



BAT32A2x9 User Manual

An ultra-low-power 32-bit microcontroller based on the ARM® Cortex®-M0+

Rev 1.00

Please be reminded about following CMS's policies on intellectual property

* Cmsemicon Limited(denoted as 'our company' for later use) has already applied for relative patents and entitled legal rights. Any patents related to CMS's MCU or other products is not authorized to use. Any individual, organization or company which infringes our company's intellectual property rights will be disabled and stopped by our company through any legal actions, and our company will claim the lost and required for compensation of any damage to the company.

* The name of Cmsemicon Limited and logo are both trademarks of our company.

* Our company preserve the rights to further elaborate on the improvements about products' function, reliability and design in this manual. However, our company is not responsible for any usage about this manual. The applications and their purposes in this manual are just for clarification, our company does not guarantee that these applications are feasible without further improvements and changes, and our company does not recommend any usage of the products in areas where people's safety is endangered during accident. Our company's products are not authorized to be used for life-saving or life support devices and systems.our company has the right to change or improve the product without any prior notification, for latest news, please visit our website: www.mcu.com.cn

Documentation Instructions

This manual is a technical reference manual for BAT32A239/BAT32A279 microcontroller products, and the technical reference manual is an application note on how to use this series of products, including the structure, functional description, and function description of each functional module. Details such as operating modes and register configuration.

The Technical Reference Manual is a description of all functional modules in this series of products, please refer to the data sheet for the description of the characteristics of the product (i.e. the function carrying situation).

The data sheet information is as follows:

BAT32A239xx: BAT32A239_datasheet_vx.xx. pdf

BAT32A279xx: BAT32A279_datasheet_vx.xx. pdf

Usually in the early stage of chip selection, the first thing to see is to look at the data sheet to evaluate whether the product can meet the functional requirements of the design; After basically selecting the required product, it is necessary to check the technical reference manual to determine whether the working mode of each functional module meets the requirements; When determining that the selection enters the programming design phase, a detailed technical reference manual is required to understand the specific implementation of each function and the register configuration. Refer to the data sheet when designing your hardware for information such as voltage, current, drive capability, and pin assignment.

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

BAT32A2x9 User Manual Chapter List

	BAT32A239xx	BAT32A279xx
Chapter 1: CPU	•	•
Chapter 2: Pin Functions	•	•
Chapter 3: System structure	•	•
Chapter 4: Clock generation circuit	•	•
Chapter 5: Hardware Divider	•	•
Chapter 6: Universal timer unit Timer4/8	•	•
Chapter 7: TimerA	•	•
Chapter 8: TimerB	•	•
Chapter 9: TimerC	•	•
Chapter 10: TimerM	•	•
Chapter 11: Real-Time Clock	•	•
Chapter 12: 15-Bit Interval Timer	•	•
Chapter 13: Clock output/buzzer output control circuitry	•	•
Chapter 14: Watchdog Timer	•	•
Chapter 15: A/D Converter	•	•
Chapter 16: D/A Converters	•	•
Chapter 17: Comparator	•	•
Chapter 18: Programmable Gain Amplifier	•	•
Chapter 19: Universal Serial Communication Unit	•	•
Chapter 20: Serial Interface IICA	•	•
Chapter 21: Serial Interface SPI	×	•
Chapter 22: CAN Control	•	•
Chapter 23: LCD Bus Interface	×	•
Chapter 24: Enhanced DMA	•	•
Chapter 25: Linkage Controller	•	•
Chapter 26: Interrupt function	•	•
Chapter 27: Key Interrupt Function	•	•
Chapter 28: Standby Function	•	•
Chapter 29: Reset Function	•	•
Chapter 30: Power-on Reset Circuit	•	•
Chapter 31: Voltage Detection Circuit	•	•
Chapter 32: Safety Functions	•	•
Chapter 33: Temperature sensor and internal reference voltage	•	•
Chapter 34: Option Bytes	•	•
Chapter 35: Flash Control	•	•

Index

Documentation Instructions	2
Chapter 1 CPU	25
1.1 overview	25
1.2 Cortex-M0+ core features	25
1.3 Debug features	25
1.4 SWD interface pin	27
1.5 ARM reference documentation	28
Chapter 2 Pin function	29
2.1 Port capabilities	29
2.2 Port multiplexing function	29
2.3 Registers that control port functionality	30
2.3.1 Port Mode Register (PMxx)	33
2.3.2 Port register (Pxx)	34
2.3.3 Port Set Control Register (PSETxx)	35
2.3.4 Port Clear Control Register (PCLRxx)	36
2.3.5 Pull-up resistor selection register (PUxx)	37
2.3.6 Port input mode register (PIMxx)	38
2.3.7 Port output mode register (POMxx)	39
2.3.8 Port Mode Control Register (PMCxx)	40
2.3.9 Port read-back register (PREADxx)	41
2.3.10 Peripheral I/O redirect register 0 (PIOR0)	42
2.3.11 Peripheral I/O redirect register 1 (PIOR1)	43
2.3.12 Peripheral I/O redirect register 2(PIOR2)	44
2.3.13 Peripheral I/O redirect register 3(PIOR3)	45
2.3.14 Peripheral I/O redirect register 4 (PIOR4)	46
2.3.15 Global Digital Input Disable Register (GDIDIS)	47
2.4 Unused pin handling	48
2.5 Register settings when using the multiplexing function	49
2.5.1 Basic principle when using the multiplexing feature	49
2.5.2 Examples of register settings using the port function and the multiplexing function	50
Chapter 3 System structure	69
3.1 overview	69
3.2 System address partition	70
3.3 Peripheral address assignment	72
Chapter 4 Clock generation circuit	73
4.1 Function of the clock generation circuit	73
4.2 Structure of the clock generation circuit	75

4.3	Control Registers of the clock generation circuit.....	78
4.3.1	Clock operating mode control register (CMC).....	78
4.3.2	System Clock Control Register (CKC).	80
4.3.3	Clock Operating State Control Register (CSC).....	81
4.3.4	PLL Control Register (PLLCR) for System Clock.....	82
4.3.5	The state register (OSTC) of the oscillation settling time counter	83
4.3.6	Oscillation settling time selection register (OSTS).....	85
4.3.7	Peripheral enable registers 0, 1, 2, 3 (PER0, PER1, PER2, P ER2 PER3)	86
4.3.8	Subsystem clock supply mode control register (OSMC).....	92
4.3.9	Frequency Selection Register (HOCODIV) for high-speed internal oscillators	93
4.3.10	Trimming Register (HIOTRM) for high-speed internal oscillator.....	94
4.3.11	Subsystem Clock Select Register (SUBCKSEL)	95
4.3.12	Master System Clock Control Register (MCKC).	96
4.4	System clock oscillation circuit.....	97
4.4.1	X1 oscillation circuit	97
4.4.2	XT1 oscillation circuit.....	97
4.4.3	High-speed internal oscillator	101
4.4.4	Low-speed internal oscillator	101
4.5	Clock generation circuit operation	102
4.6	Clock control	104
4.6.1	Example of setting up a high-speed internal oscillator.....	104
4.6.2	Example of setting up the X1 oscillation circuit.....	106
4.6.3	Example of setting up the XT1 oscillation circuit.....	107
4.6.4	State transition graph of the CPU clock	108
4.6.5	Conditions before CPU clock transfer and processing after transfer.....	114
4.6.6	Time required to switch between the CPU clock and the master system clock	116
4.6.7	Conditions before clock oscillation stops.....	117
4.7	High-speed internal vibration correction function.....	118
4.7.1	High-speed internal vibration self-adjustment function	118
4.7.2	Register description	119
4.7.3	Description of the operation.....	120
4.7.4	Precautions for use.....	123
4.8	Vibration stop detection circuit	124
4.8.1	Vibration stops the function of the detection circuit.....	124
4.8.2	Composition of the vibration-stop detection circuit.....	124
4.8.3	The register used by the oscillator stops detection circuit	125
4.8.4	The operation of the vibration stop detection circuit	127
4.8.5	Precautions for vibration stop detection function	128

Chapter 5	Hardware divider.....	129
-----------	-----------------------	-----

5.1	Features.....	129
5.2	Feature description	129
5.3	Registers for the hardware divider	129
5.3.1	DIVIDEND.....	130
5.3.2	Divider (DIVISAR).....	130
5.3.3	Quotient	130
5.3.4	REMAINDER	130
5.3.5	Status register (STATUS).....	131
Chapter 6	Universal timer unit Timer4/8.....	132
6.1	Functions of the universal timer unit.....	134
6.1.1	Stand-alone channel operation function.....	134
6.1.2	Multi-channel linkage operation function	136
6.1.3	8-bit timer operation function (channel 1 and channel 3 of unit 0 only).	137
6.1.4	LIN-bus support function (limited to channel 3 of unit 0).....	137
6.2	Structure of a universal timer unit.....	138
6.2.1	List of general-purpose timer unit registers	141
6.2.2	Timer count register mn (TCRmn).....	144
6.2.3	Timer data register mn (TDRmn).....	146
6.3	Control registers of the universal timer unit.....	147
6.3.1	Peripheral enable register 0 (PER0).	148
6.3.2	Timer clock selection register m (TPSm).	149
6.3.3	Timer mode register mn (TMRmn).	152
6.3.4	Timer status register mn (TSRmn).	156
6.3.5	Timer channel enable status register m (TEm).	157
6.3.6	Timer channel start register m(TSm).....	158
6.3.7	Timer channel stop register m(TTm).	159
6.3.8	Timer input and output selection register (TIOS0).	160
6.3.9	Timer output enable register m (TOEm).....	161
6.3.10	Timer output register m (TOm).	162
6.3.11	Timer output level register m(TOLm).	163
6.3.12	Timer output mode register m (TOMm).....	164
6.3.13	Input Select Control Register (ISC).	165
6.3.14	Noise filter enable registers (NFEN1/NFEN2).....	166
6.3.15	Registers that control the function of the timer input/output pin ports.....	167
6.4	The basic rules of the universal timer unit.....	168
6.4.1	The basic rules of the multi-channel linkage operation function	168
6.4.2	Timer channel start register m(TSm).....	170
6.4.3	The basic rules for the 8-bit timer to operate the function (limited to Channel 1 and Channel 3 of Unit 0).	171

6.5	The operation of the counter	172
6.5.1	Count clock (f_{TCLK}).	172
6.5.2	The start timing of the counter.....	174
6.5.3	The operation of the counter	175
6.6	Control of the channel output (TOMn pin).	180
6.6.1	Structure of the TOMn pin output circuit.....	180
6.6.2	Output setting of the TOMn pin	181
6.6.3	Considerations for channel output operation	182
6.6.4	one-time operation of the TOMn bit.....	186
6.6.5	About the timer interrupt and TOMn pin output when starting to count	187
6.7	Control of the timer input (TIMn).....	188
6.7.1	Structure of the TIMn pin input circuit.....	188
6.7.2	Noise filters	188
6.7.3	Considerations when operating channel input	189
6.8	Stand-alone channel operation of the universal timer unit.....	190
6.8.1	Operation as an interval timer/square wave output.....	190
6.8.2	Run as an external event counter	194
6.8.3	Operation as a divider.....	197
6.8.4	Operation as input pulse interval measurements.....	200
6.8.5	Operation as input signal high and low level width measurements	203
6.8.6	Runs as a delay counter.....	207
6.9	Multi-channel linkage operation function of the universal timer unit	210
6.9.1	Operation as a single-trigger pulse output function.....	210
6.9.2	Operates as a PWM function.....	217
6.9.3	Operation as a multi-PWM output function.....	224
6.10	Considerations when using a universal timer unit.....	232
6.10.1	Considerations when using timer outputs	232
Chapter 7 Timer A.....		233
7.1	Function of timer A.....	233
7.2	Structure of timer A.....	234
7.3	Controls the registers of timer A.....	235
7.3.1	Peripheral enable register 1 (PER1).	236
7.3.2	The subsystem clock provides a mode control register (OSMC).....	237
7.3.3	Timer A counts register 0 (TA0).	238
7.3.4	Timer A controls register 0 (TACR0).	239
7.3.5	Timer AI/O control register 0 (TAIOC0).....	240
7.3.6	Timer A controls register 0 (TAMR0).....	242
7.3.7	Timer A event pin select register 0 (TAISR0).....	243
7.3.8	Port mode register x (PMx).....	244

7.4	Operation of timer A	245
7.4.1	Overrides to reload registers and counters	245
7.4.2	Timer mode.....	246
7.4.3	Pulse output mode.....	247
7.4.4	Event counter pattern	248
7.4.5	Pulse width measurement mode	250
7.4.6	Pulse period measurement mode	251
7.4.7	Collaboration with EVENTC	252
7.4.8	Output settings for each mode	252
7.5	Considerations when using Timer A.....	253
7.5.1	Start and stop control of counting.....	253
7.5.2	Access to flags (TEDGF bits and TUNDF bits of the TACR0 register).....	253
7.5.3	Access to the Counting register.....	253
7.5.4	Change in Operational mode.....	254
7.5.5	Setup steps for TAO pins and TAIO pins	255
7.5.6	When timer A is not used	255
7.5.7	Timer A runs the stop of the clock.....	255
7.5.8	Setup steps for deep sleep mode (event counter mode).....	256
7.5.9	Functional limitations in deep sleep mode (event counter mode only).....	256
7.5.10	Forced count stop via the TSTOP bit	256
7.5.11	Digital filters	256
7.5.12	The case where fIL is selected as the count source	256
Chapter 8	Timer B.....	257
8.1	Function of timer B.....	257
8.2	Structure of timer B.....	258
8.3	Control registers of timer B.....	259
8.3.1	Peripheral enables register 1 (PER1).....	260
8.3.2	Timer B mode register (TBMR).	261
8.3.3	Timer B counts the control register (TBCNTC).	262
8.3.4	Timer B Control Register (TBCR).....	263
8.3.5	Timer B interrupt enable register (TBIER).....	264
8.3.6	Timer B status register (TBSR).	265
8.3.7	Timer BI/O Control Register (TBIOR).....	268
8.3.8	Timer B counter (TB).	270
8.3.9	Timer B universal registers A, B, C, D (TBGRA, TBGRB, TBGRC, TBGRD).....	271
8.3.10	Port registers and port mode registers	273
8.4	Operation of timer B	274
8.4.1	Common things about multiple patterns and features	274
8.4.2	Timer mode (input capture function)	279

8.4.3	Timer mode (output comparison function).....	282
8.4.4	PWM mode	286
8.4.5	Phase count mode.....	290
8.5	Timer B interrupt.....	293
8.6	Considerations when using timer B.....	295
8.6.1	Phase difference, overlap, and pulse width in phase count mode.....	295
8.6.2	Switching modes.....	295
8.6.3	Count the switching of the source	295
8.6.4	Setup steps for TBIO0 pins and TBIO1 pins	296
8.6.5	External clocks TBBCLK0 and TBBCLK1	296
8.6.6	Read and write access to SFR.....	296
8.6.7	Stop counting when the input snap runs	297
Chapter 9 Timer C		298
9.1	Function of timer C	298
9.2	Structure of timer C	299
9.3	Control registers of timer C.....	300
9.3.1	Peripheral enable register 1 (PER1).	300
9.3.2	Timer C count register (TC).....	301
9.3.3	Timer C count buffer register (TCBUF).	301
9.3.4	Timer C controls register 1 (TCCR1).....	302
9.3.5	Timer C controls register 1 (TCCR2).....	303
9.3.6	Timer C status register (TCSR).	304
9.4	Operation of timer C	305
9.4.1	Count the sources.....	305
9.4.2	Timer C starts counting the actions.....	305
9.4.3	Timer C counts stopped actions.....	308
9.4.4	Enter the capture motion	309
9.4.5	Timer C counts the reset action.....	310
9.4.6	Interrupt of timer C.....	312
9.5	Precautions when using Timer C	313
9.5.1	Read and write registers.....	313
9.5.2	Overflow interruption.....	313
9.5.3	Input capture and timer C count reset action	313
9.5.4	Timer C and Timer M, comparator 1 are linked	313
Chapter 10 Timer M.....		314
10.1	Function of timer M.....	314
10.2	Structure of timer M	315
10.3	Control register of timer M	316
10.3.1	Peripheral enable register 1 (PER1).	317

10.3.2	Timer M EVENTC register (TMELC).....	318
10.3.3	Timer M Start Register (TMSTR).....	319
10.3.4	Timer M mode register (TMMR).....	320
10.3.5	Timer M PWM function select register (TMPMR).....	321
10.3.6	Timer M function control register (TMFCR).....	322
10.3.7	Timer M output master enable register 1 (TMOER1).....	323
10.3.8	Timer M output main enable register 2 (TMOER2).....	324
10.3.9	Timer M output control register (TMOCR).....	325
10.3.10	The timer M digital filter function selects register i (TMDFi) (i=0, 1).....	328
10.3.11	Timer M controls register i (TMCRI) (i=0, 1).....	330
10.3.12	Timer M I/O control register Ai (TMIORAi) (i =0, 1).....	335
10.3.13	Timer M I/O control register Ci (TMIORCi) (i=0, 1).....	337
10.3.14	Timer M status register 0 (TMSR0).....	339
10.3.15	Timer M status register 1 (TMSR1).....	345
10.3.16	Timer M interrupt enable register i (TMIERi) (i= 0, 1).....	351
10.3.17	The timer MPWM function outputs level control register i (TMPOCRi) (i=0, 1).....	352
10.3.18	Timer M counter i(TMi) (i=0, 1).....	353
10.3.19	Timer M General Purpose registers Ai, Bi, Ci, Di.....	355
10.3.20	Port mode registers (PMxx, PMCxx).....	364
10.4	Common things about multiple patterns.....	365
10.4.1	Counting sources.....	365
10.4.2	The buffer operation.....	366
10.4.3	Synchronous Operation.....	369
10.4.4	Forced cutoff of the pulse output.....	370
10.4.5	Event inputs from the Event Linkage Controller (EVENTC).....	372
10.4.6	Event output to the Event Linkage Controller (EVENTC)/Data Transfer Controller (DMA).....	372
10.5	Operation of timer M.....	373
10.5.1	Enter the capture function.....	373
10.5.2	Output comparison function.....	378
10.5.3	PWM function.....	384
10.5.4	Reset synchronous PWM mode.....	388
10.5.5	Complementary PWM mode.....	391
10.5.6	PWM3 mode.....	395
10.6	Timer M interrupt.....	398
10.7	Considerations when using timer M.....	401
10.7.1	Read and write access to SFR.....	401
10.7.2	Switching modes.....	401
10.7.3	Counting sources.....	402
10.7.4	Enter the capture function.....	402

10.7.5	Configuration steps (i=0, 1) for TMIOAi, TMIOBi, TMIOCi, TMIODi pins	402
10.7.6	External clock TMCLK	403
10.7.7	Complementary PWM mode	403
10.8	PWMOP	408
10.8.1	Features of PWMOP	409
10.8.2	Registers for PWMOP	409
10.8.3	Operation of PWMOP	416
10.8.4	Precautions	437
Chapter 11	Real-time clock	438
11.1	The function of a real-time clock	438
11.2	The structure of the real-time clock	438
11.3	Control Registers the real-time clock	440
11.3.1	Peripheral enable register 0 (PER0).	441
11.3.2	Real-time clock selection register (RTCCL).	442
11.3.3	Real-time clock control register 0 (RTCC0).	443
11.3.4	Real-time clock control register 1 (RTCC1).	444
11.3.5	Clock Error Correction Register (SUBCUD).	446
11.3.6	Seconds Count Register (SEC).	447
11.3.7	Minute Count Register (MIN).	447
11.3.8	Hour Count Register (HOUR).	448
11.3.9	Day count register (DAY).	450
11.3.10	Week Count Register (WEEK).	451
11.3.11	Month count register (MONTH).	452
11.3.12	Year Count Register (YEAR).	452
11.3.13	Alarm clock minute register (ALARMWM).	453
11.3.14	Alarm hour register (ALARMWH).	453
11.3.15	Alarm Clock Week Register (ALARMWW).	453
11.3.16	Port mode registers and port registers	454
11.4	Operation of a real-time clock	455
11.4.1	The operation of the real-time clock begins	455
11.4.2	Start the transfer of sleep mode after running.	456
11.4.3	Read and write to the real-time clock counter	457
11.4.4	Alarm settings for the real-time clock	459
11.4.5	1Hz output of the real-time clock	460
11.4.6	An example of clock error correction for a real-time clock	461
Chapter 12	15-bit interval timer	463
12.1	Function of 5-bit interval timer	463
12.2	Structure of the 15-bit interval timer.	463
12.3	control Registers of the 15-bit interval timer.	464

12.3.1	Peripheral enable register 0 (PER0).	464
12.3.2	Real-time clock selection register (RTCCL).	465
12.3.3	Control Register (ITMC) for 15-bit interval timers	466
12.4	15-bit interval timer operation	467
12.4.1	Operation sequence of the 1 5-bit interval timer	467
12.4.2	After returning from sleep mode, the operation of the counter begins and then transition to sleep mode again	468
Chapter 13	Clock output/buzzer output control circuitry	469
13.1	the function of controls circuitry of Clock output/buzzer output	469
13.2	Structure of the clock output/buzzer output control circuit	470
13.3	control Registers of the clock output/buzzer output control circuitry	470
13.3.1	Clock output select register n (CKSn).	470
13.3.2	Control Registers of the clock output/buzzer output pin port function	472
13.4	the operation of Clock output/buzzer output controls circuitry	473
13.4.1	Operation of the output pins	473
13.5	Considerations for clock output/buzzer output control circuitry	473
Chapter 14	Watchdog timer	474
14.1	The function of the watchdog timer	474
14.2	Structure of the watchdog timer	474
14.3	Control registers of the watchdog timer	476
14.3.1	The Watchdog Timer's enable Register (WDTE).	476
14.3.2	LockUP Control Register (LOCKCTL) and its Protection Register (PRCR).	477
14.3.3	WDTCFG Configuration Register (WDTCFG0/1/2/3)	478
14.4	Operation of the watchdog timer	479
14.4.1	Operational control of the watchdog timer	479
14.4.2	Setting of the watchdog timer overflow timer	480
14.4.3	The setting during which the watchdog timer window is open	481
14.4.4	Setting of watchdog timer interval interrupts	482
14.4.5	Operation of the watchdog timer during LOCKUP	482
14.4.6	WDTCFG is not configured when the watchdog timer is running	482
Chapter 15	A/D converter	483
15.1	Functions of the A/D converter	483
15.2	Control registers of the A/D converter	485
15.2.1	Peripheral enable register 0 (PER0).	486
15.2.2	The mode register 0 (ADM0) of the A/D converter	487
15.2.3	The mode register 1 (ADM1) of the A/D converter	492
15.2.4	The mode register 2 (ADM2) of the A/D converter	493
15.2.5	The A/D converter's trigger mode register (ADTRG).	494
15.2.6	Analog input channel specified register (ADS).	495

15.2.7	12-bit A/D conversion result register (ADCR).	498
15.2.8	8-bit A/D conversion result register (ADCRH).	499
15.2.9	The conversion result compares the upper limit value of the set register (ADUL).	500
15.2.10	The conversion results compare the lower limit value set register (ADLL).	500
15.2.11	A/D Sampling Time Control Register (ADNSMP).	501
15.2.12	A/D sampling time extended register (ADSMPWAIT).	502
15.2.13	A/D Test Register (ADTES).	503
15.2.14	A/D status register (ADFLG).	504
15.2.15	A/D Charge and Discharge Control Register (ADNDIS).	505
15.2.16	Registers that control the pin function of the analog input pins	506
15.3	Input voltage and conversion result.	507
15.4	The operating mode of the A/D converter	508
15.4.1	Software-triggered mode (selection mode, continuous conversion mode).	508
15.4.2	Software-triggered mode (select mode, single-shot conversion mode).	509
15.4.3	Software trigger mode (scan mode, continuous conversion mode).	510
15.4.4	Software trigger mode (scan mode, single-shot conversion mode).	511
15.4.5	Hardware-triggered no-wait mode (select mode, continuous transition mode)	512
15.4.6	Hardware-triggered no-wait mode (select mode, single-shot transition mode)	513
15.4.7	Hardware-triggered no-wait mode (scan mode, continuous transition mode)	514
15.4.8	Hardware-triggered no-wait mode (scan mode, single-shot conversion mode)	515
15.4.9	Hardware-triggered wait mode (selection mode, continuous transition mode).	516
15.4.10	Hardware-triggered wait mode (select mode, single-shot transition mode)	517
15.4.11	Hardware-triggered wait mode (scan mode, continuous transition mode)	518
15.4.12	Hardware-triggered wait mode (scan mode, single-shot transition mode)	519
15.5	Setup flowchart for the converter	520
15.5.1	The setting of the software trigger mode.	520
15.5.2	The hardware triggers the setting of no wait mode	521
15.5.3	Hardware triggers the setting of wait mode.	522
15.5.4	Select the setting for the output voltage/internal reference voltage of the temperature sensor 523	
15.5.5	The setting of the test mode	524
Chapter 16	D/A converter.	525
16.1	The functionality of the D/A converter	525
16.2	The structure of the D/A converter	526
16.3	Registers that control the D/A converter	527
16.3.1	Peripheral enable register 1 (PER1).	527
16.3.2	The mode register (DAM) of the D/A converter.	528
16.3.3	The D/A conversion value sets register i (DACSi) (i=0, 1).	528
16.3.4	The event output target selection register n (ELSELRn), n=00~21	529

16.3.5	Registers that control the function of the analog input pin port.....	529
16.4	Operation of the D/A converter.....	530
16.4.1	Normal mode of operation.....	530
16.4.2	Operation of real-time output mode.....	531
16.4.3	The output timing of the D/A conversion values.....	532
16.5	Considerations when using D/A converters	533
Chapter 17	Comparator.....	534
17.1	The functionality of the comparator	534
17.2	The structure of the comparator	535
17.3	Control registers of the comparator.....	537
17.3.1	Peripheral enable register 1 (PER1).	538
17.3.2	Comparator Mode Setting Register (COMPMDR).	539
17.3.3	Comparator Filter Control Register (COMPFIR).	540
17.3.4	Comparator Output Control Register (COMPOCR).	542
17.3.5	The comparator has a built-in reference control register (CVRCTL).	544
17.3.6	The comparator has a built-in reference voltage selection register (CiRVM).	545
17.3.7	The input signal of comparator 0 selects the control register (CMPSEL0).	546
17.3.8	The input signal of comparator 1 selects the control register (CMPSEL1).	547
17.3.9	Hysteresis control register (CMP0HY) for comparator 0.....	548
17.3.10	Hysteresis control register (CMP1HY) for comparator 1	549
17.3.11	Registers that control the function of the analog input pin port.....	550
17.4	Run the instructions.....	551
17.4.10	The digital filter of comparator i (i=0, 1).	553
17.4.11	Comparator i interrupt (i=0, 1).	553
17.4.12	The event signal output to the linkage controller (EVENTC).	554
17.4.13	The output of comparator i (i=0, 1).	555
17.4.14	Stop and provision of the comparator clock.....	555
Chapter 18	Programmable Gain Amplifier (PGA).....	556
18.1	Programmable gain amplifier function.....	556
18.2	Structure of a programmable gain amplifier	557
18.3	Register of a programmable gain amplifier	558
18.3.1	Peripheral enable register 1 (PER1).	558
18.3.2	Programmable Gain Amplifier Control Register (PGAnCTL)	559
18.3.3	Registers that control the function of the analog input pin port.....	559
18.4	Operation of a programmable gain amplifier.....	560
18.4.1	The start-up procedure for the programmable gain amplifier.....	560
18.4.2	The stop-run step of the programmable gain amplifier	561
Chapter 19	Universal serial communication unit	562

19.1	Functions of the Universal Serial Communication Unit.....	563
19.1.1	3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21, SSPI30, SSPI31) ..	563
19.1.2	UART (UART0~UART3)	564
19.1.3	Simple I ² C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21, IIC30, IIC31).....	565
19.2	The structure of a universal serial communication unit.....	566
19.2.1	Shift Register (SCI0).....	569
19.2.2	The serial data register mn (SDRmn) is either 8 bits low or 9 bits low (SCI0).....	569
19.2.3	Shift Register (SCI1/SCI2).....	571
19.2.4	Serial data register mn(SDRmn) (SCI1/SCI2).....	571
19.3	Control registers of the universal serial communication unit	573
19.3.1	Peripheral enable register 0/2 (PER0/PER2).....	575
19.3.2	Serial clock selection register m (SPSm).....	576
19.3.3	Serial mode register mn (SMRmn).....	577
19.3.4	Serial communication runs the set register mn (SCRmn).....	579
19.3.5	Serial data register mn(SDRmn) (SCI0 i.e. m=0).....	582
19.3.6	Serial data register mn(SDRmn) (SCI1/SCI2 i.e. m=1/2).....	583
19.3.7	The serial flag clears the trigger register mn (SIRmn).	584
19.3.8	Serial status register mn (SSRmn).....	585
19.3.9	Serial channel start register m(SSm).....	587
19.3.10	Serial channel stop register m(STm).....	588
19.3.11	Serial channel enable status register m (SEm).....	589
19.3.12	Serial output enable register m (SOEm).....	590
19.3.13	Serial output register m(SOm).....	591
19.3.14	Serial output level register m(SOLm).....	592
19.3.15	Input Switch Control Register (ISC).....	594
19.3.16	Noise filter enable register 0 (NFEN0).....	595
19.3.17	Registers that control serial input/output pin port functions	596
19.4	Run stop mode	597
19.4.1	Case when the operation is stopped on a unit basis	597
19.4.2	Case of stop operation by channel.....	598
19.5	3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI20, SSPI20, SSPI20, Operation of SSPI21, SSPI30, SSPI31) communication	599
19.5.1	Master send	600
19.5.2	Master receive	608
19.5.3	Sending and receiving of the master.....	616
19.5.4	Slave sending	624
19.5.5	Slave receive	632
19.5.6	Slave sending and receiving.....	638
19.5.7	Calculate the transmit clock frequency.....	647
19.5.8	In 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20 , SSPI21, SSPI30, SSPI31)	

Processing steps when an error occurs during communication	649
19.6 The operation of the clock synchronization serial communication of the slave selection input function 650	
19.6.1 Slave sending	653
19.6.2 Slave receive	663
19.6.3 Slave sending and receiving.....	670
19.6.4 Calculate the transmit clock frequency.....	680
19.6.5 The processing step when an error occurs during clock synchronization serial communication with the slave selection input function.....	681
19.7 Operation of UART (UART0~UART3) communication	682
19.7.1 UART sends.....	683
19.7.2 UART receives.....	692
19.7.3 Calculation of baud rate.....	699
19.7.4 The processing step when an error occurs during UART (UART0~UART3) communication 703	
19.8 Operation of LIN communications	704
19.8.1 LIN sends.....	704
19.8.2 LIN receives	707
19.9 Simple I ² C (IIC00, IIC01, IIC10, IIC11, IIC20, Operation of IIC21, IIC30, IIC31) communication 712	
19.9.1 The address segment is sent	713
19.9.2 Data sending.....	718
19.9.3 Data reception	721
19.9.4 Stop conditions generation	725
19.9.5 Calculation of the transfer rate	726
19.9.6 In simple I2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21, IIC30, IIC31) Processing steps when an error occurs during communication.....	728
Chapter 20 Serial interface IICA	729
20.1 The serial interface IICA functions	729
20.1.1 Run stop mode	729
20.1.2 I2 C-bus mode (supports multi-master).....	729
20.1.3 Wake-up mode	729
20.2 Structure of the serial interface IICA	732
20.2.1 IICA shift register n (IICAn).....	733
20.2.2 Slave address register n(SVAn).	734
20.2.3 SO latches	734
20.2.4 Wake-up control circuitry	734
20.2.5 Serial clock counter	734
20.2.6 Interrupt request signal generation circuit	734
20.2.7 Serial clock control circuitry	734
20.2.8 Serial clock wait control circuit.....	734

20.2.9	Ack generation circuit, stop condition detection circuit, start condition detection circuit, Ack detection circuit.....	734
20.2.10	Data hold time correction circuit	734
20.2.11	Start conditional generation circuitry	734
20.2.12	Stop condition generation circuitry	734
20.2.13	Bus status detection circuitry.....	734
20.3	Controls registers of the serial interface IICA.....	736
20.3.1	Peripheral enable register 0/1 (PER0/1).	737
20.3.2	IICA control register n0 (IICCTLn0).....	737
20.3.3	IICA status register n (IICSn).....	742
20.3.4	IICA flag register n (IICFn).....	745
20.3.5	IICA control register n1 (IICCTLn1).....	747
20.3.6	IICA low-level width setting register n (IICWLn).....	749
20.3.7	IICA high level width setting register n (IICWHn).....	749
20.3.8	Port mode register x (PMx).....	750
20.4	The functionality of I ² C-bus mode.....	751
20.4.1	Pin structure.....	751
20.4.2	The method of transmitting the clock is set by the IICWLn register and the IICWHn register 752	
20.5	Definition and control method of the I ² C-bus.....	753
20.5.1	Start conditions	754
20.5.2	address	755
20.5.3	The designation of the transmission direction.....	755
20.5.4	Ack (ACK).	756
20.5.5	Stop Conditions	757
20.5.6	await.....	758
20.5.7	method of release from wait state	760
20.5.8	Generation timing and waiting control of interrupt requests (INTIICAn).	761
20.5.9	The detection method for address matching.....	762
20.5.10	Detection of errors	762
20.5.11	Extension code	763
20.5.12	arbitration	764
20.5.13	Wake-up function.....	766
20.5.14	Communication appointments.....	769
20.5.15	Other considerations.....	772
20.5.16	Communication operation.....	773
20.5.17	Timing of the generation of I2C interrupt requests (INTIICAn).	781
20.6	Timing diagram	802
Chapter 21	Serial interface SPI.....	817

21.1	The serial interface SPI functions.....	817
21.2	Structure of the serial interface SPI.....	817
21.3	Registers that control the serial interface SPI.....	818
21.3.1	Peripheral enable register 2 (PER2).	819
21.3.2	SPI Operating Mode Register (SPIMn).	820
21.3.3	SPI clock selection register (n).	821
21.3.4	SPI status register (SPISn).	822
21.3.5	Transmit buffer register (SDROn).	823
21.3.6	Receive buffer register (SDRIn).	823
21.3.7	The SPI pin functions the control register	824
21.4	Serial interface for operation of the SPI	825
21.4.1	The sending and receiving of the master	826
21.4.2	The reception of the master.....	829
21.4.3	Slave sending and receiving.....	832
21.4.4	Slave reception	835
Chapter 22	CAN control	838
22.1	Summary description.....	838
22.1.1	features	838
22.1.2	Feature overview	839
22.1.3	Configuration	840
22.2	CAN protocol	841
22.2.1	Frame format	841
22.2.2	Frame type.....	842
22.2.3	Data frames and remote frames.....	842
22.2.4	Error frame.....	849
22.2.5	Overload frames	850
22.3	function	851
22.3.1	Bus prioritization	851
22.3.2	Bit padding.....	851
22.3.3	Multi-master	851
22.3.4	Multicast.....	851
22.3.5	CAN sleep mode /CAN stop mode function	851
22.3.6	Error control function	852
22.3.7	Baud rate control function.....	857
22.4	The connection to the target system	861
22.5	Internal registers of the CAN controller	862
22.5.1	CAN controller configuration.....	862
22.5.2	Register access type	864
22.5.3	Register bit configuration.....	880

22.6	Bit setting/clear function	884
22.7	Control registers	885
22.7.1	Peripheral clock selection register (PER0/PER2).	885
22.7.2	CAN Global Module Control Register (CnGMCTRL).	886
22.7.3	CAN Global Module Clock Selection Register (CnGMCS).	888
22.7.4	CAN Global Automatic Block Transfer Control Register (CnGMABT).	889
22.7.5	CAN Global Automatic Block Delay Setting Register (CnGMABTD).	891
22.7.6	CAN module mask registers (CnMASKaL, CnMASKaH) (a=1, 2, 3, or4).	892
22.7.7	CAN Module Control Register (CnCTRL).	894
22.7.8	CAN Module Last Error Code Register (CnLEC).	898
22.7.9	CAN Module Information Register (CnINFO).	899
22.7.10	CAN Module Error Counter Register (CnERC).	900
22.7.11	CAN Module Interrupt Enable Register (CnIE).	901
22.7.12	CAN Module Interrupt Status Register (CnINTS).	903
22.7.13	CAN Module Bit Rate Scaling Register (CnBRP).	904
22.7.14	CAN Module Bit Rate Register (CnBTR).	905
22.7.15	CAN module last entered the pointer register (CnLIPT).	907
22.7.16	CAN module receive History List Register (CnRGPT).	908
22.7.17	CAN module last output pointer register (CnLOPT).	909
22.7.18	CAN module send History List Register (CnTGPT).	910
22.7.19	CAN Module Timestamp Register (CnTS).	911
22.7.20	CAN message data byte register (CnMDBxm) (x=0 to 7), (CnMDBzm) (z=01, 23, 45, 67).	913
22.7.21	CAN message data length register m (CnMDLCm).	915
22.7.22	CAN Message Configuration Register (CnMCONFm).	916
22.7.23	CAN message ID register m (CnMIDLm and CnMIDHm).	917
22.7.24	CAN message control register m (CnMCTRLm).	918
22.7.25	Serial communication pin select register 1 (PIOR3).	921
22.7.26	Port mode registers 0/4/5/6 (PM0/4/5/6).	921
22.8	CAN controller initialization.....	922
22.8.1	CAN module initialization.....	922
22.8.2	Initialization of the packet cache	922
22.8.3	Redefine the message cache	922
22.8.4	Transition from initialization mode to operating mode	923
22.8.5	Resets the CAN Module Error Counter cnERC	924
22.9	Message reception	925
22.9.1	Message reception	925
22.9.2	Receive data reads.....	926
22.9.3	Receive history list function.....	927
22.9.4	Blocking function.....	929

22.9.5	Multi-buffer receive block capability	930
22.9.6	Remote frame reception	931
22.10	Message sending.....	932
22.10.1	Message sending.....	932
22.10.2	Send history list function.....	934
22.10.3	Automatic Block Transfer (ABT).....	936
22.10.4	Transfer abort processing.....	937
22.10.5	Remote frame transfer.....	938
22.11	Power-saving mode	939
22.11.1	CAN sleep mode.....	939
22.11.2	CAN stop mode	941
22.11.3	Example of power saving mode	942
22.12	Interrupt function.....	943
22.13	Diagnostic functions and special operating modes.....	944
22.13.1	Receive mode only	944
22.13.2	Single-shot mode.....	945
22.13.3	Self-test mode.....	946
22.13.4	Receive/send operations in operation mode.....	947
22.14	Timestamp function	948
22.14.1	Timestamp function	948
22.15	Baud rate setting.....	950
22.15.1	Baud rate setting.....	950
22.15.2	A representative example of baud rate settings.....	954
22.16	The operation of the CAN controller.....	958
Chapter 23	LCD bus interface.....	983
23.1	Functions of the LCD bus interface	983
23.2	LCD bus interface configuration	984
23.2.1	LCD bus interface data register (LBDATA, LBATAL).....	985
23.2.2	LCD bus interface read data registers (LBDATAR, LBATARL).	986
23.3	Control registers for the LCD bus interface.....	987
23.3.1	Peripheral enable register 1 (PER1).	987
23.3.2	LCD Bus Interface Mode Register (LBCTL).....	988
23.3.3	LCB bus interface periodic control register (LBCYC).....	989
23.3.4	The LCB bus interface waits for control registers (LBWST).	989
23.3.5	Pin-mode control registers.....	990
23.4	Runtime order	991
23.4.1	Timing relationships.....	991
23.4.2	LCD bus interface status	992
23.4.3	Write the LCD bus.....	993

23.4.4	Read from the LCD bus	995
23.4.5	Write-read-write timing on the LCD bus	997
23.5	Note for LCD bus interfaces	998
23.5.1	Write to lbDATA/LBDATAL registers.....	998
23.6	Example of LCD bus interface transmission	999
23.6.1	Example of transmission with an external LCD driver	999
23.6.2	The flow of LCD bus transmission.....	1001
Chapter 24	Enhanced DMA	1009
24.1	Features of the DMA	1009
24.2	Structure of the DMA	1011
24.3	Control registers of the DMA	1012
24.3.1	DMA control data areas and DMA vector table areas allocation	1013
24.3.2	Control data allocation	1014
24.3.3	Vector table.....	1016
24.3.4	Peripheral enable register 1 (PER1).	1018
24.3.5	DMA control register j (DMACRj) (j=0~39).	1018
24.3.6	DMA block size register j (DMBLSj) (j=0~39).	1020
24.3.7	DMA transmit count register j (DMACTj) (j=0~39).	1021
24.3.8	DMA number of transfers reloads register j (DMRLDj) (j=0~39).	1022
24.3.9	DMA source address register j (DMSARj) (j=0~39).	1023
24.3.10	DMA destination address register j (DMDARj) (j=0~39).	1023
24.3.11	DMA boot enable register i (DMAENi) (i=0~4).	1024
24.3.12	DMAENi position register (DMSETi).	1026
24.3.13	DMAENi reset register (DMCLRi).....	1026
24.3.14	DMA Base Address Register (DMABAR).....	1027
24.4	Operation of the DMA	1028
24.4.1	Startup Source	1028
24.4.2	Normal mode	1029
24.4.3	Repeat pattern	1032
24.4.4	Chain transfer	1036
24.5	Considerations when using DMA	1038
24.5.1	The DMA controls the settings of the data and vector tables	1038
24.5.2	The DMA controls the allocation of data areas and DMA vector table areas	1038
24.5.3	The number of execution clocks for the DMA	1039
24.5.4	Response time of the DMA.....	1040
24.5.5	The startup source for the DMA	1040
24.5.6	Running in standby mode.....	1041
Chapter 25	Linkage Controller (EVENTC).	1042
25.1	Features of EVENTC	1042

25.2	Structure of EVENTC	1042
25.3	Control registers	1043
25.3.1	Output target selection register n(ELSELRn) (n=00~22).....	1044
25.4	Operation of EVENTC	1047
Chapter 26	Interrupt function	1049
26.1	The types of interrupt function	1049
26.2	Interrupt sources and structures.....	1049
26.3	Registers that control interrupt function	1057
26.3.1	Interrupt request flag register (IF00~IF31).	1057
26.3.2	Interrupt Mask Flag Register (MK00~MK31).....	1058
26.3.3	Enable registers for the external interrupt rising edge (EGP0, EGP1)and enable registers for the external interrupt falling edge (EGN0, EGN1).	1062
26.4	Interrupt the operation of processing.....	1064
26.4.1	Acceptance of maskable interrupt requests	1064
26.4.2	Acceptance of unmaskable interrupt requests	1064
Chapter 27	Key interrupt function	1065
27.1	The function of the key interrupt.....	1065
27.2	The structure of the key interrupt	1065
27.3	Control Registers of key interrupts.....	1067
27.3.1	Key return mode register (KRM).....	1067
27.3.2	Port mode register (PMx).	1068
Chapter 28	Standby function.....	1069
28.1	Standby function	1069
28.2	Sleep mode.....	1070
28.2.1	The setting of the sleep mode	1070
28.2.2	Exit from sleep mode	1074
28.3	Deep sleep mode.....	1074
28.3.1	Settings for deep sleep mode.....	1074
28.3.2	Exit from deep sleep mode	1077
Chapter 29	Reset function.....	1079
29.1	Reset timing.....	1081
29.2	the registers of Confirmed reset source	1084
29.2.1	Reset Control Flag Register (RESF).	1084
Chapter 30	Power-on reset circuit.....	1087
30.1	Function of the power-on reset circuit	1087
30.2	Structure of the power-on reset circuit	1088
30.3	Operation of the power-on reset circuit	1088
Chapter 31	Voltage detection circuit	1092

31.1	The function of the voltage detection circuit.....	1092
31.2	Structure of the voltage detection circuit	1093
31.3	Control Registers of the voltage detection circuit.....	1094
31.3.1	Voltage Sense Register (LVIM).	1094
31.3.2	Voltage Sense Level Register (LVIS).	1095
31.4	Operation of the voltage detection circuit.....	1098
31.4.1	Used as a setting when reset mode	1098
31.4.2	Used as a setting when breaking mode	1100
31.4.3	Used as a setting when interrupt & reset mode	1102
31.5	Considerations for voltage detection circuits.....	1108
Chapter 32	Security features	1110
32.1	Overview of the security features	1110
32.2	Registers used by the security function	1111
32.3	Operation of security features	1111
32.3.1	Flash CRC operation function (high-speed CRC).....	1111
32.3.2	CRC operation function (universal CRC).	1115
32.3.3	RAM parity error detection function.....	1117
32.3.4	SFR protection function	1119
32.3.5	Frequency detection function	1120
32.3.6	A/D test function	1121
32.3.7	Digital output signal level detection function on input/output pins	1124
32.3.8	Product unique identifier registers.....	1125
Chapter 33	Temperature sensor and internal reference voltage.....	1126
33.1	Temperature sensor	1126
33.2	Registers for temperature sensors.....	1126
33.2.1	Temperature sensor calibration data register TSN25	1126
33.2.2	Temperature sensor calibration data register TSN85	1126
33.3	Instructions for using the temperature sensor.....	1127
33.3.1	How temperature sensors are used	1127
33.3.2	How to use the temperature sensor	1128
33.4	Internal reference voltage.....	1129
33.4.1	VDD calibration data register VDDCDR	1129
33.4.2	Instructions for using the internal reference voltage	1129
Chapter 34	Option bytes	1130
34.1	Option byte functionality	1130
34.1.1	User option bytes (000C0H~000C2H).	1130
34.1.2	Flash data protection option bytes (000C3H, 500004H).....	1131
34.2	The format of the user option bytes.....	1132

34.3	The format of the flash data protection option bytes.....	1138
Chapter 35	FLASH control	1139
35.1	FLASH control function description	1139
35.2	FLASH memory structure	1140
35.3	Controls registers of FLASH.....	1141
35.3.1	Flash Write Protection Register (FLPROT).....	1141
35.3.2	FLASH operation control register (FLOPMD1, FLOPMD2).....	1142
35.3.3	Flash Erase Control Register (FLERMD).....	1142
35.3.4	Flash Status Register (FLSTS).....	1143
35.3.5	Flash full-chip erase time control register (FLCERCNT).....	1143
35.3.6	Flash Page Erase Time Control Register (FLSERCNT).....	1144
35.3.7	Flash Write Time Control Register (FLPROCNT).....	1145
35.3.8	Flash Wipe And Write Protection Control Register (FLSECPR).....	1146
35.4	Flash operation method.....	1147
35.4.1	Sector erase.....	1147
35.4.2	Chip erase.....	1148
35.4.3	Word program.....	1148
35.5	Flash read.....	1148
35.6	Considerations for FLASH operations.....	1148
Appendix	Revision Record	1149

Chapter 1 CPU

1.1 overview

This section briefly introduces the features and debugging features of the ARM Cortex-M0+ core mounted on the BAT32A2x9 product, please refer to the ARM related documents for details.

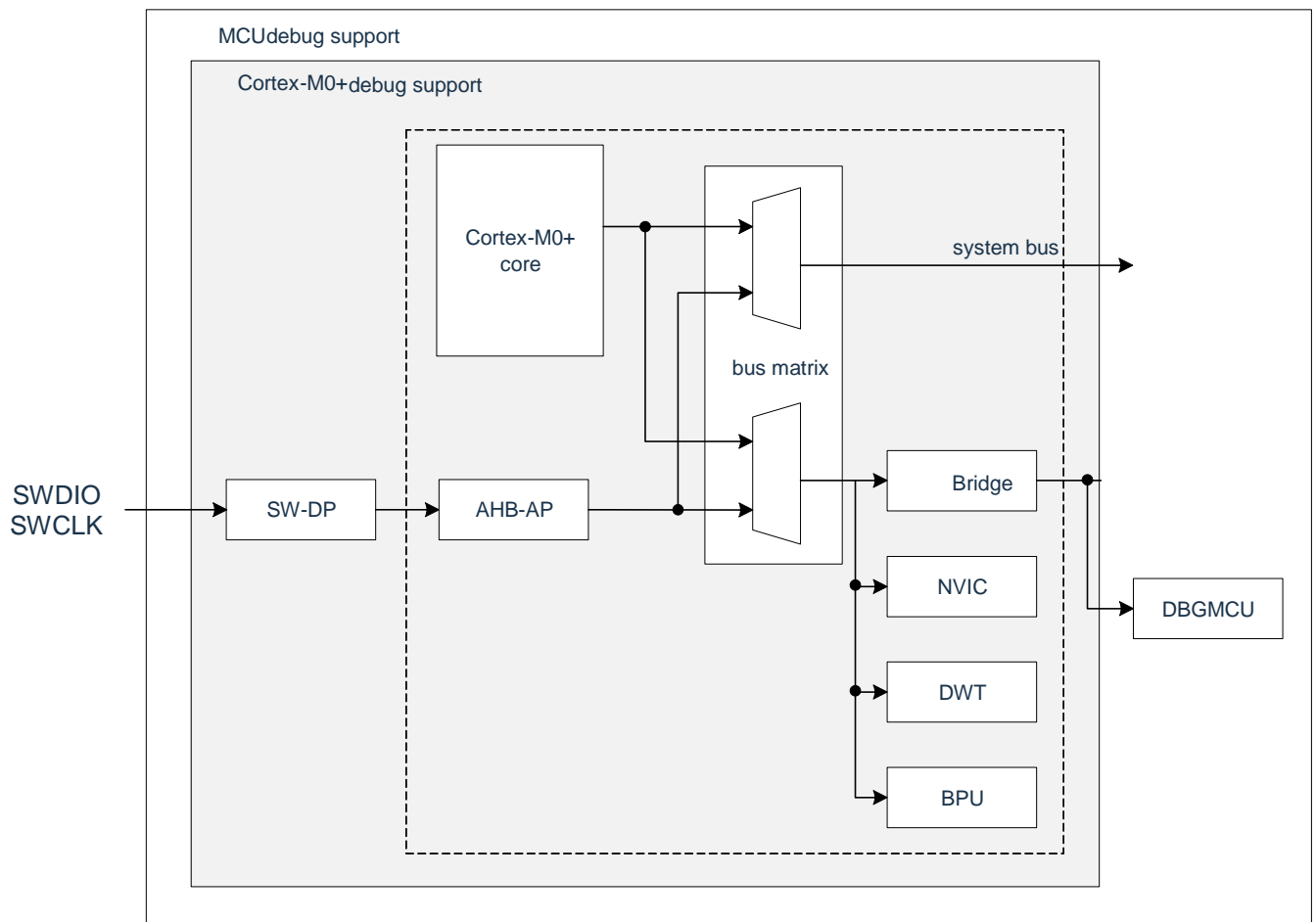
1.2 Cortex-M0+ core features

- The ARM Cortex-M0+ processor is a 32-bit RISC core with a 2-level pipeline that supports both privileged mode and user mode
- The Memory Protection Unit (MPU) supports 8 independent partitions (regions) for protection
- Single-cycle hardware multiplier
- Nested Vector Interrupt Controller (NVIC)
 - 1 Unshielded Interrupt (NMI).
 - Supports 32 maskable interrupt requests (IRQs).
 - 4 interrupt priorities
- The system timer SysTick is a 24-bit countdown timer with the option of fCLK or fIL counting clocks
- Vector Table Offset Register (VTOR).
 - The software can write the VTOR to relocate the vector table start address to a different location
 - Default value of this register is 0x0000_0000, low 8 bits write ignore, read to zero, that is, offset 256 bytes aligned

1.3 Debug features

- 2-wire SWD debug interface
- Supports pause, resume, and step-through procedures
- Access the processor's core registers and special function registers
- 4 hardware breakpoints (BPU).
- Unlimited software breakpoints (BKPT instructions).
- 2 Data Observer Points (DWT).
- Memory is accessed while the kernel is executing

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



Note: SWD does not work in Deep Sleep mode, please debug in active and sleep modes.

1.4 SWD interface pin

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 features	Pinout
SWCLK	Serial clock	P137
SWDIO	Serial data input/output	P40

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

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

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

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

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

SWDIS	SWD debug interface disable
0	SWD debug interface enabled. In the state of connecting the debugger, the P40 cannot be used as a GPIO (because the ENO and DOT of the IOBOF are controlled by the debugger at this time)
1	The SWD debug interface is disabled. The P40 can be used as a GPIO

FRZEN0	When the debugger is connected and the CPU is in the debug state (HALTED=1), the timer peripheral module action/stop Note 1
0	Peripheral action
1	Peripheral Stops

FRZEN1	When the debugger is connected and the CPU is in the debug state (HALTED=1), the communication system peripheral module action/stop Note 2
0	Peripheral action
1	Peripheral Stops

Note 1: The peripheral modules of the timer system of this product include: universal timer unit Timer4/8, timer A, timer B, timer C and timer M.

Note 2: The communication peripheral modules of this product include: communication serial communication unit, serial IICA, etc.

1.5 ARM reference documentation

The debugging features built into the Cortex-M0®+ core are part of the ARM® CoreSight design suite. For related documents, please refer to:

- Cortex-M0®+ Technical Reference Manual (TRM)
- ARM® debug interface V5
- ARM® CoreSight Design Kit Version R1p1 Technical Reference Manual
- ARM® CoreSight™ MTB-M0+ Technical Reference Manual

Chapter 2 Pin function

2.1 Port capabilities

Refer to [datasheet](#) for each product range.

2.2 Port multiplexing function

Refer to [datasheet](#) for each product range.

