits, the data for the ALDn bits must be read before reading other bits.

Remark:

1. LREL<sub>n</sub>: Bit6 of IICA control register n0 (IICCTLn0)  
IICEn: Bit7 of IICA control register n0 (IICCTLn0)
2. n=0

Figure 18-7: Format of IICA status register n (IICSn) (2/3)

EXCn	Receive detection of extension codes	
0	The extension code was not received.	
1	Extended code received.	
Clear condition (EXCn=0).		Set condition (EXCn=1).
<ul style="list-style-type: none"> <li>• When a start condition is detected</li> <li>• When a stop condition is detected</li> <li>• Cleared because the LRELn bit is "1" (Exit Communication).</li> <li>• When the IICEn bit changes from "1" to "0" (stops running).</li> <li>• When resetting</li> </ul>		<ul style="list-style-type: none"> <li>• When the high 4 bits of the received address data are "0000" or "1111" (asserted on the rising edge of the 8th clock).</li> </ul>

COIn	Detection of address matches	
0	The address is different.	
1	The address is the same.	
Clear condition (COIn=0).		Set condition (COIn=1).
<ul style="list-style-type: none"> <li>• When a start condition is detected</li> <li>• When a stop condition is detected</li> <li>• Cleared because the LRELn bit is "1" (Exit Communication).</li> <li>• When the IICEn bit changes from "1" to "0" (stops running).</li> </ul>		<ul style="list-style-type: none"> <li>• When receiving address and local station address (slave address register n (SVAn)) are the same (asserted on the rising edge of the 8th clock).</li> </ul>

TRCn	Status detection of sends/receives	
0	In the receiving state (except for the sending state). Place the SDAAn line as high impedance.	
1	in the sending 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).		Position condition (TRCn=1).
<p>&lt; master and slave &gt;</p> <ul style="list-style-type: none"> <li>• When a stop condition is detected</li> <li>• Cleared because the LRELn bit is "1" (Exit Communication).</li> <li>• When the IICEn bit changes from "1" to "0" (stops running).</li> <li>• Clear note due to WRELn bit "1" (unsheath wait).</li> <li>• When the ALDn bit changes from "0" to "1" (arbitration failed).</li> <li>• When resetting</li> <li>• Cases of non-participation in communication (MSTS<sub>n</sub>, EXC<sub>n</sub>, COIn=0).</li> </ul> <p>&lt; the master device &gt;</p> <ul style="list-style-type: none"> <li>• When the LSB (Transmit Direction Indicator bit) of byte 1 outputs "1"</li> </ul> <p>&lt; slave &gt;</p> <ul style="list-style-type: none"> <li>• When a start condition is detected</li> <li>• When the LSB (Transmit Direction Indicator bit) of byte 1 enters "0"</li> </ul>		<p>&lt; master device &gt;</p> <ul style="list-style-type: none"> <li>• When the build starts condition</li> <li>• LSB (transmit direction indication bit) when byte 1 (address is transmitted). When output "0" (master sends).</li> </ul> <p>&lt; slave &gt;</p> <ul style="list-style-type: none"> <li>• LSB (Transmit) when the master device is byte 1 (Address Transfer Direction indicator bit) when entering "1" (Slave send).</li> </ul>

Note: When the bit3 (TRCn) of IICA status register n (IICSn) is "1" (transmit state), if it is in line 9 clocks place bit5 (WRELn) of the IICA control register n0 (IICCTLn0). "1" to relieve the wait, just clear the TRCn bit (receive state) and set the SDAAn line to high impedance. Waiting for the TRCn bit of "1" (send state) must be released by writing the IICA shift register n.

Remark:

1. LRELn: Bit6 of IICA control register n0 (IICCTLn0)  
IICEn: Bit7 of IICA control register n0 (IICCTLn0)
2. n=0

Figure 18-7: Format of IICA status register n (IICSn) (3/3)

ACKDn	Detection of the Ack (ACK).	
0	No Ack was detected.	
1	An Ack was detected.	
Clear condition (ACKDn=0).		Set condition (ACKDn=1).
<ul style="list-style-type: none"> <li>• When a stop condition is detected</li> <li>• When the next byte of the 1st clock rises</li> <li>• Cleared because the LRELn bit is "1" (Exit Communication).</li> <li>• When the IICEn bit changes from "1" to "0" (stops running).</li> <li>• When resetting</li> </ul>		<ul style="list-style-type: none"> <li>• Place the SDAAn line low on the 9th clock rising edge of the SCLAn line</li> </ul>

STDn	Start the detection of conditions	
0	No start condition detected.	
1	A start condition was detected, indicating that it was in the process of address transfer.	
Clear condition (STDn=0).		Position condition (STDn=1).
<ul style="list-style-type: none"> <li>• When a stop condition is detected</li> <li>• When the 1st clock rises after the next byte of the address is transmitted</li> <li>• Cleared because the LRELn bit is "1" (Exit Communication).</li> <li>• When the IICEn bit changes from "1" to "0" (stops running).</li> <li>• When resetting</li> </ul>		<ul style="list-style-type: none"> <li>• When a start condition is detected</li> </ul>

SPDn	Detection of stop conditions	
0	No stop condition detected.	
1	A stop condition is detected, the master device ends communication and the bus is released.	
Clear condition (SPDn=0).		Set condition (SPDn=1).
<ul style="list-style-type: none"> <li>• After setting this bit, the address transmits the byte after the start condition is detected when the clock rises</li> <li>• When the WUPn bit changes from "1" to "0"</li> <li>• When the IICEn bit changes from "1" to "0" (stops running).</li> <li>• When resetting</li> </ul>		<ul style="list-style-type: none"> <li>• When a stop condition is detected</li> </ul>

Remark:

1. LRELn: Bit6 of IICA control register n0 (IICCTLn0)  
IICEn: Bit7 of IICA control register n0 (IICCTLn0)
2. n=0



### 18.3.4 IICA flag register n(IICFn)

This is the register that sets the I<sup>2</sup>C operating mode and indicates the status of the I<sup>2</sup>C-bus.

The IICFn register is set by the 8-bit memory manipulation instruction. However, only the STTn clear flag (STCFn) and the I<sup>2</sup>C-bus status flag (IICBSYn) can be read.

The communication appointment function is allowed or disabled by the IICRSVn bit setting, and the initial value of the IICBSYn bit is set by the STCENn bit. Only bit7 ( IICEn = 0) can only write IICRSVn bits and STCENn bits. After allowing operation, only the IICFn registers can be read. After generating a reset signal, the value of this register changes to "00H".

Figure 18-8: Format of IICA flag register n(IICFn)

After reset: 00H							
R/W Note							
Symbol	7	6	5	4	3	2	1 0
IICFn	STCFn	IICBSYn	0	0	0	0	STCENn IICRSVn

STCFn	STTn clear flag
0	Release Start Conditions.
1	The STTn flag cannot be cleared while the start condition cannot be issued.
Clear condition (STCFn=0).	
Set condition (STCFn=1).	
<ul style="list-style-type: none"> <li>• Clearance due to STTn bit being "1"</li> <li>• When the IICEn bit is "0" (stop running).</li> <li>• When resetting</li> </ul>	<ul style="list-style-type: none"> <li>• When the STTn bit is cleared to "0" in a state where communication reservation is set to prohibit (IICRSVn=1) and the start condition cannot be issued</li> </ul>

IICBSYn	I <sup>2</sup> C-bus status flag
0	Bus release state (initial communication state at STCENn=1).
1	Bus communication state (initial communication state at STCENn=0).
Clear condition (IICBSYn=0).	
Set condition (IICBSYn=1).	
<ul style="list-style-type: none"> <li>• When a stop condition is detected</li> <li>• When the IICEn bit is "0" (stop running).</li> <li>• When resetting</li> </ul>	<ul style="list-style-type: none"> <li>• When a start condition is detected</li> <li>• Set the IICEn bit when the STCENn bit is "0"</li> </ul>

STCENn	The initial start enable triggering
0	After allowing run (IICEn=1), a start condition is allowed to be generated by detecting a stop condition.
1	After allowing a run (IICEn=1), the start condition is allowed to be generated without detecting the stop condition.
Clear condition (STCENn=0).	
Set condition (STCENn=1).	
<ul style="list-style-type: none"> <li>• Clear by command.</li> <li>• When a start condition is detected</li> <li>• When resetting</li> </ul>	<ul style="list-style-type: none"> <li>• Set by command.</li> </ul>

IICRSVn	The communication appointment function prohibits bits
0	Allow correspondence appointments.
1	Communication appointments are prohibited.
Clear condition (IICRSVn=0).	
Set condition (IICRSVn=1).	
<ul style="list-style-type: none"> <li>• Clear by command.</li> <li>• When resetting</li> </ul>	<ul style="list-style-type: none"> <li>• Set by command.</li> </ul>

Note: Bit6 and bit7 are read-only bits.

Notice:

1. The STCENn bit can only be written when it is stopped (IICEn=0).
2. If the STCENn bit is "1", the bus is considered to be free (IICBSYn=0) regardless of the actual bus state, so as to avoid the first starting condition ( STTn=1) when breaking other communications requires confirmation that there is no third party being communicated.
3. Write IICRSVn only when it is out of operation (IICEn=0).

Remark:

1. STTn: Bit1 of IICA control register n0 (IICCTLn0)
2. IICEn: Bit7 of IICA control register n0 (IICCTLn0)

### 18.3.5 IICA control register n1(IICCTLn1)

This is a register used to set the I<sup>2</sup>C 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.

In addition to the WUPn bit, bit7 must be disabled to run in I<sup>2</sup>C (IICA control register n0 (IICCTLn0). (IICEn)=0) when setting the IICCTLn1 register.

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

Figure 18-9: Format of IICA control register n1 (IICCTLn1) (1/2)

After reset: 00H

R/W<sup>Note 1</sup>

Symbol	7	6	5	4	3	2	1	0
IICCTLn1	WUPn	0	CLDn	DADn	SMCn	DFCn	0	PRSn

WUPn	Address matching control of wake-up
0	In deep sleep mode, stop the operation of the address-matching wake feature.
1	In deep sleep mode, the address matching wake function is allowed to run.

To transfer to deep sleep mode by setting the WUPn bit to "1", at least three F<sub>MCK</sub> 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").

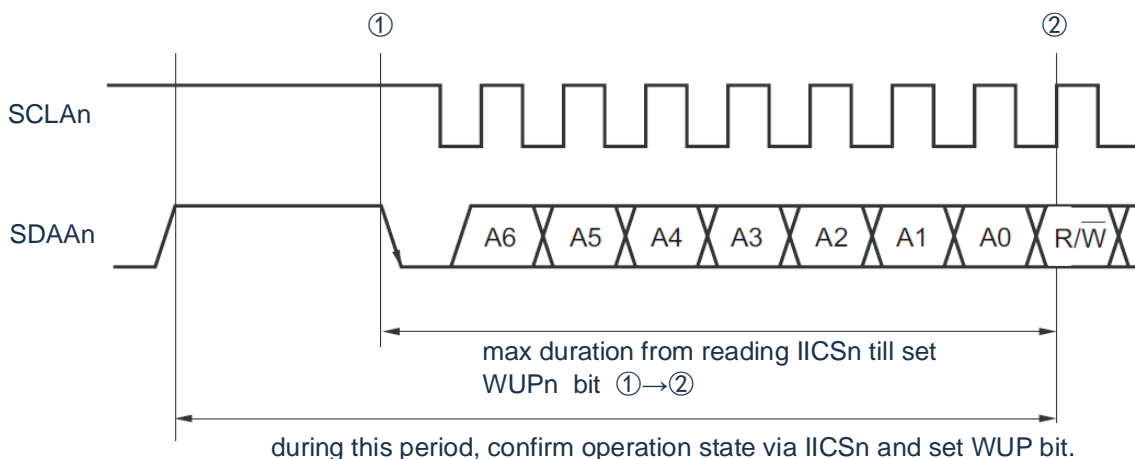
In the state of WUPn bit "1", the interrupt timing when the address is matched or the extension code is received is the same as the interrupt timing when WUPn bit is "0".

(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).
<ul style="list-style-type: none"> <li>Cleared by directive (after the address matches or the extension code is received).</li> </ul>	<ul style="list-style-type: none"> <li>Set by instruction (MSTSn=0, EXCn=0, COIn=0 and STDn=0 (do not participate in communications))<sup>Note 2</sup>.</li> </ul>

Note 1: Bit4 and bit5 are read-only bits.

Note 2: During the period shown below, it is necessary to confirm the status of the IICA status register n (IICSn) and set it.



Remark: n=0

Figure 18-9: 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	The SCLAn pin was detected low.	
1	The SCLAn pin was detected high.	
Clear condition (CLDn=0).		Set condition (CLDn=1).
<ul style="list-style-type: none"> <li>• When the SCLAn pin is low</li> <li>• When the IICEn bit is "0" (stop running).</li> <li>• When resetting</li> </ul>		<ul style="list-style-type: none"> <li>• When the SCLAn pin is high</li> </ul>

DADn	Level detection of the SDAAn pin (valid only when the IICEn bit is "1").	
0	The SDAAn pin was detected as low.	
1	The SDAAn pin was detected high.	
Clear condition (DADn=0).		Set condition (DADn=1).
<ul style="list-style-type: none"> <li>• When the SDAAn pin is low</li> <li>• When the IICEn bit is "0" (stop running).</li> <li>• When resetting</li> </ul>		<ul style="list-style-type: none"> <li>• When the SDAAn pin is high</li> </ul>

SMCn	Switching of operating modes	
0	Operates in standard mode (maximum transfer rate: 100kbps).	
1	Operates in Fast Mode (Max Transfer Rate: 400kbps) or Enhanced Fast Mode (Max Transfer Rate: 1Mbps).	

DFCn	Operational 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 the clear "0", the transmission clock is unchanged.	

PRSn	Control of the running clock (F <sub>MCK</sub> ).	
0	Select F <sub>CLK</sub> (1MHz≤F <sub>CLK</sub> ≤20MHz)	
1	Select F <sub>CLK</sub> /2 (20MHz<F <sub>CLK</sub> )	

**Notice:**

1. The maximum operating frequency of the IICA Operating Clock (F<sub>MCK</sub>) is 20MHz (Max.). The IICA control register n1 (IICCTLn1) must only be placed when the F<sub>CLK</sub> exceeds 20MHz bit0 (PRSn) is set to "1".
2. In the case of setting the transmission clock, the minimum operating frequency of the F<sub>CLK</sub> must be paid attention to. The minimum operating frequency of the F<sub>CLK</sub> of the serial interface IICA depends on the operating mode.

Fast mode: F<sub>CLK</sub>=3.5MHz(Min.)

Enhanced Fast Mode: F<sub>CLK</sub>=10MHz(Min.)

Standard mode: F<sub>CLK</sub>=1MHz(Min.)

**Remark:**

1. IICEn: Bit7 of IICA control register n0
2. n=0

### 18.3.6 IICA low-level width setting register n(IICWLn)

This register controls the SCLAn pin signal low level width ( $T_{LOW}$ ) and SDAAn pin signal from the serial interface IICA output.

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

The IICWLn register must be set when I<sup>2</sup>C 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 "18.4.2 Setting the transmit clock via IICWLn register and IICWHn register".

The data retention time is 1/4 of the time set by IICWLn.

Figure 18-10: Format of IICA low-level width setting register n (IICWLn)

After reset: FFH				R/W				
Symbol	7	6	5	4	3	2	1	0
IICWLn								

### 18.3.7 IICA high level width setting register n(IICWHn)

This register controls the high level width of the SCLAn pin signal and the SDAAn pin signal of the serial interface IICA output. The IICWHn register is set by an 8-bit memory manipulation instruction.

The IICWHn register must be set when I<sup>2</sup>C 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 18-11: Format of high level width setting register n(IICWHn)

After reset: FFH				R/W				
Symbol	7	6	5	4	3	2	1	0
IICWHn								

Remark:

1. For the method of setting the clock transmitted by the main controller, please refer to 18.4.2(1);  
For how to set the slave IICWLn register and the IICWHn register, refer to 18.4.2(2).
2. n=0

### 18.3.8 Registers for controlling the IICA pin port function

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 (VDD/EVDD withstand) mode of the ports is guaranteed by design to turn on automatically, i.e. the POMxx register does not need to be set by the user.

For detailed setting method, see "Chapter 2 Port Function".

## 18.4 Function of I<sup>2</sup>C-bus mode

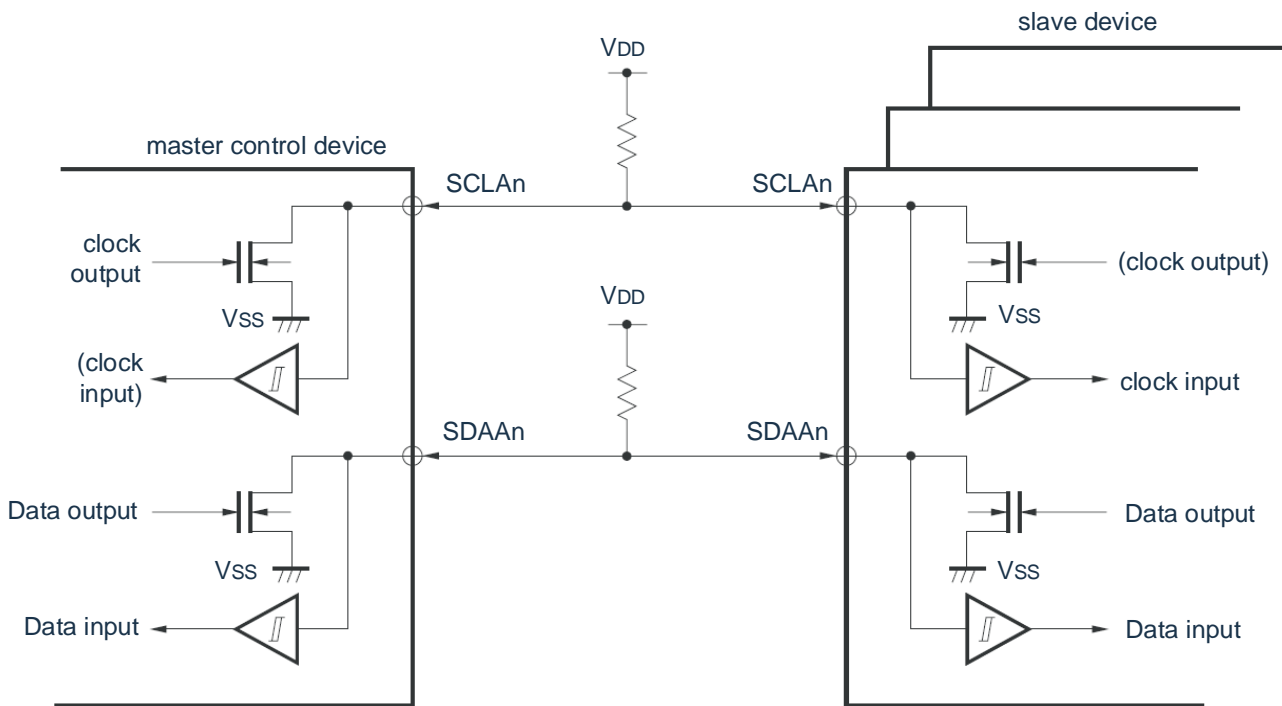
### 18.4.1 Pin structure

The serial clock pin (SCLAn) and serial data bus pin (SDAAn) are structured 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 18-12: Pin Structure Diagram



Remark: n=0

## 18.4.2 Setting the transmit clock via IICWLn register and IICWHn register

- (1) The method by which the master transmits the clock

$$\text{Transmit clock} = \frac{F_{MCK}}{IICWL + IICWH + F_{MCK} (T_R + T_F)}$$

At this point, the optimal setpoints for the IICWLn register and the IICWHn register are as follows:  
(All setpoints are rounded to decimals)

- Quick mode

$$IICWLn = \frac{0.52}{\text{transmit clock}} \times F_{MCK}$$

$$IICWHn = \left( \frac{0.48}{\text{transmit clock}} \times T_R - T_F \right) \times F_{MCK}$$

- Standard mode

$$IICWLn = \frac{0.47}{\text{transmit clock}} \times F_{MCK}$$

$$IICWHn = \left( \frac{0.53}{\text{transmit clock}} \times T_R - T_F \right) \times F_{MCK}$$

- Enhanced quick mode

$$IICWLn = \frac{0.50}{\text{transmit clock}} \times F_{MCK}$$

$$IICWHn = \left( \frac{0.50}{\text{transmit clock}} \times T_R - T_F \right) \times F_{MCK}$$

- (2) How to set the slave IICWLn register and the IICWHn register

(All setpoints are rounded to decimals)

- Quick 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 quick mode

$$IICWLn = 0.50 \mu s \times F_{MCK}$$

$$IICWHn = (0.50 \mu s - T_R - T_F) \times F_{MCK}$$

Notice:

- The maximum operating frequency of the IICA operating clock ( $F_{MCK}$ ) is 20MHz (Max.). The bit0 (PRSn) of the IICA control register n1 (IICCTLn1) must be set to "1" only when the  $F_{CLK}$  exceeds 20MHz.
- In the case of setting the transmission clock, attention must be paid to the minimum operating frequency of  $F_{CLK}$ . The minimum operating frequency of the  $F_{CLK}$  for the serial interface IICA depends on the mode of operation.  
Quick mode:  $F_{CLK} = 3.5\text{MHz (Min.)}$



Enhanced quick mode:  $F_{CLK}=10\text{MHz}(\text{Min.})$

Standard mode:  $F_{CLK}=1\text{MHz}(\text{Min.})$

Remark:

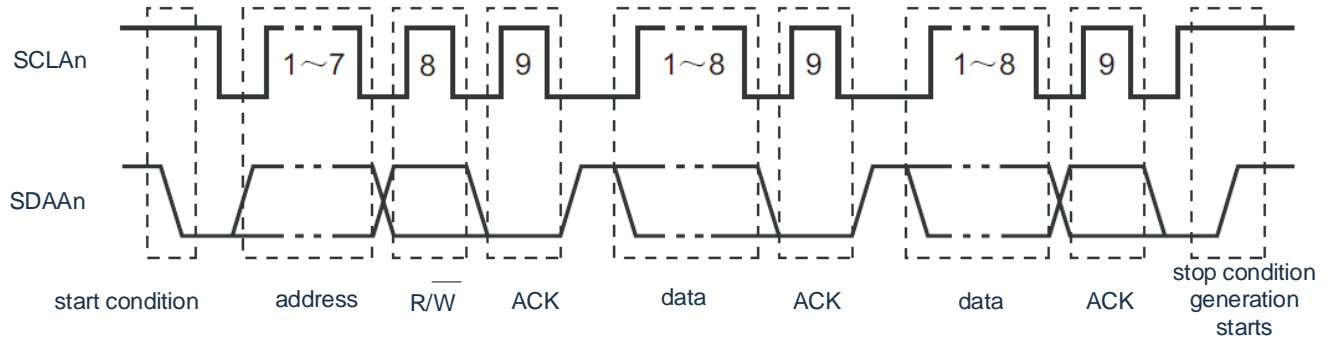
1. Because the rise time ( $T_R$ ) and fall time ( $T_F$ ) of the SDAAn signal and the SCLAn signal differ depending on the pull-up resistance and the routing capacitance, they must be calculated separately.
2. IICWLn: IICA low level width setting register n  
IICWHn: IICA high level width setting register n  
 $T_F$ : The descent time of the SDAAn signal and the SCLAn signal  
 $T_R$ : The rise time of the SDAAn signal and the SCLAn signal  
 $F_{MCK}$ : IICA operating clock frequency
3.  $n=0$

## 18.5 Definition and control method of I<sup>2</sup>C-bus

The following describes the serial data communication format and the signals used for the I<sup>2</sup>C-bus.

The Start Condition, Address, Data generated on the serial data bus of the I<sup>2</sup>C-bus. The respective transmission timings for and "Stop Condition" are shown in the following figure.

Figure 18-13: Serial data transfer timing of the I<sup>2</sup>C bus



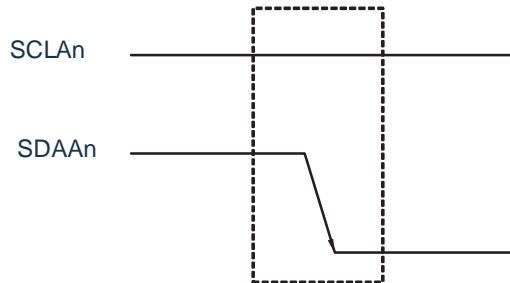
The master generates start conditions, slave addresses, and stop conditions.

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.

## 18.5.1 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 18-14: Starting condition



In the state where a stop condition (SPDn: bit0=1 of IICA status register n (IICSn)) is detected, if the IICA is detected Bit1 (STTn) of the control register n0 (IICCTLn0) is set to "1" to start the output condition. If a start condition is detected, set bit1 (STDn) of the IICSn register to "1".

Remark: n=0

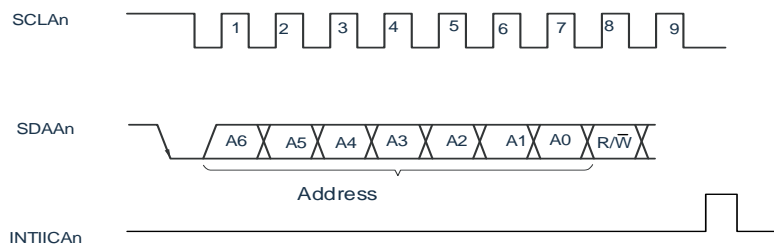
## 18.5.2 Address

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 18-15: Address



Note: If data other than the local station address or extension code is received while the slave is running, INTIICAn is not generated.

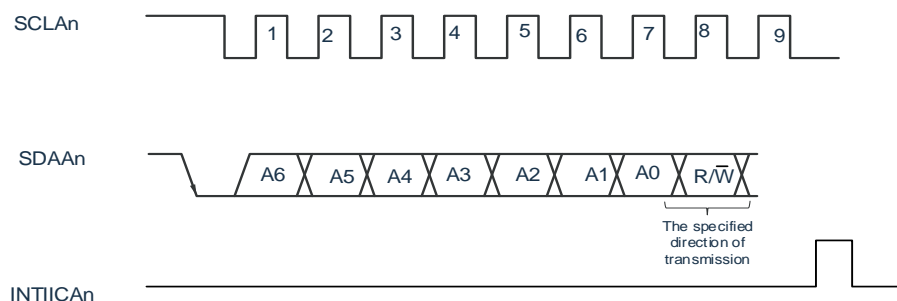
If the 8-bit data consisting of the slave address and the transfer direction described in “18.5.3 Designation of transmission direction” is written to the IICA shift register n (IICAn), the address is output. The received address is written to the IICAn register. The slave address is assigned to the higher 7 bits of the IICAn register.

## 18.5.3 Designation of transmission direction

The master sends one bit of data in the specified direction after the 7-bit address.

When this transfer direction bit is "0", the master device sends data to the slave device; when this transfer direction bit is "1", the master device receives data from the slave device.

Figure 18-16: Designation of the transmission direction



Note: If data other than the local station address or extension code is received during the slave run, no INTIICAn is generated.

Remark: n=0

## 18.5.4 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 There is no normal reception.
- ② The receipt of the last data has ended.
- ③ The address specified receiver does not exist.

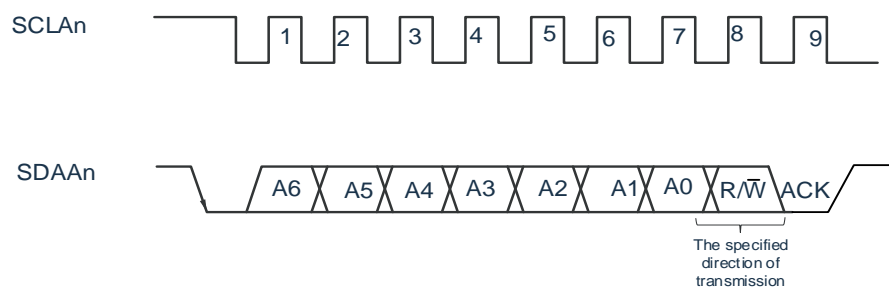
The receiver sets the SDAA<sub>n</sub> line low on the 9th clock to generate a reply (normal receive).

By setting the bit2 (ACKEn) of the IICA control register n0 (IICCTLn0) to "1", it becomes a state that automatically generates a response. Sets bit3 (TRCn) of the IICSn register by the 8th bit of data that follows from the 7-bit address information. In the case of receiving (TRCn=0), it is usually necessary to set the ACKEn bit to "1".

During the slave receive run (TRCn=0) cannot receive data or does not need the next data, the ACKEn must be cleared to "0" to inform the master that the data cannot be received.

When the next data is not needed during the master receive run (TRCn=0), in order not to generate a reply, the ACKEn bit must be cleared to "0" to notify the subordinate sender of the end of the data (stop sending).

Figure 18-17: ACK



When the address of the local station is received, regardless of the value of the ACKEn bit, a reply is automatically generated; When an address for a non-local station is received, no reply (NACK) is generated.

When an extension code is received, an answer is generated by setting the ACKEn bit to "1" in advance. The answer generation method for receiving data varies depending on the setting of the wait timing, as shown below.

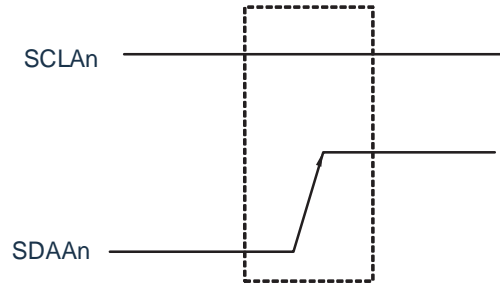
- When 8 clocks of wait are selected (bit 3 (WTIMn) = 0 in the IICCTLn0 register): an answer is generated synchronously with the 8th clock falling edge of the SCLAn pin by setting the ACKEn bit to "1" before releasing the wait.
- When 9 clocks of wait are selected (bit3 (WTIMn) = 1 in the IICCTLn0 register): an answer is generated by setting the ACKEn bit to "1" in advance.

Remark: n=0

## 18.5.5 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 18-18: 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

## 18.5.6 Await

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 18-19: Await (1/2)

(1) The master device waits for 9 clocks and the slave device waits for 8 clocks

(Master: Transmit, Slave: Receive, ACKEn=1)

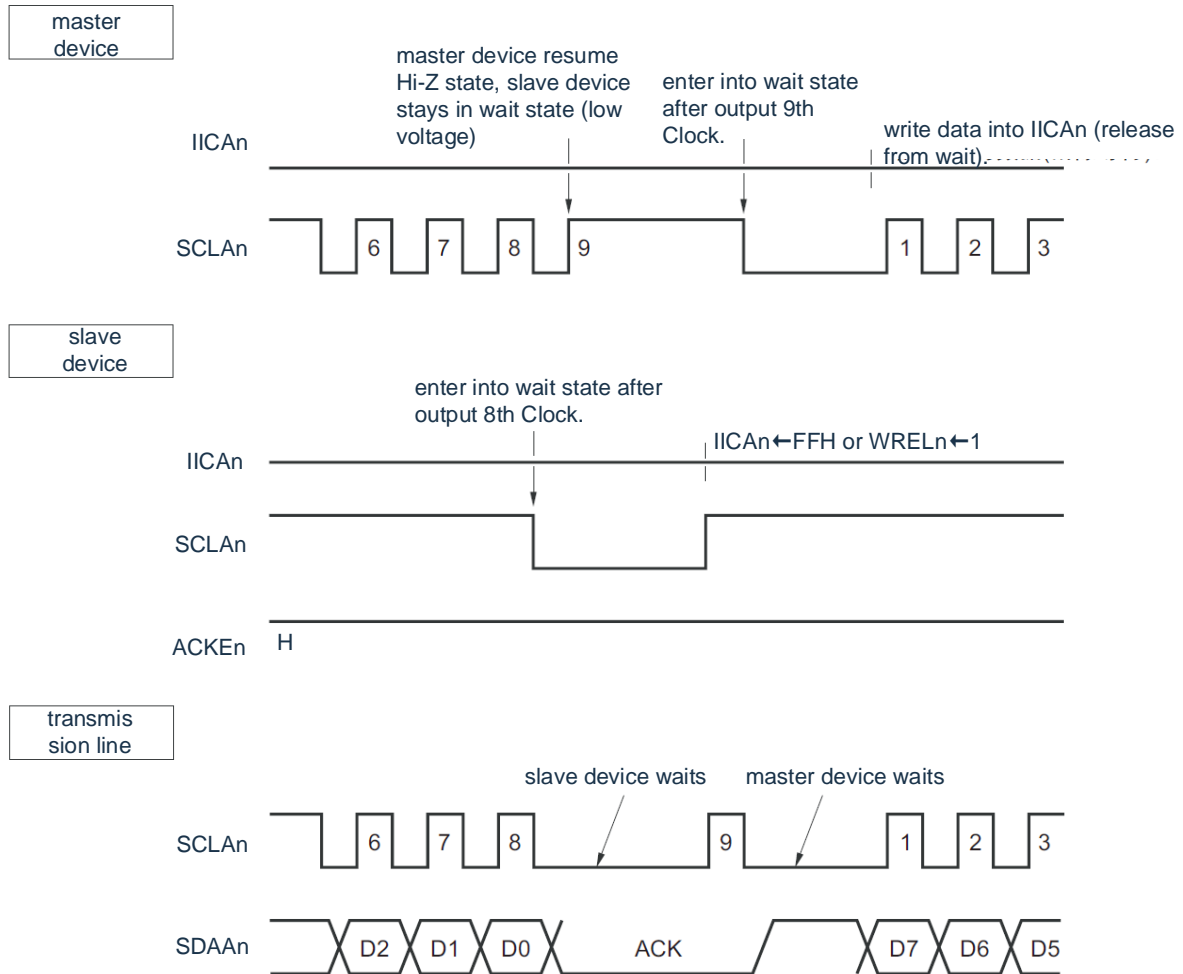
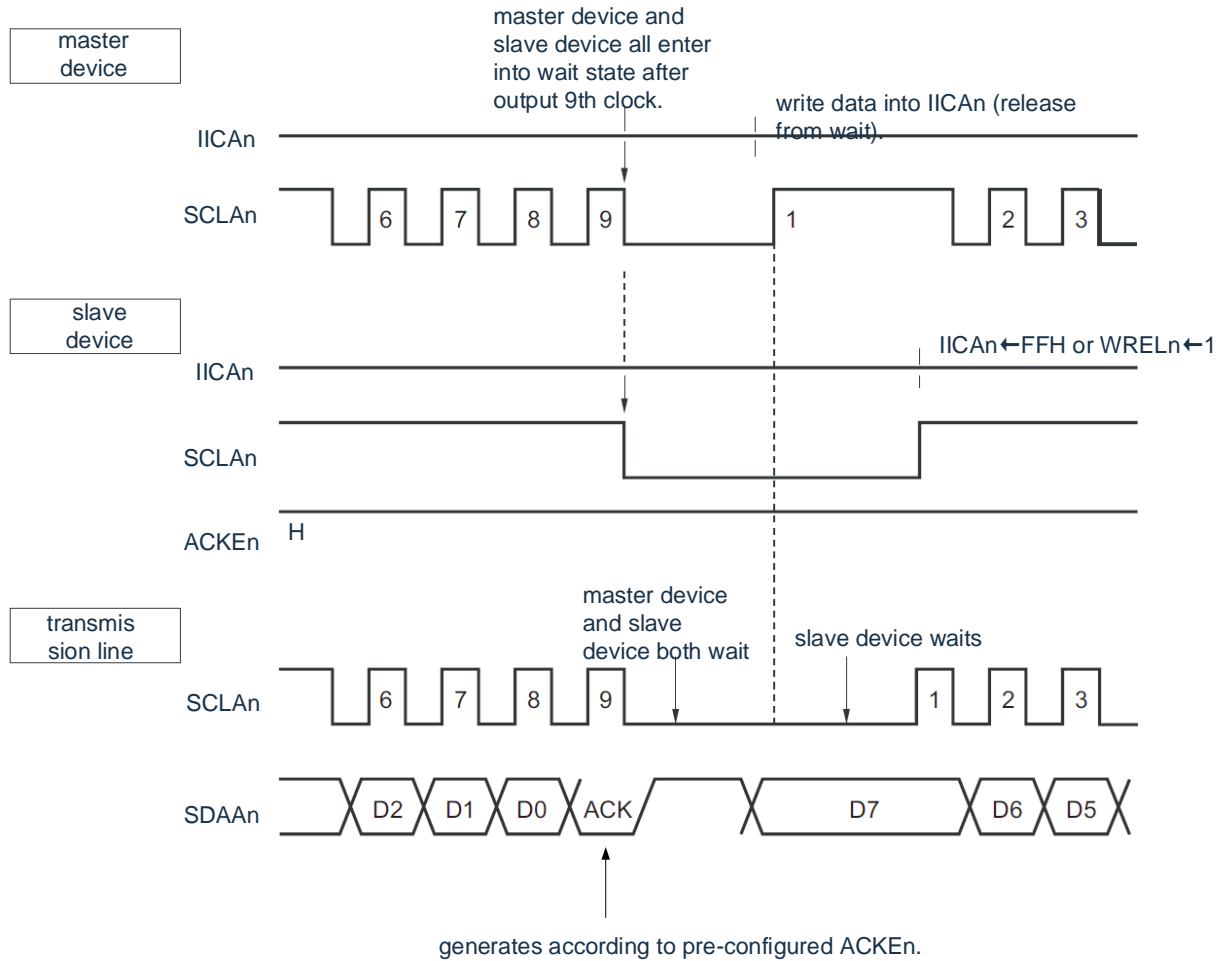


Figure 18-19: Await (2/2)

(2) A situation where both the master and slave devices are waiting for 9 clocks

(Master: Transmit, Slave: Receive, 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 receiver 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 sender side, if data is written to IICAn register, the wait is released. On the sender 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



## 18.5.7 Method of release from wait state

In general, I<sup>2</sup>C can release the wait with the following processing.

- Write data to IICA shift register n (IICAn).
- Set the bit5 (WRELn) of the IICA control register n0 (IICCTLn0) (de-wait).
- Set the bit1 (STTn) of the IICCTLn0 register (generate start condition)<sup>Note</sup>.
- Set the bit0 (SPTn) of the IICCTLn0 register (generate stop condition)<sup>Note</sup>.

Note: Limited to master devices.

If these wait release processes are performed, the I<sup>2</sup>C 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 I<sup>2</sup>C 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

## 18.5.8 Generation timing and waiting control of interrupt requests (INTIICAn)

By setting the IICA control register n0 (IICCTLn0) bit3 (WTIMn), in Table 18-2 The timing shown generates INTIICAn and is subject to wait control.

Table 18-2: I Generation timing and waiting control of INTIICAn

WTIMn	Slave operation			Master operation		
	Address	Data reception	Data transmission	Address	Data reception	Data transmission
0	9 <small>Note 1.2</small>	8 <small>Note 2</small>	8 <small>Note 2</small>	9	8	8
1	9 <small>Note 1.2</small>	9 <small>Note 2</small>	9 <small>Note 2</small>	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

Note 2: If the contents of the received address and the slave address register n(SVAn) are different and the extension code is not received, THE INTIICAn is not generated and does not enter the waiting state.

Remark: The numbers in the table represent the number of clocks for a serial clock. Both interrupt request and wait control are synchronized with the falling edge of the serial clock.

### (1) Sending and receiving addresses

- Slave operation: Independent of the WTIMn bit, the timing of interruptions and waits is determined according to the conditions in Notes 1 and 2 above.
- Master operation: Independent 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:

- Write data to IICA shift register n (IICAn).
- Set the bit5 (WRELn) of the IICA control register n0 (IICCTLn0) (de-wait).
- Set the bit1 (STTn) of the IICCTLn0 register (generate start condition) <sup>Note</sup>.
- Set the bit0 (SPTn) of the IICCTLn0 register (generate stop condition) <sup>Note</sup>.

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 the case of SPIEn=1).

### 18.5.9 Detection method for address matching

In I<sup>2</sup>C-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.

### 18.5.10 Error detection

In I<sup>2</sup>C-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

## 18.5.11 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 is 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) The processing after an interrupt request occurs depending on the subsequent data of the extension code and is processed by software. If an extension code is received while the slave is running, it is participating in the communication even if the addresses are different. For example, if you do not want to run as a slave after receiving an extension code, you must set bit6 (LRELn) of the IICA control register n0 (IICCTLn0). "1" to enter the standby state for the next communication.

Table 18-3: Bit definitions of the main extension codes

Slave address	R/W bit	illustrate
0000000	0	Full call address
11110xx	0	Designation of a 10-bit subordinate address (when the address is authenticated).
11110xx	1	The designation of a 10-bit Slave address (when a read command is issued after the address is the same).

Remark:

1. For extension codes other than those listed above, please refer to the I<sup>2</sup>C-bus datasheet issued by NXP.
2. n=0

## 18.5.12 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.

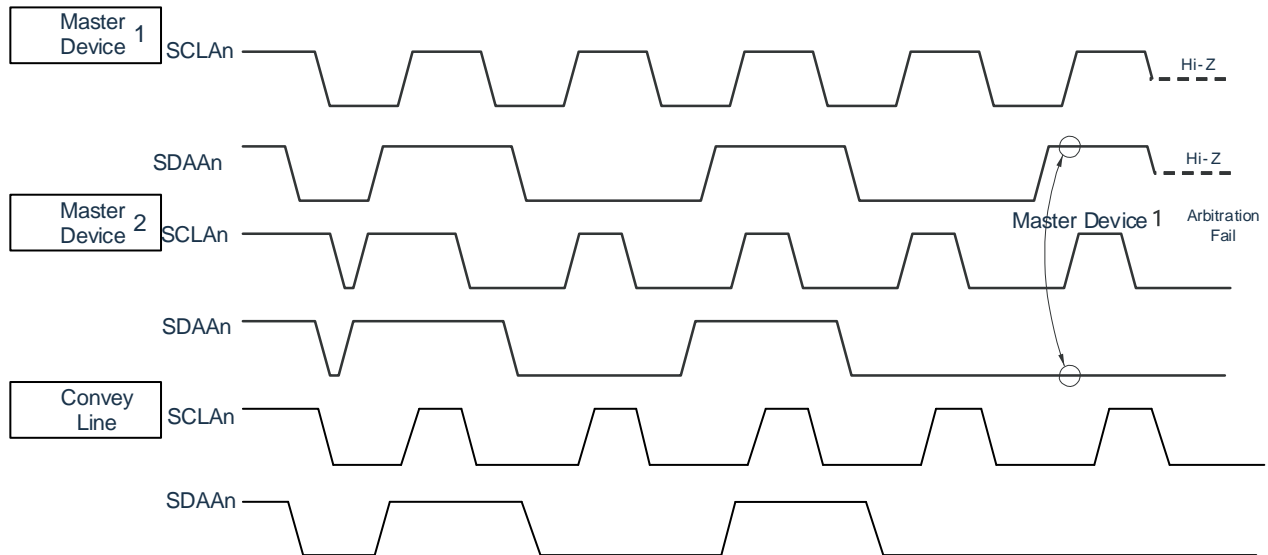
In the event of the next interrupt request (e.g., a stop condition is detected at the 8th or 9th clock), the ALDn bit is "1" via software to detect the failure of the quorum.

For the timing of interrupt requests, please refer to "18.5.8 Generation timing and waiting control of interrupt requests (INTIICAn)".

Remark: STDn: Bit1 of the IICA status register n (IICSn)

STTn: Bit1 of IICA control register n0 (IICCTLn0)

Figure 18-20: Arbitration timing example



Remark: n=0

Table 18-4: 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
Address during sending	The descending edge of the 8th or 9th clock after the byte transfer is <sup>Note 1</sup>
Read and write information after sending the address	
The extension code is being sent during process	
Read and write messages after sending extension codes	
During data sending	
After sending the data, the reply is delivered during the transfer	
A restart condition was detected during data transfer.	when generating a stop condition (SPIEn=1). <sup>Note 2</sup>
A stop condition was detected during data transfer.	
You want to generate a restart condition, but the data is low.	The descending edge of the 8th or 9th clock after the byte transfer is <sup>Note 1</sup>
You want to build a restart condition, but a stop condition is detected.	when generating a stop condition (SPIEn=1). <sup>Note 2</sup>
You want to generate a stop condition, but the data is low.	The descending edge of the 8th or 9th clock after the byte transfer is <sup>Note 1</sup>
You want to generate a restart condition, but SCLAn is low.	

Note 1: When the WTIMn bit (bit3 of the IICA control register n0 (IICCTLn0)) is "1", in the The falling edge of the 9 clocks generates an interrupt request; When the WTIMn bit is "0" and a slave address of the extension code is received, an interrupt request is generated on the descending edge of the 8th clock.

Note 2: When there is a possibility of arbitration, the SPIEn bit must be "1" when the master is running.

Remark:

1. SPIEn: Bit4 of the IICA control register n0 (IICCTLn0)
2. n=0

## 18.5.13 Wake-up function

This is a subordinate function of I<sup>2</sup>C, 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 a 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 18-21, and the flow when the WUPn bit is set to "0" by address matching is shown in Figure 18-22.

Figure 18-21: Flow when the WUPn bit is set to "1"

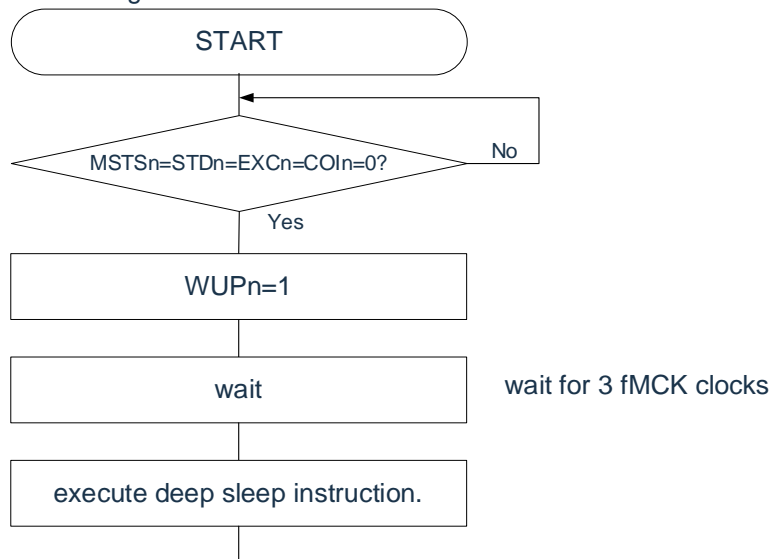
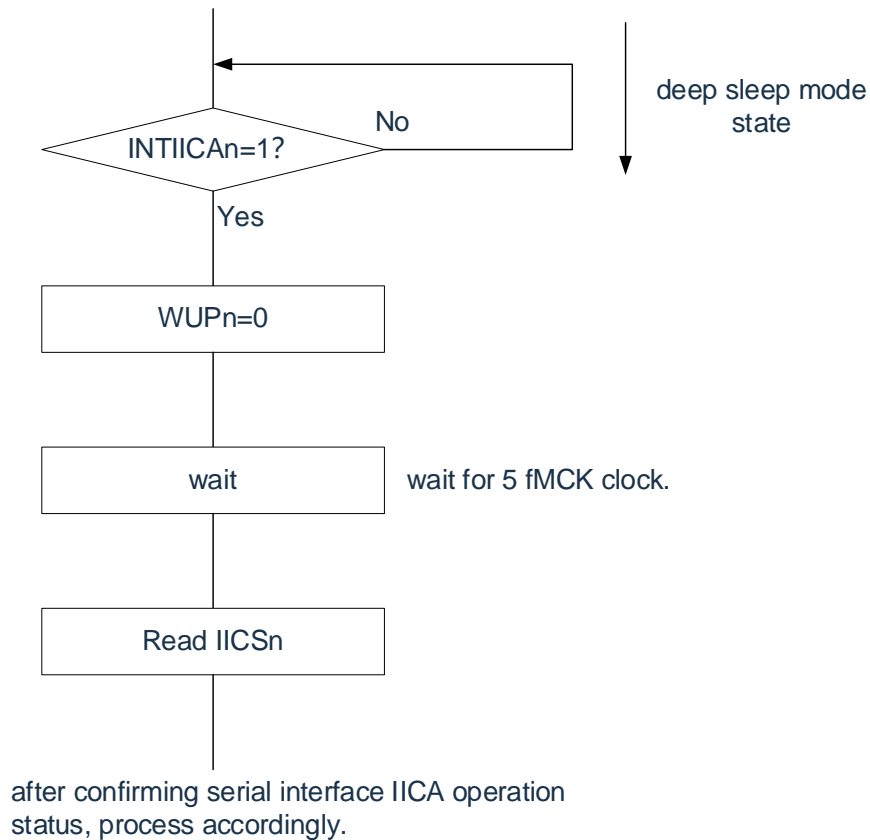


Figure 18-22: Flow when the WUPn bit is set to "0" by address matching (including receiving extension codes)



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 control device: Figure 18-23
- Next IIC communication for slave device operation:

The case of the return via the INTIICAn interrupt: the same flow as Figure 18-22.

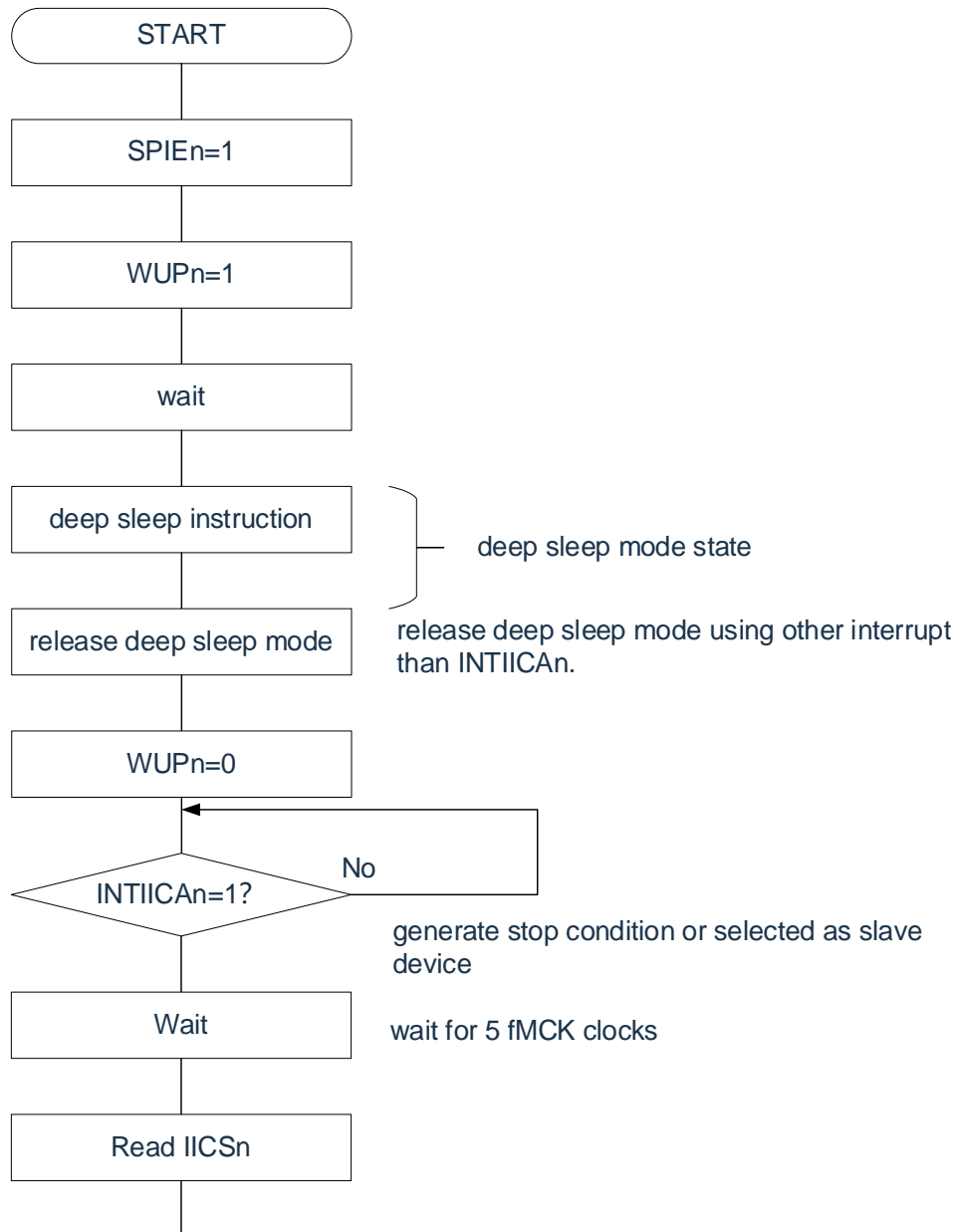
Return from interrupts other than the INTIICAn interrupt: The WUPn bit must be held at "1" to continue operation before the INTIICAn interrupt is generated.

Remark: n=0



Figure 18-23: Operation as a master device after being released from deep sleep mode by an interrupt other than

INTIICAn



after confirming serial interface IICA operation status, process accordingly.

Remark: n=0

## 18.5.14 Communication appointment

- (1) Cases where the communication appointment function is allowed (bit0 (IICRSVn) = 0 of the IICA flag register n (IICFn))

To perform the next master communication without joining the bus, you can send a start condition when the bus is released through a communication appointment. The non-joining bus at this time includes the following two states:

- When the outcome of the arbitration is neither the master nor the slave
- When not operating as a slave device after receiving an extension code (bit 6 (LRELn) of the IICA control register n0 (IICCTLn0) is set to "1" without returning an answer, and the bus is released after exiting communication)

If you set the bit1 (STTn) of the IICCTLn0 register to "1" in the state of not joining the bus, the start condition is automatically generated after the bus is released (the stop condition is detected) and enters the waiting state.

Set the bit4 (SPIEn) of the IICCTLn0 register to "1" after the release of the bus (stop condition detected) is detected by the generated interrupt request signal (INTIICAn), if given IICA shifts the register n (IICAn) to write the address and automatically begins to communicate as the master device. The data written to the IICAn register is invalid until a stop condition is detected.

When stTn is set to "1", it is decided whether to run as a start condition or as a communication appointment depending on the bus state

- Bus is in a release state.....Generate start condition
- Bus is not in the release state (standby state).....Communication appointment

After setting STTn bit to "1" and after a wait time has elapsed, pass the MSTSn bit (IICA status register n (IICSn). bit7) confirm whether it is running as a communication appointment.

The following calculations of the calculation of the wait time must be ensured by the software:

Wait time from setting STTn bit to "1" until the MSTSn flag is confirmed:  

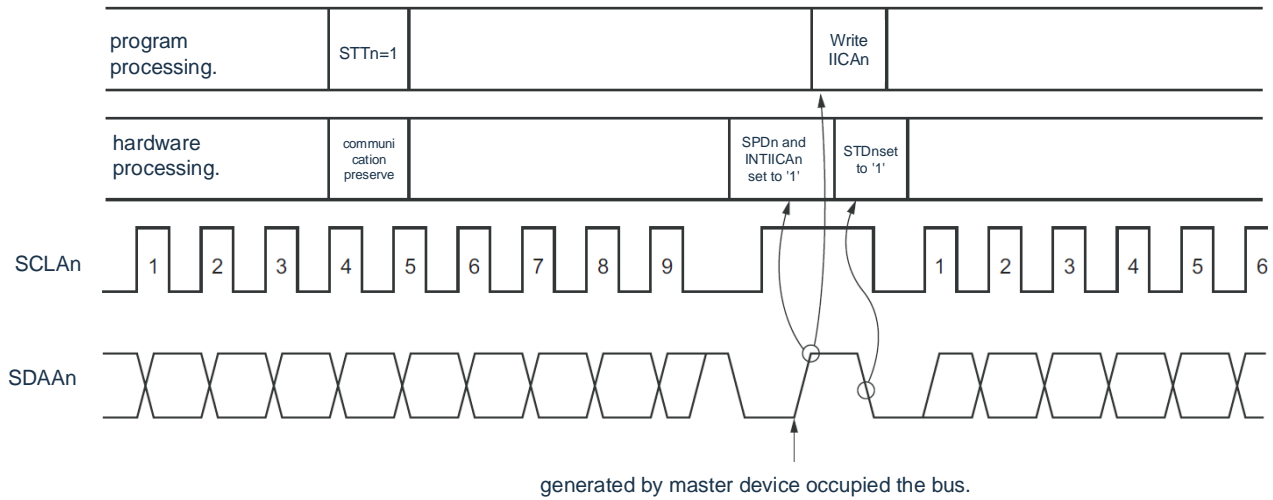
$$(IICWLn \text{ set value} + IICWHn \text{ set value} + 4) / F_{MCK} + T_F \times 2$$

Remark:

1. IICWLn: IICA low-level width setting register n  
IICWHn: IICA high level width setting register n  
TF: Descent time of the SDAAn signal and the SCLAn signal  
FMCK: IICA operating clock frequency
2. n=0

Timing of the communication appointment is shown in the following figure

Figure 18-24: Timing of communication appointment



Remark: IICAn: IICA shift register n

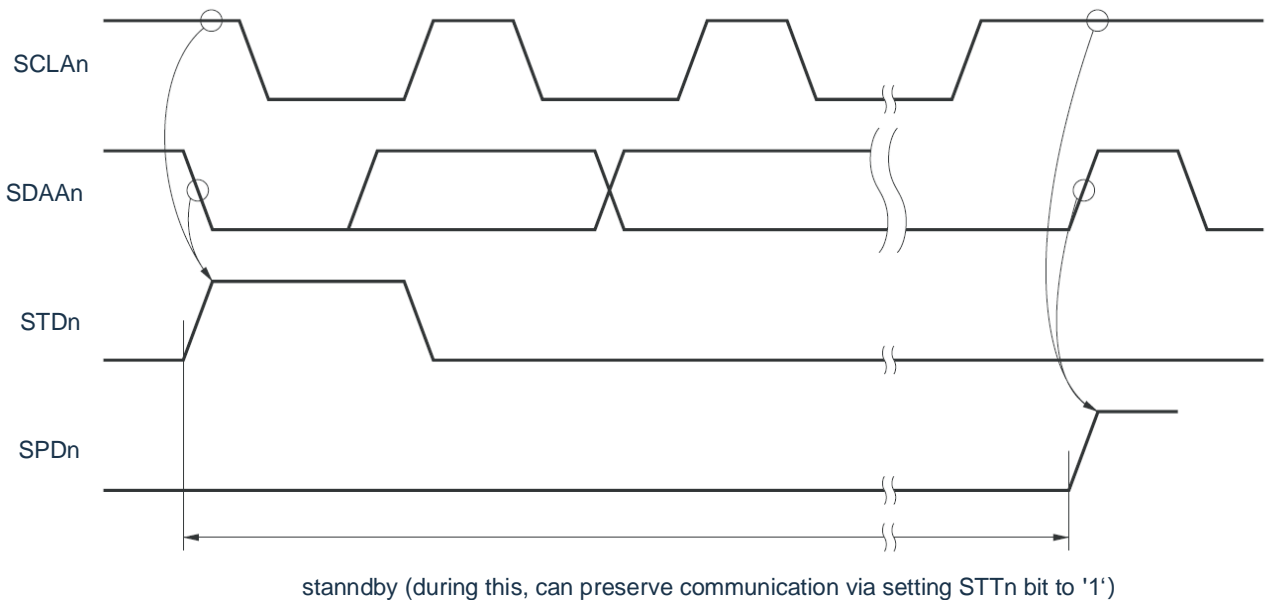
STTn: Bit1 of IICA control register n0 (IICCTLn0)

STDn: Bit1 of the IICA status register n

SPDn: Bit0 of IICA status register n (IICSn)

The communication reservation is accepted by the timing sequence shown in Figure 18-25. After bit 1 (STDn) of IICA status register n (IICSn) becomes "1" and before the stop condition is detected, set bit 1 (STTn) of IICA control register n0 (IICCTLn0) to "1" for communication appointment.

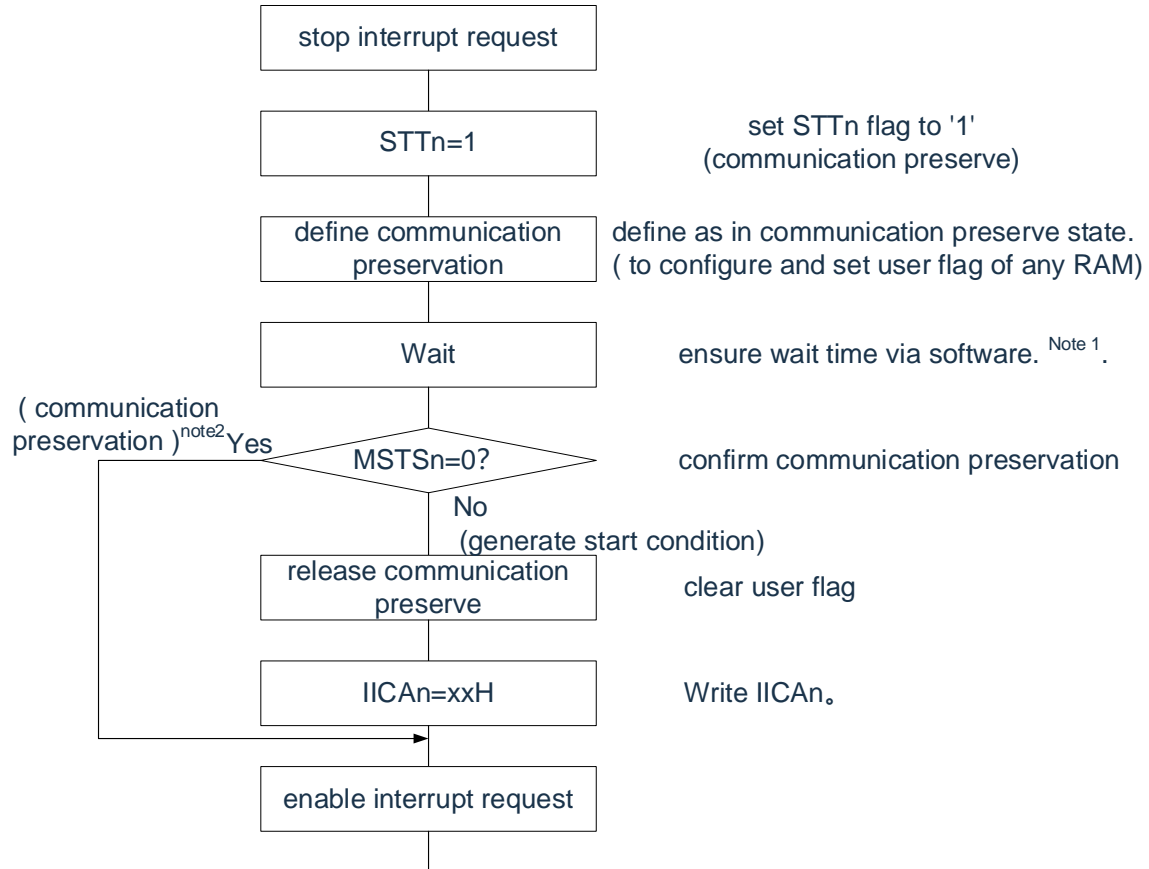
Figure 18-25: Reception timing of communication appointment



Remark: n=0

The steps of the communication appointment are shown in below.

Figure 18-26: Communication appointment step



Note 1: The waiting time is as follows:  $(IICWLn \text{ setting value} + IICWHn \text{ setting value} + 4) / F_{MCK} + T_F \times 2$

Note 2: Write the IICA shift register n (IICAn) by stopping the conditional interrupt request while the communication appointment is running.

Remark:

1. STTn: Bit1 of IICA control register n0 (IICCTLn0)  
MSTSn: Bit7 of the IICA status register n (IICSn)  
IICAn: IICA shift register n  
IICWLn: IICA low-level width setting register n  
IICWHn: IICA high level width setting register n  
T<sub>F</sub>: Descent time of the SDAAn signal and the SCLAn signal  
F<sub>MCK</sub>: IICA operating clock frequency
2. n=0

(2) Case where the communication appointment function is prohibited (bit0 (IICRSVn) of the IICA flag register n (IICFn) = 1)

If bit1 (STTn) of IICA control register n0 (IICCTLn0) is set to "1" during bus communication, the request is rejected and no start condition is generated. In this case, the non-joining bus includes the following two states

- When the outcome of the arbitration is neither the master nor the slave

- When not operating as a slave device after receiving an extension code (bit 6 (LRELn) of the IICCTLn0 register is set to "1" instead of returning an answer, and the bus is released after exiting communication)

The STCFn (bit 7 of the IICFn register) can be used to confirm whether a start condition has been generated or a request has been rejected. Since it takes  $5 f_{MCK}$  clocks from the time the STTn bit is "1" to the time the STCFn bit is set to "1", this time must be ensured by software.

Remark: n=0

## 18.5.15 Other cautions

(1) The case where the STCENn bit is "0"

Just after I<sup>2</sup>C is allowed to run (IICEn=1), it is considered a communication state (IICBSYn=1) regardless of the actual bus state. To perform master communication in a state where no stop condition is detected, the stop condition must be made and the master communication must be performed after the bus is released. For multi-master, master communication cannot occur in a state where the bus is not released (no stop condition detected). Generate stop conditions in the following order::

- ① Set IICA control register n1 (IICCTLn1).
- ② Set bit7 (IICEn) of the IICA control register n0 (IICCTLn0) to "1".
- ③ Set the bit0 (SPTn) of the IICCTLn0 register to "1".