2.3 Registers that control port functionality

- Control the ports through the following registers.
- Port Mode Register (PMxx).
- Port register (Pxx).
- Port Set control register (PSETxx).
- Port Clearing Control Register (PCLRxx).
- Pull-up resistor selection register (PUxx).
- Port Input mode register (PIMx).
- Port Output mode register (POMx).
- Port Mode Control Register (PMCxx).
- Port Readback register (PREADxx).
- Peripheral I/O redirect register (PIORx).
- Global Digital Input Disable Register (GDIDIS).

Note: The registers and bits allocated vary by product. For the registers and bits assigned to each product, please refer to the following table. The unassigned bits must be initialized.

Table2-1 Registers assigned by product and their bits (1/3).

port		Bit name								100 Pins	80 Pins	64 Pins (-A) Note	64 Pins	48 Pin (-A) Note	48 Pins
		PMxx register	Pxx register	PSETxx register	PCLRxx register	PUxx register	PIMxx register	POMxx register	PMCxx register						
Port 0	0	PM00	P00	PSET00	PCLR00	PU00	—	POM00	PMC00 Note 1	PREAD00	o	o	o	o	o
	1	PM01	P01	PSET01	PCLR01	PU01	PIM01	—	PMC01 Note 1	PREAD01	o	o	o	o	o
	2	PM02	P02	PSET02	PCLR02	PU02	—	POM02	PMC02	PREAD02	o	o	o	o	—
	3	PM03	P03	PSET03	PCLR03	PU03	PIM03	POM03	PMC03	PREAD03	o	o	o	o	—
	4	PM04	P04	PSET04	PCLR04	PU04	PIM04	POM04	PMC04	PREAD04	o	o	o	o	—
	5	PM05	P05	PSET05	PCLR05	PU05	—	—	—	PREAD05	o	o	o	o	—
	6	PM06	P06	PSET06	PCLR06	PU06	—	—	—	PREAD06	o	o	o	o	—
Port 1	0	PM10	P10	PSET10	PCLR10	PU10	PIM10	POM10	PMC10	PREAD10	o	o	o	o	o
	1	PM11	P11	PSET11	PCLR11	PU11	—	POM11	PMC11	PREAD11	o	o	o	o	o
	2	PM12	P12	PSET12	PCLR12	PU12	—	—	—	PREAD12	o	o	o	o	o
	3	PM13	P13	PSET13	PCLR13	PU13	—	POM13	—	PREAD13	o	o	o	o	o
	4	PM14	P14	PSET14	PCLR14	PU14	PIM14	POM14	—	PREAD14	o	o	o	o	o
	5	PM15	P15	PSET15	PCLR15	PU15	PIM15	POM15	—	PREAD15	o	o	o	o	o
	6	PM16	P16	PSET16	PCLR16	PU16	PIM16	—	—	PREAD16	o	o	o	o	o
Port 2	0	PM20	P20	PSET20	PCLR20	—	—	—	PMC20	PREAD20	o	o	o	o	o
	1	PM21	P21	PSET21	PCLR21	—	—	—	PMC21	PREAD21	o	o	o	o	o
	2	PM22	P22	PSET22	PCLR22	—	—	—	PMC22	PREAD22	o	o	o	o	o
	3	PM23	P23	PSET23	PCLR23	—	—	—	PMC23	PREAD23	o	o	o	o	o
	4	PM24	P24	PSET24	PCLR24	—	—	—	PMC24	PREAD24	o	o	o	o	o
	5	PM25	P25	PSET25	PCLR25	—	—	—	PMC25	PREAD25	o	o	o	o	o
	6	PM26	P26	PSET26	PCLR26	—	—	—	PMC26	PREAD26	o	o	o	o	o
Port 3	0	PM30	P30	PSET30	PCLR30	PU30	PIM30	POM30	—	PREAD30	o	o	o	o	o
	1	PM31	P31	PSET31	PCLR31	PU31	—	—	—	PREAD31	o	o	o	o	o

Note 1 Limited to 48-pin products. (-A) indicates that it is limited to BATA2xx-A series products.

Table 2-1 Registers assigned by product and their bits (2/3).

port		Bit name									100 Pins	80 Pins	64 Pins (-A) Note	64 Pins	48 Pin (-A) Note	48 Pins
		PMxx register	Pxx register	PSETxx register	PCLRxx register	PUxx register	PIMxx register	POMxx register	PMCxx register	PREADxx register						
Port 4	0	PM40	P40	PSET40	PCLR40	PU40	—	—	—	PREAD40	o	o	o	o	o	o
	1	PM41	P41	PSET41	PCLR41	PU41	—	—	—	PREAD41	o	o	o	o	o	o
	2	PM42	P42	PSET42	PCLR42	PU42	—	—	—	PREAD42	o	o	o	o	—	—
	3	PM43	P43	PSET43	PCLR43	PU43	PIM43	POM43	—	PREAD43	o	o	o	o	—	—
	4	PM44	P44	PSET44	PCLR44	PU44	PIM44	POM44	—	PREAD44	o	o	—	—	—	—
	5	PM45	P45	PSET45	PCLR45	PU45	—	—	—	PREAD45	o	o	—	—	—	—
	6	PM46	P46	PSET46	PCLR46	PU46	—	—	—	PREAD46	o	—	—	—	—	—
	7	PM47	P47	PSET47	PCLR47	PU47	—	—	—	PREAD47	o	—	—	—	—	—
Port 5	0	PM50	P50	PSET50	PCLR50	PU50	PIM50	POM50	—	PREAD50	o	o	o	o	o	o
	1	PM51	P51	PSET51	PCLR51	PU51	—	POM51	—	PREAD51	o	o	o	o	o	o
	2	PM52	P52	PSET52	PCLR52	PU52	—	—	—	PREAD52	o	o	o	o	—	—
	3	PM53	P53	PSET53	PCLR53	PU53	—	POM53 Note 1	—	PREAD53	o	o	o	o	—	—
	4	PM54	P54	PSET54	PCLR54	PU54	—	POM54 Note 1	—	PREAD54	o	o	o	o	—	—
	5	PM55	P55	PSET55	PCLR55	PU55	PIM55	POM55	—	PREAD55	o	o	o	o	—	—
	6	PM56	P56	PSET56	PCLR56	PU56	—	—	—	PREAD56	o	—	—	—	—	—
	7	PM57	P57	PSET57	PCLR57	PU57	—	—	—	PREAD57	o	—	—	—	—	—
Port 6	0	PM60	P60	PSET60	PCLR60	—	—	—	—	PREAD60	o	o	o	o	o	o
	1	PM61	P61	PSET61	PCLR61	—	—	—	—	PREAD61	o	o	o	o	o	o
	2	PM62	P62	PSET62	PCLR62	—	—	—	—	PREAD62	o	o	o	o	o	o
	3	PM63	P63	PSET63	PCLR63	—	—	—	—	PREAD63	o	o	o	o	o	o
	4	PM64	P64	PSET64	PCLR64	PU64	—	—	—	PREAD64	o	o	—	—	—	—
	5	PM65	P65	PSET65	PCLR65	PU65	—	—	—	PREAD65	o	o	—	—	—	—
	6	PM66	P66	PSET66	PCLR66	PU66	—	—	—	PREAD66	o	o	—	—	—	—
	7	PM67	P67	PSET67	PCLR67	PU67	—	—	—	PREAD67	o	o	—	—	—	—
Port 7	0	PM70	P70	PSET70	PCLR70	PU70	—	—	—	PREAD70	o	o	o	o	o	o
	1	PM71	P71	PSET71	PCLR71	PU71	—	POM71	—	PREAD71	o	o	o	o	o	o
	2	PM72	P72	PSET72	PCLR72	PU72	—	—	—	PREAD72	o	o	o	o	o	o
	3	PM73	P73	PSET73	PCLR73	PU73	—	—	—	PREAD73	o	o	o	o	o	o
	4	PM74	P74	PSET74	PCLR74	PU74	—	POM74	—	PREAD74	o	o	o	o	o	o
	5	PM75	P75	PSET75	PCLR75	PU75	—	—	—	PREAD75	o	o	o	o	o	o
	6	PM76	P76	PSET76	PCLR76	PU76	—	—	—	PREAD76	o	o	o	o	—	—
	7	PM77	P77	PSET77	PCLR77	PU77	—	—	—	PREAD77	o	o	o	o	—	—
Port 8	0	PM80	P80	PSET80	PCLR80	PU80	PIM80	POM80	—	PREAD80	o	—	—	—	—	—
	1	PM81	P81	PSET81	PCLR81	PU81	PIM81	POM81	—	PREAD81	o	—	—	—	—	—
	2	PM82	P82	PSET82	PCLR82	PU82	—	POM82	—	PREAD82	o	—	—	—	—	—
	3	PM83	P83	PSET83	PCLR83	PU83	—	—	—	PREAD83	o	—	—	—	—	—
	4	PM84	P84	PSET84	PCLR84	PU84	—	—	—	PREAD84	o	—	—	—	—	—
	5	PM85	P85	PSET85	PCLR85	PU85	—	—	—	PREAD85	o	—	—	—	—	—
	6	PM86	P86	PSET86	PCLR86	PU86	—	—	—	PREAD86	o	—	—	—	—	—
	7	PM87	P87	PSET87	PCLR87	PU87	—	—	—	PREAD87	o	—	—	—	—	—

Note 1: Limited to BAT32A279 products only.

Table 2-1 Registers assigned by product and their bits (3/3).

port		Bit name								100 Pins	80 Pins	64 Pins (-A) Note	64 Pins	48 Pin (-A) Note	48 Pins
		PMxx register	Pxx register	PSETxx register	PCLRxx register	PUxx register	PIMxx register	POMxx register	PMCxx register	PREADxx register					
Port 10	0	PM100	P100	PSET100	PCLR100	PU100	—	—	PMC100	PREAD100	o	o	—	—	—
	1	PM101	P101	PSET101	PCLR101	PU101	—	—	PMC101	PREAD101	o	—	—	—	—
	2	PM102	P102	PSET102	PCLR102	PU102	—	—	PMC102	PREAD102	o	—	—	—	—
Port 11	0	PM110	P110	PSET110	PCLR110	PU110	—	—	—	PREAD110	o	o	—	—	—
	1	PM111	P111	PSET111	PCLR111	PU111	—	—	—	PREAD111	o	o	—	—	—
Port 12	0	PM120	P120	PSET120	PCLR120	PU120	—	—	PMC120	PREAD120	o	o	o	o	o
	1	—	P121	—	—	—	—	—	—	PREAD121	o	o	o	o	o
	2	—	P122	—	—	—	—	—	—	PREAD122	o	o	o	o	o
	3	—	P123	—	—	—	—	—	—	PREAD123	o	o	o	o	o
	4	—	P124	—	—	—	—	—	—	PREAD124	o	o	o	o	o
Port 13	0	—	P130	PSET130	PCLR130	—	—	—	—	PREAD130	o	o	o	o	o
	6	PM136	P136	PSET136	PCLR136	PU136	—	—	PMC136 Note 1	PREAD136	o	o	—	o	—
	7	PM137	P137	PSET137	PCLR137	PU137	—	—	—	PREAD137	o	o	o	o	o
Port 14	0	PM140	P140	PSET140	PCLR140	PU140	—	—	—	PREAD140	o	o	o	o	o
	1	PM141	P141	PSET141	PCLR141	PU141	—	—	—	PREAD141	o	o	o	o	—
	2	PM142	P142	PSET142	PCLR142	PU142	PIM142	POM142	—	PREAD142	o	o	—	—	—
	3	PM143	P143	PSET143	PCLR143	PU143	PIM143	POM143	—	PREAD143	o	o	—	—	—
	4	PM144	P144	PSET144	PCLR144	PU144	—	POM144	PMC144	PREAD144	o	o	—	—	—
	5	PM145	P145	PSET145	PCLR145	PU145	—	—	PMC145	PREAD145	o	—	—	—	—
	6	PM146	P146	PSET146	PCLR146	PU146	—	—	PMC146	PREAD146	o	o	o	o	o
	7	PM147	P147	PSET147	PCLR147	PU147	—	—	PMC147	PREAD147	o	o	o	o	o
Port 15	0	PM150	P150	PSET150	PCLR150	—	—	—	PMC150	PREAD150	o	o	—	—	—
	1	PM151	P151	PSET151	PCLR151	—	—	—	PMC151	PREAD151	o	o	—	—	—
	2	PM152	P152	PSET152	PCLR152	—	—	—	PMC152	PREAD152	o	o	—	—	—
	3	PM153	P153	PSET153	PCLR153	—	—	—	PMC153	PREAD153	o	o	—	—	—
	4	PM154	P154	PSET154	PCLR154	—	—	—	PMC154	PREAD154	o	—	—	—	—
	5	PM155	P155	PSET155	PCLR155	—	—	—	PMC155	PREAD155	o	—	—	—	—
	6	PM156	P156	PSET156	PCLR156	—	—	—	PMC156	PREAD156	o	—	—	—	—

Note: 1 This configuration is not available for 6-pin and 4-pin 8-pin products, only 100-pin, with 80-pin products supporting it.

(-A) indicates that it is limited to BAT32A2xx-A series products.

2.3.1 Port Mode Register (PMxx).

This is the register that sets the port input/output in bits. After generating a reset signal, the values of these registers become "FFH". When using a port pin as a pin for the multiplexing function, it must be set with reference to "2.5 Register Settings When Using the Multiplexing Function".

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

Figure2-1 format of port mode register

symbol	7	6	5	4	3	2	1	0	Offset address	After reset	R/W
PM0	1	PM06	PM05	PM04	PM03	PM02	PM01	PM00	0x320	FFH	R/W
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	0x321	FFH	R/W
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20	0x322	FFH	R/W
PM3	1	1	1	1	1	1	PM31	PM30	0x323	FFH	R/W
PM4	PM47	PM46	PM45	PM44	PM43	PM42	PM41	PM40	0x324	FFH	R/W
PM5	PM57	PM56	PM55	PM54	PM53	PM52	PM51	PM50	0x325	FFH	R/W
PM6	PM67	PM66	PM65	PM64	PM63	PM62	PM61	PM60	0x326	FFH	R/W
PM7	PM77	PM76	PM75	PM74	PM73	PM72	PM71	PM70	0x327	FFH	R/W
PM8	PM87	PM86	PM85	PM84	PM83	PM82	PM81	PM80	0x328	FFH	R/W
PM10	1	1	1	1	1	PM102	PM101	PM100	0x32A	FFH	R/W
PM11	1	1	1	1	1	1	PM111	PM110	0x32B	FFH	R/W
PM12	1	1	1	1	1	1	1	PM120	0x32C	FFH	R/W
PM13	PM137	PM136	1	1	1	1	1	0	0x32D	FEH	R/W
PM14	PM147	PM146	PM145	PM144	PM143	PM142	PM141	PM140	0x32E	FFH	R/W
PM15	1	PM156	PM155	PM154	PM153	PM152	PM151	PM150	0x32F	FFH	R/W

PMmn	Selection of input/output modes for the Pmn pin
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 unassigned bits.

2.3.2 Port register (Pxx).

This is the register that sets the value of the port output latch in bits. The pin level is read in input mode and the value of the port's output latch is read in output mode. After generating a reset signal, the value of these registers changes to "00H".

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

Figure2-2: format of the port register

symbol	7	6	5	4	3	2	1	0	address	After reset	R/W
P0	0	P06	P05	P04	P03	P02	P01	P00	0x300	00H (output latch)	R/W
P1	P17	P16	P15	P14	P13	P12	P11	P10	0x301	00H (output latch)	R/W
P2	P27	P26	P25	P24	P23	P22	P21	P20	0x302	00H (output latch)	R/W
P3	0	0	0	0	0	0	P31	P30	0x303	00H (output latch)	R/W
P4	P47	P46	P45	P44	P43	P42	P41	P40	0x304	00H (output latch)	R/W
P5	P57	P56	P55	P54	P53	P52	P51	P50	0x305	00H (output latch)	R/W
P6	P67	P66	P65	P64	P63	P62	P61	P60	0x306	00H (output latch)	R/W
P7	P77	P76	P75	P74	P73	P72	P71	P70	0x307	00H (output latch)	R/W
P8	P87	P86	P85	P84	P83	P82	P81	P80	0x308	00H (output latch)	R/W
P10	0	0	0	0	0	P102	P101	P100	0x30A	00H (output latch)	R/W
P11	0	0	0	0	0	0	P111	P110	0x30B	00H (output latch)	R/W
P12	0	0	0	P124	P123	P122	P121	P120	0x30C	Indefinite value	R/W ^{Note1}
P13	P137	P136	0	0	0	0	0	P130	0x30D	00H (output latch)	R/W
P14	P147	P146	P145	P144	P143	P142	P141	P140	0x30E	00H (output latch)	R/W
P15	0	P156	P155	P154	P153	P152	P151	P150	0x30F	00H (output latch)	R/W

Pmn	Input/output of the Pmn pin	
	Control of output data (output mode)	Reading of input data (input mode)
0	Output '0'.	Enter a low level.
1	Output "1".	Enter high.

Note: 1. P121~P124 are read-only bits.

2. The initial value must be set for the unassigned bits.

2.3.3 Port Set Control Register (PSETxx)

This is the register that sets the port output latch in bits. After generating a reset signal, the values of these registers become "0000H".

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

Figure 2-3: format of port position control register

symbol	7	6	5	4	3	2	1	0	address	After reset	R/W
PSET0	0	PSET06	PSET05	PSET04	PSET03	PSET02	PSET01	PSET00	0x080	00H	In
PSET1	PSET17	PSET16	PSET15	PSET14	PSET13	PSET12	PSET11	PSET10	0x081	00H	In
PSET2	PSET27	PSET26	PSET25	PSET24	PSET23	PSET22	PSET21	PSET20	0x082	00H	In
PSET3	0	0	0	0	0	0	PSET31	PSET30	0x083	00H	In
PSET4	PSET47	PSET46	PSET45	PSET44	PSET43	PSET42	PSET41	PSET40	0x084	00H	In
PSET5	PSET57	PSET56	PSET55	PSET54	PSET53	PSET52	PSET51	PSET50	0x085	00H	In
PSET6	PSET67	PSET66	PSET65	PSET64	PSET63	PSET62	PSET61	PSET60	0x086	00H	In
PSET7	PSET77	PSET76	PSET75	PSET74	PSET73	PSET72	PSET71	PSET70	0x087	00H	In
PSET8	PSET87	PSET86	PSET85	PSET84	PSET83	PSET82	PSET81	PSET80	0x088	00H	In
PSET10	0	0	0	0	0	PSET102	PSET101	PSET100	0x08A	00H	In
PSET11	0	0	0	0	0	0	PSET111	PSET110	0x08B	00H	In
PSET12	0	0	0	0	0	0	0	PSET120	0x08C	00H	In
PSET13	PSET137	PSET136	0	0	0	0	0	PSET130	0x08D	00H	In
PSET14	PSET147	PSET146	PSET145	PSET144	PSET143	PSET142	PSET141	PSET140	0x08E	00H	In
PSET15	0	PSET156	PSET155	PSET154	PSET153	PSET152	PSET151	PSET150	0x08F	00H	In

PSETmn	Set Control of the Pmn port
0	No action
1	The corresponding Pmn is set to 1

Note: The unassigned bits must be initialized.

2.3.4 Port Clear Control Register (PCLRxx)

This is the register that sets the port output latch in bits. After generating a reset signal, the values of these registers become "0000H".

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

Figure 2-4 format of port clearing control register

symbol	7	6	5	4	3	2	1	0	address	After reset	R/W
PCLR0	0	PCLR06	PCLR05	PCLR04	PCLR03	PCLR02	PCLR01	PCLR00	0x090	00H	In
PCLR1	PCLR17	PCLR16	PCLR15	PCLR14	PCLR13	PCLR12	PCLR11	PCLR10	0x091	00H	In
PCLR2	PCLR27	PCLR26	PCLR25	PCLR24	PCLR23	PCLR22	PCLR21	PCLR20	0x092	00H	In
PCLR3	0	0	0	0	0	0	PCLR31	PCLR30	0x093	00H	In
PCLR4	PCLR47	PCLR46	PCLR45	PCLR44	PCLR43	PCLR42	PCLR41	PCLR40	0x094	00H	In
PCLR5	PCLR57	PCLR56	PCLR55	PCLR54	PCLR53	PCLR52	PCLR51	PCLR50	0x095	00H	In
PCLR6	PCLR67	PCLR66	PCLR65	PCLR64	PCLR63	PCLR62	PCLR61	PCLR60	0x096	00H	In
PCLR7	PCLR77	PCLR76	PCLR75	PCLR74	PCLR73	PCLR72	PCLR71	PCLR70	0x097	00H	In
PCLR8	PCLR87	PCLR86	PCLR85	PCLR84	PCLR83	PCLR82	PCLR81	PCLR80	0x098	00H	In
PCLR10	0	0	0	0	0	PCLR102	PCLR101	PCLR100	0x09A	00H	In
PCLR11	0	0	0	0	0	0	PCLR111	PCLR110	0x09B	00H	In
PCLR12	0	0	0	0	0	0	0	PCLR120	0x09C	00H	In
PCLR13	PCLR137	PCLR136	0	0	0	0	0	PCLR130	0x09D	00H	In
PCLR14	PCLR147	PCLR146	PCLR145	PCLR144	PCLR143	PCLR142	PCLR141	PCLR140	0x09E	00H	In
PCLR15	0	PCLR156	PCLR155	PCLR154	PCLR153	PCLR152	PCLR151	PCLR150	0x09F	00H	In
PCLRmn	Clear control of the Pmn port										
0	No action										
1	The corresponding Pmn is cleared to zero										

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

2.3.5 Pull-up resistor selection register (PUxx).

Select register for internal pull-up resistors. The internal pull-up resistor can only be specified for pins that use an internal pull-up resistor through the pull-up resistor selection register and the POMmn bit is "0" and set to the input mode (PMmn=1), and the internal pull-up resistor is used in bits. For bits set to output mode, regardless of the setting of the pull-up resistor selection register, the internal pull-up resistor is not connected. The same is true when the output pin is used as a multiplexing function or when it is set to an analog function.

After generating a reset signal, the value of these registers changes to "00H" (only PU4 is "01H" and PU13 is "80H").

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

Figure2-5 Format of pull-up resistor selection register

symbol	7	6	5	4	3	2	1	0	Offset address	After reset	R/W
PU0	0	PU06	PU05	PU04	PU03	PU02	PU01	PU00	0x030	00H	R/W
PU1	PU17	PU16	PU15	PU14	PU13	PU12	PU11	PU10	0x031	00H	R/W
PU3	0	0	0	0	0	0	PU31	PU30	0x033	00H	R/W
PU4	0	0	PU45	PU44	PU43	PU42	PU41	PU40	0x034	01H	R/W
PU5	PU57	PU56	PU55	PU54	PU53	PU52	PU51	PU50	0x035	00H	R/W
PU6	PU67	PU66	PU65	PU64	0	0	0	0	0x036	00H	R/W
PU7	PU77	PU76	PU75	PU74	PU73	PU72	PU71	PU70	0x037	00H	R/W
PU8	PU87	PU86	PU85	PU84	PU83	PU82	PU81	PU80	0x038	00H	R/W
PU10	0	0	0	0	0	PU102	PU101	PU100	0x03A	00H	R/W
PU11	0	0	0	0	0	0	PU111	PU110	0x03B	00H	R/W
PU12	0	0	0	0	0	0	0	PU120	0x03C	00H	R/W
PU13	PU137	PU136	0	0	0	0	0	0	0x03D	80H	R/W
PU14	PU147	PU146	PU145	PU144	PU143	PU142	PU141	PU140	0x03E	00H	R/W

Fist	Selection of internal pull-up resistors for the Clearing
0	No internal pull-up resistor is connected.
1	Connect an internal pull-up resistor.

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

2.3.6 Port input mode register (PIMxx).

This is the register that sets the input buffer in bits. When communicating serially with external devices with different voltage level, the TTL input buffer can be selected.

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

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

Figure2-6 format of port input mode register

symbol	7	6	5	4	3	2	1	0	Offset address	After reset	R/W
PIM0	0	0	0	PIM04	PIM03	0	PIM01	0	0x040	00H	R/W
PIM1	PIM17	PIM16	PIM15	PIM14	0	0	0	PIM10	0x041	00H	R/W
PIM3	0	0	0	0	0	0	0	PIM30	0x043	00H	R/W
PIM4	0	0	0	PIM44	PIM43	0	0	0	0x044	00H	R/W
PIM5	0	0	PIM55	0	0	0	0	PIM50	0x045	00H	R/W
PIM8	0	0	0	0	0	0	PIM81	PIM80	0x048	00H	R/W
PIM14	0	0	0	0	PIM143	PIM142	0	0	0x04E	00H	R/W

PIMmn	Selection of input buffers for the Pmn pin
0	Schmidt input buffer
1	TTL input buffer

Note That the unassigned bits must be set at an initial value.

2.3.7 Port output mode register (POMxx).

This is the register that sets the output mode in bits. When communicating serially with external devices with different voltage level and simple I2C communication with external devices with the same voltage level, the SDAxx pin can be selected for an N-channel open-drain output mode.

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

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

Note that for bits that set the N-channel open-drain output mode (POMmn=1), no internal pull-up resistor is connected.

Figure2-7 format of port output mode register

symbol	7	6	5	4	3	2	1	0	Offset address	After reset	R/W
POM0	0	0	0	POM04	POM03	POM02	0	POM00	0x050	00H	R/W
POM1	POM17	0	POM15	POM14	POM13	0	POM11	POM10	0x051	00H	R/W
POM3	0	0	0	0	0	0	0	POM30	0x053	00H	R/W
POM4	0	0	0	POM44	POM43	0	0	0	0x054	00H	R/W
POM5	0	0	POM55	PAT54	PAT53	0	POM51	POM50	0x055	00H	R/W
POM7	0	0	0	POM74	0	0	POM71	0	0x057	00H	R/W
POM8	0	0	0	0	0	POM82	POM81	POM80	0x058	00H	R/W
POM14	0	0	0	POM144	POM143	POM142	0	0	0x05E	00H	R/W
	POMmn	Selection of output mode for the Pmn pin									
	0	The usual output mode									
	1	N-channel open-drain output mode									

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

2.3.8 Port Mode Control Register (PMCxx).

The PMC registers configure digital inputs/outputs or analog inputs setting in bits.

After generating a reset signal, the values of these registers become "FFH".

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

Figure2-8 format of the port mode control register

symbol	7	6	5	4	3	2	1	0	Offset address	After reset	R/W
PMC0	1	1	1	PMC04	PMC03	PMC02	1	1	0x060	FFH	R/W
PMC1	1	1	1	1	1	1	PMC11	PMC10	0x061	FFH	R/W
PMC2	PMC27	PMC26	PMC25	PMC24	PMC23	PMC22	PMC21	PMC20	0x062	FFH	R/W
PMC10	1	1	1	1	1	PMC102	PMC101	PMC100	0x06A	FFH	R/W
PMC12	1	1	1	1	1	1	1	PMC120	0x06C	FFH	R/W
PMC13	1	PMC136 Note 1	1	1	1	1	1	0	0x06D	FEH	R/W
PMC14	PMC147	PMC146	PMC145	PMC144	1	1	1	1	0x06E	FFH	R/W
PMC15	1	PMC156	PMC155	PMC154	PMC153	PMC152	PMC151	PMC150	0x06F	FFH	R/W

PMCmn	Choice of digital input/output or analog input on the Pmn pin
0	Digital inputs/outputs (multiplexing functions other than analog inputs).
1	Analog input

Note 1. This configuration is not available on 64-pin products.

2. The initial value must be set for the unassigned bits.

2.3.9 Port read-back register (PREADxx).

This is a read-only register that can be read to get the port level when the port is an ordinary digital GPIO.

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

Figure 2-9 Format of port read-back register

symbol	7	6	5	4	3	2	1	0	address	After reset	R/W
PREAD0	0	PREAD06	PREAD05	PREAD04	PREAD03	PREAD02	PREAD01	PREAD00	0x0A0	xxH	R
PREAD1	PREAD17	PREAD16	PREAD15	PREAD14	PREAD13	PREAD12	PREAD11	PREAD10	0x0A1	xxH	R
PREAD2	PREAD27	PREAD26	PREAD25	PREAD24	PREAD23	PREAD22	PREAD21	PREAD20	0x0A2	xxH	R
PREAD3	0	0	0	0	0	0	PREAD31	PREAD30	0x0A3	xxH	R
PREAD4	PREADC47	PREADC46	PREADC45	PREADC44	PREAD43	PREAD42	PREAD41	PREAD40	0x0A4	xxH	R
PREAD5	PREAD57	PREAD56	PREAD55	PREAD54	PREAD53	PREAD52	PREAD51	PREAD50	0x0A5	xxH	R
PREAD6	PREAD67	PREAD66	PREAD65	PREAD64	PREAD63	PREAD62	PREAD61	PREAD60	0x0A6	xxH	R
PREAD7	PREAD77	PREAD76	PREAD75	PREAD74	PREAD73	PREAD72	PREAD71	PREAD70	0x0A7	xxH	R
PREAD8	PREAD87	PREAD86	PREAD85	PREAD84	PREAD83	PREAD82	PREAD81	PREAD80	0x0A8	xxH	R
PREAD10	0	0	0	0	0	PREAD102	PREAD101	PREAD100	0x0AA	xxH	R
PREAD11	0	0	0	0	0	0	PREAD111	PREAD110	0x0AB	xxH	R
PREAD12	0	0	0	PREAD124	PREAD123	PREAD122	PREAD121	PREAD120	0x0AC	xxH	R
PREAD13	PREAD137	PREAD136	0	0	0	0	0	PREAD130	0x0AD	xxH	R
PREAD14	PREAD147	PREAD146	PREAD145	PREAD144	PREAD143	PREAD142	PREAD141	PREAD140	0x0AE	xxH	R
PREAD15	0	PREAD156	PREAD155	PREAD154	PREAD153	PREAD152	PREAD151	PREAD150	0x0AF	xxH	R

*The readout value of this register after reset depends on the status of each port.

PREADmn	Pmn port status readback
	Output mode/input mode
0	The port is low.
1	The port is high.

2.3.10 Peripheral I/O redirect register 0 (PIOR0).

This is register 0 that is set to allow or disable peripheral I/O redirection. The peripheral I/O redirection feature switches ports that are assigned a multiplexing function.

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

Figure2-10 Format of peripheral I/O redirect register 0 (PIOR0).

Address: 0x40040877 After reset: 00HR/W

SYMBOL	7	6	5	4	3	2	1	0
PIOR0	PIOR07	PIOR06	PIOR05	PIOR04	PIOR03	PIOR02	PIOR01	PIOR00

PIOR0	Features	64pin/80pin		48pin	
		Set Values		Set Values	
		0	1	0	1
bit7 (PIOR07)	INTP8	P74/P42	P00	P74	P00
	INTP10	P76/P05	P01	must be set to 0 (initial value)	
	INTP11	P77/P06	P20		
bit6 (PIOR06)	RxD2	must be set to 0 (initial value)		P14	P14
	TxD2			P13	P10
	SCL20			P15	-
	SDA20			P14	-
	SDI20			P14	-
	SDO20			P13	-
	SCLK20			P15	-
bit5 (PIOR05)	RXD1			P01	P73
	TXD1			P00	P72
bit4 (PIOR04)	CLKBUZ1	P141	P55	must be set to 0 (initial value)	
	INTP5	P16	P12		
bit3 (PIOR03)	CLKBUZ0	P140	P31	P140	P31
bit2 (PIOR02)	SCLA0	P60	P14	P60	P14
	SDAA0	P61	P15	P61	P15
bit1 (PIOR01)	INTP10	P76	P05	P01	P01
	INTP11	P77	P06	P20	P20
	RXD2	P14	P76	must be set to 0 (initial value)	
	TXD2	P13	P77		
	SCL20	P15	-		
	SDA20	P14	-		
	SDI20	P14	-		
	SDO20	P13	-		
	SCLK20	P15	-		
	TXD0	P51	P17	P51	P17
	RXD0	P50	P16	P50	P16
	SCL00	P30	-	P30	-
	SDA00	P50	-	P50	-
	SDO00	P51	P17	P51	-
	SDI00	P50	P16	P50	-
	SCLK00	P30	P55	P30	-

bit0 (PIOR00)	INTP1	P50	P52	must be set to 0 (initial value)
	INTP2	P51	P53	
	INTP3	P30	P54	
	INTP4	P31	P55	
	INTP8	P74	P42	
	INTP9	P75	P43	

2.3.11 Peripheral I/O redirect register 1 (PIOR1)

This is set to allow or disable peripheral I/O redirection function register 1. The peripheral I/O redirection feature switches ports that are assigned a multiplexing function.

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

Figure2-11 Format of peripheral I/O redirect register 1 (PIOR1).

Address: 0x40040879H After reset: 00HR/W

SYMBOL	7	6	5	4	3	2	1	0
PIOR1	0	0	0	0	PANDOR13	PIOR12	PIOR11	PIOR10

PIOR13	PIOR12	Timer A's TAO pin selection
0	0	and P30 multiplexing
0	1	and P50 multiplexed
1	0	and P00 multiplexed
1	1	Disable settings

PIOR11	PIOR10	Timer A's TAO pin selection
0	0	and P01 multiplexed
0	1	and P31 multiplexed
1	0	and P41 multiplexing
1	1	and P06 multiplexing

2.3.12 Peripheral I/O redirect register 2(PIOR2)

This is set to allow or disable peripheral I/O redirection function register 2. The peripheral I/O redirection feature switches ports that are assigned a multiplexing function.

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

Figure2-12 Format of peripheral I/O redirect register 2 (PIOR2)

Address: 0x40040875 After reset: 00HR/W

SYMBOL	7	6	5	4	3	2	1	0
WORST2	WORST27	WORST26	WORST25	WORST24	WORST23	WORST22	WORST21	WORST20

PIOR2	Features	Set Values		condition
		0	1	
bit7(PIOR27)	TMIOC0	P16	Disable setting	
bit6(PIOR26)	TMIOD0	P15	P17	PIOR36, PIOR37 must both be configured as 0
bit5(PIOR25)	TMIOD1	P11	P51	PIOR36, PIOR37 must both be configured as 0
bit4(PIOR24)	TMIOC1	P13	P50	PIOR36, PIOR37 must both be configured as 0
bit3(PIOR23)	TMIOB1	P10	P30	PIOR36, PIOR37 must both be configured as 0
bit2(PIOR22)	TMIOA1	P12	P16	PIOR36, PIOR37 must both be configured as 0
bit1(PIOR21)	VCOUT1	P31	P70	PIOR32=1'b1
bit0(PIOR20)	VCOUT0	P120	P71	PIOR31=1'b1

Note: 1, PIOR36, PIOR37 must be configured to 0, PIOR22 ~ PIOR26 configuration is valid

2. When PIOR31 is 0, the VCOUT0 multiplexing port selected by PIOR20 outputs a low level.

3. When PIOR32 is 0, the VCOUT1 multiplexing port selected by PIOR21 outputs a low level.

2.3.13 Peripheral I/O redirect register 3(PIOR3)

This is set to allow or disable peripheral I/O redirection function register 3. The peripheral I/O redirection feature switches ports that are assigned a multiplexing function.

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

Figure2-13 Format of peripheral I/O redirect register 3 (PIOR3)

Address: 0x4004087C after reset: 00HR/W

SYMBOL	7	6	5	4	3	2	1	0
WORST3	PIOR37	PIOR36	PIOR35	PIOR34	WORST33	WORST32	WORST31	0

PIOR3	Features	Set Values	
		0	1
bit5(PIOR35)	TXD0	PIOR34 Control	P12
	RXD0	PIOR34 Control	P11
bit4(PIOR34)	TXD0	PIOR01 Control	P40
	RXD0	PIOR01 Control	P137
bit3(PIOR33)	CRXD0	P03	P50
	CTXD0	P02	P51
bit2(PIOR32)	VCOUT1	Output L	PIOR21 Control
bit1(PIOR31)	VCOUT0	Output L	PIOR20 Control

PIOR37	PIOR36	TMIOA0	TMIOB0	TMIOC0	TMIOD0	TMIOA1	TMIOB1	TMIOC1	TMIOD1
0	0	P17	P14	P16	PIOR26 control	PIOR22 control	PIOR23 control	PIOR24 control	PIOR25 control
0	1	P17	P12	P16	P15	P11	P10	P14	P13
1	0	P17	P15	P16	P14	P13	P12	P11	P10
1	1	Prohibit settings							

2.3.14 Peripheral I/O redirect register 4 (PIOR4)

This is set to allow or disable peripheral I/O redirection function register 4. The peripheral I/O redirection feature switches ports that are assigned a multiplexing function. PIOR4 is only used for port redirection switching for 100pin products.

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

Figure2-14 Format of peripheral I/O redirect register 4 (PIOR4).

Address: 0x40040874 After reset: 00HR/W

SYMBOL	7	6	5	4	3	2	1	0
PIOR4	PIOR47	PIOR46	PIOR45	0	0	0	PIOR41	0

PIOR4	Features	100pin	
		Set Values	
		0	1
bit7 (PIOR47)	TI14/TO14	P100	P42
	TI15/TO15	P110	P46
	TI16/TO16	P111	P102
	TI17/TO17	P05	P145
bit6 (PIOR46)	SCLK31/SCL31	P43	P54
	SDI31	P44	P53
	SDO31	P45	P52
	SDA31	P44	P53
bit5 (PIOR45)	INTP1	PIOR0[0]control	P56
	INTP2	PIOR0[0]control	P47
	INTP3	PIOR0[0]control	P57
	INTP4	PIOR0[0]control	P146
	INTP6	P140	P84
	INTP7	P141	P85
	INTP8	PIOR0[7]	P86
	INTP9	PIOR0[0]control	P87
	SCLK10/SCL10	P04	P80
	SDI10/RxD1	PIOR0[5]	P81
	SDO10/TxD1	PIOR0[5]	P82
	SDA10	P03	P81
bit1 (PIOR41)	INTP10	PIOR0[7]	P110
	INTP11	PIOR0[7]	P111

Note: Bit bit4, 3, 2, 0 must be set to the initial value of 0.

2.3.15 Global Digital Input Disable Register (GDIDIS)

When powering off the EV_{DD} , this register prevents the input buffer from leakage current for the input port that is powered from the EV_{DD} . When all input/output ports powered by EV_{DD} are not used, EV_{DD} can be cut off by setting the GDIDIS register to "1" power supply to reduce power consumption.

By placing the GDIDIS0 position "1", the input of all input buffers with EV_{DD} as the power supply is disabled, preventing the leakage current when the power supply of the EV_{DD} is cut off.

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

Figure2-15 Global Digital Input Disable Register (GDIDIS).

Address: 0x400408After 7D			reset: 00H		R/W			
SYMBOL	7	6	5	4	3	2	1	0
GDIDIS	0	0	0	0	0	0	0	NEWIS0

GDIDIS0	Setting of the input buffer of the EV _{DD} power supply
0	Input to the input buffer is allowed (default).
1	Disable input to the input buffer. Prevents leakage current flowing through the input buffer.

To power off an EV_{DD} , you must follow these steps to set it up:

- (1) Set to disable the input buffer input (GDIDIS0=1).
- (2) Cut off the power supply to the EV_{DD} .

To power up the EV_{DD} , you must follow these steps to set it up:

- (1) Turn on the power supply of the EV_{DD} .
- (2) Set to allow the input of the input buffer (GDIDIS0=0).

Note 1 For input ports powered by EV_{DD} , voltages that exceed EV_{DD} cannot be entered.

2. If the input of the disable input buffer is set (GDIDIS0=1), the read value of the port register (Pxx) of the port powered by EV_{DD} is "1". When the port output mode register (POMxx) is "1" (N-channel open-drain output (EV_{DD} withstand voltage)), the port register (Pxx) has a read value of "0".

Note 1 Even if the input of the input buffer is disabled (GDIDIS0=1), the peripheral function of the unused port function with EV_{DD} as the power supply can be used.

2.4 Unused pin handling

The handling of each unused pin is shown in the following table.

Table2-2 Handling of each unused pin

Pin name	Input/Output	The recommended connection method when not in use
P00~P06	Input/Output	Input: Separately connect the EV_{DD} or EV_{SS} via resistors. Output: Set to open circuit.
P10~P17		
P20~P27		Input: Separately connect V_{DD} or V_{SS} via resistors. Output: Set to open circuit.
P30, P31		Input: Separately connect the EV_{DD} or EV_{SS} via resistors. Output: Set to open circuit.
P40		Input: EV_{DD} is connected separately via resistors or opened. Output: Set to open circuit.
P41~P47		Input: Separately connect the EV_{DD} or EV_{SS} via resistors. Output: Set to open circuit.
P50~P57		
P60~P63		Input: Separately connect the EV_{DD} or EV_{SS} via resistors. Output: Set the output latch of the port to "0" and set it to open, or set the output latch of the port to "1" and connect the EV_{DD} or via resistors alone EV_{SS} .
P63~P67		Input: Separately connect the EV_{DD} or EV_{SS} via resistors. Output: Set to open circuit.
P70~P77		
P80~P87		
P100~P102		
P110~P111		
P120		
P121~P124	input	Separately connect V_{DD} or V_{SS} via resistors.
P130	output	Set to open circuit.
P136	Input/Output	Input: Separately connect the EV_{DD} or EV_{SS} via resistors. (64,48-pin products). Input: Separately connect V_{DD} or V_{SS} via resistors. (80-pin product, BATG179xx-A series). Output: Set to open circuit.
P137		Input: V_{DD} is connected separately via resistors or opened. Output: Set to open circuit.
P140~P147		Input: Separately connect the EV_{DD} or EV_{SS} via resistors. Output: Set to open circuit.
P150~P156	Input/Output	Input: Separately connect V_{DD} or V_{SS} via resistors. Output: Set to open circuit.
RESETB	input	Connect V_{DD} directly or via resistors.

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

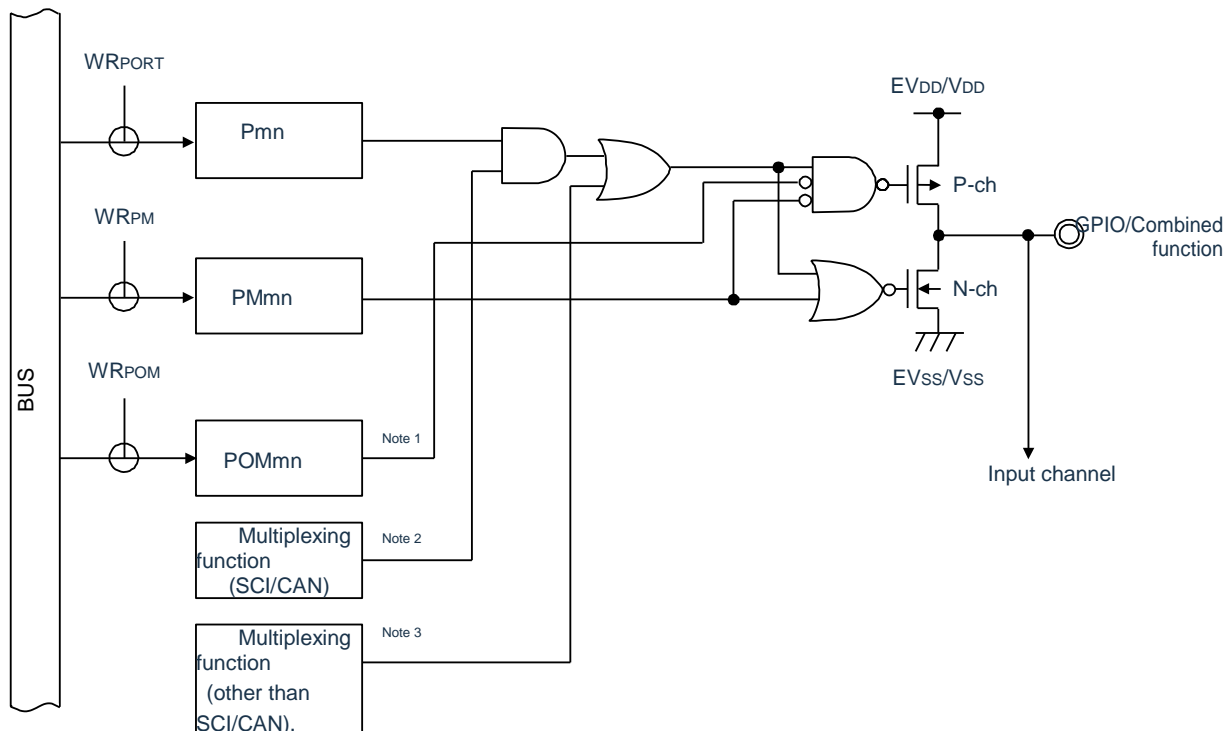
2.5 Register settings when using the multiplexing function

2.5.1 Basic principle when using the multiplexing feature

First, pins that are multiplexed with analog functions must be set via the Port Mode Control Register (PMCxx) whether they will be used as analog functions or as digital inputs/outputs.

The basic structure of the pin output circuit used as a digital input/output is shown in Figure 2-10. The outputs of the SCI and CAN functions with the output latch output multiplexing of the port are input to the AND gate, and the output of the AND gate is input to the OR gate. OR The other inputs of the gate are connected to multiplexed outputs of non-SCI/CAN functions (timer, RTC, clock/buzzer output, IICA, etc.). When such pins are used as port functions or multiplexing functions, the unused multiplexing functions cannot affect the output of the functions to be used. The basic idea of the setting at this time is shown in Table2-3 basic .

Figure2-16 The basic structure of the output circuit of the pin



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

2. When there is no multiplexing function, this signal is high level (1).

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

Table2-3 basic principle of the configuration

Use the pinout function	Output settings for the reuse function that are not used		
	Port capabilities	The output function of SCI/CAN	Output functions other than SCI/CAN
Port output function	—	High level output (1).	Low level output (0).
The output function of SCI/CAN	High (1)	—	Low level output (0).
Output functions other than SCI/CAN	Low (0)	High level output (1).	Low level output (0) Note

Note : Because it is possible for one pin to multiple output functions other than SCI/CAN, the output of the unused multiplexing function needs to be set to Low (0). For specific setting methods, please refer to "2. 5.2 Example of register settings for the port function and the multiplexing function used".

2.5.2 Examples of register settings using the port function and the multiplexing function

Examples of register settings (100-pin articles) used for the port function and multiplexing function are shown in the following table.

Table2-4 function register settings example

Pin name	Features used		PIORx	POMxx	PMCxx	PMxx	Pxx	The output of the multiplexing function	
	Feature name	Input/Output						The output function of SCI/CAN	Outside of SCI/CAN
P00	P00	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	—	(MAN)=0
		N-channel open-drain output		1	—	0	0/1		
	TI00	input	—	x	—	1	x	—	—
	TBCLK0	input	—	x	—	1	x	—	—
	(TAO)	output	PIOR13, PIOR12=10B	0	—	0	0	—	—
	(INTP8)	input	PIOR07=1, PIOR45=0	x	—	1	x	—	—
P01	P01	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	TAIO=0, TO00=0
	TO00	output	—	—	—	0	0	—	TAIO=0
	TBCLK1	input	—	—	—	1	x	—	—
	TAIO	input	PIOR13, PIOR12=00B	—	—	1	x	—	—
		output	—	—	—	0	0	—	TO00=0
	(INTP10)	input	PIOR07=1, PIOR41=0	—	—	1	x	—	—
P02	P02	input	—	x	0	1	x	—	—
		output	—	0	0	0	0/1	TxD1/SDO10=1 CTxD0=1	—
		N-channel open-drain output		1	0	0	0/1		
	ANI11	Analog input	—	x	1	1	x	—	—
	TxD1	output	PIOR45=0	0/1	0	0	1	CTxD0=1	—
	SDO10	output	PIOR45=0	0/1	0	0	1	CTxD0=1	—
	VCIN10	Analog input	—	x	1	1	x	—	—
P03	P03	input	—	x	0	1	x	—	—
		output	—	0	0	0	0/1	SDA10=1	—
		N-channel open-drain output		1	0	0	0/1		
	ANI10	Analog input	—	x	1	1	x	—	—
	SDI10	input	PIOR45=0	x	0	1	x	—	—
	RxD1	input	PIOR45=0	x	0	1	x	—	—
	SDA10	Input/Output	PIOR45=0	1	0	0	1	—	—
	VCIN11	Analog input	—	x	1	1	x	—	—
	CRxD0	input	PIOR33=0	x	0	1	x	—	—
P04	P04	input	—	x	0	1	x	—	—
		output	—	0	0	0	0/1	SCLK10/SCL10=1	—
		N-channel open-drain output		1	0	0	0/1		
	ANI13	Analog input	—	x	1	1	x	—	—
	SCLK10	input	PIOR45=0	x	0	1	x	—	—
		output		0/1	0	0	1	—	—
	SCL10	output	PIOR45=0	0/1	0	0	1	—	—
P05	P05	input	—	—	—	1	x	—	—

		output	—	—	—	0	0/1	DBRDB=1	TO17=0
	(INTP10)	input	PIOR01=1, PIOR41=0 PIOR07=0	—	—	1	x	—	—
	TI17	input	PIOR47=0	—	—	1	x	—	—
	TO17	output	PIOR47=0	—	—	0	0	DBRDB=1	—
	DBRDB	output	—	—	—	0	1	—	TO17=0

Pin name	Features used		PIORx	POMxx	PMCxx	PMxx	Pxx	The output of the multiplexing function	
	Feature name	Input/Output						The output function of SCI/CAN	Outside of SCI/CAN
P06	P06	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	DBWRB=1	(TAIO)=0
	(TAIO)	input	PIOR11, PIOR10=11B	—	—	1	x	—	—
		output		—	—	0	0	DBWRB=1	—
	(INTP11)	input	PIOR01=1, PIOR07=0 PIOR45=0	—	—	1	x	—	—
	DBWRB	output	—	—	—	0	1	—	(TAIO)=0
P10	P10	input	—	x	0	1	x	—	—
		output	—	0	0	0	0/1	SCLK11/SCL11=1	TMIOB1=0, (TPIRD1)=0
		N-channel open-drain output		1	0	0	0/1		
	ANI9	Analog input	—	x	1	1	x	—	—
	SCLK11	input	—	x	0	1	x	—	—
		output	—	0/1	0	0	1	—	TMIOB1=0 (TPIRD1)=0
	SCL11	output	—	0/1	0	0	1	—	TMIOB1=0 (TPIRD1)=0
	TMIOB1	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	0	1	x	—	—
		output		0	0	0	0	SCLK11/SCL11=1	—
	(TWED1)	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	0	1	x	—	—
		output		0	0	0	0	SCLK11/SCL11=1	—
P11	P11	input	—	x	0	1	x	—	—
		output	—	0	0	0	0/1	SDA11=1	TMIOD1=0 (TMIOA1)=0 (TMIOC1)=0
		N-channel open-drain output		1	0	0	0/1		
	ANI8	Analog input	—	x	1	1	x	—	—
	SDI11	input	—	x	0	1	x	—	—
	SDA11	Input/Output	—	1	0	0	1	—	TMIOD1=0 (TMIOA1)=0 (TMIOC1)=0
	(RxD0)	input	PIOR35=1	x	0	1	1	—	—
	TMIOD1	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	0	1	x	—	—
		output		0	0	0	0	SDA11=1	—
	(TMIOA1)	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	0	1	x	—	—
		output		0	0	0	0	SDA11=1	—
P12	P12	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	SDO11=0	TMIOA1=0 (TMIOB0)=0 (TMIOB1)=0
	SDO11	output	—	—	—	0	1	—	TMIOA1=0 (TMIOB0)=0 (TMIOB1)=0
	(TxD0)	output	PIOR35=1	—	—	0	1	—	TMIOA1=0 (TMIOB0)=0 (TMIOB1)=0
	TMIOA1	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	—	—	1	x	—	—
		output		—	—	0	0	SDO11=0	—
	(TMIOB0)	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	—	—	1	x	—	—
		output		—	—	0	0	SDO11=0	—
	(TMIOB1)	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	—	—	1	x	—	—
		output		—	—	0	0	SDO11=0	—

	(INTP5)	input	PIOR04=1	—	—	1	x	—	—
--	---------	-------	----------	---	---	---	---	---	---

Pin name	Features used		PIORx	POMxx	PMCxx	PMxx	Pxx	The output of the multiplexing function	
	Feature name	Input/Output						The output function of SCI/CAN	Outside of SCI/CAN
P13	P13	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	TxD2/SDO20=1	TMIOC1=0 (TMIOD1)=0 (TMIOA1)=0
		N-channel open-drain output		1	—	0	0/1		
	TxD2	output	PIOR01=0	0/1	—	0	1	—	TMIOC1=0 (TMIOD1)=0 (TMIOA1)=0
	SDO20	output	PIOR01=0	0/1	—	0	1	—	TMIOC1=0 (TMIOD1)=0 (TMIOA1)=0
	TMIOC1	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	—	1	x	—	—
		output		0	—	0	0	TxD2/SDO20=1	—
	(TWED1)	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	—	1	x	—	—
		output		0	—	0	0	TxD2/SDO20=1	—
	(TMIOA1)	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	—	1	x	—	—
		output		0	—	0	0	TxD2/SDO20=1	—
P14	P14	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	SDA20=1	TMIOB0=0, (TMIOC1)=0 (TMIOD0)=0 (SCLA0)=0
		N-channel open-drain output		1	—	0	0/1		
	RxD2	input	PIOR01=0	x	—	1	x	—	—
	SDI20	input	PIOR01=0	x	—	1	x	—	—
	SDA20	Input/Output	PIOR01=0	1	—	0	1	—	TMIOB0=0, (TMIOC1)=0 (TMIOD0)=0 (SCLA0)=0
	TMIOB0	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	—	1	x	—	—
		output		1	—	0	0	SDA20=1	(SCLA0)=0
	(TMIOC1)	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	—	1	x	—	—
		output		1	—	0	0	SDA20=1	(SCLA0)=0
	(TMIOD0)	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	—	1	x	—	—
		output		1	—	0	0	SDA20=1	(SCLA0)=0
	(SCLA0)	Input/Output	PIOR02=1	1	—	0	0	SDA20=1	TMIOB0=0, (TMIOC1)=0 (TMIOD0)=0
P15	P15	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	SCLK20/SCL20=1	TMIOD0=0, (TMIOB0)=0 (SDAA0)=0
		N-channel drain Open output		1	—	0	0/1		
	SCLK20	input	PIOR01=0	x	—	1	x	—	—
		output		0/1	—	0	1	—	TMIOD0=0, (TMIOB0)=0 (SDAA0)=0
	SCL20	output	PIOR01=0	0/1	—	0	1	—	—
	TMIOD0	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	—	1	x	—	—
		output		0	—	0	0	SCLK20/SCL20=1	(SDAA0)=0
	(TMIOB0)	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	—	1	x	—	—
		output		0	—	0	0	SCLK20/SCL20=1	(SDAA0)=0
	(SDAA0)	Input/Output	PIOR02=1	1	—	0	0	SCLK20/SCL20=1	TMIOD0=0, (TMIOB0)=0

Pin name	Features used		PIORx	POMxx	PMCxx	PMxx	Pxx	The output of the multiplexing function	
	Feature name	Input/Output						The output function of SCI/CAN	Outside of SCI/CAN
P16	P16	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	TO01=0, TMIOC0=0, (TMIOA1)=0
	TI01	input	—	—	—	1	x	—	—
	TO01	output	—	—	—	0	0	—	TMIOC0=0, (TMIOA1)=0
	INTP5	input	PIOR04=0	—	—	1	x	—	—
	TMIOC0	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	—	—	1	x	—	—
		output		—	—	0	0	—	TO01=0, (TMIOA1)=0
	(SDI00)	input	PIOR01=1, PIOR34=0 PIOR35=0	—	—	1	x	—	—
	(RxD0)	input	PIOR01=1, PIOR34=0 PIOR35=0	—	—	1	x	—	—
	(TMIOA1)	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	—	—	1	x	—	—
		output		—	—	0	0	—	TO01=0, TMIOC0=0
P17	P17	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	(TxD0)/(SDO00)=1	TO02=0, TMIOA0=0, (TMIOD0)=0
		N-channel drain Open output		1	—	0	0/1		
	TI02	input	—	x	—	1	x	—	—
	TO02	output	—	0	—	0	0	(TxD0)/(SDO00)=1	TMIOA0=0, (TMIOD0)=0
	TMIOA0	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	—	1	x	—	—
		output		0	—	0	0	(TxD0)/(SDO00)=1	TO02=0, (TMIOD0)=0
	TMCLK	input	—	x	—	1	x	—	—
	(SDO00)	output	PIOR01=1, PIOR34=0 PIOR35=0	0/1	—	0	1	—	TO02=0, TMIOA0=0, (TMIOD0)=0
	(TxD0)	output	PIOR01=1, PIOR34=0 PIOR35=0	0/1	—	0	1	—	TO02=0, TMIOA0=0, (TMIOD0)=0
	(TMIOD0)	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	—	1	x	—	—
		output		0	—	0	0	(TxD0)/(SDO00)=1	TO02=0, TMIOA0=0

Pins name	Features used		PIORx	PMCxx	PMxx	Pxx
	Feature name	Input/Output				
P20	P20	input	—	0	1	x
		output	—	0	0	0/1
	ANI0	Analog input	—	1	1	x
	AV _{REFP}	Reference input	—	1	1	x
	VCIN12	Analog input	—	1	1	x
	(INTP11)	input	PIOR07=1, PIOR41=0	0	1	1
P21	P21	input	—	0	1	x
		output	—	0	0	0/1
	ANI1	Analog input	—	1	1	x
	AV _{REFM}	Reference input	—	1	1	x
	VCIN13	Analog input	—	1	1	x
P22	P22	input	—	0	1	x
		output	—	0	0	0/1
	ANI2	Analog input	—	1	1	x
	ANO0	Analog output	—	1	1	x
	VCIN0	Analog input	—	1	1	x
	PGA0IN	Analog input	—	1	1	x
P23	P23	input	—	0	1	x
		output	—	0	0	0/1
	ANI3	Analog input	—	1	1	x
	ANO1	Analog output	—	1	1	x
	PGA0GND	Analog input	—	1	1	x
P24	P24	input	—	0	1	x
		output	—	0	0	0/1
	ANI4	Analog input	—	1	1	x
	PGA1IN	Analog input	—	1	1	x
P25	P25	input	—	0	1	x
		output	—	0	0	0/1
	ANI5	Analog input	—	1	1	x
	PGA1GND	Analog input	—	1	1	x
P26	P26	input	—	0	1	x
		output	—	0	0	0/1
	ANI6	Analog input	—	1	1	x
P27	P27	input	—	0	1	x
		output	—	0	0	0/1
	ANI7	Analog input	—	1	1	x

Pin name	Features used		PIORx	POMxx	PMCxx	PMxx	Pxx	The output of the multiplexing function	
	Feature name	Input/Output						The output function of SCI/CAN	Outside of SCI/CAN
P30	P30	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	SCLK00/SCL00=1	RTC1HZ=0, TAO=0, (TMIOB1)=0
		N-channel open-drain output		1	—	0	0/1		
	INTP3	input	PIOR00=0, PIOR45=0	x	—	1	x	—	—
	RTC1HZ	output	—	0	—	0	0	SCLK00/SCL00=1	PERSON=0, (TMIOB1)=0
	SCLK00	input	PIOR01=0	x	—	1	x	—	—
		output		0/1	—	0	0	—	RTC1HZ=0, TAO=0, (TMIOB1)=0
	SCL00	output	PIOR01=0	0/1	—	0	0	—	RTC1HZ=0, TAO=0, (TMIOB1)=0
	MAN	output	PIOR13, PIOR12=00B	0	—	0	0	SCLK00/SCL00=1	RTC1HZ=0, (TMIOB1)=0
	(TMIOB1)	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	—	1	x	—	—
		output		0	—	0	0	SCLK00/SCL00=1	RTC1HZ=0, TAO=0
P31	P31	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	TO03=0, VCOUT1=0, (CLKBUZ0)=0, (TAIO)=0
	TI03	input	—	—	—	1	x	—	—
	TO03	output	—	—	—	0	0	—	VCOUT1=0, (CLKBUZ0)=0, (TAIO)=0
	INTP4	input	PIOR00=0, PIOR45=0	—	—	1	x	—	—
	(TAIO)	input	PIOR11, PIOR10=01B	—	—	1	x	—	—
		output		—	—	0	0	—	TO03=0, VCOUT1=0, (CLKBUZ0)=0
	(CLKBUZ0)	output	PIOR03=1	—	—	0	0	—	TO03=0, (TAIO)=0, VCOUT1=0
	INCOU1	output	PIOR21=0	—	—	0	0	—	TO03=0, (TAIO)=0, (CLKBUZ0)=0

Pin name	Features used		PIORx	POMxx	PMCxx	PMxx	Pxx	The output of the multiplexing function	
	Feature name	Input/Output						The output function of SCI/CAN	Outside of SCI/CAN
P40	P40	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	(TxD0)=1	—
	(TxD0)	output	WORST34=1, WORST35=0	—	—	0	1	—	—
	SWDIO	input	—	—	—	1	x	—	—
		output	—	—	—	0	x	—	—
P41	P41	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	(TAIO)=0
	(TAIO)	input	PIOR11, PIOR10=10B	—	—	1	x	—	—
		output		—	—	0	0	—	—
P42	P42	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	(TO14)=0
	(INTP8)	input	PIOR00=1, PIOR07=0 PIOR45=0	—	—	1	x	—	—
	(TI14)	input	PIOR47=1	—	—	1	x	—	—
	(TO14)	output	PIOR47=1	—	—	0	0	—	—
P43	P43	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	SCLK31/SCL31 =1	—
		N-channel open-drain output		1	—	0	0/1		—
	(INTP9)	input	PIOR00=1, PIOR45=0	x	—	1	x	—	—
	SCLK31	input	PIOR46=0	x	—	1	x	—	—
		output	PIOR46=0	0/1	—	0	1	—	—
	SCL31	output	PIOR46=0	0/1	—	0	1	—	—
P44	P44	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	SDA31=1	—
		N-channel open-drain output		1	—	0	0/1		—
	SDI31	input	PIOR46=0	x	—	1	x	—	—
	SDA31	Input/Output	PIOR46=0	1	—	0	1	—	—
P45	P45	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	SDO31-1	—
	SDO31	output	PIOR46=0	—	—	0	1	—	—
P46	P46	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	CTxD2=1	(TO15)=0
	(TI15)	input	PIOR47=1	—	—	1	x	—	—
	(TO15)	output	PIOR47=1	—	—	0	0	—	—
	CTxD2	output	—	—	—	0	1	—	(TO15)=0
P47	P47	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	—
	(INTP2)	input	PIOR45=1	—	—	1	x	—	—
	CRxD2	input	—	—	—	1	x	—	—

Pin name	Features used		PIORx	POMxx	PMCxx	PMxx	Pxx	The output of the multiplexing function	
	Feature name	Input/Output						The output function of SCI/CAN	Outside of SCI/CAN
P50	P50	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	SDA00=1	TBIO0=0, (TAO)=0, (TMIOC1)=0
		N-channel open-drain output		1	—	0	0/1		
	INTP1	input	PIOR00=0, PIOR45=0	x	—	1	x	—	—
	SDI00	input	PIOR01=0, PIOR34=0 PIOR35=0	x	—	1	x	—	—
	RxD0	input	PIOR01=0, PIOR34=0 PIOR35=0	x	—	1	x	—	—
	SDA00	Input/Output	PIOR01=0, PIOR34=0 PIOR35=0	1	—	0	1	—	TBIO0=0, (MAN)=0 , (TMIOC1)=0
	(CRxD0)	input	PIOR33=1	x	—	1	x	—	—
	TBIO0	input	—	x	—	1	x	—	—
		output	—	0	—	0	0	—	(TAO)=0, (TMIOC1)=0
	(TAO)	output	PIOR13, PIOR12=01B	0	—	0	0	—	TBIO0=0, (TMIOC1)=0
	(TMIOC1)	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	—	1	x	—	—
		output		0	—	0	0	—	TBIO0=0, (TAO)=0
P51	P51	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	TxD0/SDO00=1 (CTxD0) =1	TBIO1=0, (TMIOD1)=0
		N-channel open-drain output		1	—	0	0/1		
	INTP2	input	PIOR00=0, PIOR45=0	x	—	1	x	—	—
	SDO00	output	PIOR01=0, PIOR34=0 PIOR35=0	0/1	—	0	1	(CTxD0) =1	TBIO1=0, (TMIOD1)=0
	TxD0	output	PIOR01=0, PIOR34=0 PIOR35=0	0/1	—	0	1	(CTxD0) =1	TBIO1=0, (TMIOD1)=0
	(CTxD0)	output	PIOR33=1	0/1	—	0	1	TxD0/SDO00=1	TBIO1=0, (TMIOD1)=0
	TBIO1	input	—	x	—	1	x	—	—
		output		0	—	0	0	—	(TMIOD1)=0
	(TMIOD1)	input	Please refer to the register instructions for PIOR2 and PIOR3 for configuration	x	—	1	x	—	—
		output		0	—	0	0	—	TBIO1=0
P52	P52	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	(SDO31)-1	—
	(INTP1)	input	PIOR00=1, PIOR45=0	—	—	1	x	—	—
	(SDO31)	output	PIOR46=1	—	—	0	1	—	—
P53	P53	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	(SDA31) =1	—
		N-channel open-drain output		1	—	0	0/1		—
	(SDI31)	input	PIOR46=1	x	—	1	x	—	—
	(SDA31)	Input/Output	PIOR46=1	1	—	0	1	—	—
	(INTP2)	input	PIOR00=1, PIOR45=0	x	—	1	x	—	—
P54	P54	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	(SCLK31)/(SCL31) =1	—
		N-channel open-drain output		1	—	0	0/1		—
	SPIHS0_NSS	input	—	x	—	1	x	—	—
	(INTP3)	input	PIOR00=1, PIOR45=0	x	—	1	x	—	—
	(SCLK31)	input	PIOR46=1	x	—	1	x	—	—

		output	PIOR46=1	0/1	—	0	1	—	—
	(SCL31)	output	PIOR46=1	0/1	—	0	1	—	—

Pin name	Features used		PIORx	POMxx	PMCxx	PMxx	Pxx	The output of the multiplexing function	
	Feature name	Input/Output						The output function of SCI/CAN	Outside of SCI/CAN
P55	P55	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	(SCLK00)=1	(CLKBUZ1)=0 SPIHS0_SCK=0
		N-channel open-drain output		1	—	0	0/1		
	SPIHS0_SCK	input	—	x	—	1	x	—	—
		output	—	0	—	0	0	(SCLK00)=1	(CLKBUZ1)=0
	(INTP4)	input	PIOR00=1, PIOR45=0	x	—	1	x	—	—
	(CLKBUZ1)	output	PIOR04=1	0	—	0	0	(SCLK00)=1	SPIHS0_SCK=0
	(SCLK00)	input	PIOR01=1	x	—	1	x	—	—
		output		0/1	—	0	1	—	(CLKBUZ1)=0 SPIHS0_SCK=0
P56	P56	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	SPIHS0_SO=0
	(INTP1)	input	PIOR45=1	—	—	1	x	—	—
	SPIHS0_MI	input	—	—	—	1	x	—	—
	SPIHS0_SO	output	—	—	—	0	0	—	—
P57	P57	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	SPIHS0_MO=0
	(INTP3)	input	PIOR45=1	—	—	1	x	—	—
	SPIHS0_SI	input	—	—	—	1	x	—	—
	SPIHS0_MO	output	—	—	—	0	0	—	—

Pin name	Features used		PIORx	POMxx	PMCxx	PMxx	Pxx	The output of the multiplexing function	
	Feature name	Input/Output						The output function of SCI/CAN	Outside of SCI/CAN
P60	P60	input	—	—	—	1	x	—	—
		N-channel open-drain output (6V withstand voltage).	—	—	—	0	0/1	—	SCLA0=0
	SCLA0	Input/Output	PIOR02=0	—	—	0	0	—	—
P61	P61	input	—	—	—	1	x	—	—
		N-channel open-drain output (6V withstand voltage).	—	—	—	0	0/1	—	SDAA0=0
	SDAA0	Input/Output	PIOR02=0	—	—	0	0	—	—
P62	P62	input	—	—	—	1	x	—	—
		N-channel open-drain output (6V withstand voltage).	—	—	—	0	0/1	—	SCLA1=0
	SS00	input	—	—	—	1	x	—	—
	SCLA1	Input/Output	—	—	—	0	0	—	—
P63	P63	input	—	—	—	1	x	—	—
		N-channel open-drain output (6V withstand voltage).	—	—	—	0	0/1	—	SDAA1=0
	SDAA1	Input/Output	—	—	—	0	0	—	—
P64	P64	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	CTxD1=1	TO10=0
	TI10	input	—	—	—	1	x	—	—
	TO10	output	—	—	—	0	0	—	—
	CTxD1	output	—	—	—	0	1	—	TO10=0
P65	P65	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	TO11=0
	TI11	input	—	—	—	1	x	—	—
	TO11	output	—	—	—	0	0	—	—
	CRxD1	input	—	—	—	1	x	—	—
P66	P66	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	TO12=0
	TI12	input	—	—	—	1	x	—	—
	TO12	output	—	—	—	0	0	—	—
P67	P67	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	TO13=0
	TI13	input	—	—	—	1	x	—	—
	TO13	output	—	—	—	0	0	—	—

Pin name	Features used		PIORx	POMxx	PMCxx	PMxx	Pxx	The output of the multiplexing function	
	Feature name	Input/Output						The output function of SCI/CAN	Outside of SCI/CAN
P70	P70	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	SCLK01/SCL01=1	(VCOUT1)=0
	KR0	input	—	—	—	1	x	—	—
	SCLK21	input	—	—	—	1	x	—	—
		output	—	—	—	0	1	—	(VCOUT1)=0
	SCL21	output	—	—	—	0	1	—	(VCOUT1)=0
	(INCOUT1)	output	PIOR21=1	—	—	0	0	SCLK01/SCL01=1	—
P71	P71	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	SDA21=1	(VCOUT0)=0
		N-channel open-drain output		1	—	0	0/1		
	KR1	input	—	x	—	1	x	—	—
	SDI21	input	—	x	—	1	x	—	—
	SDA21	Input/Output	—	0/1	—	0	1	—	(VCOUT0)=0
	(INCOUT0)	output	PIOR20=1	0	—	0	0	SDA21=1	—
P72	P72	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	SDO21=1	—
	KR2	input	—	—	—	1	x	—	—
	SDO21	output	—	—	—	0	1	—	—
P73	P73	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	SDO01=1	—
	KR3	input	—	—	—	1	x	—	—
	SDO01	output	—	—	—	0	1	—	—
P74	P74	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	SDA01=1	—
		N-channel open-drain output		1	—	0	0/1		
	KR4	input	—	x	—	1	x	—	—
	INTP8	input	PIOR00=0, PIOR07=0 PIOR45=0	x	—	1	x	—	—
	SDI01	input	—	x	—	1	x	—	—
	SDA01	Input/Output	—	0/1	—	0	1	—	—
P75	P75	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	SCLK01/SCL01=1	—
	KR5	input	—	—	—	1	x	—	—
	INTP9	input	PIOR00=0, PIOR45=0	—	—	1	x	—	—
	SCLK01	input	—	—	—	1	x	—	—
		output	—	—	—	0	1	—	—
	SCL01	output	—	—	—	0	1	—	—
P76	P76	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	—
	KR6	input	—	—	—	1	x	—	—
	INTP10	input	PIOR01=0, PIOR07=0 PIOR41=0	—	—	1	x	—	—
	(RxD2/IrRxD)	input	PIOR01=1	—	—	1	x	—	—
P77	P77	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	(TxD2/IrTxD)=1	—

	KR7	input	—	—	—	1	x	—	—
	INTP11	input	PIOR01=0, PIOR07=0 PIOR41=0	—	—	1	x	—	—
	(TxD2/IrTxd)	output	PIOR01=1	—	—	0	1	—	—

Pin name	Features used		PIORx	POMxx	PMCxx	PMxx	Pxx	The output of the multiplexing function	
	Feature name	Input/Output						The output function of SCI/CAN	Outside of SCI/CAN
P80	P80	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	(SCLK10)/(SCL10)=1	DBD0=0
		N-channel open-drain output		1	—	0	0/1		
	DBD0	output	—	0	—	1	0	(SCLK10)/(SCL10)=1	—
	(SCLK10)	input	PIOR45=1	x	—	1	x	—	—
		output	PIOR45=1	0	—	0	1	—	DBD0=0
	(SCL10)	output	PIOR45=1	0	—	0	1	—	DBD0=0
P81	P81	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	(SDA10)=1	DBD1=0
		N-channel open-drain output		1	—	0	0/1		
	DBD1	output	—	0	—	1	0	(SDA10)=1	—
	(SDI10)	input	PIOR45=1	x	—	1	x	—	—
	(SDA10)	Input/Output	PIOR45=1	0/1	—	0	1	—	DBD1=0
	(RxD1)	input	PIOR45=1	x	—	1	x	—	—
P82	P82	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	(SDO10)/(TxD1)=1	DBD2=0
		N-channel open-drain output		1	—	0	0/1		
	DBD2	output	—	0	—	1	0	(SDO10)/(TxD1)=1	—
	(SDO10)	output	PIOR45=1	0/1	—	0	1	—	DBD2=0
	(TxD1)	output	PIOR45=1	0/1	—	0	1	—	DBD2=0
P83	P83	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	DBD3=0
	DBD3	output	—	—	—	1	0	—	—
P84	P84	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	DBD4=0
	DBD4	output	—	—	—	1	0	—	—
	(INTP6)	input	PIOR45=1	—	—	1	x	—	—
P85	P85	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	DBD5=0
	DBD5	output	—	—	—	1	0	—	—
	(INTP7)	input	PIOR45=1	—	—	1	x	—	—
P86	P86	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	DBD6=0
	DBD6	output	—	—	—	1	0	—	—
	(INTP8)	input	PIOR45=1	—	—	1	x	—	—
P87	P87	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	DBD7=0
	DBD7	output	—	—	—	1	0	—	—
	(INTP9)	input	PIOR45=1	—	—	1	x	—	—

Pin name	Features used		PIORx	POMxx	PMCxx	PMxx	Pxx	The output of the multiplexing function	
	Feature name	Input/Output						The output function of SCI/CAN	Outside of SCI/CAN
P100	P100	input	—	—	0	1	x	—	—
		output	—	—	0	0	0/1	—	TO14=0
	ANI16	Analog input	—	—	1	1	x	—	—
	TI14	input	PIOR47=0	—	0	1	x	—	—
	TO14	output	PIOR47=0	—	0	0	0	—	—
P101	P101	input	—	—	0	1	x	—	—
		output	—	—	0	0	0/1	—	—
	ANI24	Analog input	—	—	1	1	x	—	—
P102	P102	input	—	—	0	1	x	—	—
		output	—	—	0	0	0/1	—	TO16=0
	ANI25	Analog input	—	—	1	1	x	—	—
	TI16	input	PIOR47=1	—	0	1	x	—	—
	TO16	output	PIOR47=1	—	0	0	0	—	—
P110	P110	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	TO15=0
	TI15	input	PIOR47=0	—	—	1	x	—	—
	TO15	output	PIOR47=0	—	—	0	0	—	—
	(INTP10)	input	PIOR41=1	—	—	1	x	—	—
P111	P111	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	TO16=0
	TI16	input	PIOR47=0	—	—	1	x	—	—
	TO16	output	PIOR47=0	—	—	0	0	—	—
	(INTP11)	input	PIOR41=1	—	—	1	x	—	—
P120	P120	input	—	—	0	1	x	—	—
		output	—	—	0	0	0/1	—	VCOUT0=0
	ANI14	Analog input	—	—	1	1	x	—	—
	VCOUT0	output	PIOR20=0	—	0	0	0	—	—

Pin name	Features used		Cmc	Pxx
	Feature name	Input/Output	(EXCLK, OSCSEL, EXCLKS, OSCSELS)	
P121	P121	input	00xx/10xx/11xx	x
	X1	—	01xx	—
P122	P122	input	00xx/10xx/11xx	x
	X2	—	01xx	—
	EXCLK	input	11xx	—
P123	P123	input	xx00/xx10/xx11	x
	XT1	—	xx01	—
P124	P124	input	xx00/xx10/xx11	x
	XT2	—	xx01	—
	EXCLKS	input	xx11	—

Pin name	Features used		PIORx	POMxx	PMCxx	PMxx	Pxx	The output of the multiplexing function	
	Feature name	Input/Output						The output function of SCI/CAN	Outside of SCI/CAN
P130	P130	output	—	—	—	—	0/1	—	—
P136	P136	input	—	—	0 ^{Note 2}	1	x	—	—
		output	—	—	0 ^{Note 2}	0	0/1	—	—
	INTP0 ^{Note 1}	input	—	—	—	1	x	—	—
P137	P137	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	—
	INTP0 ^{Note 2}	input	—	—	—	1	x	—	—
	SWCLK	input	—	—	—	1	x	—	—
	(RxD0)	input	PIOR34=1, PIOR35=0	—	—	1	x	—	—

Note 1.Limited to 48-pin, 64-pin products.

2.Limited to 100 pins, 80 pins, BAT32A279xx-A series products.

Pin name	Features used		PIORx	POMxx	PMCxx	PMxx	Pxx	The output of the multiplexing function	
	Feature name	Input/Output						The output function of SCI/CAN	Outside of SCI/CAN
P140	P140	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	CLKBUZ0=0
	CLKBUZ0	output	PIOR03=0	—	—	0	0	—	—
	INTP6	input	PIOR45=0	—	—	1	x	—	—
P141	P141	input	—	—	—	1	x	—	—
		output	—	—	—	0	0/1	—	CLKBUZ1=0
	SPIHS1_NSS	input	—	—	—	1	x	—	—
	CLKBUZ1	output	PIOR04=0	—	—	0	0	—	—
	INTP7	input	PIOR45=0	—	—	1	x	—	—
P142	P142	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	SCLK30/SCL30=1	SPIHS1_SCK=0
		N-channel open-drain output		1	—	0	0/1		
	SPIHS1_SCK	input	—	x	—	1	x	—	—
		output	—	0/1	—	0	0	SCLK30/SCL30=1	—
	SCLK30	input	—	x	—	1	x	—	—
		output	—	0/1	—	0	1	—	SPIHS1_SCK=0
	SCL30	output	—	0/1	—	0	1	—	SPIHS1_SCK=0
P143	P143	input	—	x	—	1	x	—	—
		output	—	0	—	0	0/1	SDA30=1	SPIHS1_SO=0
		N-channel open-drain output		1	—	0	0/1		
	SPIHS1_MI	input	—	x	—	1	x	—	—
	SPIHS1_SO	output	—	0	—	0	0	SDA30=1	—
	SDI30	input	—	x	—	1	x	—	—
	SDA30	Input/Output	—	1	—	0	1	—	SPIHS1_SO=0
	RxD3	input	—	x	—	1	x	—	—
P144	P144	input	—	x	0	1	x	—	—
		output	—	0	0	0	0/1	(TxD3)/(SDO30)=1	SPIHS1_MO=0
		N-channel open-drain output	—	1	0	0	0/1		
	ANI26	Analog input	—	x	1	1	x	—	—
	SPIHS1_SI	input	—	x	0	1	x	—	—
	SPIHS1_MO	output	—	0	0	0	0	(TxD3)/(SDO30)=1	—
	SDO30	output	—	0/1	0	0	1	—	SPIHS1_MO=0
	TxD3	output	—	0/1	0	0	1	—	SPIHS1_MO=0
P145	P145	input	—	—	0	1	x	—	—
		output	—	—	0	0	0/1	—	(TO17)=0
	ANI27	Analog input	—	—	1	1	x	—	—
	(TI17)	input	PIOR47=1	—	0	1	x	—	—
	(TO17)	output	PIOR47=1	—	0	0	0	—	—
P146	P146	input	—	—	0	1	x	—	—
		output	—	—	0	0	0/1	—	—
	(INTP4)	input	PIOR45=1	—	0	1	x	—	—
	ANI15	Analog input	—	—	1	1	x	—	—
P147	P147	input	—	—	0	1	x	—	—

		output	—	—	0	0	0/1	—	—
	ANI12	Analog input	—	—	1	1	×	—	—
	VREF0	Analog input	—	—	1	1	×	—	—

Pin name	Features used		PIORx	PMCxx	PMxx	Pxx
	Feature name	Input/Output				
P150	P150	input	—	0	1	x
		output	—	0	0	0/1
	ANI17	Analog input	—	1	1	x
P151	P151	input	—	0	1	x
		output	—	0	0	0/1
	ANI18	Analog input	—	1	1	x
P152	P152	input	—	0	1	x
		output	—	0	0	0/1
	ANI19	Analog input	—	1	1	x
P153	P153	input	—	0	1	x
		output	—	0	0	0/1
	ANI20	Analog input	—	1	1	x
P154	P154	input	—	0	1	x
		output	—	0	0	0/1
	ANI21	Analog input	—	1	1	x
P155	P155	input	—	0	1	x
		output	—	0	0	0/1
	ANI22	Analog input	—	1	1	x
P156	P156	input	—	0	1	x
		output	—	0	0	0/1
	ANI23	Analog input	—	1	1	x

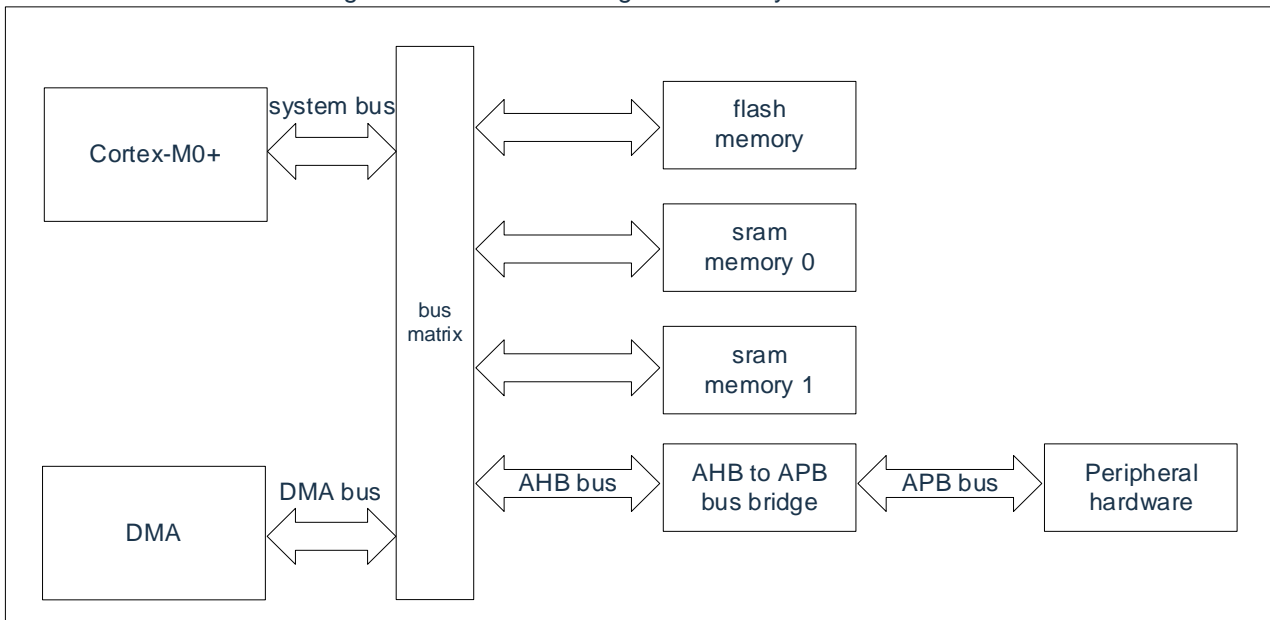
Chapter 3 System structure

3.1 overview

This product system consists of the following parts:

- 2 AHB bus Master:
 - Cortex-M0+
 - Enhanced DMA
- 4 AHB Bus Slaves:
 - FLASH memory
 - SRAM memory 0
 - SRAM memory 1
 - AHB to APB Bridge, which contains all APB interface peripherals

Figure 3-1 Schematic diagram of the system structure



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

3.2 System address partition

Figure 3-2 Schematic diagram of address area partition (BAT32A239).

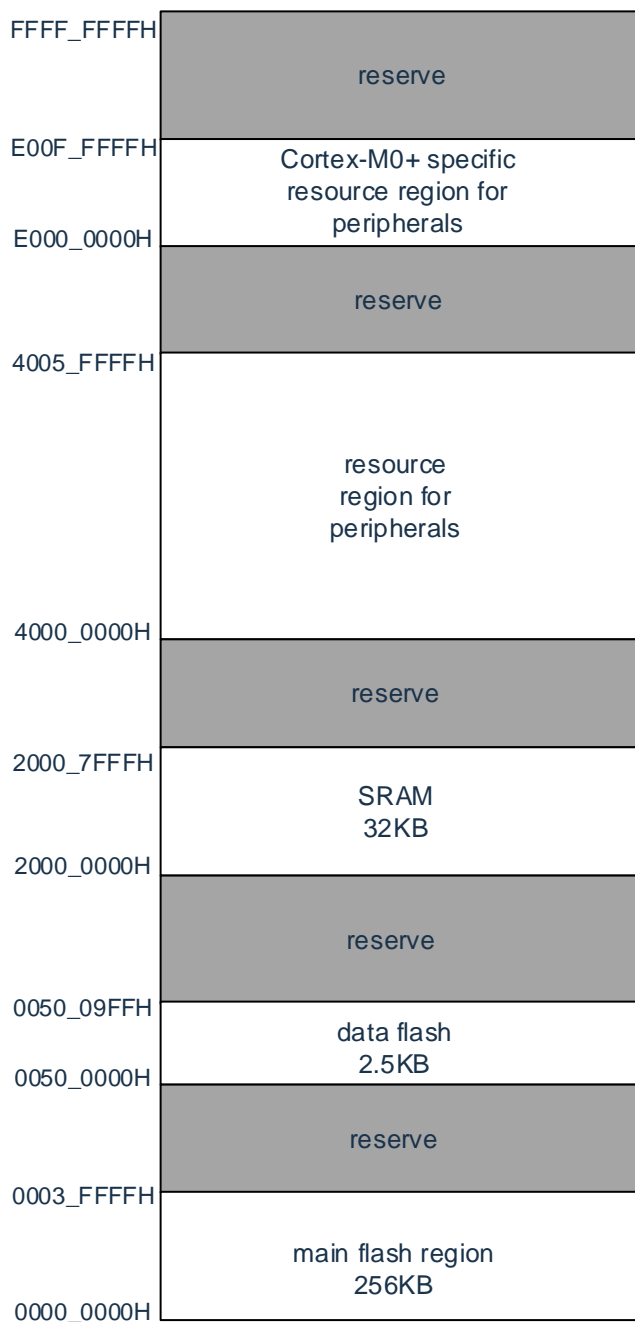
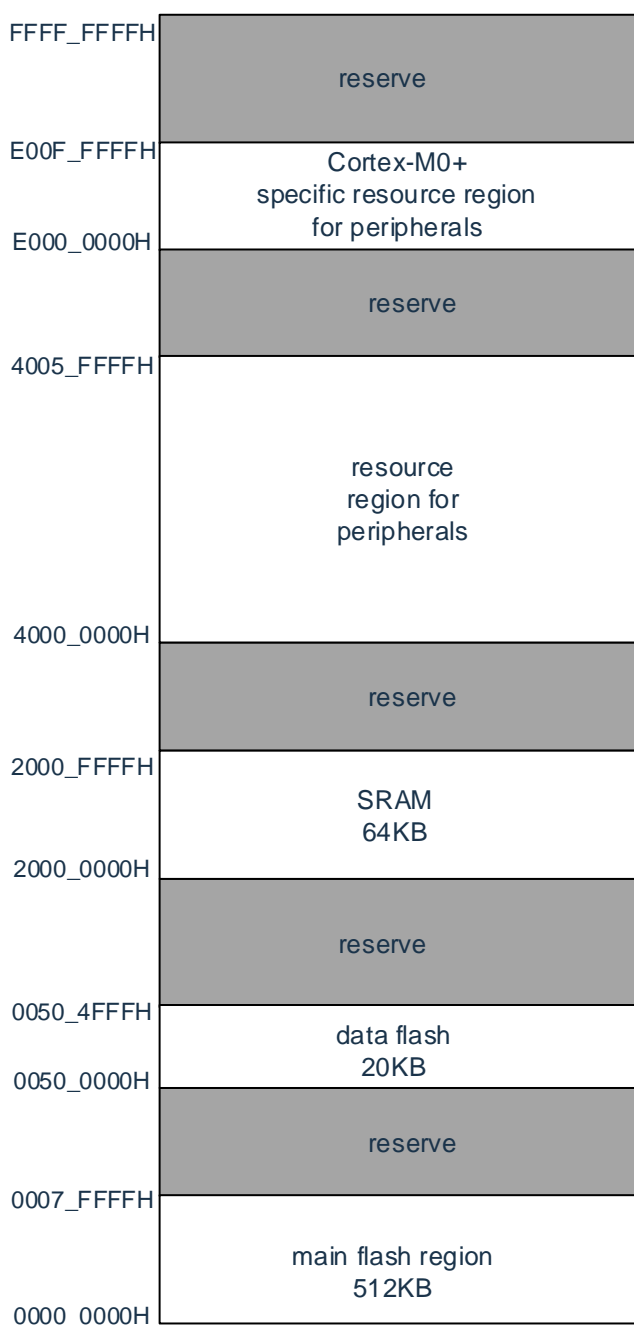


Figure 3-3 Schematic diagram of address area partition (BAT32A279).



3.3 Peripheral address assignment

Table 3-1 Register group start addresses for peripherals

The start address	peripheral	remark
0x4000_5000-0x4000_5FFF	DMA	
0x4000_6000-0x4000_6FFF	Interrupt control	
0x4001_8000-0x4001_9FFF	SRAM	
0x4001_A000-0x4001_AFFF	DIV	
0x4001_B000-0x4001_BFFF	DBGREG	
0x4002_0000-0x4002_03FF	FLASH control	
0x4002_0400-0x4002_0FFF	Clock control	
0x4002_1000-0x4002_1001	Watchdog timer	
0x4002_1002-0x4002_1800	reserve	
0x4002_1800-0x4002_1BFF	High-speed CRC	See Chapter 31: Safety Functions for details
0x4002_1C00-0x4002_1FFF	FLASH control	
0x4002_2000-0x4002_25FF	Clock control	
0x4004_0000-0x4004_0BFF	GPIO	
0x4004_0C00-0x4004_0FFF	CLKBUZ	
0x4004_1000-0x4004_17FF	Serial communication unit	
0x4004_1800-0x4004_1BFF	Serial interface IICA0	
0x4004_1C00-0x4004_1FFF	Timer4	
0x4004_2000-0x4004_23FF	Timer A	
0x4004_2400-0x4004_27FF	Timer B	
0x4004_2800-0x4004_2BFF	Timer M	
0x4004_2C00-0x4004_2FFF	Timer C	
0x4004_3000-0x4004_33FF	Universal CRC	See Chapter 31: Safety Functions for details
0x4004_3400-0x4004_37FF	Linkage controller	
0x4004_3800-0x4004_3BFF	Comparator	
0x4004_3C00-0x4004_3FFF	Timer M (PWMOP)	
0x4004_4400-0x4004_47FF	DA converter	
0x4004_4800-0x4004_4BFF	Key interrupt	
0x4004_4C00-0x4004_4FFF	Real-time clock	
0x4004_5000-0x4004_53FF	AD Converter	
0x4004_5400-0x4004_57FF	CAN controller 0	
0x4004_5800-0x4004_5BFF	CAN Controller 1	
0x4004_5C00-0x4004_5FFF	Timer 8	
0x4004_6000-0x4004_63FF	Serial interface IICA1	
0x4004_6400-0x4004_67FF	CAN Controller 2	BAT32A279 proprietary
0x4004_6800-0x4004_6BFF	Interrupt controller	
0x4004_6C00-0x4004_6FFF	Serial interface SPI0	BAT32A279 proprietary
0x4004_7000-0x4004_73FF	Serial interface SPI1	BAT32A279 proprietary
0x4004_7400-0x4004_77FF	LCD bus interface	BAT32A279 proprietary

Chapter 4 Clock generation circuit

4.1 Function of the clock generation circuit

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

(1) The master system clock

(1) X1 oscillation circuit

It can oscillate the clock from $f_X = 1$ to 20 MHz by connecting a resonator to pins X1 and X2, and can stop the oscillation via entering deep sleep mode or setting MSTOP bit (bit7 of the clock operating state control register (CSC)).

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

Can be configured to oscillate from $f_{HOCO} = 64\text{MHz}$, 48MHz, 32MHz, 24MHz, 16MHz, 12MHz, 8MHz, 6MHz, 4MHz, 3MHz, 2MHz and 1MHz (TYP.) via option byte(000C2H). The selected frequency is selected for oscillation, and the frequency of f_{IH} and f_{HOCO} is the same. After the reset is released, the CPU must start running with this high-speed internal oscillator clock. Oscillation can be stopped by entering deep sleep mode or by setting the HIOSTOP bit (bit0 of the CSC register). The frequency of the option byte setting can be changed via the frequency selection register (HOCODIV) of the high-speed internal oscillator. For frequency settings, refer to "Figure 4-14 Format of Frequency Selection Register (HOCODIV) for the high-speed internal oscillator". The oscillation frequencies that can be set by the high-speed internal oscillator are as following (the types that can be selected by the option byte and the frequency selection register (HOCODIV) of the high-speed internal oscillator).

Supply voltage	Oscillation frequency (MHz).											
	1	2	3	4	6	8	12	16	24	32	48	64
$2.7V \leq V_{DD} \leq 5.5V$	○	○	○	○	○	○	○	○	○	○	○	○
$2.4V \leq V_{DD} \leq 5.5V$	○	○	○	○	○	○	○	○	—	—	—	—
$1.8V \leq V_{DD} \leq 5.5V$	○	○	○	○	○	○	—	—	—	—	—	—

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

The high-speed system clock (X1 clock or external master clock) and the high-speed internal oscillator clock can be switched by setting the MCM0 bit (bit4 of the system clock control register (CKC)).

(2) Subsystem clock

• XT1 oscillation circuit

The clock oscillation of $f_{XT}=32.768\text{kHz}$ can be made by connecting a 32.768kHz resonator to the XT1 pin and the XT2 pin, and it can be set by XTSTOP bit (bit6 of the clock operating state control register (CSC)) to stop the oscillation.

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

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

The clock oscillator works with $f_{IL} = 15\text{kHz}$ (TYP)

A low-speed internal oscillator clock cannot be used as a CPU clock.

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

Only the following peripheral hardware can run through a low-speed internal oscillator clock:

- Watchdog timer
- Real-time clock
- 15-bit interval timer
- Timer A

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

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

Note that the low-speed internal oscillator clock (f_{IL}) can only be selected as the count clock for the real-time clock only when using the fixed-cycle interrupt function.

Note: f_X : X1 clock oscillation frequency

F_{HOCO} : Clock frequency of high-speed internal oscillators (up to 64MHz).

f_{IH} : Clock frequency of high-speed internal oscillator (64MHz max.)

f_{EX} : External master system clock frequency

f_{XT} : XT1 clock oscillation frequency

f_{EXS} : External subsystem clock frequency

f_{IL} : The clock frequency of the low-speed internal oscillator

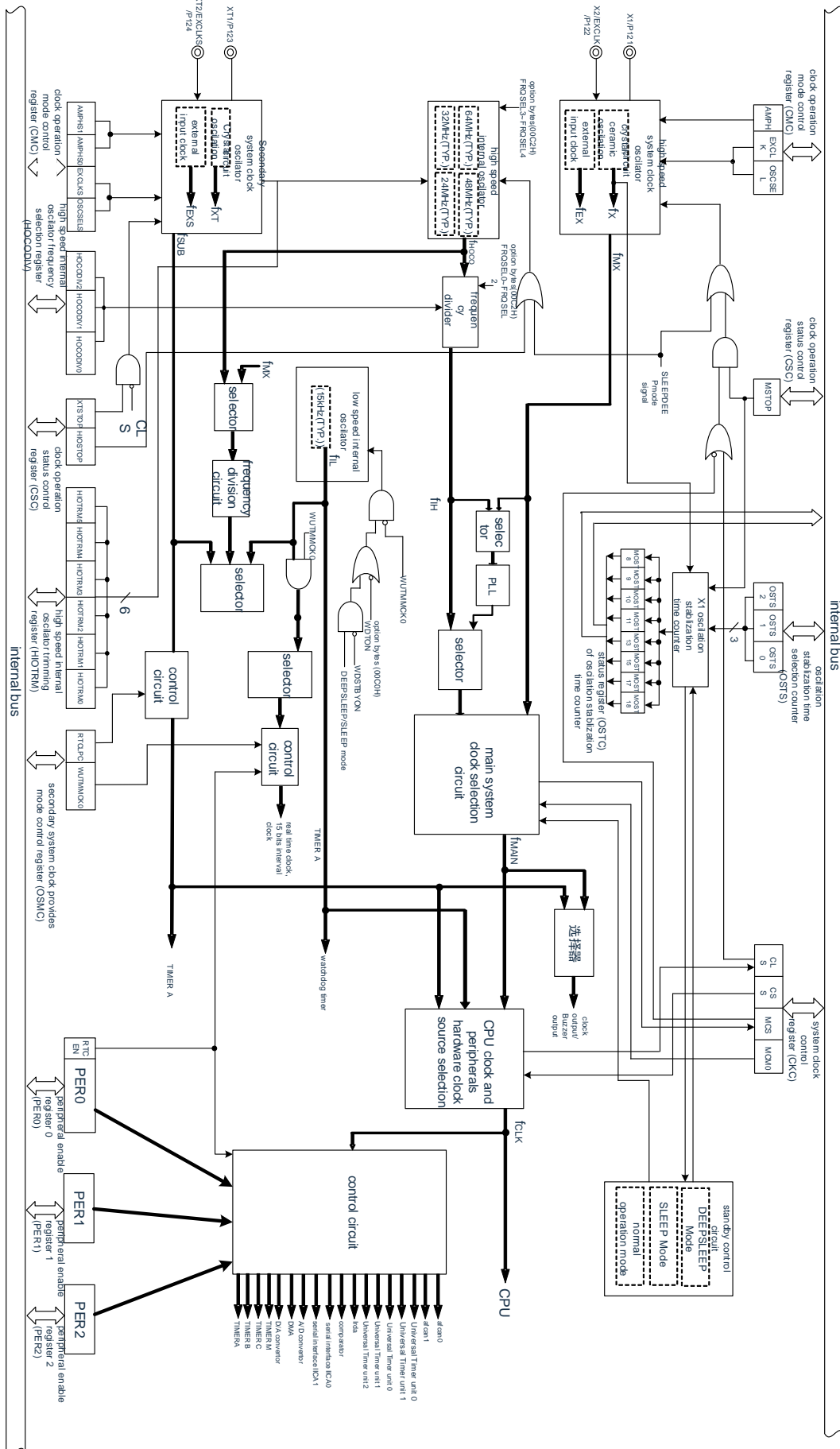
4.2 Structure of the clock generation circuit

The clock generation circuit consists of the following hardware.

Table 4-1 Structure of the clock generation circuit

project	structure
Control registers	Clock operating mode control register (CMC). System Clock Control Register (CKC). Clock Operating State Control Register (CSC). The state register (OSTC) of the oscillation settling time counter Oscillation settling time selection register (OSTS). peripheral enable register 0, 1, 2, 3(PER0, PER1, PER2, PER3) The subsystem clock provides a mode control register (OSMC). Frequency Selection Register (HOCODIV) for high-speed internal oscillators Trimming Register (HIOTRM) for high-speed internal oscillator
Oscillation circuit	X1 oscillation circuit XT1 oscillation circuit High-speed internal oscillator Low-speed internal shaker PLL

Figure 4-1 diagram of the clock generation circuit



Remark:	f_X	f_{HOCO}	: X1 clock oscillation frequency
			: Clock frequency of high-speed internal oscillator (max. 64MHz).
	f_{IH}		: Clock frequency of high-speed internal oscillator (max. 64MHz).
	f_{EX}		: External master system clock frequency
	f_{MX}		: High speed system clock frequency
	f_{MAIN}		: The master system clock frequency
	f_{XT}		: XT1 clock oscillation frequency
	f_{EXS}		: External subsystem clock frequency
	f_{SUB}		: Subsystem clock frequency
	f_{CLK}		: The clock frequency of the CPU/peripheral hardware
	f_{ll}		: Clock frequency of low-speed internal oscillator

4.3 Control Registers of the clock generation circuit

The clock generation circuit is controlled by the following registers.

- Clock Run Mode Control Register (CMC).
- System Clock Control Register (CKC).
- Clock Operating Status Control Register (CSC).
- Oscillation settling time counter status register (OSTC).
- Oscillation Settling Time Selection Register (OSTS).
- peripheral enable register 0, 1, 2, 3(PER0, PER1, PER2, PER3)
- Subsystem Clock Mode Control Registers (OSMC).
- Frequency Selection Register (HOCODIV) for high-speed internal oscillators
- Trimming Register (HIOTRM) for high-speed internal oscillator

Note that the registers and bits assigned vary by product. You must set an initial value for the unassigned bits.

4.3.1 Clock operating mode control register (CMC).

This is the register that sets the operating mode of the X1/P121, X2/EXCLK/P122, XT1/P123, XT2/EXCLKS/P124 pins and selects the gain of the oscillation circuit.

After the reset is released, the CMC register can only be written once via the 8-bit memory operation instruction. This register can be read via the 8-bit memory operation instruction.

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

Figure 4-2 Format of Clock operating mode control register (CMC)

Address: 40020400H after reset: 00HR/W

SYMBOL	7	6	5	4	3	2	1	0
Cmc	EXCLK	OSCSEL	EXCLKS Note	OSCSELS Note	0	AMPHS1 Note	AMPHS0Note	AMPH

EXCLK	OSCSEL	Operating mode of the high-speed system clock pin	X1/P121 pin	X2/EXCLK/P122 pins
0	0	Enter the port mode	Enter the port	
0	1	X1 oscillation mode	Connect a crystal or ceramic resonator.	
1	0	Enter the port mode	Enter the port	
1	1	External clock input mode	Enter the port	External clock input

EXCLKS	OSCSELS	Operating mode of the subsystem clock pins	XT1/P123 pin	XT2/EXCLKS/P124 Pins
0	0	Enter the port mode	Enter the port	
0	1	XT1 oscillation mode	Connect the crystal resonator.	
1	0	Enter the port mode	Enter the port	
1	1	External clock input mode	Enter the port	External clock input

AMPHS1	AMPHS0	Oscillation mode selection for the XT1 oscillation circuit
0	0	Low-power oscillation (default)
0	1	Usual oscillations
1	0	Ultra-low power oscillation
1	1	Prohibit settings.

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

Note: The EXCLKS, OSCSELS, AMPHS1, and AMPHS0 bits are initialized only during power-on reset, while remaining unchanged at other resets.

- Note 1. After the reset is released, the CMC register can only be written once via the 8-bit memory operation instruction. When using the CMC register at the initial value ("00H"), in order to prevent the program from malfunctioning if it is out of control (if you mistakenly write a value other than "00H", it is necessary to set the CMC register after reset released "00H").
- The CMC register must be set after the reset is released and before the X1 or XT1 oscillation begins by setting the clock operating state control register (CSC).
 - When the X1 clock oscillation frequency exceeds 10MHz, the APH position must be "1".
 - Configuration of AMPH bit, AMPHS1 bit, and AMPHS0 bit Must be after reset released and in the state of selecting f_{IH} as the f_{CLK} (toggle f_{CLK} to f_{MX} or f_{SUB} previous state).
 - The oscillation settling time of f_{XT} must be counted by software.

The upper frequency limit of the system clock is 64MHz, but the frequency limit of the X1 oscillation circuit is 20 MHz.

Note f_X : 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 master system clock.

The CKC register is set by the 8-bit memory operation instruction.

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

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

Address: 40020404H reset: 00HR/W Note1

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

CLS	The status of the CPU/Peripheral Hardware Clock (f_{CLK})
0	The main system clock (f_{MAIN})
1	Subsystem clock (f_{SUB}).

CSSnote2	Selection of CPU/Peripheral Hardware Clock (f_{CLK})
0	The main system clock (f_{MAIN})
1	Subsystem clock (f_{SUB})

MCS	The state of the master system clock (f_{MAIN})
0	High-speed internal oscillator clock (f_{IH})
1	High-speed system clock (f_{MX})

MCM0 Note	Operational 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.

2. It is forbidden to change the value of the MCM0 bit in the state of placing the CSS position "1".

Note f_{HOCO} : The clock frequency of the high-speed internal oscillator (64MHz maximum).
 f_{IH} : Clock frequency of high-speed internal oscillator (64MHz maximum)
 f_{MX} : High-speed system clock frequency
 f_{MAIN} : The main system clock frequency
 f_{SUB} : Subsystem clock frequency

Note 1 You must set bit0~3 to "0".

- Provide a clock with CSS bit settings to the CPU and peripheral hardware. If you change the CPU clock, you also change the clock of the peripheral hardware (except for the real-time clock, 15-bit interval timer, clock output/buzzer output, and watchdog timer). Therefore, if you want to change the clock of the CPU/peripheral hardware, you must stop the peripheral functions.
- If the secondary system clock is used as a peripheral hardware clock, the operation of the A/D converter and IICA cannot be guaranteed. For the operating characteristics of the peripheral hardware, refer to the sections on each peripheral hardware and the electrical characteristics of the data sheet.

4.3.3 Clock Operating State 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 sub-system clock (except for the low-speed internal oscillator clock). Set the CSC registers via the 8-bit memory operation instructions.

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

Figure 4-4 Clock Operating State Control Register (CSC).

Address: 40020401H reset: C0H R/W

symbol	7	6	5	4	3	2	1	0
CSC	MSTOn	XTSTOn	0	0	0	0	0	HIOSTOn

MSTOP	Operational control of high-speed system clocks		
	X1 oscillation mode	External clock input mode	Enter the port mode
0	The X1 oscillation circuit operates	The external clock on the EXCLK pin is valid	Enter the port
1	The X1 oscillation circuit stops	The external clock on the EXCLK pin is invalid	

XTSTOP	Operational control of the sub-system clock		
	XT1 oscillation mode	External clock input mode	Enter the port mode
0	The XT1 oscillation circuit operates	The external clock on the EXCLKS pin is valid	Enter the port
1	The XT1 oscillation circuit stops	The external clock of the EXCLKS pin is invalid	

HIOSTOP	Operational control of the high-speed internal oscillator clock	
0	High-speed internal oscillator operation	
1	High-speed internal shaker stop	

Note 1 After the reset is released, the CSC register must be set after the clock operating mode control register (CMC) is set.