(2) The case where stcenn bit is "1"

Just after I<sup>2</sup>C is allowed to run (IICEn=1), it is considered a release state (IICBSYn=0) regardless of the actual bus state. Therefore, when generating the first starting condition (STTn=1), in order not to disrupt other communications, it is necessary to confirm that the bus has been released.

(3) I<sup>2</sup>C communication with other devices ongoing

When the SDAAn pin is low and the SCLAn pin is high, I<sup>2</sup>C macros are considered SDAAn citations if I<sup>2</sup>C is allowed to run and participate in communication in the middle the foot changes from high to low (start condition detected). If the value on the bus is recognized as an extension code at this point, a reply is returned that interferes with I<sup>2</sup>C communication with other devices. To avoid this, I<sup>2</sup>C must be started in the following order:

- ① Clear the bit4 (SPIEn) of the IICCTLn0 register to "0" to disable the generation of an interrupt request signal (INTIICAn) when a stop condition is detected.
- ② Set the bit7 (IICEn) of the IICCTLn0 register to "1", allowing I<sup>2</sup>C to run.
- ③ Wait for the start condition to be detected.
- ④ Before returning the answer (within 4 to 72 F<sub>MCK</sub> clocks after setting the IICEn bit to "1"), set bit 6 (LRELn) of the IICCTLn0 register to "1" to force the detection to be disabled.

(4) After setting the STTn bit and SPTn bit (bit1 and bit0 of the IICCTLn0 register), the reset before clearing "0" is prohibited.

(5) If a communication appointment is made, the SPIEn bit (bit 4 of the IICCTLn0 register) must be set to "1" to generate an interrupt request when a stop condition is detected. After the interrupt request is generated, the communication data is written to the IICA shift register n (IICAn) to start the transmission. If no interrupt occurs when the stop condition is detected, the communication stops in the wait state because no interrupt request is generated at the start of communication. However, when the MSTSn bit (bit 7 of the IICA status register n(IICSn)) is detected by software, it is not necessary to set the SPIEn bit to "1".

Remark: n=0

## 18.5.16 Communication operation

Here, the following three running steps are represented by a flowchart.

### (1) Master operation of a single-master system

The flowchart used as a master device in a single master system is shown below.

This process is broadly divided into "initial setup" and "communication processing". Perform the Initial Setup section at startup, and if communication with the slave is required, perform the Communication Processing section after the preparation required to communicate.

### (2) Master operation of multi-master system

In a multi-master system of the I<sup>2</sup>C bus, it is not possible to judge whether the bus is in the release state or in use during the stage of participating in the communication based on the specifications of the I<sup>2</sup>C 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 setup", "communication waiting" and "communication processing". The processing designated as a slave due to the failure of the arbitration is omitted here, and only the processing used as the master device is omitted. Join the bus after performing the Initial Setup section at startup, and then wait for a communication request from the master device or a designation of the slave device via Communication Wait. The actual communication is the "Communication Processing" section, which supports arbitration with other master devices in addition to data sending and receiving with the slave.

### (3) Slave operation

An example of an I<sup>2</sup>C bus slave is shown below.

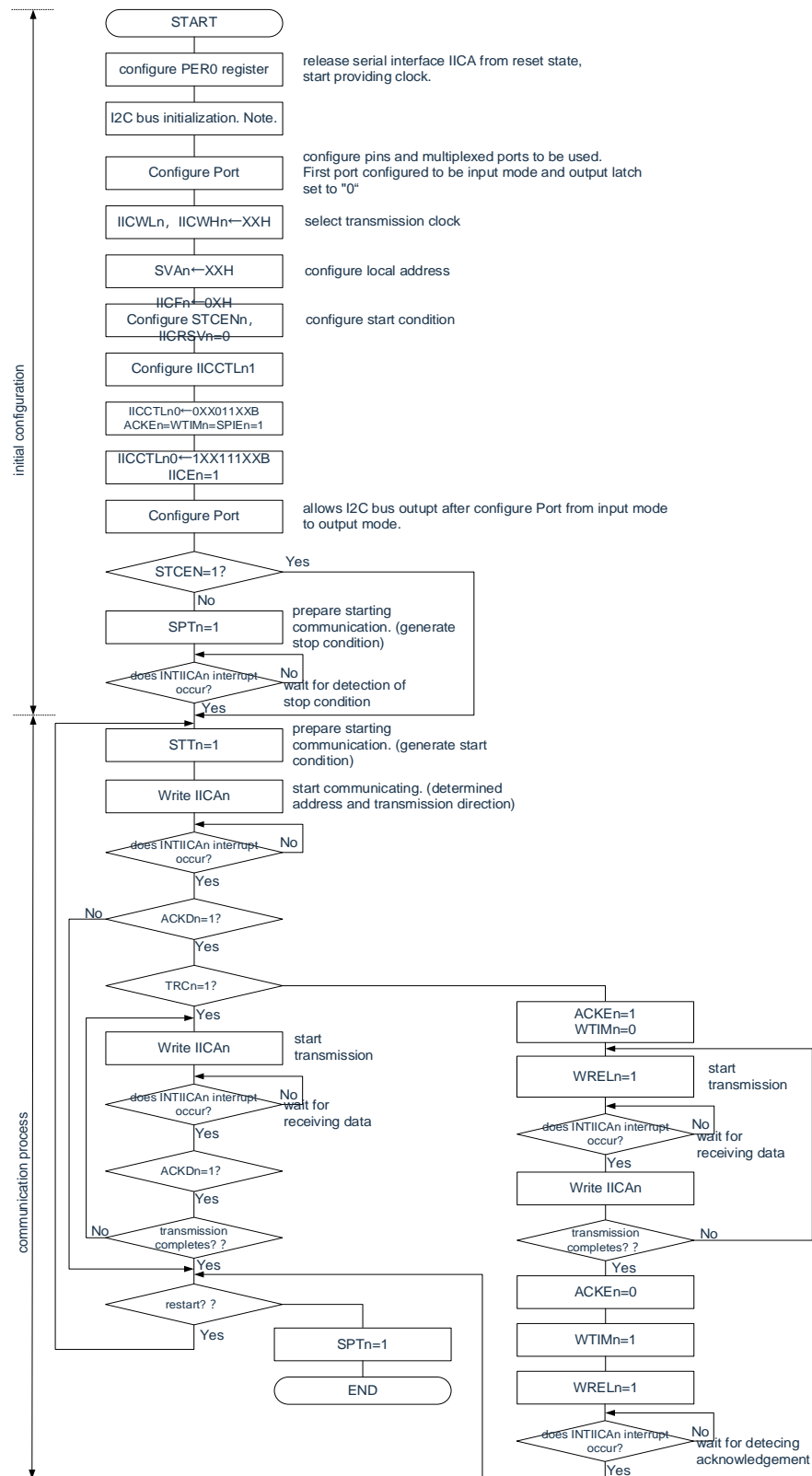
When used as a slave, it starts running with an interrupt. Perform the "Initial Setup" section at startup, then through "Communication Wait" and wait for the INTIICAn interrupt to occur. If an INTIICAn interrupt occurs, the communication status is determined and the flag is passed to the main processing department.

By checking each flag, the required "communication processing" is carried out.

Remark: n=0

### (1) Master operation of a single-master system

Figure 18-27: Master operation of the single master control system



Note: I<sup>2</sup>C-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.

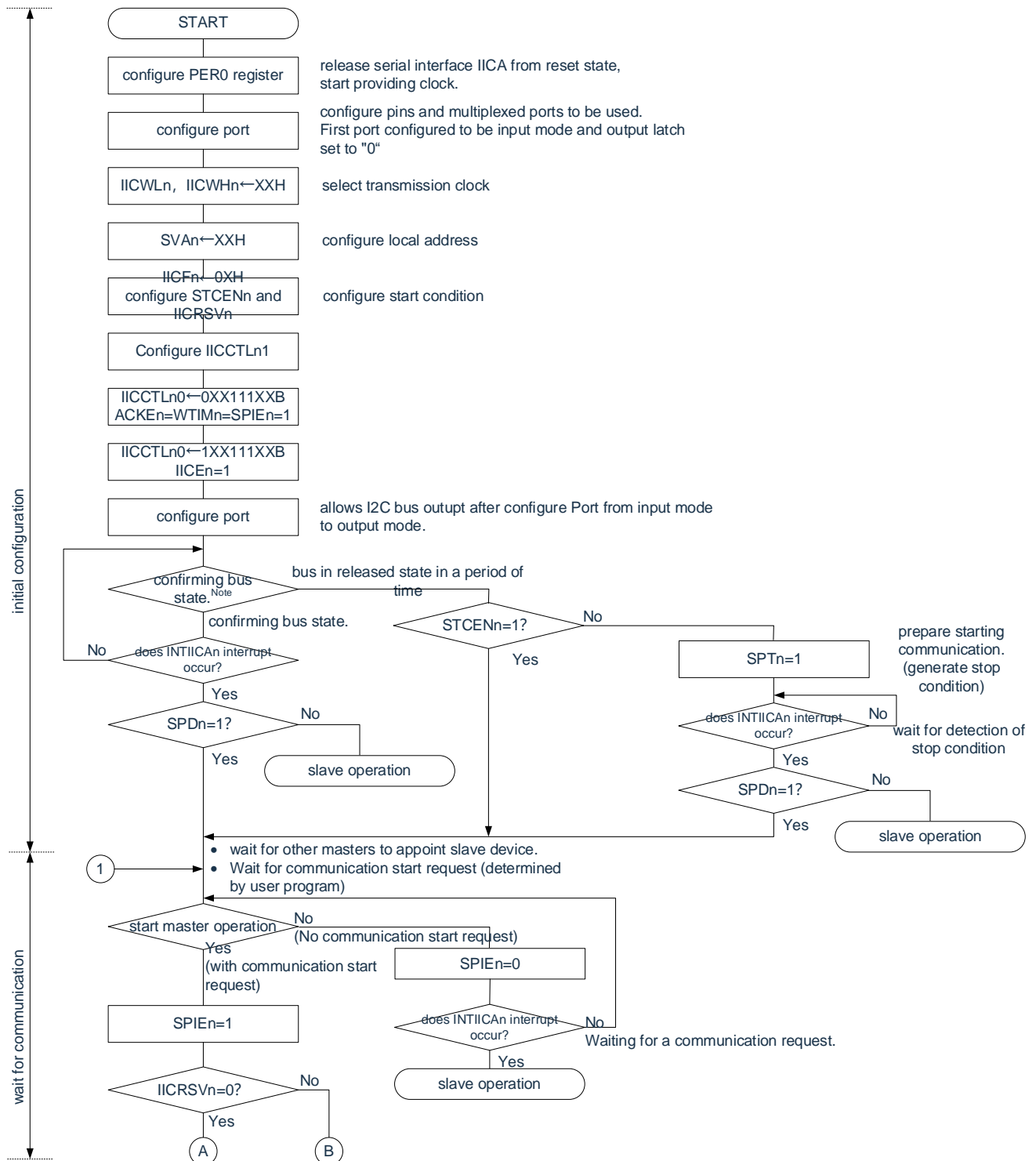


Remark:

1. The format of sending and receiving must conform to the specifications of the product in communication
2.  $n=0$ .

## (2) Master operation of multi-master system

Figure 18-28: Operation of a multi-master system (1/3)



Note: You must confirm that the bus is in a free state (CLDn bit = 1, DADn bit = 1) for a certain period of time (for example, frame 1). When the SDAAn pin is fixed low, it must be determined whether to release the I<sup>2</sup>C-bus (SCLAn pin and SDAAn) according to the specifications of the product in the communication Pin is high).

Remark: n=0

```

graph TD
    A((A)) --> STTn[STTn=1]
    STTn --> Wait[Wait]
    Wait --> MSTSn{MSTSn=0?}
    MSTSn -- Yes --> C((C))
    MSTSn -- No --> INTIICAn{does INTIICAn interrupt occur?}
    INTIICAn -- Yes --> EXCn{EXCn=1 or COIn=1?}
    INTIICAn -- No --> MSTSn
    EXCn -- Yes --> B((B))
    EXCn -- No --> MSTSn
    B --> INTIICAn
    C --> A
  
```

communication process

A allow communication preservation

STTn=1 prepare starting communication. (generate stop condition)

Wait ensure wait time via software. Note.

MSTSn=0?

Yes

No

when detecting stop condition, generate start condition via communication preservation function, then enter into wait state.

C

does INTIICAn interrupt occur?

No

Yes

wait to release bus. (in communication preservation)

EXCn=1 or COIn=1?

Yes

slave operation

No

```
graph TD
    B((B)) --> D1{IICBSYn=0?}
    D1 -- No --> D2{does INTIICAn interrupt occur?}
    D1 -- Yes --> D4((D))
    D4 --> S1[STTn=1]
    S1 --> S2[Wait]
    S2 --> D3{STCFn=0?}
    D3 -- No --> D2
    D3 -- Yes --> C((C))
    D2 -- No --> D2
    D2 -- Yes --> D5{EXCn=1 or COIn=1?}
    D5 -- No --> D6((D))
    D5 -- Yes --> S3([slave operation])
    S3 --> D6
    D6 --> D1
```

The flowchart illustrates the slave operation sequence for I2C communication. It begins with a decision diamond labeled 'IICBSYn=0?'. If the answer is 'No', the flow proceeds to a decision diamond labeled 'does INTIICAn interrupt occur?'. If the answer is 'Yes', the flow proceeds to a decision diamond labeled 'EXCn=1 or COIn=1?'. If the answer is 'No', the flow proceeds to a terminal node labeled 'D'. If the answer is 'Yes', the flow proceeds to a rounded rectangle labeled 'slave operation', which then leads to terminal node 'D'. If the answer to 'IICBSYn=0?' is 'Yes', the flow proceeds to a terminal node labeled 'D', which then leads to a rectangle labeled 'STTn=1', followed by a rectangle labeled 'Wait', and then a decision diamond labeled 'STCFn=0?'. If the answer to 'STCFn=0?' is 'No', the flow proceeds to the 'does INTIICAn interrupt occur?' decision diamond. If the answer is 'Yes', the flow proceeds to a terminal node labeled 'C'. The 'does INTIICAn interrupt occur?' decision diamond has a 'No' path that loops back to the 'IICBSYn=0?' decision diamond and a 'Yes' path that leads to the 'EXCn=1 or COIn=1?' decision diamond. The 'EXCn=1 or COIn=1?' decision diamond has a 'No' path that leads to terminal node 'D' and a 'Yes' path that leads to the 'slave operation' rounded rectangle. A vertical arrow on the left side of the flowchart is labeled 'communication process'.

1. IICWLn: IICA low-level width setting register n  
IICWHn: IICA high level width setting register n  
T<sub>F</sub>: Descent time of the SDAAn signal and the SCLAn signal  
F<sub>MCK</sub>: IICA operating clock frequency
2. n=0

```

graph TD
    C((C)) --> W1[Write IICAn]
    W1 --> D1{does INTIICAn interrupt occur?}
    D1 -- No --> W2[wait for detecting acknowledgement]
    D1 -- Yes --> D2{MSTSn=1?}
    D2 -- No --> 2((2))
    D2 -- Yes --> D3{ACKDn=1?}
    D3 -- No --> 2
    D3 -- Yes --> D4{TRCn=1?}
    D4 -- No --> 2
    D4 -- Yes --> W3[WTIMn=1]
    W3 --> W4[Write IICAn]
    W4 --> D5{does INTIICAn interrupt occur?}
    D5 -- No --> W5[wait for transmitting data]
    D5 -- Yes --> D6{MSTSn=1?}
    D6 -- No --> 2
    D6 -- Yes --> D7{ACKDn=1?}
    D7 -- No --> 2
    D7 -- Yes --> D8{transmission completes?}
    D8 -- No --> 2
    D8 -- Yes --> D9{restart?}
    D9 -- No --> W6[SPTn=1]
    W6 --> END1([END])
    D9 -- Yes --> W7[STTn=1]
    W7 --> C2((C))
    
    C2 --> D10{EXCn=1 or COLn=1?}
    D10 -- No --> 1((1))
    D10 -- Yes --> SO([slave operation])
    
    1 --> D11{does not participant communication}
    D11 --> 1
    
    2 --> W8[ACKEn=1  
WTIMn=0]
    W8 --> W9[WRELn=1]
    W9 --> D12{does INTIICAn interrupt occur?}
    D12 -- No --> W10[wait for receiving data]
    D12 -- Yes --> D13{MSTSn=1?}
    D13 -- No --> 2
    D13 -- Yes --> W11[Read IICAn]
    W11 --> D14{transmission completes?}
    D14 -- No --> 2
    D14 -- Yes --> W12[ACKEn=0]
    W12 --> W13[WTIMn=1]
    W13 --> W14[WRELn=1]
    W14 --> D15{does INTIICAn interrupt occur?}
    D15 -- No --> W16[wait for detecting acknowledgement]
    D15 -- Yes --> D16{MSTSn=1?}
    D16 -- No --> 2
    D16 -- Yes --> 2
  
```

The flowchart illustrates the I2C communication process, divided into two main sections: **communication process** (transmission) and **communication process** (reception).

**Transmission Process:**

- Starts at connector **C** with the action **Write IICAn** (Start communication. Specify address and transfer direction).
- Decision: **does INTIICAn interrupt occur?**
  - No:** **wait for detecting acknowledgement**.
  - Yes:** Proceeds to **MSTS<sub>n</sub>=1?**
- Decision: **MSTS<sub>n</sub>=1?**
  - No:** Connects to connector **2**.
  - Yes:** Proceeds to **ACKD<sub>n</sub>=1?**
- Decision: **ACKD<sub>n</sub>=1?**
  - No:** Connects to connector **2**.
  - Yes:** Proceeds to **TRC<sub>n</sub>=1?**
- Decision: **TRC<sub>n</sub>=1?**
  - No:** Connects to connector **2**.
  - Yes:** Proceeds to **WTIM<sub>n</sub>=1**.
- Action: **Write IICAn** (start transmission).
- Decision: **does INTIICAn interrupt occur?**
  - No:** **wait for transmitting data**.
  - Yes:** Proceeds to **MSTS<sub>n</sub>=1?**
- Decision: **MSTS<sub>n</sub>=1?**
  - No:** Connects to connector **2**.
  - Yes:** Proceeds to **ACKD<sub>n</sub>=1?**
- Decision: **ACKD<sub>n</sub>=1?**
  - No:** Connects to connector **2**.
  - Yes:** Proceeds to **transmission completes?**
- Decision: **transmission completes?**
  - No:** Connects to connector **2**.
  - Yes:** Proceeds to **restart?**
- Decision: **restart?**
  - No:** Proceeds to **SPT<sub>n</sub>=1**, then **END**.
  - Yes:** Proceeds to **STT<sub>n</sub>=1**, then connector **C**.

**Reception Process:**

- Starts at connector **2** with the action **ACKEn=1, WTIMn=0**.
- Action: **WRELn=1** (start receiving).
- Decision: **does INTIICAn interrupt occur?**
  - No:** **wait for receiving data**.
  - Yes:** Proceeds to **MSTS<sub>n</sub>=1?**
- Decision: **MSTS<sub>n</sub>=1?**
  - No:** Connects to connector **2**.
  - Yes:** Proceeds to **Read IICAn**.
- Decision: **transmission completes?**
  - No:** Connects to connector **2**.
  - Yes:** Proceeds to **ACKEn=0**.
- Action: **ACKEn=0**.
- Action: **WTIMn=1**.
- Action: **WRELn=1**.
- Decision: **does INTIICAn interrupt occur?**
  - No:** **wait for detecting acknowledgement**.
  - Yes:** Proceeds to **MSTS<sub>n</sub>=1?**
- Decision: **MSTS<sub>n</sub>=1?**
  - No:** Connects to connector **2**.
  - Yes:** Connects to connector **2**.

**Slave Operation:**

- Connector **C** leads to decision: **EXC<sub>n</sub>=1 or COL<sub>n</sub>=1?**
  - No:** Connects to connector **1**.
  - Yes:** Proceeds to **slave operation**.
- Connector **1** leads to decision: **does not participant communication**.

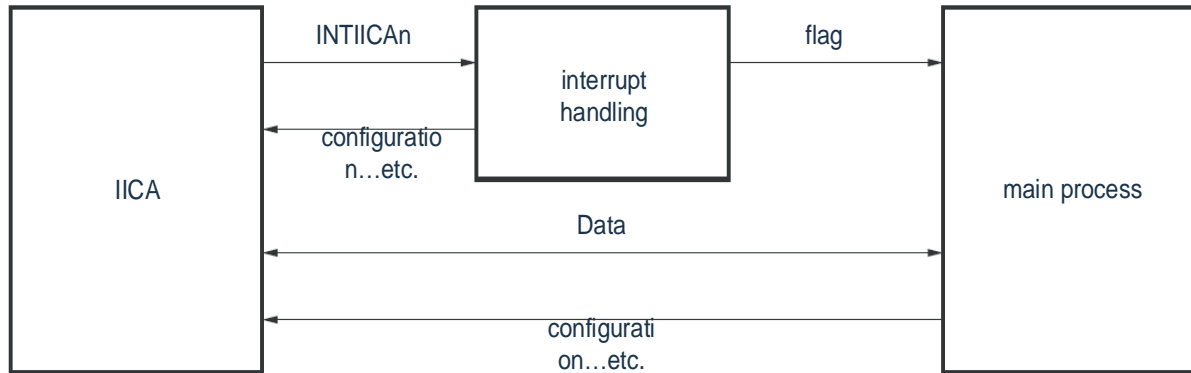
1. The format of transmission and reception must conform to the specifications of the product in the communication.
2. In the case of being used as a master device in a multi-master system, the MSTSn bit must be read at each time an INTIICAn interrupt occurs to confirm the arbitration result.
3. In the case of being used as a slave in a multi-master system, the IICA status register n (IICSn) and the IICA flag register must be passed at each INTIICAn interrupt n (IICFn) confirms the status and decides on future processing.
4. n=0

### (3) Slave operation

The processing steps for a slave run are as follows.

Slave operations are basically event-driven, so they need to be handled through INTIICAn interrupts (large changes to the operating state such as stop condition detection in communications) need to be handled).

In this description, it is assumed that the data communication does not support extension codes, THATICAn interrupt processing only performs state transition processing, and that the actual data communication is carried out by the main processing department.



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: Not in the state of data communication
- Communication mode: The status of the data communication in progress (detection of valid address ~ detection of stop condition, response of the master device not detected, address different).

#### ② Ready flag

This flag indicates that data communication can take place. In the usual data communication, as with the INTIICAn interrupt, the interrupt processing department is placed and cleared by the main processing department. When communication begins, the flag is cleared by interrupt handling. However, when sending the first data, interrupt processing does not set the ready flag in place, so the first data is sent without clearing the flag (address matching is interpreted as the next data request).

#### ③ Communication direction flag

This flag indicates the direction of communication, which is the same as the value of the TRCn bit.

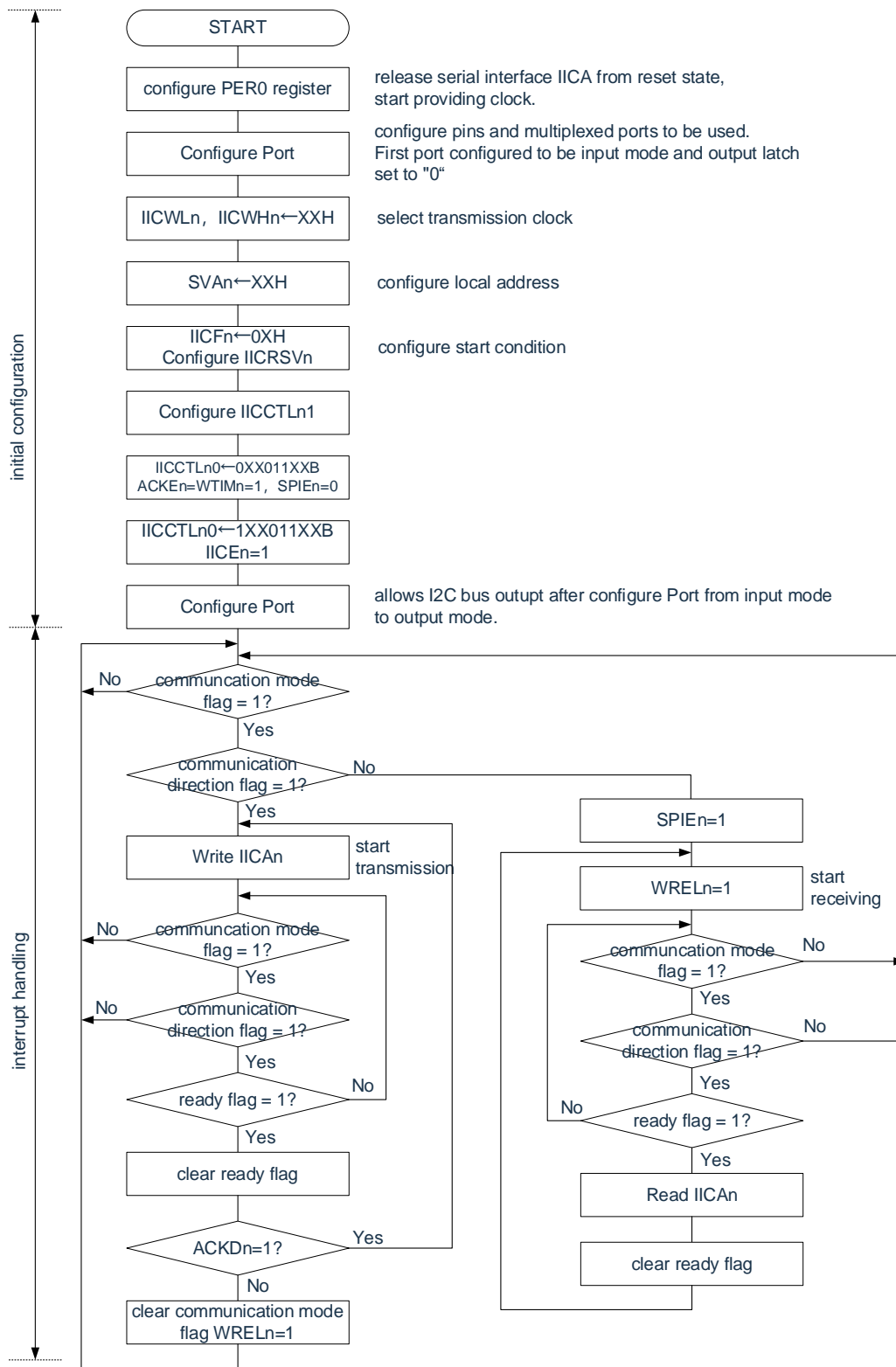
Remark: n=0

The operation of the main processing unit of the slave is as follows.

Start the serial interface IICA and wait to become communicative. If it becomes communicatable, the communication mode flag and the ready flag are used to communicate (because the processing of the stop condition and start condition is carried out by interrupt, the status is confirmed here by the flag).

At send time, the send is repeated until the master device does not return a reply. If the master does not return an Ack, the communication ends. At the time of receiving, receive the required amount of data. If the communication ends, no reply is returned at the next data. After that, the master device generates a stop condition or a restart condition, thereby exiting the communication state.

Figure 18-29: Slave operation step (1)



Remark:

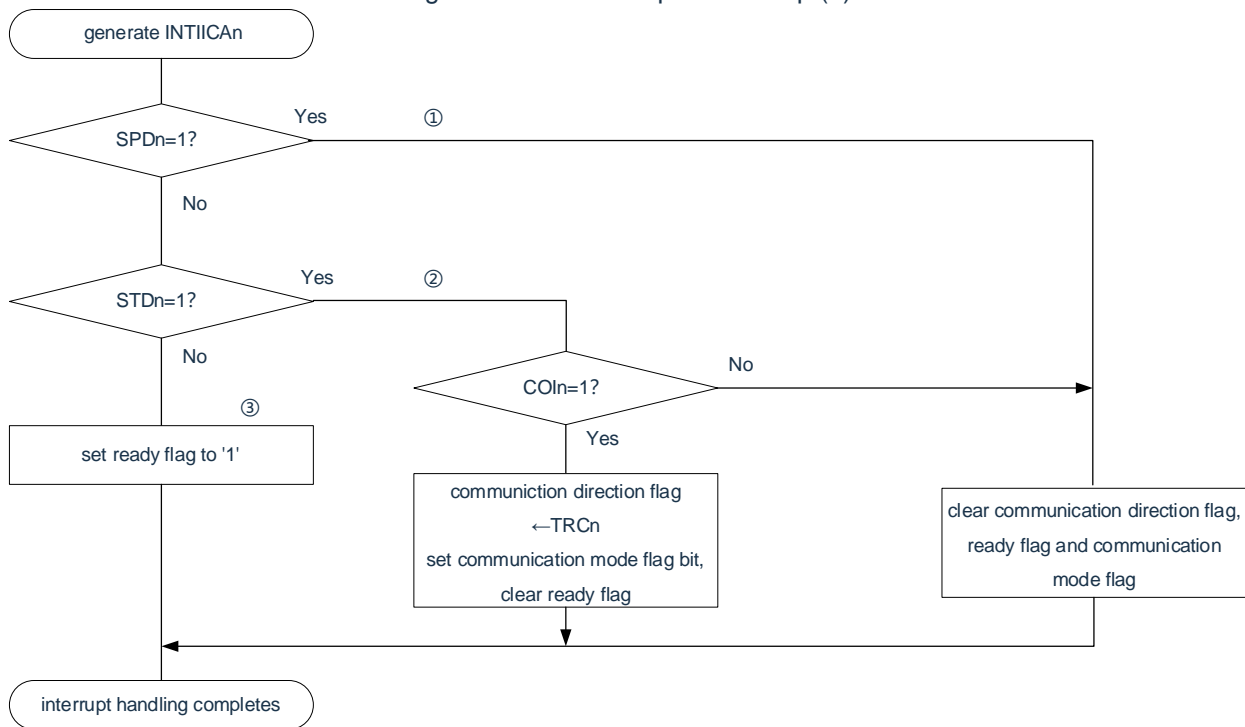
1. The format of transmission and reception must conform to the specifications of the product in communication.
2.  $n=0$

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 If a stop condition is generated, the communication ends.
- ② If a start condition is generated, the address is confirmed. If the addresses are different, the communication ends. If the addresses are the same, set to communication mode and dismiss the wait, then return from interrupt (clear the ready flag).
- ③ When sending and receiving data, the I<sup>2</sup>C bus remains waiting and returns from the interrupt as soon as the ready flag is set.

Remark: The above ① to ③ correspond to ① to ③ of “Figure 18-30: Slave operation step (2)”.

Figure 18-30: Slave operation step (2)



Remark: n=0



## 18.5.17 Timing of I<sup>2</sup>C 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.

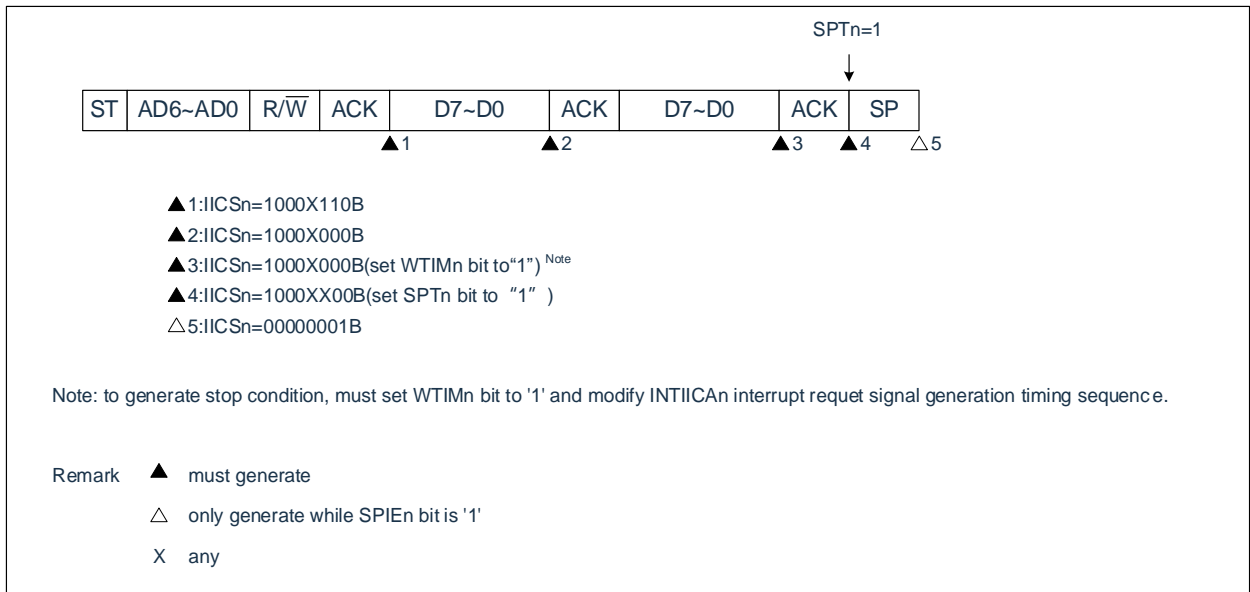
Remark:

1. ST: Start condition  
AD6~AD0: Address  
R/W: The specified transmission direction  
ACK: Acknowledge  
D7~D0: Data  
SP: Stop condition
2. n=0

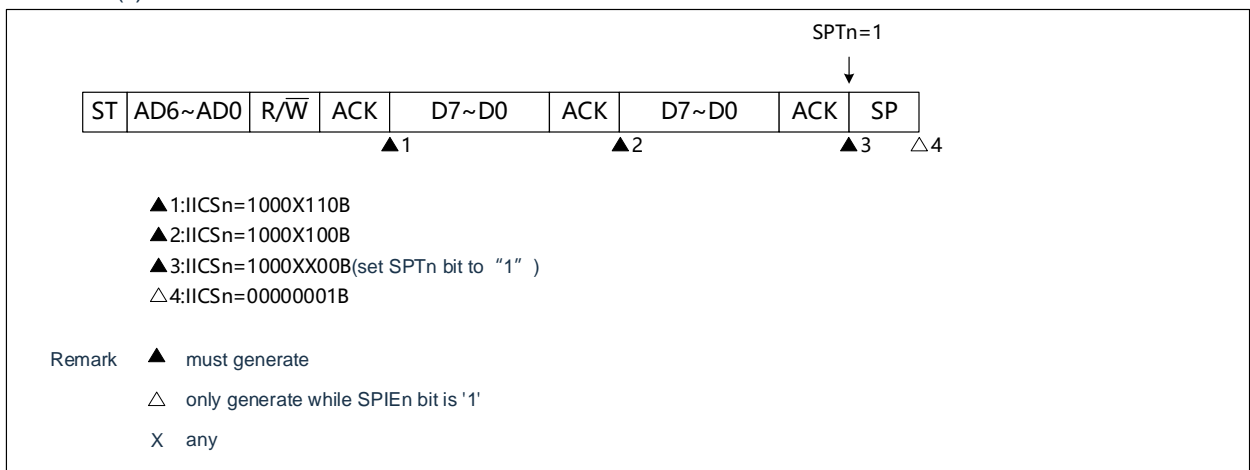
### (1) Mastet operation

#### (a) Start~Address~Data~Data~Stop(transmit and receive)

##### (i) hen WTIMn=0



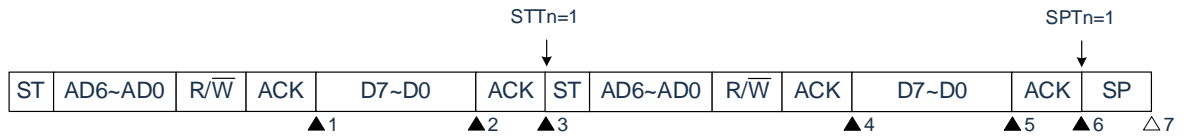
##### (ii) When WTIMn=1



Remark: n=0

(b) Start~Address~Data~Start~Address~Data~Stop(Restart)

(i) When  $WTIM_n=0$



- ▲1: ICSn=1000X110B  
 ▲2: ICSn=1000X000B(set WTIMn bit to "1") <sup>Note1</sup>  
 ▲3: ICSn=1000XX00B(set WTIMn bit to "0". Note 2 and set STTn bit to "1")  
 ▲4: ICSn=1000X110B  
 ▲5: ICSn=1000X000B(set WTIMn bit to "1") <sup>Note3</sup>  
 ▲6: ICSn=1000XX00B(set SPTn bit to "1" )  
 △7: ICSn=00000001B

Note1. to generate start condition, must set WTIIMn bit to '1' and modify INTIICAn interrupt request signal generation timing sequence.

2. To recover original configuration, must set WTIMn bit to '0'.

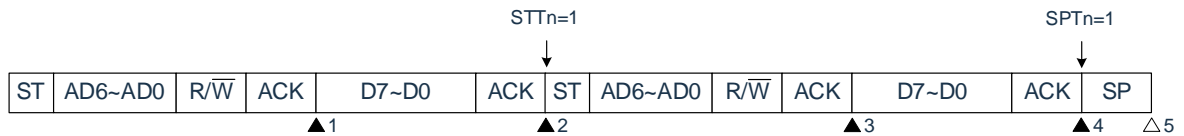
3. to generate stop condition, must set WITIMn bit to '1' and modify INTIICAn interrupt request signal generation timing sequence.

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X Any

(ii) When  $WTIM_n=1$



- ▲1: ICSn=1000X110B
- ▲2: ICSn=1000XX00B(set STTn bit to "1" )
- ▲3: ICSn=1000X110B
- ▲4: ICSn=1000XX00B(set SPTn bit to "1" )
- △5: ICSn=00000001B

Remark ▲ must generate

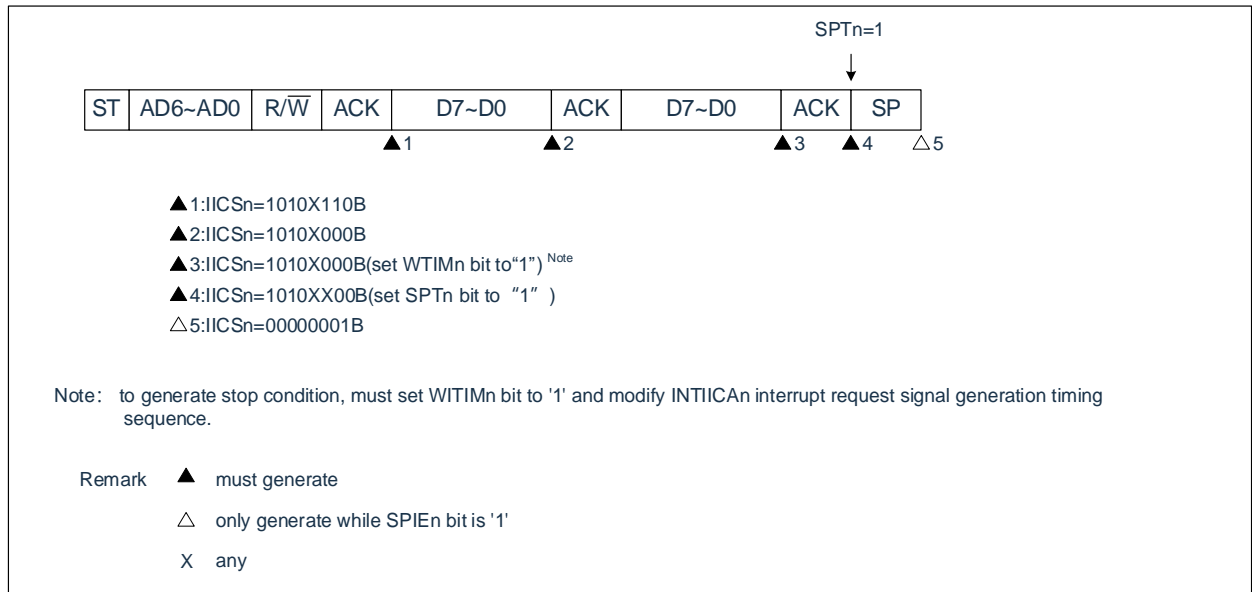
△ only generate while SPIEn bit is '1'

X any

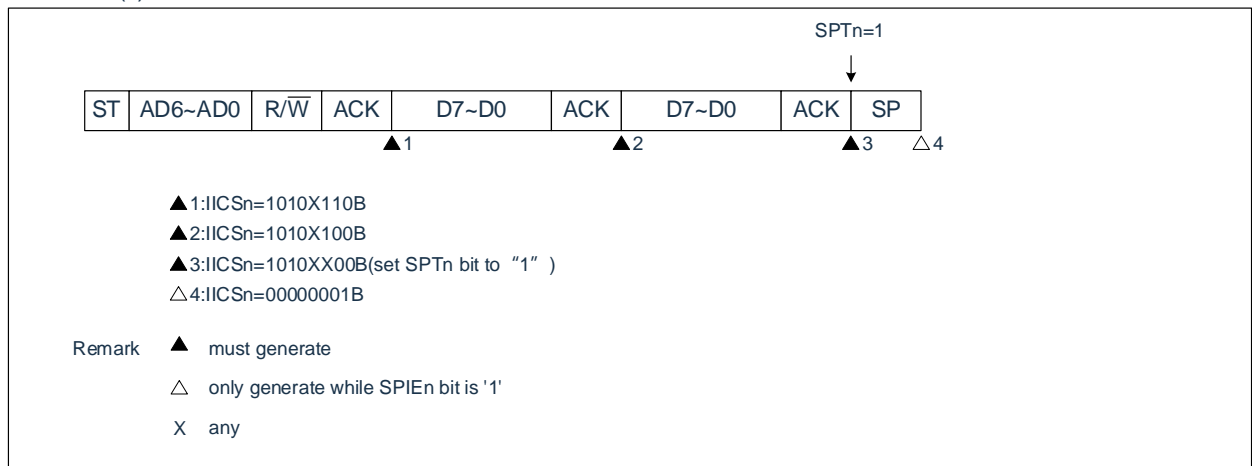
Remark:  $n=0$

(c) Start~Code~Data~Data~Stop(transmit expansion bit)

(i) When WTIMn=0



(ii) When WTIMn=1

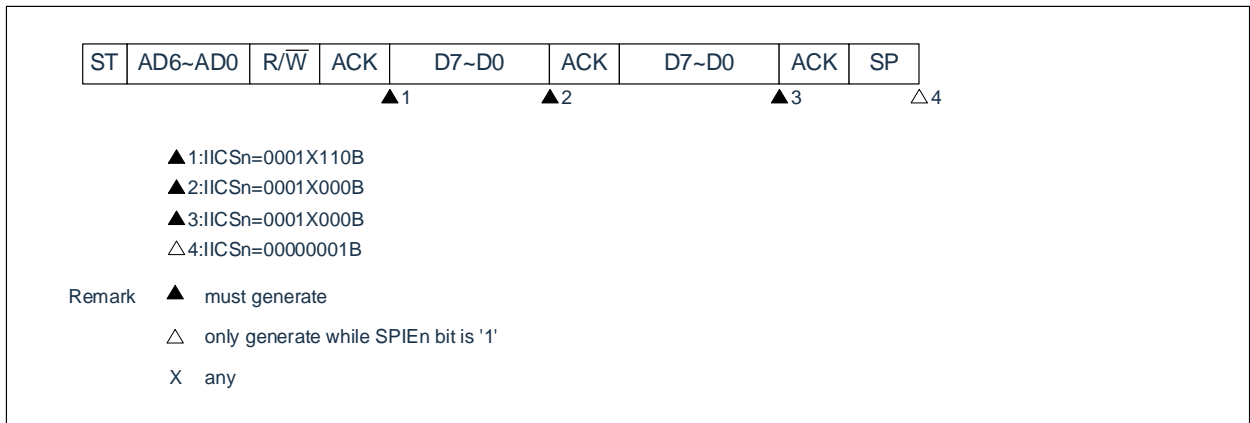


Remark: n=0

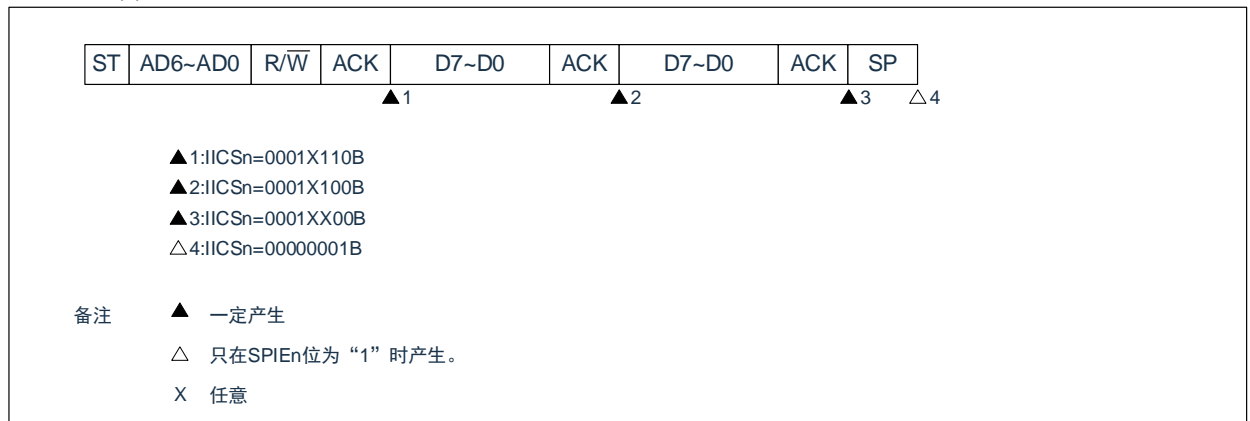
(2) Slave operation (when receiving a slave address)

(a) Start~Address~Data~Data~Stop

(i) When WTIMn=0



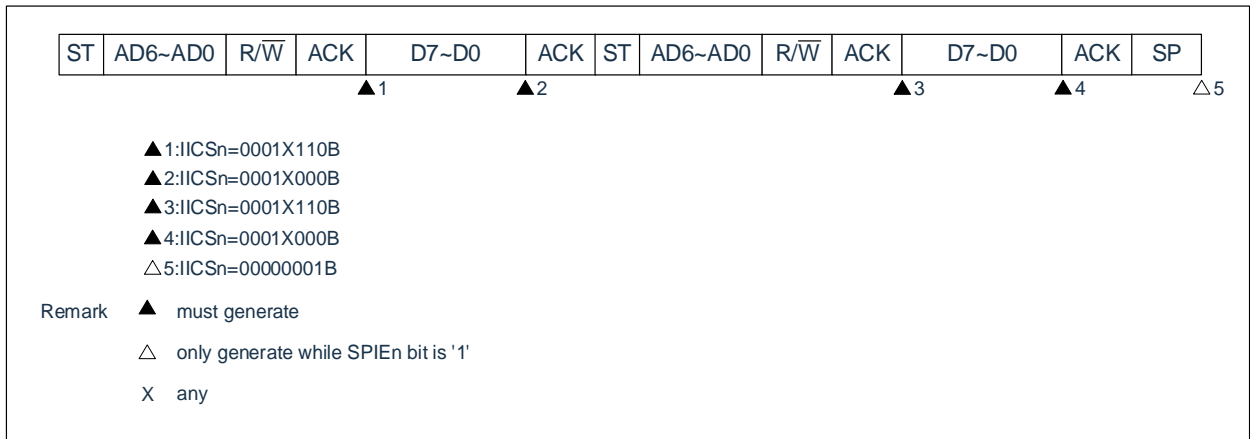
(ii) When WTIMn=1



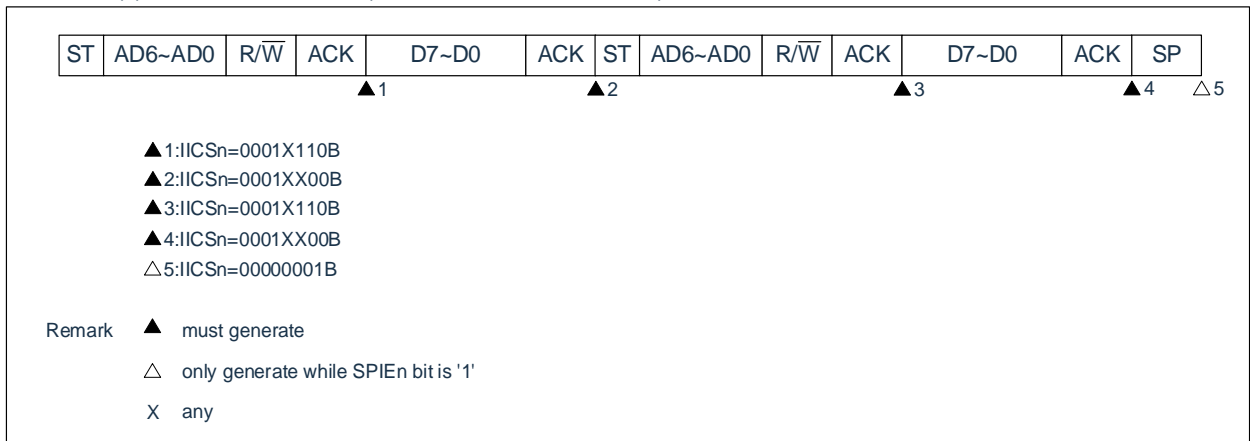
Remark: n=0

(b) Start~Address~Data~Start~Address~Data~Stop

(i) When WTIMn=0(same for SVAn after restart)



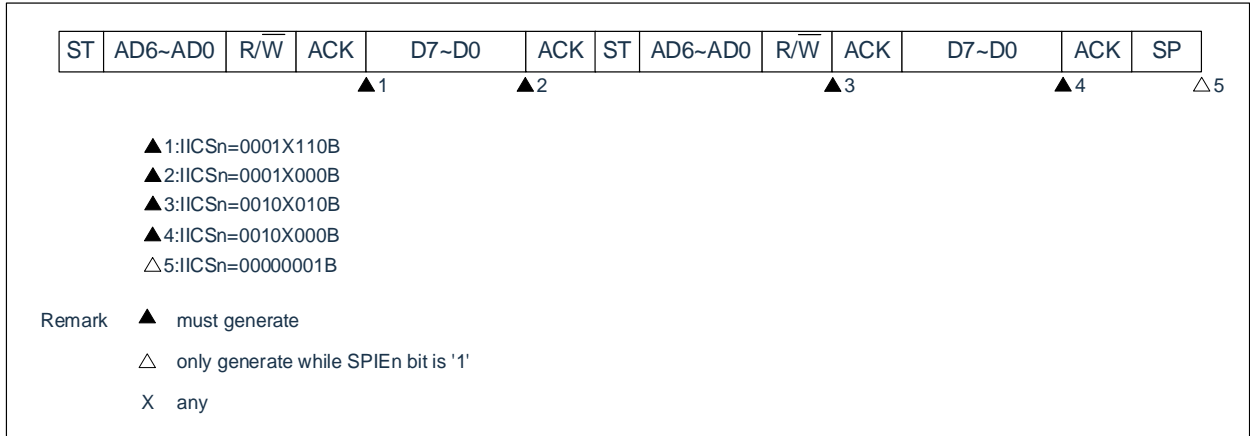
(ii) When WTIMn=1(same SVAn after restart)



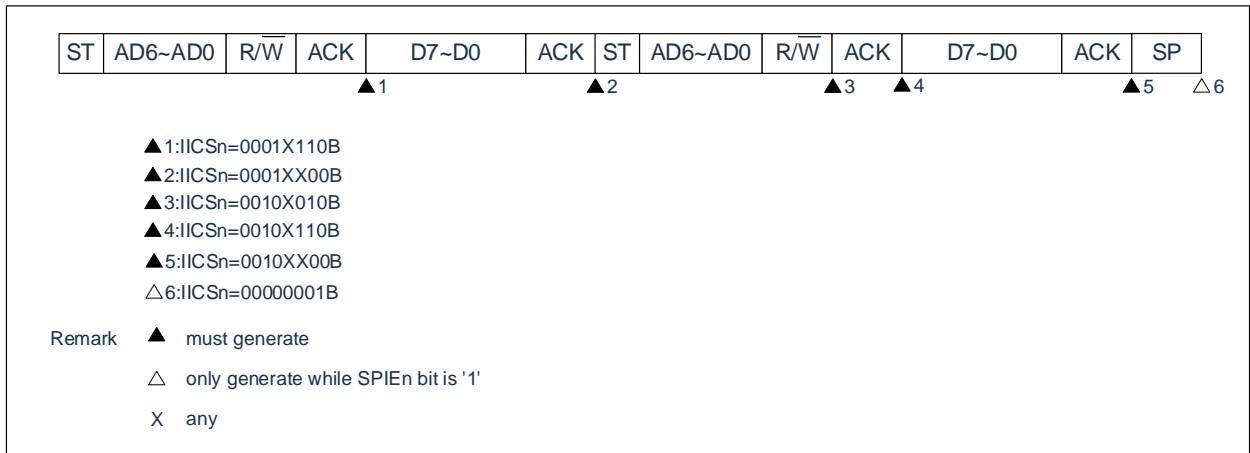
Remark: n=0

(c) Start~Address~Data~Start~Code~Data~Stop

(i) When WTIMn=0(the address is different after restart (extension code))



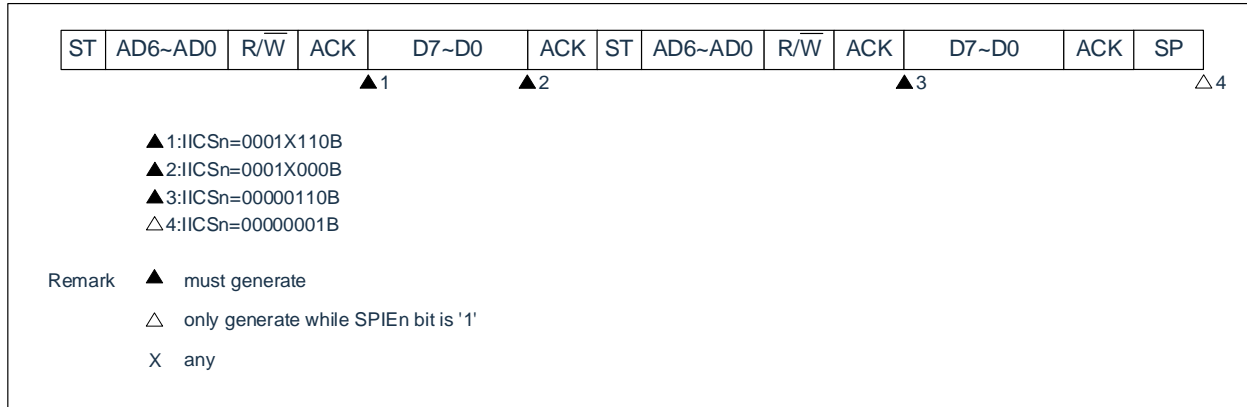
(ii) When WTIMn=1(the address is different after restart (extension code))



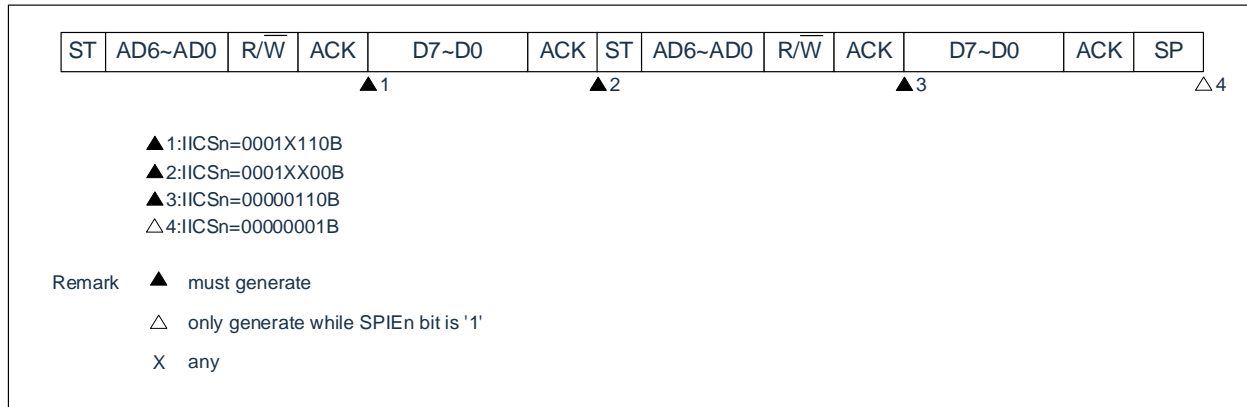
Remark: n=0

(d) Start~Address~Data~Start~Address~Data~Stop

(i) When WTIMn=0(the addresses are different after restart (non-extension code))



(ii) When WTIMn=1(the addresses are different after restart (non-extension code))



Remark: n=0

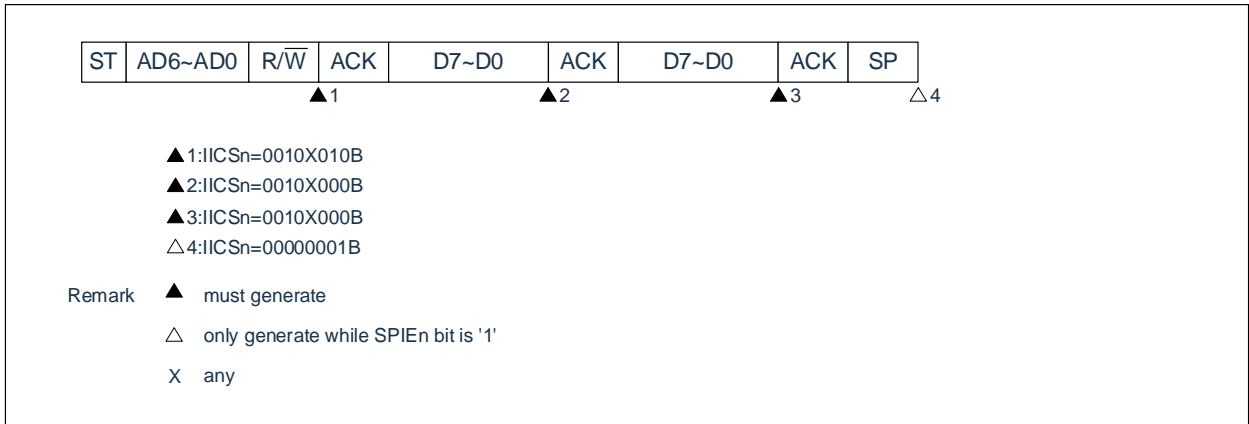


### (3) Slave operation(when receiving an extension code)

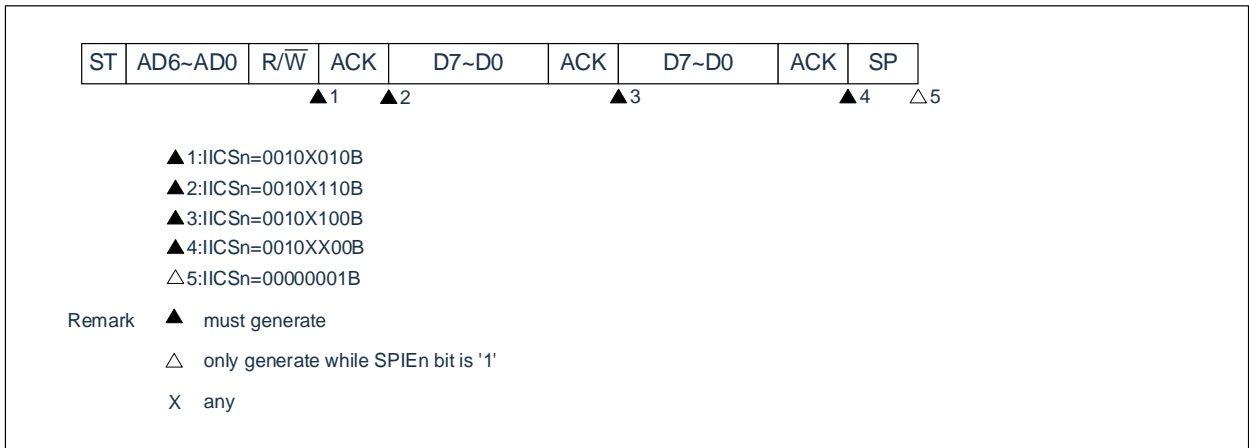
Always participate in the communication when receiving an extension code.

#### (a) Start~Code~Data~Data~Stop

##### (i) When WTIMn=0



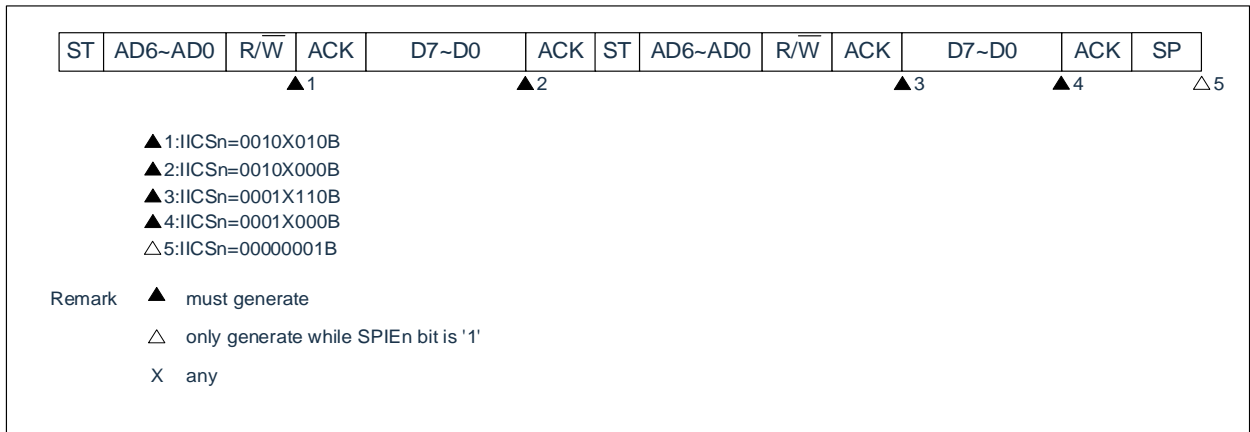
##### (ii) When WTIMn=1



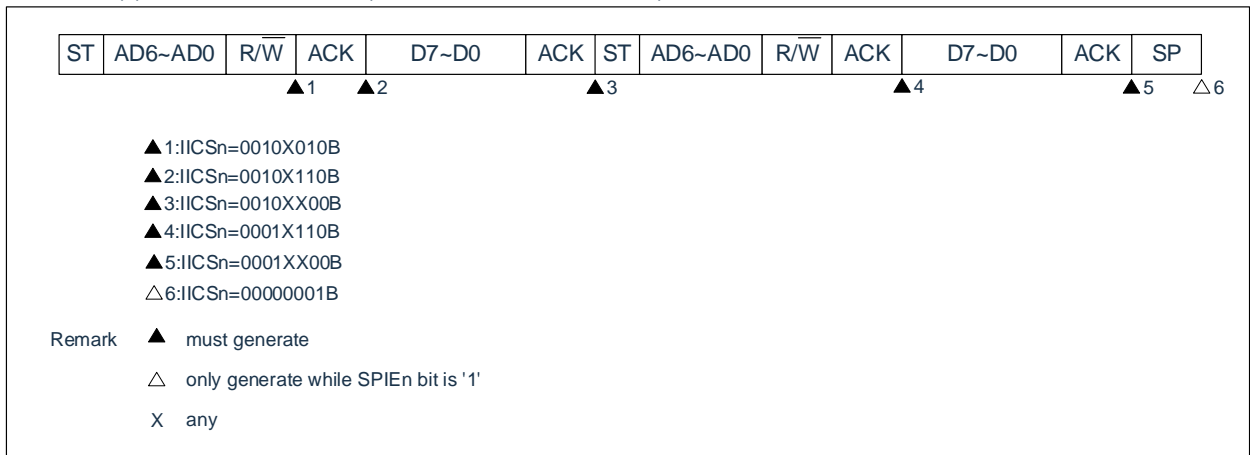
Remark: n=0

(b) Start~Code~Data~Start~Address~Data~Stop

(i) When WTIMn=0(same SVAn after restart)



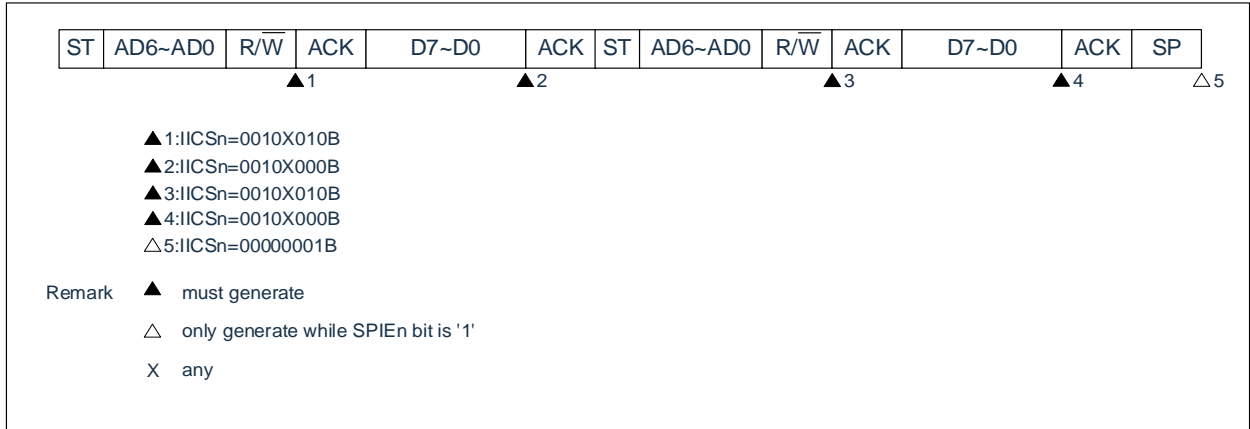
(ii) When WTIMn=1(same SVAn after restart)



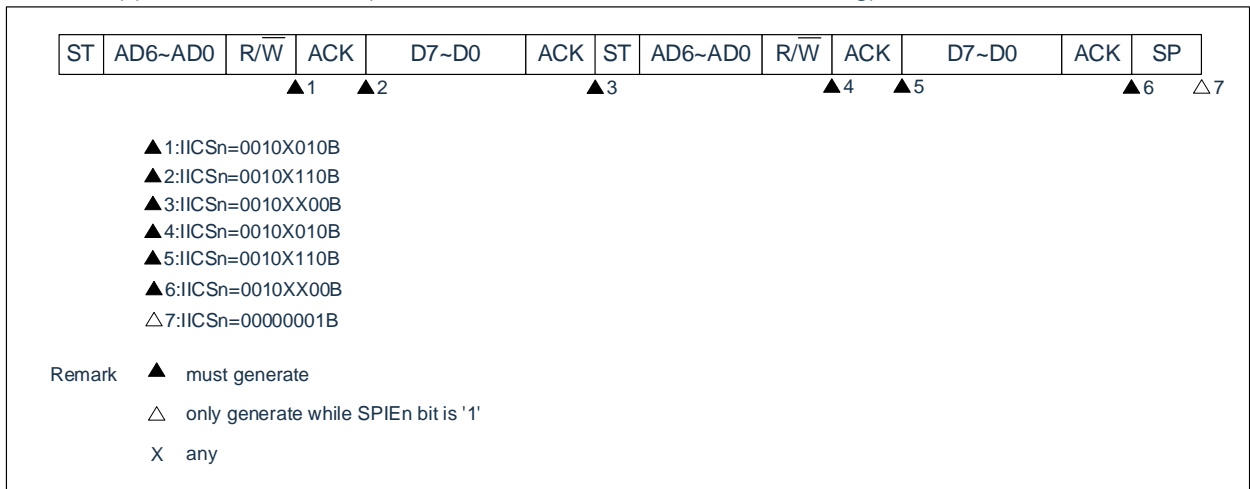
Remark: n=0

(c) Start~Code~Data~Start~Code~Data~Stop

(i) When WTIMn=0(receive the extension code after restarting)



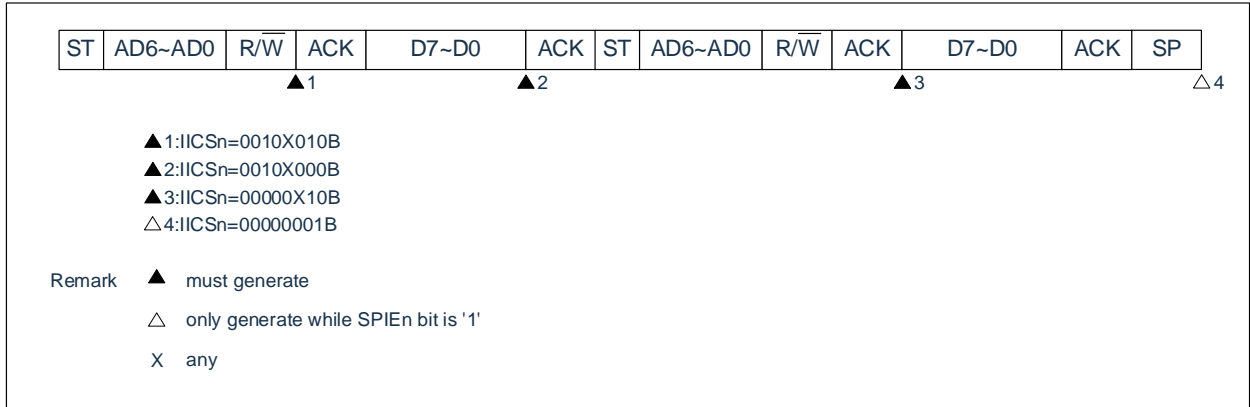
(ii) When WTIMn=1(receive the extension code after restarting)



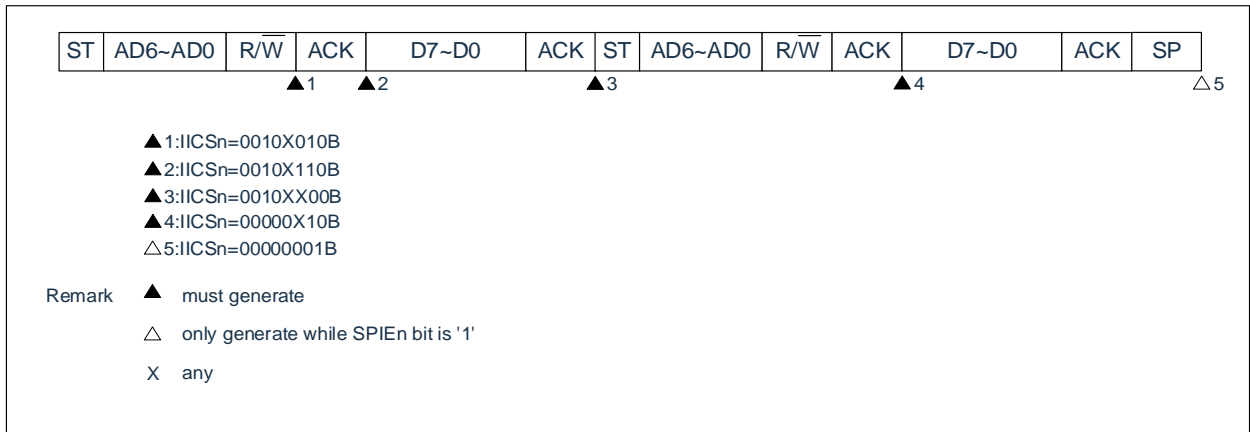
Remark: n=0

(d) Start~Code~Data~Start~Address~Data~Stop

(i) When WTIMn=0(the addresses are different after restart (non-extension code))



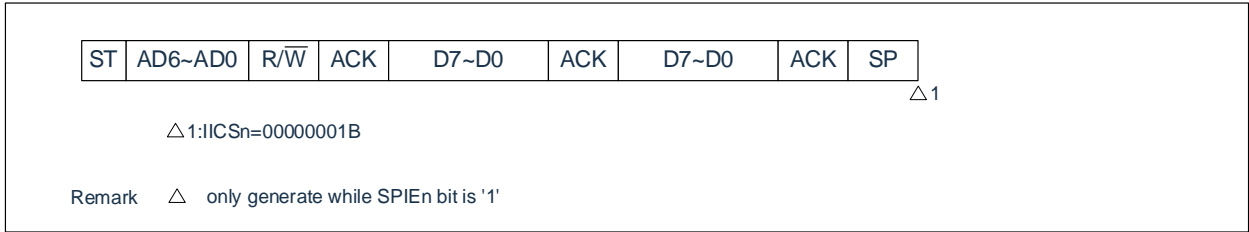
(ii) When WTIMn=1(the addresses are different after restart (non-extension code))



Remark: n=0

(4) Not participate in the operation of the communication

(a) Start~Code~Data~Data~Stop

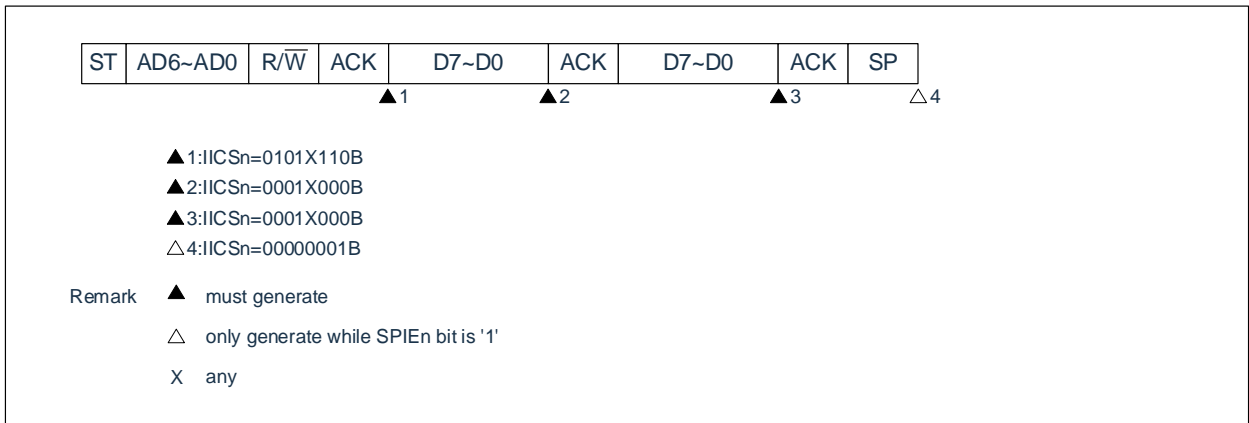


(5) Operation of the arbitration failure (operate as a slave after arbitration failure)

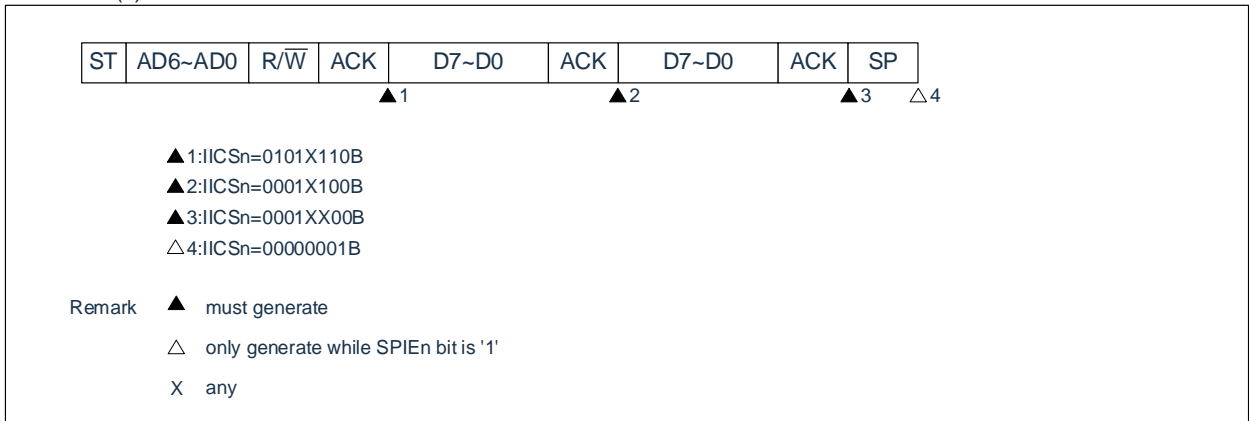
When used as a master device in a multi-master system, the MSTSn bit must be read each time an INTIICAn interrupt request signal is generated to confirm the arbitration result

(a) A condition in which arbitration fails during the sending of slave address data

(i) When WTIMn=0



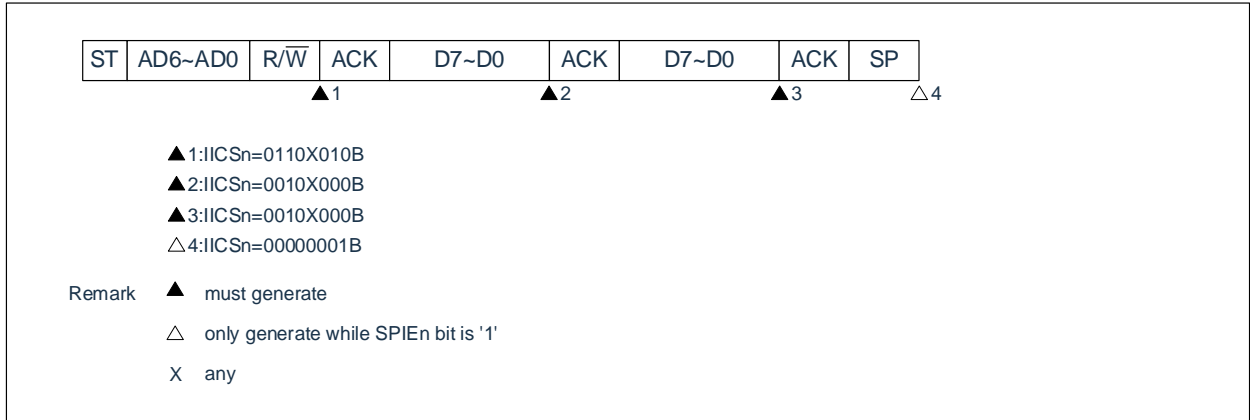
(ii) When WTIMn=1



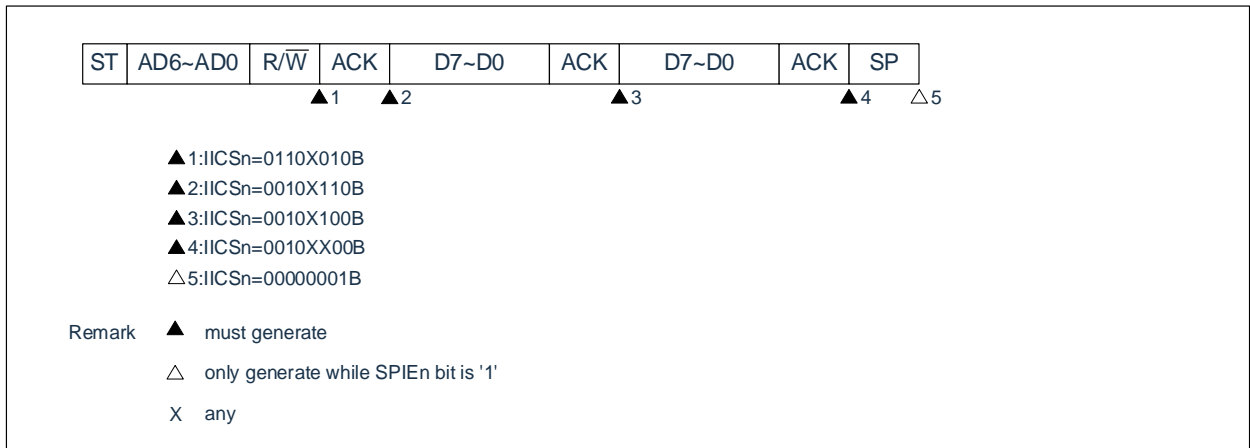
Remark: n=0

(b) A condition in which arbitration fails during the sending of an extension code

(i) When WTIMn=0



(ii) When WTIMn=1

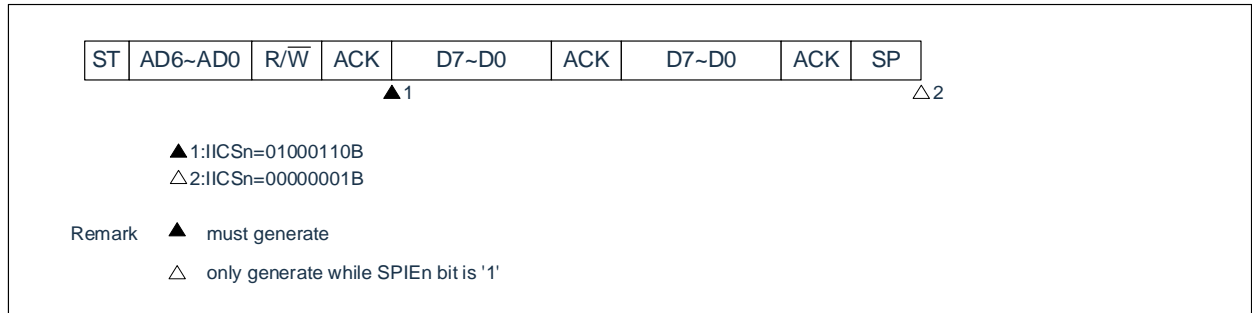


Remark: n=0

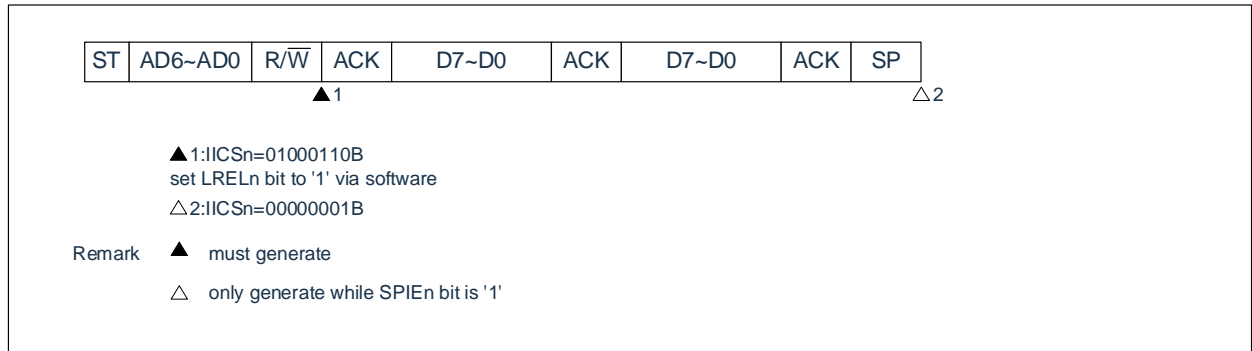
(6) Operation of arbitration failure (not participating in the communication after arbitration failure)

When used as a master device in a multi-master system, the MSTSn bit must be read each time an INTIICAn interrupt request signal is generated to confirm the arbitration result.

(a) When arbitration fails during the sending of slave address data (WTIMn=1)



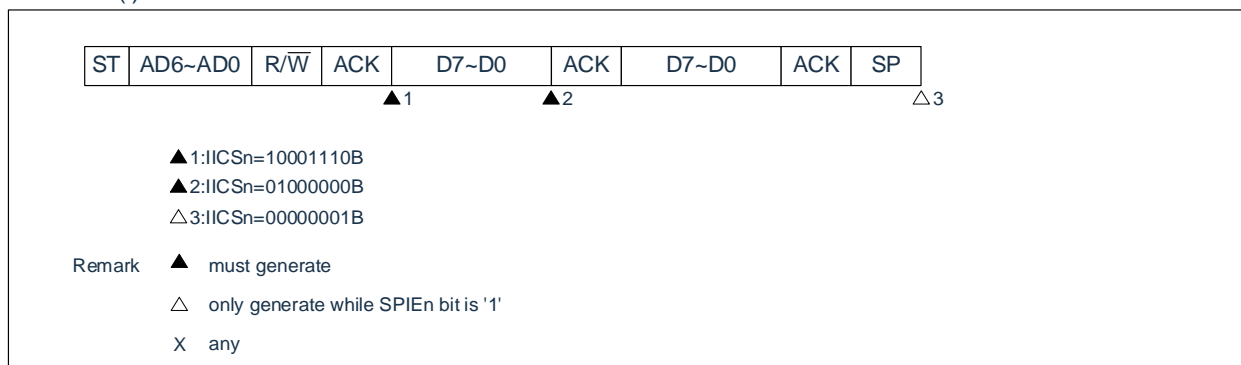
(b) When arbitration fails during the sending of an extension code



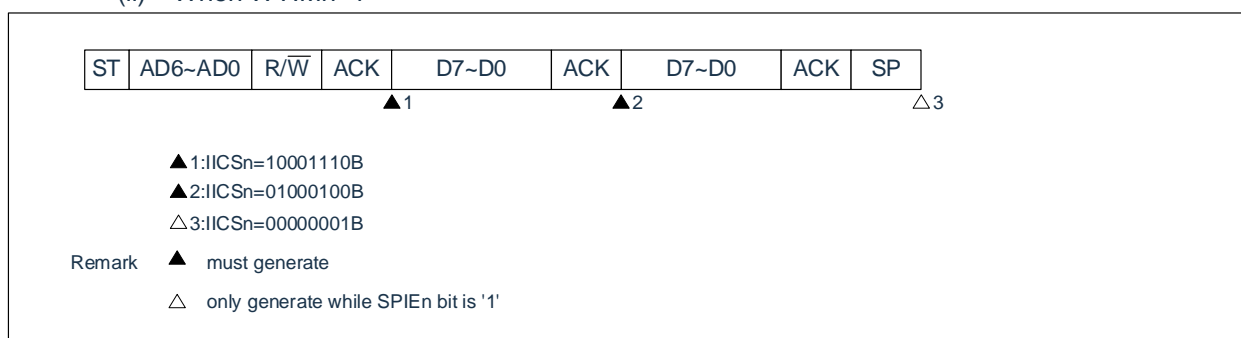
Remark: n=0

(c) When arbitration fails while transferring data

(i) When WTIMn=0



(ii) When WTIMn=1

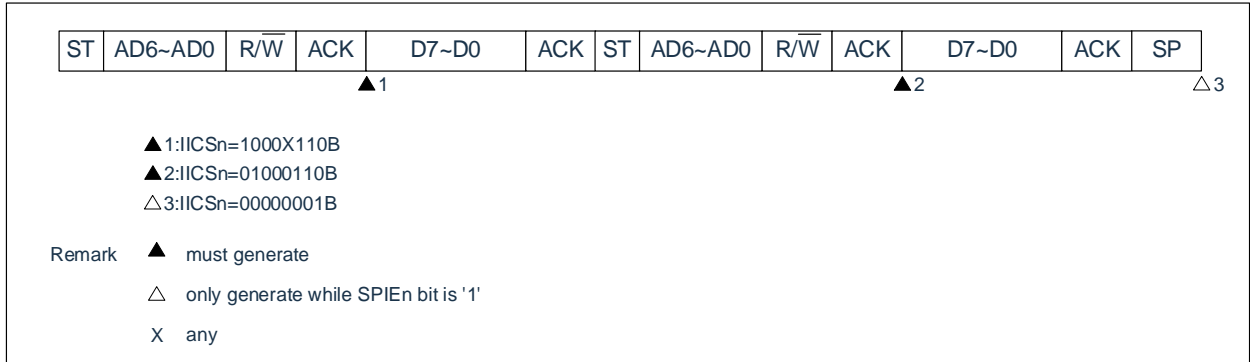


Remark: n=0

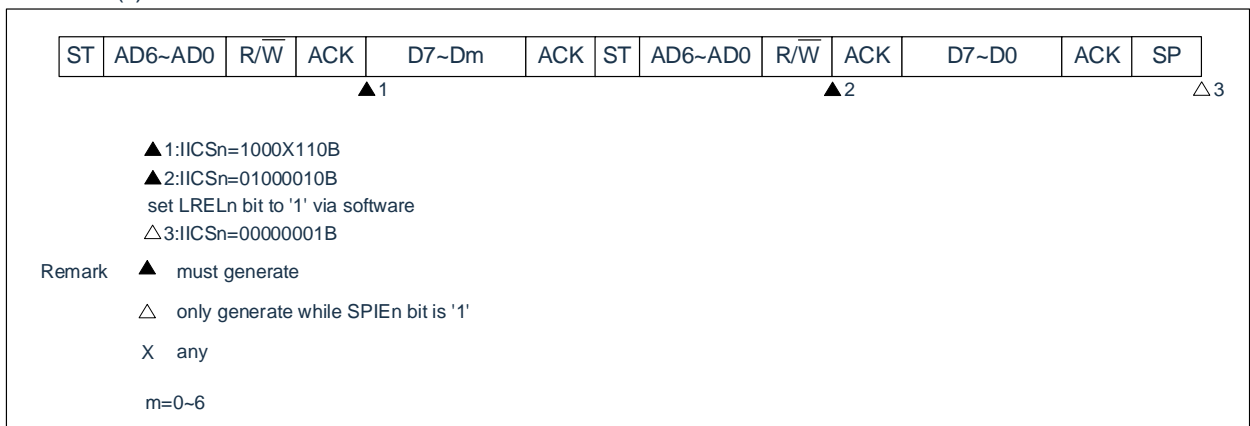


(d) When arbitration fails due to restart conditions when transferring data

(i) Non-extended codes (for example, SVAn is different)



(ii) Extension code



Remark: n=0

(e) When arbitration fails due to a stop condition when transferring data



▲ 1: ICS<sub>n</sub>=10000110B

$\Delta 2: ICS_n = 01000001B$

Remark ▲ must generate

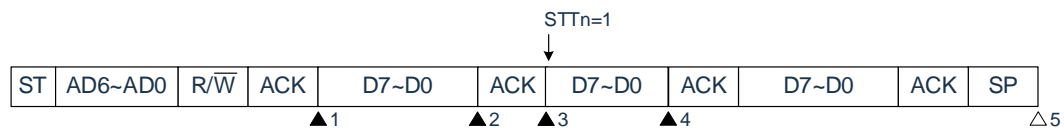
- △ only generate while SPIEn bit is '1'

$m=0\sim 6$

Remark:  $n=0$

(f) When arbitration fails because the data is low when you want to generate a restart condition

(i) When  $WTIM_n=0$



▲ 1:ICS<sub>n</sub>=1000X110B

▲2:IICS<sub>n</sub>=1000X000B(set WTIM<sub>n</sub> bit to“1”)

▲ 3: ICSn=1000X100B(set WTIMn bit to“0”)

▲4:IICSn=01000000B

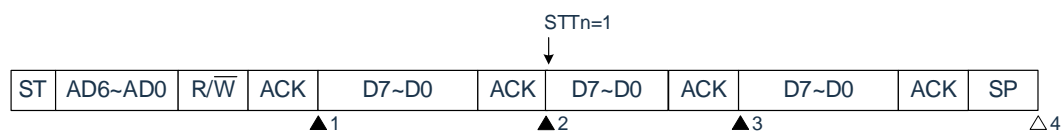
$\Delta 5: IICSn=00000001B$

Remark ▲ must generate

- △ only generate while SPIEn bit is '1'

X any

(ii) When  $WTIMn=1$



▲ 1:ICS<sub>n</sub>=1000X110B

▲2:IICS<sub>n</sub>=1000X100B(set STT<sub>n</sub> bit to "1" )

▲3:ICSn=01000100B

△4:IICS<sub>n</sub>=00000001B

Remark ▲ must generate

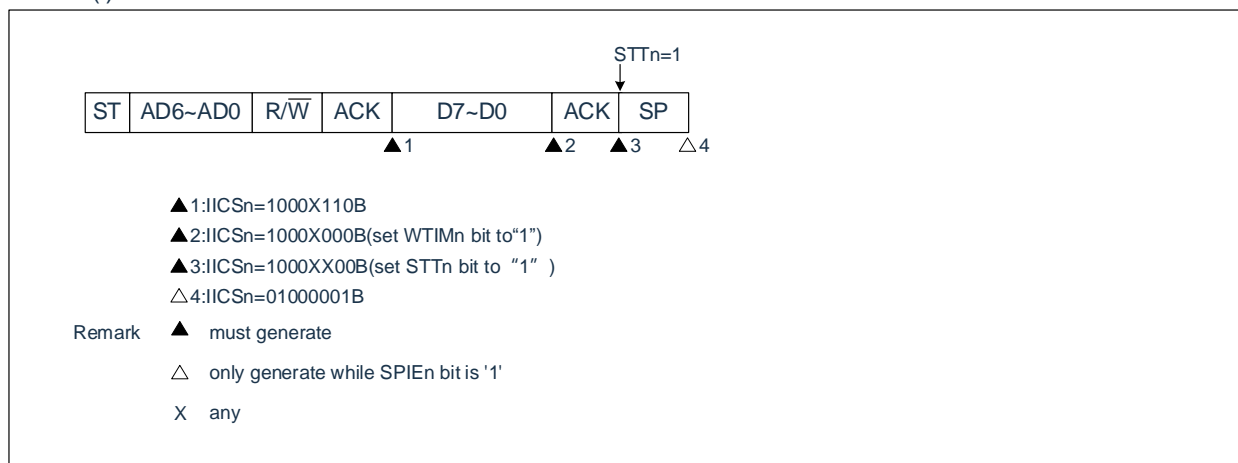
△ only generate while SPIEn bit is '1'

X any

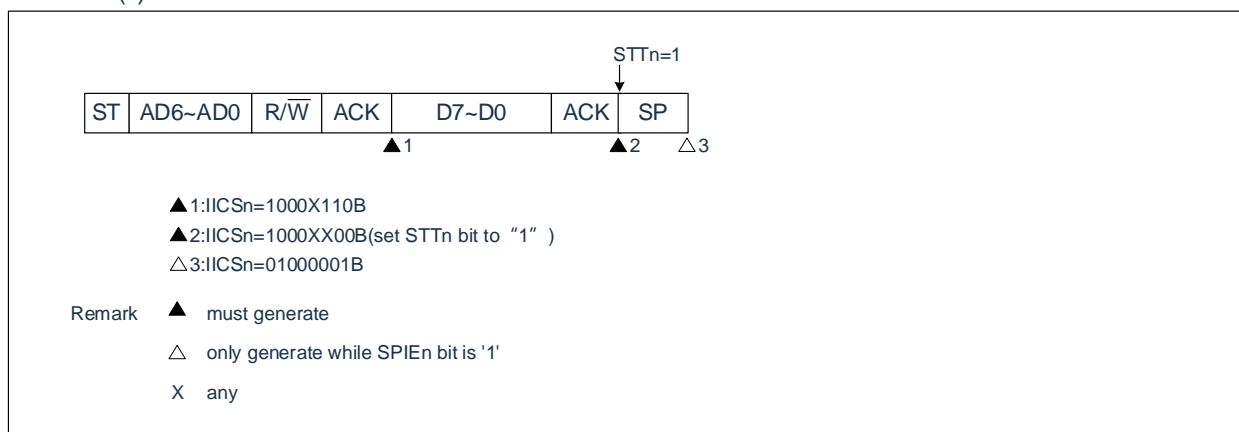
Remark:  $n=0$

(g) Arbitration failure due to a stop condition when trying to generate a restart condition

(i) When WTIMn=0



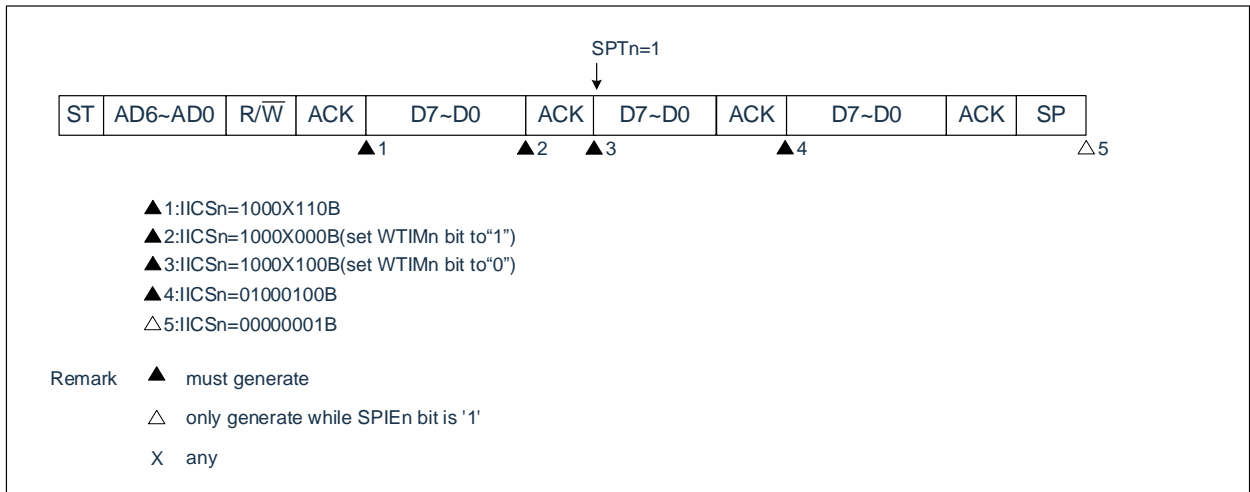
(ii) When WTIMn=1



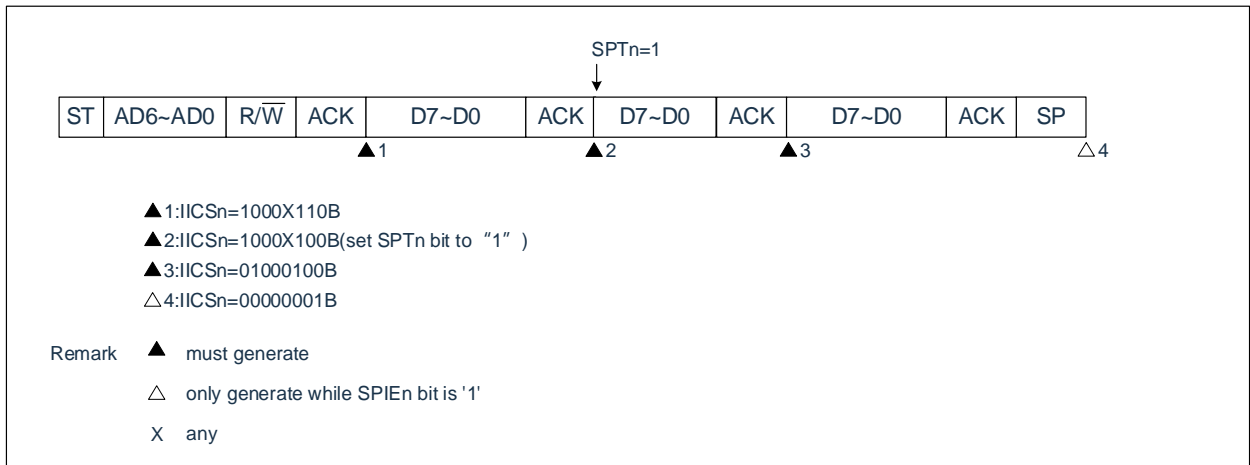
Remark: n=0

### (h) Arbitration failure due to low level data when trying to generate a stop condition

#### (i) When WTIMn=0



#### (ii) When WTIMn=1



Remark: n=0

## 18.6 Timing diagram

In the I<sup>2</sup>C bus mode, the master device selects a slave device from among multiple slave devices for communication by giving the serial bus output address. The master device sends the TRCn bit (bit 3 of the IICA status register n (IICSn)) indicating the direction of data transfer after the slave device address to start serial communication with the slave device. The timing diagram of data communication is shown in Figure18-31 and Figure18-32.

The shift of the IICA shift register n (IICAn) is carried out synchronously with the falling edge of the serial clock (SCLAn), and the transmit data is transmitted to the SO latch, in MSB Prioritize outputting data from the SDAAn pin.

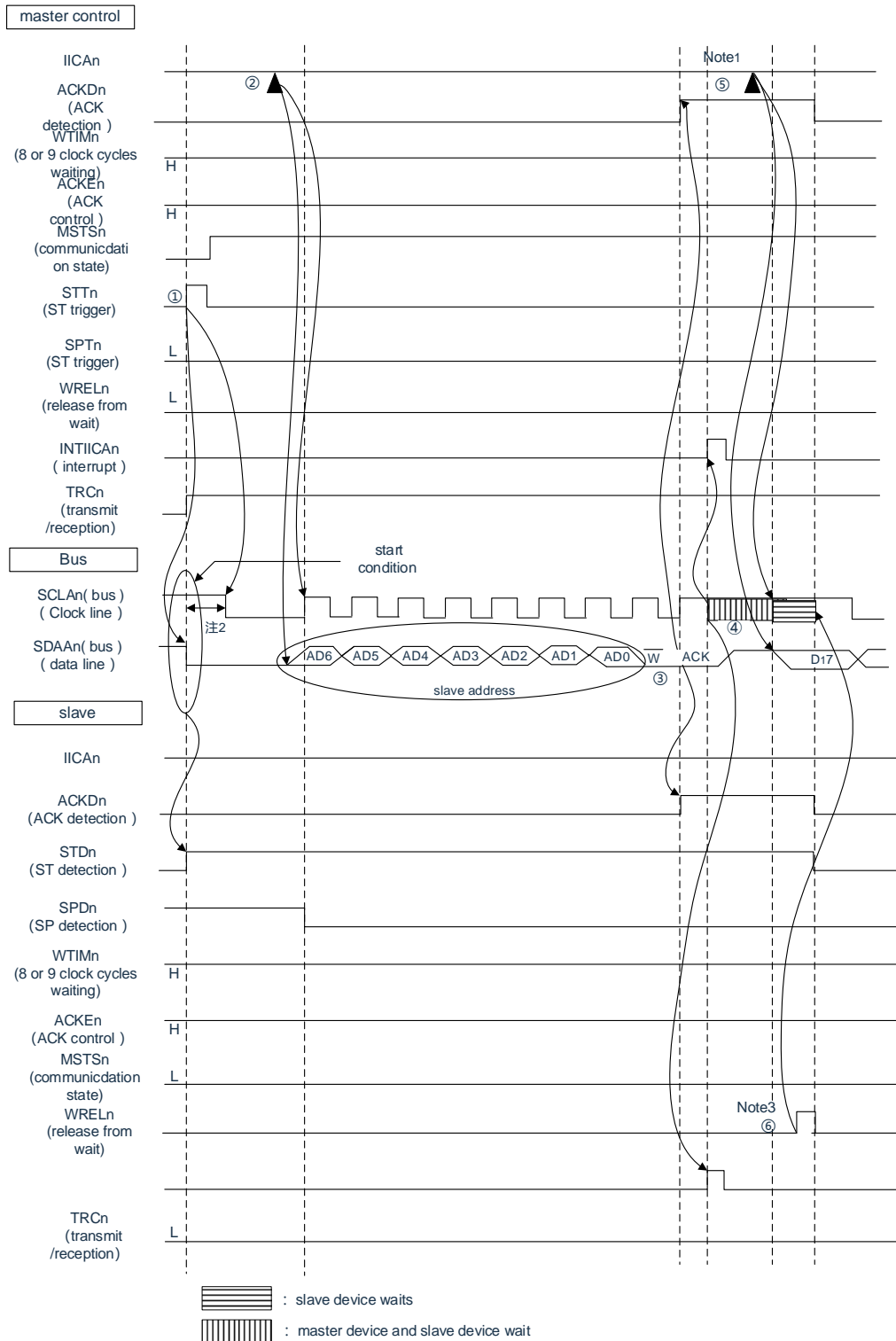
Take the data from the SDAAn pin input to the IICAn on the rising edge of the SCLAn.

Remark: n=0

Figure18-31: example of a slave device→a master device

(Master: Select 9 clocks to wait, Slave: Choose 9 clocks to wait)(1/4)

### (1) Start Condition ~ Address ~ Data



Note 1: To remove the wait during the master send, the IICAn must be written to the data instead of the set the WRELn bit.

Note 2: The time from the SDAAn pin signal drop to the SCLAn pin signal drop is at least 4.0μs when set to standard mode and at least 0.6μs when set to fast mode.

Note 3: To release the wait during the slave receive, the IICAn must be set to "FFH" or set the WRELn bit.

Figure18-31 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 (MSTSn=1) and after the hold time elapses the bus clock line drops (SCLAn=0), ending the communication preparation.
- ② If the master writes address +W (transmit) to IICA shift register n (IICAn), the slave address is sent.
- ③ On the slave, if the receiving address and the local station address (the value of the SVAn) are the same note, an ACK is sent to the master through hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).
- ④ The master generates an interrupt on the falling edge of the 9th clock (INTIICAn: address send end interrupt). Slaves with the same address enter a waiting state (SCLAn=0) and an interrupt (INTIICAn: address matching interrupt) note.
- ⑤ The master writes and transmits data to the IICAn registers, relieving the master of waiting.
- ⑥ If the slave releases the wait (WRELn=1), the master begins to transmit data to the slave.

Note: If the sent address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master, and does not generate an INTIICAn interrupt (address matching interrupt) or enter a waiting state. However, the master generates ANTIICAn interrupts (address send end interrupts) for both ACK and NAK.

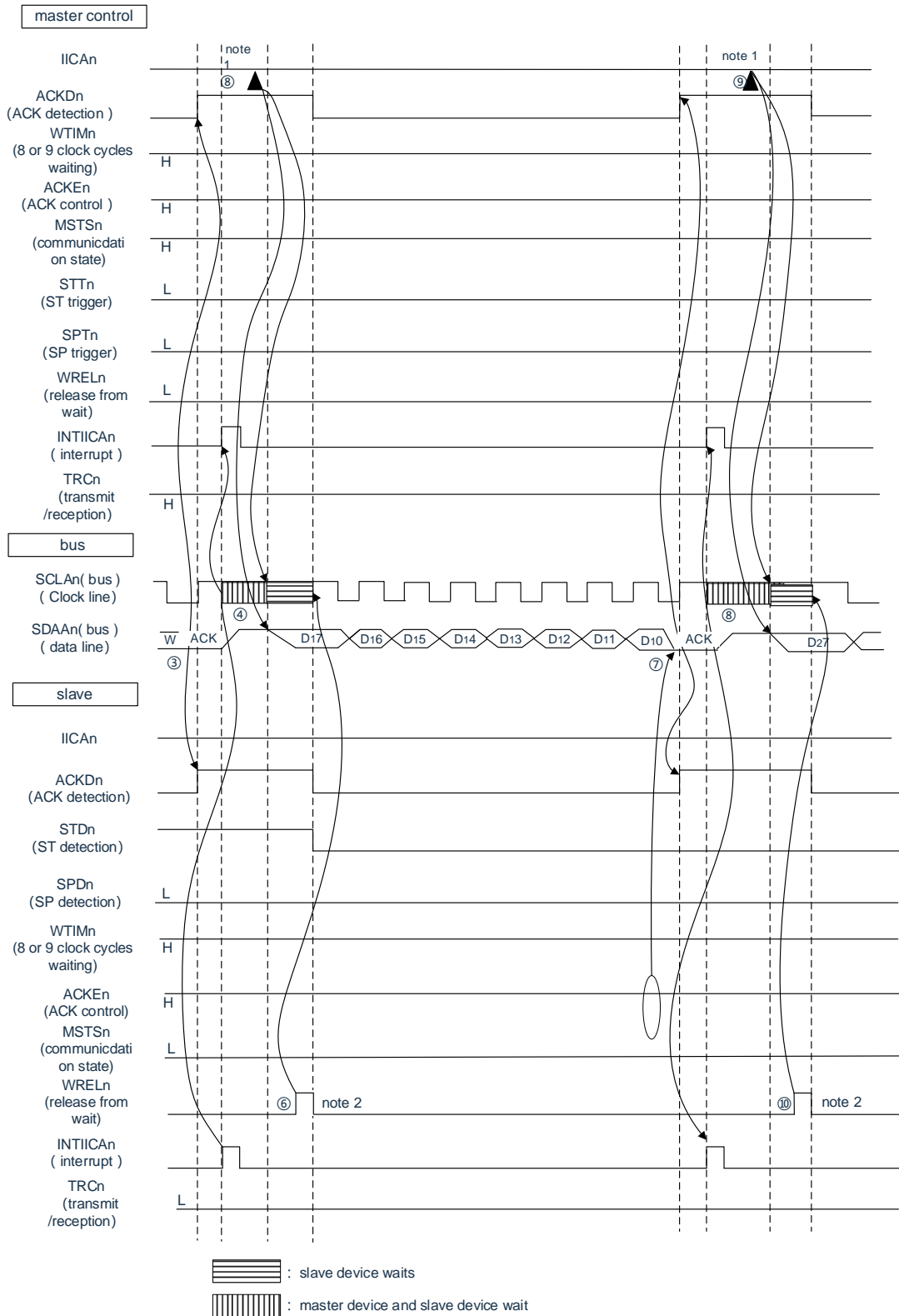
Remark:

1. Figure18-31: ① to ⑮ shows a series of operational steps for data communication via the I<sup>2</sup>C bus  
Figure18-31: "(1) Start Condition ~ Address ~ Data" illustrates steps ①~⑥.  
Figure18-31: "(2) Address ~ Data ~ Data" illustrates steps ③~⑩.  
Figure18-31: "(3) Data ~ Data ~ Stop Condition" illustrates steps ⑦~⑮.
2. n=0

Figure18-31: Communication example of a master device → slave device

(Master: Select 9 clocks to wait, Slave: Choose 9 clocks to wait) (2/4)

(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 reception, the IICAn must be set to "FFH" or the WRELn bit must be set.



Figure18-31 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) <sup>Note</sup>.

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:

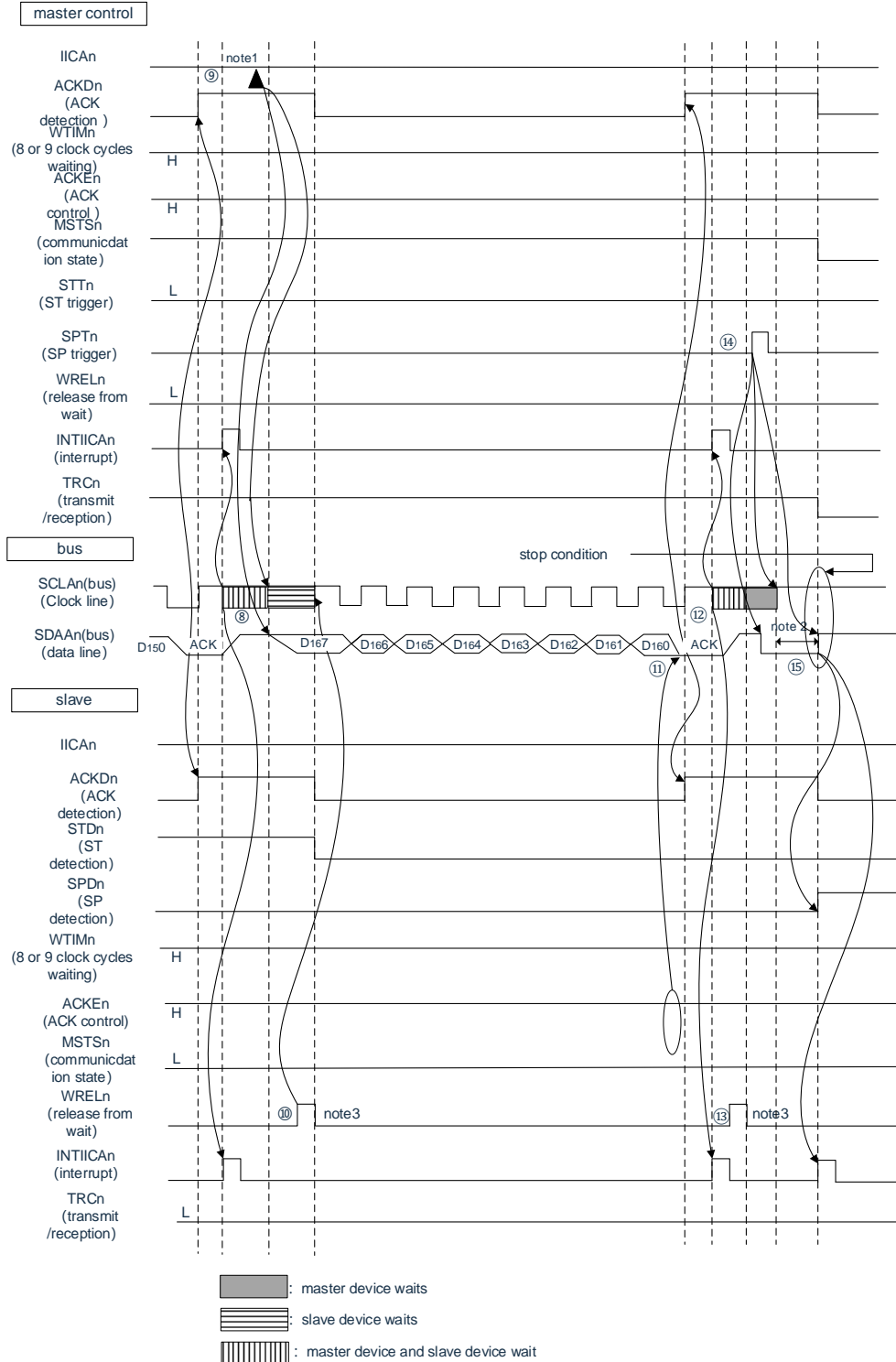
1. Figure18-31: ① to ⑮ shows a series of operational steps for data communication via the I<sup>2</sup>C bus  
Figure18-31: "(1) Start Condition ~ Address ~ Data" illustrates steps ①~⑥.  
Figure18-31: "(2) Address ~ Data ~ Data" illustrates steps ③~ ⑩.  
Figure18-31: "(3) Data ~ Data ~ Stop Condition" illustrates steps ⑦ ~ ⑮.

2. n=0

Figure18-31: Communication example of a master device → slave device

(Master: Select 9 clocks to wait, Slave: Choose 9 clocks to wait)(3/4)

### (3) Data ~ data ~ stop condition



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: After the stop condition is issued, the time from the SCLAn pin signal to generate the stop condition is at least 4.0μs when set to standard mode and at least 0.6μs when set to fast mode.

Note 3: To release the slave from waiting during reception, the IICAn must be set to "FFH" or the WRELn bit must be set.

Figure18-31 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).

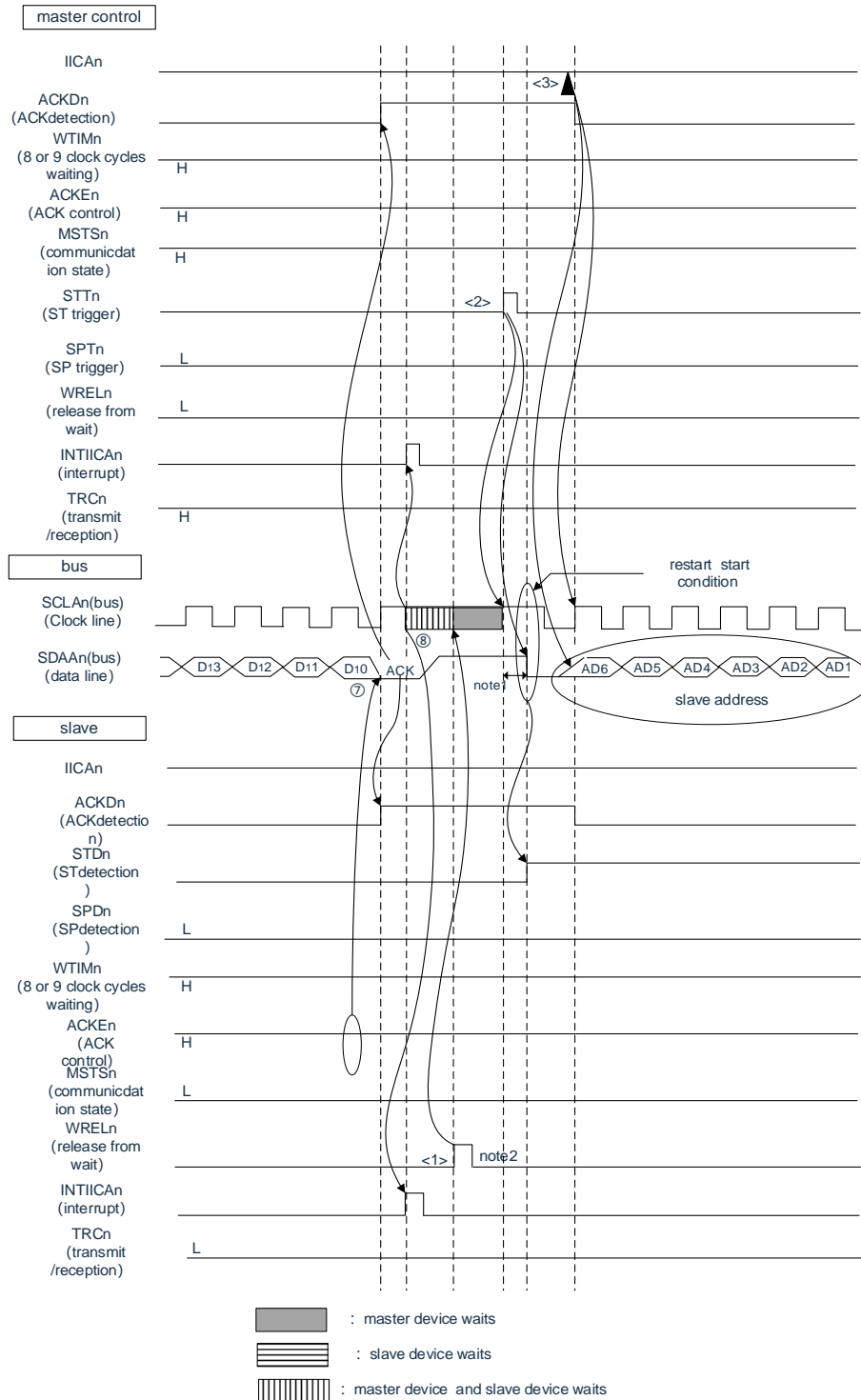
Remark:

1. Figure18-31: ① to ⑮ shows a series of operational steps for data communication via the I<sup>2</sup>C bus  
Figure18-31: "(1) Start Condition ~ Address ~ Data" illustrates steps ①~⑥.  
Figure18-31: "(2) Address ~ Data ~ Data" illustrates steps ③~ ⑩.  
Figure18-31: "(3) Data ~ Data ~ Stop Condition" illustrates steps ⑦ ~ ⑮.
2. n=0

Figure18-31: Communication example of a master device → slave device

(Master: Select 9 clocks to wait, Slave: Choose 9 clocks to wait)(4/4)

#### (4) Data ~ restart condition ~ address



Note 1: After the release of the restart condition, the time from which the SCLAn pin signal rises to generate the start condition is at least 4.7μs when set to standard mode and at least 0.6μs when set to fast mode.

Note 2: To release the slave from waiting during reception, the IICAn must be set to "FFH" or the WRELn bit must be set.

The operation of "(4) Data ~ Restart condition ~ Address" in Figure 17-31 is explained as follows. After executing steps ⑦ and ⑧, execute <1> to <3>, and return to the data sending step in 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> if the master triggers the start condition again (STTn=1), the bus clock line rises (SCLAn=1) and the bus data line drops (SDAAn=0) after the preparation time for the new start condition), generate start conditions (change SDAAn from "1" to "0" by SCLAn=1). Then, if a start condition is detected, the bus clock line drops (SCLAn=0) just after the hold time has elapsed, ending the communication preparation.

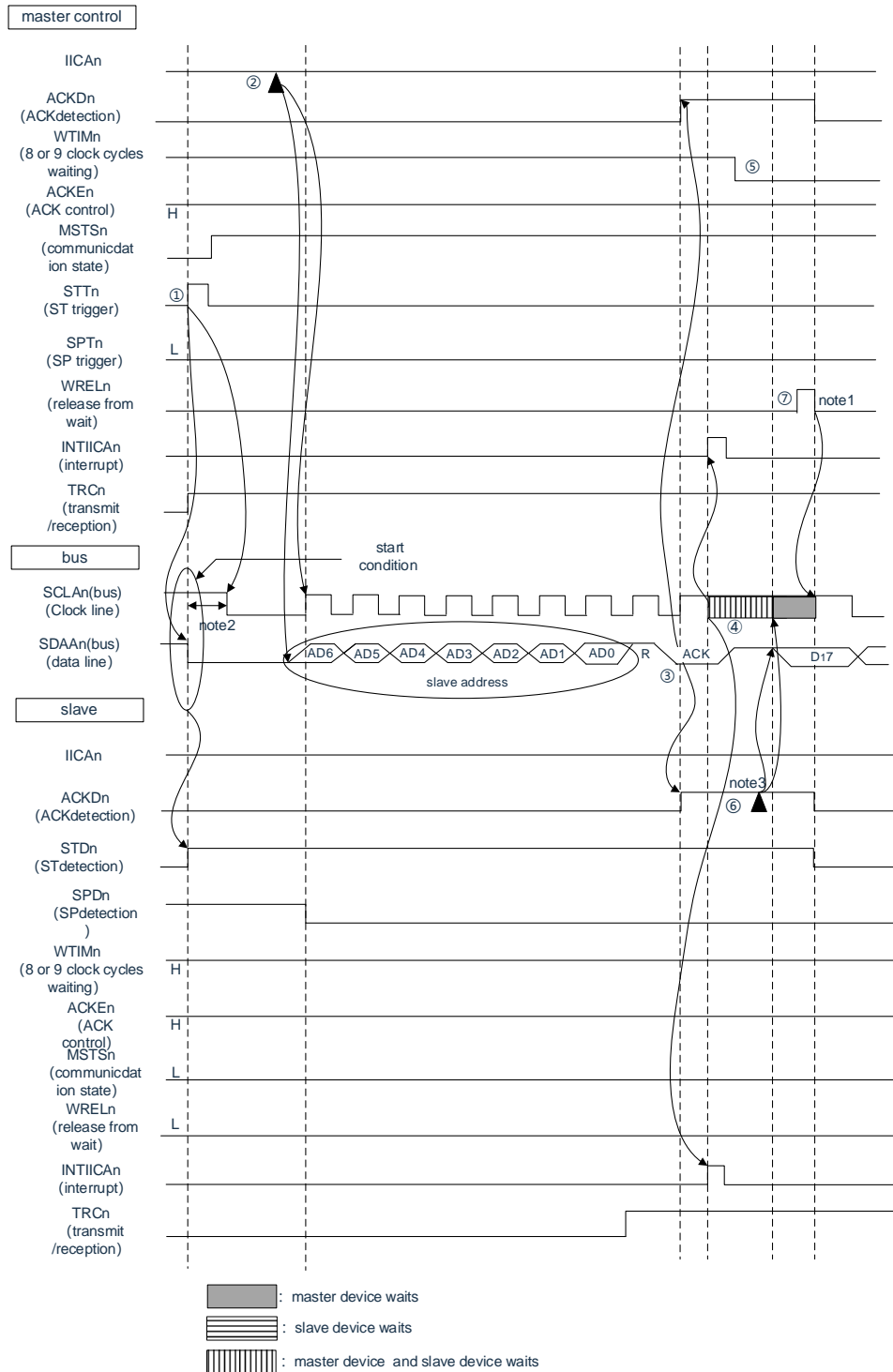
<3> if the master writes an address +R/W (transmit) to the IICA shift register n (IICAn), the slave address is sent.

Remark: n=0

Figure18-32: Communication example of a slave device→master device

(Master: Select 8 clocks to wait, Slave: Choose 9 clocks to wait)(1/3)

(1) Start Condition ~ Address ~ Data



Note 1: To release the master from waiting during transmit, you must write data to the IICAn instead of setting the WRELn bit.

Note 2: The time from the SDAAn pin signal drop to the SCLAn pin signal drop is at least 4.0us when set to standard mode and at least 0.6 μs when set to fast mode.

Note 3: To release the slave from waiting during reception, the IICAn must be set to "FFH" or the WRELn bit must be set.

Figure18-32 The descriptions of ① to ⑦ of "(1) Start Condition ~ Address ~ Data" are 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 <sup>Note</sup>, 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) <sup>Note</sup>.

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.

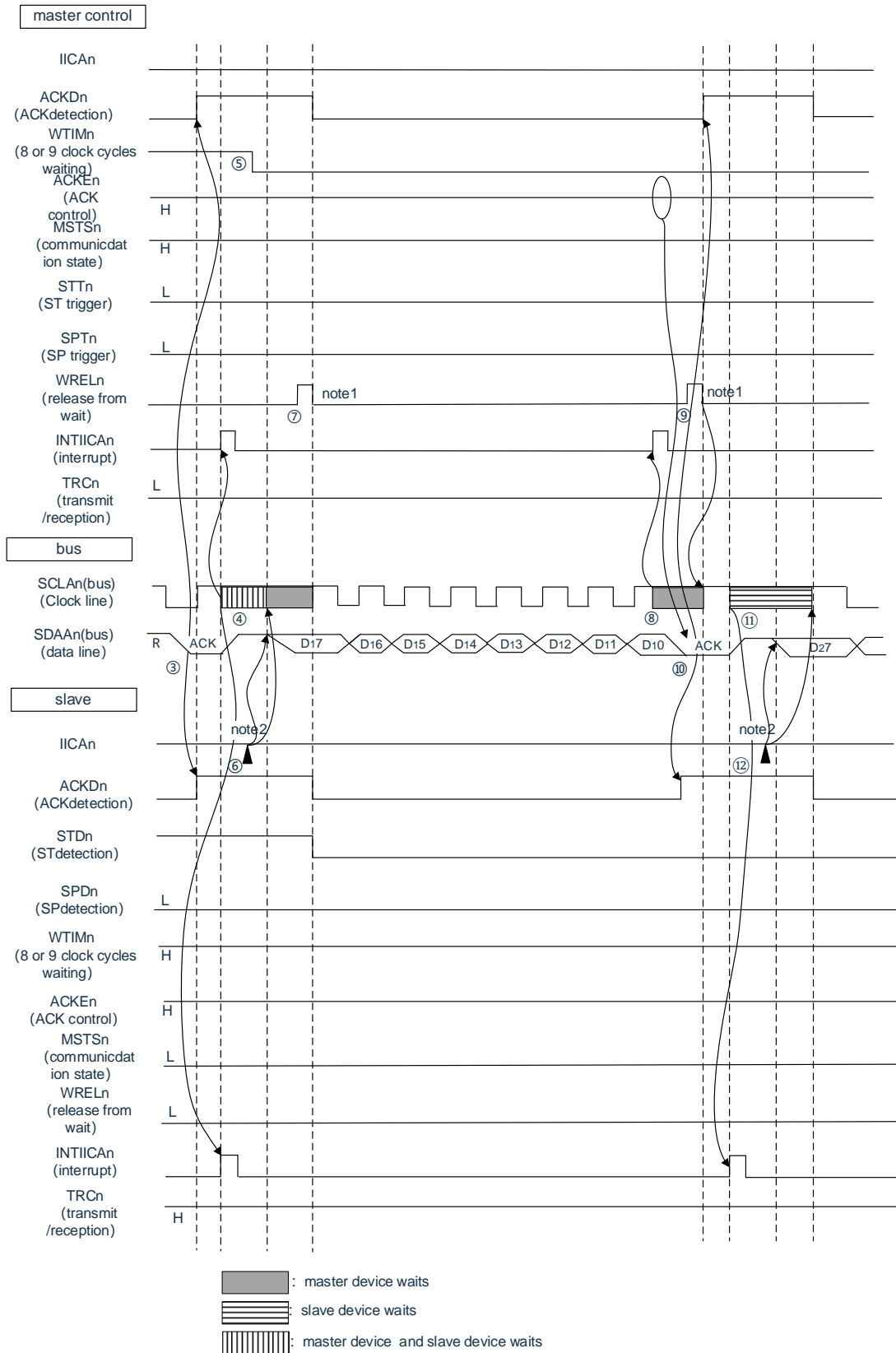
Remark:

1. Figure18-32: ①~⑱ shows a series of operation steps for data communication via the I<sup>2</sup>C bus.  
Figure18-32: "(1) Start Condition ~ Address ~ Data" illustrates steps ① to ⑦.  
Figure18-32: "(2) Address ~ Data ~ Data" illustrates steps 3 to ⑫.  
Figure18-32: "(3) Data ~ data ~ stop condition" illustrates steps ⑧ to ⑱.
2. n=0

Figure18-32: Communication example of a slave device→master device

(Master: Select 8 clocks to wait, Slave: Choose 9 clocks to wait)(2/3)

(2) Address ~ data ~ data



Note 1: To release the master from waiting during reception, the IICAn must be set to "FFH" or the WRELn bit must be set.



Note 2: To release the slave from waiting during transmission, you must write data to the IICAn instead of setting the WRELn bit.

Figure18-32 The description of ③ to ⑫ of "(2) Address ~ Data ~ Data" is as follows.

③ On the slave, if the receiving address and the local station address (the value of the SVAn) are the same <sup>Note</sup>, 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) <sup>Note</sup>.

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 releases 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. Figure18-32: ①~⑲ shows a series of operation steps for data communication via the I<sup>2</sup>C bus.

Figure18-32: "(1) Start Condition ~ Address ~ Data" illustrates steps ① to ⑦.

Figure18-32: "(2) Address ~ Data ~ Data" illustrates steps 3 to ⑫.

Figure18-32: "(3) Data ~ data ~ stop condition" illustrates steps ⑧ to ⑲.

2. n=0

(Master: Select 8  $\rightarrow$  9 clocks to wait, Slave: Select 9 clocks to wait)(3/3)

master control



Note 1: To release the wait, the IICAn must be set to "FFH" or the WRELn bit must be set.

Note 2: After the release of the stop condition, the time from the SCLAn pin signal to generate the stop condition is at least 4.0us when set to standard mode and at least 0.6us when set to fast mode.

Note 3: To release the slave from waiting during transmit, the data must be written to the IICAn instead of the WRELn bit.

Note 4: If the wait is released by setting the WRELn bit during the slave's transmit, the TRCn bit is cleared.

Figure18-32 The description of ⑧~⑲ of "(3) Data~Data~Stop Condition" is as follows.

- ⑧The 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
- ⑨The master reads the received data and dismisses the wait ( $WRELn=1$ ).
- ⑩The slave detects ACK ( $ACKDn=1$ ) on the rising edge of the 9th clock.
- ⑪The slave enters a waiting state on the falling edge of the 9th clock ( $SCLAn= 0$ ) and generates an interrupt (INTIICAn: transmit end interrupt).
- ⑫If 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.
- ⑬The master generates an interrupt (INTIICAn: transmit end interrupt) on the falling edge of the 8th clock and enters a waiting state ( $SCLAn=0$ ). Because ACK control ( $ACKEn=1$ ) occurs, the bus data line at this stage becomes low ( $SDAAn=0$ )
- ⑭The master sets the NACK Acknowledge ( $ACKEn=0$ ) and changes the wait sequence to the 9th clock ( $WTIMn=1$ ). If the master releases the wait ( $WRELn=1$ ), the slave detects THEACK ( $ACKDn=0$ ) on the rising edge of the 9th clock.
- ⑮Both 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).
- ⑯If 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 ( $SCLAn=1$ ).
- ⑰The 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 ( $SCLAn=1$ ).
- ⑱If 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 19 IrDA

The IrDA implements the transmission and reception of IrDA communication waveforms in accordance with IrDA (InfraredDataAssociation) 1.0 by cooperating with SCI.