2. After reset released and before placing the MSTOP position "0", the Oscillation Settling Time Selection Register (OSTS) must be set. However, when using the OSTS register at the initial value, there is no need to set the OSTS register.
3. To start the oscillation of X1 by setting the MSTOP bit, the oscillation settling time of the X1 clock must be confirmed by the state register (OSTC) of the oscillation settling time counter.
4. To start XT1 oscillation by setting the XTSTOP bit, it is necessary to wait through the software for the desired oscillation stabilization time of the subsystem clock.
5. The clock that is selected as the CPU/Peripheral Hardware Clock (f_{CLK}) cannot be stopped through the CSC registers.
6. Refer to Table Table 4-2).

Table 4-2 Clock Stop Method

clock	Condition before clock stop (external clock input is invalid)	Flag setting for CSC registers
X1 clock External master system clock	The CPU/peripheral hardware clock runs on a clock other than the high-speed system clock. (CLS=0 and MCS=0, or CLS=1).	MSTOP=1
XT1 clock External subsystem clock	The CPU/peripheral hardware clock runs on a clock other than the secondary system clock. (CLS=0)	XTSTOP=1
High-speed internal oscillator clock	The CPU/peripheral hardware clock runs at a clock other than the high-speed internal oscillator clock. (CLS=0 and MCS=1, or CLS=1).	HIOSTOP=1

4.3.4 PLL Control Register (PLLCR) for System Clock

This is the register that the control system clock runs with the PLL. Set the PLLCR registers via the 8-bit memory operation instructions.

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

Figure 4-5 Format of the system clock with the PLL Control Register (PLLCR).

Address: 40020C02H after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
PLLCR	PLLSRSEL	0	0	0	PLLD1	PLLD0	PLLM	PLLON

PLLSRSEL	The input clock source of the PLL is selected
0	Select the high-speed internal oscillator clock F_{IH} as the input clock source for the PLL
1	Select the external master system clock F_{MX} as the input clock source for the PLL

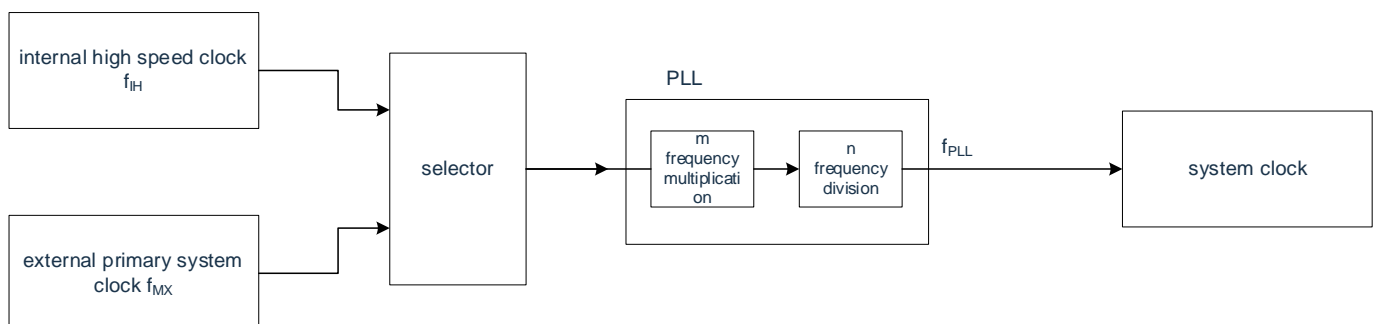
PLLD1	PLLD0	PLL Divider selection
0	0	No Dividers
0	1	Divide-by-2
1	x	Divide-by-4

PLLM	PLL multiplication option
0	12 times
1	16 times

PLLON	PLL work enable
0	PLL turns off
1	PLL turns on

When using PLL as the system clock, the clock structure is shown in the following figure, where m is 12/16, which is determined by the setting value of PLLM, and n is 1/2/4, which is determined by the setting value of PLD1/PLLD0.

Figure 4-6 The clock structure of the PLL when the system clock is used



Note: To use the PLL as the system clock, the bit4 (MCM0) and bit6 (CSS) of the CKC register must be set to 0.

4.3.5 The state register (OSTC) of the oscillation settling time counter

This is a register that represents the oscillation settling time counter count state of the X1 clock. The oscillation settling time of the X1 clock can be confirmed in the following cases:

- When the CPU clock is a high-speed internal oscillator clock or a subsystem clock and oscillation of the X1 clock begins
- When the CPU clock is a high-speed internal oscillator clock and the X1 clock oscillates is shifted into deep sleep mode and then exited

OSTC registers can be read via 8-bit memory operation instructions. By the generation of a reset signal, into deep sleep mode, or the MSTOP bit (bit7 of the Clock Operating State Control Register (CSC)) set to "1", the value of the register changes to "00H".

Note: The oscillation settling time counter starts counting in the following cases:

- When the X1 clock begins to oscillate (EXCLK, OSCSEL=0, 1 MSTOP=0).
- When deep sleep mode is exited

Figure 4-7 The format of the state register (OSTC) of the oscillation settling time counter

Address: 40020402H after reset: 00H R

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

MOS T8	MOST 9	MOST 10	MOST 11	MOST 13	MOST 15	MOST 17	MOST 18	Oscillation steady time state		
									$f_X=10\text{MHz}$	$f_X=20\text{MHz}$
0	0	0	0	0	0	0	0	Less than $2^8/f_X$	Less than 25.6us	Less than 12.8us
1	0	0	0	0	0	0	0	At least $2^8/f_X$	At least 25.6us	At least 12.8us
1	1	0	0	0	0	0	0	At least $2^9/f_X$	At least 51.2us	At least 25.6us
1	1	1	0	0	0	0	0	At least $2^{10}/f_X$	At least 102us	At least 51.2us
1	1	1	1	0	0	0	0	At least $2^{11}/f_X$	At least 204us	At least 102us
1	1	1	1	1	0	0	0	At least $2^{13}/f_X$	At least 819us	At least 409us
1	1	1	1	1	1	0	0	At least $2^{15}/f_X$	At least 3.27ms	At least 1.63ms
1	1	1	1	1	1	1	0	At least $2^{17}/f_X$	At least 13.1ms	At least 6.55ms
1	1	1	1	1	1	1	1	At least $2^{18}/f_X$	At least 26.2ms	At least 13.1ms

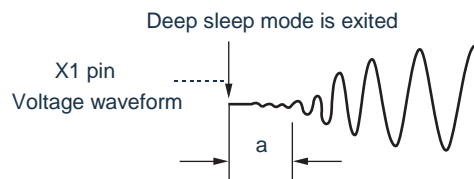
Note 1 After the above time, the bits will change to "1" in turn starting from bit MOST8 and remain in the state of "1".

2. The Oscillation Settling Time Counter only counts within the Oscillation Settling Time Select Register (OSTS) set by the Oscillation Settling Time Register (OSTS). In the following cases, the setting value of the oscillation settling time of the OSTS register must be greater than the count value confirmed by the OSTC register.

- When the CPU clock is a high-speed internal oscillator clock or a subsystem clock and you want to start oscillation of the X1 clock
- When the CPU clock is a high-speed internal oscillator clock and the deep sleep mode is exited after the X1 clock oscillates to deep sleep mode

(It must therefore be noted that the OSTC register after deactivating the deep sleep mode only sets the state within the oscillation stabilization time set by the OSTS register.)

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



Note f_X : X1 clock oscillation frequency

4.3.6 Oscillation settling time selection register (OSTS).

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

If the X1 clock oscillates, the time to automatically wait for the OSTS register to be set just after the X1 oscillation circuit runs (MSTOP=0).

If the CPU clock is switched from a high-speed internal oscillator clock or a sub-system clock to an X1 clock, or if the CPU clock is a high-speed internal oscillator clock and the deep sleep mode is transferred to deep sleep mode in the state of X1 clock oscillation, deep sleep mode is exited, The oscillation settling time counter must be used to confirm whether the oscillation settling time has passed through the state register (OSTC) of the oscillation settling time counter.

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

Set the OSTS register via the 8-bit memory operation instruction. After generating a reset signal, the value of this register becomes "07H".

Figure 4-8 Oscillation Settling Time Selection Register (OSTS) Format

Address: 40020403H after reset: 07H R/W

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

OSTS2	OSTS1	OSTS0		The selection of oscillation stabilization time	
				$f_X=10\text{MHz}$	$f_X=20\text{MHz}$
0	0	0	$2^8/f_X$	25.6us	12.8us
0	0	1	$2^9/f_X$	51.2us	25.6us
0	1	0	$2^{10}/f_X$	102us	51.2us
0	1	1	$2^{11}/f_X$	204us	102us
1	0	0	$2^{13}/f_X$	819us	409us
1	0	1	$2^{15}/f_X$	3.27ms	1.63ms
1	1	0	$2^{17}/f_X$	13.1ms	6.55ms
1	1	1	$2^{18}/f_X$	26.2ms	13.1ms

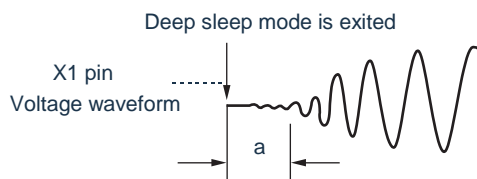
Note 1 To change the setting of the OSTS register, you must change it before placing the MSTOP position of CLOCK Operating State Control Register (CSC) to "0".

2. The oscillation settling time counter is only counted within the oscillation settling time set by the OSTS register.

In the following cases, the setting value of the oscillation settling 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 you want to start oscillation of the X1 clock
- When the CPU clock is a high-speed internal oscillator clock and the deep sleep mode is exited after the X1 clock oscillates to deep sleep mode (so it must be noted that the OSTC register after deactivating deep sleep mode only sets OSTS.) The state within the oscillation settling time set by the register).

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



Note f_X : X1 clock oscillation frequency

4.3.7 Peripheral enable registers 0, 1, 2, 3 (PER0, PER1, PER2, PER3)

This is the register that the setting enable or disables the provision of clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use.

When using the following peripheral functions controlled by these registers, the corresponding position "1" must be placed before the initial setup of the peripheral functions.

- Real-time clock, 15-bit interval timer
- A/D converter
- Serial interface IICA0/1
- Universal serial communication unit SCIO1/2
- Serial interface SPI0/1 ^{Note}
- afcan0/1/2^{note}
- Timer40
- Timer81
- D/A converter
- Timer B
- Comparator
- Timer M
- Enhanced DMA
- PWMOP
- Timer C
- Timer A
- LCD bus interface ^{note}

Note: The serial interface SPI0/1, afcan2, LCD bus interface is a BAT32A279 specific function.

Set the PER0 registers, PER1 registers, PER2 registers, and PER2 registers via the 8-bit memory operation instructions PER3 register. After generating a reset signal, the value of these registers changes to "00H".

Figure 4-9 The format of Peripheral enable register 0 (PER0) (1/3).

Address: 40020420H After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
PER0	RTCCEN Note	-	ADCIN	IICA0EN	SCIO1IN	SCIOEN	CAN0EN	TM40EN

RTCCEN	Provides control of the input clock of the real-time clock (RTC) and a 15-bit interval timer
0	Stop providing the input clock. • You cannot write SFR for real-time clocks (RTCs) and 15-bit interval timers. • The real-time clock (RTC) and 15-bit interval timer are in reset state.
1	An input clock is provided. • Read and write SFR for real-time clock (RTC) and 15-bit interval timer.

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

Figure 4-9 The format of Peripheral enable register 0 (PER0) (2/3).

Address: 40020420H After reset: 00H R/W

SYMBOL	7	6	5	4	3	2	1	0
PER0	BTIIN	-	ADCIN	IICA0ANDN	SCI1IN	SCI0EN	CAN0EN	TM40In

ADCEN	Provides control of the input clock of the A/D converter
0	Stop providing the input clock. • Cannot write SFR used by A/D converters. • The A/D converter is in reset state.
1	An input clock is provided. • Read and write SFR used by A/D converters.

IICA0EN	Provides control of the input clock of the serial interface IICA0
0	Stop providing the input clock. • Cannot write SFR used by the serial interface IICA0. • The serial interface IICA0 is in reset state.
1	An input clock is provided. • Can read and write SFR used by the serial interface IICA0.

SCI1EN	Provides control of the input clock of the Universal Serial Communication Unit 1
0	Stop providing the input clock. • Cannot write SFR for Universal Serial Communication Unit 1. • Universal Serial Communication Unit 1 is in reset state.
1	An input clock is provided. • SFR can read and write to universal serial communication unit 1.

SCI0EN	Provides control of the input clock of Universal Serial Communication Unit 0
0	Stop providing the input clock. • Cannot write SFR used by Universal Serial Communication Unit 0. • Universal serial communication unit 0 is in reset state.
1	An input clock is provided. • SFR that can read and write universal serial communication unit 0.

Figure 4-9 The format of Peripheral enable register 0 (PER0) (3/3).

Address: 40020420H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	-	ADCIN	IICA0EN	SCI1IN	SCI0EN	CAN0EN	TM40EN

CAN0EN	Provides control of the input clock of the CAN module
0	Stop providing the input clock. • Cannot write CAN0 to use SFR. • CAN0 is in reset state.
1	An input clock is provided. • Can read and write SFR used by CAN0.

TM40EN	Provides control of the input clock of Timer4
0	Stop providing the input clock. • SFR used by universal timer unit 0 cannot be written. • Universal timer unit 0 is in reset state.
1	An input clock is provided. • SFR for reading and writing universal timer unit 0.

Figure 4-10 The format of Peripheral enable register 1 (PER1) (1/2).

Address: 4002081AH after reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
PER1	DACEN	TMBAndN	PGACMPEN	TMMEN	DMAIn	PWMOPEN	TMCEN	TSTONE

DACEN	Provides control of the input clock of the D/A converter
0	Stop providing the input clock. • Cannot write SFR used by D/A converters. • The D/A converter is in reset state.
1	An input clock is provided. • Can read and write SFR used by D/A converters.

TMBEN	Provides control of the input clock of timer B
0	Stop providing the input clock. • Cannot write SFR for timer B. • Timer B is in reset state.
1	An input clock is provided. • SFR can be read and written to timer B.

PGACMPEN	Provides control of the input clock of the amplifier and comparator
0	Stop providing the input clock. • Cannot write amplifiers and comparators used by SFR. • The amplifier comparator is in reset state.
1	An input clock is provided. • Can read and write SFR for amplifiers and comparators.

TMMEN	Provides control of the input clock of timer M
0	Stop providing the input clock. • SFR cannot be written to timer M. • Timer M is in reset state.
1	An input clock is provided. • Can read and write the SFR used by the timer M.

Figure 4-10 The format of Peripheral enable register 1 (PER1) (2/2).

Address: 4002081AH after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
PER1	DACEN	TMBEN	PGACMPEN	TMMEN	DMAIn	PWMOPEN	TMPices	TMAEN

DMAEN	Provides control of the input clock of the DMA
0	Stop providing the input clock. •DMA cannot run.
1	An input clock is provided. • DMA can run.

PWMOPEN	PWM cut-off circuit control of the input clock
0	Stop providing the input clock. • Cannot write SFR for PWM cut-off circuits. •The PWM cut-off circuit is in reset state.
1	An input clock is provided. • Can read and write SFR used in PWM cut-off circuits.

TMCEN	Provides control of the input clock of Timer C
0	Stop providing the input clock. • SFR cannot be written to Timer C. • Timer C is in reset state.
1	An input clock is provided. • SFR can be read and written to Timer C.

TMAEN	Provides control of the input clock of Timer A
0	Stop providing the input clock. • Cannot write SFR for Timer A. • Timer A is in reset state.
1	An input clock is provided. • Can read and write the SFR used by Timer A.

Figure 4-11 The format of Peripheral enable register 2 (PER2) (1/2).

Address: 4002081BH reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
PER2	SPIHS1EN	SPIHS0EN	CAN2EN	OSDCIN	SCI2EN	IICA1	CAN1IN	TM81EN

OSDCIN	Control of the input clock that provides a clock stop detection function
0	Stop providing the input clock.
1	An input clock is provided.

SCI2EN	Provides control of the input clock of the universal serial communication unit 2
0	Stop providing the input clock. • Cannot write SFR for Universal Serial Communication Unit 2. • Universal Serial Communication Unit 2 is in reset state.
1	An input clock is provided. • Can read and write SFR used by Universal Serial Communication Unit 2.

IICA1EN	Provides control of the input clock of the serial interface IICA1
0	Stop providing the input clock. • Cannot write the serial interface IICA1 using the SFR. • Serial interface IICA1 is in reset state.
1	An input clock is provided. • SFR can read and write serial interface IICA1.

CAN1IN	Provides control of the input clock of the CAN1 module
0	Stop providing the input clock. • Cannot write CAN1 using SFR. • CAN1 is in reset state.
1	An input clock is provided. • Can read and write CAN1 using SFR.

TM81EN	Provides control of the input clock of the Timer8
0	Stop providing the input clock. • Cannot write SFR for universal timer unit 1. • Universal timer unit 0 is in reset state.
1	An input clock is provided. • SFR can read and write to universal timer unit 1.

Note: The serial interface SPI0/1, afcan2, LCD bus interface is a BAT32A279 specific function, so the BIT 5~7 of PER2 must be set to 1'b0 when BAT32A239 products.

Figure 4-11 The format of Peripheral enable register 2 (PER2) (2/2).

Address: 4002081BH reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
PER2	SPIHS1EN	SPIHS0EN	CAN2EN	OSDCIN	SCI2EN	IICA1	CAN1IN	TM81EN

CAN2EN	Provides control of the input clock of the CAN2 module
0	Stop providing the input clock. • Cannot write CAN2 using SFR. • CAN2 is in reset state.
1	An input clock is provided. • Can read and write CAN2 using SFR.

SPIHS0IN	Provides control of the input clock of the serial interface SPI0
0	Stop providing the input clock. • Cannot write SFR for serial interface SPI0. • Serial interface SPI0 is in reset state.
1	An input clock is provided. • Can read and write SFR used by the serial interface SPI0.

SPIHS1EN	Provides control of the input clock of the serial interface SPI1
0	Stop providing the input clock. • Cannot write SFR for serial interface SPI1. • Serial interface SPI1 is in reset state.
1	An input clock is provided. • Can read and write SFR for serial interface SPI1.

Note: The serial interface SPI0/1, afcan2, LCD bus interface is a BAT32A279 specific function, so the BIT 5~7 of PER2 must be set to 1'b0 when BAT32A239 products.

Figure 4-12 The format of Peripheral enable register 3 (PER3).

Address: 4002081CH after reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
PER3	0	0	0	0	0	0	0	LCDBIN

LCDBIN	Provides control of the input clock of the LCDB module
0	Stop providing the input clock. • Cannot write SFR used by LCDB. • The LCDB is in reset state.
1	An input clock is provided. • Can read and write SFR used by LCDB.

Note: The serial interface SPI0/1, afcan2, LCD bus interface is a unique function of BAT32A279, so the PER3 register is inaccessible when BAT32A239 products are used.

4.3.8 Subsystem clock supply mode control register (OSMC).

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

If the RTCLP position is "1", the clock is stopped in deep sleep mode or the CPU running in a sleep mode with a sub-system clock for peripheral functions other than the real-time clock and the 15-bit interval timer, thus reducing power consumption.

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

Set the OSMC registers via the 8-bit memory operation instructions.

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

Figure 4-13 subsystem clock provides the format of the mode control register (OSMC).

Address: 40020423H reset: 00H R/W

SYMBOL	7	6	5	4	3	2	1	0
OSMC	RTCLPC	0	0	WUTMMCK0	0	0	0	0

RTCLPC	The settings in deep sleep mode and sleep mode where the CPU runs on the subsystem clock
0	Allows a subsystem clock to be provided for peripheral functions (For peripheral functions that are allowed to operate, please refer to Table 2 7-1 ~ Table 2 7-3.)
1	Stop providing a subsystem clock for peripheral functions other than the real-time clock and the 15-bit interval timer.

WUTMMCK0	Selection of operating clocks for real-time clocks, 15-bit interval timers, and timer A
0	<ul style="list-style-type: none"> The secondary system clock is a real-time clock and a 1 5-bit interval timer for the operating clock. The low-speed internal oscillator cannot be selected as the counting source for timer A.
1	<ul style="list-style-type: none"> The low-speed internal oscillator clock is a real-time clock and a 15-bit interval timer for the operating clock. Low-speed internal oscillator or subsystem clock can be selected as the counting source for timer A.

4.3.9 Frequency Selection Register (HOCODIV) for high-speed internal oscillators

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

The HOCODIV register is set by the 8-bit memory operation instruction.

After generating a reset signal, the value of this register changes to the setting value of the FRQSEL2~FRQSEL0 bit of the option byte (000C2H).

Figure 4-14 Format of Frequency Selection Register (HOCODIV) for the high-speed internal oscillator

Address: 40021C20H Reset value: The value of the FRQSEL2~FRQSEL0 bits of the option byte(000C2H) R/W.

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

HOCODIV2	HOCODIV1	HOCODIV0	Selection of high-speed internal oscillator clock frequency			
			FRQSEL4=0		FRQSEL4=1	
			FRQSEL3=0	FRQSEL3=1	FRQSEL3=0	FRQSEL3=1
0	0	0	$f_{IH}=24\text{MHz}$	$f_{IH}=32\text{MHz}$	$f_{IH}=48\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=64\text{MHz}$ $f_{HOCO}=64\text{MHz}$
0	0	1	$f_{IH}=12\text{MHz}$	$f_{IH}=16\text{MHz}$	$f_{IH}=24\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=16\text{MHz}$ $f_{HOCO}=64\text{MHz}$
0	1	0	$f_{IH}=6\text{MHz}$	$f_{IH}=8\text{MHz}$	$f_{IH}=12\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=8\text{MHz}$ $f_{HOCO}=64\text{MHz}$
0	1	1	$f_{IH}=3\text{MHz}$	$f_{IH}=4\text{MHz}$	$f_{IH}=3\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=4\text{MHz}$ $f_{HOCO}=64\text{MHz}$
1	0	0	Prohibit settings.	$f_{IH}=2\text{MHz}$	Prohibit settings.	$f_{IH}=2\text{MHz}$ $f_{HOCO}=64\text{MHz}$
1	0	1	Prohibit settings.	$f_{IH}=1\text{MHz}$	Prohibit settings.	$f_{IH}=1\text{MHz}$ $f_{HOCO}=64\text{MHz}$
Other than the above			Prohibit settings			

Note 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 (fCLK).

2. After changing the frequency through the HOCODIV register, the frequency switching takes place after the following transfer times:

- Run up to 3 clocks at the frequency before the change.
- Wait up to 3 CPU/peripheral hardware clocks at the changed frequency.

4.3.10 Trimming Register (HIOTRM) for high-speed internal oscillator

This is a register that corrects for the accuracy of the high-speed internal oscillator. Self-measurement and accuracy correction of high-speed internal oscillator frequency can be performed using a timer with a high-precision external clock input, etc. The HIOTRM register is set via the 8-bit memory operation instruction.

Note that if the temperature and voltage of the V_{DD} pin change after correcting the accuracy, the frequency changes.

In the event of a change in temperature and voltage at the V_{DD} pin, corrections need to be made before requiring accuracy of the frequency or periodically.

Figure 4-15 The format of the Trimming Register (HIOTRM) of the high-speed internal oscillator

Address: 40021C00H reset value: Note R/W

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

HIOTRM5	HIOTRM4	HIOTRM3	HIOTRM2	HIOTRM1	HIOTRM0	High-speed internal oscillator	
0	0	0	0	0	0	Minimum speed	
0	0	0	0	0	1	↑	
0	0	0	0	1	0		
0	0	0	0	1	1		
0	0	0	1	0	0		
•							
•							
•							
1	1	1	1	1	0	▼	
1	1	1	1	1	1		Highest speed

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

Note 1. Each bit of the HIOTRM register can correct the clock accuracy of the high-speed internal oscillator by about 0.05%.

4.3.11 Subsystem Clock Select Register (SUBCKSEL)

The SUBCKSEL register is a register that selects the subsystem clock f_{SUB} and the low-speed internal oscillator clock F_{IL} , as well as the low-speed internal oscillator clock frequency.

Set the SUBCKSEL registers with 8-bit memory operation instructions.

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

Figure 4-16 Format of subsystem clock selection registers (SUBCKSEL).

Address: 40020407H reset: 00H R/W

SYMBOL	7	6	5	4	3	2	1	0
SUBCKSEL	0	0	0	0	0	0	LOCOSSEL	SELLOSC

LOCOSSEL	Frequency selection for low-speed internal oscillator clocks
0	• Low speed internal oscillator clock frequency is 15K.
1	• Low speed internal oscillator clock frequency is 30K

SELLOSC	Selection of sub-system clock and low-speed internal oscillator clock
0	• Select the subsystem clock.
1	• Select low-speed internal oscillator clock.

4.3.12 Master System Clock Control Register (MCKC).

MCKC registers are registers used to control the clock of the main system.

The MCKC register is set via the 8-bit memory operation instruction.

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

Figure 4-17 Format of the main system clock control register (MCKC).

Address: 40020C00H After reset: 00H R/W ^{Note 1}

SYMBOL	7	6	5	4	3	2	1	0
MCKC	CKSTR	0	0	0	0	PDIV1	PDIV0	CKSELR

CKSTR	The state of the high-speed internal oscillator clock and PLL clock selection
0	• Select high-speed internal oscillator clock.
1	• Select the PLL clock.

PDIV1	PDIV0	Divider selection for the PLL clock
0	0	No Dividers
0	1	Divide-by-2
1	0	Divide-by-4
1	1	8-way

CKSELR	High-speed internal oscillator clock and PLL clock selection
0	• Select high-speed internal oscillator clock.
1	• Select the PLL clock.

Note 1: Bit7 is read-only.

4.4 System clock oscillation circuit

4.4.1 X1 oscillation circuit

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

When using X1 oscillation circuitry, the bit7 and bit6 (EXCLK, OSCSEL) inputs of registers (CMC) must be controlled for the clock operating mode. Line the following settings:

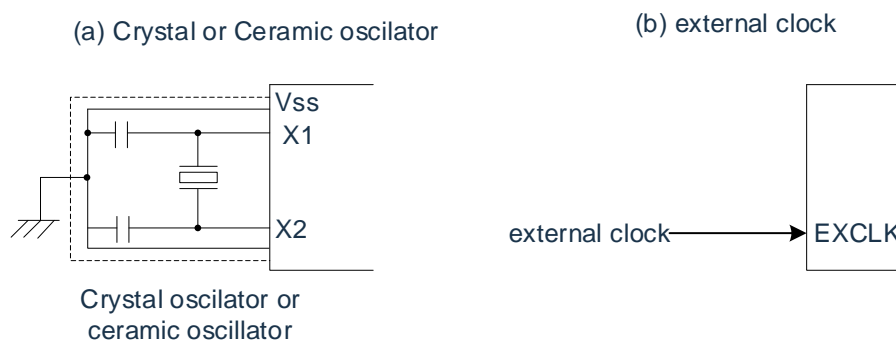
- Crystal or ceramic oscillation: EXCLK, OSCSEL=0, 1
- External clock inputs: EXCLK, OSCSEL=1, 1

When the X1 oscillation circuit is not used, it must be set to input port mode (EXCLK, OSCSEL=0, 0).

Also, when it is not used as an input port, refer to "Handling of Unused Pins in Table 2-4".

An example of an external circuit for the X1 oscillation circuit is shown in Figure Fig. 4-18.

Fig. 4-1818 X1 oscillation circuit



Precautions are shown on the following page.

4.4.2 XT1 oscillation circuit

The XT1 oscillation circuit passes through a crystal resonator (32.768kHz (TYP.)) connecting the XT1 pin to the XT2 pin) to oscillate. When using the XT1 oscillation circuit, the bit4 (OSCSELS) of the clock operating mode control register (CMC) must be set to "1" to be used to input an external clock, which must be given The EXCLKS pin input clock signal.

When using XT1 oscillation circuitry, the clock operating mode must control bit5 and bit4 (EXCLKS, OSCSELS) of registers (CMC) (make the following settings:

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

When the XT1 oscillation circuit is not used, it must be set to input port mode (EXCLKS, OSCSELS=0, 0).

Also, when it is not used as an input port, refer to "Handling of Unused Pins in Table 2-4". An example of an external circuit for the XT1 oscillation circuit is shown in Figure Fig. 4-19.

Fig. 4-19 XT1 oscillation circuit external circuit example



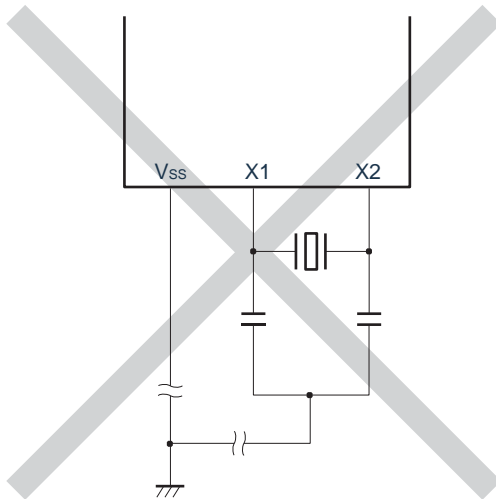
Note that when using the X1 oscillation circuit and the XT1 oscillation circuit, in order to avoid the influence of the wiring capacitance, etc., the Fig. 4-18418 Fig. 4-19419 must be routed by the following method:

- The wiring must be kept as short as possible.
- Cannot cross with other signal lines and cannot approach the wiring through which large currents vary.
- The capacitor grounding point of the oscillation circuit must always be the same as the V_{SS} potential, and the grounding pattern of the high current flowing through must not be grounded.
- Signals cannot be taken out of the oscillation circuit.

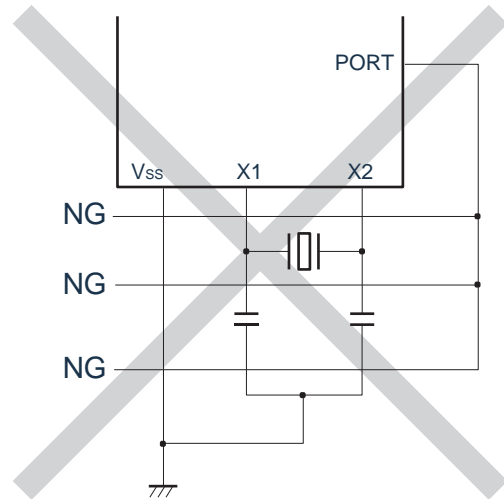
An example of an incorrect resonator connection is shown in Figure Figure 4-20.

Figure 4-20 Example of incorrect resonator connection (1/2).

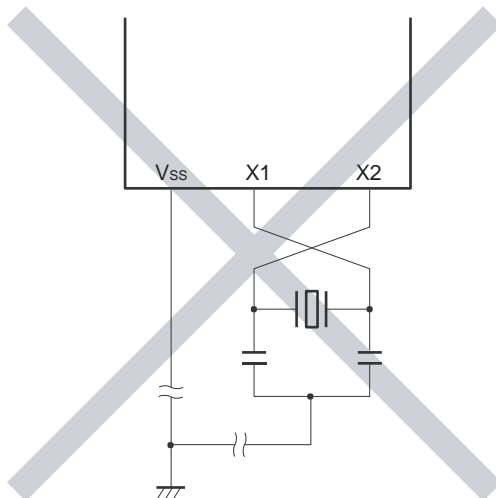
(a) The wiring connecting the circuit is too long



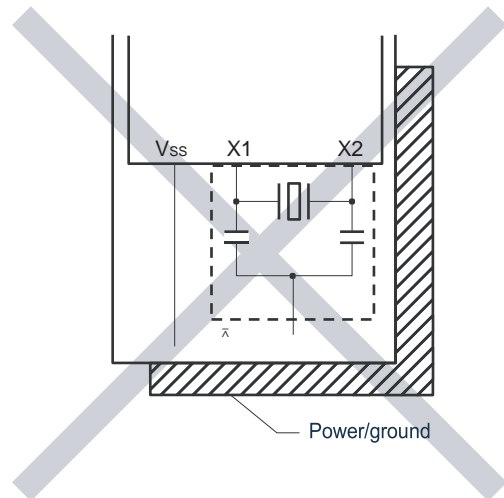
(b) the signal line crosses



(c) Signal lines for X1 and X2 cross-routing wiring of X1 and X2



(d) There is a power or ground graphic below the wiring of X1 and X2



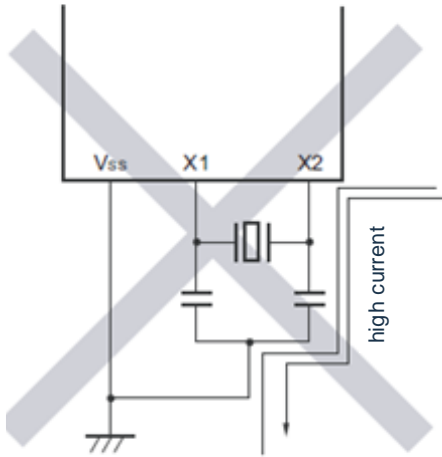
Note In a multilayer board or double-sided panel, you cannot configure the power supply or ground graphics below the wiring area of the X1 pin, X2 pin, and resonator (the dashed part of the figure). The wiring cannot produce a capacitive component that affects the oscillation characteristics.

Note In the case of using a subsystem clock, insert a series resistor on the XT2 side.

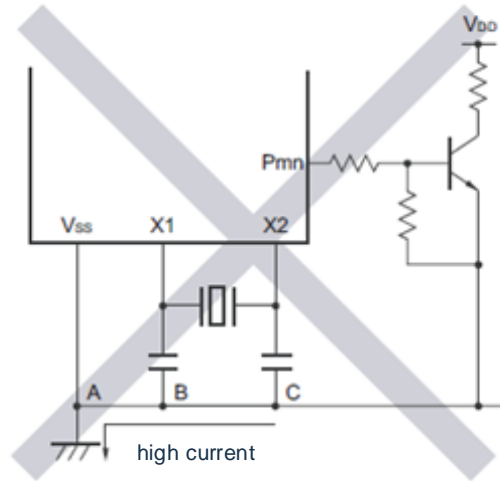
replace X1 and X2 with XT1 and XT2 respectively when reading, and

Figure 4-15 Example of incorrect resonator connection (2/2).

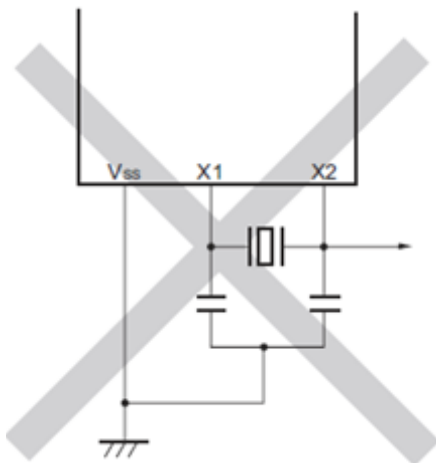
(e) varying high current source close to singal lines



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



(g) extracted signal



Note that when the X2 and XT1 are routed in parallel, the crosstalk noise of the X2 will be superimposed on the XT1 and cause malfunction.

Note In the case of using a subsystem clock, and insert a series resistor on the XT2 side.

replace X1 and X2 with XT1 and XT2 respectively when reading,

4.4.3 High-speed internal oscillator

The BAT32A279 has a built-in high-speed internal oscillator. Can choose the frequency from 64MHz, 48MHz, 32MHz, 24MHz, 16MHz, 12MHz, 8MHz, 6MHz, 4MHz, 3MHz, 2MHz and 1MHz via option byte (000C2H). Oscillation can be controlled by bit0 (HIOSTOP) of the clock operating state control register (CSC).

After the reset is released, the high-speed internal oscillator automatically begins oscillating.

4.4.4 Low-speed internal oscillator

The BAT32A279 has a low-speed internal oscillator.

The low-speed internal oscillator clock is used as a watchdog timer, a real-time clock, a clock for a 15-bit interval timer, and a clock for timer A, as well as an external reference clock for the SysTick timer, but cannot be used as a CPU clock.

When the bit4(WDTON)of option byte(000C0H) or bit4 (WUTMMCK0) of the subsystem clock provides mode control register (OSMC) oscillates at a low speed internal oscillator when it is "1".

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", then the WDSTBYON bit is "0" and is in sleep mode, deep sleep the low-speed internal oscillator stops oscillating in 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.

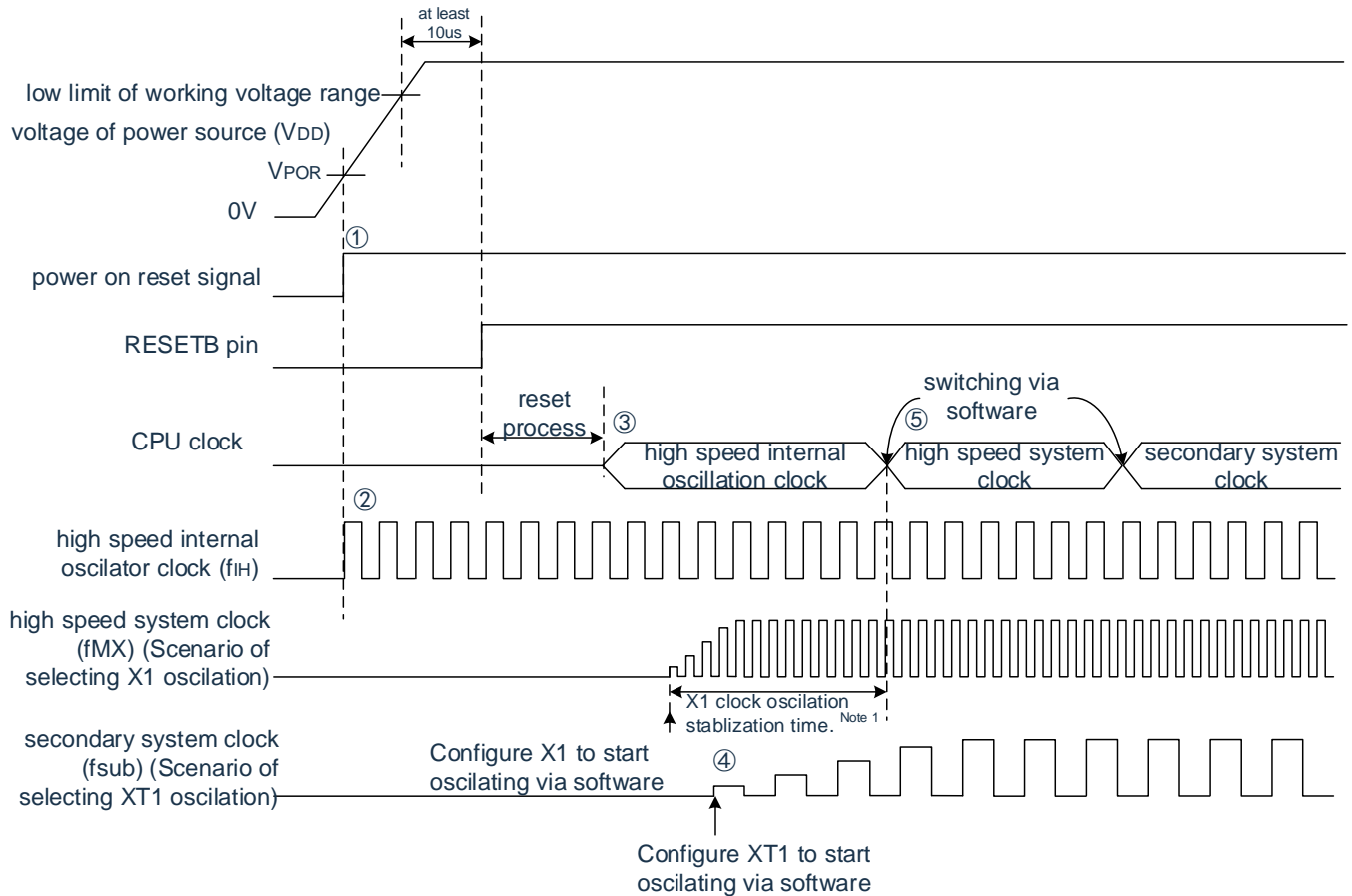
4.5 Clock generation circuit operation

The clock generation circuit generates the various clocks shown below and controls the operating mode of the CPU such as standby mode (see Figure Figure 4-1).

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

After the BAT32A279 reset is released, the CPU starts operating through the output of the high-speed internal oscillator. The operation of the clock generation circuit when the power is turned on is shown in Figure Figure 4-21.

Figure 4-21 The clock generation circuit is running when the power is turned on



- ① After power is turned on, an internal reset signal is generated by a power-on reset (POR) circuit. However, the reset state is maintained by voltage detection circuitry or an external reset until the operating voltage range shown in the AC characteristics of the data sheet is reached (the figure above is an example of when using an external reset).
- ② If the reset is released, the high-speed internal oscillator automatically begins oscillating.
- ③ After the reset is released, a voltage stabilization wait and reset process is performed, and then the CPU starts running with a high-speed internal oscillator clock.
- ④ The X1 clock or the start oscillation of the XT1 clock must be set by software (see "4.6.2 X1 Oscillation Circuit Setup Example" and "4. 6.3 Example of setup of the oscillation circuit of XT1").
- ⑤ If you want to switch the CPU clock to the X1 clock or the XT1 clock, you must switch it through the software settings after waiting for the clock oscillation to stabilize (see "4.6.2 X1.".) Example of setting up an oscillation circuit" and "Example of setting up a 4.6.3 XT1 oscillation circuit").

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

Note that if an external clock input from the EXCLK pin is used, there is no need for 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 be running at a high-speed internal oscillator clock. It can pass the FRQSEL0~FRQSEL4 bits of the option byte (000C2H) from 64MHz, 48MHz, and 32MHz, 24MHz, 16MHz, 12MHz, 8MHz, 6MHz, 4MHz, 3MHz, 2MHz, and 1MHz to select the frequency of the high-speed internal oscillator. In addition, the frequency can be changed by the frequency selection register (HOCODIV) of the high-speed internal oscillator.

【Setting of option bytes】

Address: 000C2H

Options byte (000C2H)	7	6	5	4	3	2	1	0
	1	1	1	FRQSEL4 0/1	FRQSANDL3 0/1	FRQSEL2 0/1	FRQSEL1 0/1	FRQSEL0 0/1

FRQSEL4	FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0	Frequency of a high-speed internal oscillator	
					f_{HOCO}	f_{IH}
1	1	0	0	0	64MHz	64MHz
1	0	0	0	0	48MHz	48MHz
0	1	0	0	0	32MHz	32MHz
0	0	0	0	0	24MHz	24MHz
0	1	0	0	1	32MHz	16MHz
0	0	0	0	1	24MHz	12MHz
0	1	0	1	0	32MHz	8MHz
0	0	0	1	0	24MHz	6MHz
0	1	0	1	1	32MHz	4MHz
0	0	0	1	1	24MHz	3MHz
0	1	1	0	0	32MHz	2MHz
0	1	1	0	1	32MHz	1MHz
Other than the above					Prohibit settings.	

[Setting of the Frequency Selection Register (HOCODIV) of the high-speed internal oscillator].

Address: 0x40021C20

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

HOCODIV2	HOCODIV1	HOCODIV0	Selection of high-speed internal oscillator clock			
			FRQSEL4=0		FRQSEL4=1	
			FRQSEL3=0	FRQSEL3=1	FRQSEL3=0	FRQSEL3=1
0	0	0	$f_{IH}=24\text{MHz}$	$f_{IH}=32\text{MHz}$	$f_{IH}=48\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=64\text{MHz}$ $f_{HOCO}=64\text{MHz}$
0	0	1	$f_{IH}=12\text{MHz}$	$f_{IH}=16\text{MHz}$	$f_{IH}=24\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=16\text{MHz}$ $f_{HOCO}=64\text{MHz}$
0	1	0	$f_{IH}=6\text{MHz}$	$f_{IH}=8\text{MHz}$	$f_{IH}=12\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=8\text{MHz}$ $f_{HOCO}=64\text{MHz}$
0	1	1	$f_{IH}=3\text{MHz}$	$f_{IH}=4\text{MHz}$	$f_{IH}=6\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=4\text{MHz}$ $f_{HOCO}=64\text{MHz}$
1	0	0	Prohibit settings.	$f_{IH}=2\text{MHz}$	$f_{IH}=3\text{MHz}$ $f_{HOCO}=48\text{MHz}$	$f_{IH}=2\text{MHz}$ $f_{HOCO}=64\text{MHz}$
1	0	1	Prohibit settings.	$f_{IH}=1\text{MHz}$	Prohibit settings.	$f_{IH}=1\text{MHz}$ $f_{HOCO}=64\text{MHz}$
Other than the above			Prohibit settings.			

4.6.2 Example of setting up the X1 oscillation circuit

After the reset is released, the CPU/Peripheral Hardware Clock (f_{CLK}) must be running at a high-speed internal oscillator clock. Thereafter, if the oscillation clock is changed to X1, the oscillation circuit is set and the oscillation start is controlled by the oscillation settling time selection register (OSTS), the clock operation mode control register (CMC), and the clock operating state control register (CSC). And wait for the oscillation to stabilize through the state register (OSTC) of the oscillation settling time counter. Set the X1 oscillation clock to f_{CLK} via the system clock control register (CKC) after waiting for the oscillation to stabilize.

【Register Setting】Registers must be set in the order of (1) to (5).

(1) The OSCSEL position of the CMC register is "1", and when f_X is greater than or equal to 10MHz, AMPH Bit set to "1" to make the X1 oscillation circuit run.

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

(2) The oscillation stabilization time of the X1 oscillation circuit when the deep sleep mode is deactivated through the OSTS register.

Example) To wait at least 102 us through a 10MHz resonator, it must be set to the following values.

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

(3) Clear the MSTOP bit of the CSC register to "0" so that the X1 oscillation circuit begins to oscillate.

	7	6	5	4	3	2	1	0
CSC	MSTOP	XTSTOP						HIOSTOP
	0	1	0	0	0	0	0	0

(4) Wait for the oscillation stabilization of the X1 oscillation circuit through the OSTC register.

Example) To wait at least 102 us through a 10MHz resonator, you must wait until you change to the following values.

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

(5) Set the X1 oscillation clock to the CPU/peripheral hardware clock through the MCM0 bit of the CKC register.

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

4.6.3 Example of setting up the XT1 oscillation circuit

After the reset is released, the CPU/Peripheral Hardware Clock (f_{CLK}) must be running at a high-speed internal oscillator clock. Thereafter, if the XT1 oscillation clock is changed, the mode control register (OSMC), clock operating mode control register (CMC), and clock operating state control register (CSC) are provided through the sub-system clock. The oscillation circuit is set up and the oscillation start is controlled, and the XT1 oscillation clock is set to f_{CLK} via the system clock control register (CKC).

【Register Setting】Registers must be set in the order of (1) to (5).

- (1) In deep sleep mode or the CPU running on the subsystem clock, the RTCPC bit must be set to "1" when the real-time clock and the 15-bit interval timer are running at the sub-system clock (ultra-low consumption current).

	7	6	5	4	3	2	1	0
OSMC	RTCLPC 0/1	0	0	WUTMMCK0 0	0	0	0	0

- (2) Place the OSCSELS position of the CMC register "1" so that the XT1 oscillation circuit runs.

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

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

- (3) Clear the XTSTOP bit of the CSC register to "0" so that the XT1 oscillation circuit begins to oscillate.

	7	6	5	4	3	2	1	0
CSC	MSTOP 1	XTSTOP 0	0	0	0	0	0	HIOSTOP 0

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

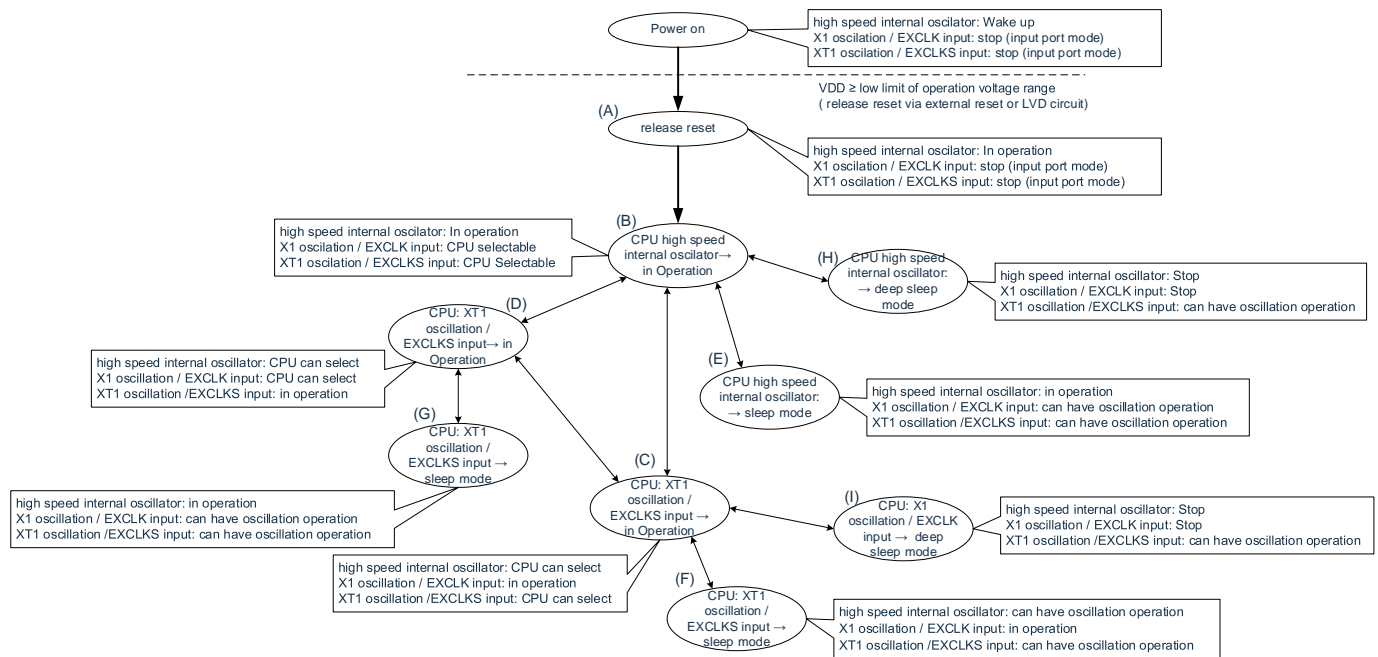
- (5) Set the XT1 oscillation clock to the CPU/peripheral hardware clock through the CSS bit of the CLC register.

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

4.6.4 State transition graph of the CPU clock

The CPU clock state transition diagram of this product is shown in Figure 4-22.

Figure 4-22 State transition diagram of the CPU clock



Examples of CPU clock transfers and SFR register settings are shown in Table 4-3.

Table 4-3 Example of CPU clock transfer and SFR register setting (1/5).

(1) After reset released (A), the CPU moves to high-speed internal oscillator clock operation (B).

State transition	Settings for the SFR registers
(A) (B)	There is no need to set the SFR register (the initial state after the reset is released).

(2) After reset released(A), the CPU is transferred to high-speed system clock operation (C).

(The CPU runs on a high-speed internal oscillator clock (B)) immediately after the reset is released.)

(SFR registers are set in order).

Setting flag for the SFR register State transition	CMC register Note 1			OSTS register	CSC register MSTOP	OSTC register	CKC register MCM0
	EXCLK	OSCSEL	AMPH				
(A)→(B)→(C) (X1 clock: $1\text{MHz} \leq f_X \leq 10\text{MHz}$)	0	1	0	Note 2	0	Confirmati on is required	1
(A)→(B)→(C) (X1 Clock: $10\text{MHz} < f_X \leq 20\text{MHz}$)	0	1	1	Note 2	0	Confirmati on is required	1
(A)→(B)→(C) (External Master Clock)	1	1	x	Note 2	0	No confirmati on is required	1

Note 1 After the reset is released, the control register (CMC) can only be written once through the 8-bit memory operation instruction.

2. The oscillation settling time of the Oscillation Settling Time Selection Register (OSTS) must be set as follows:

- The oscillation settling time of the State Register (OSTC) of the Expected Oscillation Settling Time Counter \leq the Oscillation Settling Time of the OSTC Register Settings

Note The clock must be set after the supply voltage reaches the set clock operatable voltage (refer to datasheet).

(3) After reset released(A), the CPU shifts to the subsystem clock run (D).

(The CPU runs on a high-speed internal oscillator clock (B)) immediately after the reset is released.)

(SFR registers are set in order).

Setting flag for the SFR register State transition	CMC register note				CSC register XTSTOP	Oscillati on stable	CKC register CSS
	EXCLKS	OSCSELS	AMPHS1	AMPHS0			
(A) →(B)→(D) (XT1 clock).	0	1	0/1	0/1	0	need	1
(A) →(B)→(D) (External Sub-Clock)	1	1	x	x	0	need	1

Note After reset released, the control register(CMC) can only be written once through the 8-bit memory operation instruction in clock run mode.

Note 1 x: Ignore

2. (A)~(I)in table 43 corresponding to(A)~(I) in figure 4-22.

Table 4-3 Example of CPU clock transfer and SFR register setting (2/5).

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

(SFR registers are set in order).