## 19.1 Function of IrDA

If the IrDA function is enabled by the IRE bit of the IRCR register, the TxD1 signal and RxD1 signal of the Universal Serial Communication Unit 1 (UART1) can encode or decode the waveform conforming to the IrDA1.0 protocol (IrTxD/IrRxD pins), and then by connecting the infrared transmitter/receiver to the transmitter or receiver to support IrDA1.0 protocol infrared transmission and reception.

In systems that support the IrDA 1.0 protocol, the transmission rate can be changed by software as needed after starting communication at 9600bps.

When selecting a high-speed internal oscillator (FIH=24, 12, 6, 3MHz), 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 Figure19-1.

Figure19-1: Schematic block diagram of IrDA's collaboration with SCI

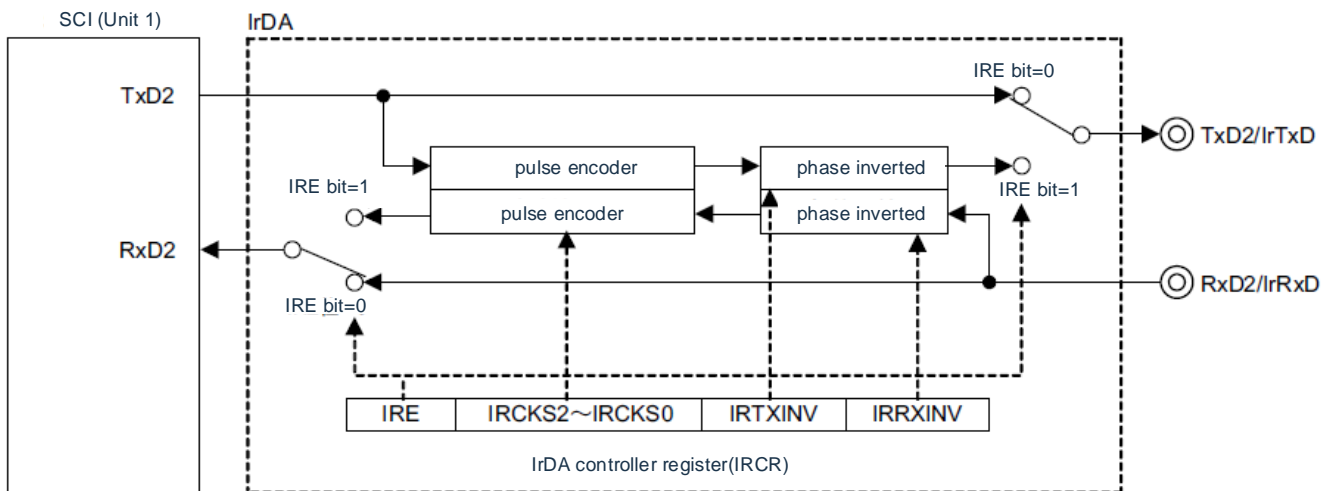


Table 19-1: Pin structure of IrDA

Pin name	I/O	Function
IrTxD	Output	Output pin for sending data
IrRxD	Input	Input pin for receiving data

## 19.2 Registers for controlling IrDA

IrDA functions are controlled through the following registers.

- Peripheral enable register 0 (PER0)
- IrDA control register (IRCR)

### 19.2.1 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 IrDA, bit5(IRDAEN) of PER0 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".

Figure19-2: Format of peripheral enabled register 0 (PER0)

Address: 40020420H	After reset: 00H				R/W			
Symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	IICAEN	IRDAEN	SCI2EN	SCI1EN	SCI0EN	TMAEN	TM80EN

IRDAEN	Control of input clock of IrDA
0	Stop provide an input clock. • Cannot write the SFR used by IrDA. • IrDA is in a reset state.
1	Provides an input clock. Can read and write SFRs used by IrDA.

Notice: When you set up IrDA, you must first set the IRDAEN location "1". When the IRDAEN bit is '0', the write operation of the control register of the IrDA is ignored, and the read values are all initial.

## 19.2.2 IrDA control register (IRCR)

This is the register that controls the IrDA function. It performs polarity switching of received data and transmitted data, clock selection of IrDA, and selection of serial input/output pin function (usually serial function and IrDA function) switching. The IRCR register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

Figure19-3: Format of the system clock control register (IRCR)

Address: 40048000H

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	Serial input/output pins are used as IrDA functionality

IRCKS2	IRCKS1	IRCKS0	Clock Selection for IrDA
0	0	0	B3/16 (B= bit rate)
0	0	1	F <sub>CLK</sub> /2
0	1	0	F <sub>CLK</sub> /4
0	1	1	F <sub>CLK</sub> /8
1	0	0	F <sub>CLK</sub> /16
1	0	1	F <sub>CLK</sub> /32
1	1	0	F <sub>CLK</sub> /64
1	1	1	Settings are disabled.

IRTXINV	Polarity Switching of IrTxD Data
0	IrTxD output of sent data
1	IrTxD output after reversing transmit data

IRRXINV	Polarity Switching of IrRxD Data
0	Use input data for IrRxD pins as received data
1	The data after the input data of the inverted IrRxD pin is used as the receiving data

Notice:

1. bit1 and bit0 must be set to 0.
2. The IRCKS[2:0] bits, IRTXINV bits and IRRXINV bits can be set only if the IRE bit is "0".



## 19.3 Operation of IrDA

### 19.3.1 Procedure for IrDA communication

(1) Initial set-up process for IrDA communication

Follow these steps to initially set up IrDA.

- (a) Set the IRDAEN bit of the PER0 register to "1".
- (b) Set IRCR register.
- (c) Set SCI related registers (see UART mode set-up step).

(2) Stop process for IrDA communication

- (a) Set the IrTxD pin status after IrDA communication is stopped by setting the port register and the port mode register.

Remark: When IrDA reset is performed by step 3, the IrTxD pin may change the output state by switching to the data output of the usual serial interface UART.

- When outputs low level from IrTxD pin  
Set the port register to "0". Immediately after this setting, the IrTxD pin is fixed at a low level.
- When outputs a high level from the IrTxD pin  
Set the port register to "1". With this setting, the IrTxD pin is immediately fixed to a high level after the IrDA reset in step 3.
- Setting the port mode register "1" with the IrTxD pin to the Hi-Z state  
Immediately after this setting, the IrTxD pin changes to the Hi-Z state
- (b) Set the STm0 and STm1 bits of the STm register (SCI's related register) to "1" (to stop the operation of SCI's channel 0 and channel 1).
- (c) Set the IRDAEN bit of PER0 register to "0", and perform IrDA reset.

You cannot set the STm0 and STm1 bits of the STm register to "1" or the IRE bit of the IrDA to "0" in addition to the above steps.

(3) Steps when sending IrDA frame errors

When a frame error occurs during IrDA communication, the following settings must be made in order to be set to a state where subsequent data can be received.

- (a) The STm1 bit of the STm register of the SCI is "1" (stopping the operation of channel 1)
- (b) The SSm1 bit of the SCI's SSm register is "1" (start running channel 1 of SCI)

Remark: m: unit number (m=0),

Please refer to "Chapter 16 General-purpose Serial Communications Unit" for more information on the handling of frame errors in SCI.

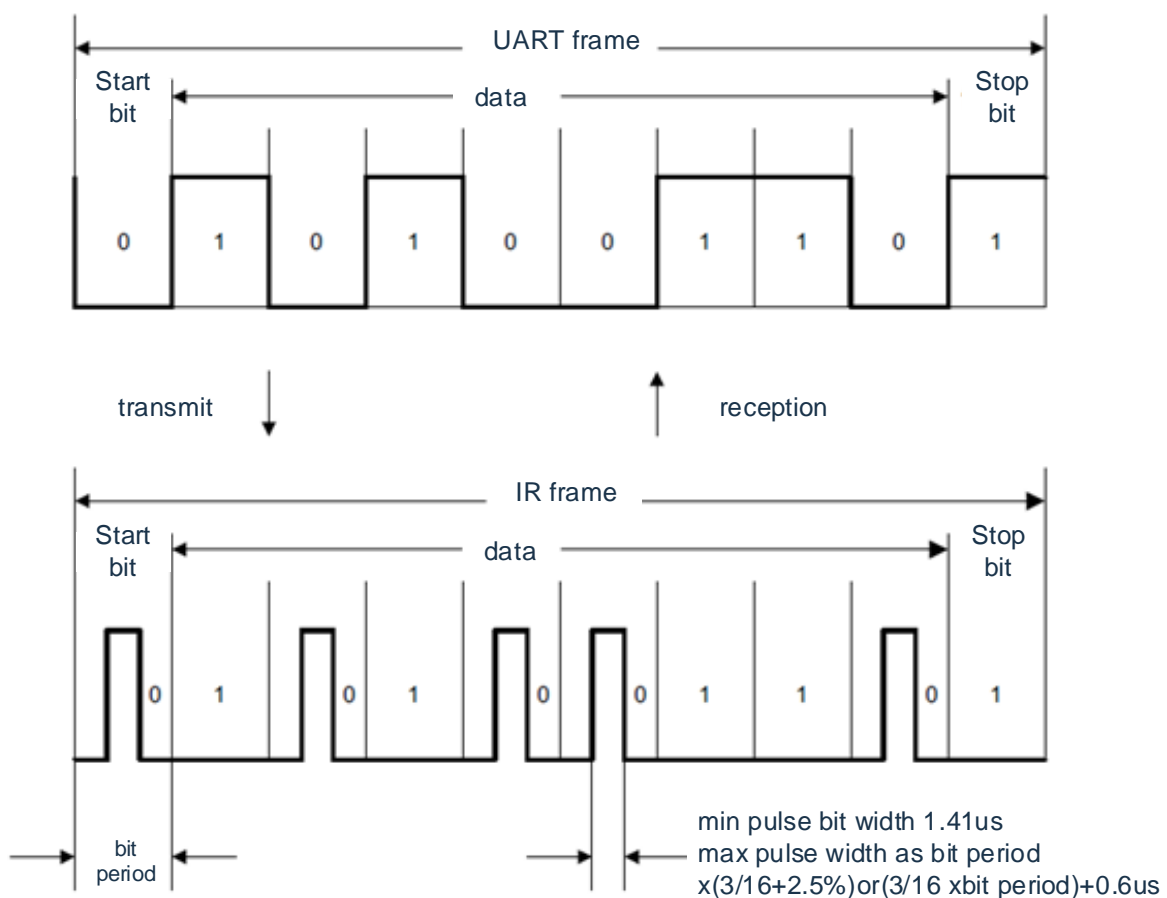
## 19.3.2 Transmission

At transmission, the output signal (UART frames) from the SCI is converted to IR frames via IrDA (see Figure 19-4).

When the IRTXINV bit is "0" and the serial data is "0", the output bit period (1 bit width period)  $\times 3/16$  high level pulse (initial value). In addition, the high level pulse width can be changed according to the setting value of IRCKS2~IRCKS0 bits. According to the standard, the minimum high level pulse width is 1.41 $\mu$ S, and the maximum is  $(3/16+2.5\%) \times \text{bit cycle}$ , or  $(3/16 \times \text{bit cycle})+0.6\mu\text{S}$ .

When the CPU or peripheral hardware clock ( $F_{\text{CLK}}$ ) is 24 MHz, the minimum high level pulse width is 1.5 $\mu$ s. In addition, no pulses are output when the serial data is "1".

Figure 19-4: Transmit/Receive operation of IrDA



## 19.3.3 Reception

When receiving, the data of IR frame is converted to UART frame through IrDA, then input to SCI. Low level data is output when the IRRXINV bit is '0' and a high level pulse is detected. If no pulse is present in a 1-bit period, high level data is output. It must be note that pulses less than that minimum pulse width of 1.41  $\mu$ s cannot be identified.

## 19.3.4 Selection of high level pulse width

If the pulse width at transmission is less than  $\frac{3}{16}$ , the suitable setting (minimum pulse width) of IRCKS2~IRCKS0 bit and the setting high level pulse width are shown in Table 19-2.

Table 19-2: Set value of IRCKS2~IRCKS0 bits

F <sub>CLK</sub> [MHz]	Item	<Upper Segment> Bit Rate [kbps]					
		<Lower Segment> Bit Rate 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	_Note 1	_Note 1	_Note 1
	High level pulse width[us]	2.00	2.00	2.00	_Note 1	_Note 1	_Note 1
2	IRCKS2~IRCKS0	010	010	010	010	010	_Note 1
	High level pulse width[us]	2.00	2.00	2.00	2.00	2.00	_Note 1
3	IRCKS2~IRCKS0	011	011	011	011	011	_Note 1
	High level pulse width[us]	2.67	2.67	2.67	2.67	2.67	_Note 1
4	IRCKS2~IRCKS0	011	011	011	011	011	000 Note 2
	High level pulse width[us]	2.00	2.00	2.00	2.00	2.00	1.50
6	IRCKS2~IRCKS0	100	100	100	100	100	000 Note 2
	High level pulse width[us]	2.67	2.67	2.67	2.67	2.67	1.50
8	IRCKS2~IRCKS0	100	100	100	100	100	000 Note 2
	High level pulse width[us]	2.00	2.00	2.00	2.00	2.00	1.50
12	IRCKS2~IRCKS0	101	101	101	101	101	000 Note 2
	High level pulse width[us]	2.67	2.67	2.67	2.67	2.67	1.50
16	IRCKS2~IRCKS0	101	101	101	101	101	000 Note 2
	High level pulse width[us]	2.00	2.00	2.00	2.00	2.00	1.50
24	IRCKS2~IRCKS0	110	110	110	110	110	000 Note 2
	High level pulse width[us]	2.67	2.67	2.67	2.67	2.67	1.50

Note 1: “-” indicates that communication standards are not met.

Note 2: The pulse width cannot be less than: bit rate×3/16.

## 19.4 Cautions when using IrDA

1. A runtime clock that allows or disables the provision of IrDA can be set through a peripheral admission register. The register cannot be accessed because the initial state is to disable clock provisioning. Prior to setting the register, it is necessary to set the state of the peripheral admission register to allow the provision of the IrDA running clock.
2. In sleep mode, the IrDA feature runs continuously.
3. Do not use SCI's initialization during IrDA communication (SS bit=1).
4. The IRRXINV bit, IRTXINV bit, and IRCKS[2:0] bit of the IRCR register can only be set if the IRE bit is '0'.

# Chapter 20 LCD Controller/Driver

## 20.1 Function of LCD controller / driver

The functions of the LCD controller / driver are shown below.

- (1) You can select either waveform A or waveform B.
- (2) The LCD driver voltage generation circuit is capable of internal boost, capacitive splitting and resistive splitting switching.
- (3) Automatic output of segment and common signals by automatic reading of display data registers.
- (4) The ability to select from 16 reference voltages (to adjust the contrast) generated during the operation of the boost circuit.
- (5) LCD blinking display is available.

The maximum number of pixels that can be displayed in each display mode is shown in Table 20-1.

Table 20-1: Maximum display pixels

Drive waveform of LCD driver	Drive voltage generation circuit of LCD driver	Bias method	Time slice	Maximum display pixels
Waveform A	Resistance splitting	-	Static	42 (42 segments x 1 common)
		1/2	2	84 (84 segments x 2 common)
			3	126 (42 segments x 3 common)
		1/3	3	
			4	168 (42 segments x 4 common)
		1/4	8	304 (38 segments x 8 common)
	Internal boost	1/3	3	126 (42 segments x 3 common)
			4	168 (42 segments x 4 common)
		1/4	6	240 (40 segments x 6 common)
			8	304 (38 segments x 8 common)
	Capacitive splitting	1/3	3	126 (42 segments x 3 common)
			4	168 (42 segments x 4 common)
Waveform B	Resistor split, internal boost	1/3	4	304 (38 segments x 8 common)
		1/4	8	
	Capacitive splitting	1/3	4	42 (42 segments x 1 common)

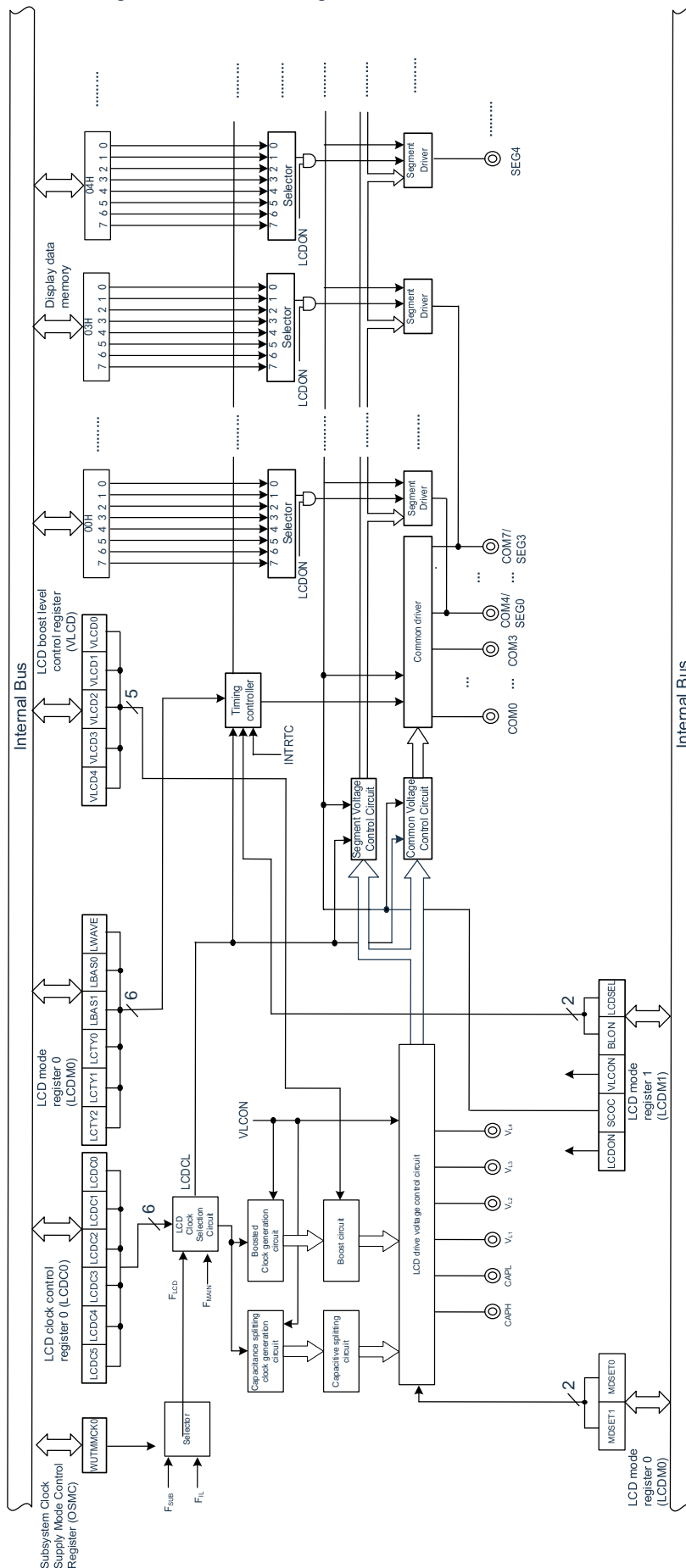
## 20.2 Structure of LCD controller / driver

The LCD controller / driver consists of the following hardware

Table 20-2: LCD controller/driver structure

Item	Structure
Control register	LCD mode register 1 (LCDM1)
	Subsystem clock supply mode control register (OSMC)
	LCD clock control register 0 (LCDC0)
	LCD Boost Level Control Register (VLCD)
	LCD input switching control register (ISCLCD)
	LCD port function register (SEGEN0,SEGEN1,SEGEN2,SEGEN3)

Figure 20-1: Block diagram of LCD controller/driver



## 20.3 Registers for controlling LCD controller/driver

The LCD controller/driver is controlled by the following 7 registers.

- LCD mode register 0(LCDM0)
- LCD mode register 1(LCDM1)
- Subsystem clock supply mode control register (OSMC)
- LCD clock control register 0(LCDC0)
- LCD boost level control register ( VLCD)
- LCD input switching control register ( ISCLCD)
- LCD port function register (SEGEN0, SEGEN1, SEGEN2, SEGEN3)



## 20.3.1 LCD mode register 0(LCDM0)

This is the register for setting the LCD operation.

The LCDM0 register is set by an 8-bit memory manipulation instruction.

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

Figure 20-2: Format of LCD mode register 0 (LCDM0)

Address: 0x40049040H

After reset: 00H

R/W

Symbol	7	6	5	4	3	2	1	0
LCDM0	MDSET1	MDSET0	LWAVE	LDTY2	LDTY1	LDTY0	LBAS1	LBAS0

MDSET1	MDSET0	Internal resistance splitting method
0	0	Internal boost method
0	1	Capacitive splitting method
1	0	External resistor splitting method
1	1	Internal resistance splitting method

LWAVE	LCD display waveform selection
0	Waveform A
1	Waveform B

LDTY2	LDTY1	LDTY0	Time slice selection for LCD display
0	0	0	Static
0	0	1	2 time slices
0	1	0	3 time slices
0	1	1	4 time slices
1	0	0	6 time slices
1	0	1	8 time slices
Others:			Settings are disabled.

LBAS1	LBAS0	Selection of bias method for LCD display
0	0	1/2 bias method
0	1	1/3 bias method
1	0	1/4 bias method
1	1	Settings are disabled.

Notice:

1. When the SCOC bit of LCDM1 register is "1", the value of LCDM0 register cannot be overwritten.
2. When static (LDTY2~LDTY0=000B) is selected, the LBAS1 bit and LBAS0 bits must be set to the initial value ("00B"). If a value other than the initial value is set, operation is not guaranteed.
3. Only the combination of display waveform, time slice, and bias method shown in Table 20-3 is set. The combinations other than those shown in Table 20-3 are prohibited.

Table 20-3: Combination of display waveform, time slice, bias method and frame rate

Display Mode			Set value						Drive voltage generation method		
Display waveform	Time slice	Bias method	LWAVE	LDTY2	LDTY1	LDTY0	LBAS1	LBAS0	External resistance splitting	Internal splitting	Capacitive splitting
Waveform A	8	1/4	0	1	0	1	1	0	○ (24~128Hz)	○ (24~64Hz)	X
Waveform A	6	1/4	0	1	0	0	1	0	X	○ (32~86Hz)	X
Waveform A	4	1/3	0	0	1	1	0	1	○ (24~128Hz)	○ (24~128Hz)	○ (24~128Hz)
Waveform A	3	1/3	0	0	1	0	0	1	○ (32~128Hz)	○ (32~128Hz)	○ (32~128Hz)
Waveform A	3	1/2	0	0	1	0	0	0	○ (32~128Hz)	X	X
Waveform A	2	1/2	0	0	0	1	0	0	○ (24~128Hz)	X	X
Waveform A	Static		0	0	0	0	0	0	○ (24~128Hz)	X	X
Waveform B	8	1/4	1	1	0	1	1	0	○ (24~128Hz)	○ (24~64Hz)	X
Waveform B	4	1/3	1	0	1	1	0	1	○ (24~128Hz)	○ (24~128Hz)	○ (24~128Hz)

Remark: ○: Match

X: Not match

## 20.3.2 LCD mode register 1(LCDM1)

This register enables or disables the display operation, the operation of the booster and capacitor splitting circuits and sets the display data area and low voltage mode.

The LCDM1 register is set by an 8-bit memory manipulation instruction.

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

Figure 20-3: Format of LCD mode register 1 (LCDM1)

Address: 0x40049041H		After reset: 00H			R/W			
Symbol	7	6	5	4	3	2	1	0
LCDM1	LCDON	SCOC	VLCON	BLON	LCDSEL	0	0	LCDVLM

SCOC	LCDON	LCD display enable and disable
0	0	Outputs ground level to segment pin or common pin
0	1	
1	0	Display OFF (all segment outputs are non-selective signal outputs)
1	1	Display ON

VLCON	Operation of the booster circuit or capacitor divider circuit
0	Stop the operation of the booster circuit or capacitor divider circuit
1 <sup>Note 1</sup>	Allows operation of booster circuits or capacitor split circuits

BLON	LCDSEL	Control of display data area
0 <sup>Note 2</sup>	0	Display data in the waveform A (lower 4 bits of the LCD display data register)
0	1	Display data in the waveform B (higher 4 bits of the LCD display data register)
1	0	Alternate display of data in waveform A and waveform B (blinking display corresponding to the fixed-cycle interrupt (INTRTC) timing of real-time clock 2 (RTC2))
1	1	

LCDVLM <sup>Note 3</sup>	Switching control of the initial value of the boost pin
0	$V_{DD}$ voltage $\geq 2.7V$
1	$V_{DD}$ voltage $\leq 4.2V$

Note 1: In external resistor splitting mode, setting is prohibited.

Note 2: To select  $F_{IL}$  as the LCD source clock ( $F_{LCD}$ ), you must set BLON to "0".

Note 3: This is a function that sets the initial state of the VLx pin when using a boost circuit and reduces the boost stabilization time.

If the voltage at  $V_{DD}$  is greater than or equal to 2.7V at the beginning of the boost, the LCDVLM bit must be set to "0"; if the voltage at  $V_{DD}$  is less than or equal to 4.2V, the LCDVLM bit must be set to "1". However, when  $2.7V \leq V_{DD} \leq 4.2V$ , either LCDVLM bit "0" or LCDVLM bit "1" can be operated.

Notice:

1. When the boost circuit is used without LCD display, the SCOC and VLCON bits must be set to "0" and the MDSET1 and MDSET0 bits must be set to "00B" to reduce power consumption. When the

MDSET1 and MDSET0 bits are "01B", the internal reference voltage generation operates internally and therefore consumes power.

2. When setting the external resistor splitting method (MDSET1, MDSET0=00B for LCDM0) or capacitor splitting method (MDSET1, MDSET0=10B), LCDVLM must be set to "0".
3. When the SCOC bit is "1", the VLCON bit and LCDVLM bit cannot be overwritten.
4. When selecting the display mode for 8 time slices, the BLON bit and LCDSEL bit must be set to "0".
5. When using the internal boost method, you must wait for the reference voltage preparation time (5ms (MIN.)) after setting the reference voltage through the VLCD register (when using the reference voltage with the default value, after selecting the internal boost method (MDSET1, MDSET0=01B in the LCDM0 register), and then set VLCON bit to "1").

### 20.3.3 Subsystem Clock Supply Mode Control Register (OSMC)

OSMC registers is a register that reduces power consumption by stopping unneeded clock functions.

If RTCLPC is set to "1", it stops clocking peripheral functions other than the real time clock, 15-bit interval timer, clock output/buzzer output and LCD controller/driver in STOP mode or HALT mode where the CPU is running with a sub-system clock, thus reducing power consumption.

The operating clock of the LCD controller/driver can be selected through the OSMC register.

For details, please refer to "4.3.9 Subsystem clock supply mode control register (OSMC)".

## 20.3.4 LCD clock control register 0 (LCDC0)

This is the register for setting the LCD clock. The frame rate is determined by the LCD clock and time slice.

The LCDC0 register is set by an 8-bit memory manipulation instruction.

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

Figure 20-4: Format of LCD clock control register (LCDC0)

Address: 0x40049042H	After reset: 00H	R/W						
Symbol	7	6	5	4	3	2	1	0
LCDC0	0	0	LCDC05	LCDC04	LCDC03	LCDC02	LCDC01	LCDC00

LCDC05	LCDC04	LCDC03	LCDC02	LCDC01	LCDC00	LCD Clock(LCDCL)
0	0	0	1	0	0	$F_{SUB}/2^5$ or $F_{IL}/2^5$
0	0	0	1	0	1	$F_{SUB}/2^6$ or $F_{IL}/2^6$
0	0	0	1	1	0	$F_{SUB}/2^7$ or $F_{IL}/2^7$
0	0	0	1	1	1	$F_{SUB}/2^8$ or $F_{IL}/2^8$
0	0	1	0	0	0	$F_{SUB}/2^9$ or $F_{IL}/2^9$
0	0	1	0	0	1	$F_{SUB}/2^{10}$
0	1	0	0	1	1	$F_{MAIN}/2^{10}$
0	1	0	1	0	0	$F_{MAIN}/2^{11}$
0	1	0	1	0	1	$F_{MAIN}/2^{12}$
0	1	0	1	1	0	$F_{MAIN}/2^{13}$
0	1	0	1	1	1	$F_{MAIN}/2^{14}$
0	1	1	0	0	0	$F_{MAIN}/2^{15}$
0	1	1	0	0	1	$F_{MAIN}/2^{16}$
0	1	1	0	1	0	$F_{MAIN}/2^{17}$
0	1	1	0	1	1	$F_{MAIN}/2^{18}$
1	0	1	0	1	1	$F_{MAIN}/2^{19}$
Others:						Settings are disabled.

Notice:

1. The LCDC0 register cannot be set when the SCOC bit of the LCDM1 register is "1".
2. Bit6 and bit7 must be set to "0".
3. The following settings must be made for the LCD clock (LCDCL) when setting the internal boost method or the capacitor split method.
  - When  $F_{SUB}$  is selected, it cannot exceed 512Hz.
  - When  $F_{IL}$  is selected, it cannot exceed 235Hz.

For details, please refer to "Table 20-3: ".

Remark:  $F_{MAIN}$ : Master system clock frequency

$F_{IL}$  : Clock frequency of the low-speed internal oscillator

$F_{SUB}$  : Subsystem clock frequency

## 20.3.5 LCD Boost Level Control Register (VLCD)

This is a register to select the generated reference voltage (to adjust the contrast) during the operation of the boost circuit. 16 reference voltages can be selected.

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

After a reset signal is generated, the value of this register becomes "04H".

Figure 20-5: Format of LCD Boost Level Control Register (VLCD)

Address: 0x40049043H

After reset: 04H

R/W

Symbol	7	6	5	4	3	2	1	0
VLCD	0	0	0	VLCD4	VLCD3	VLCD2	VLCD1	VLCD0

VLCD4	VLCD3	VLCD2	VLCD1	VLCD0	Selection of reference voltage (to adjust the contrast)	VL <sub>4</sub> Voltage	
						1/3 bias method	1/4 bias method
0	0	0	1	1	1.00V	3.00V	4.00V
0	0	1	0	0	1.05V	3.15V	4.20V
0	0	1	0	1	1.10V	3.30V	4.40V
0	0	1	1	0	1.15V	3.45V	4.60V
0	0	1	1	1	1.20V	3.60V	4.80V
0	1	0	0	0	1.25V	3.75V	5.00V
0	1	0	0	1	1.30V	3.90V	5.20V
0	1	0	1	0	1.35V	4.05V	Settings are disabled.
0	1	0	1	1	1.40V	4.20V	Settings are disabled.
0	1	1	0	0	1.45V	4.35V	Settings are disabled.
0	1	1	0	1	1.50V	4.50V	Settings are disabled.
0	1	1	1	0	1.55V	4.65V	Settings are disabled.
0	1	1	1	1	1.60V	4.80V	Settings are disabled.
1	0	0	0	0	1.65V	4.95V	Settings are disabled.
1	0	0	0	1	1.70V	5.10V	Settings are disabled.
1	0	0	1	0	1.75V	5.25V	Settings are disabled.
Others:						Settings are disabled.	

## Notice:

1. The VLCD register setting is valid only when the boost circuit is running.
2. Bits 5 to 7 must be set to "0".
3. The value of the VLCD register must be changed after the operation of the boost circuit is stopped (VLCON=0).
4. When using the internal boost method, you must wait for the reference voltage preparation time (5ms (MIN.)) after setting the reference voltage through the VLCD register (when using the reference voltage at the default value, after selecting the internal boost method (MDSET1, MDSET0=01B in the LCDM0 register), and then set VLCON bit to "1").
5. In the case of external resistor splitting or capacitor splitting, the VLCD register must be the initial value "04H".



## 20.3.6 LCD input switching control register (ISCLCD)

The input of Schmitt trigger buffer needs to be disabled in order to prevent through-current flow during the period when the CAPL, CAPH, and VLx pins are set to operate as LCD functions.

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

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

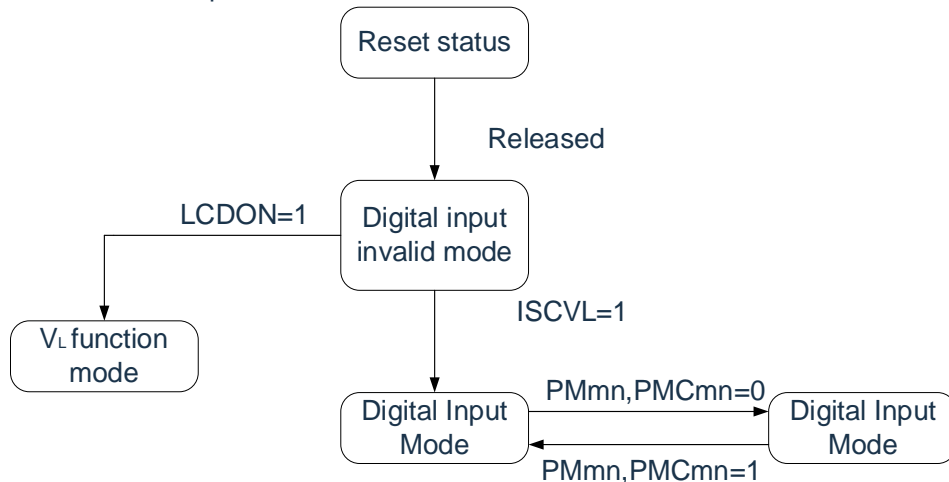
Figure 20-6: LCD input switching control register (ISCLCD)

Address: 0x40040804H	7	6	5	4	3	2	1	0
Symbol								
ISCLCD	0	0	0	0	0	0	ISCVL	ISCCAP

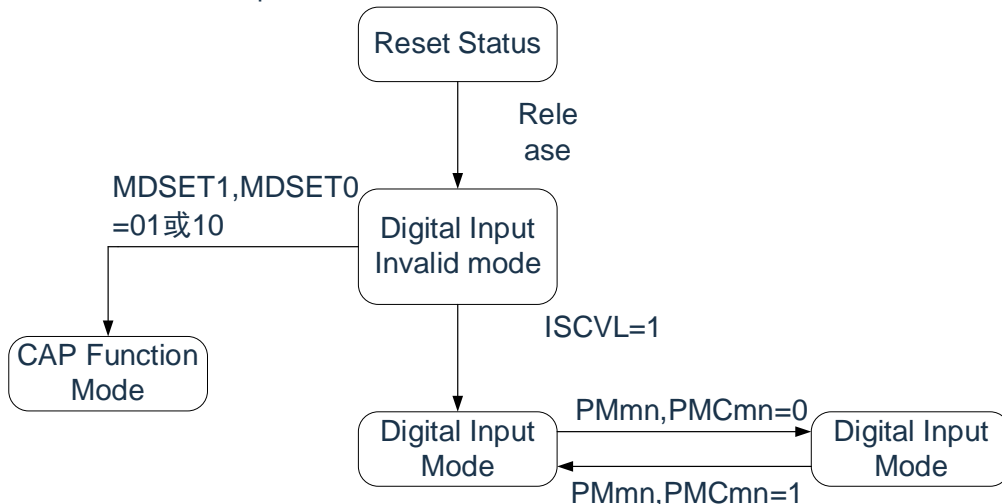
ISCVL	Control of the Schmitt trigger buffer on the VLx pin
0	Invalid input
1	Input valid

ISCCAP	Control of the Schmitt trigger buffers on the CAPL and CAPH pins
0	Invalid input
1	Input valid

The state transfer of the VLx pin function is shown below.



The state transfer of the CAPx pin function is shown below.



## 20.3.7 LCD port function register

This is the register that sets whether to use the Pmn pin as a port or as an LCD output.

The SEGEN0, SEGEN1, SEGEN2 and SEGEN3 registers are set by 16-bit memory manipulation instructions.

After a reset signal is generated, the value of these registers becomes "0000H".

Figure 20-7: Format of LCD port function register

Address:	After reset:		R/W					
0x40040806H	0000H							
Symbol	15	14	13	12	11	10	9	8
SEGEN0	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	COMEN7/ SEGEN3	COMEN6/ SEGEN2	COMEN5/ SEGEN1	COMEN4/ SEGEN0	COMEN3	COMEN2	COMEN1	COMEN0

Address:	After reset:							
0x40040808H	0000H R/W							
Symbol	15	14	13	12	11	10	9	8
SEGEN1	SEGEN15	SEGEN14	SEGEN13	SEGEN12	SEGEN11	SEGEN10	SEGEN9	SEGEN8
	7	6	5	4	3	2	1	0
	SEGEN7	SEGEN6	SEGEN5	SEGEN4	0	0	0	0

Address:	After reset:							
0x4004080AH	0000H							
Symbol	15	14	13	12	11	10	9	8
SEGEN2	SEGEN31	SEGEN30	SEGEN29	SEGEN28	SEGEN27	SEGEN26	SEGEN25	SEGEN24
	7	6	5	4	3	2	1	0
	SEGEN23	SEGEN22	SEGEN21	SEGEN20	SEGEN19	SEGEN18	SEGEN17	SEGEN16

Address:	After reset:							
0x4004080CH	0000H							
	R/W							
Symbol	15	14	13	12	11	10	9	8
SEGEN3	0	0	0	0	0	0	SEGEN41	SEGEN40
	7	6	5	4	3	2	1	0
	SEGEN39	SEGEN38	SEGEN37	SEGEN36	SEGEN35	SEGEN34	SEGEN33	SEGEN32

COMENm	Function of the Pmn pin
0	Used as a port
1	Used as the COM output

SEGENm	Function of the Pmn pin
0	Used as a port
1	Used as the SEG output

Remark: When used as COM/SEG output (SEGENxx=1), the PUMn bit of PUM register, the POMmn bit of POMm register, the PIMmn bit of PIMm register and the Pmn bit of Pm register must be set to "0".  
m=(1-3).

## 20.4 LCD display data register

The LCD display data register is mapped as follows Table 20-4. The LCD display can be changed by changing the contents of the LCD display data register.

Table 20-4: Relationship between the contents of the LCD display data register, segment output, and common output (1/2)

(a) 6 time slices and 8 time slices beyond (static, 2 time slices, 3 time slices, 4 time slices)

Register Name	Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
		COM3	COM2	COM1	COM0	COM3	COM2	COM1	COM0
SEG0	0x40049000H	SEG0 (Graphic area B)				SEG0 (Graphic area A)			
SEG1	0x40049001H	SEG1 (Graphic area B)				SEG1 (Graphic area A)			
SEG2	0x40049002H	SEG2 (Graphic area B)				SEG2 (Graphic area A)			
SEG3	0x40049003H	SEG3 (Graphic area B)				SEG3 (Graphic area A)			
SEG4	0x40049004H	SEG4 (Graphic area B)				SEG4 (Graphic area A)			
SEG5	0x40049005H	SEG5 (Graphic area B)				SEG5 (Graphic area A)			
SEG6	0x40049006H	SEG6 (Graphic area B)				SEG6 (Graphic area A)			
SEG7	0x40049007H	SEG7 (Graphic area B)				SEG7 (Graphic area A)			
SEG8	0x40049008H	SEG8 (Graphic area B)				SEG8 (Graphic area A)			
SEG9	0x40049009H	SEG9 (Graphic area B)				SEG9 (Graphic area A)			
SEG10	0x4004900AH	SEG10 (Graphic area B)				SEG10 (Graphic area A)			
SEG11	0x4004900BH	SEG11 (Graphic area B)				SEG11 (Graphic area A)			
SEG12	0x4004900CH	SEG12 (Graphic area B)				SEG12 (Graphic area A)			
SEG13	0x4004900DH	SEG13 (Graphic area B)				SEG13 (Graphic area A)			
SEG14	0x4004900EH	SEG14 (Graphic area B)				SEG14 (Graphic area A)			
SEG15	0x4004900FH	SEG15 (Graphic area B)				SEG15 (Graphic area A)			
SEG16	0x40049010H	SEG16 (Graphic area B)				SEG16 (Graphic area A)			
SEG17	0x40049011H	SEG17 (Graphic area B)				SEG17 (Graphic area A)			
SEG18	0x40049012H	SEG18 (Graphic area B)				SEG18 (Graphic area A)			
SEG19	0x40049013H	SEG19 (Graphic area B)				SEG19 (Graphic area A)			
SEG20	0x40049014H	SEG20 (Graphic area B)				SEG20 (Graphic area A)			
SEG21	0x40049015H	SEG21 (Graphic area B)				SEG21 (Graphic area A)			
SEG22	0x40049016H	SEG22 (Graphic area B)				SEG22 (Graphic area A)			
SEG23	0x40049017H	SEG23 (Graphic area B)				SEG23 (Graphic area A)			
SEG24	0x40049018H	SEG24 (Graphic area B)				SEG24 (Graphic area A)			
SEG25	0x40049019H	SEG25 (Graphic area B)				SEG25 (Graphic area A)			
SEG26	0x4004901AH	SEG26 (Graphic area B)				SEG26 (Graphic area A)			
SEG27	0x4004901BH	SEG27 (Graphic area B)				SEG27 (Graphic area A)			
SEG32	0x40049020H	SEG32 (Graphic area B)				SEG32 (Graphic area A)			
SEG33	0x40049021H	SEG33 (Graphic area B)				SEG33 (Graphic area A)			
SEG34	0x40049022H	SEG34 (Graphic area B)				SEG34 (Graphic area A)			
SEG35	0x40049023H	SEG35 (Graphic area B)				SEG35 (Graphic area A)			
SEG36	0x40049024H	SEG36 (Graphic area B)				SEG36 (Graphic area A)			
SEG37	0x40049025H	SEG37 (Graphic area B)				SEG37 (Graphic area A)			
SEG38	0x40049026H	SEG38 (Graphic area B)				SEG38 (Graphic area A)			
SEG39	0x40049027H	SEG39 (Graphic area B)				SEG39 (Graphic area A)			

Table 20-4: Relationship between the contents of the LCD display data register, segment output, and common output (2/2)

(b) 6 time slices and 8 time slices

Register Name	Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
		COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0
SEG0	0x40049000H	SEG0 <sup>Note</sup>							
SEG1	0x40049001H	SEG1 <sup>Note</sup>							
SEG2	0x40049002H	SEG2 <sup>Note</sup>							
SEG3	0x40049003H	SEG3 <sup>Note</sup>							
SEG4	0x40049004H	SEG4							
SEG5	0x40049005H	SEG5							
SEG6	0x40049006H	SEG6							
SEG7	0x40049007H	SEG7							
SEG8	0x40049008H	SEG8							
SEG9	0x40049009H	SEG9							
SEG10	0x4004900AH	SEG10							
SEG11	0x4004900BH	SEG11							
SEG12	0x4004900CH	SEG12							
SEG13	0x4004560DH	SEG13							
SEG14	0x4004900EH	SEG14							
SEG15	0x4004900FH	SEG15							
SEG16	0x40049010H	SEG16							
SEG17	0x40049011H	SEG17							
SEG18	0x40049012H	SEG18							
SEG19	0x40049013H	SEG19							
SEG20	0x40049014H	SEG20							
SEG21	0x40049015H	SEG21							
SEG22	0x40049016H	SEG22							
SEG23	0x40049017H	SEG23							
SEG24	0x40049018H	SEG24							
SEG25	0x40049019H	SEG25							
SEG26	0x4004901AH	SEG26							
SEG27	0x4004901BH	SEG27							
SEG32	0x40049020H	SEG32							
SEG33	0x40049021H	SEG33							
SEG34	0x40049022H	SEG34							
SEG35	0x40049023H	SEG35							
SEG36	0x40049024H	SEG36							
SEG37	0x40049025H	SEG37							
SEG38	0x40049026H	SEG38							
SEG39	0x40049027H	SEG39							

Note: COM4 ~ COM7 pins and SEG0 ~ SEG3 pins are multiplexed.

When used for static, 2 time slices, 3 time slices or 4 time slices, the lower 4 bits of each address of the LCD display data register are in the graphic area A and the upper 4 bits are in the graphic area B.

The data in the graphic area A and the COM signal correspond to bit0->COM0, bit1->COM1, bit2->COM2, bit3->COM3.

The data in the graphic area B and the COM signal correspond to bit4->COM0, bit5->COM1, bit6->COM2, bit7->COM3.

When the BLON bit and LCDSEL bit are both "0", the LCD display shows the data in the graphic area A; when the BLON bit is "0" and the LCDSEL bit is "1", the LCD display shows the data in the graphic area B.

## 20.5 LCD display register selection

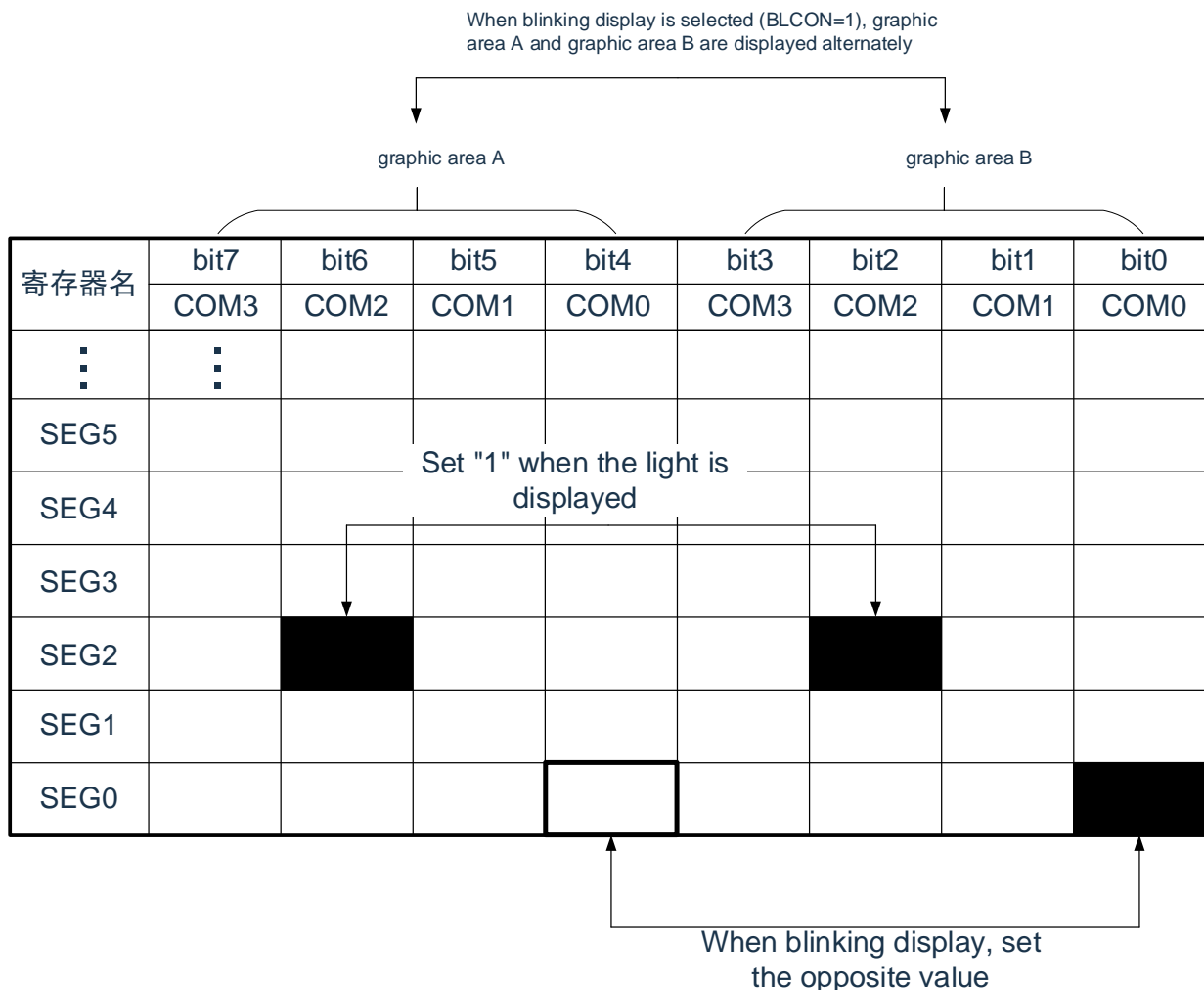
When used for static, 2 time slice, 3 time slice or 4 time slice, the following 3 LCD display data registers can be selected by setting the BLON bit and LCDSEL bit.

- Display data in the graphic area A (the lower 4 bits of the LCD display data register).
- Display data in the graphic area B (the upper 4 bits of the LCD display data register).

The data in graphic area A and graphic area B are displayed alternately (blinking display corresponding to the fixed-cycle interrupt timing of the real time clock (RTC)).

Notice: When using 6 or 8 time slices, the LCD display data register (graphic area A, graphic area B or blinking display) cannot be selected.

Figure 20-8: Example of LCD display register setting for graphic switching display



## 20.5.1 Data display in graphic area A and graphic area B

When both BLON bit and LCDSEL bit are "0", the data in graphic area A (the lower 4 bits of LCD display data register) is output as LCD display register.

When the BLON bit is "0" and the LCDSEL bit is "1", the data in the graphic area B (the upper 4 bits of the LCD display data register) is output as the LCD display register.

For the display area, please refer to "20.4 LCD".

## 20.5.2 Blinking display (alternate display of data in graph area A and graph area B)

When the BLON bit is "1", the data in the A and B graphics areas are displayed alternately in response to the fixed-cycle interrupt (INTRTC) timing of the real time clock (RTC). Refer to "Chapter 7 Real-Time Clock" for setting the timing of the fixed-cycle interrupt (INTRTC, 0.5s setting only) of the RTC.

When the LCD is blinking, the inverse value must be set for the bit in the B graphics area corresponding to the bit in the graphic area A (ex. set bit 0 of SEG0 to "1" and set bit 4 of SEG0 to "0" when the display is blinking); when the LCD is not blinking, the same value must be set (ex. set bit2 of SEG2 to "1" and set bit6 of SEG2 to "1" when the display is lit).

For the display area, please refer to "20.4 LCD display data registerLCD".

The timing of the display is shown below.

Figure 20-9: Switching operation from A graphical display to blinking display

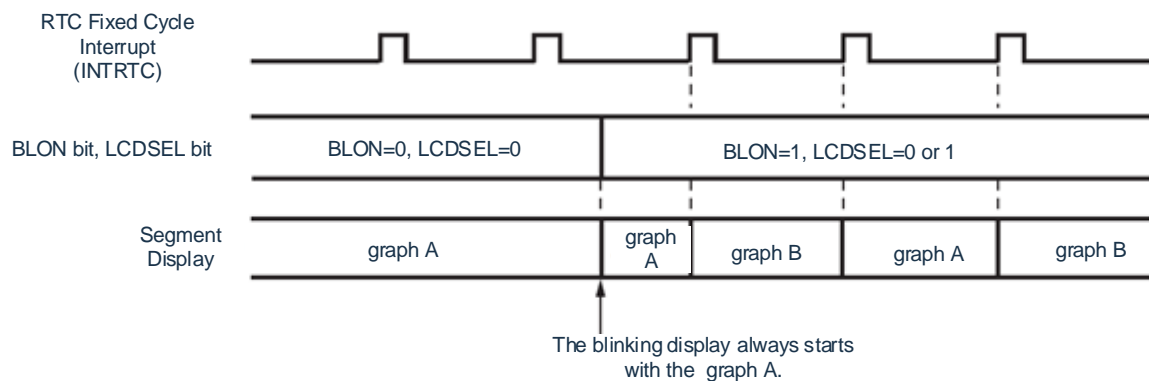
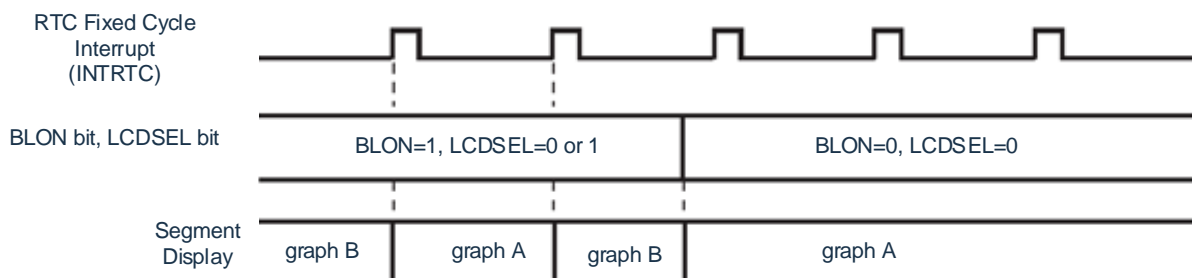


Figure 20-10: Switching operation from blinking display to A graphical display



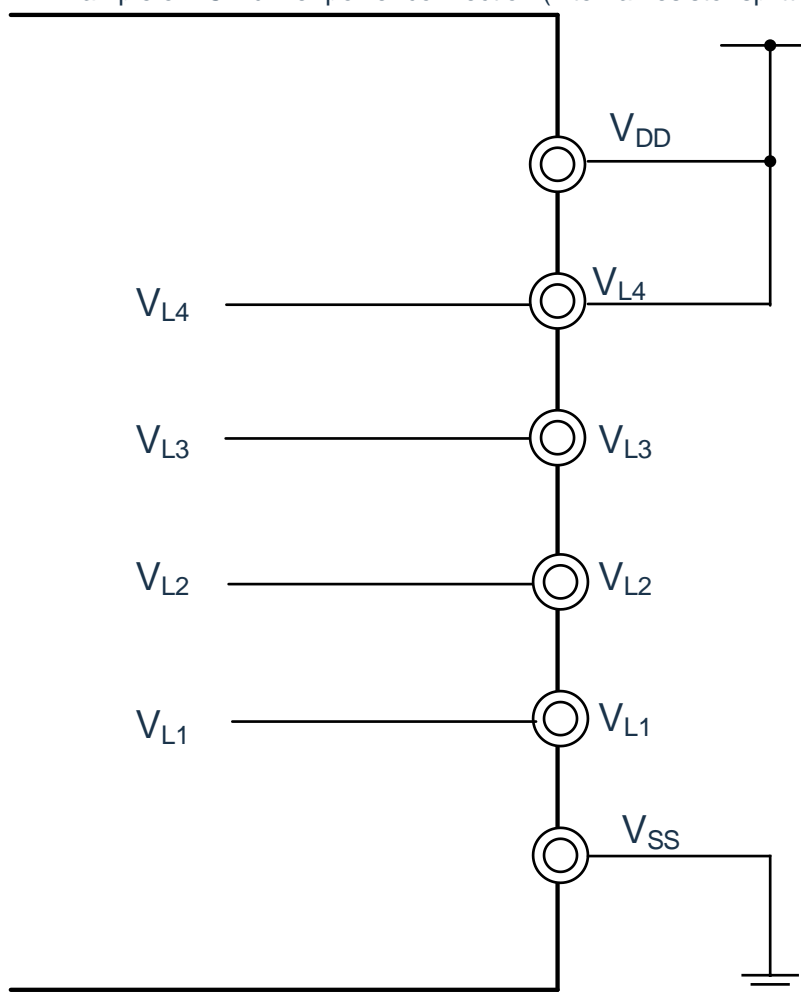
## 20.6 LCD drive voltage provided by $V_{L1}$ , $V_{L2}$ , $V_{L3}$ , $V_{L4}$

The LCD driver power supply generation method can be selected from internal resistance splitting method, external resistance splitting method, internal boost method and capacitance splitting method.

### 20.6.1 Internal resistor splitting method

The chip has an internal resistor splitting circuit for the LCD driver power supply, which is connected in the following way as Figure 20-11.

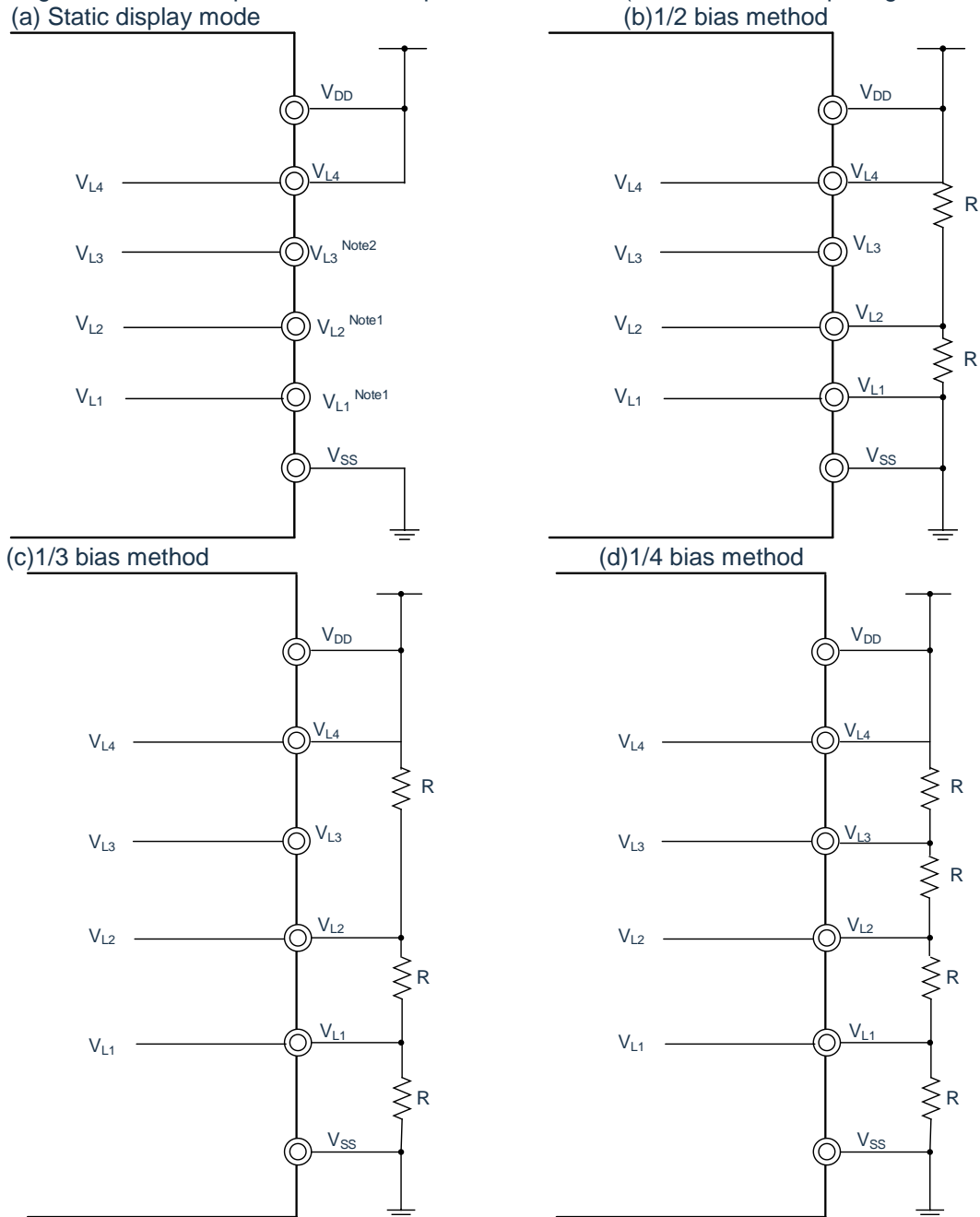
Figure 20-11: Example of LCD driver power connection (internal resistor splitting method)



## 20.6.2 External resistance splitting method

An example of LCD drive voltage connection according to each bias method is shown in Figure 20-12.

Figure 20-12: Example of LCD driver power connection (external resistor splitting method)



Note 1:  $V_{L1}$  and  $V_{L2}$  must be connected to GND or set to open circuit.

Note 2:  $V_{L3}$  can be used as a port.

Notice: The reference value of resistor  $R$  for external resistor decomposition is  $10\text{k}\Omega \sim 1\text{M}\Omega$ . To stabilize the potential of pins  $V_{L1} \sim V_{L4}$ , a capacitor must be connected between pins  $V_{L1} \sim V_{L4}$  and GND, as needed, with a reference value of approximately  $0.22\mu\text{F}$ , depending on the LCD display used, the number of segment pins, the number of common pins, the frame rate and the environment in which it is used. The capacitance value must be adjusted and determined based on a thorough evaluation of the system.



## 20.6.3 Internal boost method

The chip is equipped with an internal boost circuit for LCD drive power. The LCD drive voltage is generated by an external capacitor ( $0.47\mu\text{F}\pm 30\%$ ) of the internal boost circuit. The internal boost method can only use 1/3 bias method or 1/4 bias method.

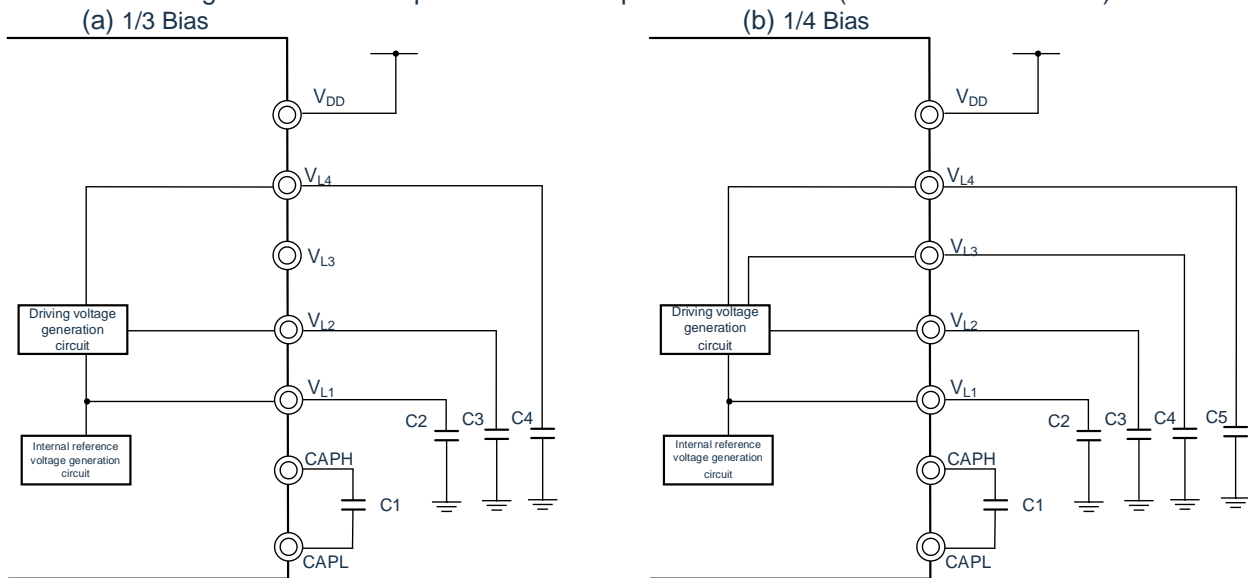
The LCD drive voltage of the internal boost method is not the same power supply as the device itself, so it can provide a fixed voltage independent of the variation of  $V_{DD}$ .

The contrast can be adjusted by setting the LCD boost control register ( $V_{LCD}$ ).

Table 20-5: L LCD drive voltage (internal boost method)

Display Mode LCD driver power pins	1/3 bias method	1/4 Bias method
$V_{L4}$	$3\times V_{L1}$	$4\times V_{L1}$
$V_{L3}$	-	$3\times V_{L1}$
$V_{L2}$	$2\times V_{L1}$	$2\times V_{L1}$
$V_{L1}$	LCD reference voltage	LCD reference voltage

Figure 20-13: Example of LCD driver power connection (internal boost method)



Remark:

1. Capacitors with low leakage current must be used as much as possible.
2. C1 must be a non-polarized capacitor.

## 20.6.4 Capacitance splitting method

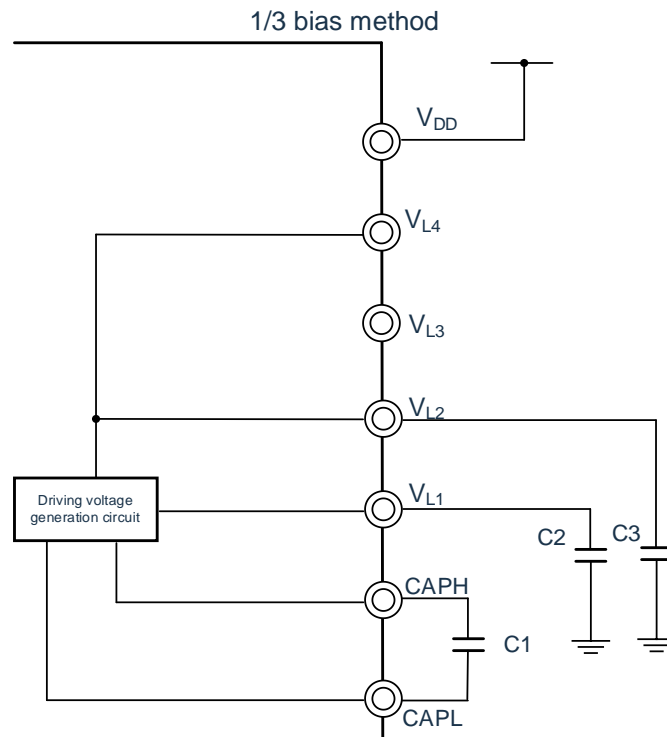
The chip has a built-in capacitance splitting circuit for LCD drive power. The LCD drive voltage is generated by an external capacitor ( $0.47\mu\text{F}\pm 30\%$ ) of the capacitance splitting circuit. Only 1/3 bias method can be used for capacitance splitting.

Unlike the external resistance splitting method, the capacitance splitting method has no current flowing through it, thus reducing the consumption current.

Table 20-6: LCD drive voltage (capacitance splitting method)

Display Mode	1/3 bias method
LCD driver power pins	
VL4	$V_{DD}$
VL3	-
VL2	$2/3 \times V_{L4}$
VL1	$1/3 \times V_{L4}$

Figure 20-14: Example of LCD driver power connection (capacitance splitting method)



Remark:

1. Capacitors with low leakage current must be used as much as possible.
2. C1 must be a non-polarized capacitor.

# Chapter 21 Enhanced DMA

## 21.1 Function of DMA

DMA is the function of transferring data between memories without using the CPU. Start the DMA for data transfer via peripheral interrupts. When the DMA and CPU access the same unit in flash, SRAM0, SRAM1, or peripheral modules at the same time, their bus usage is higher than the CPU. When the DMA and CPU access flash, SRAM0, SRAM1, or different units in the peripheral module, respectively, the two do not interfere with each other and can be executed in parallel.

The specifications of the DMA are shown in Table 21-1.

Table 21-1: Specification of the DMA (1/2)

Item		specification
Startup Source		Up to 24 startup sources
Assignable control data		24 groups
The address space that can be transmitted	Address space	Full address range space
	source	Full address range space is optional
	target	Full address range space is optional
Maximum number of transfers	Normal mode	65535 times
	Repeat pattern	65535 times
The maximum transfer block size	Normal mode (8-bit transmission).	65535 bytes
	Normal mode (16-bit transmission).	131070 bytes
	Normal mode (32-bit transmission).	262140 bytes
	Repeat pattern	65535 bytes
Teleportation units		8-bit/16-bit/32-bit
Transfer mode	Normal mode	Ends after a transfer of the DMACTj register from "1" to "0".
	Repeat pattern	After the transfer of the DMCTj register from "1" to "0" is completed, the address of the duplicate region is initialized, and the DMRLDj is changed
Address control	Normal mode	Fixed or incremented
	Repeat pattern	Fixed or incremented the address of the distinct area.
The priority of the startup source		Refer to "Table 21-5: ".

Table 21-1: Specification of DMA (2/2)

Item		specification
Interrupt the request	Normal mode	When a data transfer with a DMCTj register changed from "1" to "0", an interrupt of the source is requested to the CPU and interrupt processing is performed.
	Repeat pattern	When the RPTINT bit of the DMACRj register is "1" (enable an interrupt to be generated) and the data transfer of the DMACTj register changes from "1" to "0", it is directed to The CPU requests an interrupt that initiates the source and performs interrupt processing.
Transfer begins		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.
Transfer stops	Normal mode	<ul style="list-style-type: none"> <li>Set DMAENi0~DMAENi7 bits to "0" (boot is prohibited).</li> <li>When the DMACTj register changes from "1" to "0" at the end of the data transfer</li> </ul>
	Repeat pattern	<ul style="list-style-type: none"> <li>Set DMAENi0~DMAENi7 bits to "0" (boot is prohibited).</li> <li>Data transfer ends when the RPTINT bit is "1" (interrupts are allowed) and the DMCTj register changes from "1" to "0"</li> </ul>

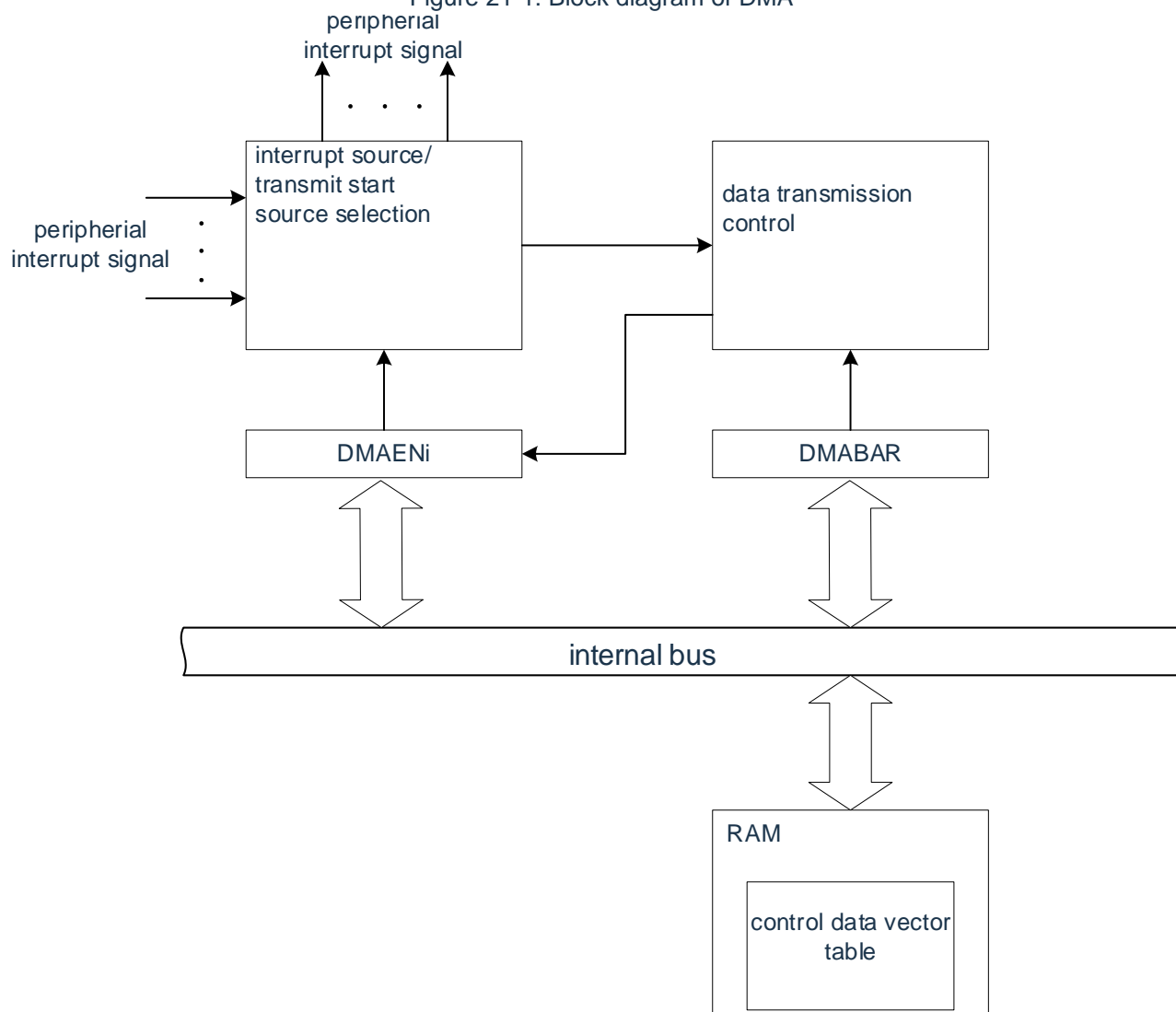
Note: In deep sleep mode, the flash memory stops running and cannot be used as a DMA transfer source.

Remark: i=0~2, j=0~23

## 21.2 Structure of DMA

A block diagram of the DMA is Figure 21-1.

Figure 21-1: Block diagram of DMA



## 21.3 Registers for controlling DMA

The registers that control the DMA are shown in Table 21-2.

Table 21-2: Registers for controlling DMA

Register Name	Symbol
Peripheral enable register 1	PER1
DMA start enable register 0	DMAEN0
DMA start enable register 1	DMAEN1
DMA start enable register 2	DMAEN2
DMAENi set register 0	DMSET0
DMAENi set register 1	DMSET1
DMAENi set register 2	DMSET2
DMAENi reset register 0	DMCLR0
DMAENi reset register 1	DMCLR1
DMAENi reset register 2	DMCLR2
DMA Base Address Register	DMABAR

Remark: i=0~2

The control data for the DMA is shown in Table 21-3.

The control data for the DMA is allocated in the DMA control data area of the RAM. The DMA control data area and the 704-byte region containing the DMA vector table area (the starting address where the control data is saved) are set up via the DMABAR register.

Table 21-3: DMA control data

Register Name	Symbol
DMA control register j	DMACRj
DMA block size register j	DMBLSj
DMA transmit times register j	DMACTj
The number of DMA transfers reloads register j	DMRLDj
DMA source address register j	DMSARj
DMA destination address register j	DMDARj

Remark: j=0~23

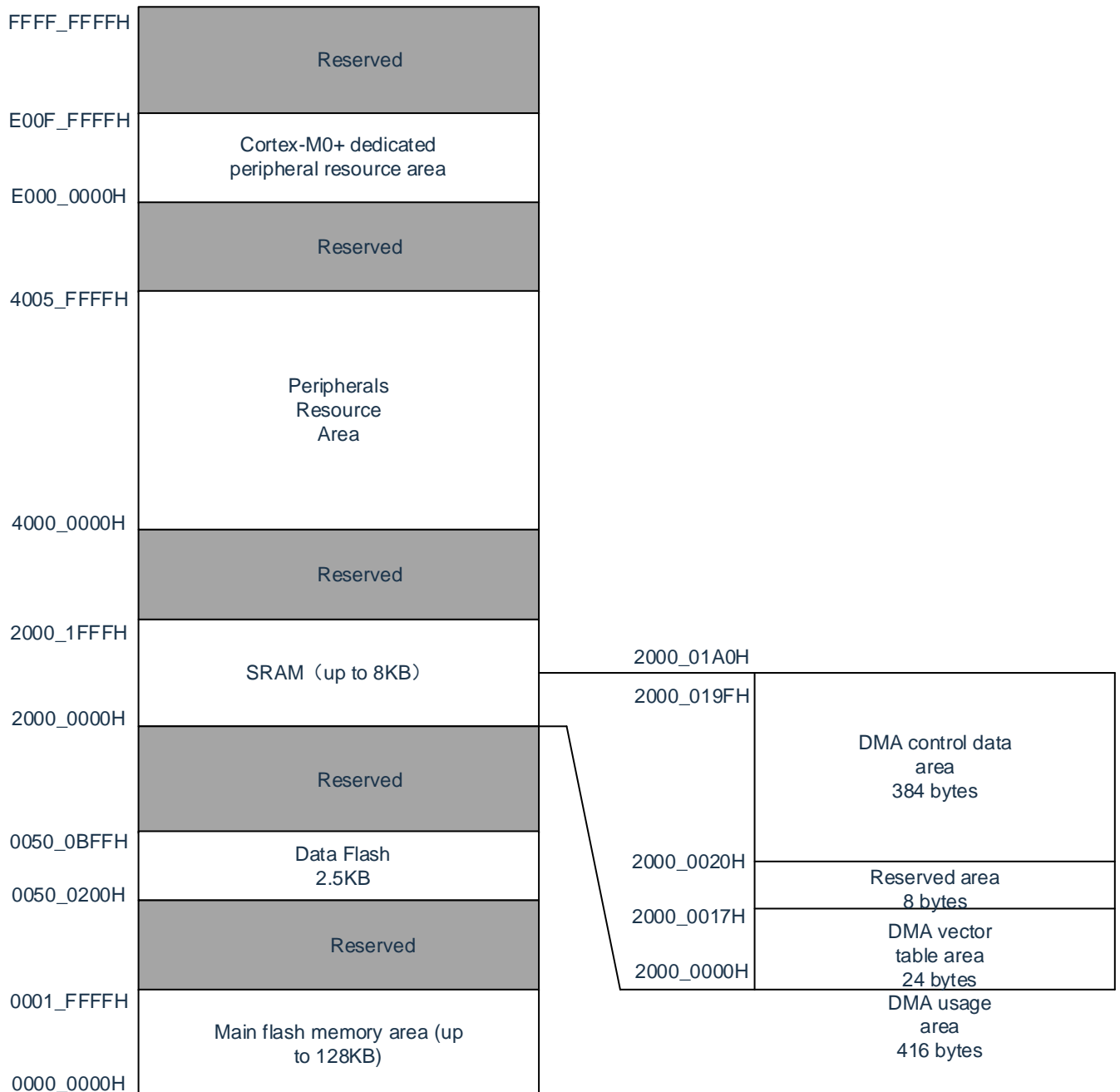
## 21.3.1 DMA control data areas and DMA vector table areas allocation

The DMABAR register is used to set the 416 byte area of the control data and vector table of the allocated DMA to the RAM area.

An example of the memory image when the DMABAR register is set to "2000\_0000H" is shown in Figure 21-2.

The 384 bytes of the DMA control data area that are not used by the DMA can be used as RAM.

Figure 21-2: Example of memory image when the DMABAR register is set to "2000\_0000H"



## 21.3.2 Control data allocation

Starting from the start address, follow DMACR<sub>j</sub>, DMBLS<sub>j</sub>, DMACT<sub>j</sub>, DMRLD<sub>j</sub>, DMSAR<sub>j</sub>, DMDAR<sub>j</sub> (  $j=0\sim 23$  ) Registers are assigned control data sequentially.

The start address is set by the DMABAR register, and the low 10 bits are set separately by the vector table assigned by each boot source.

The distribution of control data is shown in Figure 21-3.

Notice:

1. The DMAEN<sub>i0</sub>~DMAEN<sub>i7</sub> bits of the corresponding DMAEN<sub>i</sub> (  $i=0\sim 2$  ) must be "0" (No Boot) when changing DMCR<sub>j</sub>, DMBLS<sub>j</sub>, DMACT<sub>j</sub>, DMRLD<sub>j</sub>, DMSAR<sub>j</sub>, Data from dmdar<sub>j</sub> registers.
2. DMACR<sub>j</sub>, DMBLS<sub>j</sub>, DMACT<sub>j</sub>, DMRLD<sub>j</sub>, DMSAR<sub>j</sub> and DMSAR<sub>j</sub> cannot be transmitted via DMA access to DMDAR<sub>j</sub>

Figure 21-3: Control data allocation (DMABAR is set to 2000\_0000H)

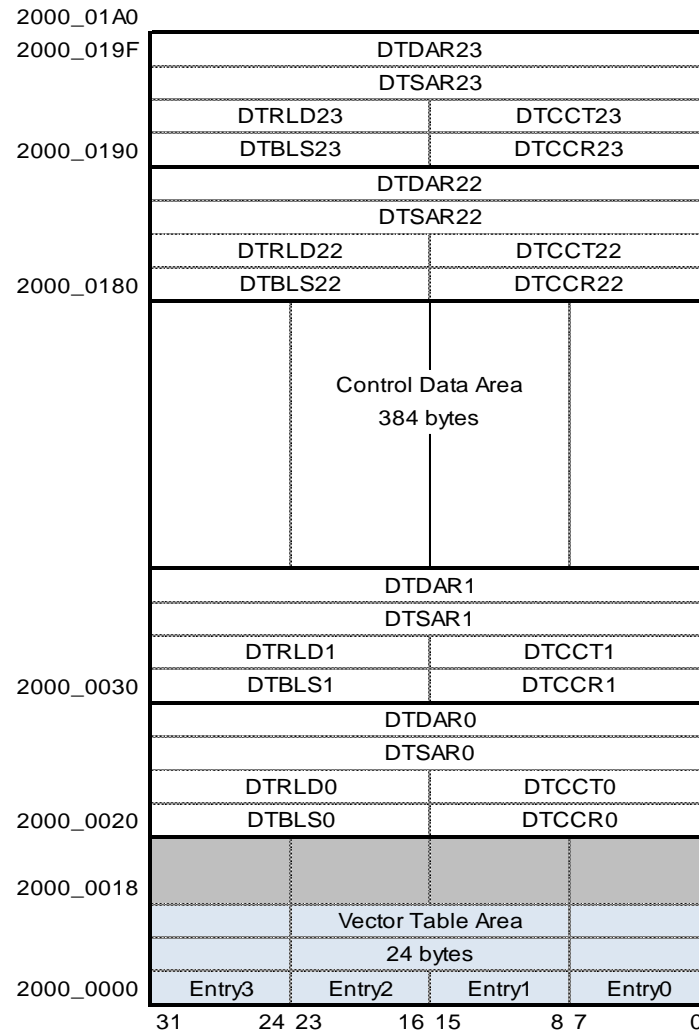




Table 21-4: Starting address of control data

j	Address
23	baseaddr+190H
22	baseaddr+180H
21	baseaddr+170H
20	baseaddr+160H
19	baseaddr+150H
18	baseaddr+140H
17	baseaddr+130H
16	baseaddr+120H
15	baseaddr+110H
14	baseaddr+100H
13	baseaddr+0F0H
12	baseaddr+0E0H
11	baseaddr+0D0H
10	baseaddr+0C0H
9	baseaddr+0B0H
8	baseaddr+0A0H
7	baseaddr+090H
6	baseaddr+080H
5	baseaddr+070H
4	baseaddr+060H
3	baseaddr+050H
2	baseaddr+040H
1	baseaddr+030H
0	baseaddr+020H

Remark: baseaddr: The setting value of the DMABAR registe

### 21.3.3 Vector table

Once the DMA is started, the control data is determined by reading the data from the vector table allocated by each boot source, and the control data is read to be allocated in the DMA control data area.

The DMA boot source and vector addresses are shown in Table 21-5. The vector table of each startup source has 1 byte, saves the data from "00H" to "17H", and selects 1 from 24 sets of control data Group data. The high 22 bits of the vector address are set by the DMABAR register, and the lower 10 bits are assigned "00H" to "17H" corresponding to the startup source.

Notice: DMAENi0~DMAENi7 bits of the corresponding DMAENi (i=0~2) register must be "0" (Disable Startup) when changing the start address of the DMA control data area set in the vector table.

Figure 21-4: Starting address and vector table of control data

When DMABAR register is "2000\_0000H" (example)

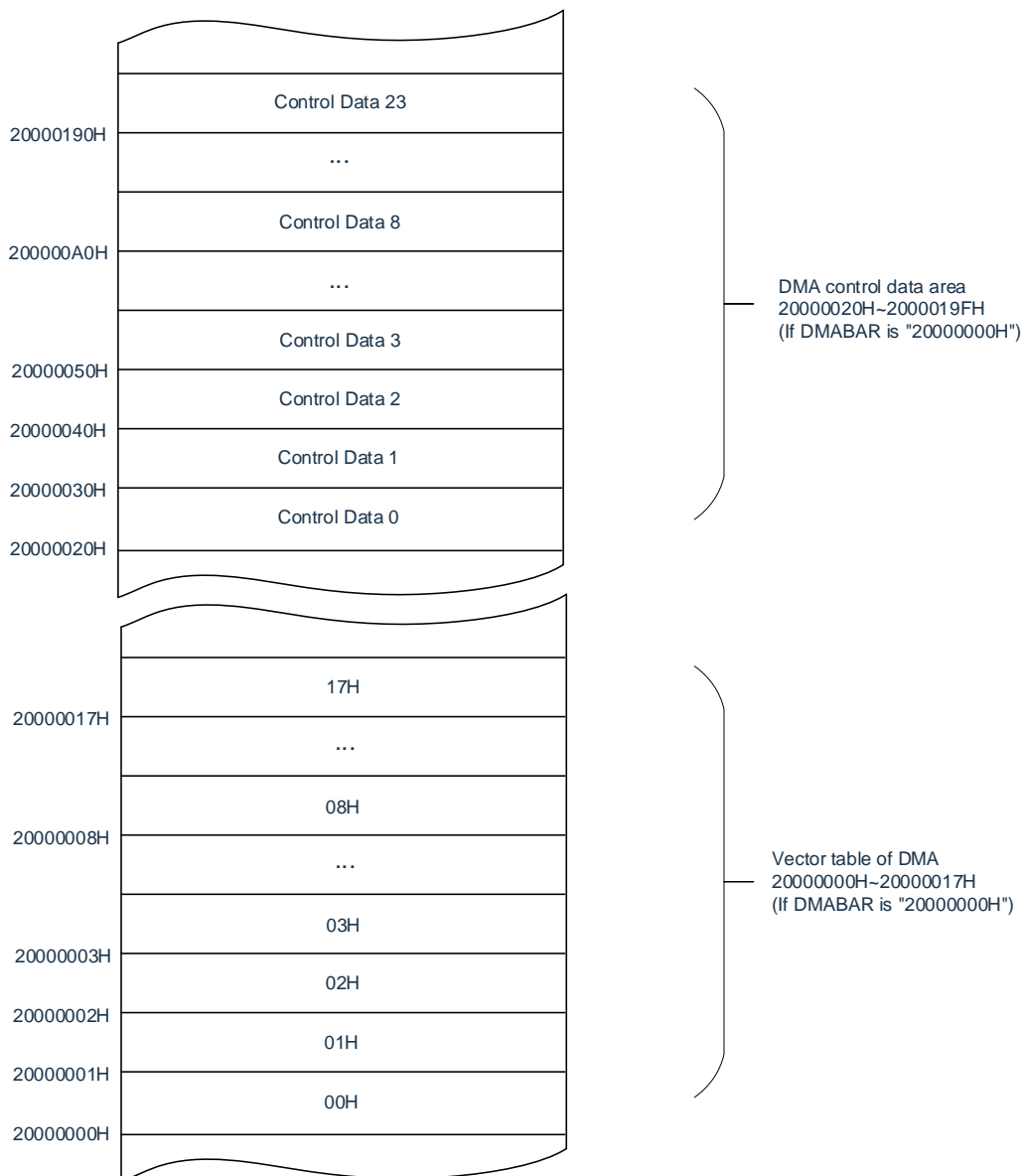


Table 21-5: DMA startup source and vector addresses

DMA start source (interrupt request generation source)	Source number	Vector address	Priority
External interrupt pin input edge detection 0	0	Set address of DMABAR register +00H	<div>High</div> <div>↑</div> <div>↓</div> <div>Low</div>
External interrupt pin input edge detection 1	1	Set address of DMABAR register+01H	
External interrupt pin input edge detection 2	2	Set address of DMABAR register+02H	
External interrupt pin input edge detection 3	3	Set address of DMABAR register+03H	
Key interrupt	4	Set address of DMABAR register+04H	
End of A/D conversion	5	Set address of DMABAR register+05H	
End of transmission of UART0 transmit or buffer null interrupt/end of transmission of SSPI00 or buffer null interrupt/end of transmission of IIC00	6	Set address of DMABAR register+06H	
End of transmission received from UART0/end of transmission from SSPI01 or end of buffer null interrupt/end of transmission from IIC01	7	Set address of DMABAR register+07H	
End of transmission of UART1 send or buffer air-break/end of transmission of SSPI10 or buffer air-break/end of transmission of IIC10	8	Set address of DMABAR register+08H	
End of transmission received by UART1/end of transmission of SSPI11 or buffer air-break/end of transmission of IIC11	9	Set address of DMABAR register+09H	
End of transmission or buffer air-break for UART2 transmit/end of transmission or buffer air-break for SSPI20/end of transmission for IIC20	10	Set address of DMABAR register+0AH	
End of transmission received by UART2/end of transmission of SSPI21 or buffer air-break/end of transmission of IIC21	11	Set address of DMABAR register+0BH	
End of IICA0 communication	12	Set address of DMABAR register+0CH	
End of high-speed SPI communication	13	Set address of DMABAR register+0DH	
End of timer channel 00 count or end of capture	14	Set address of DMABAR register+0EH	
Timer channel 01 end of count or end of capture	15	Set address of DMABAR register+0FH	
Timer channel 02 end of count or end of capture	16	Set address of DMABAR register+10H	
Timer channel 03 end of count or end of capture	17	Set address of DMABAR register+11H	
TimerA underflow	18	Set address of DMABAR register+12H	
RTC fixed cycle signal/alarm clock consistent detection	19	Set address of DMABAR register+13H	
Interval timer interrupt detection	20	Set address of DMABAR register+14H	
Comparator detection 0	21	Set address of DMABAR register+15H	
Comparator detection 1	22	Set address of DMABAR register+16H	
FLASH programming end interrupt	23	Set address of DMABAR register+17H	

### 21.3.4 Peripheral enable register 1 (PER1)

The PER1 register is a register that sets to enable or disable providing clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocks to hardware that is not in use.

To use DMA, bit3 (DMAEN) must be set to "1".

The PER1 register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

For details, see 4.3.7: Peripheral enable register 1(PER1)

## 21.3.5 DMA control register j(DMACRj)(j=0~23)

DMACRj register controls the operation mode of DMA.

Figure 21-5: Format of DMA control register j (DMACRj)

Address: See“21.3.2 Control data allocation”.

After reset: Indefinite value

R/W

Symbol: 15 14 13 12 11 10 9 8

DMACRj

0	0	0	0	0	0	0	FIFO
7	6	5	4	3	2	1	0
SZ		RPTINT	CHNE	DAMOD	SAMOD	RPTSEL	MODE

FIFO	FIFO block transfer control
0	Not a FIFO block transfer
1	is a FIFO block transfer, and the source address (SAMOD=0) or destination address (DAMOD=0) is absolutely fixed

SZ	Selection of transmission data length
00	8-bit
01	16-bit
10	32-bit
11	Settings are disabled.

RPTINT	Allow/disable repeat mode interrupts
0	An interrupt is prohibited.
1	Interrupts are allowed.
When the MODE bit is "0" (normal mode), the setting of the RPTINT bit is invalid.	

CHNE	Allow/Prohibit of chain transfer
0	Chain transfer is prohibited.
1	Chain transfer is allowed.

DAMOD	Control of the delivery destination address
0	fixed
1	Increasing
When the MODE bit is "1" (repeat mode) and the RPTSEL bit is "0" (the transfer target is the repeating region), the SETTING of the DAMOD bit is invalid.	

SAMOD	Control of the delivery source address
0	fixed
1	Increasing
When the MODE bit is "1" (repeat mode) and the RPTSEL bit is "1" (the transmission source is a repeating region), the setting of the SAMOD bit is invalid.	

RPTSEL	Selection of repeating regions
0	The delivery destination is a repeat zone.
1	The transfer source is a repeat zone.
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 pattern

Notice: Access to the DMACRj register cannot be made via DMA transfer.

## 21.3.6 DMA block size register j (DMBLSj)(j=0~23)

This register sets the block size of the data transfer that is started once.

Figure 21-6: Format of DMA block size register j (DMBLSj)

Address: see “21.3.2 Control data allocation”.	After reset: indefinite value							R/W
Symbol:	15	14	13	12	11	10	9	8
DMBLSj	DMBLSj15	DMBLSj14	DMBLSj13	DMBLSj12	DMBLSj11	DMBLSj10	DMBLSj9	DMBLSj8
	7	6	5	4	3	2	1	0
	DMBLSj7	DMBLSj6	DMBLSj5	DMBLSj4	DMBLSj3	DMBLSj2	DMBLSj1	DMBLSj0

DMBLSj	Size of the transfer block		
	8-bit transmission	16-bit transmission	32-bit transmission
00H	Settings are disabled.	Settings are disabled.	Settings are disabled.
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: Access to the DMBLSj register cannot be made via DMA transfer.

## 21.3.7 DMA transmit count register j(DMACTj)(j=0~23)

This register sets the number of data transfers for the DMA. 1 is minus each time DMA transfer is initiated.

Figure 21-7: Format of DMA Transfer Times Register j (DMACTj)

Address: see “21.3.2 Control data allocation”. After reset: Indefinite value								R/W	
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	

DMACTj	Number of transmission
00H	Settings are disabled.
01H	1 time
02H	2 times
03H	3 times
⋮	⋮
FDH	253 times
FEH	254 times
FFH	255 times
⋮	⋮
FFFFH	65535 times

Notice: Access to the DMACTj register cannot be made via DMA transfer.



## 21.3.8 DMA transfer count reload register j (DMRLDj)(j=0~23)

This register sets the initial value of the transfer count register in repeat mode. In repeat mode, since the value of this register is reloaded into the DMACT register, the value must be the same as the initial value of the DMACT register.

Figure 21-8: Format of DMA transfer count reload register j (DMRLDj)

Address: see “21.3.2 Control data allocation”.				After reset: indefinite 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: Access to the DMRLDj register cannot be made via DMA transfer.

## 21.3.9 DMA source address register j(DMSARj)(j=0~23)

This register specifies the delivery source address at the time of data transfer.

When the SZ bit of the DMACRj register is "01" (16 bits transmitted), the lowest bit is ignored and treated as a even address.

When the SZ bit of the DMACRj register is "10" (32-bit transfer), the low 2 bits are ignored and treated as word addresses.

Figure 21-9: Format of DMA source address register j (DMSARj)

Address: see "21.3.2 Control data allocation". After reset: indefinite value								R/W
Symbol	31	30	29	28	27	26	25	24
DMSARj	DMSARj31	DMSARj30	DMSARj29	DMSARj28	DMSARj27	DMSARj26	DMSARj25	DMSARj24
	23	22	21	20	19	18	17	16
	DMSARj23	DMSARj22	DMSARj21	DMSARj20	DMSARj19	DMSARj18	DMSARj17	DMSARj16
	15	14	13	12	11	10	9	8
	DMSARj15	DMSARj14	DMSARj13	DMSARj12	DMSARj11	DMSARj10	DMSARj9	DMSARj8
	7	6	5	4	3	2	1	0
	DMSARj7	DMSARj6	DMSARj5	DMSARj4	DMSARj3	DMSARj2	DMSARj1	DMSARj0

Notice: Access to DMSARj registers cannot be made via DMA transfer.

## 21.3.10 DMA destination address register j(DMDARj)(j=0~23)

This register specifies the destination address at which the data is transferred.

When the SZ bit of the DMACRj register is "01" (16 bits transmitted), the lowest bit is ignored and treated as a even address.

When the SZ bit of the DMACRj register is "10" (32-bit transfer), the low 2 bits are ignored and treated as word addresses.

Figure 21-10: Format of DMA destination address register j(DMDARj)

Address: see "21.3.2 Control data allocation". After reset: indefinite value								R/W	
Symbol	31	30	29	28	27	26	25	24	
DMDARj	DMDARj31	DMDARj30	DMDARj29	DMDARj28	DMDARj27	DMDARj26	DMDARj25	DMDARj24	
	23	22	21	20	19	18	17	16	
	DMDARj23	DMDARj22	DMDARj21	DMDARj20	DMDARj19	DMDARj18	DMDARj17	DMDARj16	
	15	14	13	12	11	10	9	8	
	DMDARj15	DMDARj14	DMDARj13	DMDARj12	DMDARj11	DMDARj10	DMDARj9	DMDARj8	
	7	6	5	4	3	2	1	0	
	DMDARj7	DMDARj6	DMDARj5	DMDARj4	DMDARj3	DMDARj2	DMDARj1	DMDARj0	

Notice: Access to the DMDARj register cannot be made via DMA transfer.

## 21.3.11 DMA start enable register i (DMAENi)(i=0~2)

This is an 8-bit register that controls enable or disable the startup of the DMA through each interrupt source. The interrupt source corresponds to the DMAENi0~DMAENi7 bits as shown in Table 21-6.

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

Notice:

1. The DMAENi0~DMAENi7 bits must be changed where the boot source to the bits is not generated.
2. Access to the DMAENi register cannot be carried out via DMA transmission.
3. The assigned function varies from product to product, and the bits without the assigned function must be set to "0".

Figure 21-11: Format of DMA start 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	DMAENi7	DMAENi6	DMAENi5	DMAENi4	DMAENi3	DMAENi2	DMAENi1	DMAENi0

DMAENi7	DMA start enable i7
0	disable startup.
1	enable startup.
Depending on the condition under which the end-of-transmit interrupt occurs, the DMAENi7 bit becomes "0" (disable)	

DMAENi6	DMA start enable i6
0	disable startup.
1	enable startup.
Depending on the condition under which the end-of-transmit interrupt occurs, the DMAENi6 bit becomes "0" (disable)	

DMAENi5	DMA start enable i5
0	disable startup.
1	enable startup.
Depending on the condition under which the end-of-transmit interrupt occurs, the DMAENi5 bit becomes "0" (disable)	

DMAENi4	DMA start enable i4
0	disable startup.
1	enable startup.
Depending on the condition under which the end-of-transmit interrupt occurs, the DMAENi4 bit becomes "0" (disable)	

DMAENi3	DMA start enable i3
0	disable startup.
1	enable startup.
Depending on the condition under which the end-of-transmit interrupt occurs, the DMAENi3 bit becomes "0" (disable)	

DMAENi2	DMA start enable i2
0	disable startup.
1	enable startup.
Depending on the condition under which the end-of-transmit interrupt occurs, the DMAENi2 bit becomes "0" (disable)	

DMAENi1	DMA start enable i1
0	disable startup.
1	enable startup.
Depending on the condition under which the end-of-transmit interrupt occurs, the DMAENi1 bit becomes "0" (disable)	

DMAENi0	DMA start enable i0
0	disable startup.
1	enable startup.
Depending on the condition under which the end-of-transmit interrupt occurs, the DMAENi0 bit becomes "0" (disable)	

Table 21-6: Interrupt source corresponds to DMAENi0~DMAENi7 bits

Register	DMAENi7 bits	DMAENi6 bit	DMAENi5 bits	DMAENi4 bits	DMAENi3 bits	DMAENi2 bits	DMAENi1 bit	DMAENi0 bits
DMAEN0	End of transmission for UART0 reception/end of transmission for SSPI01 or buffer null interrupt/end of transmission for IIC01	End of transmission of UART0 send or buffer null interrupt/end of transmission of SSPI00 or buffer null interrupt/end of transmission of IIC00	End of A/D conversion	Key interrupt	External interrupt pin input edge detection3	External interrupt pin input edge detection2	External interrupt pin input edge detection1	External interrupt pin input edge detection0
DMAEN1	Timer channel 01 end of count or end of capture	Timer channel 00 end of count or end of capture	End of high-speed SPI communication	End of IICA0 communication	End of transmission for UART2 reception/end of transmission for SSPI21 or buffer null interrupt/end of transmission for IIC21	End of transmission of UART2 transmission or buffer null interrupt/end of transmission of SSPI20 or buffer null interrupt/end of transmission of IIC20	End of transmission for UART1 reception/end of transmission for SSPI11 or buffer null interrupt/end of transmission for IIC11	End of transmission of UART1 send or buffer null interrupt/end of transmission of SSPI10 or buffer null interrupt/end of transmission of IIC10
DMAEN2	FLASH programming end interrupt	Comparator detection1	Comparator detection 0	Interval timer interrupt checkout	RTC fixed-cycle signal/alarm clock consistent detection	TimerA Underflow	Timer channel 03 end of count or end of capture	Timer channel 02 end of count or end of capture

Notice: Bits that are not assigned a function must be set to "0".

Remark: i=0~2

## 21.3.12 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 disable startup.
2. The DMABAR register can only be rewritten once.
3. Access to the DMABAR register cannot be carried out via DMA transmission.
4. For the allocation of DMA control data areas and DMA vector table areas, please refer to “21.3.1 DMA control data areas and DMA vector table areas allocation”.
5. Set the register to keep 1024Byte aligned, that is, set the lower 10 bits to zero. DMA hardware ignores low 10 bits.
6. The register can only be accessed by WORD, IGNORED and HALFWORD access.

Figure 21-12: Format of DMA base address register (DMABAR)

Address: 40005008H		After reset: 00000000H				R/W		
Symbol	31	30	29	28	27	26	25	24
DMABAR	DMABAR31	DMABAR30	DMABAR29	DMABAR28	DMABAR27	DMABAR26	DMABAR25	DMABAR24
	23	22	21	20	19	18	17	16
	DMABAR23	DMABAR22	DMABAR21	DMABAR20	DMABAR19	DMABAR18	DMABAR17	DMABAR16
	15	14	13	12	11	10	9	8
	DMABAR15	DMABAR14	DMABAR13	DMABAR12	DMABAR11	DMABAR10	0	0
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	0

### 21.3.13 DMAENi set register i(DMSETi)(i=0~2)

This is a set register for the DMA start enable register DMAENi, and setting the corresponding bit to 1 will set the corresponding bit of DMAENi to 1.

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

Figure 21-13: Format of DMAENi set register i(DMSETi)(i=0~2)

Address: 40005018H(DMSET0), 40005019H(DMSET1), 4000501AH(DMSET2) After reset: 00H W

Symbol	7	6	5	4	3	2	1	0
DMSETi	DMSETi7	DMSETi6	DMSETi5	DMSETi4	DMSETi3	DMSETi2	DMSETi1	DMSETi0

DMSETin	DMAENin set bit
0	No operation.
1	Set DMAENin bit to 1.

Note: i=0~2, n=0~7

### 21.3.14 DMAENi reset register i(DMCLRi)(i=0~2)

This is a reset register for the DMA start enable register DMAENi. Setting the corresponding bit to 1 will reset the corresponding bit of DMAENi to 0.

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

Figure 21-14: Format of DMAENi reset register i(DMCLRi)(i=0~2)

Address: 40005020H(DMCLR0), 40005021H(DMCLR1), 40005022H(DMCLR2) After reset: 00H W

Symbol	7	6	5	4	3	2	1	0
DMCLRi	DMCLRi7	DMCLRi6	DMCLRi5	DMCLRi4	DMCLRi3	DMCLRi2	DMCLRi1	DMCLRi0

DMCLRin	Clear DMAENin bit
0	No operation.
1	Set DMAENin bit to 0.

Note: i=0~2, n=0~7

## 21.4 Operation of DMA

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 40 sets of control data to the DMA control data area and transmit 24 sets of data. The transmission modes have normal mode and repeat mode, and the transmission size has 8-bit transmission, 16-bit transmission and 32-bit transmission. Passes 1 when the CHNE bit of the DMCRj (j=0~23) register is "1" (allow chain transfer is allowed). A boot source reads multiple control data for continuous data transfer (chain transfer).

The 32-bit DMSARj register and the 32-bit DMDARj register specify the transmit source and destination addresses, respectively. After data transfer, increment or fix the values of the DMSARj register and the DMDARj register according to the control data.

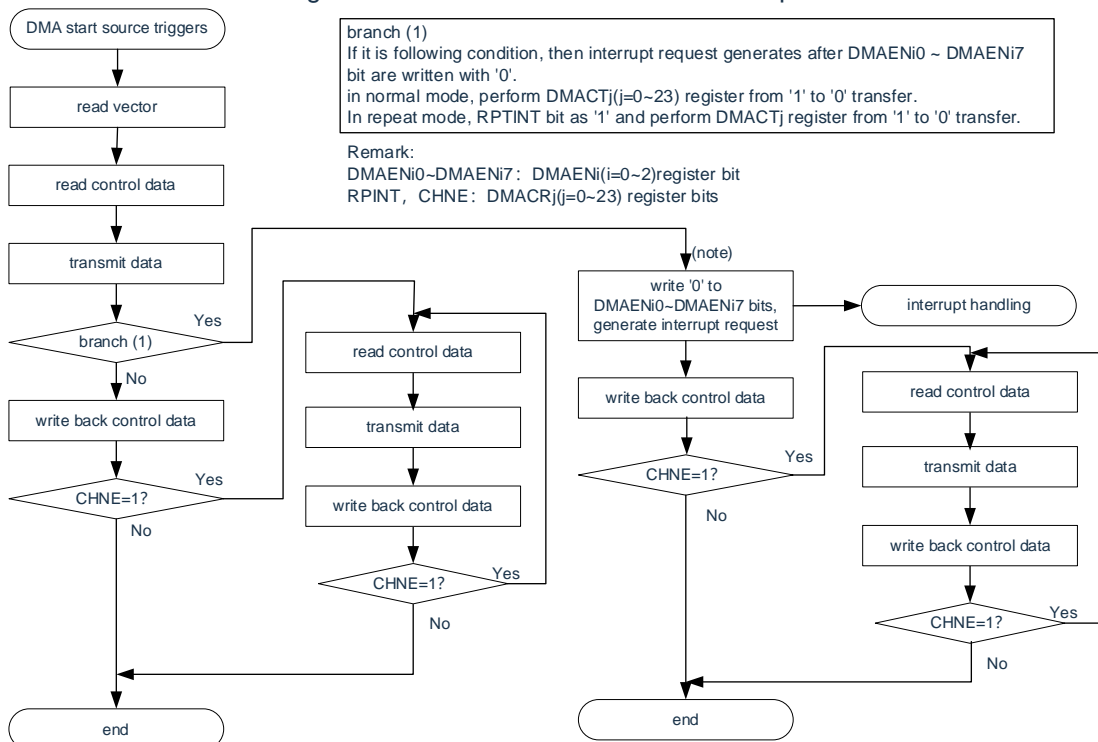
### 21.4.1 Startup Source

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" (interrupt is allowed) and the DMACTj register is changed to "0" for transmission.

The internal flowchart of the DMA is shown in Figure 21-15.

Figure 21-15: Flowchart of DMA internal operation



Note: In data transfer initiated through the allow chain transfer (CHNE=1) setting, no "0" is written to DMAENi0~DMAENi7 bits and no interrupt request is generated.



## 21.4.2 Normal mode

In 8-bit transmission, the transmission data of one boot is 1 to 65535 bytes; In 16-bit transmission, the transmission data initiated once is 2~131070 bytes; In 32-bit transfers, the transmission data for one boot is 4 to 262140 bytes. The number of transmissions is 1 to 65535 times. If you do a data transfer where the DMATj (j=0~23) register becomes "0", it is in the DMA During operation, an interrupt request corresponding to the startup source is generated to the interrupt controller, and the DMAENi0~DMAENi7 of the corresponding DMAENi (i=0~2) register is generated Position "0" (disable startup).

The register function and data transfer in normal mode are shown in Table 21-7 and Figure 21-16.

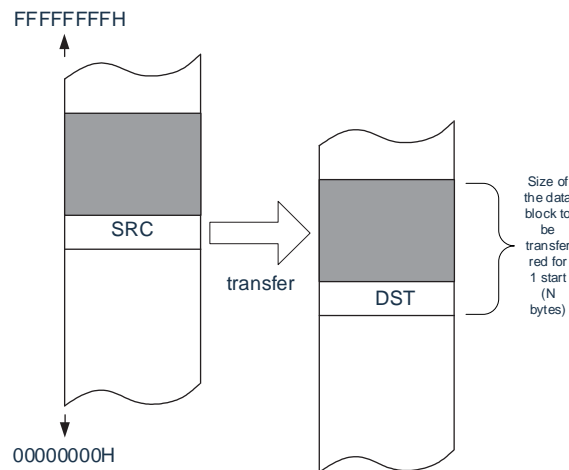
Table 21-7: Register function in normal mode

Register Name	Symbol	Function
DMA block size register j	DMBLSj	The size of the data block to be transferred by 1 boot
DMA transmit times register j	DMACTj	The number of times the data was transferred
DMA transfer count reload register j	DMRLDj	Not used <sup>Note</sup>
DMA source address register j	DMSARj	The address of the source from which the data is transmitted
DMA destination address register j	DMDARj	Destination address for data transmission

Note: Initialization (00H) is required when parity error reset (RPERDIS=0) is allowed to be generated by the RAM parity error detection function.

Remark: j=0~23

Figure 21-16: Data transfer in normal mode



Settings for 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	0	X	0	fixed	fixed	SRC	DST
0	1	X	0	Incremental	fixed	SRC+N	DST
1	0	X	0	fixed	Incremental	SRC	DST+N
1	1	X	0	Incremental	Incremental	SRC+N	DST+N

Remark: DMBLSj register=N

DMSARj register =SRC

DMDARj register =DST

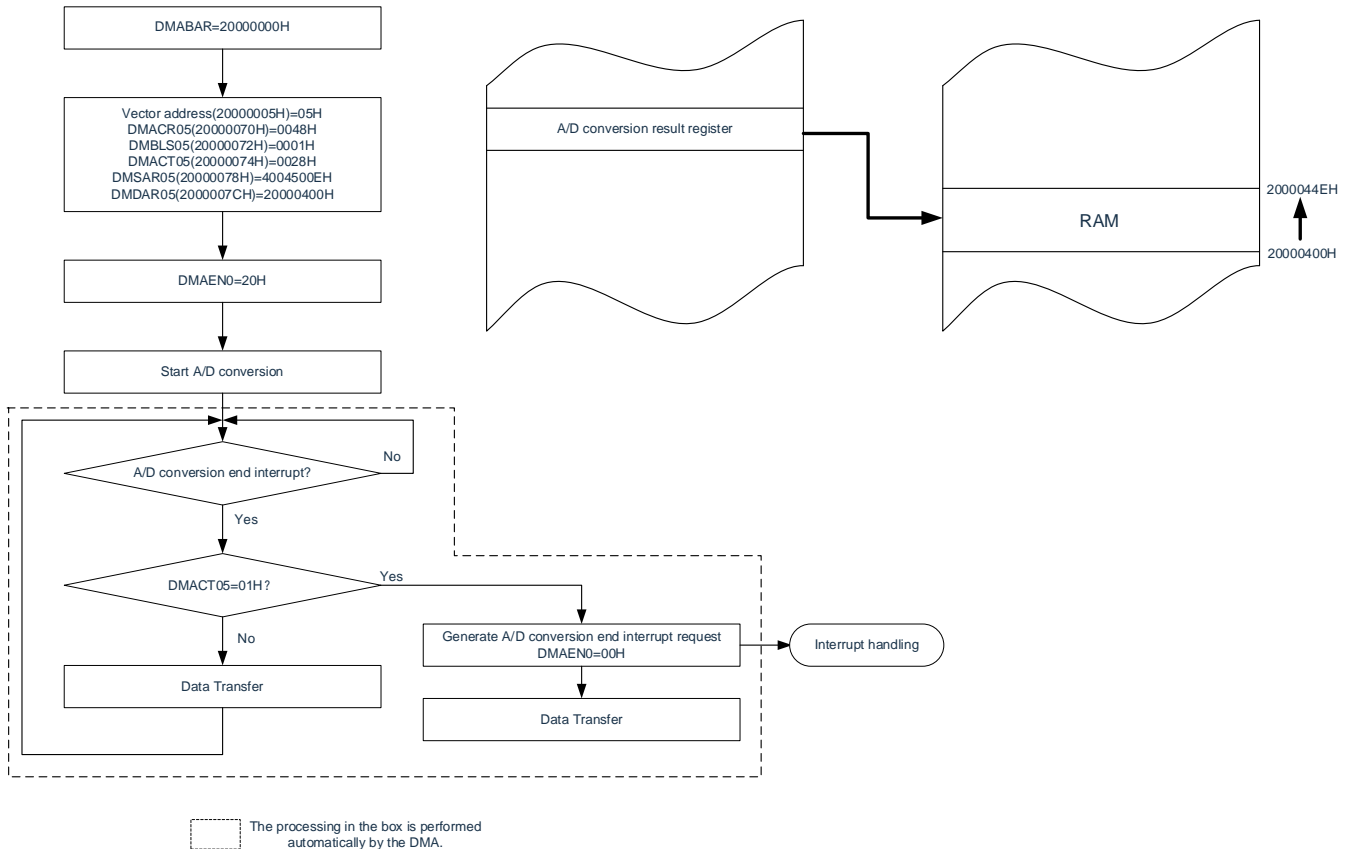
X: "0" or "1"

### (1) Example 1 of the use of normal mode: Continuous A/D conversion results

The DMA CH5 is started by the A/D conversion end interrupt, and the value of the A/D conversion result register is transferred to RAM.

- The vector address is assigned to 2000\_0005H, and the control data is assigned to 2000\_0070H~2000\_007FH.
- The 2 bytes of A/D conversion result register (4004\_500EH, 4004\_500FH) are transferred 40 times to 80 bytes from 2000\_0400H to 2000\_044FH of RAM.

Figure 21-17: Example of using normal mode 1: Continuous A/D conversion results



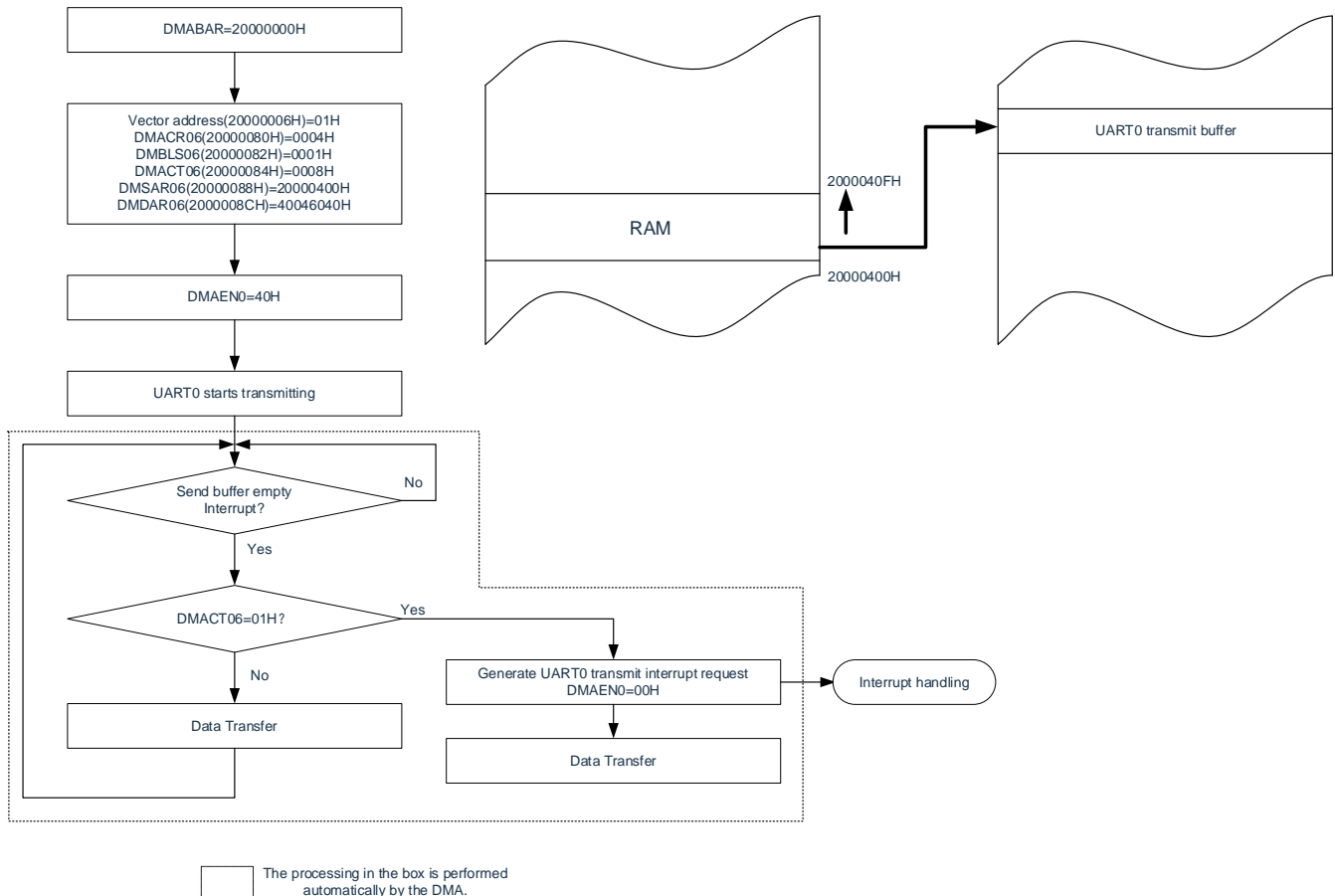
Because it is normal mode, the value of the DMRLD05 register is not used. However, when parity error reset (RPERDIS=0) is allowed to occur via the RAM parity error detection function, the DMRLD05 register must be initialized (0000H).

## (2) Example 2 of the use of normal mode: UART0 transmits continuously

The DMA CH6 is started through the UART0 transmit buffer null interrupt, and the RAM value is transferred to the UART0 transmit buffer.

- The vector address is allocated in 2000\_0006H, and the control data is allocated in 2000080H~2000008FH.
- Transfer 8 bytes from 2000\_0400H to 2000\_0407H of RAM to UART0's transmit buffer (4004\_6040H).

Figure 21-18: Example of normal mode usage 2: UART0 continuous transmission



Because it is normal mode, the value of the DMRLD06 register is not used. However, when parity error reset (RPERDIS=0) is allowed to occur via the RAM parity error detection function, the DMRLD06 register must be initialized (0000H).

The 1st UART0 transmission must be started via the software. The DMA is initiated via the transmit buffer null interrupt, and then the second subsequent send is automatically made.

### 21.4.3 Repeat pattern

The transmission data for 1 boot is 1 to 65535 bytes. Specify the transmission source or transmission destination as the repeating area, and the number of transmissions is 1 to 65535 times. Once the specified number of transfers has ended, the DMCTj (j=0~23) register and the address designated as the repeat region are initialized and then repeated. When the RPTINT bit of the DMACRj register is "1" (interrupt is allowed) and the data transfer of the DMACTj register becomes "0", the interrupt request of the corresponding start source is generated to the interrupt controller during the DMA operation and the corresponding DMAENi (i=0~2) register is set to "0" (disable start). When the RPTINT bit of DMACRj register is "0" (interrupt is prohibited), no interrupt request is generated even if the data transfer of DMACTj register becomes "0", and the DMAENi0~DMAENi7 bits remain unchanged to "0".

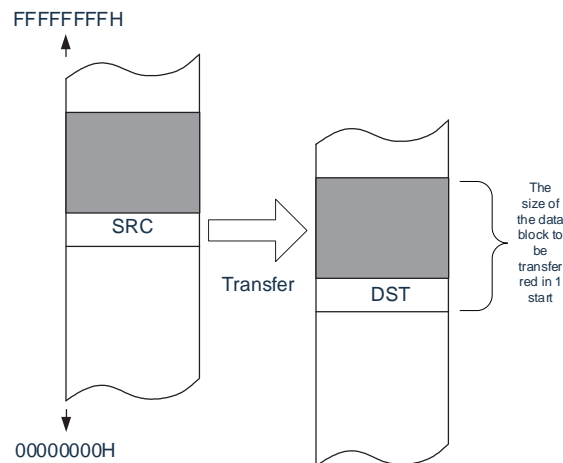
The repeating mode register function and data transfer are shown in Table 21-8 and Figure 21-19.

Table 21-8: Register functions for repeat mode

Register Name	Symbol	Function
DMA block size register j	DMBLSj	The size of the data block to be transferred by 1 boot
DMA transmit times register j	DMACTj	The number of times the data was transferred
DMA transfer count reload register j	DMRLDj	Reload the value of this register to the DMACT register. (Initialize the number of data transfers)
DMA source address register j	DMSARj	The address of the source from which the data is transmitted
DMA destination address register j	DMDARj	Destination address for data transmission

Remark: j=0~23

Figure 21-19: Data transfer in normal mode



Settings for 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+N	DST
1	X	1	1	repeat area	Incremental	SRC+N	DST+N
X	0	0	1	fixed	repeat area	SRC	DST+N
X	1	0	1	Incremental	repeat area	SRC+N	DST+N

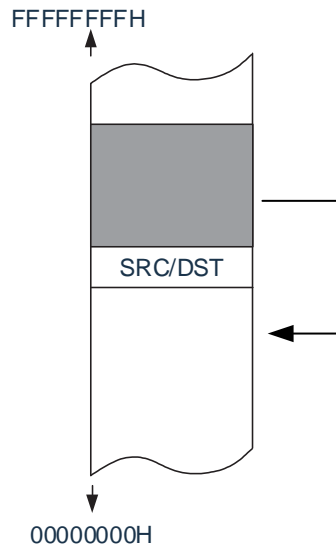
Remark: X: "0" or "1"

DMBLSj register=N

DMACTj register≠1

DMSARj register=SRC

DMDARj register=DST  
j=0~23



Settings for 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	Incremental	SRC	DST+N
X	0	0	1	fixed	repeat area	SRC	DST
X	1	0	1	Incremental	repeat area	SRC+N	DST

Remark: X: "0" or "1"

DMBLSj register=N

DMACTj register=1

DMSARj register=SRC

DMDARj register=DST

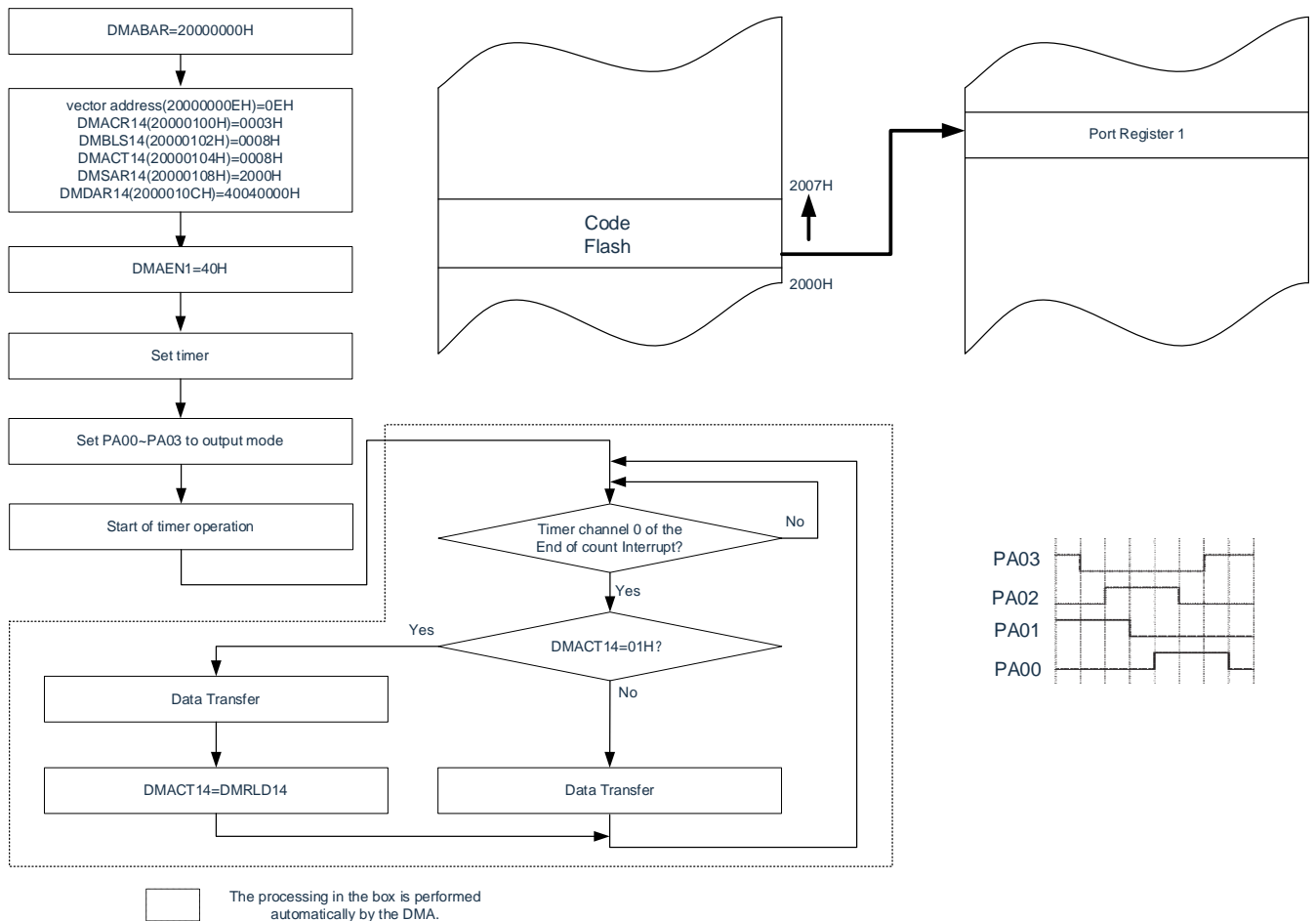
j=0~23

Notice: When using the repeat pattern, the data length of the repeat area must be set to within 65535 bytes.

(1) Example of the use of repeat mode: Use the stepper motor of the port to control the pulse output  
The DMA CH14 is started using Timer8's channel 0 interval timer function, and the pattern of motor control pulses stored in the code flash is transferred to the general purpose port.

- The vector address is assigned to 2000\_000EH, and the control data is assigned to 2000\_0100H~2000\_010FH.
- Transfer 8 bytes from 02000H to 02007H of the code flash to port register A (4004\_0000H).
- Disable repeat mode interrupt.

Figure 21-20: Repeat mode example 1: Using the stepper motor of the port to control the pulse output



Notice: To stop the output, bit6 of DMAEN1 must be cleared after stopping the operation of the timer

## 21.4.4 Chain transmission

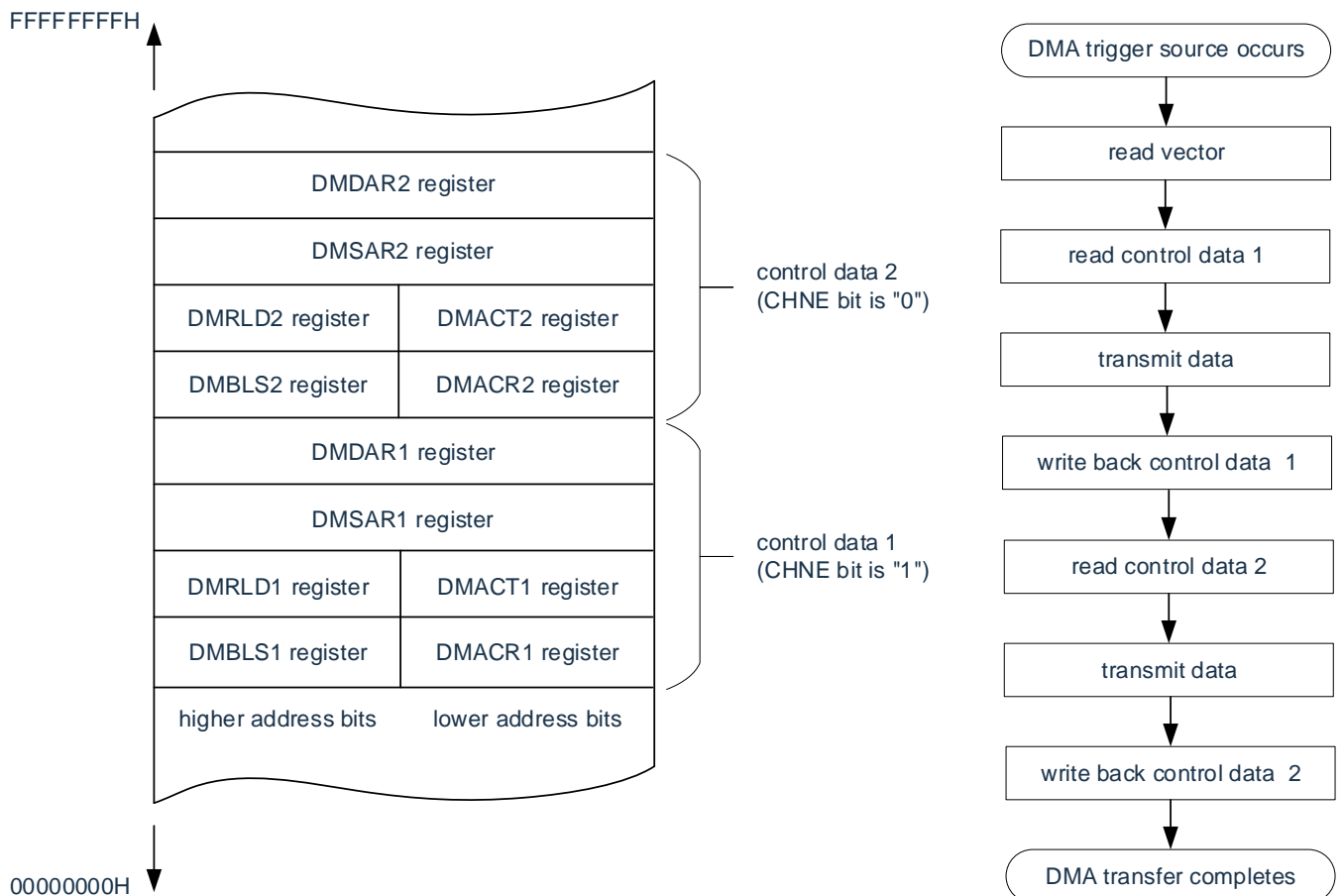
When the DMACRj (j=0~23) register has a CHNE bit of "1" (allow chain transfer), multiple data can be transferred continuously through one start source.

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 is allocated in the DMA control data area. If the CHNE bit of the read control data is "1" (allow chain transfer), the next assigned control data is read at the end of the transfer and the transfer continues. Repeat until the control data transfer with the CHNE bit "0" (disable chain transfer) has ended.

When using multiple control data for chain transfer, the number of transfers of the first control data setting is valid, while the number of transmissions of control data processed later by the second is invalid.

The flowchart of the chain transfer is shown in Figure 21-21.

Figure 21-21: Flowchat of chain transmission



Notice:

- The CHNE bit of the DMACR23 register must be set to "0" (chain transfer is prohibited).
- The DMAENi0~DMAENi7 of the DMAENi (i=0~2) register after the second data chain transmission The bit does not change to "0" (disables DMA startup) and does not generate interrupt requests.