Setting flag for the SFR register State transition	CMC register ^{Note 1}			OSTS register	CSC register	OSTC register	CKC register
	EXCLK	OSCSEL	AMPH		MSTOP		
(B)→(C) (X1clock:1MHz≤f _x ≤10MHz)	0	1	0	Note 2	0	Confirmation is required	1
(B)→(C) (X1Clock:10MHz<f _x ≤20MHz)	0	1	1	Note 2	0	Confirmation is required	1
(B)→(C) (External Master Clock)	1	1	×	Note 2	0	No confirmation is required	1

Not required if it is already set.

Not required during high-speed system clock operation.

Note 1 After the reset is released, only one clock run mode control register (CMC) can be set. Not required if it is already set.

2. The oscillation settling time of the Oscillation Settling Time Selection Register (OSTS) must be set as follows:

- The oscillation settling time of the State Register (OSTC) of the Expected Oscillation Settling Time Counter

≤ the Oscillation Settling Time of the OSTS Register Settings

Note The clock must be set after the supply voltage reaches the set clock operatable voltage (refer to

datasheet).

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

(SFR registers are set in order).

Setting flag for the SFR register State transition	CMC register ^{note}			CSC register	Oscillation stable	CKC register
	EXCLKS	OSCSELS	AMPHS1, 0	XTSTOP		CSS
(B) →(D) (XT1 clock).	0	1	00: Low power oscillation 01: normal oscillation 10: Ultra-low power oscillation	0	need	1
(B) →(D) (External Sub-Clock)	1	1	×	0	need	1

Not required if it is already set.

Not required in subsystem clock operation.

Note After reset released, the control register(CMC) can only be written once through the 8-bit memory operation instruction in clock run mode. Not required if it is already set.

Note 1 ×: Ignore

2. (A)~(I) in table 43 corresponding to (A)~(I) in figure 4-22.

Table 4-3 Example of CPU clock transfer and SFR register setting (3/5).

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

(SFR registers are set in order). →

Setting flag for the SFR register State transition	CSC registers	Wait for the oscillation accuracy to stabilize	CKC registers
	HIOSTOP		MCM0
(C)→(B)	0	note	0

Not required during high-speed internal oscillator clock operation.

note: FRQSEL4=0時: 45μs~65μs

FRQSEL4=1時: 45μs~135μs

Note The oscillation accuracy of the high-speed internal oscillator clock is stable and waits for change due to temperature conditions and during deep sleep mode.

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

(SFR registers are set in order). →

Setting flag for the SFR register State transition	CSC registers	Wait for the oscillation accuracy to stabilize	CKC registers
	XTSTOP		CSS
(C) →(D)	0	required	1

Not required in subsystem clock operation.

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

(SFR registers are set in order). →

Setting flag for the SFR register State transition	CSC registers	Wait for the oscillation accuracy to stabilize	CKC registers
	HIOSTOP		CSS
(D) →(B)	0	note	0

Not required during high-speed internal oscillator clock operation.

note: FRQSEL4=0時: 45μs~65μs

FRQSEL4=1時: 45μs~135μs

Note 1 Table 4-3A) ~ (I) of Table 4-3 Figure 4-22A) ~ (I) in figure 4-22.

2. The oscillation accuracy of the high-speed internal oscillator clock is stable and waiting due to temperature conditions and deep sleep mode periods.

Table 4-3 Example of CPU clock transfer and SFR register setting (4/5).

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

(SFR registers are set in order).

Setting flag for the SFR register State transition	OSTS register	CSC registers	OSTC registers	CKC registers
		MSTOP		CSS
(D) → (C) (X1clock: $1\text{MHz} \leq f_x \leq 10\text{MHz}$)	note	0	Confirmation is required	0
(D) → (C) (X1Clock: $10\text{MHz} < f_x \leq 20\text{MHz}$)	note	0	Confirmation is required	0
(D) → (C) (External Master Clock)	note	0	No confirmation is required	0

Not required during high-speed system clock operation.

Note The oscillation settling time of the Oscillation Settling Time Selection Register (OSTS) must be set as follows:

- The oscillation settling time of the State Register (OSTC) of the Expected Oscillation Settling Time Counter

≤ the Oscillation Settling Time of the OSTS Register Settings

Note The clock must be set after the supply voltage reaches the set clock operatable voltage (see data sheet).

- (10) • The CPU is transferred to sleep mode (E) while the high-speed internal oscillator clock is running(B).
- The CPU is transferred to sleep mode (F) during high-speed system clock operation (C).
 - The CPU is transferred to sleep mode (G) while the subsystem clock is running (D).

State transition	Content of configuration
(B) → (E) (C) → (F) (D) → (G)	Execute WFI instructions.

Note Table 4-3of (A) ~ (I) corresponds to Figure 4-22of (A) ~ (I).

Table 4-3: CPU Clock Transfer and SFR Register Setting Example (5/5).

- (11) • The CPU is transferred to deep sleep mode (H) during high-speed internal oscillator clock operation (B).
• The CPU is transferred to deep sleep mode (I) during high-speed system clock operation (C).

(Set Order) →

State transition		Content of configuration		
(B) →(M)		Stop it	—	The SCR register bit2 (SLEEPDEEP) is set to 1 and the WFI instruction is executed.
(C) →(E)	X1 oscillates	Peripheral features that do not run in deep sleep mode.	Set the OSTS register.	
	External clock		—	

Note Table 4-3of (A) ~ (I) corresponds to Figure 4-22of (A) ~ (I)。

4.6.5 Conditions before CPU clock transfer and processing after transfer

The conditions before the CPU clock shift and the processing after the transfer are as follows.

Table 4-4 the transfer of CPU clocks (1/2).

CPU clock		Conditions before transfer	Treatment after transfer
Before transfer	After transfer		
High-speed internal oscillator clock	X1 clock	The X1 oscillates stably. • OSCSEL=1, EXCLK=0, MSTOP=0 • After a stable time of oscillation	If the oscillation of the high-speed internal oscillator is stopped (HIOSTOP=1), the operating current can be reduced.
	External master system clock	Set the external clock of the EXCLK pin input to active. • OSCSEL=1, EXCLK=1, MSTOP=0	
	XT1 clock	The XT1 oscillates stably. • OSCSELS=1, EXCLKS=0, XTSTOP=0 • After a stable time of oscillation	
	External subsystem clock	Set the external clock of the EXCLKS pin input to active. • OSCSELS=1, EXCLKS=1, XTSTOP=0	
X1 clock	High-speed internal oscillator clock	Allows high-speed internal oscillator oscillation. • HIOSTOP=0 • After a stable time of oscillation	Can stop oscillation of X1 (MSTOP=1).
	External master system clock	Cannot be transferred.	—
	XT1 clock	The XT1 oscillates stably. • OSCSELS=1, EXCLKS=0, XTSTOP=0 • After a stable time of oscillation	Can stop oscillation of X1 (MSTOP=1).
	External subsystem clock	Set the external clock of the EXCLKS pin input to active. • OSCSELS=1, EXCLKS=1, XTSTOP=0	Can stop oscillation of X1 (MSTOP=1).
External master system clock	High-speed internal oscillator clock	Allows high-speed internal oscillator oscillation. • HIOSTOP=0 • After a stable time of oscillation	The input of the external master system clock can be invalidated (MSTOP=1)。
	X1 clock	Cannot be transferred.	—
	XT1 clock	The XT1 oscillates stably. • OSCSELS=1, EXCLKS=0, XTSTOP=0 • After a stable time of oscillation	The input of the external master system clock can be invalidated (MSTOP=1)。
	External subsystem clock	Set the external clock of the EXCLKS pin input to active. • OSCSELS=1, EXCLKS=1, XTSTOP=0	The input of the external master system clock can be invalidated (MSTOP=1)。

Table 4-4 on the transfer of CPU clocks (2/2).

CPU clock		Conditions before transfer	Treatment after transfer
Before transfer	After transfer		
XT1 clock	High-speed internal oscillator clock	<p>The high-speed internal oscillator is oscillating and selects high-speed internal</p> <p>The oscillator clock acts as the main system clock.</p> <ul style="list-style-type: none"> • HIOSTOP=0, MCS=0 	Can stop the oscillation of XT1 (XTSTOP=1).
	X1 clock	<p>The X1 oscillates stably and selects the high-speed system clock as the main system Unified clock.</p> <ul style="list-style-type: none"> • OSCSEL=1, EXCLK=0, MSTOP=0 • After a stable time of oscillation • MCS=1 	
	External master system clock	<p>Set the external clock of the EXCLK pin input to be active and select the high-speed system clock as the main system clock.</p> <ul style="list-style-type: none"> • OSCSEL=1, EXCLK=1, MSTOP=0 • MCS=1 	
	External subsystem clock	Cannot be transferred.	—
External subsystem clock	High-speed internal oscillator clock	<p>The high-speed internal oscillator is oscillating and selects high-speed internal</p> <p>The oscillator clock acts as the main system clock.</p> <ul style="list-style-type: none"> • HIOSTOP=0, MCS=0 	The input of the external subsystem clock can be invalidated (XTSTOP=1)。
	X1 clock	<p>The X1 oscillates stably and selects the high-speed system clock as the main system Unified clock.</p> <ul style="list-style-type: none"> • OSCSEL=1, EXCLK=0, MSTOP=0 • After a stable time of oscillation • MCS=1 	
	External master system clock	<p>Set the external clock of the EXCLK pin input to be active and select the high-speed system clock as the main system clock.</p> <ul style="list-style-type: none"> • OSCSEL=1, EXCLK=1, MSTOP=0 • MCS=1 	
	XT1 clock	Cannot be transferred.	—

4.6.6 Time required to switch between the CPU clock and the master system clock

The CPU clock can be switched by setting bit6 and bit4 (CSS, MCM0) of the system clock control register (CKC)(Master system clock Subsystem Clock) and switching of the main system clock (High Speed Internal Oscillator Clock High Speed System Clock).

Instead of making the actual switchover immediately after rewriting the CKC registers, the CKC registers are changed and several clocks continue to run on the clock before the switch (see Table Table 4-5to Table Table 4-7).

The bit7 (CLS) of the CKC register can be used to determine whether the CPU is running on the main or secondary system clock. The bit5 (MCS) of the CKC register can be used to determine whether the master system clock is running on a high-speed system clock or a high-speed internal oscillator clock.

If you switch the CPU clock, you also switch the peripheral hardware clock.

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

Clock A	Switch directions	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} f_{MX}

The setting value before switching		The setting value after the switch	
MCM0		MCM0	
		0 ($f_{MAIN}=f_{IH}$)	1 ($f_{MAIN}=f_{MX}$)
0 ($f_{MAIN}=f_{IH}$)	$f_{MX} \geq f_{IH}$		2 clocks
	$f_{MX} < f_{IH}$		$2 f_{IH}/f_{MX}$ clock
1 ($f_{MAIN}=f_{MX}$)	$f_{MX} \geq f_{IH}$	$2 \uparrow f_{MX}/f_{IH}$ clock	
	$f_{MX} < f_{IH}$	2 clocks	

Table 4-7 f_{MAIN} f_{SUB} Maximum number of clocks required

The setting value before switching		The setting value after the switch	
CSS		CSS	
		0 ($f_{CLK}=f_{MAIN}$)	1 ($f_{CLK}=f_{SUB}$)
0 ($f_{CLK}=f_{MAIN}$)			$1+2 f_{MAIN}/f_{SUB}$ clock
1 ($f_{CLK}=f_{SUB}$)		3 clocks	

Note 1 Table 4-6Table 4-747 is the number of CPU clocks before switching.

2. The number of Table 4-6Tables Table 4-7is the number of clocks rounded to the fractional part.

Example: Switch the master system clock from a high-speed system clock to a high-speed internal oscillator clock ($f_{IH}=8\text{MHz}$, $f_{MX}=10\text{MHz}$ oscillation).

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

4.6.7 Conditions before clock oscillation stops

The register flag settings used to stop clock oscillation (invalid external clock input) and the conditions before stopping are as follows.

Table 4-8 Conditions and flag settings before clock oscillation stops

clock	Condition before clock stop (external clock input is invalid)	Flag setting for SFR registers
High-speed internal oscillator clock	MCS=1 or CLS=1 (The CPU runs on a clock other than the high-speed internal oscillator clock).	HIOSTOP=1
X1 clock External master system clock	MCS=0 or CLS=1 (The CPU runs on a clock other than the high-speed system clock.)	MSTOP=1
XT1 clock External subsystem clock	CLS=0 (The CPU runs on a clock other than the sub-system clock).	XTSTOP=1

4.7 High-speed internal vibration correction function

4.7.1 High-speed internal vibration self-adjustment function

This function measures the frequency of the high-speed internal oscillator using the sub-system clock f_{SUB} (32.768KHz) as the reference and corrects the frequency accuracy of the high-speed internal oscillator f_{HOCO} in real time.

Table 4-9 is the operating specification of the high-speed internal vibration frequency correction function, and Fig. 4-23 is the action block diagram of the high-speed internal vibration frequency correction function.

Table 4-9 Operating specifications for the high-speed internal vibration frequency correction function

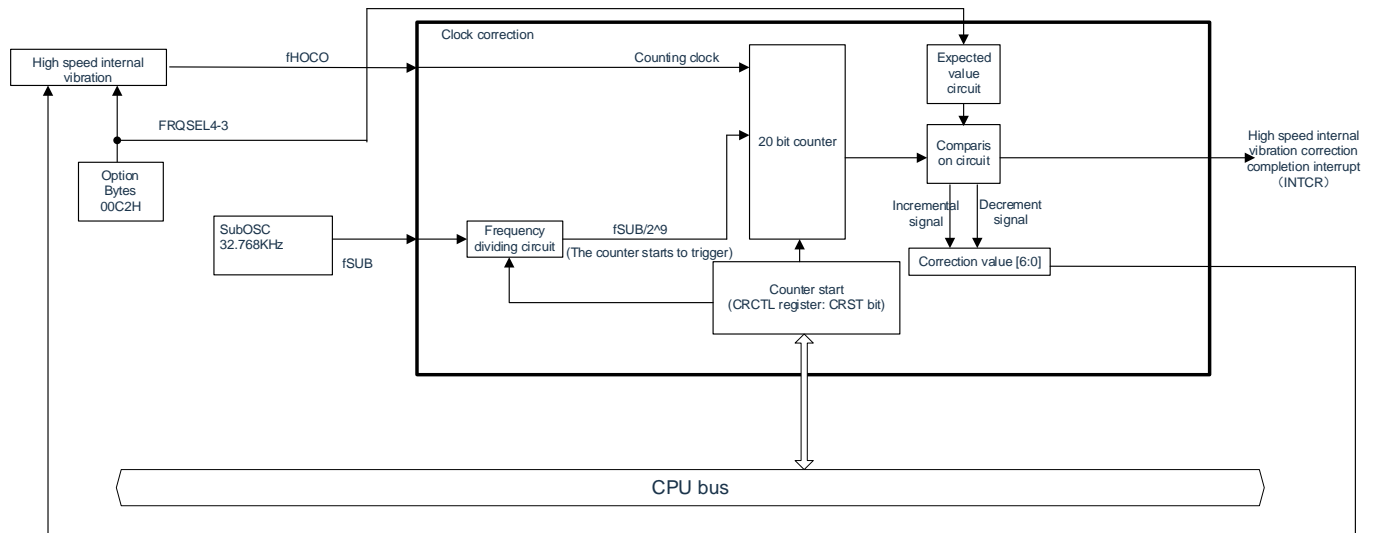
project	content
Reference clock	• $f_{SUB}/2^9$ (subsystem clock 32.768KHz).
target clock of Correction	• f_{HOCO} (fast inner swing)
Action mode	• Continuous action mode A mode of continuous high-speed internal vibration frequency correction • Interval action mode A mode that uses a timer clock end, etc., to perform high-speed internal vibration frequency correction at intervals
Clock accuracy adjustment function	• Correction time: Correction period (31.2ms) × (number of corrections - 0.5) ^{Note}
interrupt	• Interrupt when high-speed internal frequency correction is complete (when interrupt permission is open)

Note: Correction time: Varies depending on the number of corrections.

Correction Period: The total time of the frequency determination phase and the frequency correction phase.

Number of Corrections: The number of corrections by which the frequency is bound to the expected value range.

Fig. 4-23 Action block diagram of high speed internal vibration frequency value correction function



4.7.2 Register description

Table 4-10 shows a list of registers used for the high-speed internal oscillation frequency correction function.

Table 4-10 High-speed internal oscillation frequency correction function registers at a glance

project	structure
Control registers	High Speed Internal Vibration Frequency Correction Control Register (HOCOFC)

4.7.2.1 High Speed Internal Vibration Frequency Correction Control Register (HOCOFC)

Control register for high-speed internal oscillation frequency correction.

The HOCOFC register is set by the 8-bit memory operation instruction.

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

Figure 4-24 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	FCIE	0	0	0	0	0	FCST

FCMD ^{Note 1}	High-speed internal vibration frequency correction function action mode
0	Continuous action mode
1	Interval action mode

FCIE	Interrupt control with high-speed internal vibration frequency correction
0	There is no interruption after the high-speed internal vibration frequency correction is
1	An interrupt occurs after the high-speed internal oscillation frequency correction is complete

FCST ^{Note 2}	High-speed internal vibration frequency correction circuit action control/status
0	High-speed internal vibration frequency correction circuit action stop/stop
1	High-speed internal vibration frequency correction circuit action start/action
In continuous motion mode, the software writes 0 to stop the action. In interval mode, the hardware clears the FCST bit after the correction is complete.	

Note 1. When the FCST bit is 1, rewriting the FCMD bit is prohibited.

2. When writing 1 to the FCST bit, first confirm that the value of the current FCST bit is 0 and then write 1 to it. Due to the priority of hardware clearance, when the FCST bit is written to 1 immediately after the interval operation is completed (when the high-speed internal vibration frequency correction completes the interrupt generation), the operation should be performed at least 1 cycle after the high-speed internal vibration frequency correction is completed after the interrupt is generated.

After writing 0 to the FCST bit (the high-speed internal oscillation frequency correction circuit action stops), fHOCO disables writing 1 to the FCST bit for 2 cycles (the high-speed internal vibration frequency correction circuit action begins).

Note: Bits 5 to 1 must be written 0

4.7.3 Description of the operation

4.7.3.1 Operation summary

The high-speed internal oscillation frequency correction function uses the sub-system clock (fSUB) as the reference to generate a correction period, measures the frequency of the high-speed internal oscillation, and corrects the frequency accuracy of the high-speed internal oscillation in real time. Clock adjusts the operation of the repeating frequency measurement phase and the frequency correction phase. The correction calculus is performed during the frequency measurement phase, and the correction values reflecting the results of the correction calculus are saved during the frequency correction phase.

Table 4-11 is the high-speed internal vibration input frequency and correction period, and Figure Fig. 4-25 is the timing diagram of the high-speed internal vibration frequency correction operation (detailed).

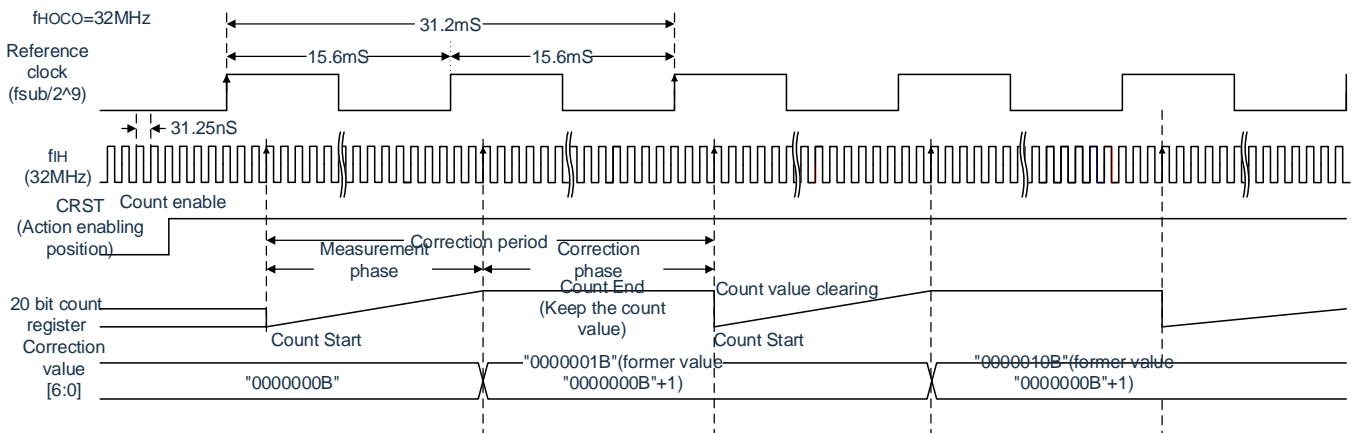
Table 4-11 High-speed internal oscillator input frequency and correction period

fHOCO (MHz)	FRQSEL4-FRQSEL3 ^{Note}	Correction Period (ms)
64	11	31.2 (Frequency measurement phase + frequency correction phase)
48	10	
32	01	
24	00	

Note: FRQSEL4-FRQSEL3 is bit4-bit3 for option bytes0 0C2H

During the frequency measurement phase of the correction period, the frequency of the high-speed internal oscillation is corrected according to the size of the count value and the expected value.

Fig. 4-25 High-speed internal oscillation frequency correction action timing diagram (detailed).



Note: The basic operation of the continuous action mode and the interval action mode are the same. The difference is whether the FCST bit is cleared by software or hardware. In addition, only the system reset can clearly correct the correction value.

(1) Continuous operation mode

In continuous operation mode, the high-speed internal oscillating clock frequency correction action is carried out all the time. The FCMD bit of the HOCOFC register is set to 0, which is a continuous operation mode.

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

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

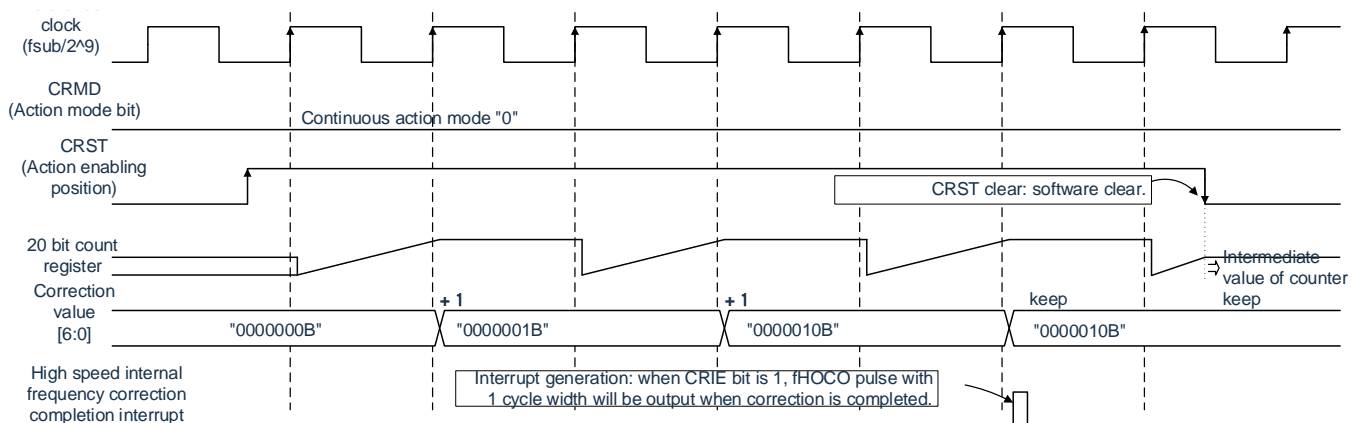
The count value is then compared to the expected value and the correction values are adjusted as described below. (Frequency Correction Phase)

- When the count value is larger than expected: The correction value is -1
- Count value is hours longer than expected: Correction value +1
- When the count value is within the expected value: the correction value is maintained (the high-speed internal oscillation clock frequency correction ends)

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

Figure 4-26 is a timing diagram of the continuous motion mode.

Figure 4-26 Sequential diagram of continuous motion mode



(2) Interval operation mode

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

The FCST bit of the HOCOFC register is set to 1 when the high-speed internal oscillator clock frequency correction action begins.

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

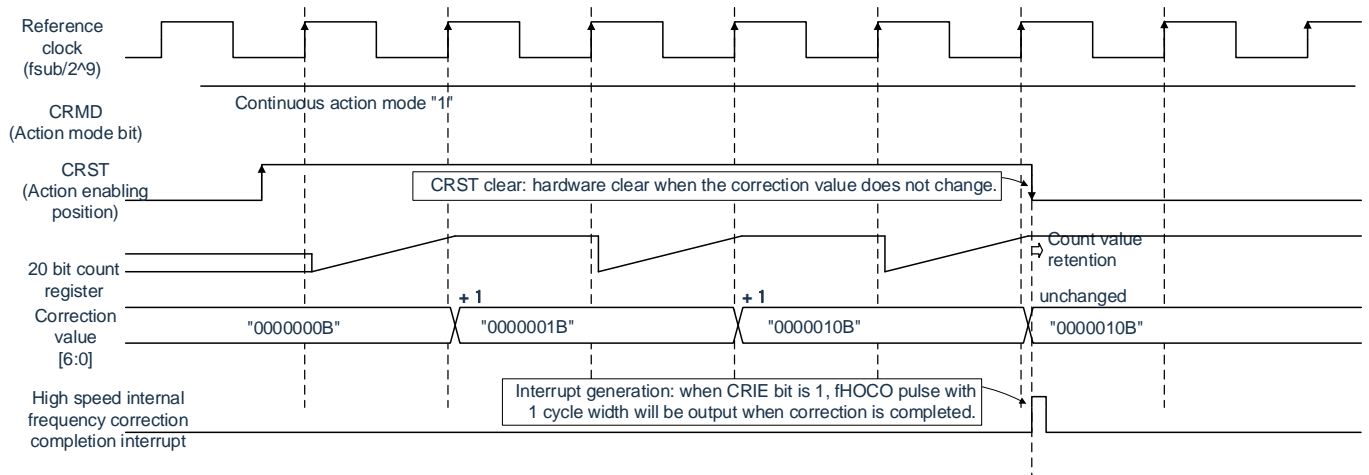
The count value is then compared to the expected value and the correction values are adjusted as described below. (Frequency Correction Phase)

- When the count value is larger than expected: The correction value is -1
- Count value is hours longer than expected: Correction value +1
- When the count value is within the expected value: the correction value is maintained (the high-speed internal oscillation clock frequency correction ends)

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

Figure 4-27 is a timing diagram of the continuous action mode.

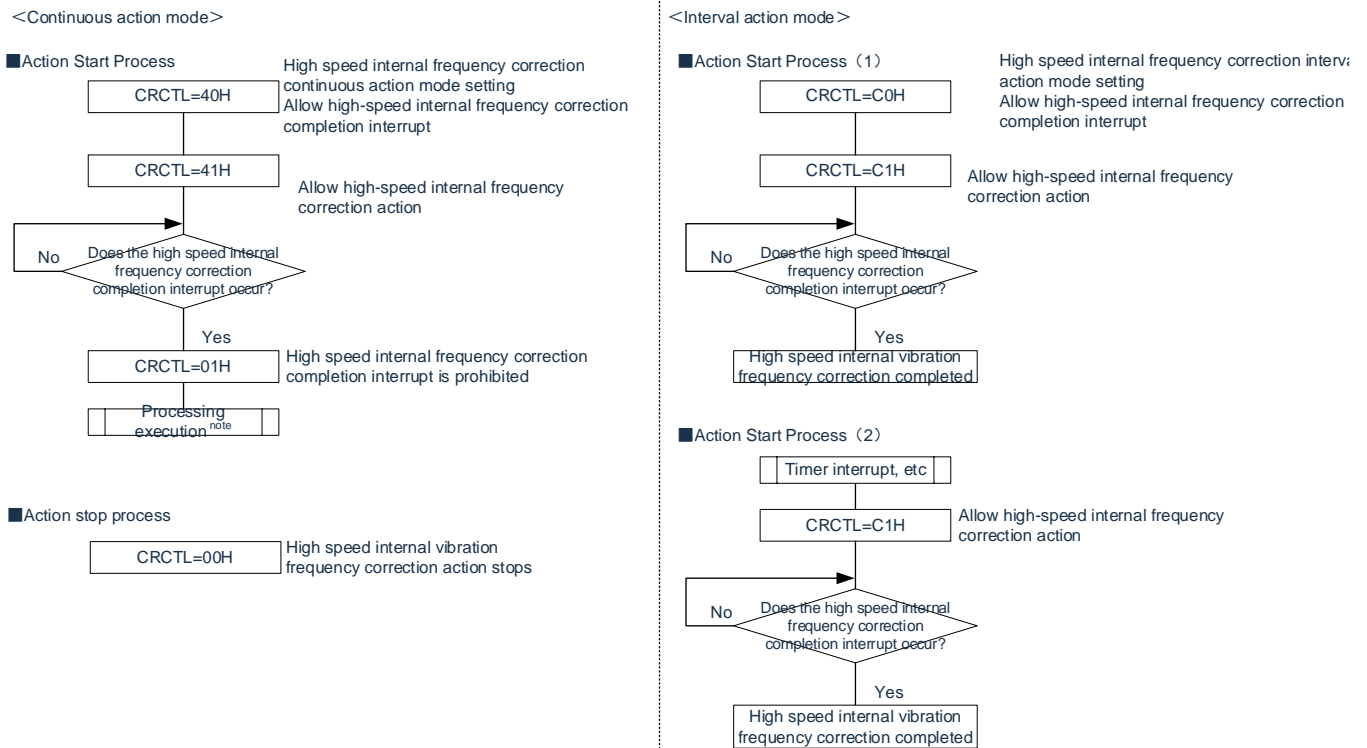
Figure 4-27 Interval action mode timing diagram



4.7.3.2 Operation setup process

The Operation mode start/stop process is shown in the following figure when the high-speed internal oscillating clock frequency correction function is used.

Figure 4-28 Operation mode setting process (example).



Note: Before stopping the high-speed internal oscillating clock frequency correction function, the high-speed internal oscillating clock frequency correction operation is performed repeatedly.

4.7.4 Precautions for use

4.7.4.1 SFR access

Regarding the control of the FCST bit in the interval mode, when writing 1 to the FCST bit, you must first confirm that the value of the current FCST bit is 0 and then write 1 to it. Due to the priority of hardware clearance, when the FCST bit is written to 1 immediately after the interval operation is completed (when the high-speed internal vibration frequency correction completes the interrupt generation), the operation should be performed at least 1 cycle after the high-speed internal vibration frequency correction is completed after the interrupt is generated.

4.7.4.2 Action during reset

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

4.8 Vibration stop detection circuit

4.8.1 Vibration stops the function of the detection circuit

Vibration stop detection function is to use the internal low-speed oscillation clock (fIL) to monitor the operation state of the main system clock (fmx) or the secondary system clock (fsx), in a period of time, when the action stop is detected, it is judged that the X1 vibration circuit or XT1 vibration circuit is abnormal, and the output vibration stop detection signal can be used as an interrupt signal or a reset signal.

The vibration stop detection circuit needs to be enabled by software settings after the reset is released.

Vibration stop detection circuit, through the software setting to stop the detection action. Alternatively, the vibration detection action is stopped because of a terminal reset or other internal reset. After the reset occurs, the software settings need to be set again to enable the vibration stop detection action.

The vibration stop detection circuit determines the vibration stop time (vibration stop judgment time) is set by the OSDCCMP11~OSDCCMP0 of the vibration stop detection control register (OSDC).

Oscillation stop determination time = Internal low-speed oscillation clock (fIL) period × ((OSDCCMP11~OSDCCMP0 setting value) + 1).

Take the internal low-speed oscillation clock (fIL) frequency of 15K as an example:

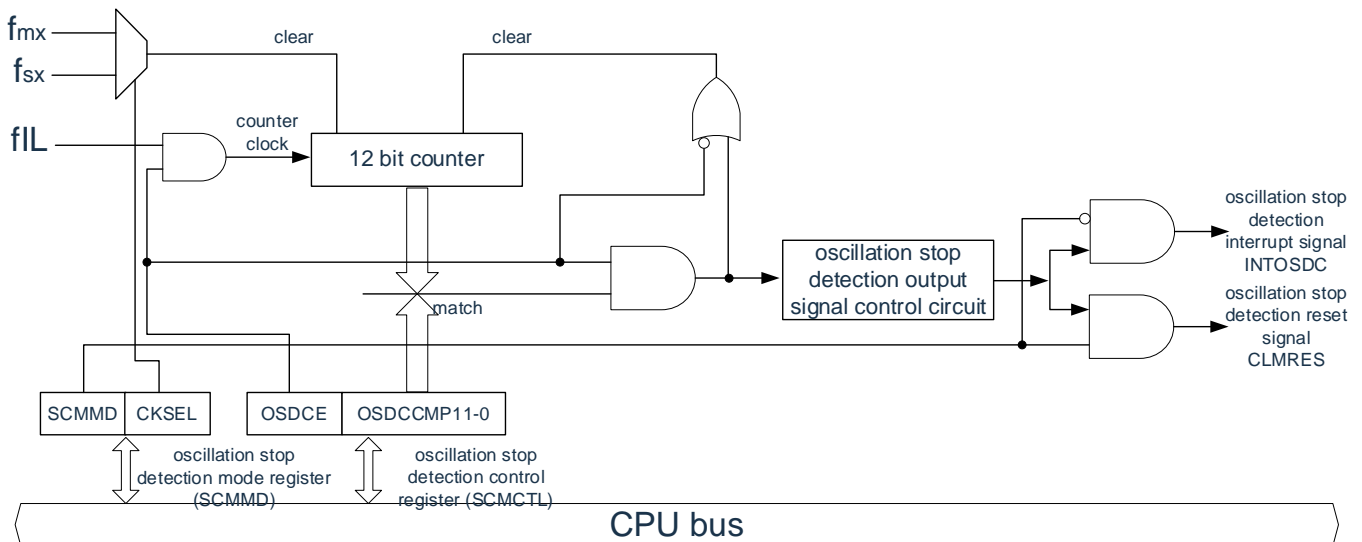
OSDCCMP11~OSDCCMP0=003H时: 232us(MIN.),267us(TYP.),314us(MAX.)

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

4.8.2 Composition of the vibration-stop detection circuit

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

Fig. 4-29 vibration stop detection circuit (example).



4.8.3 The register used by the oscillator stops detection circuit

4.8.3.1 Peripheral Enable Register 2 (PER2)

When using the vibration-stop detection circuit, the BIT4 (OSDCEN) of PER2 must be set to 1.

Registers are described in detail in "4.3.7Peripheral enable registers 0, 1, 2, 3 (PER0, PER1, PER2, P ER2 PER3) ".

4.8.3.2 Vibration Stop Detection Control Register (SCMCTL)

The Vibration Stop Detection Control Register (SCMCTL) is a register that controls the start of the action of the vibration stop detection circuit, the stop of the action, and the setting of the vibration stop determination time.

When the OSCDE bit is 0, the vibration stop detection circuit does not start the operation.

Use 16-bit operation instructions to manipulate the SCMCTL registers.

Figure 4-30Format of the Vibration Stop Detection Control Register (SCMCTL).

Address: 0x40022200 After reset: 0FFFF 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	Vibration stops detecting the action of the action
0	Vibration stop detection action stops
1	Vibration stop detection action begins

OSDCCMP11-0	Vibration stop determination time
000H ... 002H	Prohibit settings
003H ... FFFH	Set the vibration stop judgment time. Oscillation stop determination time = internal low-speed oscillation clock (fIL) period × ((OSDCCMP11~ OSDCCMP0 setting value) +1).

Note 1. When modifying the setting value of OSDCCMP11~OSDCCMP0, OSDCE must be set to 0.

2. The vibration detection circuit stops the vibration detection action by setting OSDCE=0 (vibration stop detection action stop) or generates terminal reset and other internal resets to stop the vibration detection action.

3.Bit14-12 must be set to 0.

4.8.3.3 Vibration Stop Detection Mode Register (SCMMD)

The Oscillation Stop Detection Mode Register (SCMMD) is a register that selects the object of vibration stop detection as the primary system clock (fmx) or the secondary system clock (fsx), and whether the action after the vibration stop is detected is to generate a reset or an interrupt.

Use 16-bit operation instructions to manipulate the SCMMD registers.

Figure 4-31 Format of the Vibration Stop Detection Mode Register (SCMMD).

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

symbol	15	14	13	12	11	10	9	8
SCMMD	KEY ^{Note}							
symbol	7	6	5	4	3	2	1	0
SCMMD	0	0	0	0	0	0	MDSEL	CKSEL
CKSEL	Vibration stops detecting objects							
0	Detects the vibration status of the main system clock (fmx).							
1	Detects the vibration status of the subsystem clock (fsx).							
MDSEL	Vibration stops the action after detection							
0	An interruption occurs after the vibration stop is detected							
1	Vibration stops detecting and produces a reset							

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

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

4.8.3.4 Vibration Stop Detection Status Register (SCMST)

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

Use 8-bit operating instructions to manipulate the SCMST registers.

Figure 4-32 Format of the Vibration Stop Detection Status Register (SCMST).

Address: 0x40022204 After reset: 000H R/W ^{Note}

symbol	7	6	5	4	3	2	1	0
SCMST	0	0	0	0	0	0	0	OSTDF
OSTDF	The state of vibration stop detection							
0	Vibration stop is not detected							
1	Vibration stops being detected							

Note: After the vibration stops having a checkout, place OSTDF at position 1, which can only be written to 0 through the write register.

4.8.4 The operation of the vibration stop detection circuit

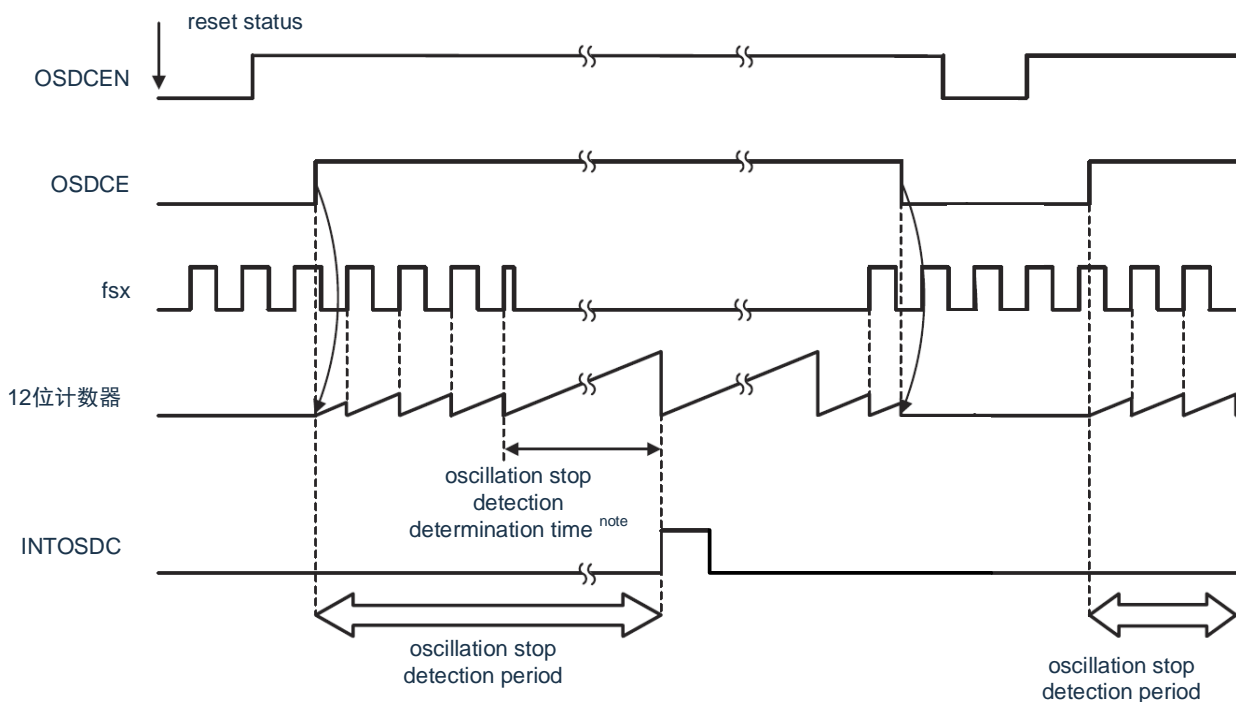
4.8.4.1 The operation method of the vibration stop detection circuit

1. After the external reset is released, the master system clock (fmx)/secondary system clock (fsx) begins to vibrate.
2. Writes the peripheral enable register (PER2) to enable the oscillation stop detection circuit ^{note}.
3. Write the Oscillation Stop Detection Mode Register (SCMMD), select the vibration state that detects the main system clock (fmx) or the subsystem clock (fsx), and select whether the action after the vibration stop is detected, whether to generate a reset or an interrupt.
4. Write the OSDCE bit of the vibration stop detection control register (SCMCTL), and the vibration stop detection circuit begins to operate.
5. During the operation of the vibration stop detection circuit, the main system clock (fmx)/sub-system clock (fsx) has been stopping the vibration during the vibration stop judgment time, and the output vibration stop detects the signal, generating a reset or interrupt.
6. When the CPU clock is the main system clock (fmx) or the CPU clock is a PLL clock with the main system clock (fmx) as the PLL input, and the main system clock (fmx) oscillation stop detection, the CPU clock switches to the partition 8 (fHOCO/8) of the internal high-speed oscillation clock, and when the CPU clock is a sub-system clock (fsx) and there is a subsystem clock (fsx) vibration stop detection, The CPU clock switches to the internal low-speed oscillation clock (fIL). Clearing OSTDF by software writing the SCMST register to 0 will cut the CPU clock back to the original clock.

Note: Because the BIT4 (OSDCEN) of PER2 may be detected by mistake when writing 1, when using vibration to stop detecting interrupts, you must clear the interrupt flag bit after writing PER2, and then turn on interrupt enable. For more information about interrupt registers corresponding to oscillation stop detection interrupts, refer to Chapter 25 Interrupt Functions.

Figure 4-33 Timing of the vibration-stop detection circuit

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



Note: Vibration stop determination time = internal low-speed oscillation clock (fIL) period × ((OSDCCMP11~OSDCCMP0 setting value) + 1).

4.8.5 Precautions for vibration stop detection function

Vibration-stop detection circuit, used with a watchdog timer.

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

When bit0 (WDSTBYON) of the option byte (00C0H), bit4 (WDTON) is 1, and bit4 (WUTMMCK0) of the OSMC register is 0;

When bit4 (WUTMMCK0) of the OSMC register is 1;

Chapter 5 Hardware divider

Hardware dividers are specific-built hardware to support high-performance computing. The hardware divider is a 32-bit signed integer divider that outputs a 32-bit signed quotient and remainder result.

5.1 Features

- 32-bit signed (complement of 2) integer division calculations
- 32 bits signed dividend and 32 bits signed divisors
- 32-bit signed quotient and 32-bit signed remainder output
- Writing divisor register automatically triggers the division calculation
- Divided by 0 warning flag
- Indicates the BUSY flag in the operation
- There is an interrupt request for calculation completion
- Each calculation takes 4 or 8 CPU clock cycles
 - It takes 4 CPU clock cycles at double speed
 - It takes 8 CPU clock cycles in the non-double speed state

5.2 Feature description

When using a hardware divider, you need to set the DIVIDEND register (DIVIDEND) first and then the DIVISOR, because a write to the divisor register automatically triggers the division calculation. By querying the STATUS BIT of the BUZZ or by using an interrupt at the end of the calculation, you can know when the settlement ended. The calculation results can be read through the quotient (QUOTIENT) and remainder (REMAINDER) registers.

Note: Do not write dividend or divisor registers during calculation, and do not read quotient or remainder registers, otherwise the result is unpredictable.

5.3 Registers for the hardware divider

The registers for the hardware divider are as follows:

Register base address: DIV_BASE = 4008_0000H;

Register name	Register description	R/W	Reset value	Register address
DIVIDEND	The divisor register	R/W	0000_0000H	DIV_BASE+00H
DIVISOR	Dividend register	R/W	0000_0000H	DIV_BASE+04H
QUOTIENT	Quotient register	R	0000_0000H	DIV_BASE+08H
REMAINDER	Remainder register	R	0000_0000H	DIV_BASE+0CH
STATUS	Status register	R	0000_0000H	DIV_BASE+10H

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

5.3.1 DIVIDEND

A Dividend register is a register that holds the Dividend, whose value participates in division operations as a 32-bit signed integer.

31	30	29	28	27	26	25	24
DIVIDEND[31:24]							
23	22	21	20	19	18	17	16
DIVIDEND[23:16]							
15	14	13	12	11	10	9	8
DIVIDEND[15:8]							
7	6	5	4	3	2	1	0
DIVIDEND[7:0]							

5.3.2 Divider (DIVISAR)

A divisor register is a register that holds a divisor whose value participates in division as a 32-bit signed integer. A write to this register automatically triggers a division calculation.

31	30	29	28	27	26	25	24
DIVISOR[31:24]							
23	22	21	20	19	18	17	16
DIVISOR[23:16]							
15	14	13	12	11	10	9	8
DIVISOR[15:8]							
7	6	5	4	3	2	1	0
DIVISOR[7:0]							

5.3.3 Quotient

After the division calculation is completed, the register holds the quotient of the division calculation result, whose value is a 32-bit signed integer.

31	30	29	28	27	26	25	24
QUOTIENT[31:24]							
23	22	21	20	19	18	17	16
QUOTIENT[23:16]							
15	14	13	12	11	10	9	8
QUOTIENT[15:8]							
7	6	5	4	3	2	1	0
QUOTIENT[7:0]							

5.3.4 REMAINDER

After the division calculation is complete, the register holds the remainder of the division calculation, the value of which is a 32-bit signed integer.

31	30	29	28	27	26	25	24
REMAINDER[31:24]							
23	22	21	20	19	18	17	16
REMAINDER[23:16]							
15	14	13	12	11	10	9	8
REMAINDER[15:8]							
7	6	5	4	3	2	1	0
REMAINDER[7:0]							

5.3.5 Status register (STATUS)

The status of the hardware divider can be queried through the status register, including the divide-zero flag and the BUSY flag.

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						DIVBYZE RO	BUSY
7	6	5	4	3	2	1	0
reserved							

DIVBYZERO	Used to indicate the case of division by zero, updated each time the divisor register is written.
0	The divisor is not 0.
1	The divisor is 0

BUSY	Used to indicate the status of the division operation.
0	Division is complete
1	The divisor operation is in progress

Chapter 6 Universal timer unit Timer4/8

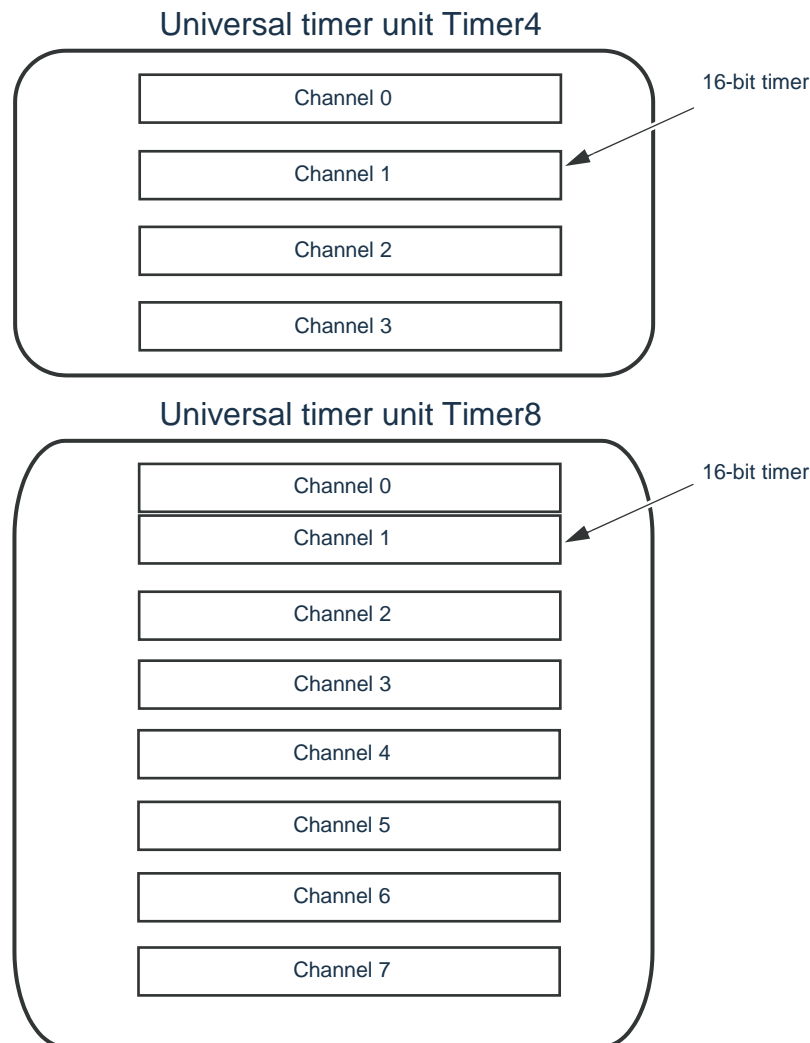
This product is equipped with two universal timer units, including Timer4, containing 4 channels. Timer8, with 8 channels. The number of channels for a universal timer unit varies by product.

Note:

1. The label "m" later in this chapter represents the unit number, and this product is equipped with two general-purpose timers Timer4 and Timer8, so $m=0,1$.
2. The designator "n" later in this chapter represents the channel number (in this chapter when $m=0$: $n=0\sim3/m=1$: $n=0\sim7$), and the presence or absence of timer input/output pins for each channel varies depending on the product. For details, please refer to "Chapter 2 Pin Functions".
3. The following in this chapter are primarily described for 100-pin products.

The universal timer unit Timer4 has four 16-bit timers. The universal timer unit Timer8 has eight 16-bit timers.

Each 16-bit timer is called a "channel" and can be used as a stand-alone timer or as a combination of multiple channels for advanced timer functions.



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

Stand-alone channel operation function	Multi-channel linkage operation function
<ul style="list-style-type: none"> Interval timer (→reference 6.8.1). Square wave output (→ reference 6.8.1). External event counter (→reference 6.8.2). Frequency divider (→ with reference to 6.8.3). Measurement of input pulse interval (→ refer to 6.8.4). Measurement of the high and low level width of the input Delay counter (→reference 6.8.6). 	<ul style="list-style-type: none"> Single-trigger pulse output (→reference 6.9.1). PWM output (→ refer to 6.9.2). Multiple PWM outputs (→ refer to 6.9.3).

The Channel 1 and Channel 3 16-bit timers of Unit 0 can be used as two 8-bit timers (high and low). Channels 1 and 3 can be used as 8-bit timers as follows:

- Interval timers (high 8-bit and low 8-bit timers) /square wave output (limited to low 8-bit timers only).
- External event counter (low 8-bit timer only).
- Delay counter (low 8-bit timer only).

LIN-bus communication is achieved through the coordination of channel 3 of unit 0 and UART0 of the universal serial communication unit.

6.1 Functions of the universal timer unit

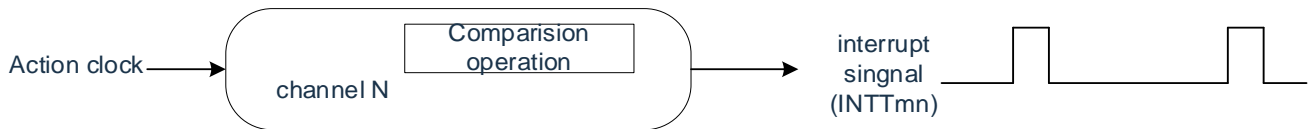
The universal timer unit has the following functions:

6.1.1 Stand-alone channel operation function

The stand-alone channel operation function is a function that can use any channel independently without being affected by the operating modes of other channels.

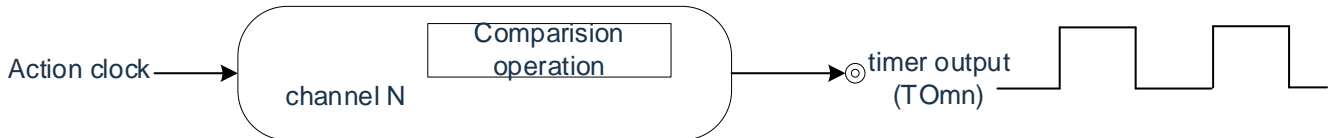
(1) Interval timer

It can be used as a reference timer that generates interrupts (INTTMn) at regular intervals.



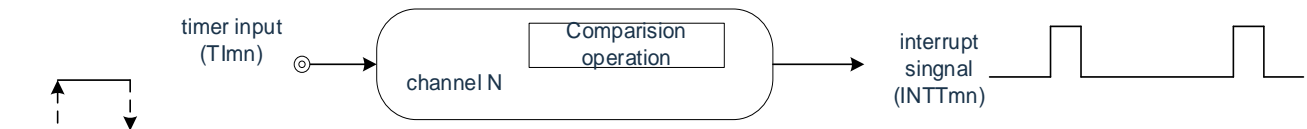
(2) Square wave output

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



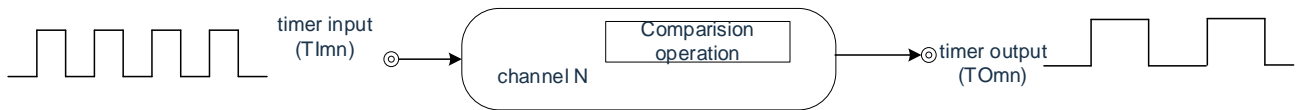
(3) External event counters

The effective edges of the input signal of the timer input pin (TIMn) are counted and, if a specified number of times are reached, can be used as an event counter that generates an interrupt.