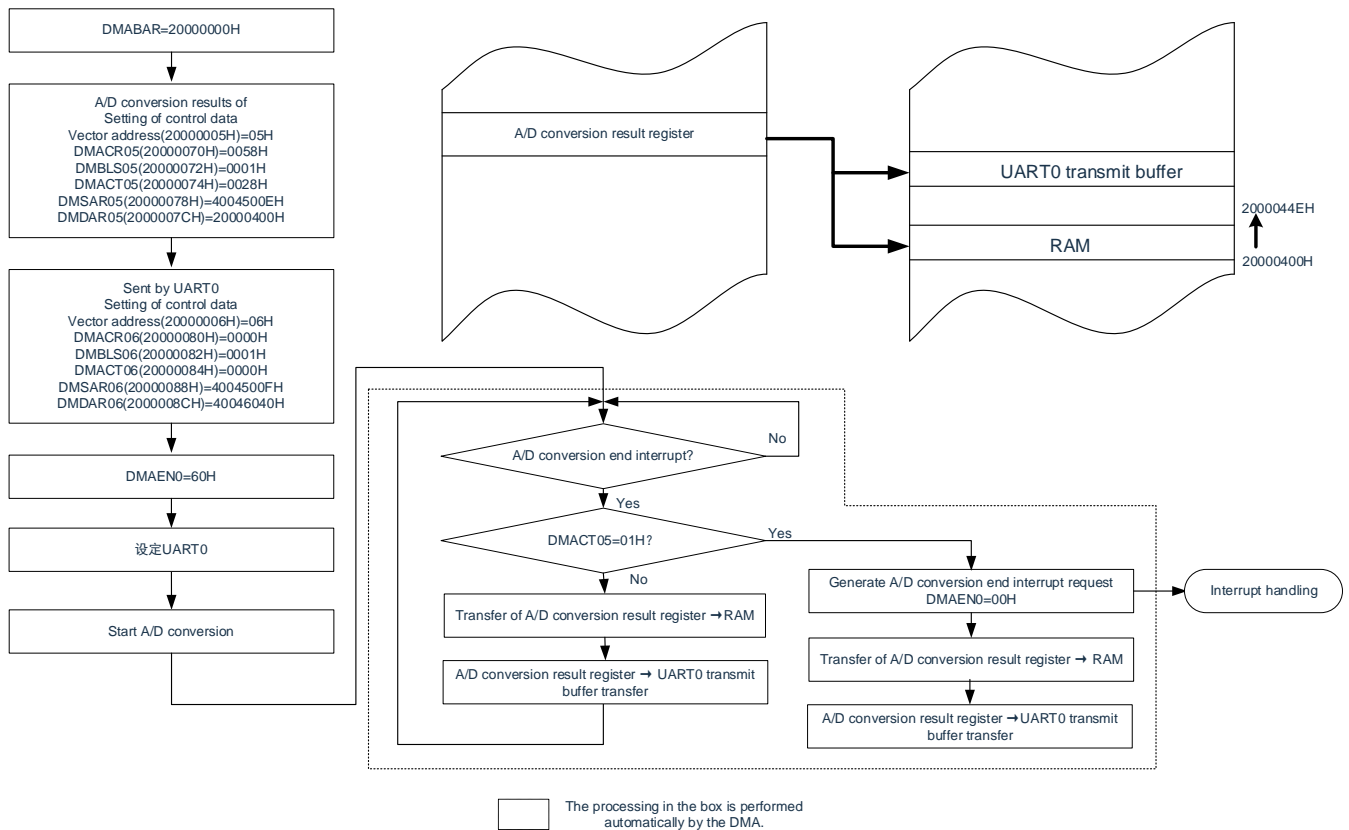
(1) Example of using chain transfer: Continuous A/D conversion result for UART0 transmission

The DMA CH5 and CH6 are started by the A/D conversion end interrupt, and the A/D conversion result is transferred to RAM for UART0 transmission.

The vector addresses are 2000\_0005H and 2000\_0006H respectively.

- The control data of A/D conversion result is allocated in 2000\_0070H~2000\_007FH.
- The control data sent by UART0 is allocated in 2000\_0080H~2000\_008FH.
- Transfer 2 bytes of data from A/D conversion result register (4004500EH, 4004500FH) to RAM 20000400H~2000044FH, and transfer high 1 byte of A/D conversion result register (4004500FH) to UART0 transmit buffer (40046040H).

Figure 21-22: Example of chain transfer: Continuous A/D conversion results are used for UART0 transmission





## 21.5 Cautions when using DMA

### 21.5.1 DMA control data and vector table settings

- The DMA base address register (DMABAR) must be changed with all DMA start sources set to disabled.
- DMA base address register (DMABAR) can only be rewritten once.
- It must be "0" (prohibited) at the DMAENi0~DMAENi7 bits of the corresponding DMAENi (i=0~2) register DMA Startup) when changing DMCRj, DMBLSj, DMACTj, DMRLDj, DMSARj, the data of the DMDARj register.
- It must be "0" (prohibited) at the DMAENi0~DMAENi7 bits of the corresponding DMAENi (i=0~2) register DMA Startup) when you change the start address of the DMA control data area that is set in the vector table.

### 21.5.2 Allocation of DMA control data area and DMA vector table area

The regions where DMA control data and vector tables can be assigned vary depending on the product and conditions of use.

- Stack area, DMA control data area, and DMA vector table area cannot overlap.
- When parity error reset (RPERDIS=0) is allowed to be generated through the RAM parity error detection function, the DMRLD register must be initialized (0000H) even when using normal mode.

## 21.5.3 Number of execution clocks for DMA

The execution of the DMA at startup and the number of clocks required are shown in Table 21-9.

Table 21-9: Execution and number of clocks required when DMA is started

Read vector	Control data		Read data	Write data
	Read	Write back		
1	4	Note1	Note2	Note2

Note 1: For the number of clocks required to write back control data, please refer to “Table 21-10:

Note 2: For the number of clocks required to read and write data, please refer to “Table21-11:

Table 21-10: Number of clocks required to write back control data

Settings for the DMACR register				Address setting		Control writeback of register				Number of clocks
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	Incremental	fixed	write back	write back	write back	No write back	2
1	0	X	0	fixed	Incremental	write back	write back	No write back	write back	2
1	1	X	0	Incremental	Incremental	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		Incremental	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	Incremental		write back	write back	write back	write back	3

Remark: j=0~39, X: “0” or “1”

Table21-11: Number of clocks required to read and write data

Execution status	RAM	Code flash	Data flash	Special function register (SFR)	Extended Special Function Register (2ndSFR).	
					No waiting	wait
Read data	1	2	4	1	1	1+wait number <sup>Note</sup>
Write data	1	-	-	1	1	1+ wait number <sup>Note</sup>

## 21.5.4 Response time of DMA

The DMA response time is shown in Table 21-12. DMA response time is the time from the time when the DMA initiates the source when the DMA is detected and the DMA transmission begins, excluding the number of execution clocks for the DMA.

Table 21-12: Response times 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.

- Maximum response time in case of executing instructions from internal RAM: 20 clocks

Remark: 1 clock:  $1/F_{CLK}$  ( $F_{CLK}$ : CPU/ peripheral hardware clock)

## 21.5.5 Startup source of DMA

- You cannot enter the same startup source between entering the DMA boot source and ending the DMA transfer.
- The DMA boot allow bit corresponding to that boot source cannot be manipulated at the location where the DMA boot source is generated.
- If the DMA boot source sends a competition, the CPU determines the priority when it accepts the DMA transfer and decides to start the boot source. For the priority of the startup source, refer to “21.3.3 Vector table”.
- If DMA startup is allowed in one of the following states, DMA transmission begins and an interruption occurs after the transfer ends. Therefore, it is necessary to allow DMA to start after the monitor flag (CnMON) of the acknowledging comparator.
  - Set to generate interrupt requests through the one-edge detection of the comparator <sup>Note</sup> (CnEDG=0) and interrupt requests through the rising edge of the comparator (CnEPO=0) and  $IVCMP > IVREF$  (or internal reference voltage of 1.45V).
  - Set to generate interrupt request by single edge detection of comparator (CnEDG=0) and generate interrupt request by falling edge of comparator (CnEPO=1) and  $IVCMP < IVREF$  (or internal reference voltage 1.45V).

Remark: (n=0, 1)

## 21.5.6 Operation in standby mode

State	DMA operation
Sleep mode	Capable of operation (disable operation in low-power RTC mode).
Deep sleep mode	Can accept DMA start source, and perform DMA transfer <sup>Note 1</sup>

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 the transfer source.

# Chapter 22 Linkage Controller (EVENTC)

## 22.1 Function of EVENTC

EVENTC links the events output by each peripheral function to each other between the peripheral functions. It can be connected through event chaining without going through the CPU and directly perform collaborative operation between peripheral functions.

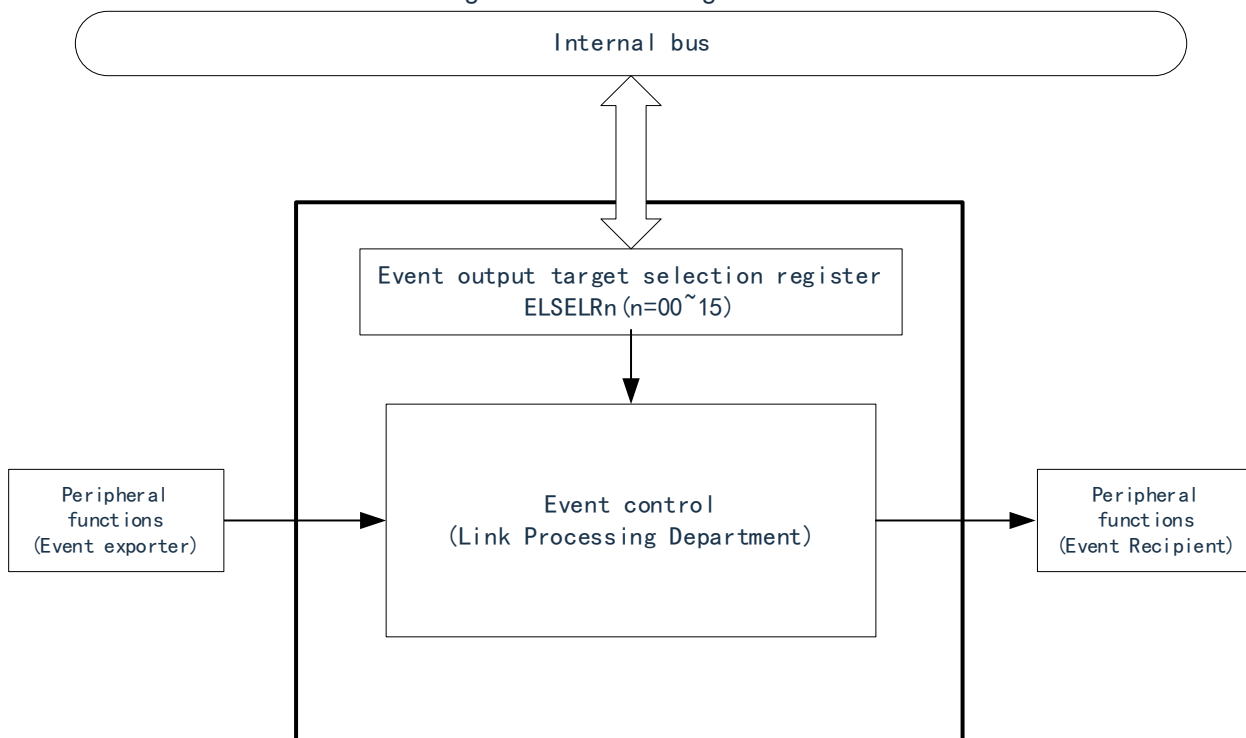
EVENTC has the following features:

- Depending on the product, event signals for 16 peripheral functions can be linked directly to specified peripheral functions.
- Depending on the product, the event signal can be used as the starting source for the operation of 1 of the 6 peripheral functions.

## 22.2 Structure of EVENTC

Block diagram of EVENTC is shown in Figure 22-1.

Figure 22-1: Block diagram of EVENTC



## 22.3 Control register

The control registers are shown in Table 22-1.

Table 22-1: 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
Event output target selection register 15	ELSELR15

## 22.3.1 Output target selection register n(ELSELRn)(n=00~15)

The ELSELRn register links each event signal to the event receiver peripheral function (link target peripheral function) that runs when an event is accepted. You cannot link multiple event inputs to the same event output destination (event receiver). Otherwise, the event receiver peripheral function may operate indefinitely and may not accept the event signal properly. Also, you cannot set the event link occurrence source and event output destination to the same functionality.

The ELSELRn register must be set during periods when the peripheral functions of all event outputs do not generate an event signal.

The correspondence between the ELSELRn register (n=00~22) and the peripheral functions is Table 22-2, ANDLRn register (n= setting value of 00~2) and the corresponding operation of the link target peripheral function when it accepts an event are shown in Table 22-3.

Figure 22-2: Format of event output target selection register n (ELSELRn)

Address: 40041800H(ELSELR00)~4004180FH(ELSELR15)

After reset: 00H

R/W

Symbol	7	6	5	4	3	2	1	0
ELSELRn	0	0	0	0	ELSELn3	ELSELn2	ELSELn1	ELSELn0

ELSELn3 <sup>Note 1</sup>	ELSELn2	ELSELn1	ELSELn0	Selection of event links
0	0	0	0	Disable event links.
0	0	0	1	Selection of the linked peripheral function 1 operation <sup>Note 1</sup> .
0	0	1	0	Selection of the linked peripheral function 2 operation <sup>Note 1</sup> .
0	0	1	1	Selection of the linked peripheral function 3 operation <sup>Note 1</sup> .
0	1	0	0	Selection of the linked peripheral function 4 operation <sup>Note 1</sup> .
0	1	0	1	Selection of the linked peripheral function 5 operation <sup>Note 1</sup> .
0	1	1	0	Selection of the linked peripheral function 6 operation <sup>Note 1</sup> .
Others:				Settings are disabled.

Note 1: Refer to "Table 22-3: ".

Table 22-2: Correspondence of ELSELRn registers (n=00~15) and peripheral functions

Register Name	Event occurrence source (output source for event input n).	Content
ELSELR00	External interrupt edge detection 0	INTP0
ELSELR01	External interrupt edge detection 1	INTP1
ELSELR02	External interrupt edge detection 2	INTP2
ELSELR03	External interrupt edge detection 3	INTP3
ELSELR04	External interrupt edge detection 4	INTP4
ELSELR05	External interrupt edge detection 5	INTP5
ELSELR06	Key return signal detection	INTKR
ELSELR07	RTC fixed cycle/alarm clock consistency detection	INTRTC
ELSELR08	Timer A underflow	INTTMA
ELSELR09	End of count/end of capture for Timer8 channel 00	INTTM00
ELSELR10	Timer8 end-of-count/end-of-capture for channel 01	INTTM01
ELSELR11	End of count/end of capture for Timer8 channel 02	INTTM02
ELSELR12	End of count/end of capture for Timer8 channel 03	INTTM03
ELSELR13	Comparator detection 0	INTCMP0
ELSELR14	Comparator detection 1	INTCMP1
ELSELR15	Stop detection interrupt	INTOSDC



Table 22-3: Correspondence between the setting value of ELSELRn register (n=00~15) and the operation when the link target peripheral function accepts the event

ELSELRn register ELSELRn3~ELSELRn0 bits	Link target No.	Linking target peripheral functions	Running when accepting events
0001B	1	AD converter	Start of A/D conversion.
0010B	2	Timer input for Timer8 channel 0 <sup>Note 1</sup>	Delay counter, measurement of input pulse interval, external event counter
0011B	3	Timer input for Timer8 channel 1 <sup>Note2</sup>	Delay counter, measurement of input pulse interval, external event counter
0100B	4	TimerA	Counting source
0101B	5	DA0 <sup>Note 3</sup>	Real-time output

Note 1: To select the timer input of Timer8 channel 0 as the link target peripheral function, you must first set the running clock of channel 0 to  $F_{CLK}$  via Timer Clock Select Register 0 (TPS0), set the noise filter of TI00 pin to OFF (TNFEN00=0) via Noise Filter Enable Register 1 (NFEN1), and set the timer input used by channel 0 to the event input signal of the link controller via Timer Input Select Register 0 (TIS0).

Note 2: To select the timer input of Timer8 channel 1 as the link target peripheral function, you must first set the running clock of channel 1 to  $F_{CLK}$  via Timer Clock Select 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).

Note 3: If you want to enter the deep sleep state when the real-time output mode of the D/A conversion is active, you must disable event links for EVENTC before entering deep sleep mode.

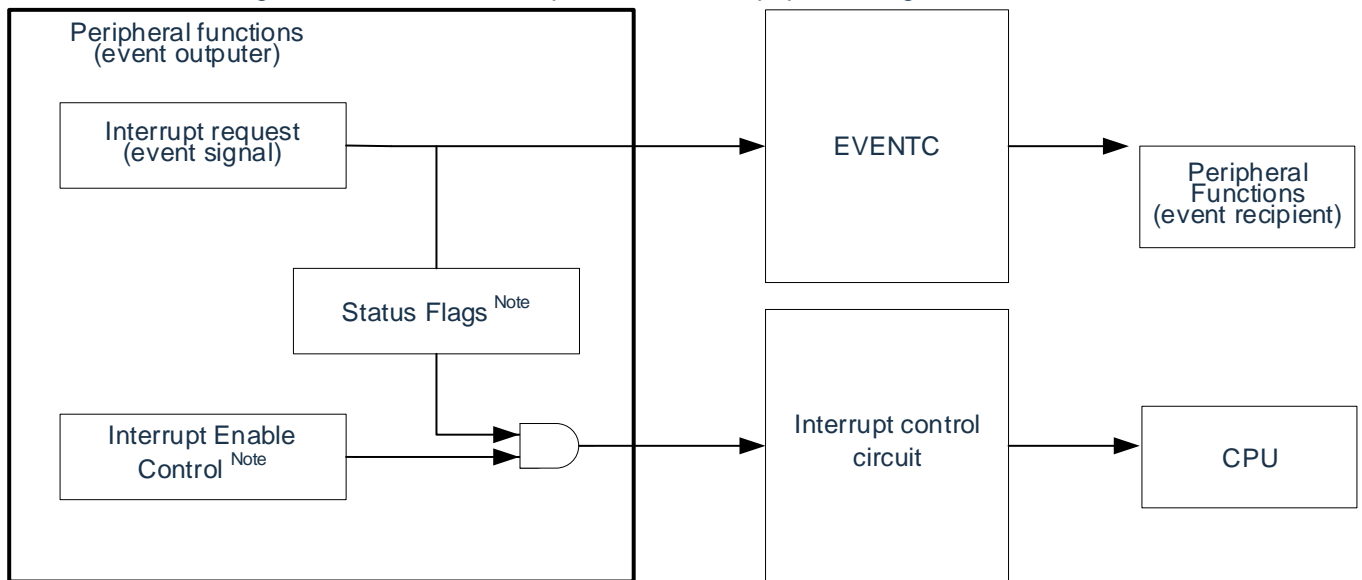
## 22.4 Operation of EVENTC

The paths used by the event signals generated by each peripheral function as interrupt requests for the interrupt control circuit and the paths used as EVENTC events are independent of each other. Therefore, each event signal is independent of the interrupt control and can be used as an event signal for the operation of the peripheral function of the event recipient.

The relationship between interrupt processing and EVENTC is shown in Figure 22-3. This diagram shows the relationship for a peripheral function with an interrupt request status flag and an interrupt enable bit (control enable or disable).

The operation of the peripheral function that receives an event via EVENTC is based on the operation of the recipient peripheral function after receiving the event (refer to “Table 22-3:”).

Figure 22-3: The relationship between interrupt processing and EVENTC



Note: Some peripheral functions do not have this feature.

The responses of the peripheral functions that accept the events are shown in Table 22-4.

Table 22-4: Response of the peripheral function of the received event

Event Acceptance Target No.	Function of event link target	Operation after event acceptance	Response
1	AD converter	A/D conversion	EventC events become hardware-triggered directly for A/D conversions.
2	Timer8 Timer input for channel 0	The delay counter inputs the pulse width of the measured external event counter	Edge detection is performed after 3 or 4 $F_{CLK}$ cycles from the occurrence of an EVENTC event.
3	Timer8 Timer input for channel 1	The delay counter inputs the pulse width of the measured external event counter	Edge detection is performed after 3 or 4 $F_{CLK}$ cycles from the occurrence of an EVENTC event.
4	TimerA	Counting source	EventC events become the counting source for timer A directly.
5	Channel 0 of the D/A converter	Real-time output (channel 0)	The D/A conversion of channel 0 begins after 2 or 3 $F_{CLK}$ cycles from the time of the EVENTC event.

# Chapter 23      Interrupt Function

The Cortex-M0+ processor has a built-in nested vector interrupt controller (NVIC), which supports up to 32 interrupt request (IRQ) inputs and one non-maskable interrupt (NMI) input, in addition to multiple internal exceptions.

In this system, the interrupt sources for 32 interrupt request (IRQ) inputs and one non-maskable interrupt (NMI) input are handled. This user's manual only explains the processing in this system. Please refer to the user's manual of Cortex-M0+ processor for the functions of the built-in NVIC of Cortex-M0+ processor.

## 23.1      Types of interrupt function

There are 2 types of interrupt functions as follows.

(1) Maskable interrupt

This is a mask-controlled interrupt. If the interrupt mask flag register is not opened, the interrupt request will not be responded even if it is generated

It can generate standby release signal, release deep sleep mode, sleep mode.

Maskable interrupts are divided into external interrupt requests and internal interrupt requests.

(2) Non-maskable interrupt

This is an interrupt that is not controlled by masking. Once an interrupt request is generated, the CPU must respond to it.

## 23.2      Interrupt Sources and Structures

Refer to Table 23-1 for a list of interrupt sources.

Table 23-1: List of interrupt sources (1/2)

Interrupt handling	Interrupt source No.	Interrupt source		Internal/External	Basic structure type <sup>Note 1</sup>
		Name	Trigger		
Maskable	0	INTLVI	Voltage detection <sup>Note 2</sup>	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	INTP4	Detection of pin input edges		
	6	INTP5	Detection of pin input edges		
	7	INTKR	Key interrupt		
	8	INTRTC	Fixed period/alarm consistency detection for real time clocks	Internal	(A)
	9	INTIT	Detection of interval signals		
	10	INTST0/ INTCSI00/ INTIIC00	End of transmission of UART0 send or buffer null interrupt/end of transmission of SSPI00 or buffer null interrupt/end of transmission of IIC00		
	11	INTSR0/ INTCSI01/ INTIIC01	End of transmission for UART0 reception/end of transmission for SSPI01 or buffer null interrupt/end of transmission for IIC01		
	12	INTST1/ INTCSI10/ INTIIC10	End of transmission of UART1 transmission or buffer null interrupt/end of transmission of SSPI10 or buffer null interrupt/end of transmission of IIC10		
	13	INTSR1/ INTCSI11/ INTIIC11	End of transmission for UART1 reception/end of transmission for SSPI11 or buffer null interrupt/end of transmission for IIC11		
	14	INTST2/ INTCSI20/ INTIIC20	End of transmission of UART2 transmission or buffer null interrupt/end of transmission of SSPI20 or buffer null interrupt/end of transmission of IIC20		
	15	INTSR2/ INTCSI21/ INTIIC21	End of transmission for UART2 reception/end of transmission for SSPI21 or buffer null interrupt/end of transmission for IIC21		

Note 1: The basic composition types (A) to (C) correspond to Figure 23-1 (A)~(C).

Note 2: This is the case when bit 7 (LVIMD) of the voltage detection level register (LVIS) is set to "0".

Table 23-1 List of interrupt sources (2/2)

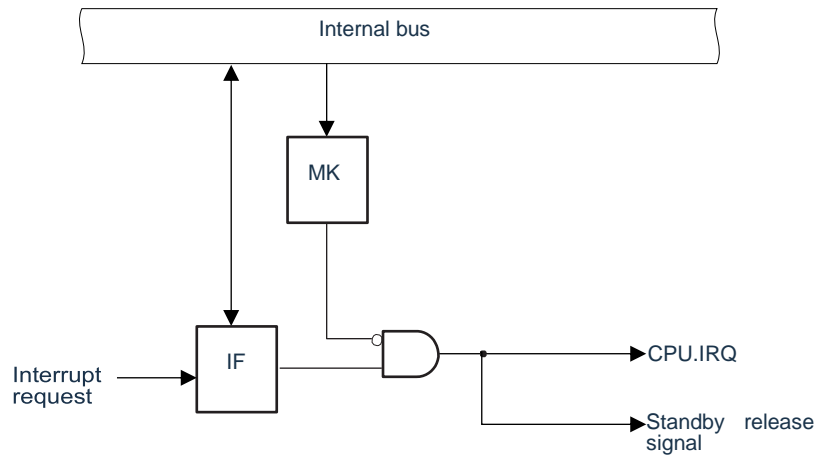
Interrupt handling	Interrupt source No.	Interrupt source		Internal/External	Basic structure type <sup>Note 1</sup>
		Name	Trigger		
Maskable	16	INTIICA0	End of IICA0 communication	Internal	(A)
	17	INTSPI00	End of SPI communication		
	18	INTTM00	End of count or end of capture for timer channel 00		
	19	INTTM01	End of count or end of capture for timer channel 01		
	20	INTTM02	End of count or end of capture for timer channel 02		
	21	INTTM03	End of count or end of capture for timer channel 03		
	22	INTTM04	End of count or end of capture for timer channel 04		
	23	INTTM05	End of count or end of capture for timer channel 05		
	24	INTTM06	End of count or end of capture for timer channel 06		
	25	INTTM07	End of count or end of capture for timer channel 07		
	26	INTTMA0	TimerA underflow		
	27	INTAD	End of AD conversion		
	28	INTCMP0	Comparator detection 0		
	29	INTCMP1	Comparator detection1		
	30	INTOSDC	OSDC deactivation detection		
	31	INTFL	End of FLASH programming		
Non-maskable	-	INTWDT	Watchdog Timer Interval Interrupt <sup>Note 2</sup>	Internal	(C)

Note 1: The basic composition types (A) to (C) correspond to Figure 23-1 (A)~(C).

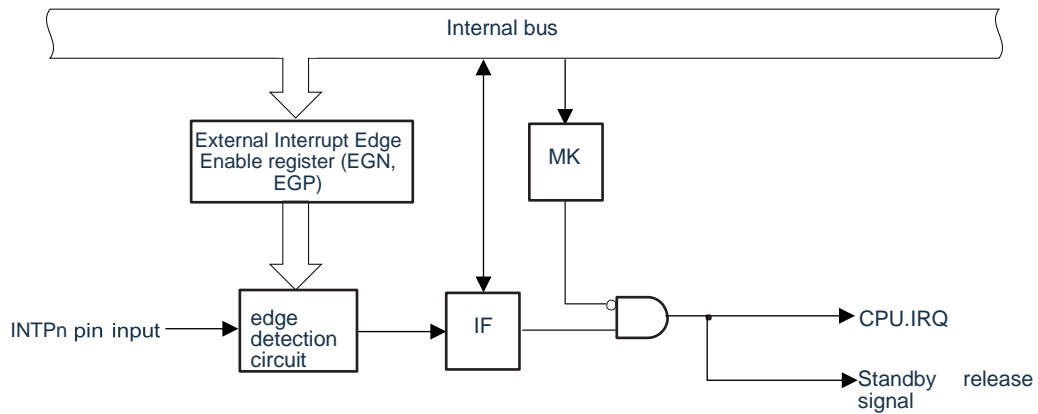
Note 2: This is the case when bit7 (WDTINT) of the option byte (000C0H) is set to "1".

Figure 23-1: Basic structure of interrupt function

### (A) Internal maskable interrupt

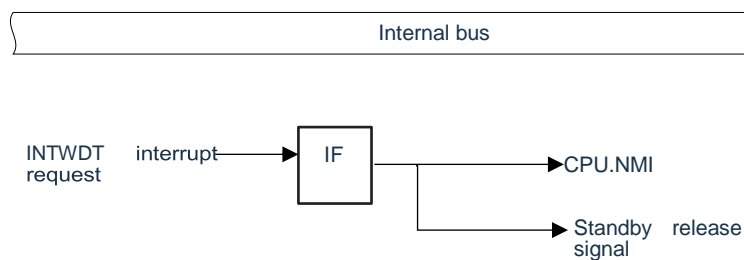


### (B) External Maskable Interrupt (INTPn)



Remark: n=0~5

### (C) Non-maskable interrupt



Note: The interrupt request flag IF of non-maskable interrupts does not have a physical register and cannot be used to generate interrupt requests by reading or writing registers on the bus.

## 23.3 Registers for controlling interrupt function

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)
- 

### 23.3.1 Interrupt request flag register (IF00~IF31)

The interrupt request flag is set to "1" by the occurrence of the corresponding interrupt request or execution instruction.

The interrupt request flag is cleared to "0" by generating a reset signal or by executing an instruction.

Set IF00L~IF31L registers by an 8-bit memory manipulation instruction

Or set IF00~IF31 registers by a 32-bit memory manipulation instruction.

After a reset signal is generated, the value of these registers becomes "0000\_0000H".

Figure 23-2: Format of interrupt request flag register (IFm) (m=0~31)

Address: IF00: 40006000H, IF01: 40006004H, IF02: 40006008H, IF03: 4000600CH  
 IF04: 40006010H, IF05: 40006014H, IF06: 40006018H, IF07: 4000601CH  
 IF08: 40006020H, IF09: 40006024H, IF10: 40006028H, IF11: 4000602CH  
 IF12: 40006030H, IF13: 40006034H, IF14: 40006038H, IF15: 4000603CH  
 IF16: 40006040H, IF17: 40006044H, IF18: 40006048H, IF19: 4000604CH  
 IF20: 40006050H, IF21: 40006054H, IF22: 40006058H, IF23: 4000605CH  
 IF24: 40006060H, IF25: 40006064H, IF26: 40006068H, IF27: 4000606CH  
 IF28: 40006070H, IF29: 40006074H, IF30: 40006078H, IF31: 4000607CH

Reset value: 0000\_0000H R/W

Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
IFmL	Reserved						IF

IFmL	Interrupt request flag for interrupt sources numbered 0 to 31
0	No interrupt request signal is generated.
1	Generates an interrupt request and is in the interrupt request state.

Note 1: The correspondence between the interrupt source and interrupt request mask register is shown in Table 23-2.

Note 2: The correspondence between the interrupt request flag register and CPU.IRQ is shown in Figure 23-4.



## 23.3.2 Interrupt mask flag register (MK00~MK31)

The interrupt mask flag is set to enable or disable the corresponding maskable interrupt processing.

Set MK00L~MK31L registers by an 8-bit memory manipulation instruction or MK00~MK31 registers by a 32-bit memory manipulation instruction.

After the reset signal is generated, the value of these registers becomes "FFFF\_FFFF".

Figure 23-3: Format of interrupt request mask register (MKm) (m=0~31)

Address: MK00: 40006100H, MK01: 40006104H, MK02: 40006108H, MK03: 4000610CH

MK04: 40006110H, MK05: 40006114H, MK06: 40006118H, MK07: 4000611CH

MK08: 40006120H, MK09: 40006124H, MK10: 40006128H, MK11: 4000612CH

MK12: 40006130H, MK13: 40006134H, MK14: 40006138H, MK15: 4000613CH

MK16: 40006140H, MK17: 40006144H, MK18: 40006148H, MK19: 4000614CH

MK20: 40006150H, MK21: 40006154H, MK22: 40006158H, MK23: 4000615CH

MK24: 40006160H, MK25: 40006164H, MK26: 40006168H, MK27: 4000616CH

MK28: 40006170H, MK29: 40006174H, MK30: 40006178H, MK31: 4000617CH

Reset value: FFFF\_FFFFH R/W

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
MKmL	Reserved						MKL

MKmL	Interrupt processing control for interrupt sources numbered 0 to 31 <sup>Note 1</sup>
0	Enable interrupt processing.
1	Disable interrupt processing.

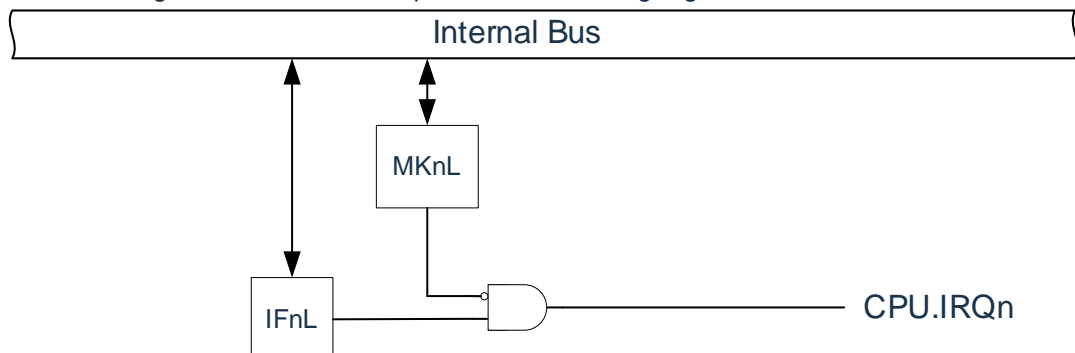
Note 1: The correspondence between the interrupt source and the interrupt request mask register is shown in Table 23-2.

Note 2: The correspondence between the interrupt request mask register and CPU.IRQ is shown in Figure 23-4.

Table 23-2: Correspondence between interrupt sources and flag registers

Number	Interrupt source	Interrupt request flag register	Interrupt Mask Flag Register
0	INTLVI	IF00.IFL	MK00.MKL
1	INTP0	IF01.IFL	MK01.MKL
2	INTP1	IF02.IFL	MK02.MKL
3	INTP2	IF03.IFL	MK03.MKL
4	INTP3	IF04.IFL	MK04.MKL
5	INTP4	IF05.IFL	MK05.MKL
6	INTP5	IF06.IFL	MK06.MKL
7	INTKR	IF07.IFL	MK07.MKL
8	INTRTC	IF08.IFL	MK08.MKL
9	INTIT	IF09.IFL	MK09.MKL
10	INTST0/INTCSI00/INTIIC00	IF10.IFL	MK10.MKL
11	INTSR0/INTCSI01/INTIIC01	IF11.IFL	MK11.MKL
12	INTST1/INTCSI10/INTIIC10	IF12.IFL	MK12.MKL
13	INTSR1/INTCSI11/INTIIC11	IF13.IFL	MK13.MKL
14	INTST2/INTCSI20/INTIIC20	IF14.IFL	MK14.MKL
15	INTSR2/INTCSI21/INTIIC21	IF15.IFL	MK15.MKL
16	INTICA0	IF16.IFL	MK16.MKL
17	INTSPI00	IF17.IFL	MK17.MKL
18	INTTM00	IF18.IFL	MK18.MKL
19	INTTM01	IF19.IFL	MK19.MKL
20	INTTM02	IF20.IFL	MK20.MKL
21	INTTM03	IF21.IFL	MK21.MKL
22	INTTM04	IF22.IFL	MK22.MKL
23	INTTM05	IF23.IFL	MK23.MKL
24	INTTM06	IF24.IFL	MK24.MKL
25	INTTM07	IF25.IFL	MK25.MKL
26	INTTMJ0	IF26.IFL	MK26.MKL
27	INTAD	IF27.IFL	MK27.MKL
28	INTCMP0	IF28.IFL	MK28.MKL
29	INTCMP1	IF29.IFL	MK29.MKL
30	INTOSDC	IF30.IFL	MK30.MKL
31	INTFL	IF31.IFL	MK31.MKL

Figure 23-4: Relationship between each flag register and CPU.IRQ



### 23.3.3 External interrupt rising edge enable register (EGP0), External interrupt falling edge enable register (EGN0)

These registers set the active edges of INTp0 to INTp5.

The EGP0, EGN0 registers are set by 8-bit memory manipulation instruction.

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

Figure 23-5: Format of External interrupt rising edge enable register (EGP0), External interrupt falling edge enable register (EGN0)

Address: 40041C00H			After reset: 00H			R/W		
Symbol	7	6	5	4	3	2	1	0
EGP0	0	0	EGP5	EGP4	EGP3	EGP2	EGP1	EGP0

Address: 40041C01H		After reset: 00H			R/W			
Symbol	7	6	5	4	3	2	1	0
EGN0	0	0	EGN5	EGN4	EGN3	EGN2	EGN1	EGN0

EGPn	EGNn	Effective edge selection of INTpN pin (n=0~5)
0	0	Disable edge detection.
0	1	Falling edge
1	0	Rising edge
1	1	Rising and falling edges

The ports corresponding to the EGPn and EGNn bits are shown in Table 23-3.

Table 23-3: Interrupt request signals corresponding to the EGPn and EGNn bits

Detection enable bits		Interrupt request signal
EGP0	EGN0	INTp0
EGP1	EGN1	INTp1
EGP2	EGN2	INTp2
EGP3	EGN3	INTp3
EGP4	EGN4	INTp4
EGP5	EGN5	INTp5

Notice: If you switch the input port used by the external interrupt function to output mode, an INTpN interrupt may be detected and an INTpN interrupt may be detected. When switching to output mode, the port mode register (PMxx) must be set to "0" after disabling the detection edge (EGPn, EGNn=0, 0).

Remark:

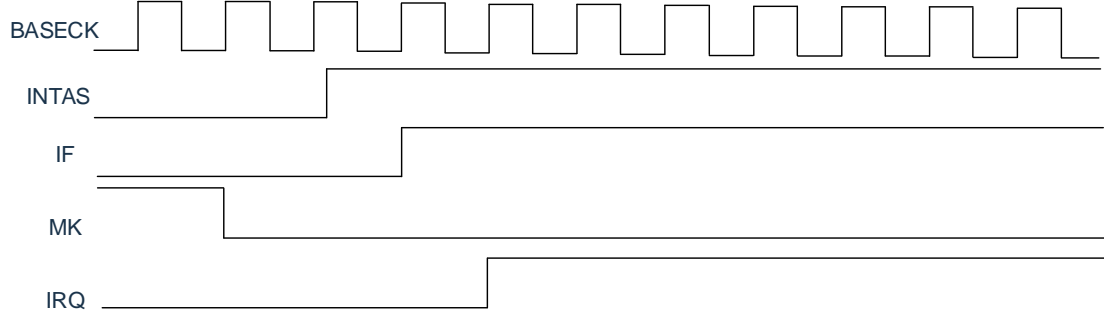
- For ports for edge detection, refer to "Chapter 2 Port function".
- n=0~5

## 23.4 Operation of interrupt handling

### 23.4.1 Acceptance of maskable interrupt requests

If the interrupt request flag is set to "1" and the mask (MK) flag for the interrupt request is cleared to "0", the interrupt request is accepted and can be passed to the NVIC.

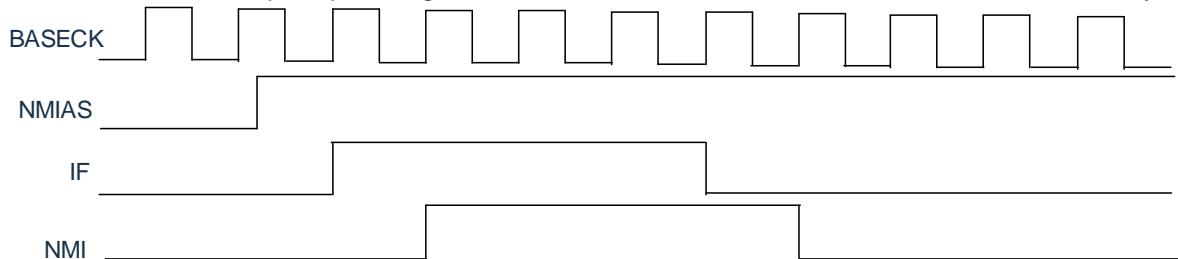
From the time the interrupt request flag is set to 1, to the time the CPU's IRQ is set to 1, it takes only 1 clock.



### 23.4.2 Acceptance of non-maskable interrupt requests

If a non-maskable interrupt request is generated, the interrupt request flag will be set to "1" and passed directly to the NVIC.

From the time the interrupt request flag is set to 1, to the time the CPU's NMI is set to 1, it takes only 1 clock.



## Chapter 24 Key Interrupt Function

The number of channels for key interrupt input varies by product.

### 24.1 Function of key interrupt

A key interrupt (INTKR) can be generated by inputting a falling edge to the key interrupt input pins (KR0~KR7).

Table 24-1: Assignment of key interrupt detection pins

Key interrupt pin	Key return mode register (KRM)
KR0	KRM0
KR1	KRM1
KR2	KRM2
KR3	KRM3
KR4	KRM4
KR5	KRM5
KR6	KRM6
KR7	KRM7

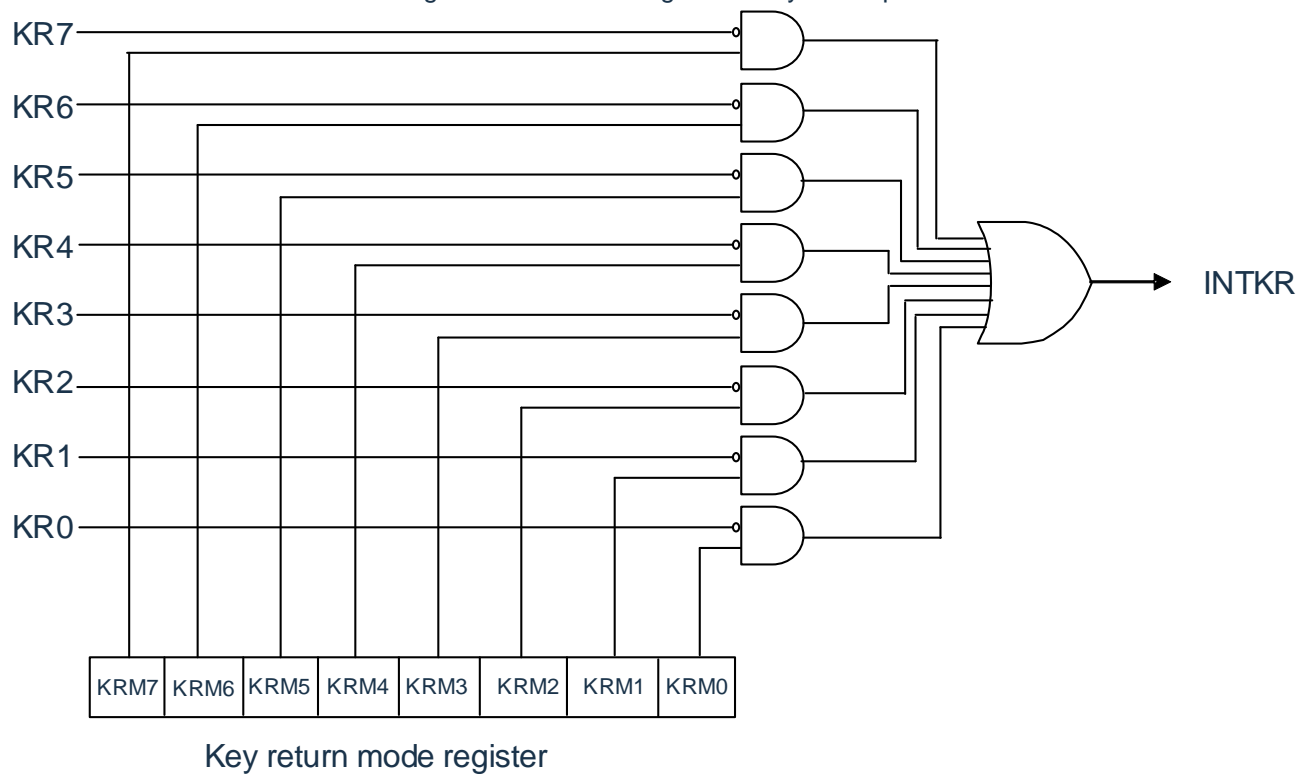
### 24.2 Structure of key interrupt

Key interrupts are made up of the following hardware.

Table 24-2: Structure of key interrupts

Item	Control register
Control register	Key return mode register (KRM) Port mode register (PMx). Port mode control register (PMCx).

Figure 24-1: Block diagram of key interrupt



## 24.3 Register controlling key interrupts

The key interrupt function is controlled by the following registers.

- Key Return Mode Register (KRM)
- Port Mode Register (PMx)

### 24.3.1 Key return mode register (KRM)

The KRM0~KRM7 bits control the KR0~KR7 signals.

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

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

Figure24-2: Format of the Oscillation Stop Detection Mode Register (KRM)

Address: 40042001H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
KRM	KRM7	KRM6	KRM5	KRM4	KRM3	KRM2	KRM1	KRM0

KRMn	Control of key interrupt mode
0	No key interrupt signal is detected.
1	Detects key interrupt signals.

Notice:

1. The internal pull-up resistor can be used by setting the object bit of the pull-up resistor register (PUx) of the key interrupt input pin to "1".
2. If the object bit of the KRM register is entered low on the input pin of the key interrupt, an interrupt is generated. To ignore this interrupt, the KRM register must be set after interrupt processing is disabled by the interrupt mask flag. The interrupt request flag must be cleared after waiting for the key interrupt input's low level width ( $T_{KR}$ ) (see data sheet) to allow interrupt processing.
3. Unused pins in key interrupt mode can be used as the usual port.

Remark: n=0~7.

# Chapter 25 Standby Function

## 25.1 Standby function

The standby function is a function that further reduces the operating current of the system, and there are two modes as follows.

### (1) Sleep mode

Sleep mode is the mode in which the CPU is stopped from running the clock. If the high-speed system clock oscillation circuit, high-speed internal 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 internal 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 31 Option Bytes".



## 25.2 Sleep mode

### 25.2.1 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 Table25-1.

Sleep mode can be set regardless of whether the CPU clock before setup is a high-speed system clock, a high-speed internal 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 decommission sleep mode. Therefore, even if the WFI command is executed in this case, it does not shift to sleep mode.

Table25-1: 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 with high speed internal oscillator clock (F <sub>IH</sub> ) run	CPU running at X1 clock (F <sub>X</sub> )	CPU with external master system clock (F <sub>EX</sub> ) run
Item					
System clock			Stop providing clocks to the CPU.		
	Main system Clock	F <sub>IH</sub>	Continues running (cannot be stopped).	Disables operation.	
		F <sub>X</sub>	Disables operation.	Continues running (cannot be stopped).	Cannot run.
		F <sub>EX</sub>		Cannot run.	Continues running (cannot be stopped).
	Subsystem Clock	F <sub>XT</sub>	Remain in the state before sleep mode.		
	F <sub>EXS</sub>				
Low-speed internal oscillation clock	F <sub>IL</sub>	Bit0 (WDSTBYON) and bit4 (WDTON) via option bytes (000C0H) and subsystem clocks Programmed for the WUTMMCK0 bit of the Mode Control Register (OSMC). WUTMMCK0=1: Oscillation WUTMMCK0=0 and WDTON=0: Stop WUTMMCK0=0, WDTON=1, and WDSTBYON=1: Oscillation WUTMMCK0=0, WDTON=1, and WDSTBYON=0: Stop			
CPU			Stop running.		
Code Flash					
RAM			Stop running (can run when DMA is executed).		
Port (Latch)			Remain in the state before sleep mode.		
DIV			Can run.		
Timer8					
Real Time Clock (RTC)					
15-bit interval timer					
Watch dog timer					
TimerA			Can run.		
Clock output/buzzer output					
AD converter					
D/A Converter					
Comparator					
General-purpose Serial Communication Unit (SCI)					
Serial Interface (IICA)					
aFCAN					
Data transfer controller (DMA)					
Linkage controller					
Power-on reset function			Can run.		
Voltage detection function					
External Interrupts					
Key interrupt function					
CRC operation function	High-speed CRC	It can be run when DMA is performed in the operation of the RAM area.			
	General CRC				
RAM parity check function			It can be run when the DMA is performed.		
SFR protection function					

Remark: Stop Run: Automatically stops running when shifting to sleep mode.

Disable Run: Stops running before shifting to sleep mode.

$F_{IH}$  : High-speed internal oscillator clock  $F_{IL}$  : Low-speed internal oscillator clock

$F_X$  : X1 clock  $F_{EX}$  : External master system clock

$F_{XT}$  : XT1 clock  $F_{EXS}$  : External subsystem clock

Table 25-1: Operation status in sleep mode (2/2)

Sleep mode setting Item			Execution of WFI instructions while the CPU is running at the subsystem clock	
			CPU running at XT1 clock (F <sub>XT</sub> )	CPU running on external subsystem clock (F <sub>EXS</sub> )
System Clock			Stop providing clocks to the CPU.	
	Main system Clock	F <sub>IH</sub>	Disable operation.	
		F <sub>X</sub>		
		F <sub>EX</sub>		
	Subsystem Clock	F <sub>XT</sub>	Continues running (cannot be stopped).	Cannot run.
		F <sub>EXS</sub>	Cannot run.	Continues running (cannot be stopped).
Low-speed internal oscillation clock	F <sub>IL</sub>	Bit0 (WDSTBYON) and bit4 (WDTON) via option bytes (000C0H) and subsystem clocks Programmed for the WUTMMCK0 bit of the Mode Control Register (OSMC). <ul style="list-style-type: none"><li>WUTMMCK0=1: Oscillation</li><li>WUTMMCK0=0 and WDTON=0: Stop</li><li>WUTMMCK0=0, WDTON=1, and WDSTBYON=1: Oscillation</li><li>WUTMMCK0=0, WDTON=1, and WDSTBYON=0: Stop</li></ul>		
CPU		Stop running.		
Code Flash				
RAM		Stop running (can run when DMA is executed).		
Port (Latch)		Remain in the state before sleep mode.		
DIV		When RTCLPC=0, it can run (otherwise it is prohibited).		
Time4		When RTCLPC=0, it can run (otherwise it is prohibited).		
Real Time Clock (RTC)		Can run.		
15-bit interval timer				
Watch dog timer		Refer to “Chapter 10 Watchdog Timer”.		
TimerA		When RTCLPC=0, it can run (otherwise it is prohibited).		
Clock output/buzzer output				
AD converter		Prohibit operation.		
D/A Converter		When RTCLPC=0, it can run (otherwise it is prohibited).		
Comparator		When the reference voltage of the comparator selects the external input (IVREFn), it can operate.		
General-purpose Serial Communication Unit (SCI)		When RTCLPC=0, it can run (otherwise it is prohibited).		
Serial Interface (IICA)		Prohibit operation.		
aFCAN		When RTCLPC=0, it can run (otherwise it is prohibited).		
Data transfer controller (DMA)		When RTCLPC=0, it can run (otherwise it is prohibited).		
Linkage controller		The ability to link between runnable function blocks.		
Power-on reset function		Can run.		
Voltage detection function				
External Interrupts				
Key interrupt function				
CRC operation function	High-speed CRC	Disables operation.		
	General CRC	It can be run when DMA is performed in the operation of the RAM area.		
RAM parity check function		It can be run when the DMA is performed.		
SFR protection function				

Remark: Stop Run: Automatically stops running when moving to sleep mode.

Disable Run: Stops running before moving to sleep mode.

F<sub>IH</sub> : High-speed internal oscillator clock F<sub>IL</sub> : Low-speed internal oscillator clock

F<sub>X</sub> : X1 clock F<sub>EX</sub> : External master system clock

F<sub>XT</sub> : XT1 clock F<sub>EXS</sub> : External subsystem clock

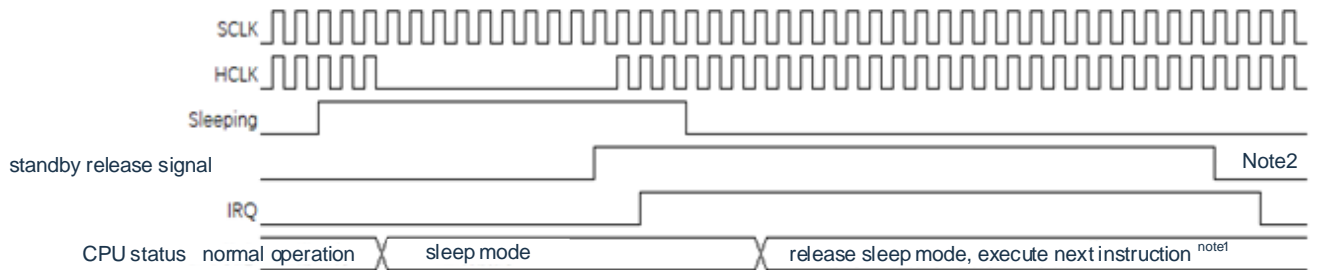
## 25.2.2 Release of sleep mode

Sleep mode can be released by any interrupt and external reset terminal, POR reset, low voltage detect reset, RAM parity error reset, WDT reset, software reset.

### (1) Dismissed by interrupt

When an unshielded interrupt is generated and the interrupt is allowed to be accepted, sleep mode is dismissed and the CPU begins processing interrupt services

Figure 25-1: Release sleep mode by interrupt request



Note 1: From the generation of the standby release signal to the release of sleep mode, it takes 16 clocks to start executing the interrupt service program.

Note 2: The standby release signal cannot be cleared by itself, and the register must be cleared. Write registers are typically cleared in interrupt service programs.

Notice: Before entering sleep mode, only the shield bits corresponding to the interrupts expected to be used to release sleep mode should be cleared.

### (2) Dismissed by resetting

When a reset signal is generated, the CPU is in a reset state and sleep mode is dismissed. As with the usual reset, the procedure is executed after the transfer to the reset vector address

Figure 25-2: Release sleep mode by reset



Note 1: For the reset processing time, please refer to “ Chapter 26 Reset Function ”. For reset processing time for power-on reset (POR) circuits and voltage detection (LVD) circuits, refer to “Power-on Reset Circuit”.

## 25.3 Deep sleep mode

### 25.3.1 Setting of deep sleep mode

When the SLEEP DEEP 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 stop functioning. 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 24-2.

Deep sleep mode can only be set if the CPU clock before setting is 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 dismiss deep sleep mode. Therefore, if the WFI instruction is executed in this case, it is dismissed as soon as it enters deep sleep mode. Returns to run mode after executing the WFI instruction and after a deep sleep mode release time has elapsed.

Table 25-2: Operating status in deep sleep mode

Deep sleep mode setting			Execution of WFI instructions while the CPU is running at the main system clock			
Item			CPU runs on a high-speed internal oscillator clock (F <sub>IH</sub> )		CPU running at X1 clock (F <sub>X</sub> )	CPU runs on an external master system clock (F <sub>EX</sub> )
System clock			Stop providing clocks to the CPU.			
	Main system Clock	F <sub>IH</sub>	Stop			
		F <sub>X</sub>				
		F <sub>EX</sub>				
	Subsystem Clock	F <sub>XT</sub>	Stay in the state before deep sleep mode.			
		F <sub>EXS</sub>				
Low-speed internal oscillation clock	F <sub>IL</sub>	Bit0 (WDSTBYON) and bit4 (WDTON) via option bytes (000C0H) and subsystem clocks Programmed for the WUTMMCK0 bit of the Mode Control Register (OSMC). WUTMMCK0=1: Oscillation WUTMMCK0=0 and WDTON=0: Stop WUTMMCK0=0, WDTON=1, and WDSTBYON=1: Oscillation WUTMMCK0=0, WDTON=1, and WDSTBYON=0: Stop				
CPU			Stop running.			
Code flash						
RAM						
Port (latch)			Stay in the state before deep sleep mode.			
DIV			Disables operation.			
Timer array unit			Disables operation.			
Real-time clock (RTC).			Can run.			
15-bit interval timer						
Watch dog timer			Refer to “Chapter10 Watchdog Timer”.			
TimerA			• Can be run in event counting mode without TAIO input filter selected. • Can run when the subsystem clock is selected as the counting source and the RTCLPC bit of the OSMC register is "0". • Operates when a low-speed internal oscillator is selected as the counting source. • Outside of the above: Operation is prohibited.			
Clock output/buzzer output			Disables operation.			
AD converter			Wake-up is possible.			
D/A Converter			Can run (remain in the state before deep sleep mode is set).			
Comparator			The comparator can operate (only if no digital filter is used and the reference voltage of the comparator selects the external input (IVREFn)).			
General-purpose Serial Communication Unit (SCI)			Only SSPIp and UARTq can wake up. Operation is prohibited except for SSPIp and UHRTq.			
Serial Array Unit (IICA).			Wake-up can be performed through address matching.			
aFCAN			Disables operation.			
Data Transfer Controller (DMA).			Can accept DMA boot source.			
Linkage controller			Links between runnable function blocks.			
Power-on reset function			Can run.			
Voltage detection function						
External interrupts						
Key interrupt function						
CRC operation function	Stop running.		Stop running.			
	General CRC					
RAM parity function						
SFR protection function						

Remark: Stop Run: Automatically stops running when shifting to deep sleep mode.

Disable run: Stops running before shifting to deep sleep mode.

$F_{IH}$  : High-speed internal oscillator clock  $F_{IL}$  : Low-speed internal oscillator clock

$F_X$  : X1 clock  $F_{EX}$  : External master system clock

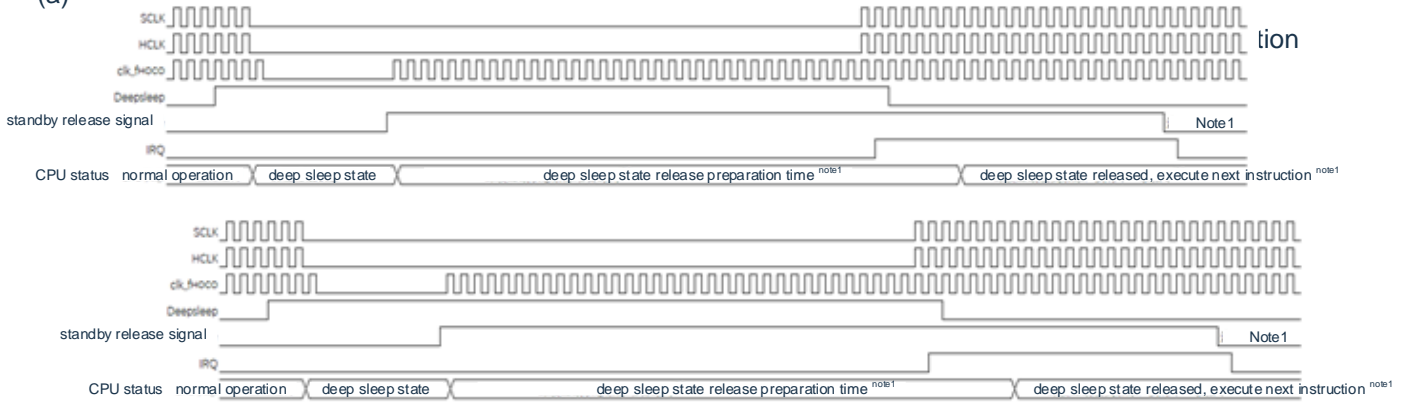
$F_{XT}$  : XT1 clock  $F_{EXS}$  : External subsystem clock



## 25.3.2 Release of deep sleep mode

Deep sleep mode can be released in 2 ways.

### (a) Release by unmasked interrupt request



Note 1: Standby release signal: For details of the standby release signal, please refer to “Figure 23-1 Basic structure of interrupt function”.

Note 2: Deep sleep release preparation time.

When the CPU clock is a high-speed internal 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).

Additional LOCKUP time is required when the CPU clock is PLL clock before entering deep sleep mode.

Note 3: Wait: 14 clocks are required from the time CPU.IRQ is valid to the interrupt service program start.

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 internal oscillator clock before the WFI instruction is executed.

Note: The oscillation accuracy of the high-speed internal oscillator clock varies steadily depending on temperature conditions and during deep sleep mode.

### (b) Release by generating a reset signal

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 25-4: Release deep sleep mode by resetting



Note: For reset processing time, see “Chapter 26 Reset Function”. For reset processing time for power-on reset (POR) circuits and voltage detection (LVD) circuits, see “Chapter 27 Power-on Reset Circuit”.

# Chapter 26 Reset Function

The following 8 methods generate the reset signal.

- (1) External reset is input via the RESETB pin.
- (2) Internal reset is generated by programmed runaway detection of the watchdog timer.
- (3) Internal reset is generated by comparison of the power supply voltage and the detection voltage of the power-on reset (POR) circuit.
- (4) An internal reset is generated by comparing the supply voltage of the voltage detection circuit (LVD) with the detection voltage.
- (5) An internal reset is generated by setting the system reset request register bit (AIRCR.SYSRESETREQ) to 1.
- (6) Internal reset due to RAM parity error.
- (7) Internal reset due to illegal memory access.
- (8) The stop detection function selects the reset mode and generates an internal reset when a stop is detected.

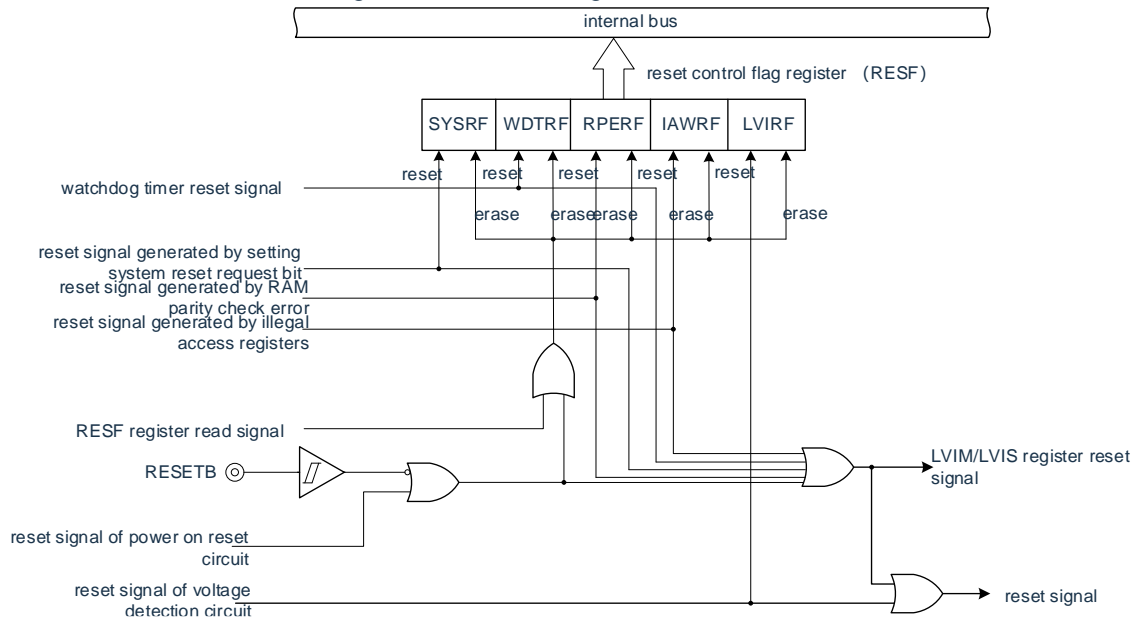
Internal reset is the same as external reset, and after generating a reset signal, the program is executed starting from the user-defined program start address.

When the RESETB pin is given a low input level, or the watchdog timer detects that the program is out of control, or the voltage of the POR circuit and the LVD circuit is detected, or the system reset request bit is set, or the RAM parity test error occurs, or the illegal memory is accessed. Alternatively, when a damping is detected, a reset is generated, and each hardware becomes in a state shown in Table 25-1.

## Notice:

1. When performing an external reset, a low of 10us must be input to the RESETB pin. If an external reset is performed while the supply voltage rises, the power must be switched on after inputting a low level to the RESETB pin and maintaining at least 10us of operating voltage as shown in the AC characteristics of the datasheet low level, and then enter high level.
2. Stop oscillating of 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 the external subsystem clock are invalid.

Figure 26-1: Block diagram of reset function



Notice: The internal reset of the LVD circuit does not reset the LVD circuit.

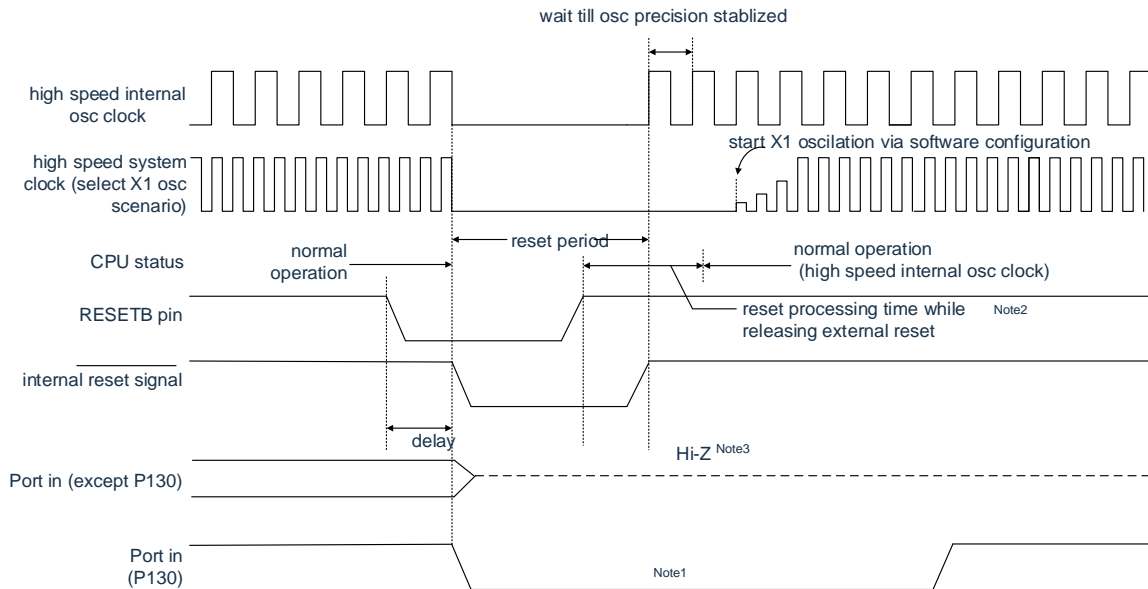
Remark:

1. LVIM: Voltage detection register
2. LVIS: Voltage detection level register

## 26.1 Reset timing

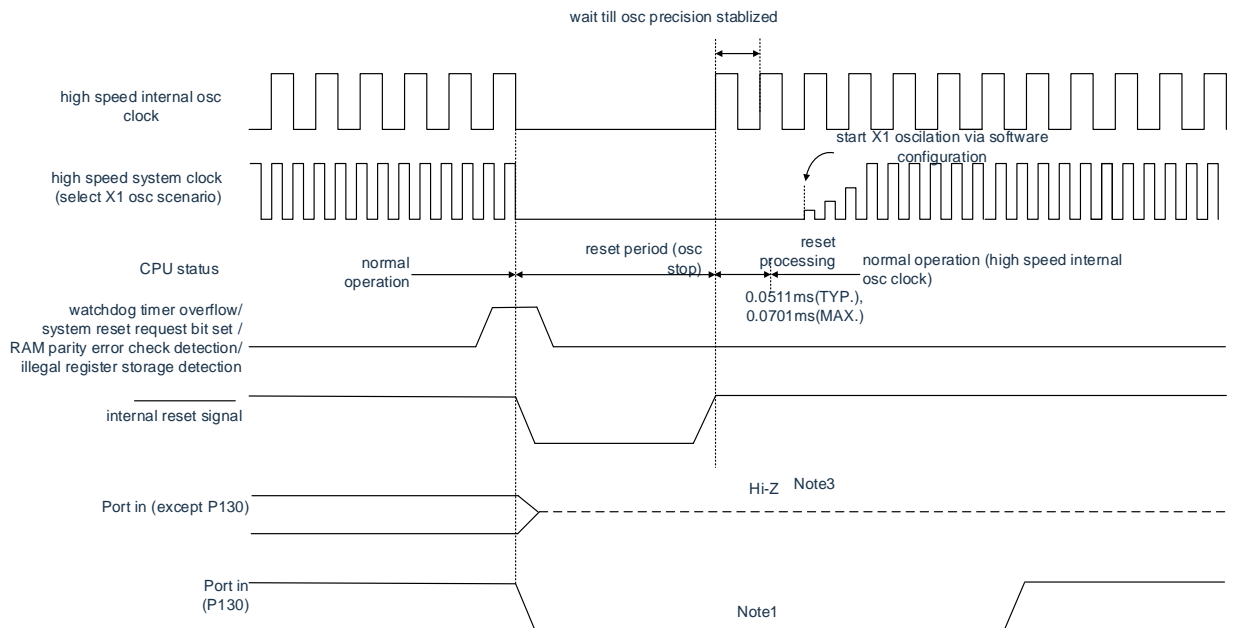
When the RESETB pin is input low, a reset is generated. The reset state is then released if the RESETB lead is entered high and the program begins with a high-speed internal oscillator clock after the reset process is complete.

Figure 26-2: Timing of RESETB input



For resets caused by overflow of watchdog timers, assertion of system reset request bits, detection of RAM parity errors, or detection of illegal memory access or vibration stopping detection, the reset state is automatically released, and the program is executed with a high-speed internal oscillator clock after the reset process is completed.

Figure 26-3: Reset timing due to overflow of watchdog timer, assertion of system reset request bits, detection of RAM parity errors, or detection of illegal memory access



Note 1: Reset processing time when removing external reset:

1st time after POR released:

0.672ms(TYP.), 0.832ms(MAX.)(using LVD)

0.399ms(TYP.), 0.519ms(MAX.)(not using LVD)

2nd time after the POR released:

0.531ms(TYP.), 0.675ms(MAX.)(using LVD)

0.259ms(TYP.), 0.362ms(MAX.)(not using LVD)

When the power supply voltage rises, a voltage stabilization wait time of 0.99ms (TYP.), 2.30ms (MAX.) is required before the reset processing time at external reset is released

Note 2: Port pins PA13, PA14 change to the following state:

- High impedance during external reset or POR reset.
- High level during other resets and after accepting a reset (internal pull-up resistor 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, the reset state is released and the program is executed with a high-speed internal oscillator clock after the reset is processed. For details, please refer to “Chapter 27 Power-on Reset Circuit” and “Chapter 28 Voltage Detection Circuit”.

Remark:  $V_{POR}$ : POR supply voltage rise detection voltage

$V_{LVD}$ : LVD detection voltage

Table 26-1: Operational status during reset

Item			During reset
System clock			Stop providing clocks to the CPU.
	Main system Clock	F <sub>IH</sub>	Stop running.
		F <sub>X</sub>	Stop operation (pins X1 and X2 are in input port mode).
		F <sub>EX</sub>	The clock input is invalid (the pin is in input port mode).
	Subsystem Clock	F <sub>XT</sub>	Can run.
		F <sub>EXS</sub>	The clock input is invalid (the pin is in input port mode).
	Low-speed internal oscillation clock	F <sub>IL</sub>	Stop running.
CPU			
Code flash			Stop running.
RAM			Stop running.
Port (latch)			High impedance <sup>Note 1</sup>
DIV			Stop running.
Timer array unit			
TimerA			
Real-time clock (RTC).			At POR reset, stops running. On other resets, it can run.
15-bit interval timer			Stop running.
Watch dog timer			
Clock output/buzzer output			
AD converter			
D/A converter <sup>Note 1</sup>			
Comparator <sup>Note 1</sup>			
General-purpose Serial Communication Unit (SCI)			
Serial Interface (IICA)			
aFCAN			
Data transfer controller (DMA)			
Power-on reset function			Can perform test runs.
Voltage detection function			When LVD is reset, it can run. At other resets, it stops running.
External Interrupts			Stop running.
Key interrupt function			
CRC operation function	High-speed CRC		
	General CRC		
RAM parity check function			
SFR protection function			

Remark: F<sub>IH</sub> : High-speed internal oscillator clock F<sub>X</sub> : X1 oscillator clock

F<sub>EX</sub> : External master system clock F<sub>XT</sub> : XT1 oscillation clock

F<sub>EXS</sub> : External subsystem clock F<sub>IL</sub> : Low-speed internal oscillator clock

## 26.2 Register for confirming the reset source

### 26.2.1 Reset control flag register (RESF)

The CMS32H6157 microcontroller has multiple internal reset generation sources. The Reset Control Flag register (RESF) holds the reset source where the reset request occurs. The RESF register can be read by an 8-bit memory manipulation instruction.

The SYSRF, WDTRF, CLMRF, RPERF, IAWRF, LVIRF flags are cleared by inputting RESETB, resetting the power-on reset (POR) circuit, and reading the RESF register. To determine the reset source, the value of the RESF register must be saved to any RAM and then determined by its RAM value.

Figure 26-4: Format of reset control flag register (RESF)

Address: 40020440H

After reset: indefinite value<sup>Note 1</sup>

R

Symbol	7	6	5	4	3	2	1	0
RESF	SYSRF		0	WDTRF	CLMRF	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	Generate 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	Generate an internal reset request.

CLMRF	Internal reset request generated by the deactivation detection function
0	No internal reset request is generated or the RESF register is cleared.
1	Generate 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	Generate 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	Generate 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	Generate an internal reset request.

Note 1: Varies depending on the reset source. Please refer to Table 26-2.

Notice: In the case of ALLOWing RAM parity error reset (RPERDIS=0), when accessing data, the "RAM region used" must be initialized; When executing an instruction from a RAM region, the area of "RAM area +10 bytes used" must be initialized. By generating a reset, enter a state that allows the generation of a RAM parity error reset (RPERDIS=0). For details, please refer to "28.3.3 RAM Parity Error Detection Function".

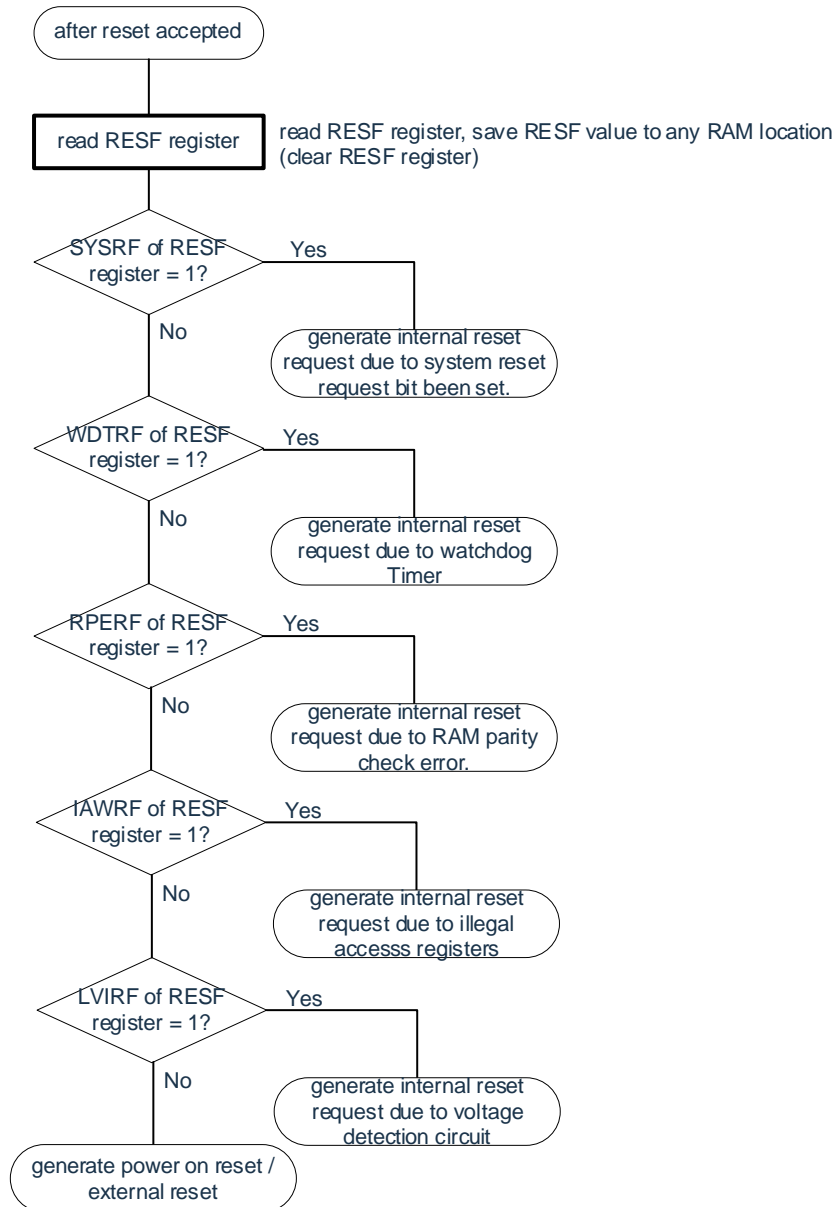
The status of the RESF register when a reset request occurs is shown in Table 26-2.

Table 26-2: RESF register status when a reset request occurs

Reset Source Flag	RESETB input	Reset of POR generated	System reset request bit Reset generated by the set bit	WDT- generated reset	Reset due to vibration stop detection	Reset due to RAM parity errors	Access to the reset generated by illegal memory	LVD- generated reset
SYSRF	Set "0"	Set "0"	Set "1"	Keep	Keep	Keep	Keep	Keep
WDTRF			Keep	Set "1"				
CLMRF				Keep	Set "1"	Set "1"	Set "1"	
RPERF					Keep			Keep
IAWRF						Keep	Keep	Keep
LVIRF					Keep		Keep	Set "1"

The confirmation steps for resetting the source are shown as below.

Figure 26-5: Confirmation steps for reset source



Notice: The above process is an example of a confirmation step.



# Chapter 27 Power-on Reset Circuit

## 27.1 Function of power-on reset circuit

The power-on reset circuit (POR) has the following functions.

- Generates an internal reset signal when power is turned on.  
If the supply voltage ( $V_{DD}$ ) exceeds the sense voltage ( $V_{POR}$ ), the reset is dismissed. However, the reset state must be maintained by voltage detection circuitry or an external reset until the supply voltage reaches the operating voltage range shown in AC characteristics of the data sheet.
- Drag the supply voltage ( $V_{DD}$ ) and the detection voltage ( $V_{PDR}$ ) to compare. While  $V_{DD} < V_{PDR}$ , an internal reset signal is generated. However, when the supply voltage drops, the supply voltage must be lower than AC characteristics of the data sheet before the operating voltage range shown, It is reset by means of transfer in deep sleep mode, voltage detection circuitry, or external reset. When restarting operation, you must confirm that the supply voltage has returned to the operating voltage range.

Notice: When the power-on reset circuit generates an internal reset signal, the reset control flag register (RESF) is cleared to "00H"

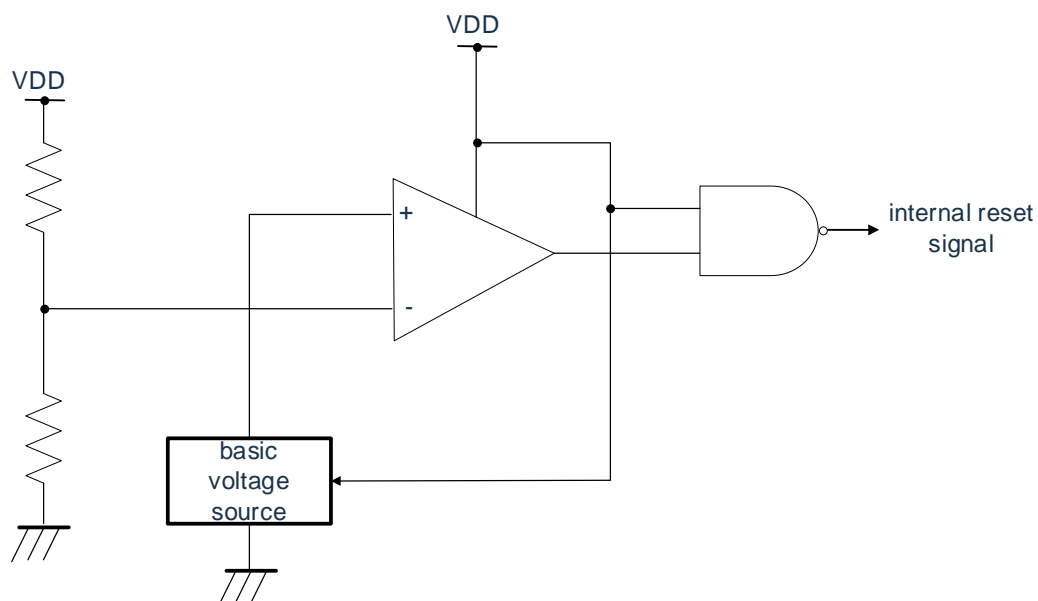
Remark:

1. The CMS32H6157 has built-in hardware to generate several internal reset signals. When the internal reset signal is generated by the watchdog timer (WDT), voltage detection (LVD) circuit, system reset request bit, RAM parity error or illegal memory access, the flag to indicate the reset source is assigned in the RESF register; when the internal reset signal is generated by the WDT, LVD, system reset request bit setting, RAM parity error or illegal memory access, the flag is set to "1" instead of clearing "00H" in the RESF register. When the internal reset signal is generated by WDT, LVD, system reset request bit setting, RAM parity error or illegal memory access, the RESF register is not cleared to "00H" and the flag is set to "1". For details of the RESF register, please refer to "Chapter 26 Reset function".
2.  $V_{POR}$ : POR supply voltage rise detection voltage  
 $V_{PDR}$ : POR supply voltage drop detection voltage  
Please refer to the POR circuit characteristics in the data sheet for details.

## 27.2 Structure of power-on reset circuit

The block diagram of the power-on reset circuit is shown in Figure 27-1.

Figure 27-1: Block diagram of power-on reset circuit

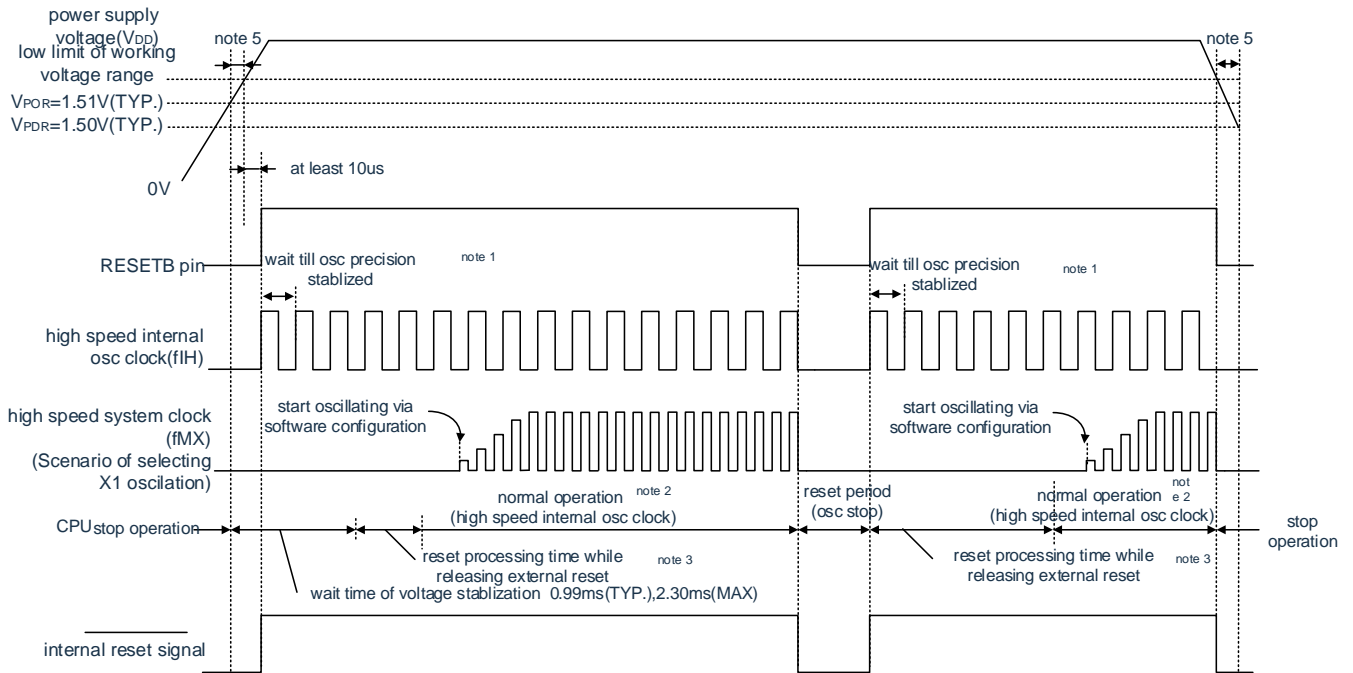


## 27.3 Operation of power-on reset circuit

The timing of the internal reset signal generation for the power-on reset circuit and the voltage detection circuit is shown below.

Figure 27-2: Timing of internal reset signal generation for power-on reset circuit and voltage detection circuit(1/3)

(1) The case of using an external reset input from the RESETB pin



1st time after POR is released: 0.672ms(TYP.), 0.832ms(MAX.) (using LVD)

0.399ms(TYP.), 0.519ms(MAX.)(not using LVD)

Note 4: The reset processing time for the second and subsequent release of the external reset after the release of POR is shown below.

2nd time after POR is released: 0.531ms(TYP.), 0.675ms(MAX.) (using LVD)

0.259ms(TYP.), 0.362ms(MAX.)( not using LVD)

Note 5: 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.

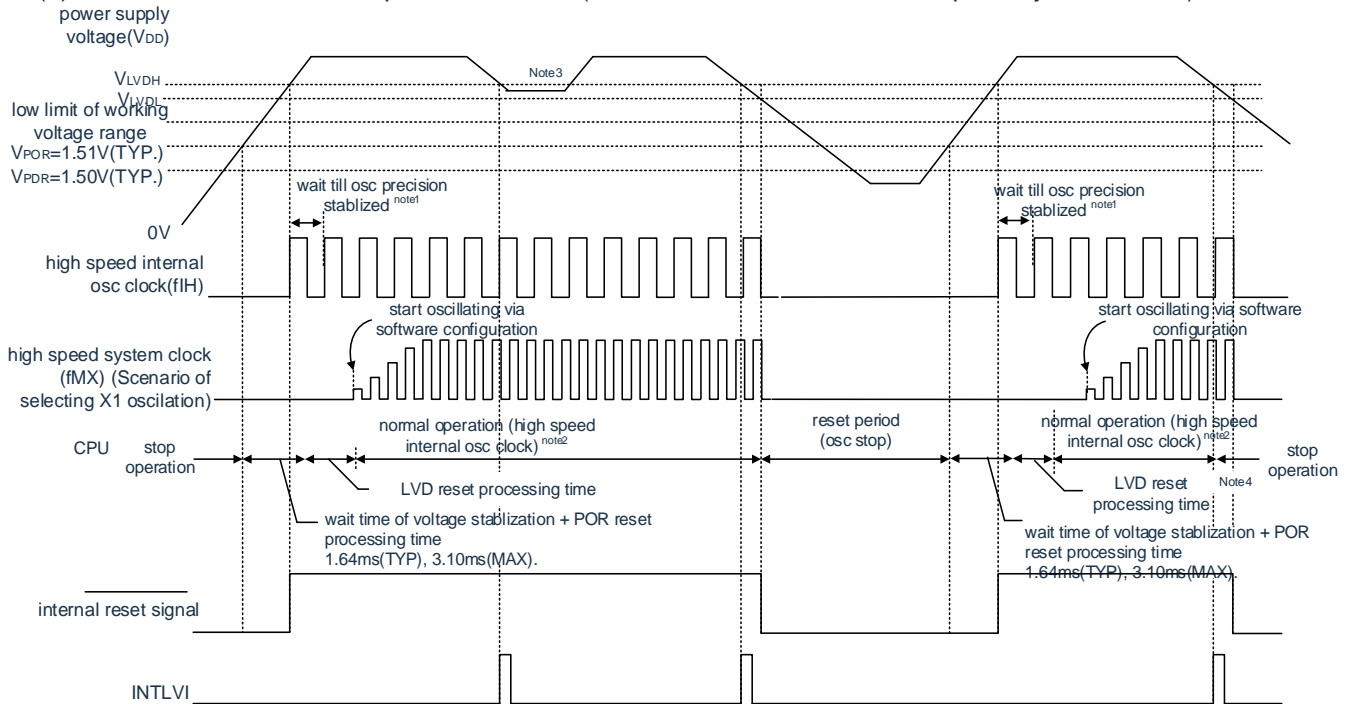
Remark:  $V_{POR}$ : POR supply voltage rise detection voltage

$V_{PDR}$ : POR supply voltage drop detection voltage

Notice: When LVD is OFF, the external reset of RESETB pin must be used. For details, please refer to “Chapter 28 Voltage Detection Circuit”.

Figure 27-3: Timing of internal reset signal generation for power-on reset circuit and voltage detection circuit (2/3)

(2) LVD is the case of interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0 for option bytes 000C1H)



Note 1: The internal reset processing time includes the oscillation accuracy stabilization wait time of the high-speed internal oscillator clock.

Note 2: The CPU clock can be switched from the high-speed internal oscillator clock to the high-speed system clock or the subsystem clock. In the case of X1 clock, the switch must be made after checking the oscillation stability time by the status register of the oscillation stability time counter (OSTC); in the case of XT1 clock, the switch must be made after checking the oscillation stability time by the timer function, etc.

Note 3: After generating the interrupt request signal (INTLVI), the LVILV bit and the LVIMD bit of the voltage detection level register (LVIS) are automatically set to "1". Therefore, it must be considered that the supply voltage may return to the high voltage detection voltage (VLVDH) or higher without falling below the low voltage detection voltage (VLVDL), and after generating INTLVI, follow the steps in "Figure 28-7 Setting procedure for confirmation/reset of operating voltage" and "Figure 28-8: Initial setting procedure for interrupt & reset mode".

Note 4: In addition to the "Voltage stabilization wait time + POR reset processing time" after reaching  $V_{POR}$  (1.51V(TYP.)), the following "LVD reset processing time" is required after reaching the LVD detection level ( $V_{LVDH}$ ). "

LVD reset processing time: 0ms~0.0701ms(MAX.)

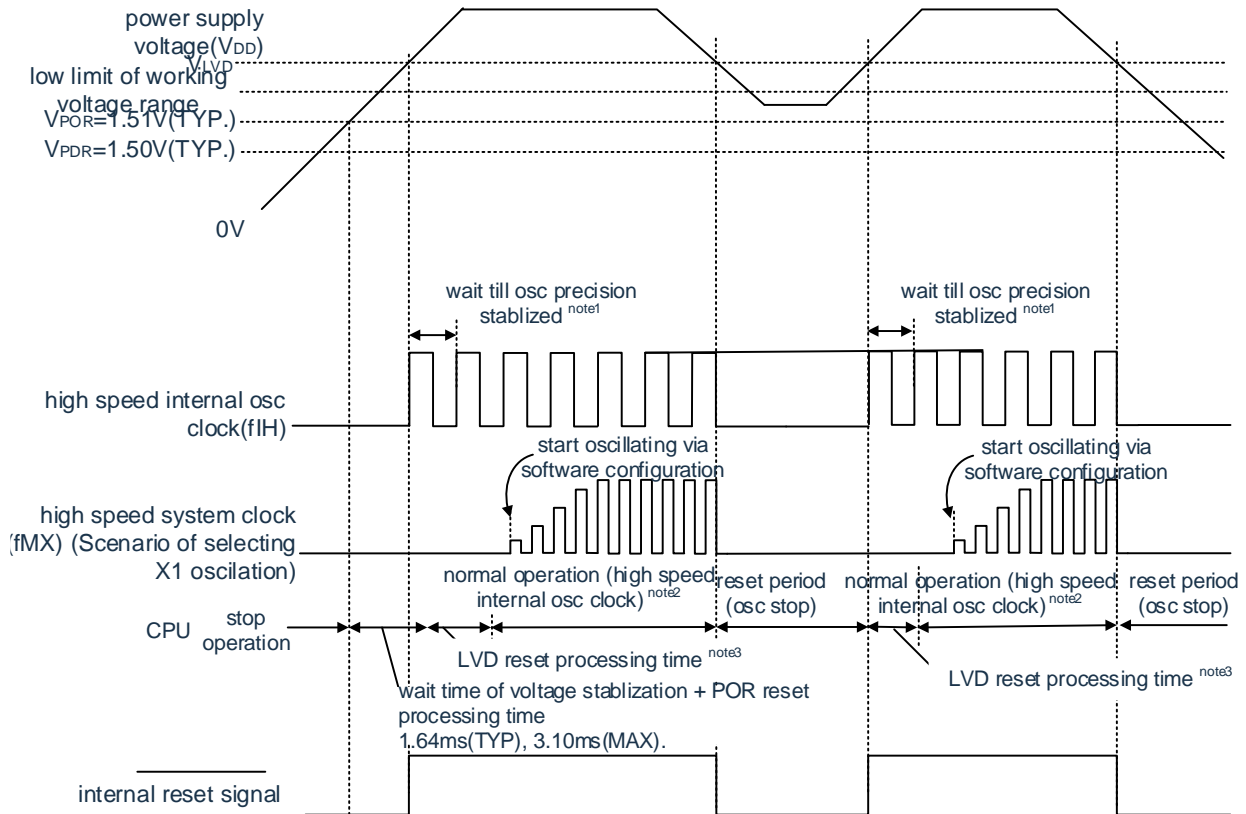
Remark:  $V_{LVDH}$ ,  $V_{LVDL}$ : LVD detection voltage

$V_{POR}$ : POR supply voltage rise detection voltage

$V_{PDR}$ : POR supply voltage drop detection voltage

Figure 27-4: Timing of internal reset signal generation for power-on reset circuit and voltage detection circuit (3/3)

### (3) LVD reset mode (LVIMDS1, LVIMDS0=1, 1 for option byte 000C1H)



Note 1: The internal reset processing time includes the oscillation accuracy stabilization wait time of the high-speed internal oscillator clock

Note 2: The CPU clock can be switched from the high-speed internal oscillator clock to the high-speed system clock or the subsystem clock. In the case of X1 clock, the switch must be made after checking the oscillation stability time by the status register of the oscillation stability time counter (OSTC); in the case of XT1 clock, the switch must be made after checking the oscillation stability time by the timer function, etc.

Note 3: The time until the start of normal operation except for reaching  $V_{POR}$  (1.51V (TYP.)). In addition to the "Voltage Stabilization Wait Time + POR Reset Processing Time", the following is required after the LVD detection level ( $V_{LVD}$ ) is reached LVD Reset Processing Time.

LVD reset processing time: 0ms~0.0701ms(MAX.)

Note 4: When the supply voltage drops, if the supply voltage is only restored after the internal reset of the voltage detection circuit (LVD) occurs, the following "LVD" is required after reaching the LVD detection level ( $V_{LVD}$ ). Reset Processing Time".

LVD reset processing time: 0.0511ms(TYP.), 0.0701ms(MAX.)

Remark:

1.  $V_{LVDH}$ ,  $V_{LVDL}$ : LVD detection voltage  
 $V_{POR}$ : POR supply voltage rise detection voltage  
 $V_{PDR}$ : POR supply voltage drop detection voltage
2. When the LVD interrupt mode is selected (LVIMD1, LVIMD0=0, 1 for option byte 000C1H), the time from the time of power on to the start of normal operation and the "Note 3" time of "Figure 26-2(3) LVD bit mode " are the same.

# Chapter 28 Voltage Detection Circuit

## 28.1 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 Sense (LVD) circuit 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 31 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 for option byte)

A detection voltage ( $V_{LVD}$ ) selected by option byte 000C1H is used to generate or unmake the reset.

(c) Interrupt mode (option byte of LVIMDS1, LVIMDS0=0, 1)

A detection voltage ( $V_{LVD}$ ) selected by option byte 000C1H is used to generate an interrupt or to release the reset. In each mode, the following interrupt signals and internal reset signals are generated.

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, the internal reset is released.	When $V_{DD} \geq V_{LVD}$ is detected, the internal reset is released; when $V_{DD} < V_{LVD}$ is detected, the internal reset is generated.	After a reset occurs, the internal reset state of LVD continues until $V_{DD} \geq V_{LVD}$ . When $V_{DD} \geq V_{LVD}$ is detected, the 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 The interrupt request signal (INTLVI) is generated.

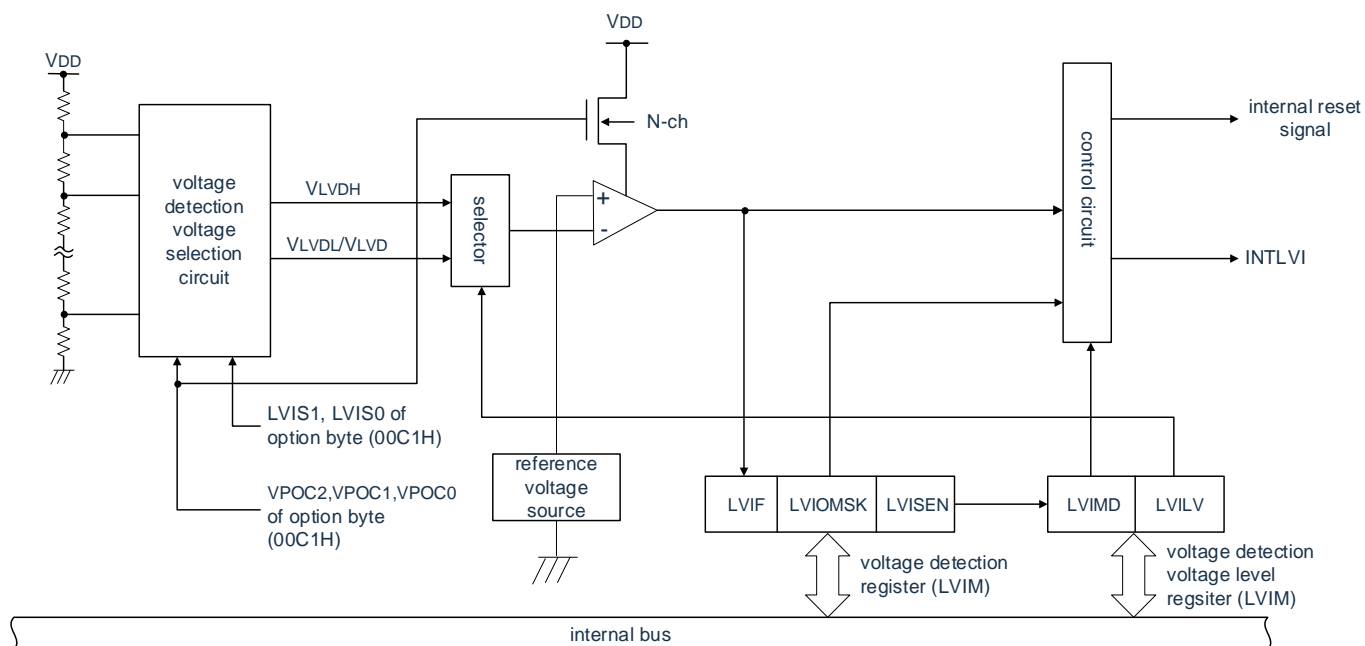
When the voltage detection circuit is in operation, it is possible to check whether the power supply voltage is greater than or less than the detection voltage by reading the voltage detection flag (LVIF: bit 0 of the voltage detection register (LVIM)).

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 26 Reset Function”.

## 28.2 Structure of voltage detection circuit

The block diagram of the voltage detection circuit is shown in Figure 28-1.

Figure 28-1: Block diagram of voltage detection circuit





## 28.3 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)

### 28.3.1 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 28-2: Format of voltage detection register

Address: 40020441H		After reset: 00H <sup>Note 1</sup>		R/W <sup>Note 2</sup>			
Symbol	7	6	5	4	3	2	1 0
LVIM	LVISEN <sup>Note 3</sup>	0	0	0	0	0	LVIOMSK LVIF

LVISEN <sup>Note 3</sup>	Setting of the voltage detection level register (LVIS)
0	Disable rewriting LVIS register (LVIOMSK=0 (LVD output mask is invalid)).
1	Enable rewriting LVIS register (LVIOMSK=1 (LVD output mask valid)).

LVIOMSK	Mask status flag for LVD output
0	LVD output masking is invalid.
1	LVD output masking is valid <sup>Note 4</sup> .

LVIF	Voltage detection mark
0	Supply voltage ( $V_{DD}$ ) $\geq$ detection voltage ( $V_{LVD}$ ) or LVD is OFF.
1	Supply voltage ( $V_{DD}$ ) $<$ detection voltage ( $V_{LVD}$ )

Note 1: The reset value varies depending on the reset source.

When the LVD is reset, the value of the LVIM register is not reset and the original value is maintained;  
During other resets, clear LVISEN to "0".

Note 2: Bit0 and bit1 are read-only bits.

Note 3: It can only be set when the interrupt & reset mode is selected (lvIMDS1 bits and LVIMDS0 bits of the option bytes are "1" and "0" respectively), the initial value cannot be changed in other modes.

Note 4: Only when the interrupt & reset mode is selected (the LVIMDS1 bit and LVIMDS0 bits of the option byte are "1" and "0" respectively). The LVIOMSK bit automatically changes to "1" during the following periods, masking the reset or interrupt generated by LVD.

- When LVISEN=1
- Waiting time from the occurrence of LVD interrupt to the stabilization of LVD detection voltage
- Wait time from changing the value of the LVILV bit until the LVD detection voltage stabilizes

## 28.3.2 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”<sup>Note 1</sup>.

Figure 28-3: Format of voltage detection level register (LVIS)

Address: FFFAAHAfter reset: 00H/01H/81H <sup>Note 1</sup>								R/W
Symbol	7	6	5	4	3	2	1	0
LVIS	LVIMD <sup>Note 2</sup>	0	0	0	0	0	0	LVILV <sup>Note 2</sup>

LVIMD <sup>Note 2</sup>	Operation mode of voltage detection
0	Interrupt mode
1	Reset mode

LVILV <sup>Note 2</sup>	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:

- Option bytes for LVIMDS1, LVIMDS0=1, 0: 00H
- Option bytes for LVIMDS1, LVIMDS0=1, 1: 81H
- Option bytes for LVIMDS1, LVIMDS0=0, 1: 01H

Note 2: Write “0” only if interrupt & reset mode is selected (LVIMDS1 bit and LVIMDS0 bits for option bytes are “1” and “0” respectively). In other cases, it cannot be set. In interrupt & reset mode, value substitution is performed automatically by generating a reset or interrupt.

Notice:

1. To rewrite the LVIS registers, it must be done in accordance with the steps in Figure 28-7 and Figure 28-8.
2. Select the operating mode of LVD and the detection voltage of each mode (VLVDH, VLVDL, V) by option byte 000C1H LVD). The format of the user option bytes (000C1H/010C1H) is shown in Table 28-1. For details on option bytes, refer to “Chapter 31 Option Bytes”.

Table 28-1: Format of user option bytes (000C1H/010C1H) (1/2)

Address: 000C1H/010C1H<sup>Note</sup>

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 <sub>LVDH</sub>		V <sub>LVDL</sub>	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		
-			Setting values other than those described above is prohibited.						

- LVD settings (reset mode)

Detection voltage		Setting value of option byte						
V <sub>LVD</sub>		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		
-			Setting values other than those described above is prohibited					

Remark:

- For details on LVD circuits, refer to “Chapter 28 Voltage Detection Circuit”.
- The detection voltage is TYP Value. For details, please refer to the LVD circuit characteristics in the data sheet.

Table 28-1: Format of user option byte (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						
VLVD		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		
-		Setting values other than those described above is prohibited.						

• LVD is OFF(External reset using the RESETB pin)

Detection voltage		Setting value of option byte						
VLVD		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rising	falling						LVIMDS1	LVIMDS1
-	-	1	x	x	x	x	x	x
-		Setting values other than those described above is prohibited.						

Notice:

1. Write "1" to bit4.
2. When the power supply voltage rises, the reset state must be maintained through the voltage detection circuit or external reset before the power supply voltage reaches the working voltage range shown in the AC characteristics of the data sheet; When the supply voltage drops, it must be reset by transferring in deep sleep mode, voltage detection circuitry, or external reset before the supply voltage falls below the operating voltage range.
3. The operating voltage range depends on the setting of the user option byte (000C2H).

Remark:

1. x: Ignore
2. For details of LVD circuits, please refer to "Chapter 28 Voltage Detection Circuit".
3. The detection voltage is TYP Value. For details, please refer to the LVD circuit characteristics in the data sheet.

## 28.4 Operation of voltage detection circuit

### 28.4.1 Settings when used in 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.

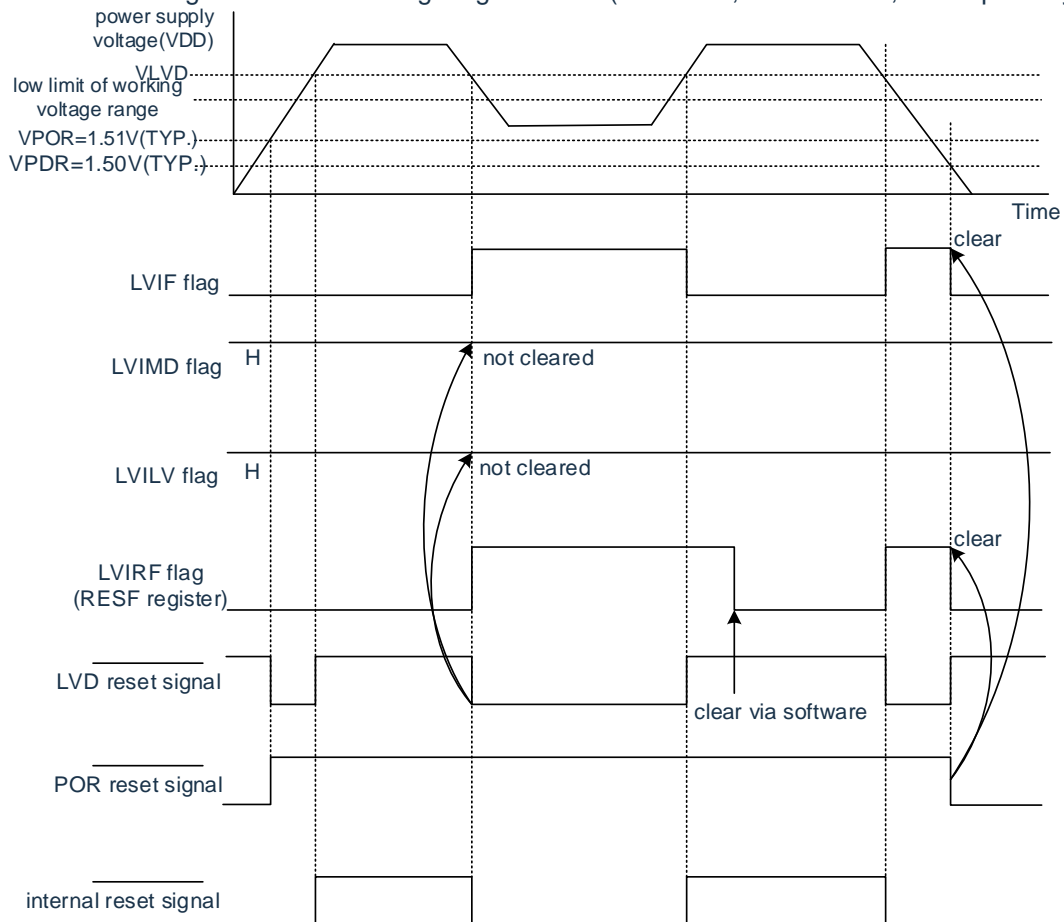
- Set bit 7 (LVISEN) of the voltage detection register (LVIM) to "0" (disable rewriting the voltage detection level register (LVIS))
- Set the initial value of the voltage detection level register (LVIS) to "81H". Set bit7(LVIMD) to "1" (reset mode). Set bit0(LVILV) to "1" (voltage detection level:  $V_{LVD}$ ).
- LVD reset mode operation

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 drops, 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 28-4.

Figure 28-4: Timing of internal reset signal generation (LVIMDS1, LVIMDS0=1, 1 for option byte)



Remark:  $V_{POR}$ : POR supply voltage rise detection voltage

$V_{PDR}$ : POR supply voltage drop detection voltage

## 28.4.2 Settings when used in 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.

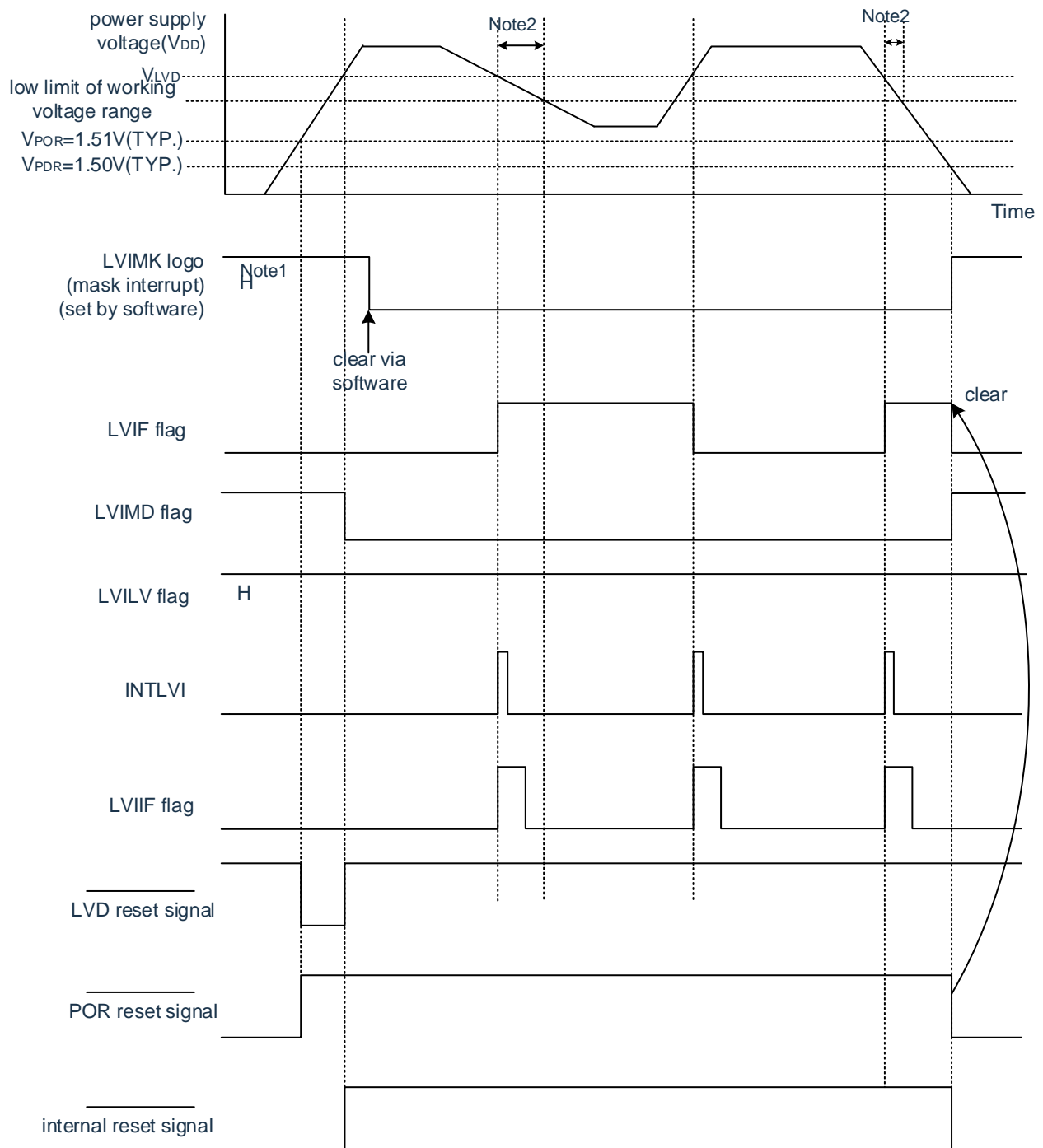
- Set bit 7 (LVISEN) of the voltage detection register (LVIM) to "0" (disables rewriting the voltage detection level register (LVIS)).
- Set the initial value of the voltage detection level register (LVIS) to "01H". Set bit7 (LVIMD) to "0" (interrupt mode).  
Set bit0(LVILV) to "1" (voltage detection level: VLVD).
- Operation of LVD interrupt mode

After generating a reset, the interrupt mode (LVIMDS1, LVIMDS0=0, 1 of the option byte) 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 28-5.

Figure 28-5: Timing of interrupt signal generation (LVIMDS1, LVIMDS0=0, 1 for option byte)



Note 1: After generating a reset signal, the LVIMK flag changes to "1".

Note 2: When the operating voltage drops, it must be reset by deep sleep mode transfer or external reset before the operating voltage falls below the operating voltage range shown in the AC characteristics of the data sheet. When restarting operation, it must be verified that the supply voltage returns to the operating voltage range.

Remark:  $V_{POR}$ : POR supply voltage rise detection voltage

$V_{PDR}$ : POR supply voltage drop detection voltage

### 28.4.3 Settings for 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:  $V_{LVDH}$ ).

#### LVD interrupt & reset mode operation

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.

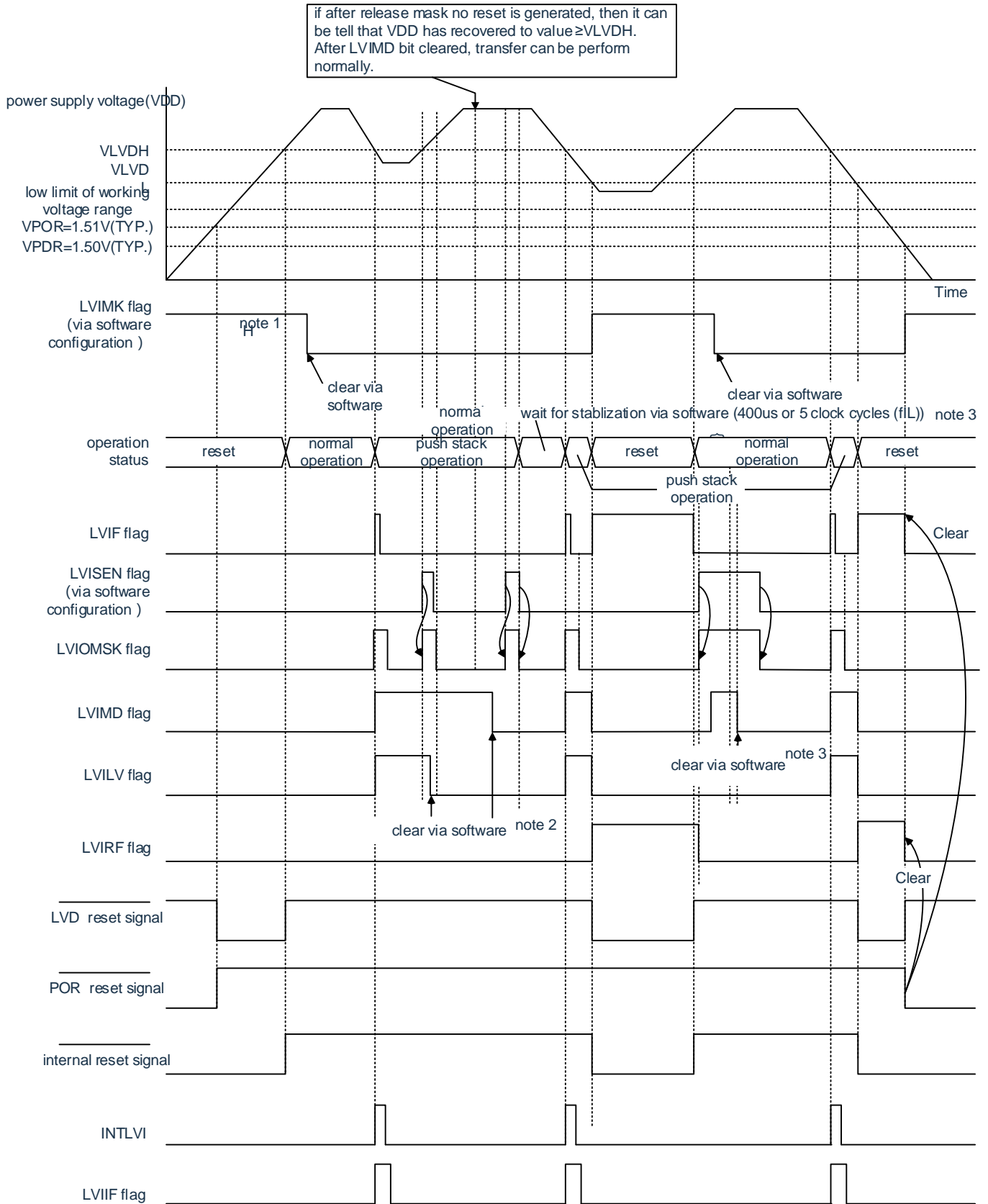
When the operating voltage drops, if the supply voltage ( $V_{DD}$ ) is below the high voltage detection level ( $V_{LVDH}$ ), an interrupt request signal (INTLVI) is generated for the LVD and any stacking process can be performed. After that, if the supply voltage ( $V_{DD}$ ) is below the low voltage 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 28-7: Setting procedure for confirmation /reset of operating voltage" and "Figure 28-8: 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 28-6.

Figure 28-6: Reset & Interrupt Signal Generation Timing (LVIMDS1, LVIMDS0=1, 0)(1/2)



Note 1: After the reset signal is generated, the LVIMK flag becomes "1".

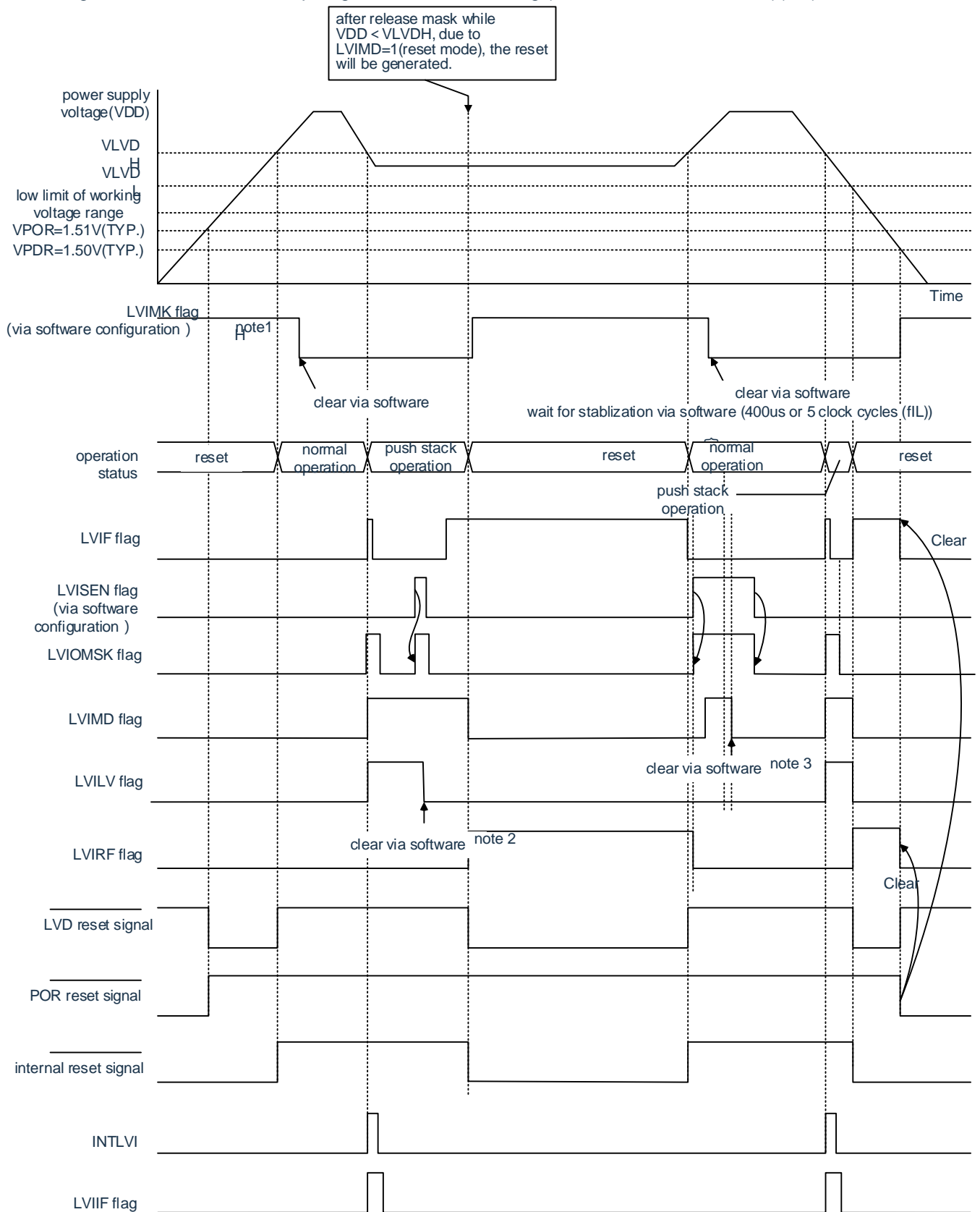
Note 2: When using the interrupt & reset mode, you must follow "Figure 28-7: Setting procedure for confirmation /reset of operating voltage" after an interrupt occurs.

Note 3: : When using the interrupt&reset mode, you must follow the steps in "Figure 28-8: Initial setting procedure for interrupt & reset mode" after the reset is released.

Remark:  $V_{POR}$ : POR supply voltage rise detection voltage

$V_{PDR}$ : POR supply voltage drop detection voltage

Figure 28-6: Reset & Interrupt Signal Generation Timing (LVIMDS1, LVIMDS0=1, 0)(2/2)



Note 1: After the reset signal is generated, the LVIMK flag becomes "1".

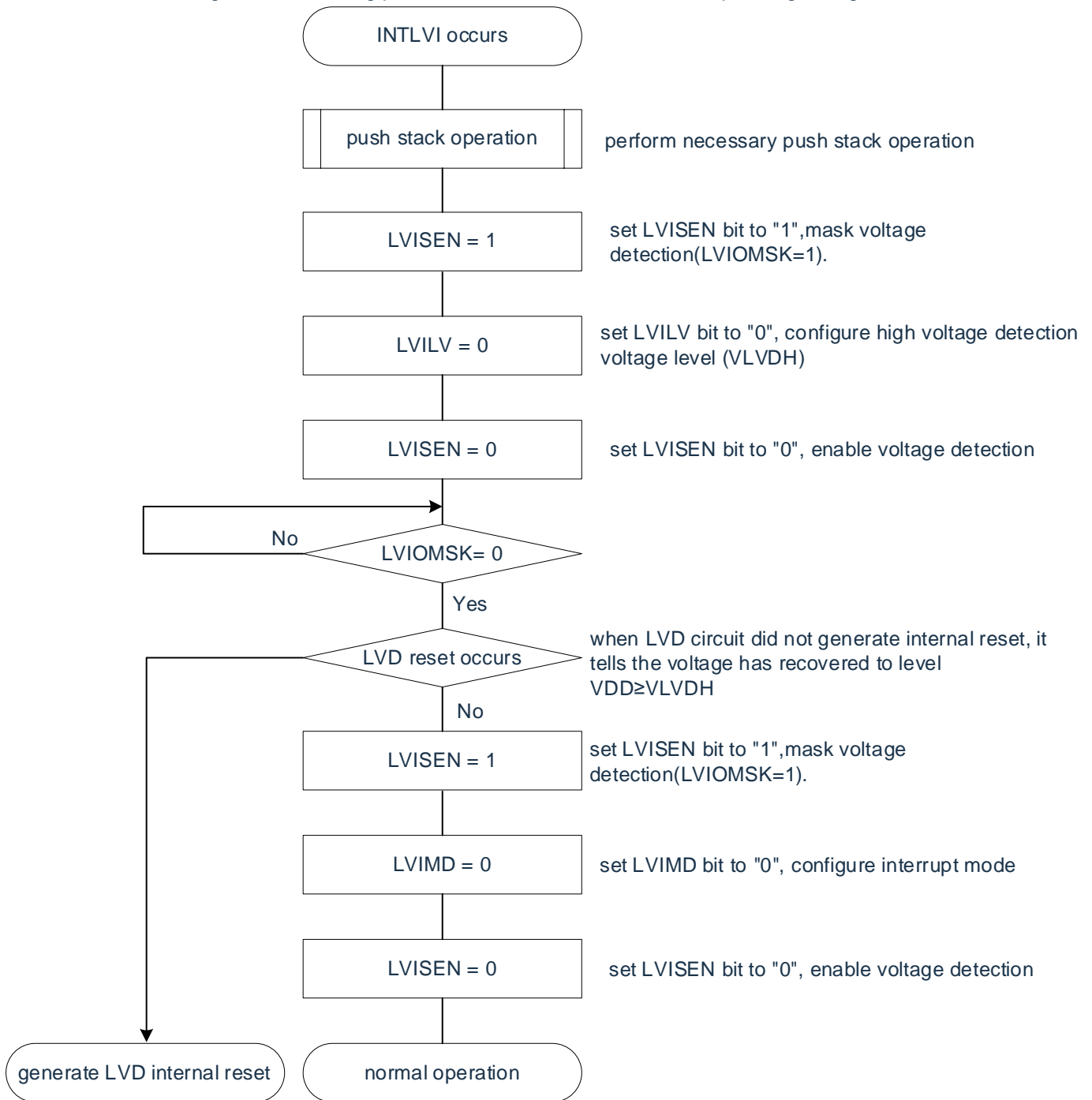
Note 2: When using the interrupt & reset mode, you must follow "Figure 28-7: Setting procedure for confirmation /reset of operating voltage" after an interrupt occurs.

Note 3: When using the interrupt&reset mode, you must follow the steps in "Figure 28-8: Initial setting procedure for interrupt & reset mode" after the reset is released.

Remark:  $V_{POR}$ : POR supply voltage rise detection voltage

$V_{PDR}$ : POR supply voltage drop detection voltage

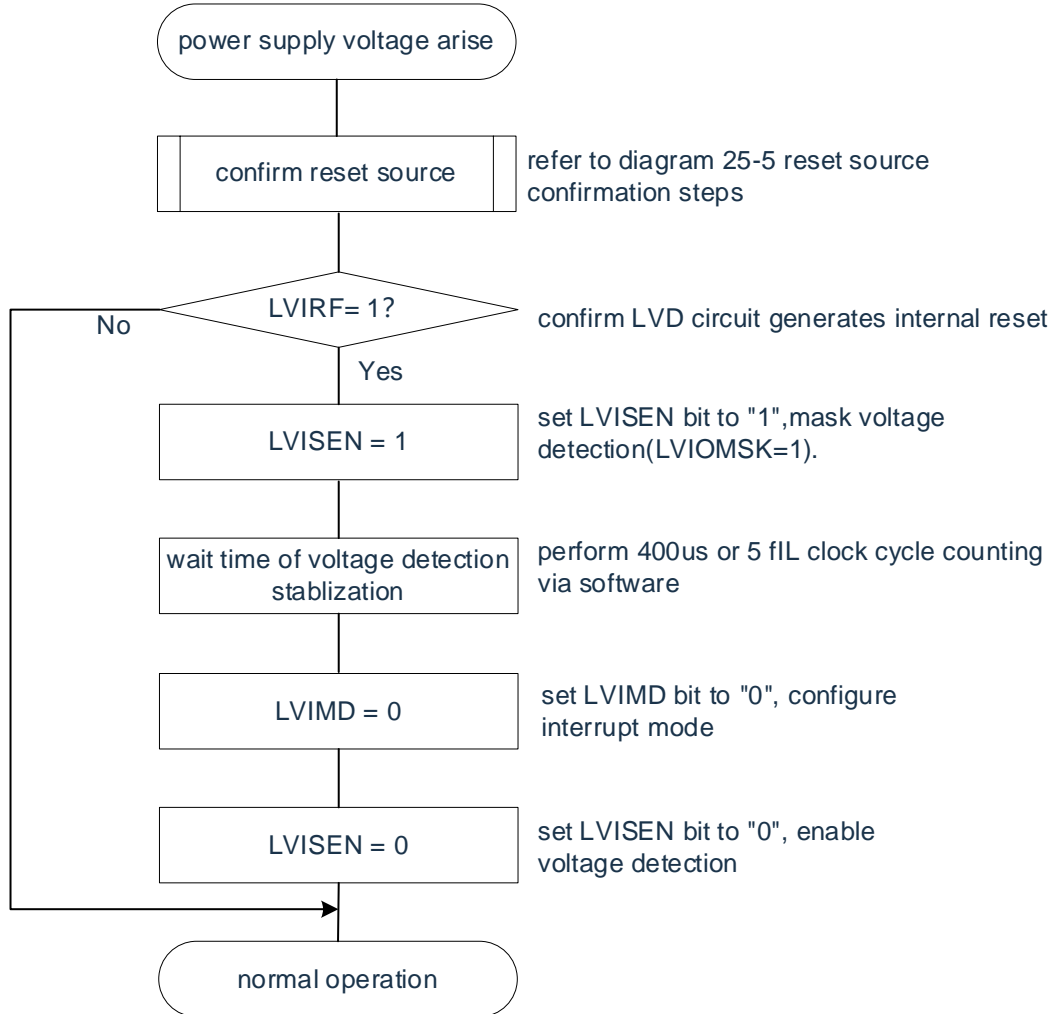
Figure 28-7: Setting procedure for confirmation/reset of operating voltage



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 28-8.

Figure 28-8: Initial setting procedure for interrupt & reset mode



Remark:  $F_{IL}$ : Low-speed internal oscillator clock frequency

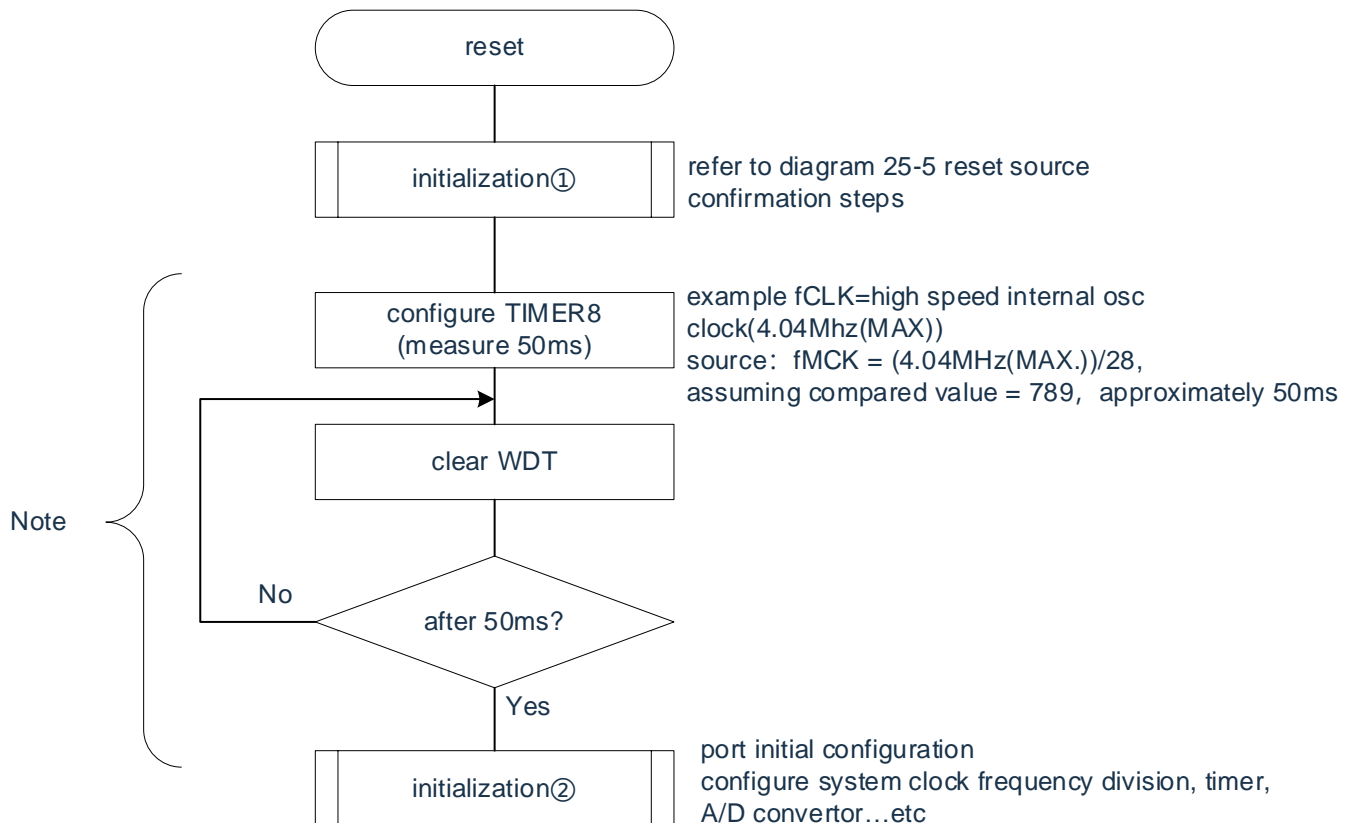
## 28.5 Cautions for voltage detection circuits

### (1) About voltage fluctuations when power is turned on

For systems where the supply voltage (VDD) fluctuates over time near the LVD detection voltage, it is possible to repeatedly enter the reset state and the reset release state. Through the following processing, any setting can be set to reset to the time when the microcontroller starts running.