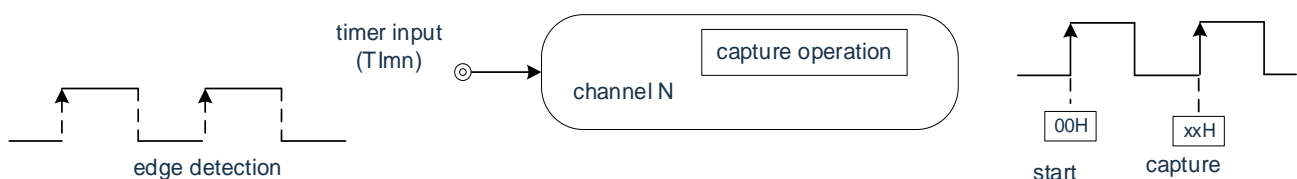
(4) Crossover function (limited to channel 0 of unit 0).

Divides the input clock of the timer input pin (TI00) and outputs from the output pin (TO00).



(5) Input the measurement of the pulse interval

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



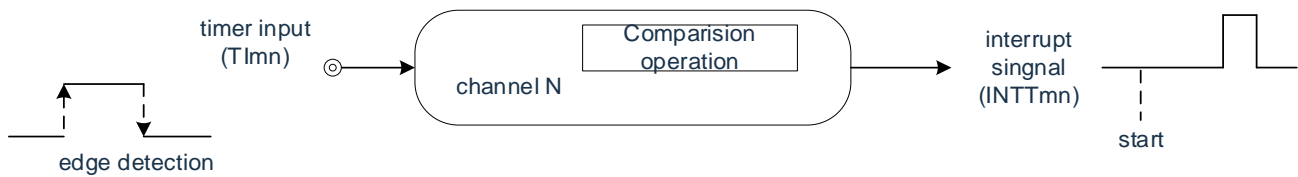
(6) Measurement of the high and low level width of the input signal

The input signal starts counting on one edge of the timer input pin (TImn) and captures the count value on the other edge, measuring the high and low level width of the input signal.



(7) Latency counters

The effective edges of the input signal at the timer input pin (TImn) start counting and generate an interrupt after an arbitrary delay period.



Note: 1.m: unit number (m=0,1) n: channel number (when m=0: n=0~3, m=1: n=0~7)

2. The presence or absence of timer input/output pins for each channel varies depending on the product. For details, please refer to "Timer Input/Output Pins for Each Product in Table 6-2".

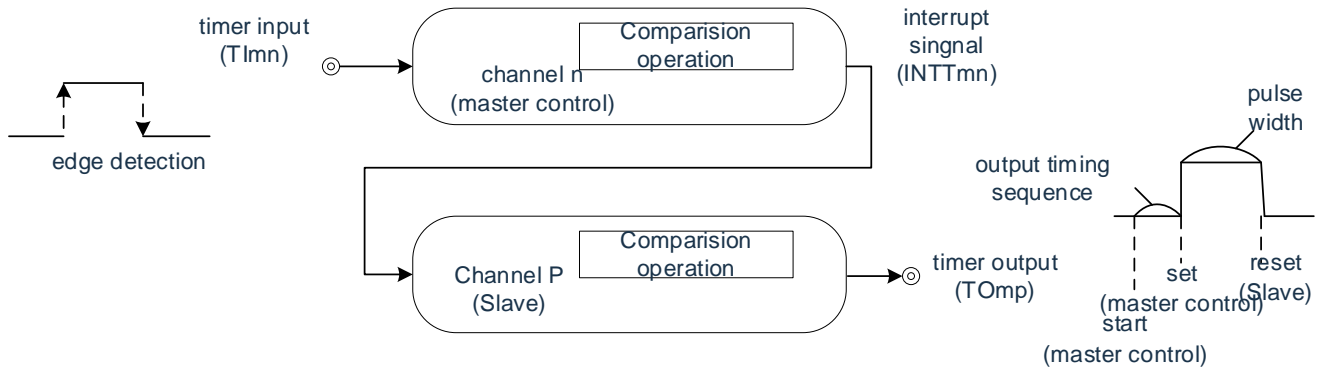
6.1.2 Multi-channel linkage operation function

The multi-channel linkage operation function is a function that combines the master channel (the reference timer of the main control period) and the slave channel (the timer that follows the operation of the master channel).

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

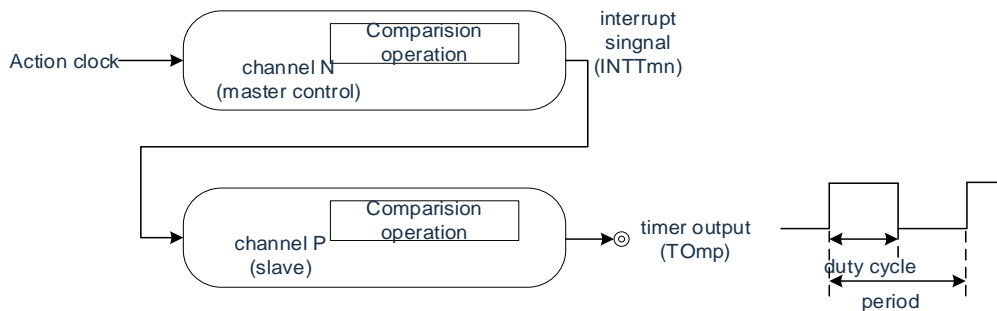
(1) Single trigger pulse output

Pairing the two channels generates a single-trigger pulse that arbitrarily sets the output timing and pulse width.



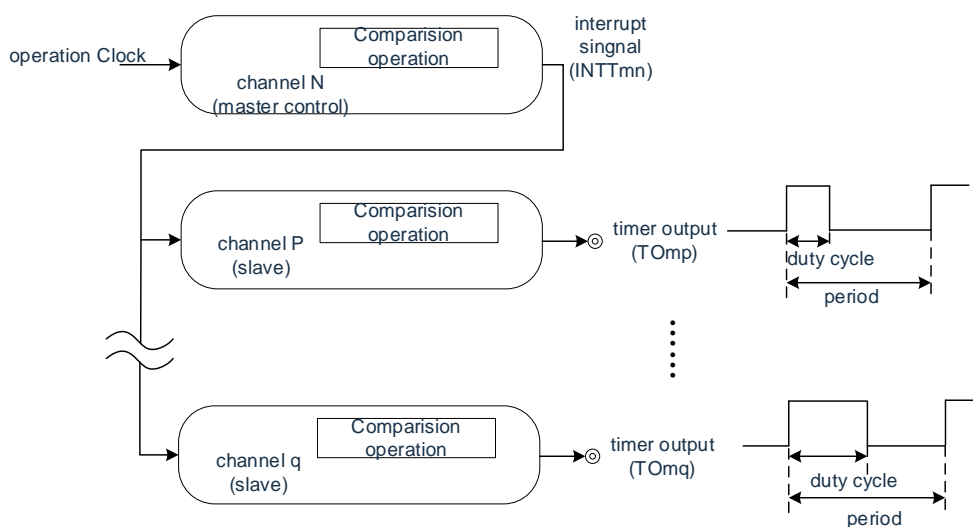
(2) PWM (Pulse Width Modulation) output

Pairing the two channels generates pulses with arbitrary set periods and duty cycles.



(3) Multiple PWM (Pulse Width Modulation) outputs

It can generate up to three PWM signals with arbitrary duty cycles at fixed periods by extending the PWM function and using 1 master channel and multiple slave channels.



Note: For details of the rules for the multi-channel linkage operation function, please refer to the Multi-channel Linkage Operation Function ".

"6.4.1 Basic Rules of

Note: m: unit number (m=0,1) n: channel number (when m=0: n=0~3, m=1: n=0~7).
p, q: Slave channel number (m=0Time:n<p<q≤3) (m=1Time:n<p<q≤7)

6.1.3 8-bit timer operation function (channel 1 and channel 3 of unit 0 only).

The 8-bit timer operation function is the function of using a 16-bit timer channel as two 8-bit timer channels. Only Channel 1 and Channel 3 of Unit 0 can be used.

Note: When running the feature with an 8-bit timer, there are several rules.

For details, please refer to "Basic Rules for the Operation of the 6.4.2 8-Bit Timer Function (Channel 1 and Channel 3 Only)".

6.1.4 LIN-bus support function (limited to channel 3 of unit 0).

Check the received signal in the LIN-bus communication for the LIN-bus communication table by means of a universal timer unit.

(1) Detection of wake-up signals

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

(2) Detection of interval segments

After the wake-up signal is detected, the low level width is measured by starting the falling edge of the input signal from 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 space segment.

(3) Measurement pulse width of Synchronization segment

After the interval segment is detected, measure the low and high width of the input signal of the UART0 serial data input pin (RxD0). Calculates the baud rate based on the bit spacing of the synchronization segment measured in this way.

Remark: Regarding the operating settings of the supported functions of LIN-bus. Please refer to "6.3.13 Input switching control registers (ISC)" and "6.8.5 As the input signal high and low level operation of the width measurement".

6.2 Structure of a universal timer unit

The universal timer unit consists of the following hardware.

Table 6-1 Structure of the Universal Timer Unit

Item	structure
counter	Timer count register mn (TCRmn).
register	Timer data register mn (TDRmn).
The input of the timer	TI00~TI03 ^{Note 1} , RxD0 pin (for LIN-bus).
The output of the timer	TO00~TO03 ^{Note 1} , output control circuit
Control registers	< unitSettings DepartmentRegister > <ul style="list-style-type: none"> • Peripheral enable register 0 (PER0). • Timer clock selection register m (TPSm). • Timer channel enable status register m(TEm). • Timer channel start register m(TSm). • Timer channel stop register m(TTm). • Timer input selection register 0 (TIS0). • Timer output enable register m (TOEm). • Timer output register m (TOM). • Timer output level register m (TOLm). • Timer output mode register m (TOMm).
	< register > for each channel <ul style="list-style-type: none"> • Timer mode register mn (TMRmn). • Timer status register mn (TSRmn). • Input Switching Control Register (ISC). • Noise filter enable registers 1 and 2 (NFEN1, NFEN2). • Port Mode Control Register (PMCxx) ^{Note 2} • Port Mode Register (PMxx) ^{Note 2} • Port register (Pxx) ^{Note 2}

Note: 1 The availability of timer input/output pins for each channel varies by product. For details, please refer to "Timer Input/Output Pins for Each Product in Table 6-2".

2. The set port mode control register (PMCxx), port mode register (PMxx) and port register (Pxx) vary depending on the product. For details, please refer to "Chapter 2 Pin Functions".

Note: m: unit number (m=0,1) n: channel number (when m=0: n=0~3, m=1: n=0~7).

The availability of timer input/output pins for each channel of the universal timer unit varies by product.

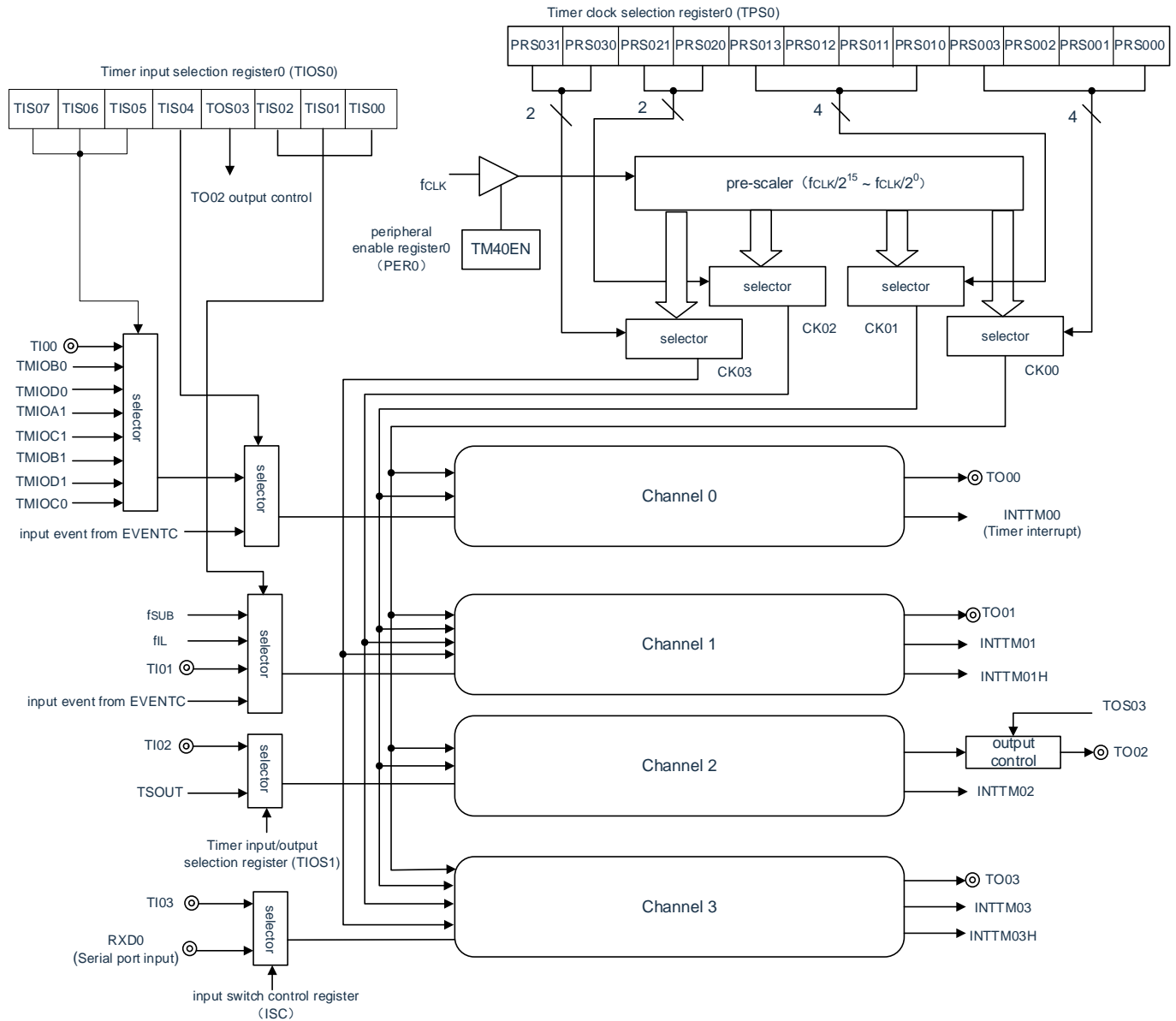
Table 6-2 Timer Input/Output Pins for Each Product

Channel of the timer array unit		The presence or absence of input/output pins for each product	
		64 pins	48 pins
Unit 0	Channel 0	TI00/TO00	TI00/TO00
	Channel 1	TI01/TO01	TI01/TO01
	Channel 2	TI02/TO02	TI02/TO02
	Channel 3	TI03/TO03	TI03/TO03
Unit 1	Channel 0	TI10/TO10	TI10/TO10
	Channel 1	TI11/TO11	TI11/TO11
	Channel 2	TI12/TO12	TI12/TO12
	Channel 3	TI13/TO13	TI13/TO13
Unit 0	Channel 4	TI14/TO14	TI14/TO14
	Channel 5	TI15/TO15	TI15/TO15
	Channel 6	TI16/TO16	TI16/TO16
	Channel 7	TI17/TO17	TI17/TO17

Remarks: 1. When the input of the timer and the output of the timer are multiplexed by the same pin, it can only be used as the input of the timer or the output of the timer.

A block diagram of the universal timer unit is shown in Figure 6-1.

Figure 6-1 Overall block diagram of universal timer unit 0



Note: f_{SUB} : Subsystem clock frequency
 f_{IL} : Low-speed internal oscillator clock frequency

6.2.1 List of general-purpose timer unit registers

Unit 0 (Timer4) register base address: 0x40041C00

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

Unit 1 (Timer8) register base address: 0x40042000

Offset address	Register name	Read-write properties	Bit width	Reset value
0x180	TCR10	R	16	FFFFH
0x182	TCR11	R	16	FFFFH
0x184	TCR12	R	16	FFFFH
0x186	TCR13	R	16	FFFFH
0x188	TCR14	R	16	FFFFH
0x18A	TCR15	R	16	FFFFH
0x18C	TCR16	R	16	FFFFH
0x18E	TCR17	R	16	FFFFH
0x190	TMR10	R/W	16	0000H
0x192	TMR11	R/W	16	0000H
0x194	TMR12	R/W	16	0000H
0x196	TMR13	R/W	16	0000H
0x198	TMR14	R/W	16	0000H
0x19A	TMR15	R/W	16	0000H
0x19C	TMR16	R/W	16	0000H
0x19E	TMR17	R/W	16	0000H
0x1A0	TSR10	R	16	0000H
0x1A0	TSR10L	R	8	00H
0x1A2	TSR11	R	16	0000H
0x1A2	TSR11L	R	8	00H
0x1A4	TSR12	R	16	0000H
0x1A4	TSR12L	R	8	00H
0x1A6	TSR13	R	16	0000H
0x1A6	TSR13L	R	8	00H
0x1A8	TSR14	R	16	0000H
0x1A8	TSR14L	R	8	00H
0x1AA	TSR15	R	16	0000H
0x1AA	TSR15L	R	8	00H
0x1AC	TSR16	R	16	0000H
0x1AC	TSR16L	R	8	00H
0x1AE	TSR17	R	16	0000H
0x1AE	TSR17L	R	8	00H

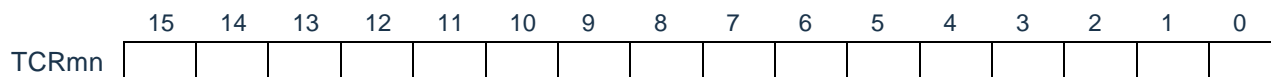
Offset address	Register name	Read-write properties	Bit width	Reset value
0x1B0	TE1	R	16	0000H
0x1B0	TE1L	R	8	00H
0x1B2	TS1	R/W	16	0000H
0x1B2	TS1L	R/W	8	00H
0x1B4	TT1	R/W	16	0000H
0x1B4	TT1L	R/W	8	00H
0x1B6	TPS1	R/W	16	0000H
0x1B8	TO1	R/W	16	0000H
0x1B8	TO1L	R/W	8	00H
0x1BA	TOE1	R/W	16	0000H
0x1BA	TOE1L	R/W	8	00H
0x1BC	TOL1	R/W	16	0000H
0x1BC	TOL1L	R/W	8	00H
0x1BE	TOM1	R/W	16	0000H
0x1BE	TOM1L	R/W	8	00H
0x318	TDR10	R/W	16	0000H
0x31A	TDR11	R/W	16	0000H
0x364	TDR12	R/W	16	0000H
0x366	TDR13	R/W	16	0000H
0x368	TDR14	R/W	16	0000H
0x36A	TDR15	R/W	16	0000H
0x36C	TDR16	R/W	16	0000H
0x36E	TDR17	R/W	16	0000H

6.2.2 Timer count register mn (TCRmn).

The TCRmn register is a 16-bit read-only register that counts the counting clock. Increments or decrements synchronously with the rising edge of the counting clock.

The operating mode is selected by the MDmn3 to MDmn0 bits of the timer mode register mn (TMRmn), and the increment and decrement counts are switched (see "6.3.3 Timer Mode Registers"). mn (TMRmn)).

Figure 6-2 Timer count register mn (TCRmn).



Note: m: unit number (m=0,1) n: channel number (when m=0: n=0~3, m=1: 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 TM4EN/TM8EN bit of peripheral enable register 0 (PER0).
- End of count of dependent channels in PWM output mode
- The count of dependent channels ends in delay counting mode
- At the end of the count of the master/slave channel in single trigger pulse output mode
- End of count of dependent channels in multiple PWM output mode

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

- Enter when the trigger starts in capture mode
- Capture at the end 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 value of the TCRmn register varies depending on the operating mode and operating state.

Table 6-3 The timer count register mn (TCRmn) read value in each operating mode

Operating mode	Counting mode	The timer counts the read value of the register mn (TCRmn).			
		The value at which the operating mode is changed after the reset is lifted	Count paused The value for (TTmn=1).	Count paused (TTmn=1) after changing the value at run mode	The value at which to wait after a single count when the trigger began
Interval timer mode	Decrement the count	FFFFH	The value at the time of stop	Indefinite value	—
Capture mode	Increment the count	0000H	The value at the time of stop	Indefinite value	—
Event counter pattern	Decrement the count	FFFFH	The value at the time of stop	Indefinite value	—
Single count mode	Decrement the count	FFFFH	The value at the time of stop	Indefinite value	FFFFH
Capture & single count mode	Increment the count	0000H	The value at the time of stop	Indefinite value	The capture value of the TDRmn register is +1

Note: Indicates the read value of the TCRmn register when channel n is in the stopped state of timer operation (TEmn=0) and the count allowable state (TSmn=1). Keep this value in the TCRmn register until the count starts.

Note: m: unit number (m=0,1) n: channel number (when m=0: n=0~3, m=1: n=0~7).

6.2.3 Timer data register mn (TDRmn).

This is a 16-bit register that can be used to switch between the capture function and the compare function. The operating mode is selected by the MDmn3 to MDmn0 bits of the timer mode register mn (TMRmn), and the capture function and the comparison function are switched.

The TDRmn register can be rewritten at any time.

This register can be read and written in 16-bit increments.

Timer4 is SPLIT in 8-bit timer mode (timer mode registers m1, m3 (TMRm1, TMRm3). The bit is "1", which can read and write the TDRm1 register and the TDRm3 register in 8 bits, where TDRm1H and TDRm3H is used as the high 8 bit, and TDRm1L and TDRm3L are used as the low 8 bit.

After generating the reset signal, the value of the TDRmn register changes to "0000H".

Figure 6-3 Timer data register mn(TDRmn) (n=0, 2, 4, 5, 6, 7).

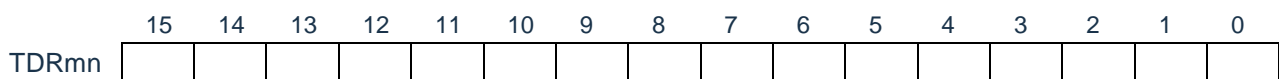
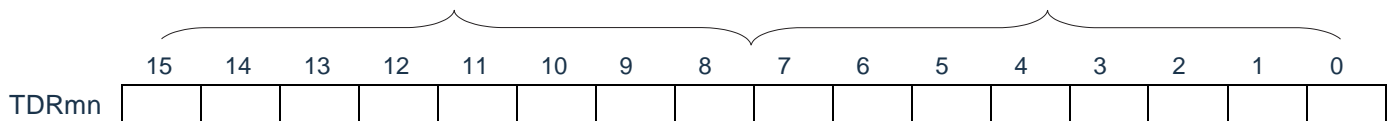


Figure 6-4 Timer data registers mn(TDRmn) (n=1, 3).

(TDR01H can support 8bit operation) (TDR01L can support 8bit operation).



- i. The case where the timer data register mn (TDRmn) is used as a comparison registerThe count is decremented from the setpoint of the TDRmn register and generates an interrupt signal (INTTMmn) when the count value changes to "0000H". The value of the TDRmn register is maintained until it is rewritten.

Note: Even if the input capture trigger signal, the TDRmn register set to the comparison function does not perform a capture operation.

- ii. The case where the timer data register mn (TDRmn) is used as the capture registerThe count value of the timer count register mn (TCRmn) is captured to the TDRmn register via input capture trigger. The effective edge of the TImn pin can be selected as the capture trigger signal. The timing mode register mn (TMRmn) sets the selection of capture triggers.

Note: m: unit number (m=0,1) n: channel number (when m=0: n=0~3, m=1: n=0~7).

6.3 Control registers of the universal timer unit

The registers that control the universal 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(TTm).
- Timer input and 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 Switch Control Register (ISC).
- Noise filter enable register 1 (NFEN1).
- Port Mode Control Register (PMCxx).
- Port Mode Register (PMxx).
- Port register (Pxx).

Note: The registers and bits allocated vary by product. The unassigned bits must be initialized.

Note: m: unit number (m= 0,1) n: channel number (when m=0: n=0~3, m=1: n=0~7).

6.3.1 Peripheral enable register 0 (PER0).

The PER0 register is a register that is set to enable or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use.

To use universal timer unit 0, bit0 (TM4EN) must be placed at "1". To use universal timer unit 1, bit1 (TM8EN) must be set to "1". The PER0 register is set via the 8-bit memory operation instruction. After generating a reset signal, the value of the PER0 register changes to "00H".

Figure 6-5 Table of Peripheral enable register 0 (PER0).

Address: 0x40020420 reset: 00HR/W

symbol	7	6	5	4	3	2	1	0
PER0	Xx	Xx	Xx	Xx	Xx	Xx	TM8EN	TM4EN

TM4EN	Control of the input clock of the universal timer unit 0
0	Stop providing the input clock. • SFR used by universal timer unit 0 cannot be written. • Universal timer unit 0 is reset.
1	An input clock is provided. • SFR for reading and writing universal timer unit 0.
TM8IN	Control of the input clock of the universal timer unit 1
0	Stop providing the input clock. • SFR for universal timer unit 1 cannot be written. • Universal timer unit 1 is reset.
1	An input clock is provided. • SFR that can read and write for use in Universal Timer Unit 1.

Note 1 To set the general-purpose timer unit, the following registers must first be set while the TM4EN/TM8EN bit is "1".

When the TM4EN/TM8EN bit is "0", the value of the control register of the timer array unit is the initial value, ignoring the write operation (timer input and output select register 0 (TIOS0), Input Switching Control Register (ISC), Noise Filter Enable Register 1 (NFEN1), Port Mode Control Register PMC x, except 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(TTm).
- Timer output enable register m (TOEm).
- Timer output register m(TOM).
- Timer output level register m(TOLm).
- Timer output mode register m (TOMm).

6.3.2 Timer clock selection register m (TPSm).

The TPSm registers are 16-bit registers that can be supplied to each channel with 2 or 4 common operating clocks (CKm0, CKm1, CKm2, CKm3). Select CKm0 by bit3~0 of the TPSm register and CKm1 by bit7~4 of the TPSm register. In addition, only channels 1 and 3 can select CKm2 and CKm3, via bit9~8 of the TPSm register to Select CKm2, selected by bit13 and bit12 of the TPSm register CKm3.

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

the case of PRSm00~PRSm03 bits can be rewritten (when m=0: n=0~3, m=1: n=0~7);

Channels that select CKm0 as the running clock (CKSmn1, CKSmn0=0, 0) are all in the stopped state (TEmn=0).

the case of PRSm10~PRSm13 bits can be rewritten (when m=0: n=0~3, m=1: n=0~7);

Channels selecting CKm2 as the operating clock (CKSmn1, CKSmn0=0, 1) are all in the stopped state (TEmn=0).

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

Channels that select CKm1 as the operating clock (CKSmn1, CKSmn0=1, 0) are all in the stopped state (TEmn=0).

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

Channels selecting CKm3 as the operating clock (CKSmn1, CKSmn0=1, 1) are all in the stopped state (TEmn=0).

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

Figure 6-6 timer clock selection 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	Select the operating clock (CKmk) (k=0, 1).					
					$f_{CLK}=2MHz$	$f_{CLK}=4MHz$	$f_{CLK}=8MHz$	$f_{CLK}=20MHz$	$f_{CLK}=32MHz$
0	0	0	0	f_{CLK}	2MHz	4MHz	8MHz	20MHz	32MHz
0	0	0	1	$f_{CLK}/2$	1MHz	2MHz	4MHz	10MHz	16MHz
0	0	1	0	$f_{CLK}/2^2$	500kHz	1MHz	2MHz	5MHz	8MHz
0	0	1	1	$f_{CLK}/2^3$	250kHz	500kHz	1MHz	2.5MHz	4MHz
0	1	0	0	$f_{CLK}/2^4$	125kHz	250kHz	500kHz	1.25MHz	2MHz
0	1	0	1	$f_{CLK}/2^5$	62.5kHz	125kHz	250kHz	625kHz	1MHz
0	1	1	0	$f_{CLK}/2^6$	31.3kHz	62.5kHz	125kHz	313kHz	500kHz
0	1	1	1	$f_{CLK}/2^7$	15.6kHz	31.3kHz	62.5kHz	156kHz	250kHz
1	0	0	0	$f_{CLK}/2^8$	7.81kHz	15.6kHz	31.3kHz	78.1kHz	125kHz
1	0	0	1	$f_{CLK}/2^9$	3.91kHz	7.81kHz	15.6kHz	39.1kHz	62.5kHz
1	0	1	0	$f_{CLK}/2^{10}$	1.95kHz	3.91kHz	7.81kHz	19.5kHz	31.25kHz
1	0	1	1	$f_{CLK}/2^{11}$	977Hz	1.95kHz	3.91kHz	9.77kHz	15.6kHz
1	1	0	0	$f_{CLK}/2^{12}$	488Hz	977Hz	1.95kHz	4.88kHz	7.81kHz
1	1	0	1	$f_{CLK}/2^{13}$	244Hz	488Hz	977Hz	2.44kHz	3.91kHz
1	1	1	0	$f_{CLK}/2^{14}$	122Hz	244Hz	488Hz	1.22kHz	1.95kHz
1	1	1	1	$f_{CLK}/2^{15}$	61.0Hz	122Hz	244Hz	610Hz	977Hz

Note: The universal timer unit (TTm=0,100FH) must be stopped when changing the clock selected as f_{CLK} (changing the value of the system clock control register (CKC)). Even when selecting the operating clock (f_{MCK}) or the effective edge of the input signal of the TImn pin, the general-purpose timer unit needs to be stopped.

Note: 1. Bit15, 14, 11, 10 must be set to "0".

2. If f_{CLK} (no divider) is selected as the operating clock (CKmk) and TDRnm is set to "0000H" (n =0, 1, m=0~3), you cannot use interrupt requests for a universal timer unit.

Note: 1. f_{CLK} : Cpu/peripheral hardware clock frequency

2. The clock waveform selected by the TPSm register starts from the rising edge with only 1 f_{CLK} period as high (m=1~15). For details, please refer to "6.5.1 Counting Clock (f_{TCLK})".

Figure 6-7 timer clock selection 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

PRS m21	PRS m20	Select ^{note} for the running clock (CKm2).					
			$f_{CLK}=2MHz$	$f_{CLK}=4MHz$	$f_{CLK}=8MHz$	$f_{CLK}=20MHz$	$f_{CLK}=32MHz$
0	0	$f_{CLK}/2$	1MHz	2MHz	4MHz	10MHz	16MHz
0	1	$f_{CLK}/2^2$	500kHz	1MHz	2MHz	5MHz	8MHz
1	0	$f_{CLK}/2^4$	125kHz	250kHz	500kHz	1.25MHz	2MHz
1	1	$f_{CLK}/2^6$	31.3kHz	62.5kHz	125kHz	313kHz	500kHz

PRS m31	PRS m30	Run clock (CKm3) selection ^{note}					
			$f_{CLK}=2MHz$	$f_{CLK}=4MHz$	$f_{CLK}=8MHz$	$f_{CLK}=20MHz$	$f_{CLK}=32MHz$
0	0	$f_{CLK}/2^8$	7.81kHz	15.6kHz	31.3kHz	78.1kHz	125kHz
0	1	$f_{CLK}/2^{10}$	1.95kHz	3.91kHz	7.81kHz	19.5kHz	31.3kHz
1	0	$f_{CLK}/2^{12}$	488Hz	977Hz	1.95kHz	4.88kHz	7.81kHz
1	1	$f_{CLK}/2^{14}$	122Hz	244Hz	488Hz	1.22kHz	1.95kHz

Note: The universal timer unit (TTm=0,100FH) must be stopped when changing the clock selected as f_{CLK} (changing the value of the system clock control register (CKC)). Even when selecting the operating clock (f_{MCK}) or the effective edge of the input signal of the TImn pin, the general-purpose timer unit needs to be stopped.

Note: Bit15, 14, 11, 10 must be set to "0".

If you use Channel 1 and Channel 3 in 8-bit timer mode and use CKm2 and CKm3 as operating clocks, you can implement with the interval timer function as interval time shown in Tables 6-4.

Table 6-4 The interval time that can be set by the operating clocks CKSm2 and CKSm3

clock		Interval time ^{note} ($f_{CLK}=32MHz$).			
		10μs	100μs	1ms	10ms
CKm2	$f_{CLK}/2$	○	—	—	—
	$f_{CLK}/2^2$	○	—	—	—
	$f_{CLK}/2^4$	○	○	—	—
	$f_{CLK}/2^6$	○	○	—	—
CKm3	$f_{CLK}/2^8$	—	○	○	—
	$f_{CLK}/2^{10}$	—	○	○	—
	$f_{CLK}/2^{12}$	—	—	○	○
	$f_{CLK}/2^{14}$	—	—	○	○

Note: ○ Errors within 5% are included.

Note: 1. f_{CLK} : Cpu/peripheral hardware clock frequency

2. For details of the $f_{CLK}/2^r$ waveform selected for the TPSm register, please refer to "6.5.1 Counting Clock (f_{TCLK})".

6.3.3 Timer mode register mn (TMRmn).

The TMRmn register is a register that sets the operating mode of channel n, and performs the selection of the operating clock (f_{MCK}), the selection of the counting clock, the selection of the master/slave, the selection of 16 bits/ Selection of 8-bit timers (channels 1 and 3 only), setting of start and capture triggers, selection of effective edges of timer inputs, and operating modes (Interval, Capture, Event Counter, Single Count, Capture & Single Count) setting.

It is forbidden to overwrite the TMRmn register in operation (TE_{mn}=1). However, bit7 and bit6 (CIS_{mn}1, CIS_{mn}0) can be rewritten in some function operations (TE_{mn}=1). (For details, please refer to "Independent Channel Operation Function of 6.8 Universal Timer Unit" and "Multi-channel Linkage Operation Function of 6.9 Timer Array Unit".).

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

Note that bit11 of the TMRmn register varies by channel.

TMRm2, TMRm4, TMRm6: MASTER_{mn} bit (n=2, 4, 6)

TMRm1, TMRm3 : SPLIT_{mn} bit (n=1, 3)

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

Figure 6-8 Timer mode register mn (TMRmn) (1/4).

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMRMN (n=2,4,6)	CKSmn 1	CKSmn 0	0	CCSmn	MASTER mn	STSm n2	STSm n1	STSm n0	CISm n1	CISm n0	0	0	MDm n3	MDm n2	MDm n1	MDm n0

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMRMN (n=1,3)	CKSmn 1	CKSmn 0	0	CCSmn	SPLITmn	STSm n2	STSm n1	STSm n0	CISm n1	CISm n0	0	0	MDm n3	MDm n2	MDm n1	MDm n0

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMRMN (n=0,5,7)	CKSmn 1	CKSmn 0	0	CCSmn	0 Note 1	STSm n2	STSm n1	STSm n0	CISm n1	CISm n0	0	0	MDm n3	MDm n2	MDm n1	MDm n0

CKSmn1	CKSmn0	Channel n running clock (f _{MCK}) selection
0	0	The timer clock selects the operating clock CKm0 set by register m (TPSm).
0	1	The timer clock selects the operating clock CKm2 set by register m (TPSm).
1	0	The timer clock selects the operating clock CKm1 set by register m (TPSm).
1	1	The timer clock selects the operating clock CKm3 set by register m (TPSm).
The operating clock (f _{MCK}) is used for edge detection circuitry. The sample clock and count clock (f _{TCLK}) are generated by setting the CCSmn bit. Only Channels 1 and 3 can select the operating clocks CKm2 and CKm3.		

CCSmn	Selection of channel n count clock (f _{TCLK}).
0	CKSmn0 bits and CKSmn1 bits specify the operating clock (f _{MCK}).
1	The effective edge of the input signal at the TImn pin • In the case of Unit 0: Channel 0: The effective edge of the input signal selected by TIS0 Channel 1: The effective edge of the input signal selected by TIS0
The counting clock (f _{TCLK}) is used for counters, output control circuits, and interrupt control circuits.	

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

Remark: 1. Bit13, 5, 4 must be placed with "0".

- To change the clock selected as f_{CLK} (change the value of the System Clock Control Register (CKC)), even if the CKSmn0 bit and CKSmn1 bits are selected, the specified operating clock (f_{MCK}) or the effective edge of the input signal of the TImn pin as a counting clock (f_{TCLK}), must also stop the timer array monolithic ($f_{TCLK} TIm=0, 10FFH$)

Note: m: unit number (m=0,1) n: channel number (when m=0: n=0~3, m=1: n=0~7).

Figure 6-9 timer mode register mn (TMRmn) (2/4).

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMRMN (n=2,4,6)	CKSmn 1	CKSmn 0	0	CCSmn	MASTER mn	STSm n2	STSm n1	STSm n0	CISm n1	CISm n0	0	0	MDm n3	MDm n2	MDm n1	MDm n0

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMRMN (n=1,3)	CKSmn 1	CKSmn 0	0	CCSmn	SPLITmn	STSm n2	STSm n1	STSm n0	CISm n1	CISm n0	0	0	MDm n3	MDm n2	MDm n1	MDm n0

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMRMN (n=0,5,7)	CKSmn 1	CKSmn 0	0	CCSmn	0 Note 1	STSm n2	STSm n1	STSm n0	CISm n1	CISm n0	0	0	MDm n3	MDm n2	MDm n1	MDm n0

(TMRmn(n=2,4,6) bit11)

MASTERmn	The choice of independent channel operation/multi-channel linkage operation (subordinate or master) of channel n
0	A slave channel that is used as a stand-alone channel operation function or a multi-channel linkage operation function.
1	It is used as the main control channel of the multi-channel linkage operation function.
Channels 2, 4, 6 can only be set as the main control channel (MASTERmn=1). Channel 0 is fixed to "0" (because channel 0 is the channel with the highest bit, it is irrelevant to the setting of this bit and is used as the master channel). For channels used as stand-alone channel operation functions, place the MASTERmn position "0".	

(TMRmn(n=1,3) bit11)

SPLITmn	Operational selection for 8-bit timers/16-bit timers for Channel 1 and Channel 3
0	Used as a 16-bit timer. (Used as a slave channel for stand-alone channel operation function or multi-channel linkage operation function)
1	Acts as an 8-bit timer.

STSmn2	STSmn1	STSmn0	The start trigger of channel n and the setting of the capture trigger
0	0	0	Only software triggers start to work (no other trigger source is selected).
0	0	1	Use the effective edge of the TImn pin input for start triggering and capture triggering.
0	1	0	Use the bilateral edges of the TImn pin input for start triggering and capture triggering, respectively.
1	0	0	Use the interrupt signal of the master channel (in the case of slave channels for multi-channel linkage operation).
Other than the above			Disable settings.

Note: 1. bit11 is a read-only bit, fixed to "0", ignoring write operations.

Note: m: unit number (m=0,1) n: channel number (when m=0: n=0~3, m=1: n=0~7).

Figure 6-10 timer mode register mn(TMRmn) (3/4).

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMRMN (n=2,4,6)	CKSmn 1	CKSmn 0	0	CCSmn	MASTER mn	STSm n2	STSm n1	STSm n0	CISm n1	CISm n0	0	0	MDm n3	MDm n2	MDm n1	MDm n0

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMRMN (n=1,3)	CKSmn 1	CKSmn 0	0	CCSmn	SPLITmn	STSm n2	STSm n1	STSm n0	CISm n1	CISm n0	0	0	MDm n3	MDm n2	MDm n1	MDm n0

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMRMN (n=0,5,7)	CKSmn 1	CKSmn 0	0	CCSmn	0 Note 1	STSm n2	STSm n1	STSm n0	CISm n1	CISm n0	0	0	MDm n3	MDm n2	MDm n1	MDm n0

CISmn1	CISmn0	Effective edge selection for the TImn pin
0	0	Falling edge
0	1	Rising edge
1	0	Bilateral edges (when measuring low-level widths) Start Trigger: Falling Edge, Capture Trigger: Rising Edge
1	1	Bilateral edges (when measuring high level width) Start trigger: Rising edge, Capture trigger: Falling edge
When the STSmn2~STSmn0 bit is not "010B" and is specified using a bilateral edge, the CISmn1~CISmn0 position must be "10B".		

Note: 1. bit11 is a read-only bit, fixed to "0", ignoring write operations.

Note: m: Unit number (m=0,1) n: Channel number (when m=0: n=0~3, m=1: n=0~7).

Figure 6-11 timer mode register mn(TMRmn) (4/4).

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMRMN (n=2,4,6)	CKSmn 1	CKSmn 0	0	CCSmn	MASTER mn	STSm n2	STSm n1	STSm n0	CISm n1	CISm n0	0	0	MDm n3	MDm n2	MDm n1	MDm n0

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMRMN (n=1,3)	CKSmn 1	CKSmn 0	0	CCSmn	SPLITmn	STSm n2	STSm n1	STSm n0	CISm n1	CISm n0	0	0	MDm n3	MDm n2	MDm n1	MDm n0

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMRMN (n=0,5,7)	CKSmn 1	CKSmn 0	0	CCSmn	0 Note 1	STSm n2	STSm n1	STSm n0	CISm n1	CISm n0	0	0	MDm n3	MDm n2	MDm n1	MDm n0

MD mn3	MD mn2	MD mn1	Setting of channel n operating mode	Corresponding functions	The count of TCR runs
0	0	0	Interval timer mode	Interval timer/square wave output/ Divider function/PWM output (master).	Decrement the count
0	1	0	Capture mode	Input the measurement of the pulse interval	Increment the count
0	1	1	Event counter pattern	External event counters	Decrement the count
1	0	0	Single count mode	Delay counter/single trigger pulse output/PWM output (Subordinate)	Decrement the count
1	1	0	Capture & Single Count mode	Measurement of the high and low level width of the input signal	Increment the count
Other than the above			Disable settings.		
The operation of each mode varies depending on the MDmn0 bit (see the table below).					

Operating mode (MDmn3~MDmn1 bit setting (refer to table above)).	MD mn0	Start count and interrupt settings
<ul style="list-style-type: none"> Interval timer mode (0, 0, 0) capture mode (0, 1, 0). 	0	No timer interrupt is generated at the start of counting (nor does the output of the timer change).
	1	A timer interrupt is generated at the start of counting (the output of the timer also changes).
<ul style="list-style-type: none"> Event counter pattern (0, 1, 1). 	0	No timer interrupt is generated at the start of counting (nor does the output of the timer change).
<ul style="list-style-type: none"> Single count mode Note 2 (1, 0, 0). 	0	The start trigger in the count run is invalid. There is no interruption at this point.
	1	Count the start of the run to trigger valid note 3. There is no interruption at this point.
<ul style="list-style-type: none"> Capture & Single Count mode (1, 1, 0). 	0	No timer interrupt is generated at the start of counting (nor does the output of the timer change). The start trigger in the count run is invalid. There is no interruption at this point.

Note: 1 bit11 is a read-only bit, fixed to "0", ignoring write operations.

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

3. If a start trigger (TSmn=1) is generated during the run, the counter is initialized and the count is restarted (no interrupt request is generated). Note: m: unit number (m=0,1) n: channel number (when m=0: n=0~3, m=1: n=0~7).

6.3.4 Timer status register mn (TSRmn).

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

The TSRmn register is only valid in capture mode (MDmn3~MDmn1=010B) and capture single counting mode (MDmn3~MDmn1=110B). For changes in the OVF bits in each operating mode and set/clear conditions, refer to Table 6-5.

Read the TSRmn registers via the 16-bit memory operation instructions.

User can read the lower 8 bits of the TSRmn register with TSRmnL and read the TSRmn register through the 8-bit memory operation instruction. After generating a reset signal, the value of the TSRmn register changes to "0000H".

Figure 6-12 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 for channel n
0	No overflow occurred.
1	Overflow occurs.
If the OVF bit is "1", clear this flag (OVF=0) just when the next count does not overflow and the count value is captured.	

Note: m: unit number (m=0,1)n: channel number (when m=0: n=0~3, m=1: n=0~7).

Table 6-5 Variation of OVF bits and setting/clearing conditions in each operating mode

Timer operating mode	OVF bit	Set/clear conditions
• Capture mode	clear	No overflow occurred during capture
• Capture & Single Count mode	Set	An overflow occurs while capturing
• Interval timer mode	clear	— (Cannot be used)
• Event counter mode	Set	
• Single count mode		

Note: Even if the counter overflows, the OVF bit does not change immediately, but during subsequent captures.

6.3.5 Timer channel enable status register m (TEm).

TEm registers are registers that indicate the Enabled or stopped states in which each channel timer operates.

Each of the TEm registers corresponds to the timer channel start register m(TSm) and the timer channel stop register m(TTm). If the position of the TSm register is "1", the corresponding position of the TEm register is "1". If you place the TTm register "1" at each position, it corresponds to the bit "0".

Read the TEm registers via the 16-bit memory operation instructions.

User can read the lower 8 bits of the TEm register with TEmL and through the 8-bit memory operation instruction. After generating the reset signal, the value of the TEm register changes to "0000H".

Figure 6-13 timer channels Enabled status register m(TEm).

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEm	0	0	0	0	TEHm ₃	0	TEHm ₁	0	0	0	0	0	TEm ₃	TEm ₂	TEm ₁	TEm ₀
m=0																

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEm	0	0	0	0	0	0	0	0	TEm ₇	TEm ₆	TEm ₅	TEm ₄	TEm ₃	TEm ₂	TEm ₁	TEm ₀
m=1																

TEHm3	Channel 3 is an indication of the high 8-bit timer when the operation of the high 8-bit timer enable or stops the state
0	Run stopped state
1	Run Enabled status

TEHm1	Channel 1 is an indication of the high 8-bit timer when the operation of the high 8-bit timer enable or stops the state
0	Run stopped state
1	Run Enabled status

TEmn	The operation of channel n enable or stops the indication of the state
0	Run stopped state
1	Run Enabled status
When Channel 1 and Channel 3 are in 8-bit timer mode, TEm1 and TEm3 indicate the operation of the low 8-bit timer Enabled or stopped.	

Note: m: unit number (m=0,1)n: channel number (when m=0: n=0~3, m=1: n=0~7).

6.3.6 Timer channel start register m(TSm).

The TSm register is a trigger register that initializes the timer count register mn (TCRmn) and sets the start of each channel count run. If each position is "1", the corresponding bit of the Enabled status register m (TEm) of the timer channel is placed "1". Because the TSmn bit, TSHm1 bit, and TSHm3 bit are trigger bits, if it becomes a run-Enabled state (TEmn, TEHm1, TEHm3=1), immediately clear the TSmn bit, TSHm1 bit, and TSHm3 bit.

The TSm register is set via the 16-bit memory operation instruction.

User can set the lower 8 bits of the TSm register with TSmL and through the 8-bit memory operation instruction. After generating a reset signal, the value of the TSm register changes to "0000H".

Figure 6-14 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	TSHm ₃	0	TSHm ₁	0	0	0	0	0	TSm ₃	TSm ₂	TSm ₁	TSm ₀

TSHm ₃	The operation of the high 8-bit timer when Channel 3 is in 8-bit timer mode enable (start) triggering
0	No triggering.
1	Place teem ₃ at position "1" and enter the Count Enabled state. If the count of TCRm ₃ registers is started in the counting Enabled state, it enters interval timer mode (see Table 6-6 of "Start Timing of the Counter 6.5.2").

TSHm ₁	The operation of the high 8-bit timer when Channel 1 is in 8-bit timer mode enable (start) triggering
0	No triggering.
1	Place TEHm ₁ at position "1" and enter the Count Enabled state. If the count of TCRm ₁ registers is started in the counting Enabled state, it enters interval timer mode (see Table 6-6 of "Start Timing of 6.5.2 Counters").

TSmn	The operation of channel n enable (start) triggering
0	No triggering.
1	Place the TEMn position "1" and enter the Count Enabled state. The start of counting the TCRmn registers in the count-Enabled state varies depending on each operating mode (see Table 6-6 of "Start Timing of Counters 6.5.2"). In Channel 1 and Channel 3 are 8-bit timer modes, TSm ₁ and TSm ₃ are 8 lows The operation of the bit timer enable (start) triggering.

Note 1 Bit15~12, 10, 8~4 must be set to "0".

- When switching from the function of not using the TImn pin input to the function of using the TImn pin input, from setting the timer mode register mn (TMRmn) to setting TSmn (TSHm₁, TSHm₃) position "1", the following period of waiting is required:

When the TImn pin noise filter is in effect (TNFENmn=1): 4 operating clocks (fMCK).

When the TImn pin noise filter is invalid (TNFENmn=0): 2 operating clocks (fMCK).

Note: 1. The read value of the TSm register is always "0".

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

6.3.7 Timer channel stop register m(TTm).

The TTm register is the trigger register that sets the stop of each channel count.

If each position is "1", the corresponding bit of the Enabled status register m (TEm) of the timer channel is cleared "0". Because the TTmn bits, TTHm1 bits, and TTHm3 bits are trigger bits, if it becomes a running stop state (TEmn, TEHm1, TEHm3=0), immediately clear the TTmn bits, TTHm1 bits, and TTHm3 bits.

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

User can set the lower 8 bits of the TTm register with TTmL and through the 8-bit memory operation instruction. After generating a reset signal, the value of the TTm register changes to "0000H".

Figure 6-15 timer channel stop register m(TTm).

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TTm	0	0	0	0	TTHm ₃	0	TTHm ₁	0	0	0	0	0	TTm ₃	TTm ₂	TTm ₁	TTm ₀
m=0																

symbol	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	TTm ₇	TTm ₆	TTm ₅	TTm ₄	TTm ₃	TTm ₂	TTm ₁	TTm ₀
m=1																

TTHm3	The operation of the high 8-bit timer is triggered when Channel 3 is in 8-bit timer mode
0	No triggering.
1	Clear the TEHm3 bit to "0" and enter the count stop state.

TTHm1	The operation of the high 8-bit timer is triggered when Channel 1 is in 8-bit timer mode
0	No triggering.
1	Clear the TEHm1 bit to "0" and enter the count stop state.

TTmn	The operation of channel n stops triggered
0	No triggering.
1	Clear the TEmn bit to "0" and enter the count stopped state. When Channel 1 and Channel 3 are in 8-bit timer mode, TTm1 and TTm3 are triggered by the operation of the low 8-bit timer.

Note: Bit15~12, 10, 8~4 must be set to "0".

Note: 1.The read value of the TTm register is always "0".

2.m: unit number (m=0,1)n: channel number (m=0: n=0~3, m=1: n=0~7).

6.3.8 Timer input and output selection register (TIOS0).

The TIOS0 register selects the timer inputs for Channel 0 and Channel 1 for Unit 0 and the timer output for Channel 2. The TIOS0 register is set via the 8-bit memory operation instruction. After generating the reset signal, the value of the TIOS0 register changes to "00H".

Figure 6-16 timer input selection register 0 (TIOS0).

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

TIS07	TIS06	TIS05	Channel 0 uses the choice of timer input
0	0	0	Input signal from the timer input pin (TI00).
other			Setting Prohibited

TIS04	Channel 0 uses the choice of timer input
0	Input signal selected by TIS07~TIS05
1	EIC's event input signal

TOS03	Enable of the timer output of channel 2
0	Output is Enabled
1	Disable output (output fixed to 0).

TIS02	TIS01	TIS00	Channel 1 uses the choice of timer inputs
0	0	0	The input signal of the timer input pin (TI01).
0	0	1	Event input signal for EVENTC
0	1	0	The input signal of the timer input pin (TI01).
0	1	1	
1	0	0	Low-speed internal oscillator clock (f_{IL}).
1	0	1	Subsystem clock (f_{SUB}).
Other than the above			Disable settings.

Note: 1. The selected timer input needs to have a high or low level width greater than or equal to $1/fMCK+10ns$.

Therefore, when selecting f_{SUB} as the f_{CLK} ($CSS=1$ for the CKC register), it is not possible to place TIS02 at "1".

- When selecting the event input signal of the ELC through timer input select register 0 (TIOS0), register 0 (TPS0) must be selected through the timer clock f_{CLK} .

6.3.9 Timer output enable register m (TOEm).

ToEm registers are registers that are set to enable or disable the output of each channel timer.

For channel n that enable the timer output, the value of the TOMn bit of the timer output register m(TOm) described below cannot be overridden by software, and the value reflected by the timer output function of the counting operation is from the output pin of the timer (TOMn) output.

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

User can set the low 8 bits of the TOEm register by toemL and through the 8-bit memory operation instruction. After generating a reset signal, the value of the TOEm register changes to "0000H".

Figure 6-17 timer outputs Enabled register m (TOEm).

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOEm	0	0	0	0	0	0	0	0	0	0	0	0	TOE m3	TOE m2	TOE m1	TOE m0

m=0

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 m7	TOE m6	TOE m5	TOE m4	TOE m3	TOE m2	TOE m1	TOE m0

m=1

TOEmn	Allow/disable of the timer output of channel n
0	Disables timer output. The operation of the timer is not reflected to the TOMn bit, the output is fixed. User can write the TOMn bit and output the level set by the TOMn bit from the TOMn pin.
1	Timer output is Enabled. The operation of the timer is reflected to the TOMn bit, resulting in an output waveform. Ignore the write operation of the TOMn bit.

Note: Bit15~4 must be set to "0".

Note: m: unit number (m=0,1)n: channel number (when m=0: n=0~3, m=1: n=0~7).