<processing> after the reset is released, the initial setting of the port, etc. must be made after waiting for different supply voltage fluctuation times in each system by using the software counter of the timer.

Figure 28-9: Example of software processing when the supply voltage fluctuation near the LVD detection voltage does not exceed 50ms

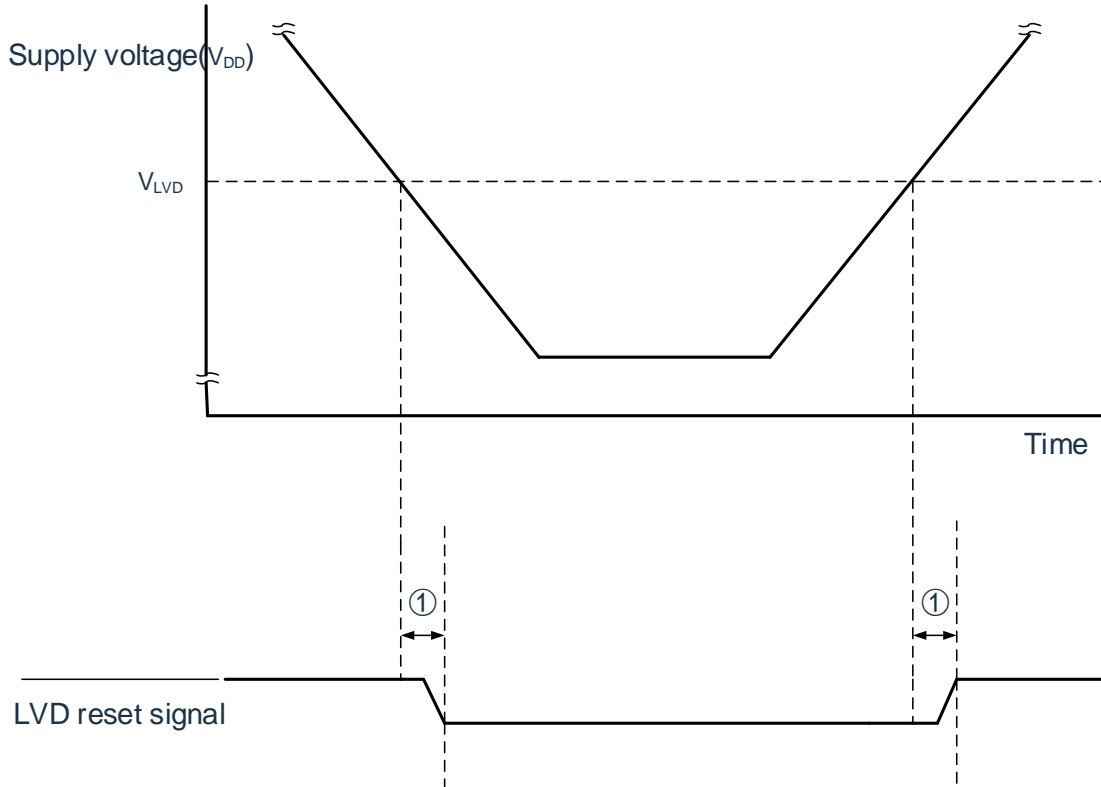


Note: If the reset occurs again during this period, it is not transferred to initialization processing (2).

(2) Delay from generating the LVD reset source to generating or releasing the LVD reset

A delay occurs from the time the supply voltage ( $V_{DD}$ ) < LVD detection voltage ( $V_{LVD}$ ) is met to the time the LVD reset is generated. Similarly, a delay occurs from the time the LVD detection voltage ( $V_{LVD}$ ) ≤ the supply voltage ( $V_{DD}$ ) to the time the LVD reset is released (see figure Figure 28-10).

Figure 28-10: : Delay from generation of LVD reset source to generation or release of LVD reset



Remark: ①: Detection delay (300us(MAX.))

(3) About turning on the power when LVD is set to OFF

When LVD is set to OFF, an external reset must be performed using the RESETB pin.

When performing an external reset, the RESETB pin must be input low for at least 10us. If an external reset is performed while the supply voltage is rising, the power must be turned on after a low level is input to the RESETB pin, and must be held low for at least 10us within the operating voltage range shown in the AC characteristics of the datasheet, followed by a high level.

(4) About the operating voltage drop when LVD is set to OFF and set to LVD interrupt mode

If the operating voltage drops when LVD is set to OFF and LVD interrupt mode is set, it must be reset by deep sleep mode transfer or external reset before the operating voltage falls below the operating voltage range shown in the AC characteristics of the data sheet. When restarting operation, it is necessary to verify that the supply voltage is restored in the operating voltage range



# Chapter 29 Security Feature

## 29.1 Overview

In order to comply with IEC60730 and EC61508 safety standards, the CMS32H6157 has the following built-in security features.

The purpose of this function is to safely stop the operation when a fault is detected through the self-diagnosis of the microcontroller.

(1) Flash CRC function (high-speed CRC, general-purpose CRC)

The data error of flash memory is detected by CRC operation. The following two CRCs can be used depending on the application and usage conditions.

- "High-speed CRC"...During the initialization program, it is possible to stop the CPU and check the entire code flash area at high speed.
- "General-purpose CRC"...can be used for multi-purpose checks during CPU operation, not limited to the code flash area.

(2) RAM parity error detection function

Detects parity errors when reading RAM data.

(3) SFR protection function

Prevents rewriting of SFR due to uncontrolled CPU.

(4) Frequency detection function

Self-testing of CPU/peripheral hardware clock frequency using a general-purpose timer unit.

(5) A/D test function

A/D converter self-test by A/D conversion of positive (+) reference voltage, negative (-) reference voltage, analog input channel (ANI), temperature sensor output and internal reference voltage output of A/D converter.

(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.

(7) Product unique identification register (128 bits)

## 29.2 Registers Used for Security Functions

The following registers are used for each function of the security function.

Register Name	Function
• Flash CRC control register (CRC0CTL). • Flash CRC Operation Result Register (PGCRCL).	Flash CRC operation function (High Speed CRC).
• CRC Input Register (CRCIN). • CRC Data Register (CRCD).	CRC arithmetic function (Universal CRC).
• RAM Parity Error Control Register (RPECTL).	RAM parity error detection function
• Special SFR Protection Control Register (SFRGD).	SFR protection function
• Timer input selection register 0 (TIS0).	Frequency detection function
• A/D Test Register (ADTES).	A/D test function
• Port Mode Select Register (PMS).	Digital output signal level detection function on input/output pins

For the contents of each register, see the section “29.3 Operation of security features”.

## 29.3 Operation of security features

### 29.3.1 Flash CRC operational function (high-speed CRC)

The IEC60730 standard requires verification of the data in the flash memory and recommends CRC as a means of verification. This high-speed CRC can check the entire code flash area in the initial setup (initialization) procedure.

High-speed CRC stops the CPU and reads 32 bits of data from the flash memory with one clock for operation. Therefore, it is characterized by a short time to complete the verification (e.g., 64KB flash: 512us@32MHz).

The CRC generation polynomial corresponds to the CRC-16-CCITT " $X^{16} + X^{12} + X^5 + 1$ ".

MSB of bit31→bit0 is operated first.

Remark: The operational result is different due to the general-purpose CRC is LSB-preferred.

#### 29.3.1.1 Flash CRC control register (CRC0CTL)

This is the register that sets the operation control and operation range of the high-speed CRC operator. The CRC0CTL register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

Figure29-1: 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

FEA6

FEA5

FEA4

FEA3

FEA2

FEA1

FEA0

CRC0EN

Operation control of high-speed CRC operator

0

Stops running.

1

Start the operation by executing the HALT instruction.

FEA6	FEA5	FEA4	FEA3	FEA2	FEA1	FEA0	high speed CRC calculation range
0	0	0	0	0	0	0	00000H ~ 1FFBH(8K-4byte)
0	0	0	0	0	0	1	00000H ~ 3FFBH(16K-4byte)
0	0	0	0	0	1	0	00000H ~ 5FFBH(24K-4byte)
0	0	0	0	0	1	1	00000H ~ 7FFBH(32K-4byte)
0	0	0	0	1	0	0	00000H ~ 9FFBH(40K-4byte)
0	0	0	0	1	0	1	00000H ~ BFFBH(48K-4byte)
0	0	0	0	1	1	0	00000H ~ DFFBH(56K-4byte)
0	0	0	0	1	1	1	00000H ~ FFFBH(64K-4byte)
0	0	0	1	0	0	0	00000H ~ 11FFBH(72K-4byte)
0	0	0	1	0	0	1	00000H ~ 13FFBH(80K-4byte)
0	0	0	1	0	1	0	00000H ~ 15FFBH(88K-4byte)
0	0	0	1	0	1	1	00000H ~ 17FFBH(96K-4byte)
0	0	0	1	1	0	0	00000H ~ 19FFBH(104K-4byte)
0	0	0	1	1	0	1	00000H ~ 1BFFBH(112K-4byte)
0	0	0	1	1	1	0	00000H ~ 1DFFBH(120K-4byte)
0	0	0	1	1	1	1	00000H ~ 1FFFBH(128K-4byte)
1	x	x	x	x	x	x	00000H ~ 1EFFBH(124K-4byte)

Remark: The expected value of the CRC operation result for comparison must be stored in the last 4 bytes of flash memory in advance, so the operation range is the range minus 4 bytes.

### 29.3.1.2 Flash CRC operation result register (PGCRCL)

This is the register where the results of high-speed CRC operations are stored

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

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

Figure29-2: Format of flash CRC result register(PGCRCL)

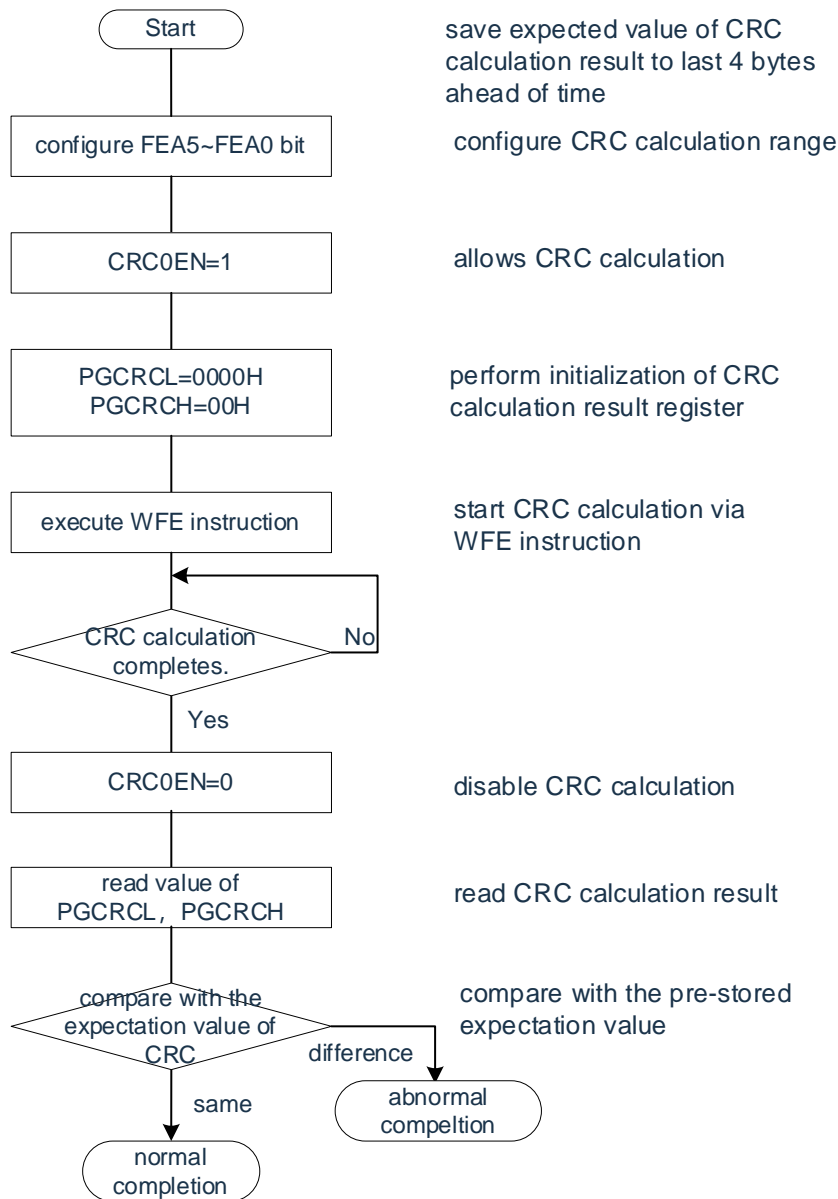
Address:	0x40021812		After reset: 0000H				R/W	
Symbol	15	14	13	12	11	10	9	8
PGCRCL	PGCRC15	PGCRC14	PGCRC13	PGCRC12	PGCRC11	PGCRC10	PGCRC9	PGCRC8
	7	6	5	4	3	2	1	0
	PGCRC7	PGCRC6	PGCRC5	PGCRC4	PGCRC3	PGCRC2	PGCRC1	PGCRC0
	PGCRC15~0		Operation result of high-speed CRC					
	0000H~FFFFH		Save the results of high-speed CRC operations.					

Notice: The PGCRCL register can only be written if the CRC0EN (bit 7 of CRC0CTL register) is "1".

The flowchart of the flash memory CRC operation function (high-speed CRC) is shown in Figure 29-3.

<Operation Flow>

Figure 29-3: Flow chart of flash CRC operation function (high-speed CRC)



Notice:

1. Only the code flash is used for CRC operations.
2. The expected value of the CRC operation must be stored in the area after the operation range in the code flash.

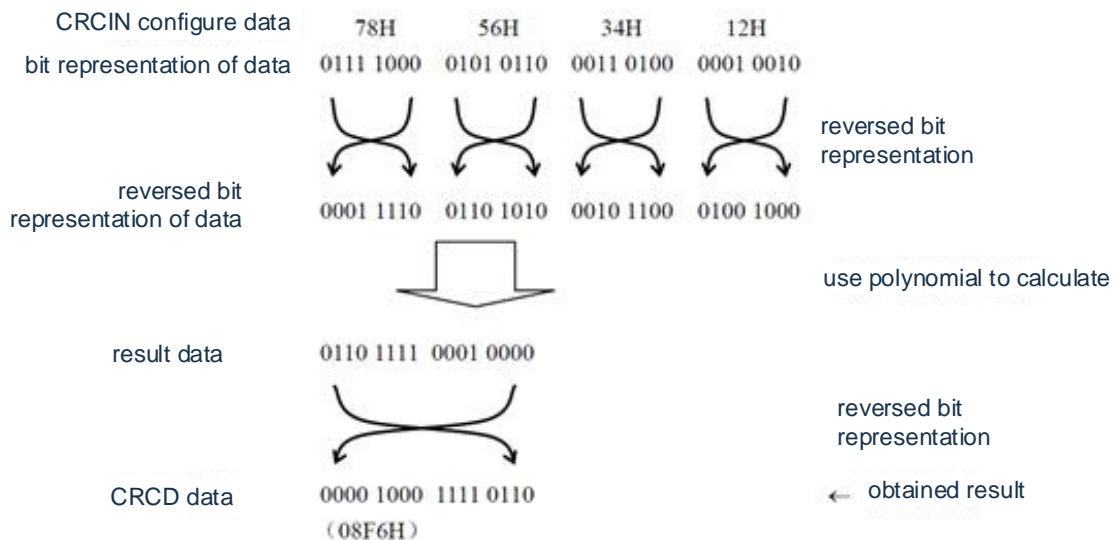
## 29.3.2 CRC operation function (general-purpose CRC)

In order to ensure safety during operation, the IEC61508 standard requires that the data need to be confirmed even during CPU operation.

This general-purpose CRC can perform CRC operations as a peripheral function during CPU operation. The general-purpose CRC is not limited to the code flash area and can be used for multi-purpose inspection. Specify the data to be confirmed by software (user program). The CRC operation function in sleep mode can only be used during DMA transfer.

The CRC operation function can be used in either the main system clock operation mode or the secondary system clock operation mode.

The CRC-generated polynomial uses CRC-16-CCITT's " $X^{16}+X^{12}+X^5+1$ ". Because the communication is carried out with LSB priority, the calculation is performed after the bit order of the input data is reversed. For example, if the data "12345678H" is sent from LSB, it is as follows as "78H", "56H", "34H", etc. The order of "12H" is written to the CRCN register, and the value of "08F6H" is obtained from the CRCRD register. This is the result of a CRC operation on the following bit order after the bit order of the data "12345678H" is reversed.



Notice: During the execution of the program, because the modulator overrides the setting 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.

### 29.3.2.1 CRC input register (CRCIN)

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

Figure29-4: Format of CRC input register

Address: 40041400H		After reset: 00H		R/W				
Symbol	7	6	5	4	3	2	1	0
CRCIN								
bit7~0		Function						
00H~FFH		Data input						

### 29.3.2.2 CRC data register (CRCD)

This is a register that holds the results of the general-purpose CRC operation. The range that can be set is "0000H~FFFFH".

After writing the CRCIN register, it goes through a CPU/peripheral hardware clock ( $F_{CLK}$ ) to save the CRC operation result to the CRCD Register. The CRCD register is set by a 16-bit memory manipulation instruction.

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

Figure 29-5: Format ofCRC data register (CRCD)

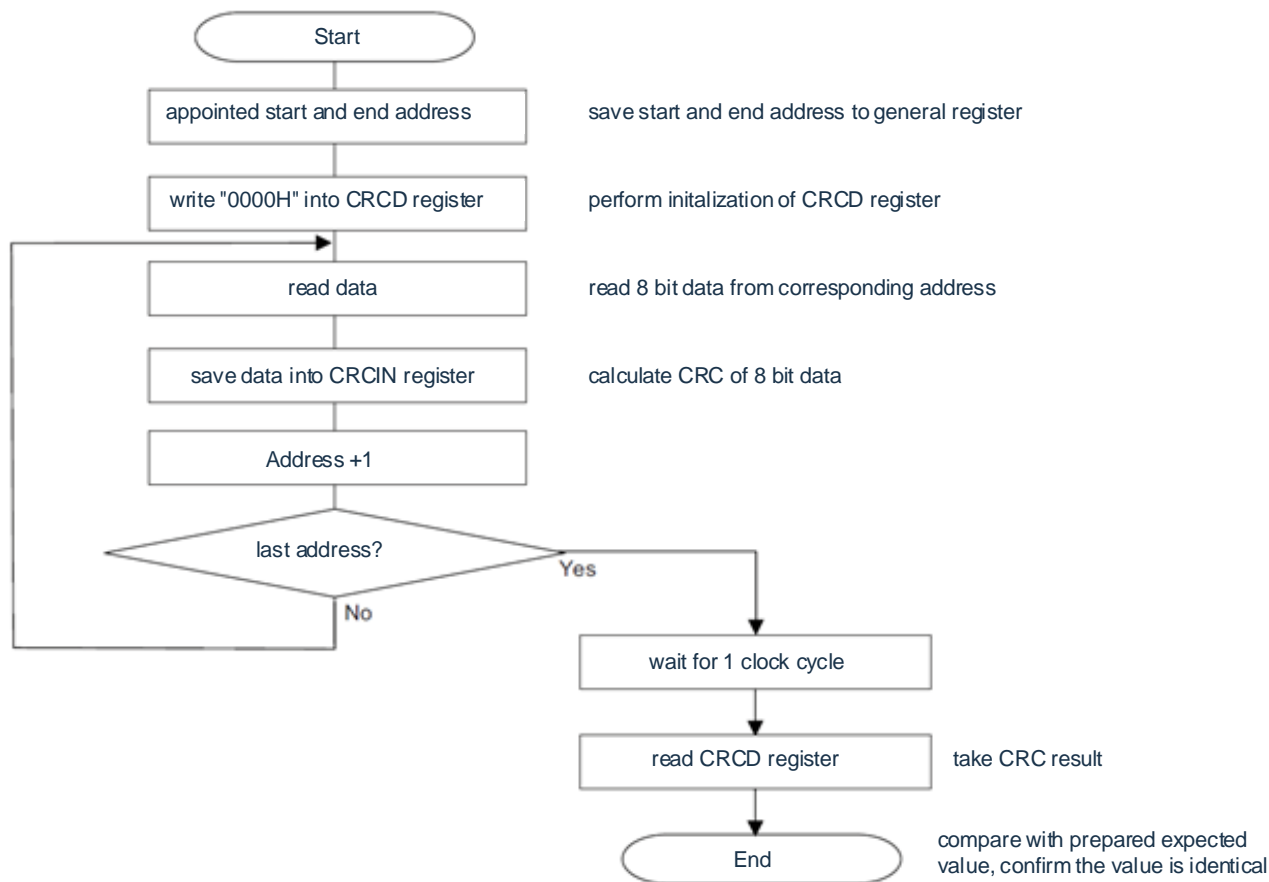
Address: 40041402H				After reset: 0000H				R/W								
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCD																

Notice:

1. To read the write value of a CRCD register, the CRCD register must be read before writing the CRCAN register.
2. If the write operation of the CRCD register competes with the saving of the operation result, the write operation is ignored.

# <Operation process>

Figure29-6: Flowchart of CRC operation function (Universal CRC)





### 29.3.3 RAM parity error detection function

The IEC60730 standard requires validation of RAM data. Therefore, the CMS32H6157's RAM is appended with 1 bit of parity bits for every 8 bits. The RAM parity error detection feature appends parity bits when writing data, checks parity bits when reading data, and can generate resets when parity errors occur.

#### 29.3.3.1 RAM parity error control register (RPECTL)

This register controls the error confirmation bit for parity and the reset due to parity error. The RPECTL register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "00H".

Figure29-7: Format of RAM parity error control register (RPECTL)

Address: 40020425H

After reset: 00H

R/W

Symbo

7

6

5

4

3

2

1

0

IRPECTL

RPERDIS

0

0

0

0

0

0

0

RPEF

RPERDIS	Mask flag for parity error reset
0	Enable parity error reset to occur.
1	Parity error reset is prohibited.

RPEF	Parity error status flag
0	No parity errors occurred.
1	A parity error occurred.

Notice: Attach parity bits when writing data, and check parity bits when reading data.

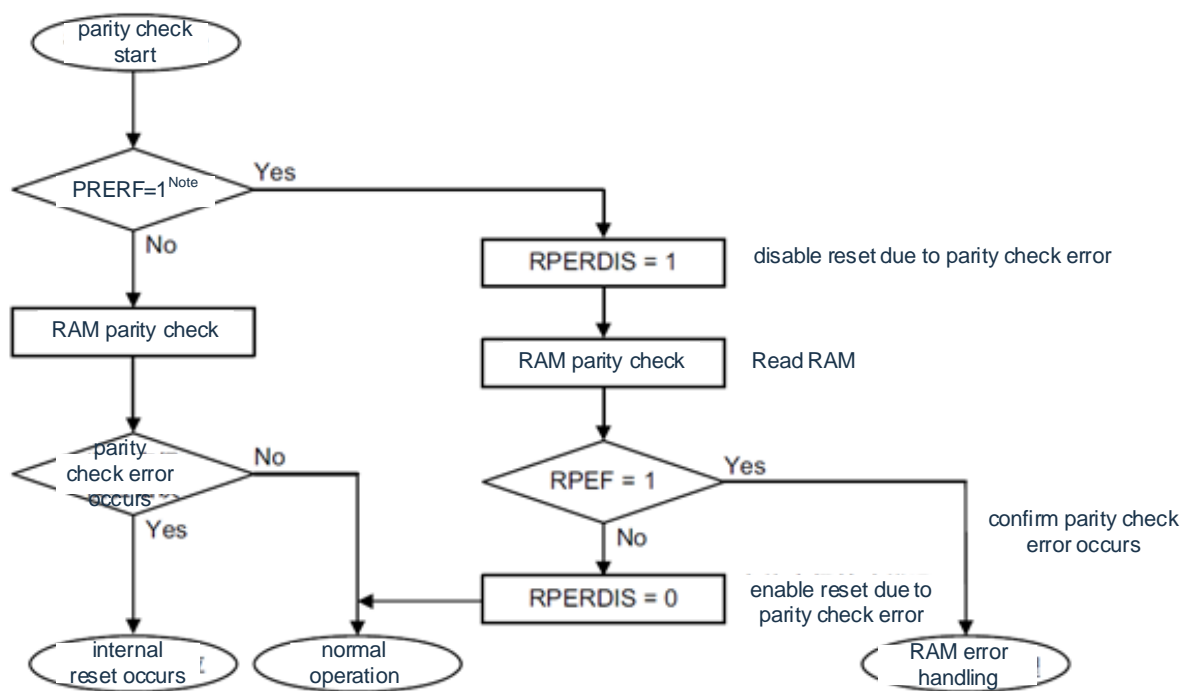
Therefore, to allow ram parity error reset (RPERDIS=0) to be generated, the "RAM region used" must be initialized when the data is accessed and before reading the data.

Because it is running on a pipeline, the CPU performs a read-ahead, and a RAM parity error may occur due to the uninitialized RAM area before reading the RAM area used. Therefore, to allow for a RAM parity error reset (RPERDIS=0), it is necessary to execute instructions from the RAM region to the "RAM region used." +10 bytes" of the region is initialized.

Remark:

1. The initial state is to allow parity error reset (RPERDIS=0).
2. Even if set to disable parity error reset (RPERDIS=1), the RPEF flag is set to "1" when a parity error occurs. If the RPEF bit is set to allow parity test error reset (RPERDIS=0) in the state where the RPEF bit is "1", the RPERDIS is cleared to "0" A parity error is generated when reset.
3. Set the RPF flag of the RUBTL register to "1" due to a RAM parity error, and reset the source by writing "0" or resetting all the sources Flag "0". When the RPF flag is "1", the RPEF flag remains in the "1" state even if the RAM without parity error is read.
4. The scope of RAM parity detection does not include general-purpose registers.

Figure29-8: Flow of RAM parity check



Note: For confirmation of internal reset of RAM parity errors, please refer to “Chapter 26 Reset Function”.

## 29.3.4 SFR protection function

In order to ensure safety during operation, the IEC61508 standard requires that even if the CPU is out of control, it is necessary to protect important SFR from being rewritten. The SFR protection function is used to protect data from the control registers of the comparator function, port function, interrupt function, clock control function, voltage detection circuitry, and RAM parity error detection function.

If the SFR protection function is set, the write operation of the protected SFR is invalid, but it can be read normally.

### 29.3.4.1 SFR protect control register(SFRGD)

This register controls whether the SFR protection function is valid.

The SFR protection function uses GCOMP bits, GPORT bits, GINT bits, and GCSC bits.

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

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

Figure29-9: Format of the SFR Protection Control Register (SFRGD)

Address: 40040C08H

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 the control registers of the comparator function
0	Invalid. The control register of comparator function can be read and written.
1	Valid. The control register for the port function is invalid and can be read. [Protected SFR] COMPMDR, COMPFIR, COMPOCR, CMPSEL0, CMPSEL1, C0REFS, C1REFS, CMP0HY, CMP1HY, AMPCTL, AMPDAC

GPORT	Protection of control registers for port functions
0	Invalid. The control registers of port functions can be read and written.
1	Valid. The write operation of the control register of the port function is invalid, and it can be read. [Protected SFRs] ISCLCD, SEGn(n=0~3), PMxx, PUxx, PDxx, POMxx, PMCxx, PSETxx, PCLRxx, PREADxx(xx=A, B, C, D, H) Note

GINT	Protection of registers for interrupt function
0	Invalid. Can read and write the control register of interrupt function.
1	Valid. The write operation of the control register of the interrupt function is invalid, and it can be read. [Protected SFRs] IFxx, MKxx, PRxx, EGPx, EGNx

GCSC	Protection of control registers for clock control functions, voltage detection circuitry, and RAM parity error detection functions
0	Invalid. The control registers of clock control function, voltage detection circuit and RAM parity error detection function can be read and written.
1	Valid. The write operation of the control registers of the clock control function, voltage detection circuit and RAM parity error detection function is invalid and readable. [Protected SFRs] CMC, CSC, OSTs, CKC, PERx, OSMC, LVIM, LVIS, RPECTL

Note: Pxx (port registers) are not protected.

## 29.3.5 Frequency detection function

The IEC60730 standard requires confirmation that the oscillation frequency is normal.

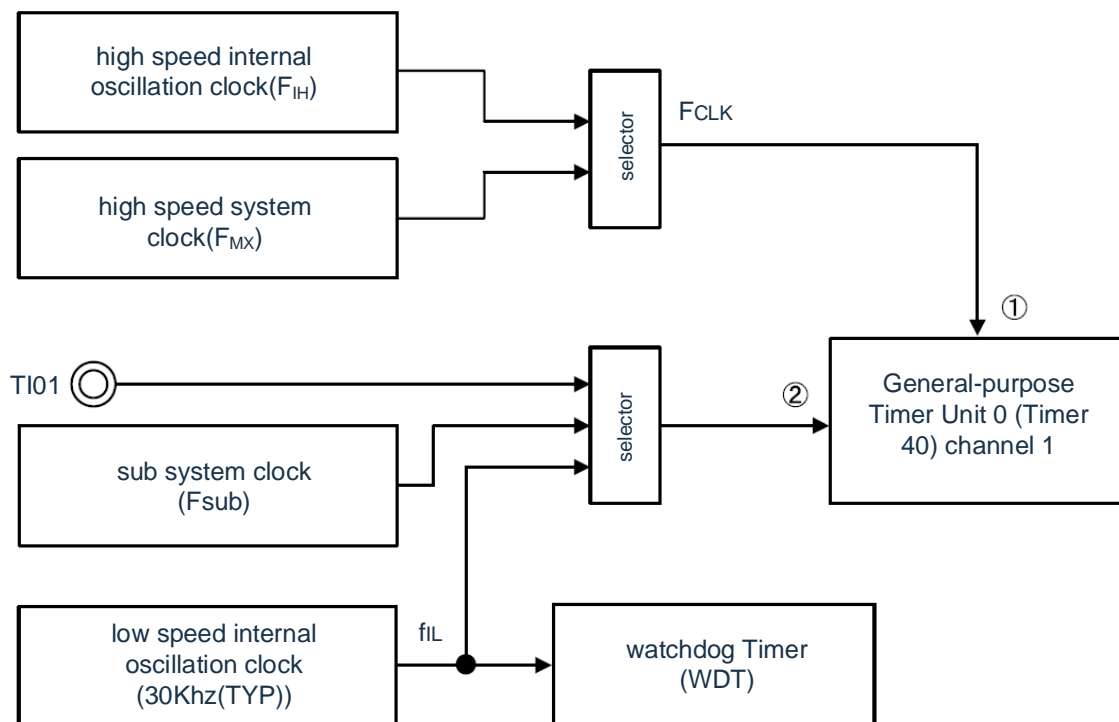
The frequency detection function uses the CPU/peripheral hardware clock frequency ( $F_{CLK}$ ) and can determine whether the ratio of the two clocks is correct by measuring the Channel 1 input pulse of Timer8.

However, if a certain clock or 2 clocks stop oscillating, it is impossible to judge the ratio relationship of the 2 clocks.

<Clock to be compared>

- ① CPU/peripheral hardware clock frequency ( $F_{CLK}$ ):
  - High-speed internal oscillator clock ( $F_{IH}$ )
  - High-speed system Clock ( $F_{MX}$ )
- ② Channel 1 input of Timer8:
  - Timer input for channel 1 (TI01)
  - Low-speed internal oscillator clock ( $F_{IL}$ : 30KHz(TYP.))
  - Subsystem clock ( $F_{SUB}$ )<sup>Note</sup>

Figure29-10: Structure of frequency detection function



If the measurement result of the input pulse interval is an abnormal value, it can be judged as "clock frequency abnormality". For the measurement method of the input pulse interval, refer to "5.8.4 Operation as input pulse interval measurement".

Note: Only products with built-in subsystem clocks can be selected.

### 29.3.5.1 Timer input selection register0(TIS0)

For a register description, refer to Timer input output selection register (TIOS0) for register description.

## 29.3.6 A/D test function

The IEC60730 standard requires testing of A/D converters. This A/D test function is performed on the positive (+) reference voltage, negative (–) reference voltage, analog input channel (ANI), output voltage of the temperature sensor, and internal reference voltage of the A/D converter A/D conversion to confirm that the A/D converter is running normally.

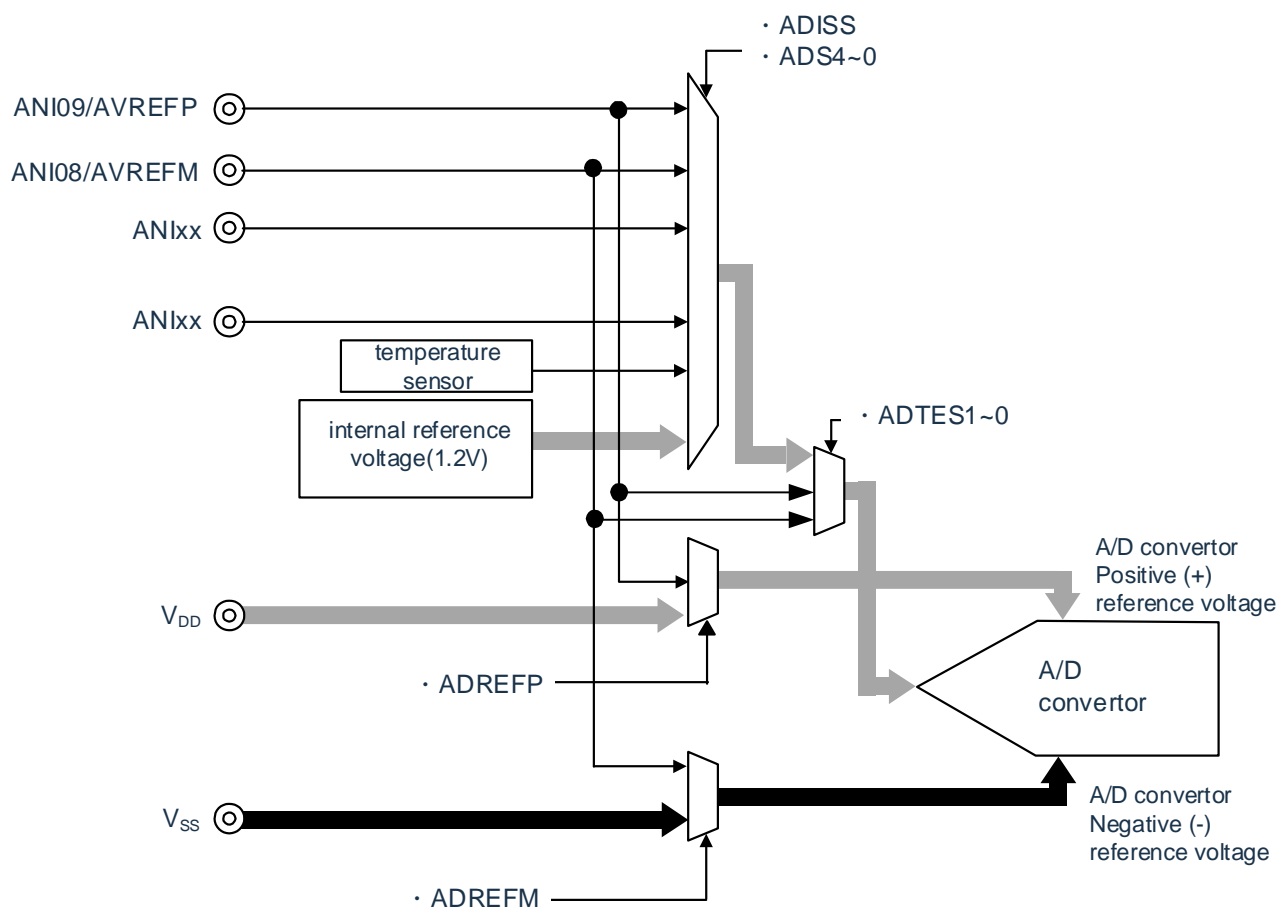
The analog multiplexer can be confirmed by following these steps:

- ① The ANIx pin is selected as the A/D conversion object (ADTES1, ADTES0=0, 0) through the ADTES register.
- ② A/D conversion of the ANIx pins (conversion result 1-1).
- ③ The negative (–) reference voltage of the A/D converter is selected by the ADTES register as the A/D conversion object (ADTES1, ADTES0=1, 0)
- ④ A/D conversion of the negative (–) reference voltage of the A/D converter (conversion result 2-1).
- ⑤ The ANIx pin is selected as the A/D conversion object (ADTES1, ADTES0=0, 0) through the ADTES register.
- ⑥ A/D conversion of the ANIx pins (conversion result 1-2).
- ⑦ The positive (+) reference voltage of the A/D converter is selected by the ADTES register as the A/D conversion object (ADTES1, ADTES0=1, 1)
- ⑧ A/D conversion of the positive (+) reference voltage of the A/D converter (conversion result 2-2).
- ⑨ The ANIx pin is selected as the A/D conversion object (ADTES1, ADTES0=0, 0) through the ADTES register.
- ⑩ A/D conversion of the ANIx pins (conversion result 1-3).
- ⑪ Verify that "Conversion Result 1-1", "Conversion Result 1-2", and "Conversion Result 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 conversion result is all "1". With the above steps, you can select an analog multiplexer and confirm that the wiring is not broken.

Remark:

1. During the conversion process of (1) to (10), if the analog input voltage is variable, other methods must be used to confirm the analog multiplexer.
2. The conversion result contains an error, so the error must be considered appropriately when comparing the conversion result.

Figure29-11: Structure of the A/D test function



### 29.3.6.1 A/D test register (ADTES)

This register selects the positive (+) reference voltage, negative (–) reference voltage, analog input channel (ANlxx), output voltage of the temperature sensor, and internal reference voltage (1.45V) of the A/D converter ) 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 full scale, select a positive (+) reference voltage as the A/D conversion object.

For the register description, refer to 11.2.10.

### 29.3.6.2 Analog input channel specification register (ADS)

This register specifies the input channel for the analog voltage converted from A/D.

To measure ANlxx, temperature sensor output, or internal reference voltage (1.45V) via the A/D test function, the A/D test register (ADTES) must be placed at "00H".

For the register description, refer to 11.2.7.

## 29.3.7 Product Unique Identification Register

The unique identification of the product is perfect 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 security of the code in the flash memory.
- Used to activate a bootstrap process with a safety mechanism

The reference number provided by the 128-bit product unique identifier is unique to any BAT32 microcontroller in any case. Under any circumstances, the user cannot modify this identity

Base address: 0x0050\_0E4C

Address offset: 0x00

Read-only, whose values are written at the factory

U_ID[31:0]
------------

Address offset: 0x04

Read-only, whose values are written at the factory

U_ID[63:32]
-------------

Address offset: 0x08

Read-only, whose values are written at the factory

U_ID[95:64]
-------------

Address offset: 0x0C

Read-only, whose values are written at the factory

U_ID[127:96]
--------------

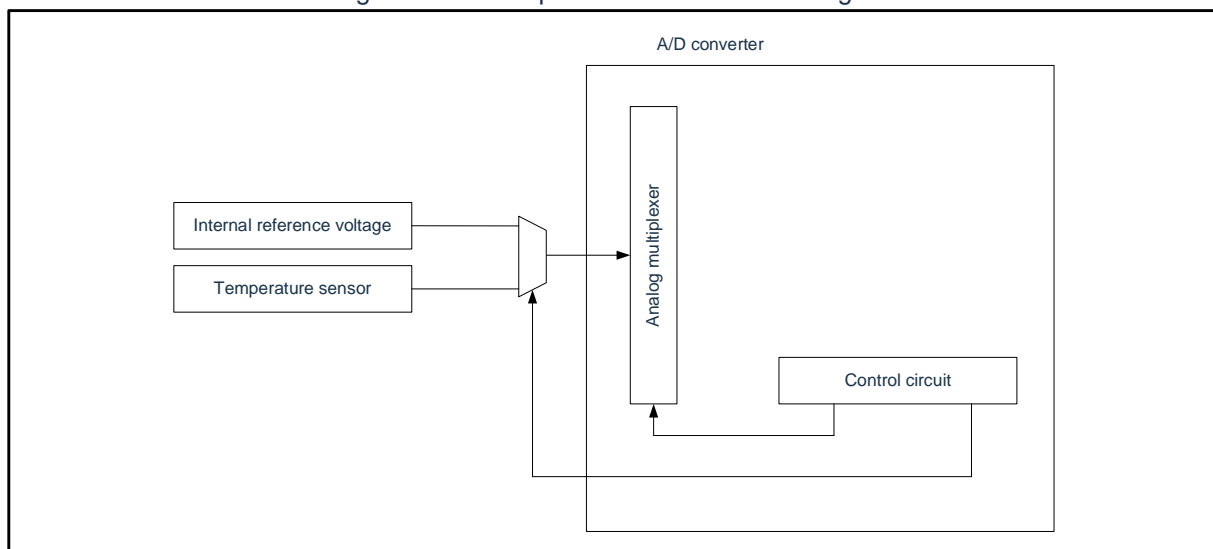


# Chapter 30 Temperature sensor and internal reference voltage

## 30.1 Temperature sensor

The on-chip temperature sensor measures and monitors the core temperature of the product, thus ensuring reliable operation of the product. The voltage output by the temperature sensor is proportional to the core temperature, and there is a linear relationship between the voltage and temperature. Its output voltage is supplied to the ADC for conversion. Figure 30-1 shows a block diagram of a temperature sensor.

Figure 30-1: Temperature sensor block diagram



## 30.2 Register for temperature sensor

### 30.2.1 Temperature sensor calibration data register TSN25

Address: 0x500C6C

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

Read-only registers for recording calibration data for temperature sensors<sup>1</sup> are automatically loaded when powered on or reset is initiated, with each chip having its own calibration data.

### 30.2.2 Temperature sensor calibration data register TSN85

Address: 0x500C68

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

Read-only registers for recording calibration data for temperature sensors<sup>2</sup>, loaded automatically when powered on or reset started, with each chip having its own calibration data.

## 30.3 Instructions for using the temperature sensor

### 30.3.1 How temperature sensors are used

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 at temperature measurement (V)

T1: Temperature measured experimentally at the first point (°C)

V1: Voltage output when the temperature sensor measures T1 (V)

T2: Temperature measured experimentally at the second point (°C)

V2: Voltage output when the temperature sensor measures T2 (V)

Slope: The temperature slope of the temperature sensor (V/°C),  $\text{slope} = (V_2 - V_1) / (T_2 - T_1)$ .

Different sensors have different characteristics, so we recommend measuring the following two different sample temperatures:

1. Use an A/D converter to measure the voltage V1 output of the temperature sensor at temperature T1.
2. Use an A/D converter to measure the voltage V2 output by the temperature sensor at the second temperature T2.
3. The temperature slope ( $\text{slope} = (V_2 - V_1) / (T_2 - T_1)$ ) is calculated from the two results
4. Subsequently, the temperature ( $T = (V_s - V_1) / \text{slope} + T_1$ ) is obtained by substituting the slope into the formula for the temperature characteristics.

### 30.3.2 How to use temperature sensor

Method 1: In this product, the TSN25 register stores the voltage conversion value (CAL25) of the temperature sensor measured at  $T_A=25^{\circ}\text{C}$  and  $V_{DD}=3.0\text{v}$ . The TSN85 register stores the voltage conversion value of the temperature sensor measured at  $T_A=85^{\circ}\text{C}$  and  $V_{DD}=3.0\text{v}$  (CAL85). Using these two sets of values, the temperature slope can be calculated:

$$\text{slope} = (V_2 - V_1) / (85 - 25).$$

$$V_1 = 3.0 \times \text{CAL25} / 4096 \text{ [V]}$$

$$V_2 = 3.0 \times \text{CAL125} / 4096 \text{ [V]}$$

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 ( $^{\circ}\text{C}$ )

Vs: Output voltage (V) of the temperature sensor at T temperature obtained using the A/D converter

Method 2: If you use the temperature slope given in "Electrical Characteristics", you can directly calculate the measured temperature using the following formula:

$$T = (V_s - V_1) / \text{slope} + 25 \text{ [}^{\circ}\text{C]}$$

Remark: This method produces temperatures with lower accuracy than those measured by Method 1.

## 30.4 Internal reference voltage

The Band-gap reference voltage ( $V_{BG}$ ) is an internal fixed reference voltage, independent of the external supply. The  $V_{BG}$  output can be converted by connecting it internally to an ADC, so that  $V_{DD}$  can be calculated from the ADC conversion of  $V_{BG}$ .

Due to the process, the  $V_{BG}$  of each chip is slightly different, so the calculation of  $V_{DD}$  may be a little bit different. This product has built-in A/D conversion result of  $V_{BG}$  when  $V_{DD} = 3.0V$ , user can calculate the voltage value of  $V_{DD}$  accurately by this value and current A/D conversion result of  $V_{BG}$ .

### 30.4.1 VDD calibration data register VDDCDR

Address: 0x500C64

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	After reset:	R/W
VDDCDR	-	-	-	-	VDDCDR[11:0]												-	R

Read-only register, used to record the A/D conversion result of  $V_{BG}$  when  $V_{DD} = 3.0V$ , automatically loaded when power on or reset start, and each chip has its own calibration data.

### 30.4.2 Instructions for using the internal reference voltage

In this product, the VDDCDR register stores the internal reference voltage conversion value measured at  $T_A = T_J = 25^\circ C$  and  $V_{DD} = 3.0V$ . Using this result, the user can calculate the current  $V_{DD}$  voltage according to the following formula.

$$V_{DD} = 3.0V * VDDCDR / CALVBG$$

CALVBG: The current  $V_{BG}$  conversion result obtained by using the A/D converter, using  $V_{BG}$  as the ADC input channel.

# Chapter 31 Option Bytes

## 31.1 Function of option bytes

CMS32H6157 flash memory 000C0H~000C4H, 500004H is the option byte region.

The option bytes consist of user option bytes (000C0H~000C2H) and flash data protection option bytes (000C3H, 500004H) composition. When powered on or reset is initiated, the specified function is set with reference to the option byte. When using this product, the following functions must be set by the option byte. For bits that do not have configuration capabilities, you cannot change the initial value.

Note Regardless of whether or not to use each function, you must set the option byte.

### 31.1.1 User option bytes (000C0H~000C2H)

#### (1) 000C0H

- Operation of watchdog timer
  - Allow or disallow the operation of counters.
  - Allow or stop the operation of the counter in sleep/deep sleep mode.
- The setting of the overflow timer of the watchdog
- The setting during which the watchdog timer window is opened
- Watchdog timer interval interrupt
  - Use or not use interval interrupts.

#### (2) 000C1H

- Setting of LVD operating mode
  - Interrupt & Reset mode
  - Reset mode
  - Interrupt mode
  - LVD is OFF (using the external reset input of the RESETB pin).
- Setting of LVD detection level (VLVDH, VLVDL, VLVD)

Notice: When the supply voltage rises, the reset state must be maintained by voltage detection circuits or external resets before the supply voltage reaches the operating voltage range shown in the AC characteristics of the data sheet; When the supply voltage drops, it must be reset by transferring in deep sleep mode, voltage detection circuitry, or external reset before the supply voltage falls below the operating voltage range.

The operating voltage range depends on the setting of the user option byte (000C2H).

#### (3) 000C2H

- Frequency setting of high-speed internal oscillator
  - Choose from 1MHz~32MHz.

### 31.1.2 Flash data protection option bytes (000C3H, 500004H)

- Control of flash data protection when debugging on-chip
  - Level0: Allows read/write/erase operations on flash data via debugger
  - Level1: Allows chip full erase of flash data via debugger, read and write operations are not allowed.
  - Level2: Operations on flash data via debugger are not allowed.

## 31.2 Format of the user option bytes

Figure 31-1: Format of user option bytes (000C0H)

Address: 000C0H

Symbol	7	6	5	4	3	2	1	0
	WDTINT	WINDOW1	WINDOW0	WDTON	WDCS2	WDCS1	WDCS0	WDSTBYO

WDTINT	Interval interrupt of watchdog timer
0	Interval interrupt is not used.
1	When 75% of the overflow time + 1/2F <sub>IL</sub> is reached, an interval interrupt is generated.

WINDOW1	WINDOW0	When watchdog timer window opens <sup>Note 1</sup>
0	-	Settings are disabled.
1	0	75%
1	1	100%

WDTON	Controlling counter operation of watchdog timer
0	Disable counter operation (stop counting after the reset is released).
1	Enable counter operation (start counting after the reset is released).

WDCS2	WDCS1	WDCS0	Overflow time of watchdog timer (F <sub>IL</sub> =20KHz(MAX.))
0	0	0	2 <sup>6</sup> /f <sub>IL</sub> (3.2ms)
0	0	1	2 <sup>7</sup> /F <sub>IL</sub> (6.4ms)
0	1	0	2 <sup>8</sup> /F <sub>IL</sub> (12.8ms)
0	1	1	2 <sup>9</sup> /F <sub>IL</sub> (25.6ms)
1	0	0	2 <sup>11</sup> /F <sub>IL</sub> (102.4ms)
1	0	1	2 <sup>13</sup> /F <sub>IL</sub> (409.6ms)
1	1	0	2 <sup>14</sup> /F <sub>IL</sub> (819.2ms)
1	1	1	2 <sup>16</sup> /F <sub>IL</sub> (3276.8ms)

WDSTBYON	Counter run control (sleep mode) of the watchdog timer
0	In sleep mode, stop the counter from running <sup>Note 2</sup> .
1	In sleep mode, counters are allowed to run.

Note 1: When the WDSTBYON bit is "0", regardless of the values of the WINDOW1 bit and the WINDOW0 bit, it is 100% during window opening.

Remark: F<sub>IL</sub> : Clock frequency of the low-speed internal oscillator.



Figure31-2: Format of user option bytes (000C1H) (1/4)

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD settings (interrupt & reset mode)

Detect voltage			Setting value of option byte						
V <sub>LVDH</sub>		V <sub>LVDL</sub>	VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode Setting	
Rising	Falling	Falling						LVIMDS1	LVIMDS0
1.98V	1.94V	1.84V	0	0	1	1	0	1	0
2.09V	2.04V					0	1		
3.13V	3.06V					0	0		
2.61V	2.55V	2.45V		1	0	1	0		
2.71V	2.65V					0	1		
3.75V	3.67V					0	0		
2.92V	2.86V	2.75V		1	1	1	0		
3.02V	2.96V					0	1		
4.06V	3.98V					0	0		
-			It is prohibited to set values other than the above.						

Notice: You must write "1" to bit4.

Remark:

- For details of LVD circuit, please refer to "Chapter 28 Voltage Detection Circuit".
- The detection voltage is the TYP value. For details, please refer to the LVD circuit characteristics in the data sheet.

Figure31-2: Format of user option bytes (000C1H) (2/4)

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

• LVD setting (reset mode)

Detect voltage		Setting value of option byte						
V <sub>LVD</sub>		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		
-		It is prohibited to set values other than the above.						

Notice: You must write "1" to bit4.

Remark:

- For details of LVD circuit, please refer to "Chapter 28 Voltage Detection Circuit".
- The detection voltage is the TYP value. For details, please refer to the LVD circuit characteristics in the data sheet.

Figure 31-2: Format of user option bytes (000C1H) (3/4)

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD setting (interrupt mode)

Detect voltage		Setting value of option byte						
V <sub>LVD</sub>		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		
-		It is prohibited to set values other than the above.						

Notice: You must write "1" to bit4.

Remark:

- For details of LVD circuit, please refer to "Chapter 28 Voltage Detection Circuit".
- The detection voltage is the TYP value. For details, please refer to the LVD circuit characteristics in the data sheet.

Figure 31-2: Format of user option bytes (000C1H) (4/4)

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- Setting when LVD is OFF (external reset input using the RESETB pin)

Detect voltage		Setting value of option byte						
V <sub>LVDH</sub>							Mode Setting	
rising	falling						LVIMDS1	LVIMDS0
—	—	1	x	x	x	x	x	1
—		It is prohibited to set values other than the above.						

Notice:

- You must write "1" to bit4.
- When the power supply voltage rises, the reset state must be maintained through the voltage detection circuit or external reset before the power supply voltage reaches the working voltage range shown in the AC characteristics of the data sheet; When the supply voltage drops, it must be reset through sleep mode transfer, voltage detection circuitry, or external reset before the supply voltage falls below the operating voltage range.

The operating voltage range depends on the setting of the user option byte (000C2H/010C2H).

Remark:

- x: Ignore
- For details of LVD circuit, please refer to "Chapter 28 Voltage Detection Circuit".
- The detection voltage is the TYP value. For details, please refer to the LVD circuit characteristics in the data sheet.

Figure 31-3: Format of user option bytes (000C2H)

Address: 000C2H

7	6	5	4	3	2	1	0
1	1	1	0	1	FRQSEL2	FRQSEL1	FRQSEL0

FRQSEL2	FRQSEL1	FRQSEL0	High-speed internal oscillator frequency	
			F <sub>HOCO</sub>	F <sub>IH</sub>
0	0	0	32MHz	32MHz
0	0	1	32MHz	16MHz
0	1	0	32MHz	8MHz
0	1	1	32MHz	4MHz
1	0	0	32MHz	2MHz
1	0	1	32MHz	1MHz
Others:			Settings are disabled.	

Notice:

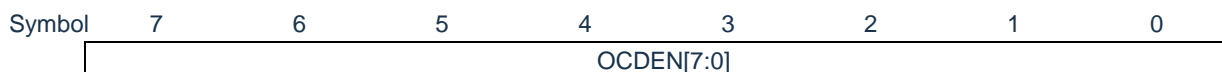
1. You must write "1" to bit7~ 5.
2. The operating frequency range and working voltage range vary according to the operating modes of the flash memory. For details, please refer to the AC characteristics of the data sheet.

## 31.3 Format of flash data protection option bytes

The format of the flash data protection option bytes is as follows.

Figure 31-4: Format of flash data protection option bytes

Address: 000C3H



Address: 500004H



OCDM	OCDEN	Control of flash data protection
3C	C3	Operations on flash data via debugger are not allowed.
Values other than 3C	C3	Chip full erase of flash data via debugger is allowed, read and write operations are not allowed.
Others		Allows read/write/erase operations on flash data via debugger

# Chapter 32 FLASH Control

## 32.1 Description of FLASH control

This product contains a 128KByte FLASH memory, divided into 256 sectors, each sector has a capacity of 512Byte. This module supports erase, program and read operations of this memory. In addition, this module supports the protection for FLASH memory erase and write protection for control registers.

## 32.2 Structure of FLASH memory

FFFF_FFFFH	Reserved
E00F_FFFFH	Cortex-M0+ dedicated peripheral resource area
E000_0000H	Reserved
4005_FFFFH	Peripherals Resource Area
4000_0000H	Reserved
2000_1FFFFH	SRAM (up to 8KB)
2000_0000H	Reserved
0050_0BFFFH	Data Flash 2.5KB
0050_0200H	Reserved
0001_FFFFH	Main flash memory area (up to 128KB)
0000_0000H	

## 32.3 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 erase time control register (FLSERCNT).
- Flash write time control register (FLPROCNT).
- Flash mode time control register

### 32.3.1 Flash write protection register (FLPROT)

Flash protection registers are used to protect the FLASH operating control registers.

Address: 0x40020020      After reset: 00000000H      R/W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	PRKEY[7:1]							WRP

Symbol

FLPROT

WRP	The operation register (FLOPMD1/FLOPMD2) is write-protected
0	Rewriting FLOPMD1/FLOPMD2 is not permitted
1	Rewriting of FLOPMD1/FLOPMD2 is allowed

PRKEY[7:1]	WRP write protection
78h	Rewriting of WRP is allowed
Others	Rewriting WRP is not allowed



## 32.3.2 FLASH operation control register (FLOPMD1,FLOPMD2)

Flash operation control registers to set the erase and write operations of FLASH.

Address: 0x40020004    After reset: 00000000H    R/W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	FLOPMD1[7:0]						

SymbolFLOPMD1

Address: 0x40020008    After reset: 00H    R/W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	FLOPMD2[7:0]						

SymbolFLOPMD2

FLOPMD1	FLOPMD2	OPERATE
55	AA	Erase
AA	55	Write
00	00	Read
Others		Settings are disabled.

### 32.3.3 Flash erase control register (FLERMD)

Flash erase control register 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	OPERATE
0	0	Sector erase
1	0	Settings are disabled
0	1	Chip erasure <sup>Note</sup>
1	1	Settings are disabled

Note: Chip Erase only erases the code flash area, not the data flash area. And chip wipe does not support hardware validation.

### 32.3.4 Flash status register (FLSTS)

The status register allows to query the status of the FLASH controller.

Address: 0x40020000

After reset: 00H

R/W

Symbol	7	6	5	4	3	2	1	0
FLSTS	0	0	0	0	0	0	0	OVF <sup>Note</sup>

OVF	FLASH erase operation finished flag
0	FLASH erase operation not completed
1	FLASH erase operation completed

Note: The OVF needs to be cleared by writing "1" in software. If it is not cleared, the next erase operation cannot be performed.

EVF	FLASH erase the hardware validation error flag
0	No hardware verification error after FLASH erase
1	Hardware verification error occurs after FLASH erase

Note: EVF needs to be cleared by writing "1" in software.

### 32.3.5 Flash full-chip erase time control register (FLCERCNT)

The FLCERCNT register allows you to set the FLASH full erase time.

Address: 0x40020010

After reset: indefinite R/W

Symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLCERCNT	load	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	FLCERCNT[9:0]									

Load	Selection of erase time settings <sup>Note</sup>
0	Use the hardware-set erase time
1	Use the software-set erase time (F- <b>LCERCNT</b> [9:0]).

Note: When the master clock is an internal high-speed OCO, or when the external input clock ≤20M, the hardware time can be set without setting FLCERCNT.

FLCERCNT[9:0]	Software erase time setting
Chip erase time = (CERCNT*2048*T <sub>clk</sub> ), need to meet the hardware requirement of >20ms	

## 32.3.6 Flash sector erase time control register (FLSERCNT)

The FLSERCNT register allows you to set the FLASH full erase time.

Address: 0x40020014				After reset: indefinite R/W												
Symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLSERCNT	load	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	FLSERCNT[9:0]									

Load	Selection of erase time settings <sup>Note</sup>
0	Use the hardware-set erase time
1	Use the software-set erase time (FLSERCNT [9:0]).

Note: If the master clock is internal high-speed OCO or external input clock  $\leq 20\text{M}$ , you can use hardware to set the time without setting FLSERCNT.

FLSERCNT[9:0]	Software erase time setting
sector erase time = (SERCNT*256*Tfclk), which meets the hardware requirement of $>4\text{ms}$	

### 32.3.7 Flash write time control register (FLPROCNT)

The FLPROCNT register allows you to set the FLASH WORD write time.

Address: 0x4002001C				After reset: indefinite				R/W								
Symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLPROCNT	Load1	-	-	-	-	-	-	FLPGSCNT[8:0]								
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Load0	-	-	-	-	-	-	FLPROCNT[8:0]								

Load0	Write Time (Tprog) Setting	Note
0	Use the hardware-set write time	
1	Use the software-set erase time (FLPROCNT [9:0]).	

Note: When the master clock is an internal high-speed OCO, or if the external input clock  $\leq 20\text{M}$ , the hardware time can be set without setting FLPROCNT

FLPROCNT[8:0]	Software erase time setting
Write time = $(\text{PROCNT} \times 4 \times \text{Tfclk})$ , which meets the hardware requirement of $>24\mu\text{s}$	

Load1	Write Action Setup Time (Tpgs) Setting	Note
0	Use the hardware-set write action to establish the time	
1	Use the software-set erase time (FLPGSCNT8:0).	

Note: When the master clock is an internal high-speed OCO, or when the external input clock  $\leq 20\text{M}$ , the hardware time can be set without setting FLPGSCNT.

FLPGSCNT[8:0]	Software erase time setting
Write operation setup time = $(\text{PGSCNT} \times \text{Tfclk})$ , which meets the hardware requirements of $>5\text{ us}$	

### 32.3.8 Flash erase protection control register (FLSECPR)

When the Sector is protected, all erase operations on the Sector are invalid.

Address: 0x40020210 After reset: 00000000H R/W

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY[15:8]								-	-	-	SECPR[3:0]				

Symbol

FLSECPR

KEY	Register SECPR write protection
5AA5F1	Allow rewriting SECPR[3:0]
Others:	No rewriting of SECPR[3:0] is allowed

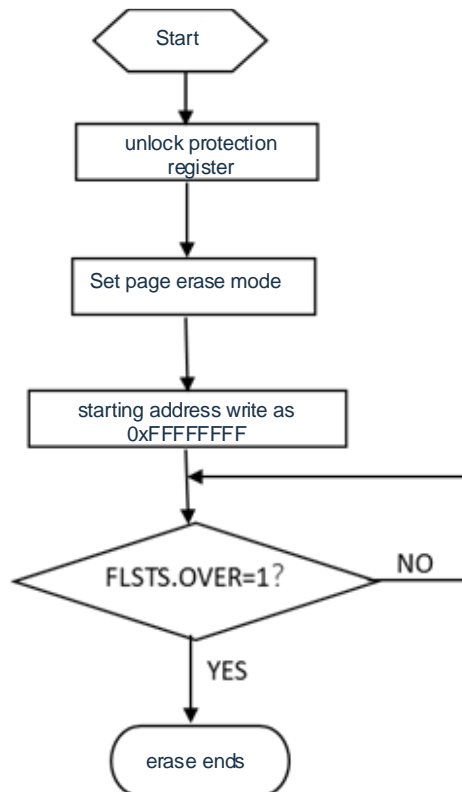
SECPR	Register SECPR write protection
0001	The 8 sectors of 00_0000H~00_0FFFH cannot be erased
0010	The 16 sectors of 00_0000H~00_1FFFH cannot be erased
0011	The 32 sectors of 00_0000H~00_3FFFH cannot be erased
0100	The 64 sectors of 00_0000H~00_7FFFH cannot be erased
0101	128 sectors from 00_0000H to 00_FFFFH cannot be erased
Others:	Sector is not protected, erasure allowed

## 32.4 How to operate FLASH

### 32.4.1 Sector erase

Sector erase, and the erase time are implemented by hardware or can be configured by FLSERCNT. The operation flow is as follows:

- (1) Set up FLERMD. ERMD0 is 1'b0, select the sector erase mode;
- (2) Set FLPROT to 0xF1 to de-protect FLOPMD. Then set FLOPMD1 to 0x55 and FLOPMD2 to 0xAA
- (3) Write arbitrary data to the first address of the wiped target sector. Example: \*((unsigned long \*) 0x00000200)=0xffffffff.
- (4) Software query status register FLSTS. OVF, when OVF=1, indicates that the erase operation is complete.
- (5) Before proceeding to the next operation, the software sets "1" to clear FLSTS.



## 32.4.2 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 up FLERMD. ERMD0 is 1'b 1, select chip erase mode;
- (2) Set FLPROT to 0xF1 to de-protect 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) Software query status register FLSTS. OVF, when OVF=1, indicates that the erase operation is complete.
- (5) Before proceeding to the next operation, the software sets "1" to clear FLSTS

## 32.4.3 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 up FLERMD. ERMD0 is 1'b 1, select chip erase mode;
- (2) Set FLPROT to 0xF1 to de-protect 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) Software query status register FLSTS. OVF, when OVF=1, indicates that the erase operation is complete.
- (5) Before proceeding to the next operation, the software sets "1" to clear FLSTS



## 32.5 Flash Read

The fastest finger frequency supported by flash built into this device is 20 MHz. When the HCLK frequency exceeds 20 MHz, the hardware inserts a 1 wait period when the CPU accesses flash.

## 32.6 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 internal 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 operation is performed from within FLASH, the CPU stops taking the finger and the hardware automatically waits for the operation to complete before proceeding to the next instruction. If the operation is performed from the RAM, the CPU does not stop taking the finger and can now proceed to the next instruction.
- While FLASH is in programming, if the CPU executes the instruction to enter a deep sleep, the system will wait for the programming action to end before entering a deep sleep.

## Chapter 33 Revision History

Version	Date	Revised Content
V0.1.0	February 2023	Initial Version
V0.1.1	March 2023	<ol style="list-style-type: none"><li>1) Correct a clerical error in 20.3.2</li><li>2) Modify some section descriptions in 32.3.3 Flash erase control register (FLERMD)</li><li>3) Modify the register value in section 32.3.4 Flash status register (FLSTS)</li><li>4) Modify some section descriptions in section 32.4.1 Sector erase</li><li>5) Modify the low-speed internal oscillator clock value</li><li>6) Modify the formula in section 15.2.3 Operational amplifier digital-to-analog control register (OPADAC)</li></ol>