6.3.10 Timer output register m (TOM).

The TOM register is a buffer register for the output of each channel timer.

The values of this register are output from the output pin (TOMn) of each channel timer.

The TOMn bit of this register can only be rewritten by software when the timer output (TOEmn=0) is disabled.

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

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

The TOM register is set by the 16-bit memory operation instructions.

User can set the lower 8 bits of the TOM register with TOML and through the 8-bit memory operation instruction. After generating a reset signal, the value of the TOM register changes to "0000H".

Figure 6-18 timer output register m(TOM).

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOM	0	0	0	0	0	0	0	0	0	0	0	0	TOM3	TOM2	TOM1	TOM0
m=0																

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
m=1																

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

Note: Bit15~4 must be set to "0".

Note: m: unit number (m=0,1)n: channel number (when m=0: n=0~3, m=1: n=0~7).

6.3.11 Timer output level register m(TOLm).

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

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

The TOLm register is set via the 16-bit memory operation instruction.

User can set the low 8 bits of the TOLm register with TOLmL and through the 8-bit memory operation instruction. After generating a reset signal, the value of the TOLm register changes to "0000H".

Figure 6-19 Timer output level register m (TOLm).

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Dust	0	0	0	0	0	0	0	0	0	0	0	0	TOLm3	TOLm2	TOLm1	0
m=0																

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Dust	0	0	0	0	0	0	0	0	Dust7	DUST6	Dust5	Dust4	TOLm3	TOLm2	TOLm1	0
m=1																

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

Note: Bit15~4 and bit0 must be set to "0".

Remarks: 1 If you override the value of this register during timer operation, the output logic of the timer is inverted the next timer output signal changes, rather than inverting immediately after the rewrite.

2.m: unit number (m=0,1)n: channel number (m=0: n=0~3, m=1: n=0~7).

6.3.12 Timer output mode register m (TOMm).

The TOMm register is the register that controls the output mode of each channel timer. When used as a stand-alone channel operation function, the corresponding position of the channel used is "0".

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

When the timer output (TOEmn=1) is Enabled, the timing of the set and reset of the timer output signal reflects the setting of each channel n by this register.

The TOMm register is set via 16-bit memory operation instruction.

User can set the low 8 bits of the TOMm register by tomml and through the 8-bit memory operation instruction. After generating a reset signal, the value of the TOMm register changes to "0000H".

Figure 6-20 timer output mode register m(TOMm).

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOMm	0	0	0	0	0	0	0	0	0	0	0	0	TOM _{m3}	TOM _{m2}	TOM _{m1}	0
m=0																
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 _{m7}	TOM _{m6}	TOM _{m5}	TOM _{m4}	TOM _{m3}	TOM _{m2}	TOM _{m1}	0
m=1																

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

Note: Bit15~4 and bit0 must be set to "0".

Remark:

m: unit number (m=0,1); n: Channel number (when m=0: n=0~3, m=1: n=0~7)

Master Channel Number:

While m=0,n=0, 2;while m=1,n=0, 2, 4, 6

Slave channel number p:

While m=0,n<p≤3;while m=1,n<p≤7

(For details on the relationship between master channels and slave channels, see "6. 4.1 Basic rules of the multi-channel linkage operation function").

6.3.13 Input Select Control Register (ISC).

The ISC1 bits and ISC0 bits of the ISC registers are used for the coordination of Channel 3 and the Universal Serial Communication Unit to achieve LIN-bus communication. If the ISC1 position is "1", the input signal of the serial data input pin (RxD0) is selected as the input to the timer.

For setting the SSIE00 bits, refer to "19.3.14 Input Switching Control Register (ISC)". The ISC register is set by the 8-bit memory operation instruction.

After generating a reset signal, the value of the ISC register changes to "00H".

Figure 6-21 Input Switching Control Registers (ISC).

Address: 0x40040473		After reset: 00H			R/W			
symbol	7	6	5	4	3	2	1	0
ISC	SYESE00	0	0	0	0	0	ISC1	ISC0

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

ISC1	Input switching of channel 3 of the universal timer unit 0
0	Use the input signal from the TI03 pin as the input to the timer (usually operating).
1	Use the input signal from the RxD0 pin as the input to the timer (detect the wake-up signal and measure the low width of the interval segment and the pulse width of the sync segment).

ISC0	Input switching of external interrupt (INTP0).
0	Use the input signal from the INTP0 pin as input to an external interrupt (usually operating).
1	Use the input signal from the RxD0 pin as the input to an external interrupt (detect wake-up signal).

Note: Bit6~2 must be set to "0".

Note: To communicate using LIN-bus, the input signal of the RxD0 pin must be selected by placing ISC1 at "1".

6.3.14 Noise filter enable registers (NFEN1/NFEN2).

The NFEN1/NFEN2 registers set whether the noise filter is used for the input signal of each channel timer input pin. For pins that need to be noise-cancelling, the corresponding position "1" must be placed 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 target channel; When the noise filter is invalid, synchronization is performed only through the operating clock (f_{MCK}) of the target channel.

The NFEN1/NFEN2 registers are set by the 8-bit memory operation instructions. After generating a reset signal, the value of the NFEN1/NFEN2 register changes to "00H".

Note: For details, please refer to "6.5.1(2) To select the effective edge of the input signal of the TImn pin (CCSmn=1)", "6.5.2" Counter Start Timing" and "Control of 6.7 Timer Input (TImn)".

Figure 6-22 noise filter allowable register 1 (NFEN1).

Address: 0x40040471	After reset: 00H				R/W			
symbol	7	6	5	4	3	2	1	0
NFEN1	0	0	0	0	TNFEN03	TNFEN02	TNFEN01	TNFEN00

Address: 0x40040472	After reset: 00H				R/W			
symbol	7	6	5	4	3	2	1	0
NFEN2	TNFEN17	TNFEN16	TNFEN15	TNFEN14	TNFEN13	TNFEN12	TNFEN11	TNFEN10

TNFEN03	the usage (or not) of input signal noise filters at the TI03 pin or RxD0 pin <small>Note</small>
0	Noise filter OFF
1	Noise filter ON

TNFEN02	The input signal noise filter of the TI02 pin is used or not
0	Noise filter OFF
1	Noise filter ON

TNFEN01	The input signal noise filter of the TI01 pin is used or not
0	Noise filter OFF
1	Noise filter ON

TNFEN00	The input signal noise filter of the TI00 pin is used or not
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: The noise filter of the TI03 pin can be selected. ISC1=1: The noise filter of the RxD0 pin can be selected.

TNFEN1n	Ti1n pin input signal noise filter is not used or not
0	Noise filter OFF
1	Noise filter ON

n=0~7

Note: The availability of timer input /output pins for a channel varies by product. For details, please refer to "Timer Input/Output Pins for Each Product in Table 6-2".

6.3.15 Registers that control the function of the timer input/output pin ports

When using a general-purpose timer unit, control registers for port functions that are multiplexed with the target channel (port mode register (PMxx), port register (Pxx), and port mode control register (PMCxx)) must be set. For details, please refer to "2.3.1 Port Mode Register (PMxx)", "2.3.2 Port Register (Pxx.) for details and "2.3.6 Port Mode Control Registers (PMCxx)".

The set port mode registers (PMxx), port registers (Pxx), and port mode control registers (PMCxx) vary by product. For details, please refer to "Register Settings when Using the Multiplexing Function in 2.5".

When using the multiplex port of the timer output pin as the output of the timer, the bit of the port mode control register (PMCxx), the bit of the port mode register (PMxx), and the position of the port register (Pxx) "0" must be used for each port.

(Example) When P01/TO00 is used as a timer output

Place the port mode control register 0 at PMC01 position "0".

Place the PORT mode register 0 at the PM01 position "0".

Place the P01 position of port register 0 at "0".

When using the multiplex port of the timer input pin as the input to the timer, the position of the corresponding port mode register (PMxx) for each port must be "1" and the position of the port mode control register (PMCxx) "0". At this point, the bit of the port register (Pxx) can be "0" or "1".

(Example) When P00/TI00 is used as a timer input

Place the port mode control register 0 at pmc00 position "0".

Place the PORT mode register 0 at PM00 position "1".

Place the P00 position of port register 0 at "0" or "1".

6.4 The basic rules of the universal timer unit

6.4.1 The basic rules of the multi-channel linkage operation function

The multi-channel linkage operation function is a function that combines the master channel (the reference timer that mainly counts the cycles) and the slave channel (the timer that follows the operation of the master channel), and several rules need to be observed when using it.

The basic rules of the multi-channel linkage operation function are as follows.

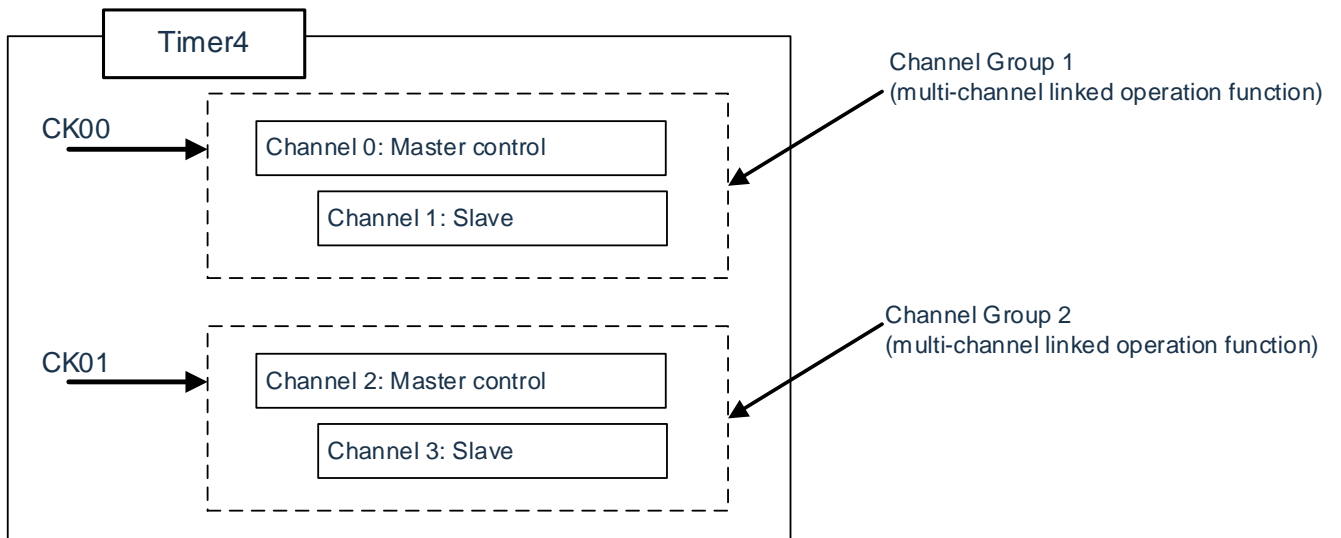
- 1) Only an even number of channels (channel 0, channel 2) can be set as the main control channel.
- 2) Any channel other than channel 0 can be set as a slave channel.
- 3) Only the low-level channel of the master channel can be set as a slave channel.
For example, when channel 0 is set to the main control channel, the channel (channel 1, channel 2, channel 3) that starts with channel 1 can be set as the slave channel.
- 4) Multiple slave channels can be set for one master channel.
- 5) When using multiple master channels, you cannot set a slave channel that spans the master channel.
For example, when channel 0 and channel 2 are set to the main control channel, channel 1 can be set to the slave channel of the main control channel 0, but channel 3 cannot be set to the slave channel of the main control channel 0.
- 6) The slave channel linked to the master channel needs to set the same operating clock. Bits 15 and bit14 of the slave channels linked to the master channel are CKSmn0 bits and CKSmn1 bits (timer mode register mn(TMRmn).") values need to be the same setpoint.
- 7) The master channel can pass intTMmn (interrupt), start software trigger, and count clock to the low channel.
- 8) Slave channels can use the master channel's INTMmn (interrupt), start software trigger, and count clock as the source clock, but cannot pass their own INTMmn (interrupt), start software trigger, and count clock to the low channel.
- 9) The master channel cannot use the INTMmn (interrupt), start software trigger, and count clocks of other high-level master channels as the source clock.
- 10) In order to start the channel to be linked at the same time, the channel start trigger bit (TSmn) of the linkage channel needs to be set at the same time.
- 11) Only all channels of the linkage or the master channel can use the setting of the TSmn bit in the counting operation. You cannot use only the setting of the TSmn bit of the slave channel.
- 12) In order to stop the channel to be linked at the same time, the channel stop trigger bit (TTmn) of the linkage channel needs to be set at the same time.
- 13) In interlink operation, because the master channel and the slave channel need the same operating clock, CKm2/CKm3 cannot be selected.
- 14) The timer mode register m0 (TMRm0) is fixed to "0" without a master bit. However, because channel 0 is the highest level channel, channel 0 can be used as the master channel when the linkage is running.

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

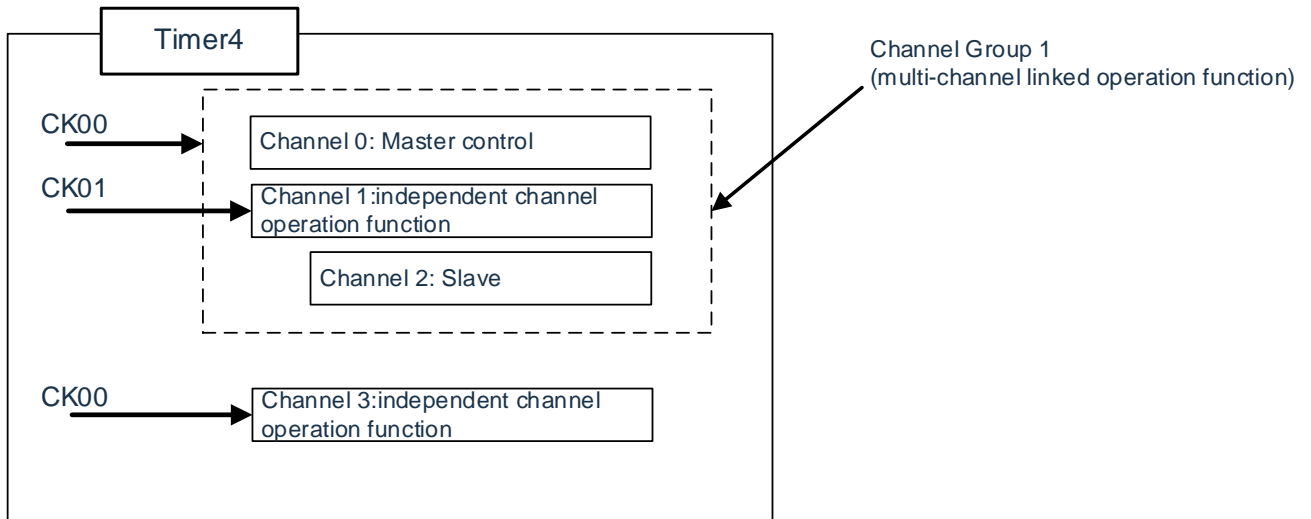
If you set 2 or more channel groups that are not linked to each other, the above basic rules do not apply to the channel groups.

Note: m: unit number (m=0,1)n: channel number (when m=0: n=0~3, m=1: n=0~7).

Example 1



Example 2



6.4.2 Timer channel start register m(TSm).

The TSm register is a trigger register that initializes the timer count register mn (TCRmn) and sets the start of each channel count run. If each position is "1", the corresponding bit of the Enabled status register m (TEm) of the timer channel is placed "1". Because the TSmn bit, TSHm1 bit, and TSHm3 bit are trigger bits, if it becomes a run-Enabled state (TEmn, TEHm1, TEHm3=1), immediately clear the TSmn bit, TSHm1 bit, and TSHm3 bit.

The TSm register is set via a 16-bit memory operation instruction.

User can set the low 8 bits of the TSm register with TSmL and through the 8-bit memory operation instruction. After generating a reset signal, the value of the TSm register changes to "0000H".

Figure 6-23 timer channel start register m(TSm).

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSm	0	0	0	0	TSHm3	0	TSHm1	0	0	0	0	0	TSm3	TSm2	TSm1	TSm0
m=0																
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
m=1																

TSHm3	The operation of the high 8-bit timer when Channel 3 is in 8-bit timer mode enable (start) triggering
0	No triggering.
1	Place teem3 at position "1" and enter the Count Enabled state. If the count of TCRm3 registers is started in the counting Enabled state, it enters interval timer mode (see Table 6-6 of "Start Timing of the Counter 6.5.2").

TSHm1	The operation of the high 8-bit timer when Channel 1 is in 8-bit timer mode enable (start) triggering
0	No triggering.
1	Place teem1 at position "1" and enter the Count Enabled state. If the count of TCRm1 registers is started in the counting Enabled state, it enters interval timer mode (see Table 6-6 of "Start Timing of 6.5.2 Counters").

TSmn	The operation of channel n enable (start) triggering
0	No triggering.
1	Place the TEmn position "1" and enter the Count Enabled state. The start of counting the TCRmn registers in the count-Enabled state varies depending on each operating mode (see Table 6-6 of "Start Timing of Counters 6.5.2"). In Channel 1 and Channel 3 are 8-bit timer modes, TSm1 and TSm3 are 8 lows the operation of the bit timer enable (start) triggering.

Note: 1 Bit15~12, 10, 8~4 must be set to "0".

- When switching from the function of not using the TImn pin input to the function of using the TImn pin input, from setting the timer mode register mn (TMRmn) to setting TSmn (TSHm1, TSHm3) position "1", the following period of waiting is required:

When the TImn pin noise filter is active (TNFENmn=1): 4 operating clocks (f_{MCK}).

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

Note: 1. The read value of the TSm register is always "0".

2.m: unit number (m=0,1)n: channel number (m=0: n=0~3, m=1: n=0~7).

6.4.3 The basic rules for the 8-bit timer to operate the function (limited to Channel 1 and Channel 3 of Unit 0).

The 8-bit timer operation function is the function of using the channel of a 16-bit timer as the channel of two 8-bit timers. Only Channel 1 and Channel 3 can run the function using the 8-bit timer, and several rules need to be observed when using it.

The basic rules for the 8-bit timer operation function are as follows.

- 1) The 8-bit timer operation function is only available for Channel 1 and Channel 3.
- 2) When used as an 8-bit timer, place the SPLIT position "1" of the timer mode register mn (TMRmn).
- 3) The high 8-bit timer can be used as an interval timer function.
- 4) At start operation, the high 8-bit timer outputs INTMm1H/INTTMm3H (interrupt) (the same operation as MDmn0 bit "1").
- 5) The choice of operating clock for a high 8-bit timer depends on the setting of the CKSmn1 bit and CKSmn0 bits of the low TMRmn register.
- 6) For high 8-bit timers, the channel operation is started by operating the TSHm1/TSHm3 bits, and the channel operation is stopped by operating the TTHm1/TTHm3 bits. The status of the channel can be confirmed by teh1/TEHm3 bits.
- 7) The operation of the low 8-bit timer depends on the setting of the TMRmn register, and there are 3 functions that support the operation of the low 8-bit timer:
 - Interval timer function
 - External event counter functionality
 - Delay counting function
- 8) For low 8-bit timers, the channel operation is started by operating the TSm1/TSm3 bits, and the channel operation is stopped by operating the TTm1/TTm3 bits. The status of the channel can be confirmed by the TEm1/TEm3 bits.
- 9) When running on a 16-bit timer, the operation of TSHm1/TSHm3/TTHm1/TTHm3 bits is invalid. Operate Channels 1 and 3 by operating TSm1/TSm3 bits and TTm1/TTm3 bits. TeHm3 bit and TEHm1 bit unchanged.
- 10) The 8-bit timer function cannot use the linkage operation function (single trigger pulse, PWM, and multiple PWM).

Note: m: Unit number (m=0)n: Channel number (n=1, 3).

6.5 The operation of the counter

6.5.1 Count clock (f_{TCLK}).

The general-purpose timer unit's counting clock (f_{TCLK}) selects any of the following clocks by the CCSmn bit of the timer mode register mn (TMRmn).

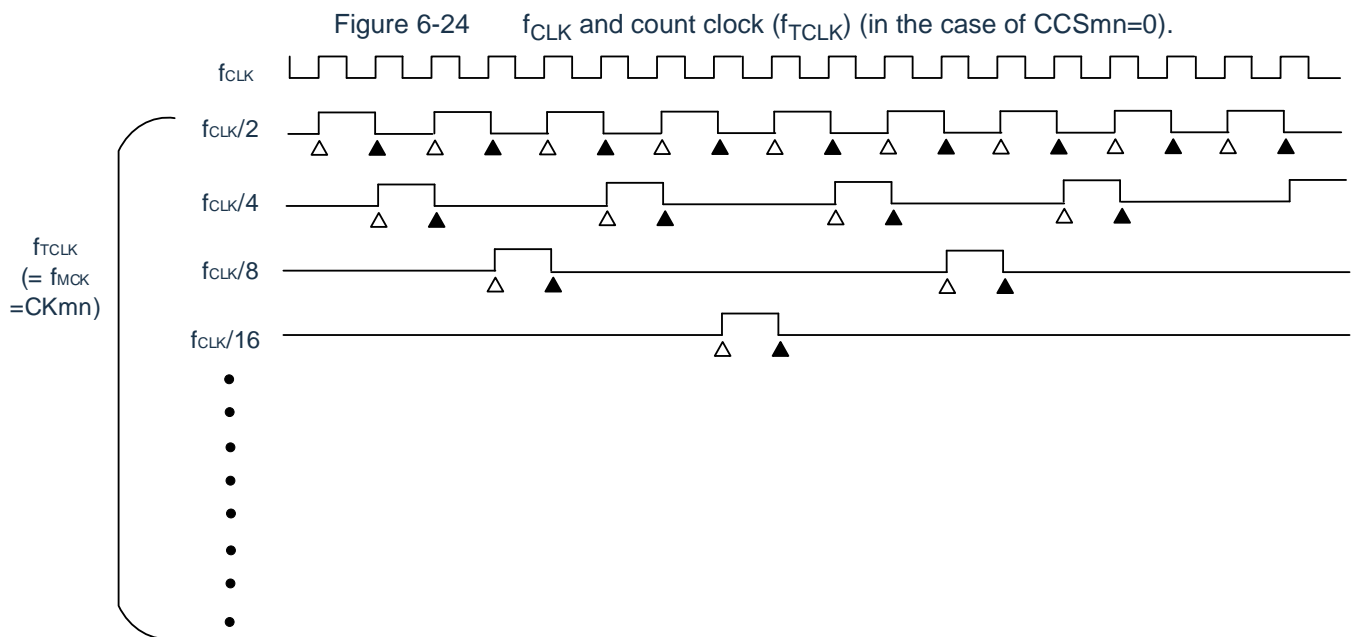
- CKSmn0 bits and CKSmn1 bits specify the operating clock (f_{MCK}).
- The effective edge of the input signal at the TImn pin

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

(1) Select the case (CCSmn=0) specified by the CKSmn0 bit and CKSmn1 bits specified for the operating clock (f_{MCK}).

Depending on the setting of the timer clock select register m (TPSm), the count clock (f_{TCLK}) is $f_{CLK} \sim f_{CLK}/2^{15}$. However, when the division of f_{CLK} is selected, the clock selected by the TPSm register is a signal that starts with only 1 f_{CLK} cycle high starting on the rising edge. When f_{CLK} is selected, it is fixed to high.

To achieve synchronization with the f_{CLK} , the timer count register mn (TCRmn) delays the count by 1 f_{CLK} clock starting from the rising edge of the count clock. It is called "counting on the rising edge of the counting clock" for convenience.



Remarks: 1 Δ: Counts the rising edge of the clock

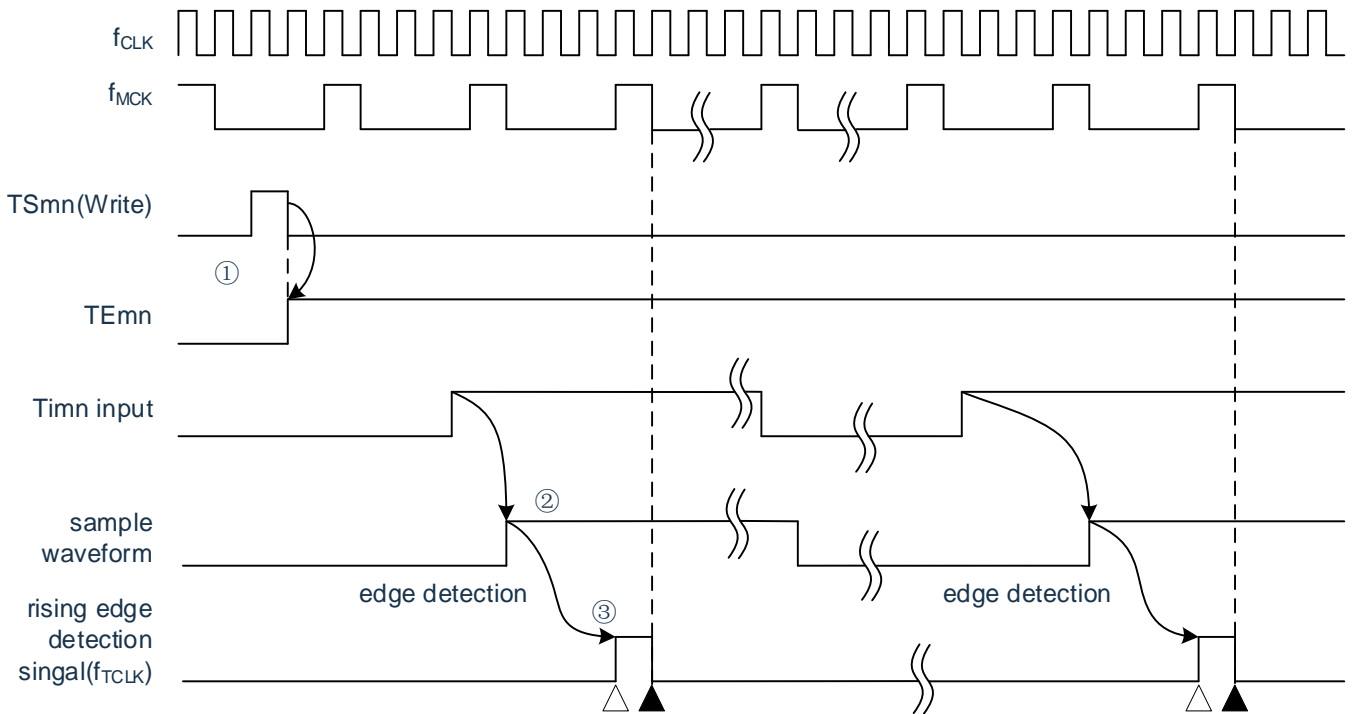
▲: Synchronization, increment/decrement of the counter

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

- (2) Select the case where the effective edge of the input signal of the TImn pin is selected (CCSmn=1).

The counting clock (f_{TCLK}) is a signal that detects the effective edge of the input signal of the TImn pin and synchronizes with the rising edge of the next f_{MCK} . In fact, this is a signal that delays the TImn pin by 1 to 2 f_{MCK} clocks (when using a noise filter, a delay of 3 to 4). f_{MCK} clocks). In order to obtain synchronization with f_{CLK} , the timer count register mn (TCRmn) delays 1 f_{CLK} from the rising edge of the counting clock. For convenience, it is called "counting at the effective edge of the input signal at the TImn pin".

Fig. 6-25 count clock (f_{TCLK}) (CCSmn=1, without the use of a noise filter).



- ① Start the operation of the timer by placing the TSmn position bit and wait for a valid edge of the TImn input.
- ② The rising edge of the TImn input is sampled by f_{MCK} .
- ③ The rising edge of the sampled signal is detected, and the heartbeat signal (counting clock) is output.

Remarks: 1 Δ : Counts the rising edge of the clock

\blacktriangle : Synchronization, increment/decrement of the counter

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

f_{MCK} : The operating clock of channel n

3. The same waveforms are also measured for the measurement of input pulse intervals, the measurement of high and low levels of the input signal, the delay counter, and the TImn input of the single-trigger pulse output function.

6.5.2 The start timing of the counter

By placing the timer channel starting the TSmn position bit of register m(TSm), the timer count register mn (TCRmn) enters the run Enabled state.

The operation from the count Enabled state to the timer count register mn (TCRmn) begins to count as shown in Table 6-6.

Table 6-6 Runs from the count enable state to the timer count register mn (TCRmn) starting to count

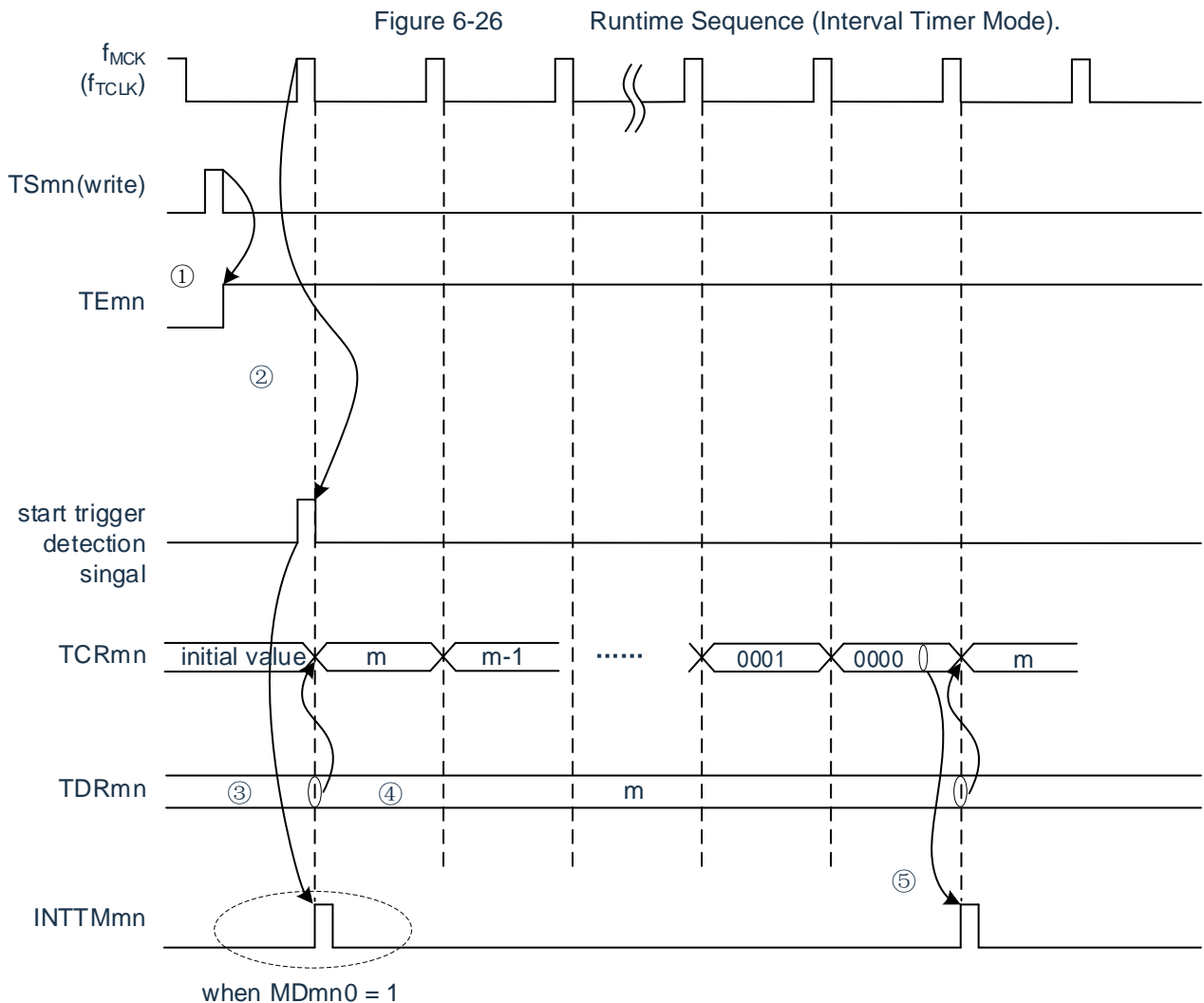
The operating mode of the timer	Run after the TSmn position "1"
• Interval timer mode	No action is taken from the time the start of detection (TSmn=1) is generated until the count clock is generated. The value of the TDRmn register is loaded into the TCRmn register by the first count clock and the count is decremented by subsequent count clocks (see "Operation of the 6.5.3(1) Interval Timer Mode").
• Event counter pattern	Load the value of the TDRmn register into the TDRmn register by writing "1" to the TSmn bit. If the input edge of TImn is detected, the count is decremented by a subsequent counting clock. (See "6.5.3(2).") Run of the event counter pattern").
• Capture mode	No action is taken from the time it is detected and triggered until the counting clock is generated. "0000H" is loaded into the TCRmn register by the first counting clock and counted incrementally by subsequent counting clocks (refer to the operation of "6.5.3(3) capture mode (interval measurement of input pulses)").
• Single count mode	By writing "1" to the TSmn bit in the state where the timer stops running (TEmn=0), you enter the one that starts triggering, etc Pending state. No action is taken from the time it is detected and triggered until the counting clock is generated. The value of the TDRmn register is loaded into the TCRmn register by the first counting clock, and subsequent meters are passed Counts the number of clocks to decrement (see "6.5.3(4) Operation of single count mode").
• Capture & Single Count mode	By writing "1" to the TSmn bit in the state where the timer stops running (TEmn=0), you enter the one that starts triggering, etc Pending state. No action is taken from the time it is detected and triggered until the counting clock is generated. "0000H" is loaded into the TCRmn register by the first counting clock and is performed by subsequent counting clocks Increment count (see "6.5.3(5) Capture & Single Count Mode Run (Measurement of High Level Width)").

6.5.3 The operation of the counter

The counter operation for each mode is described below.

(1) Operation of interval timer mode

- (1) By writing "1" to the TSmn bit, enter the run Enabled state ($TE_{mn}=1$). The timer count register mn (TCR_{mn}) remains at its initial value until a count clock is generated.
- (2) Generate a start trigger signal by allowing the first counting clock (f_{MCK}) after running.
- (3) When the MD_{mn0} bit is "1", the $INTM_{mn}$ is generated by starting the trigger signal.
- (4) Load the value of the timer data register mn (TDR_{mn}) into the TCR_{mn} register by allowing the first counting clock after running, and start counting in interval timer mode.
- (5) If the TCR_{mn} register is decremented to "0000H", the $INTM_{mn}$ is generated by the next count clock (f_{MCK}), and the timer data register mn is generated. The value of (TDR_{mn}) continues counting after loading into the TCR_{mn} register.



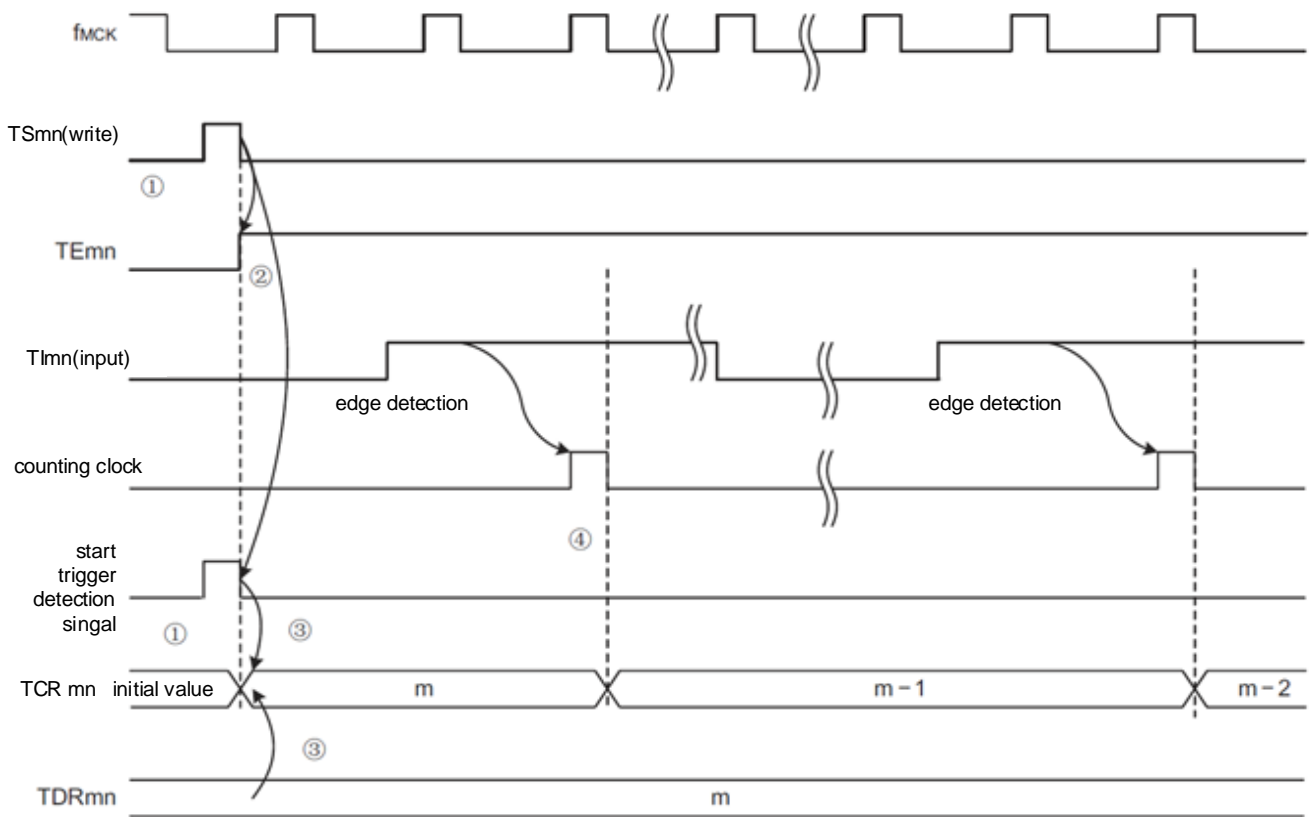
Note: Because the first count clock cycle runs after the TS_{mn} bit is written and the start of the count is delayed before the count clock is generated, an error of up to 1 clock cycle is generated. In addition, if information about starting counting timing is required, the MD_{mn0} position is "1" so that an interrupt can be generated at the start of counting.

Note: f_{MCK} , start trigger heartbeat, and INTMmn are synchronized to f_{CLK} and are valid within 1 clock.

(2) The operation of the event counter pattern

- (1) During the run stop state ($TEmn=0$), the timer count register mn ($TCRmn$) maintains the initial value.
- (2) By writing "1" to the $TSmn$ bit, enter the run Enabled state ($TEmn=1$).
- (3) Load the value of the timer data register mn ($TDRmn$) into the $TCRmn$ register while both the $TSmn$ bit and the $TEmn$ bit become "1" and start counting.
- (4) Thereafter, at the effective edge of the $TImn$ input, the value of the $TCRmn$ register is counted down by counting the clock.

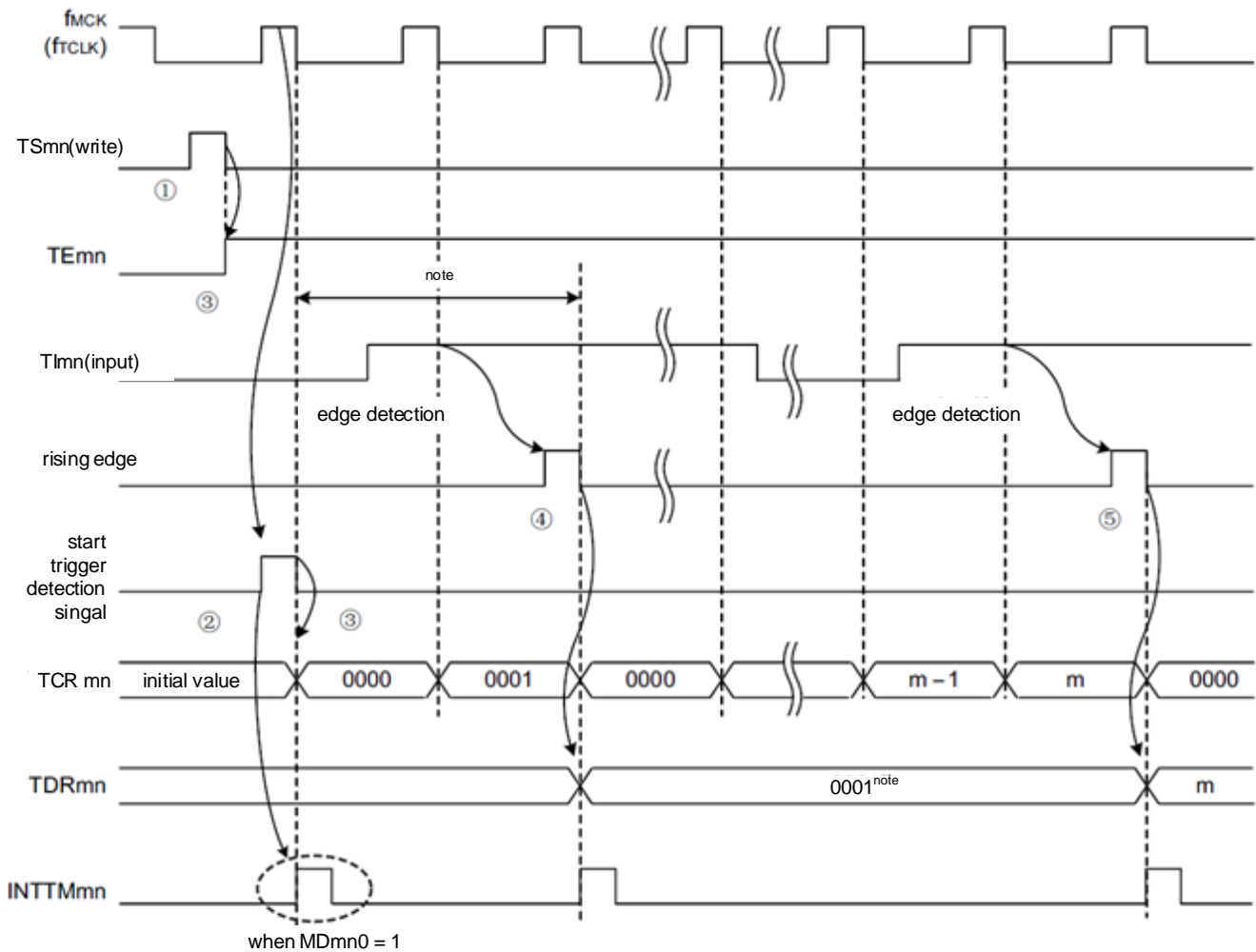
Figure 6-27 Runtime Sequence (Event Counter Pattern).



Note: This is the timing when no noise filter is used. If a noise filter is used, edge detection is delayed by 2 f_{MCK} cycles (3 to 4 cycles in total) from the $TI mn$ input. The 1-cycle error is due to the $TI mn$ input being out of sync with the counting clock (f_{MCK}).

- (3) Operation of capture mode (interval measurement of input pulses).
 - (1) By writing "1" to the TSmn bit, enter the run Enabled state (TEmn=1).
 - (2) The timer count register mn (TCRmn) maintains the initial value until the count clock is generated.
 - (3) Generate a start trigger signal by allowing the first counting clock (fMCK) after running. Then, load "0000H" into the TCRmn register and start counting in capture mode (when MDmn0 bit is "1", intTMmn is generated by starting the trigger signal) .
 - (4) If a valid edge of the TImn input is detected, the value of the TCRmn register is captured to the TDRmn register and an INTTMmn interrupt is generated. The captured value at this point is meaningless. The TCRmn register continues counting starting at "0000H".
 - (5) If a valid edge of the next TImn input is detected, the value of the TCRmn register is captured to the TDRmn register and an INTTMmn interrupt is generated.

Figure 6-28 Runtime Sequence (Capture Mode: Interval Measurement of Input Pulses).



Note: When the clock is entered to TImn (with a trigger) before starting, the count starts by detecting the trigger even if no edge is detected, so the capture value at the first capture ((4)) is not a pulse interval (in this case, 0001: 2 clock intervals) and must be ignored.

Note: Because the first count clock cycle runs after the TSmn bit is written and the start of the count is delayed before the count clock is generated, an error of up to 1 clock cycle is generated. In addition, if information about starting counting timing is required, the MDmn0 position is "1" so that an interrupt can be generated at the start of

counting.

Note: This is the timing when no noise filter is used. If a noise filter is used, edge detection is

delayed by 2 f_{MCK} cycles (3 to 4 cycles in total) from the TImn input. The 1-cycle error is due to the TImn input being out of sync with the counting clock (f_{MCK}).

(4) Operation of single count mode

(1) By writing "1" to the TSmn bit, enter the run Enabled state ($TE_{mn}=1$).

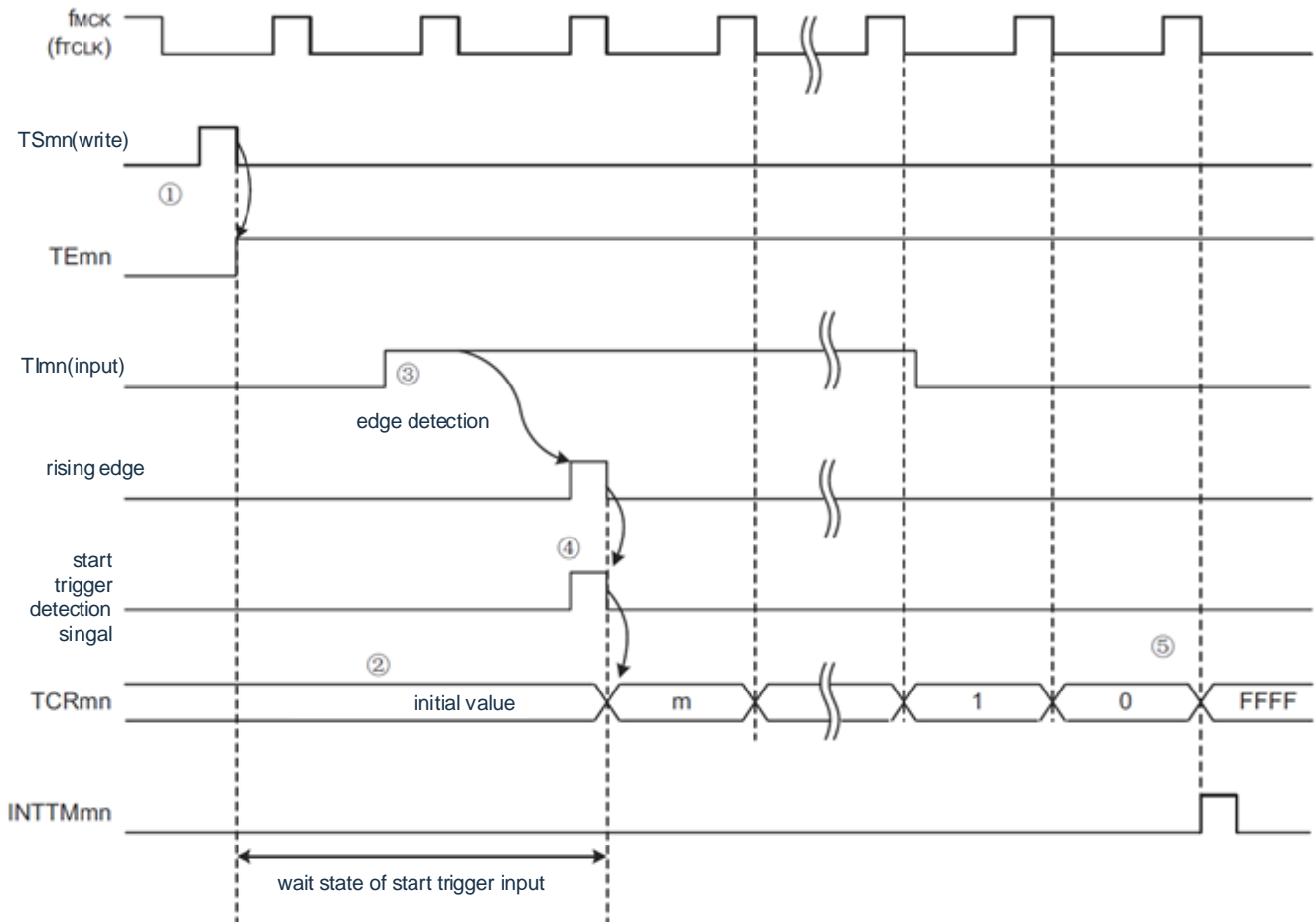
(2) The timer count register mn (TCRmn) remains at its initial value until a start trigger signal is generated.

(3) Detect the rising edge of the TImn input.

(4) Load the value of the TDRmn register (m) into the TCRmn register after generating the start trigger signal, and start counting.

(5) When the TCRmn register decreases the count to "0000H", an INTTMmn interrupt is generated, and the value of the TCRmn register becomes "FFFFH", stopping the count.

Figure 6-29 Runtime Sequence (Single Count Mode).



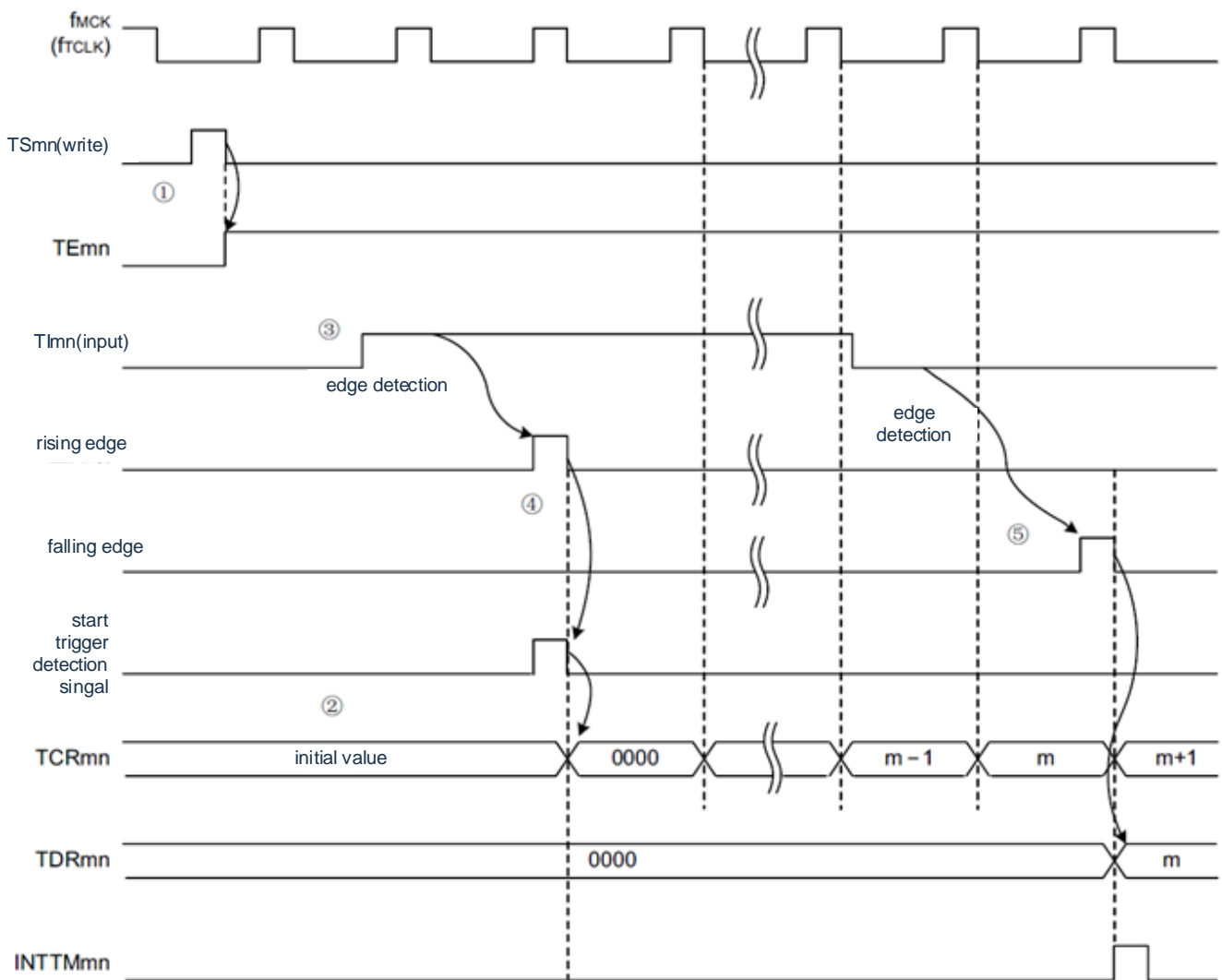
Note: This is the timing when no noise filter is used. If a noise filter is used, edge detection is

delayed by 2 f_{MCK} cycles (3 to 4 cycles in total) from the TImn input. The 1-cycle error is due to the TImn input being out of sync with the counting clock (f_{MCK}).

(5) Capture & Run of Single Count mode (measurement of high level width).

- (1) Write "1" in the TSmn bit of register m(TSm) through the given timer channel and enter the run Enabled state (TEmn=1).
- (2) The timer count register mn (TCRmn) remains at its initial value until a start trigger signal is generated.
- (3) Detect the rising edge of the TImn input.
- (4) Load "0000H" into the TCRmn register after generating the start trigger signal, and start counting.
- (5) If the falling edge of the TImn input is detected, the value of the TCRmn register is captured to the TDRmn register and an INTTmn interrupt is generated.

Figure 6-30 Runtime Sequence (Capture & Single Count Mode: Measurement of High Level Width).

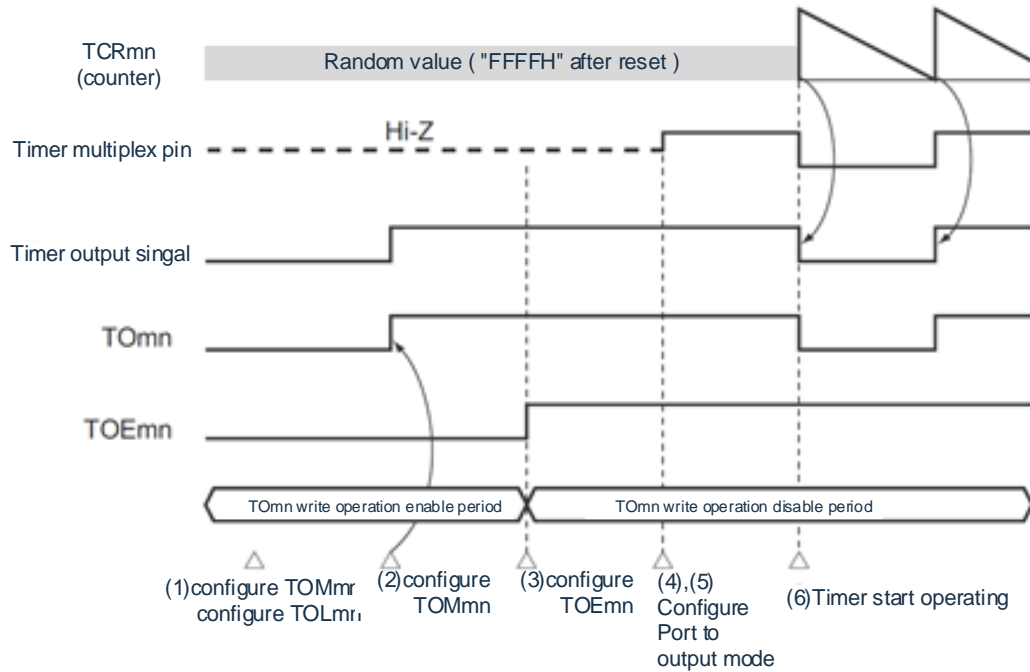


Note: This is the timing when no noise filter is used. If a noise filter is used, edge detection is delayed by 2 f_{MCK} cycles (3 to 4 cycles in total) from the TImn input. The 1-cycle error is due to the TImn input and the count clock (f_{MCK}) are out of sync.

6.6.2 Output setting of the TOMn pin

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

Figure 6-32 The state change from setting the output of the timer to starting operation



(1) Set the operating mode of the timer output.

- TOMmn bits (0: master channel output mode, 1: slave channel output mode).
- TOLmn bits (0: positive logic output, 1: negative logic output).

(2) Set the timer output signal to the initial state by setting the timer output register m(TOm).

(3) Write "1" to the TOEmn bit, allowing the timer output (it is forbidden to write the TOM register).

(4) Set the port as a digital input/output through the port mode control register (PMCxx) (refer to the register of the "6.3.15 control timer input/output pin port function").

(5) Set the input/output of the port to the output (refer to "Registers for controlling the input/output pin port function of the timer in 6.3.15").

(6) Enable the timer to run (TSmn=1).

Note: m: unit number (m=0,1)n: channel number (when m=0: n=0~3, m=1: n=0~7).

6.6.3 Considerations for channel output operation

- 1) About the configuration changes of the TOM, TOEm, TOLm, and TOMm registers in the operation of the timer

The operation of the timer (the operation of the timer count register mn (TCRmn) and the timer data register mn (TDRmn)) and the TOMn output circuit are independent of each other. Thus, the timer output register m (TOM), the timer output enable register m (TOEm), and the timer output level register m (TOLm) setpoint change does not affect the operation of the timer, you can change the setpoint during the timer operation. However, in order to output the expected waveform from the TOMn pin during the operation of each timer, it must be set to the value of the register setting content example for each run shown in 6.8 and 6.9.

If you change the setpoint of the TOEm registers and TOLm registers other than the TOM registers before and after generating the timer interrupt (INTTMmn) signal for each channel, it is based on whether the timer interrupt (INTTMmn) is generated. Whether the signal changes before or after generation, the waveform output of the TOMn pin may be different.

Remarks: m: Unit number (m=0,1)n: Channel number (when m=0: n=0~3, m=1: n=0~7).

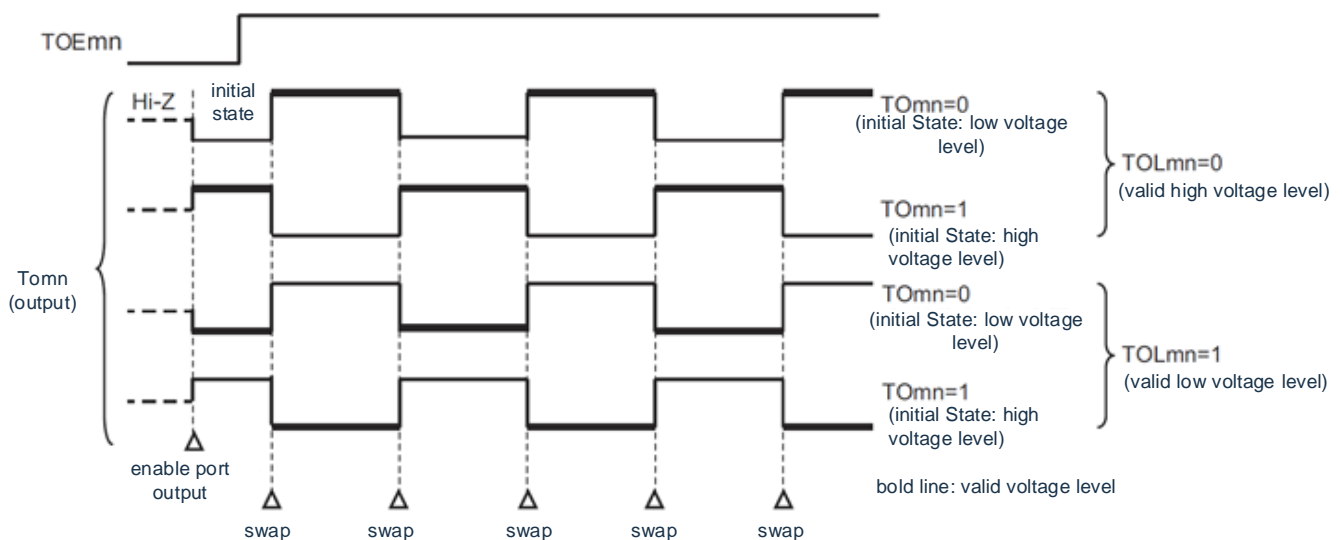
- (2) The initial level of the TOMn pin and the output level after the timer starts running

Write the timer output register m (TOM) before allowing the port output and in the state where the timer output (TOEmn=0) is disabled, and when the timer output Enabled state (TOEmn=1) is set after changing the initial level. The TOMn pin output level changes as follows.

- (a) When the operation starts in the main control channel output mode (TOMmn=0).

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

Figure 6-33 TOMn pin when alternating outputs (TOMmn=0) is 33



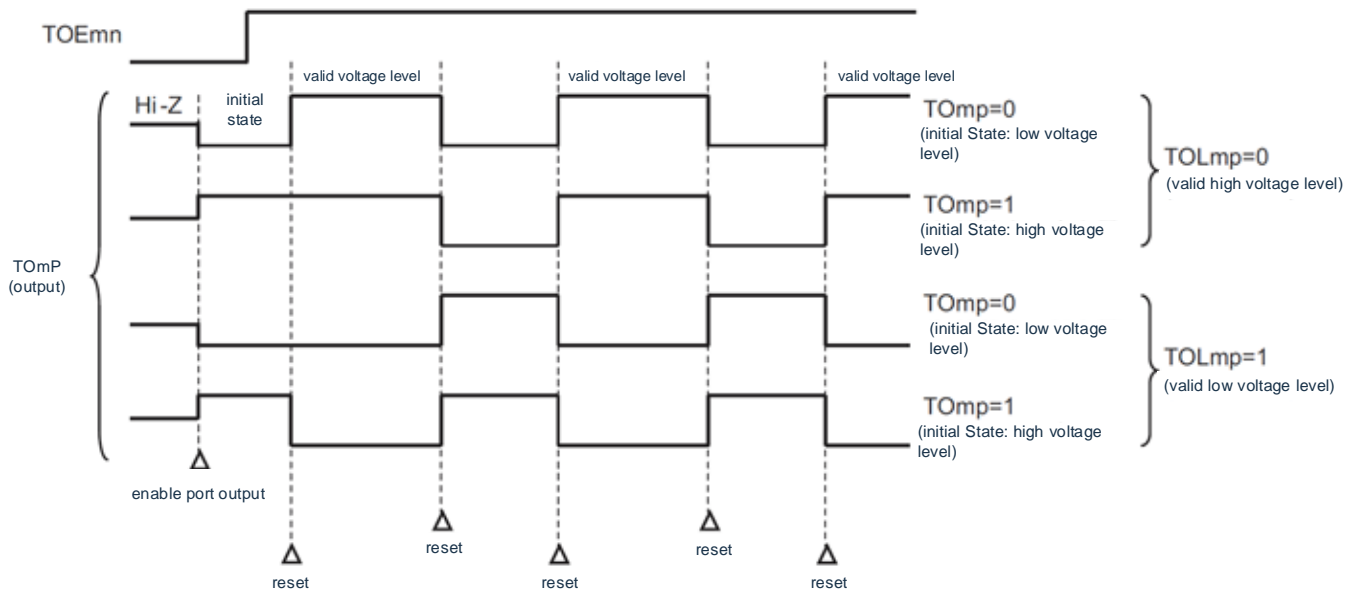
Remarks: 1 Alternate: The output state of the inverting TOMn pin.

2.m: unit number (m=0,1)n: channel number (m=0: n=0~3, m=1: n=0~7).

- (b) When running in slave channel output mode (TOMmn=1) (PWM output).

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

Figure 6-34 TOMn pin at PWM output (TOMmn=1).



Remarks: 1 Assert: The output signal of the TOMp pin changes from invalid to active.

Reset: The output signal of the TOMp pin changes from the active level to the invalid level.

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

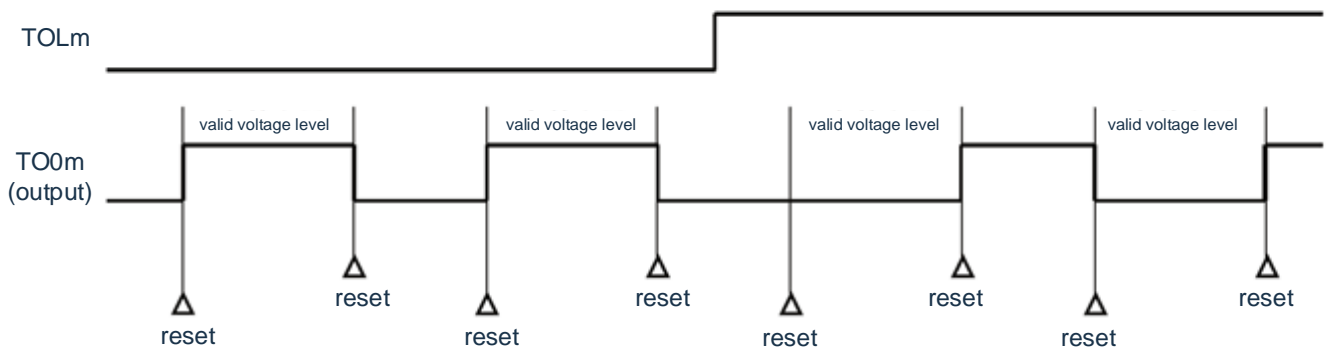
(3) TOMn pin change regarding slave channel output mode (TOMmn=1).

(a) Change the setting of the timer output level register m (TOLm) during timer operation

If you change the setting of the TOLm register during timer operation, the setting is valid when the TOMn pin change condition is generated. The output level of the TOMn pin cannot be changed by overwriting the TOLm registers.

When the TOMmn bit is "1", the run when changing the value of the TOLm register in the timer run (TEmn=1) is as follows.

Figure 6-35 When changing the contents of the TOLm register during timer operation



Remarks: 1 Assert: The output signal of the TOMn pin changes from invalid to effective.

Reset: The output signal of the TOMn pin changes from the active level to the invalid level.

2.m: unit number (m=0,1)n: channel number (m=0: n=0~3, m=1: n=0~7).

(b) Set/reset timing

To achieve 0% and 100% output at PWM output, the TOMn pin/TOMn at the master channel timer interrupt (INTTMmn) will be generated through the slave channel. The set timing of the bits delays by 1 count clock.

When a placement condition and a reset condition occur at the same time, the reset condition takes precedence.

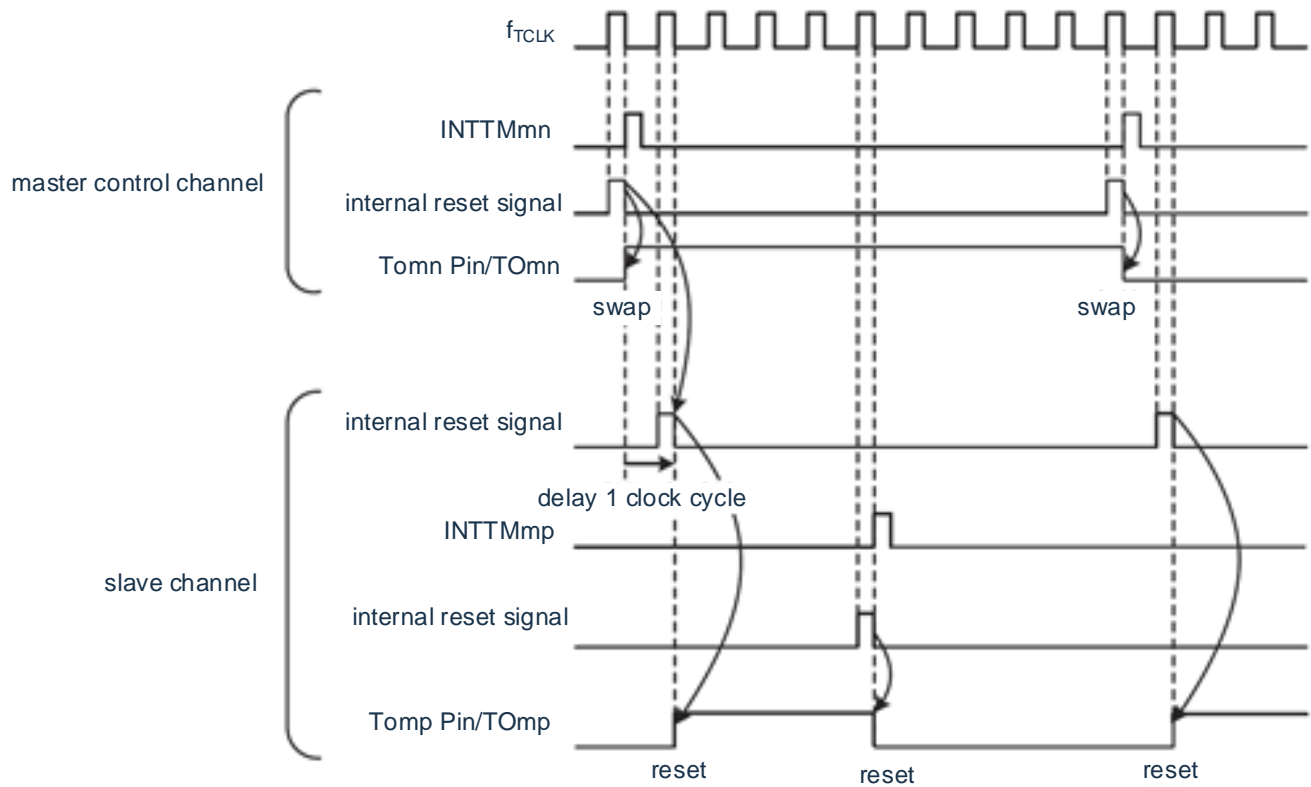
The position/reset operation state when setting the master/slave channel according to the following method is shown in Figure 6-35.

Main control channel: TOEmn=1, TOMmn=0, TOLmn=0

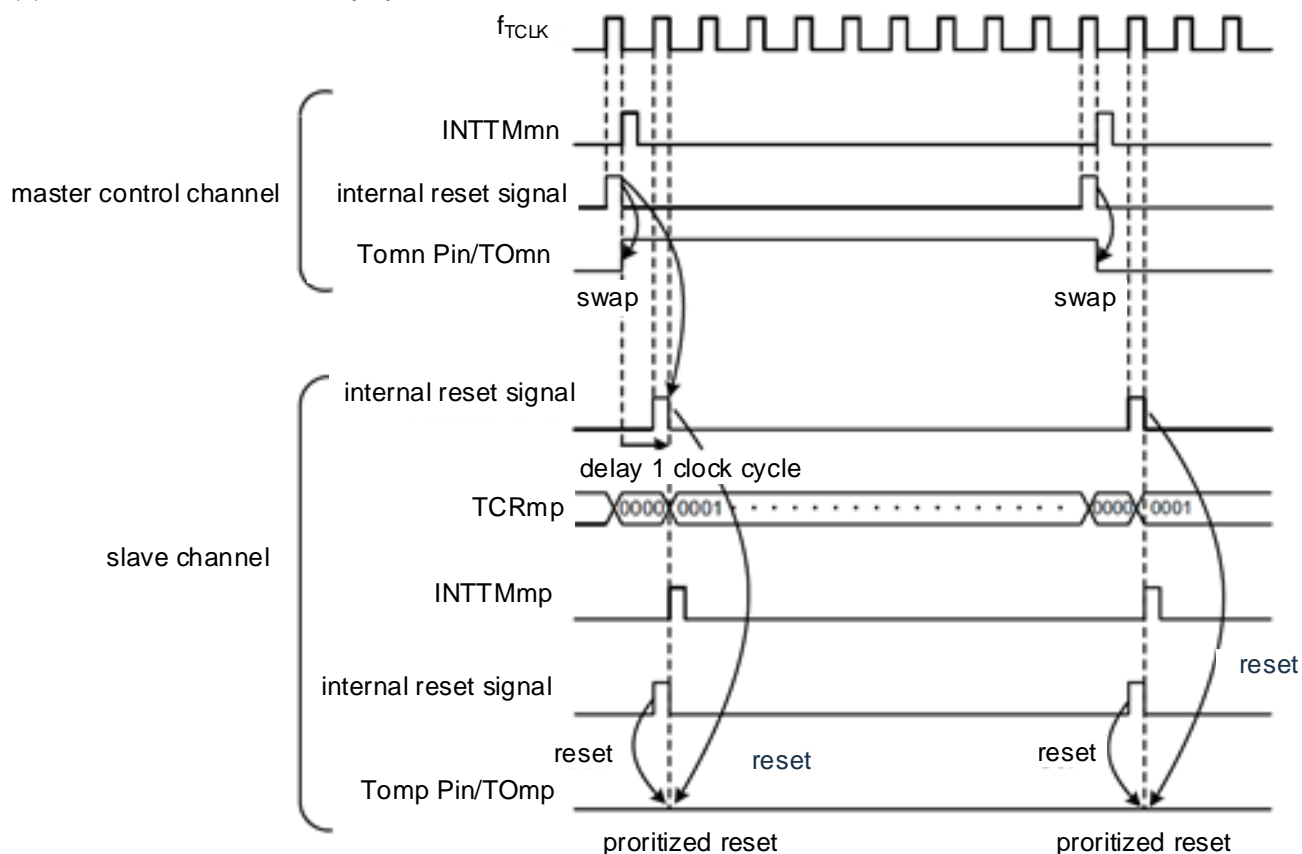
Slave channels: TOEmp=1, TOMmp=1, TOLmp=0

Figure 6-36 Assert/Reset Timing Operation Status

(1) Basic runtime order



(2) 0% runtime order of duty cycle



Note 1 Internal reset signal: Reset/alternate signal at the TOMn pin

Internal position signal: The position signal of the TOMn pin

2.m: Unit number ($m=0,1$).

n: channel number $m=0$: $n=0\sim3$, $m=1$: $n=0\sim7$ (main control channel: $n=0, 2, 4, 6$).

p: Slave channel number

$n=0$: $p=1, 2, 3$

$n=2$: $p=3$

6.6.4 one-time operation of the TOnm bit

Like the timer channel start register m (TSm), the timer output register m (TOnm) has all channel set bits (TOnm). This allows the TOnm bits of all channels to be manipulated at once.

Figure 6-37 Example of one-time operation of the TO0n bit

Before writing

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

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

The data to write

0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

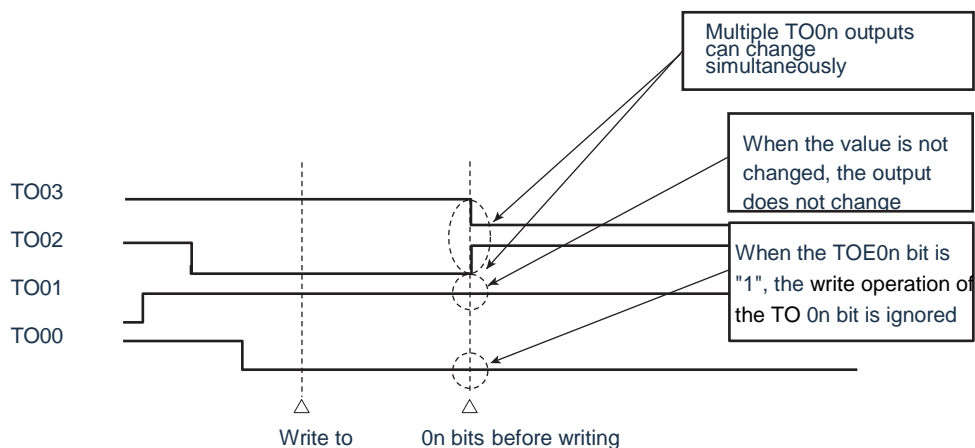
After writing

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

Only TOnm bits with TOEmn bits as "0" can be written, ignoring write operations for TOnm bits with TOEmn bits as "1".

TOnm (channel output) with TOEmn bit "1" is not affected by write operations, even the write TOnm bit is ignored, and output changes caused by timer operation are normal.

Figure 6-38 state of the TO0n pin when the TO0n bit is operated at one time



Note: m: unit number (m=0,1)n: channel number (when m=0: n=0~3, m=1: n=0~7).

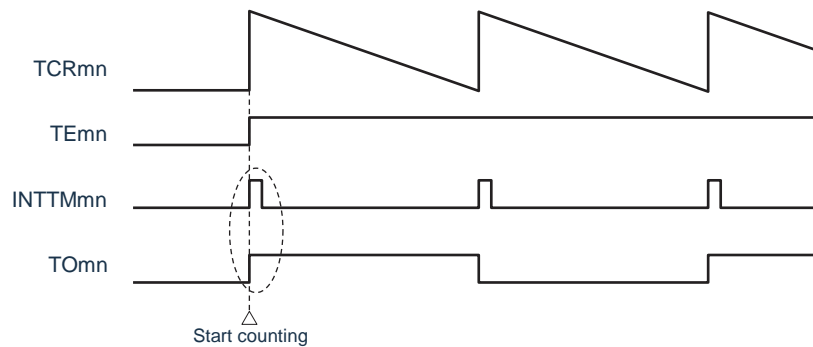
6.6.5 About the timer interrupt and TOmn pin output when starting to count

In interval timer mode or capture mode, the MDmn0 bit of the timer mode register mn (TMRmn) is the bit that sets whether a timer interrupt occurs at the start of counting.

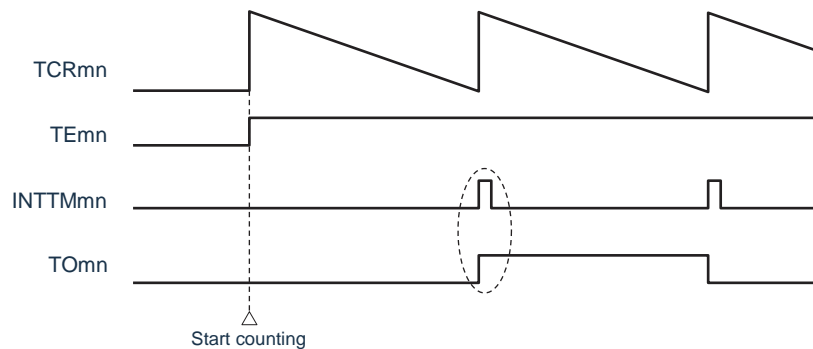
When the MDmn0 bit is "1", the start timing of the count is known by generating a timer interrupt (INTTMmn). In other modes, timer interrupts and TOmn outputs at start counting are not controlled. An example of operation when set to interval timer mode (TOEmn=1, TOMmn=0) is shown below.

Figure 6-39 An example of a timer interrupt and a TOmn output when counting starts

(a) The case where the MDmn0 bit is "1"



(b) The case where the MDmn0 bit is "0"



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

When the MDmn0 bit is "0", no timer interrupt (INTTMmn) is output at the start of counting and TOmn does not change, but INTTMmn is output after counting 1 cycle and TOmn performs alternate outputs.

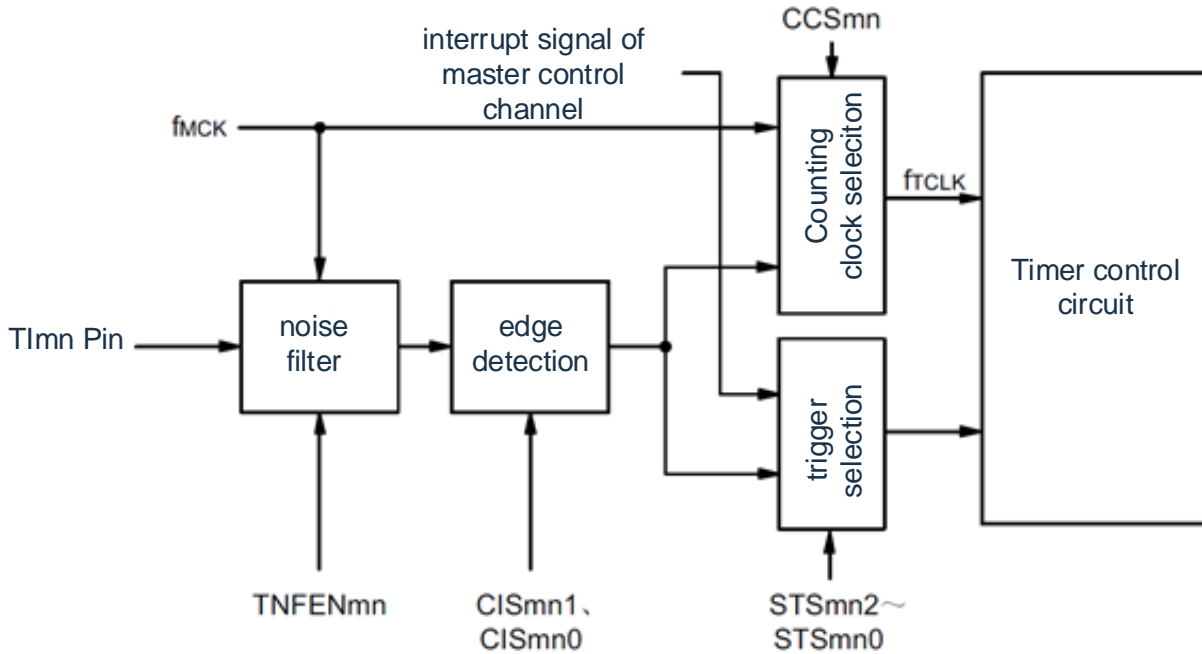
Remarks: m: Unit number (m=0,1)n: Channel number (when m=0: n=0~3, m=1: n=0~7).

6.7 Control of the timer input (TImn).

6.7.1 Structure of the TImn pin input circuit

Signals from the timer input pins are fed into the timer control circuitry through a noise filter and edge detection circuitry. For pins that need to be noise-cancelling, the corresponding pin noise filter must be set to active. The structure of the input circuit is as follows.

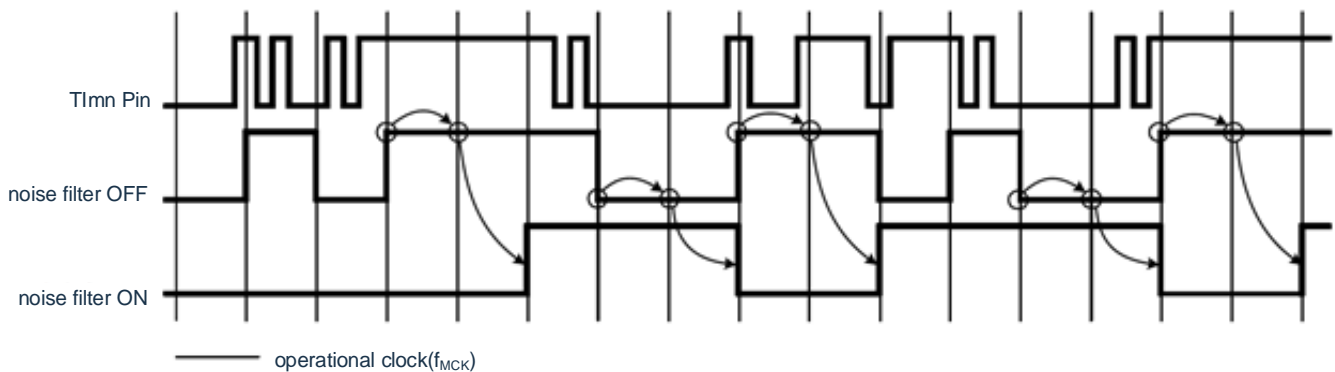
Figure 6-40 input circuit structure



6.7.2 Noise filters

When the noise filter is invalid, synchronization is performed only by the operating clock (f_{MCK}) of channel n; When the noise filter is active, the two clocks are detected to be consistent after synchronization through the operating clock (f_{MCK}) of channel n. In the case of a noise filter ON or OFF at the TM4mn input pin, the waveform after passing through the noise filter circuit is shown below.

Fig. 6-40 Sampled waveform of the TImn input pin in the case of noise filter ON or OFF



Note: The input waveform of the TImn pin is used to illustrate the operation of the noise filter ON or OFF. In actual use, the input must be made according to the TImn input level width shown in the "AC characteristics".

6.7.3 Considerations when operating channel input

When set to not use the timer input pin, the noise filter circuit is not provided with an operating clock. Therefore, the following waiting times are required from the channel operation set to use the timer input pin to the corresponding channel operation of the set timer input pin to enable triggering.

(1) When the noise filter is OFF

If the timer mode register mn (TMRmn) is bit12 (CCSmn), bit9 (STSmn1) and Bit8 (STSmn0) will be any bit in a state where it is all "0", and it must pass through at least 2 operating clocks (f_{MCK}). After the cycle, the timer channel will be run to start the operation of the register (TSM) to enable trigger setting.

(2) When the noise filter is ON

If the timer mode register mn (TMRmn) is bit12 (CCSmn), bit9 (STSmn1) and Bit8 (STSmn0) will be any bit in a state of "0" and must pass at least 4 operating clocks (f_{MCK}). After the cycle, the timer channel will be run to start the operation of the register (TSM) to enable trigger setting.

6.8 Stand-alone channel operation of the universal timer unit

6.8.1 Operation as an interval timer/square wave output

(1) Interval timer

It can be used as a reference timer for generating INTMmn (timer interrupts) at regular intervals. The interrupt generation period can be calculated using the following calculation:

The production cycle of INTMmn (Hours interrupt) = counting clock circumference Period x (setpoint +1 for TDRmn).

(2) Operation as a square wave output

TOMn alternates outputs while generating INTMmn, outputting a square wave with a duty cycle of 50%.

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

- Square wave period of TTON output = counting clock cycle (xsetpoint +1 for TDRmn) × 2

- Square wave frequency of TTON output = count clock rate / {(TDRmn.) Set the value +1) × 2}

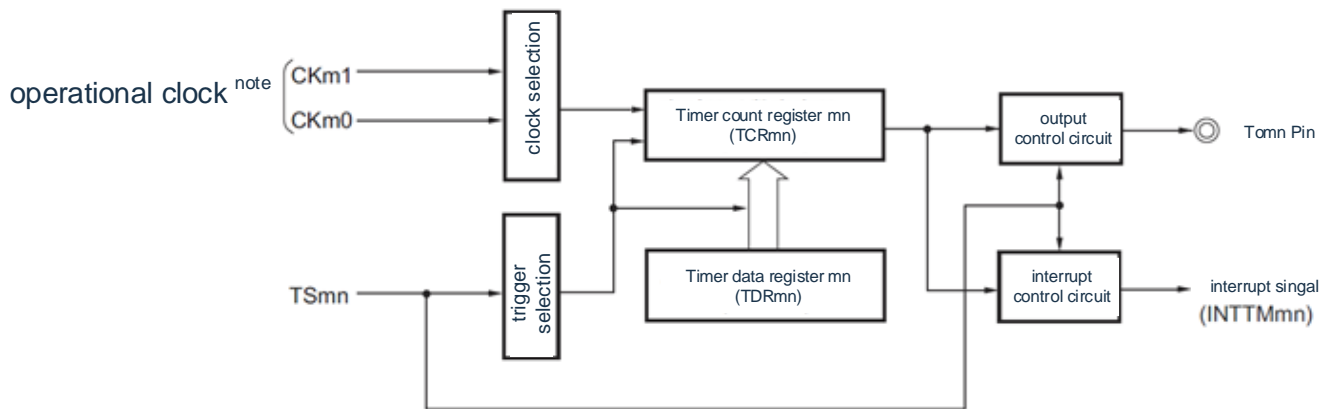
In interval timer mode, the timer count register mn (TCRmn) is used as a decrement counter.

After setting the channel start trigger bit (TSmn, TSHm1, TSHm3) of the timer channel start register m (TSM), pass the first A counting clock loads the value of the timer data register mn (TDRmn) into the TCRmn register. At this point, if the MDmn0 bit of the timer mode register n (TMRmn) is "0", intTMmn is not output and TOMn also does not have alternating outputs. If the MDmn0 bit of the TMRmn register is "1", intTMmn is output and TOMn is alternately output. The TCRmn register then decrements the count through the counting clock.

If TCRmn becomes "0000H", intTMmn is output via the next counting clock and TOMn is alternately output. At the same time, load the value of the TDRmn register into the TCRmn register again. After that, the same run continues.

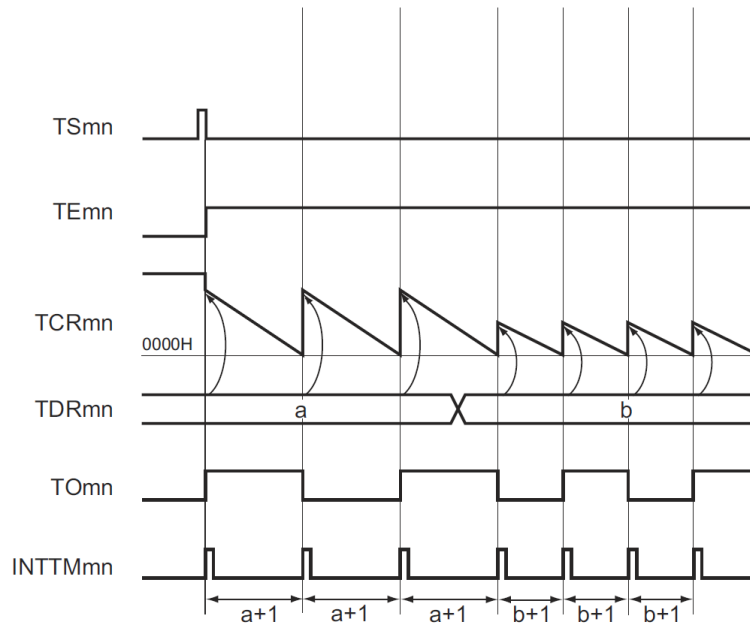
The TDRmn register can be rewritten at any time, and the value of the rewritten TDRmn register is valid from the next cycle.

Figure 6-41 basic timing example of operation as an interval timer/square wave output (MDmn0=1).



Note: Clocks can be selected from CKm0, CKm1, CKm2, and CKm3 on channel 1 and 3.

Figure 6-42 basic timing example of running as an interval timer/square wave output (MDmn0=1).

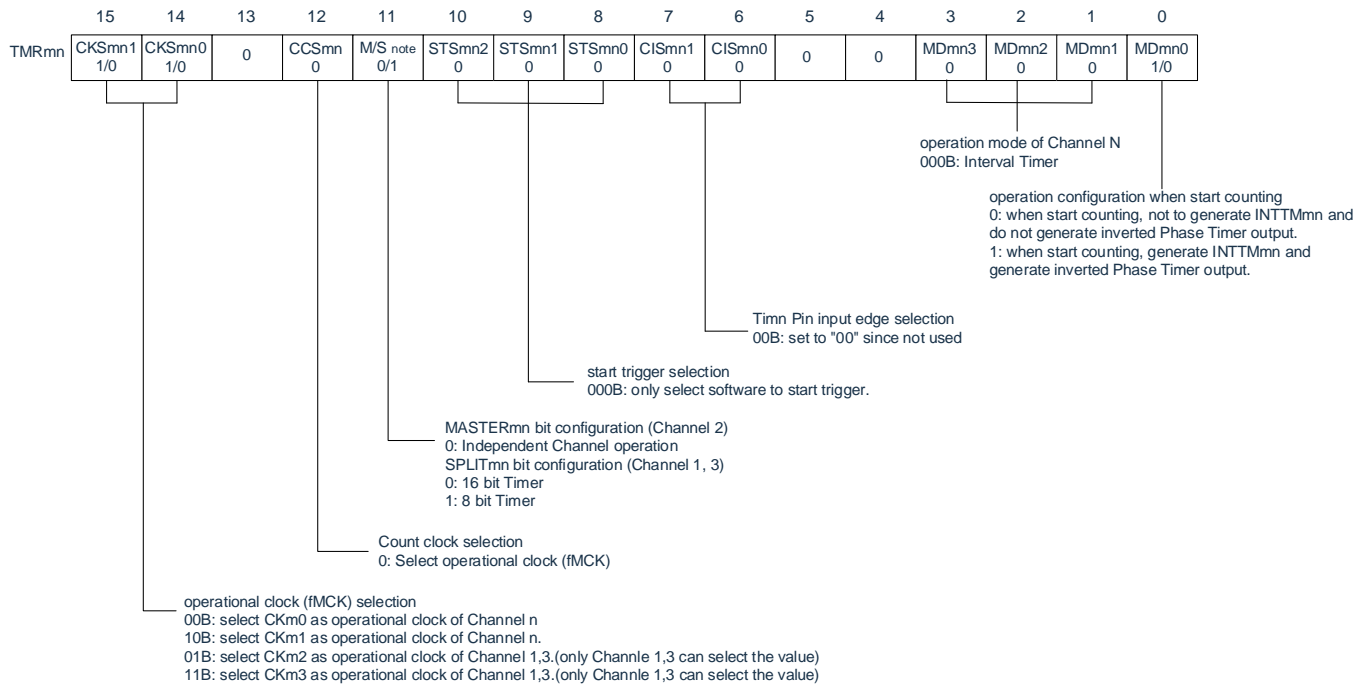


Note: 1.m: unit number (m=0,1)n: channel number (when m=0: n=0~3, m=1: n=0~7).

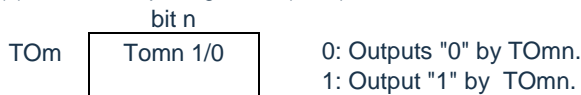
2. TSmn : The bitn of the timer channel start register m(TSm).
- TEmn : The timer channel enable bitn of the status register m(TEm).
- TCRmn : Timer count register mn (TCRmn).
- TDRmn : Timer data register mn (TDRmn).
- TOMn : The TOMn pin output signal

Fig. 6-43 Example of register setting content for interval timer/square wave output

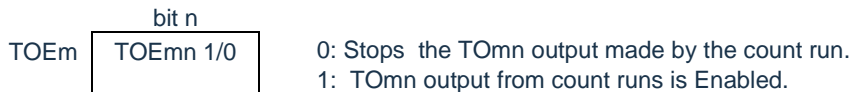
(a) Timer mode register mn (TMRmn).



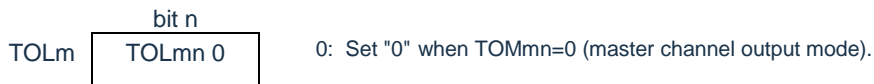
(b) Timer output register m (TOM).



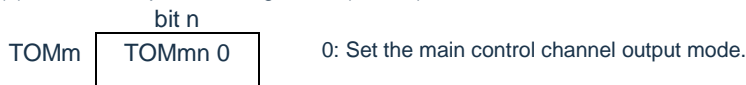
(c) The timer output enable register m (TOEm).



(d) The timer output level register m (TOLm).



(e) Timer output mode register m (TOMm).



Note: TMRm2 : MASTERmn bit

TMRm1, TMRm3 : SPLITmn bit

TMRm0: Fixed to "0".

Note: m: unit number (m=0,1) n: channel number (m=0: n=0 ~ 3, m=1: n=0 ~ 7).

Figure 6-44 the interval timer / square wave output function is in place

	software operation	Hardware Status
TAU initial settings		The input clock of the timer unit m is in a stopped supply state. (stop providing clock, cannot write registers)
	Place the TM4 mEN location "1" of the peripheral enable register 0 (PER0). →	The input clock of the timer unit m is in a supplied state. (Start providing clock capable of writing registers)
	Set timer clock selection register m (TPSm). Determine the clock frequency for CKm0 to CKm3.	
channel initial setting	Set timer mode register mn (TMRmm) (determine the channel operation mode). The timer data register mn (TDRmn) is set with interval (period) value.	The channel is in an operational stop state. (Provide clocks, consume a portion of the Power)
	Using TOMn output: The TOMmn position of the timer output mode register m (TOMm) is "0" (main control channel output mode). Position TOLmn "0". → Set the TOMn bit to determine the initial level of the TOMn output. Position TOEmn "1" to allow TOMn output. → Set the port register and port mode register to "0". →	The TOMn pin is in the Hi-Z output state. When the port mode register is in output mode and the port register is "0", the initially set level of the TOMn is output. The TOMn is unchanged because the channel is in an operational stop state. The TOMn pin outputs the level set by the TOMn.
	(TOEmn position "1" only when using TOMn output and restarting) Position TSmn (TSHm1, TSHm3) "1". → Automatically returned to '0' because the TSmn (TSHm1, TSHm3) bit is the trigger bit.	The TEmn (TEHm1, THEm3) bit becomes "1" and starts counting. Load the value of the TDRmn register into the timer count register mn (TCRmn). When the MDmn 0 bit of the TMRmn register is '1', INTTMmn is generated and TOMn is output alternately.
Start Run		
Running	You can change the settings of the TDRmn register at will. Can read TCRmn register at any time. TSRmn register cannot be used. Can change the TOM register and TOEm register settings. Prevents the setting of the TMRmn register, TOMmn bit, and TOLmn bit from being changed.	The counter (TCRmn) counts down. If the count goes to "0000H", the value of the TDRmn register is loaded again into the TCRmn register and counting continues. When TCRmn is detected as "0000H", INTTMmn is generated and TOMn is output interleaved. This run is repeated thereafter.
Stop Running	Position TTmn (TTHm1, TTHm3) "1". → Automatically returned to '0' because the TTmn (TTHm1, TTHm3) bit is the trigger bit.	The TEmn (TEHm1, TEHmn) bit changes to "0" and stops counting. The TCRmn register maintains count values and stops counting. The TOMn output is not initialized and remains in state.
	Set the TOEmn location "0" and the TOMn bit. →	The TOMn pin outputs the level set by the TOMn bit.
TAU Stop	To maintain the TOMn pin output level: Position TOMn "0" after setting the value to be maintained for the port register. The TOMn pin output level does not need to be maintained: No settings are required. →	Maintain the output level of the TOMn pin through port functionality.
	Position the TM4 mEN of the PER0 register "0".	The input clock of the timer unit m is in a stopped supply state. Initialize the SFRs of all circuits and channels. (TOMn bit becomes "0" and TOMn pin becomes port function)

Remarks: m: Unit number (m=0,1) n: Channel number (m=0: n=0~3, m=1: n=0~7).

6.8.2 Run as an external event counter

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

$$\text{The specified count value} = \text{TD Rmn's set value} + 1$$

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

By setting any channel start trigger bit (TSmn, TSHm1, TSHm3) of the timer channel start register m (TSm), "1", Load the value of the timer data register mn (TDRmn) into the TCRmn register.

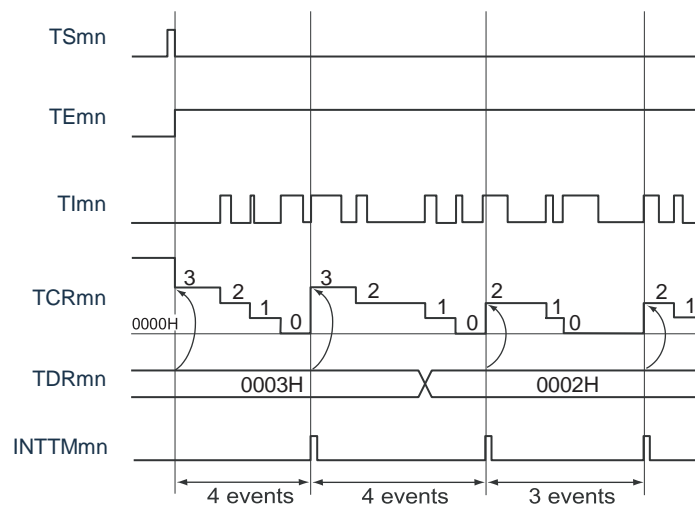
The TCRmn register counts down while detecting a valid edge of the input of the TImn pin. If TCRmn becomes "0000H", the value of the TDRmn register is loaded again and intTMmn is output.

After that, the same run continues.

Because the TOMn pin outputs an irregular waveform based on an external event, the timer output must be "0" at the TOEmn position of the enable register m (TOEm) to stop the output.

The TDRmn register can be overwritten at any time, and the value of the rewritten TDRmn register is valid for the next count period.

Figure 6-45 serves as a basic timing example of an external event counter operation



Note: 1. m : unit number (m=0,1)n: channel number (when m=0: n=0~3, m=1: n=0~7).

2. TSmn : The bitn of the timer channel start register m(TSm).

TEmn : The timer channel enable bitn of the status register m(TEm).

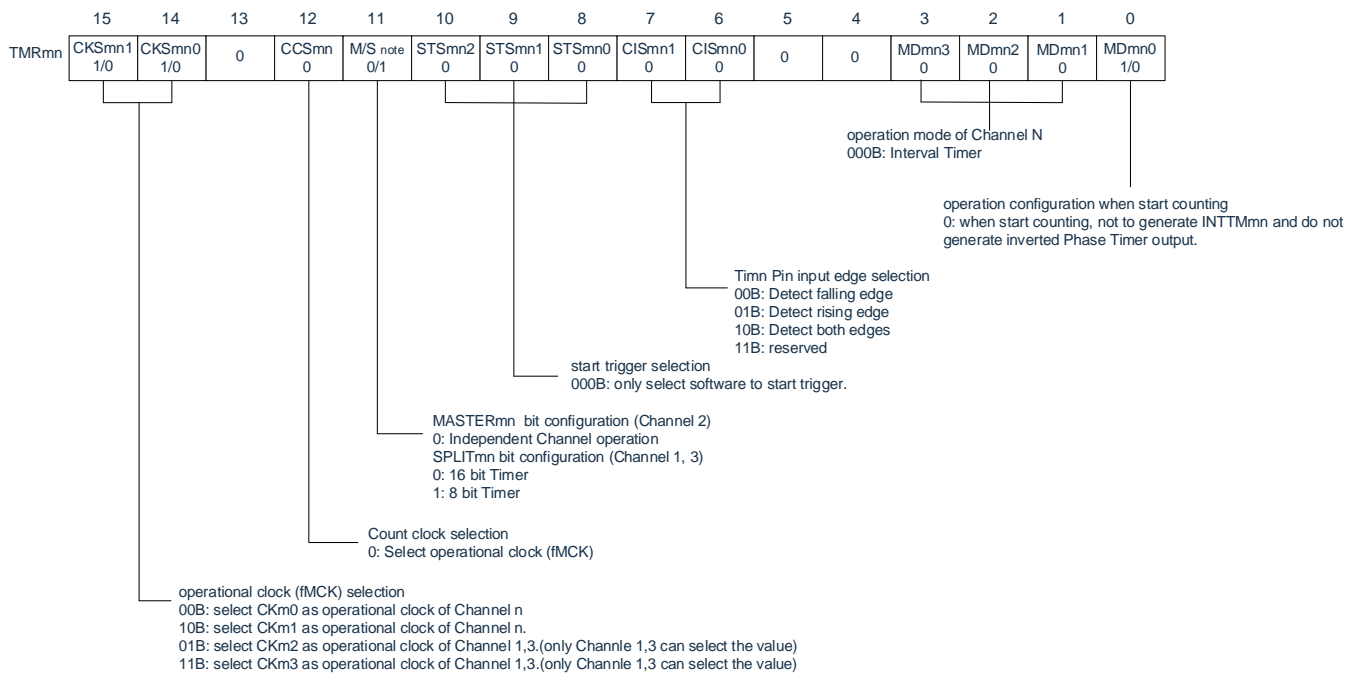
TImn : The TImn pin input signal

TCRmn: Timer count register mn (TCRmn).

TDRmn: Timer data register mn (TDRmn).

Fig. 6-46 register setting content in external event counter mode

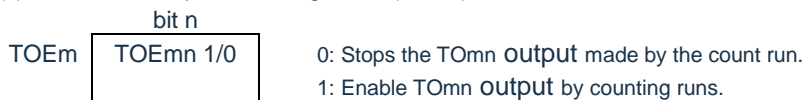
(a) Timer mode register mn (TMRmn).



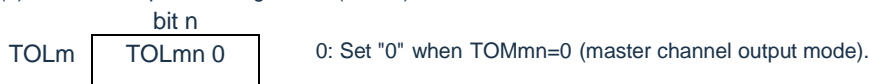
(b) Timer output register m (TOM).



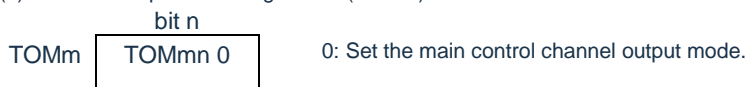
(c) The timer output enable register m (TOEm).



(d) Timer output level register m (TOLm).



(e) Timer output mode register m (TOMm).



注: TMRm2, TMRm4, TMRm6 : MASTERmn bit

TMRm1, TMRm3: SPLITmn bit

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

Note: m: unit number (m=0,1) n: channel number (when m=0: n=0~3, m=1: n=0~7).

Figure 6-47 The operation steps when the external event counter function is performed

	software operation	Hardware Status
Timer4 Initial Settings		The input clock of the timer unit m is in a state where supply is stopped. (stop providing clock, cannot write registers)
	Place the TM4 mEN location "1" of the peripheral enable register 0 (PER0). →	The input clock of the timer unit m is in a supplied state, and each channel is in an operation stop state. (Start providing clock capable of writing registers)
	A clock selection register m (TPSm) that sets the timer. Determine the clock frequency for CKm0 to CKm3.	
channel initial setting	Allow the noise filter to correspond to register 1 (NFEN12) either "OFF" or "1" (ON). A timer mode register mn (TMRmn) is set. A timer data register mn (TDRmn) is set with a count value. Output timer to allow TOEmn location "0" for register m (TOEm).	The channel is in an operational stop state. (Provide clocks, consume a portion of the Power)
Start Run	Position TSmn "1". The TSmn bit is a trigger bit and is automatically returned 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 enters the detection waiting state of the TImn pin input edge.
Running	You can change the settings of the TDRmn register at will. Can read TCRmn register at any time. The TSRmn register is not used. Prevents the setting of TMRmn registers, TOMmn bits, TOLmn bits, TOMn bits, and TOEmn bits from being changed.	The counter (TCRmn) counts down each time an input edge of the TImn pin is detected, and if the count reaches '0 000H', loads the value of the TDRmn register again into the TCRmn register and continues counting. A INTTMmn is generated when TCRmn is detected as '0000H'. This run is repeated thereafter.
Stop Running	Position TTmn "1". The TTmn bit is a trigger bit and is automatically returned to "0". →	The TEMn bit changes to "0" and stops counting. The TCRmn register maintains count values and stops counting.
Timer4 Stop	Position the TM4 mEN of the PER0 register "0". →	The input clock of the timer unit m is in a stopped supply state. Initialize the SFRs of all circuits and channels.

Restart Operation

6.8.3 Operation as a divider

It can divide the clock of the TI mn pin input and serve as a divider for the output of the TO mn pin.

The divider clock frequency of the TO mn output can be calculated using the following calculation equation:

- Select the **upper** ascending edge or the descending edge of the situation:
Divided clock rate = Input clock rate / { (set for T DRmn Value +1) ×2}
- Select the situation of the **two** edges:
The divider clock frequency ≈ input clock rate/(TDRmn set value +1).

In 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 timer data register is registered by detecting the effective edge of ti mn The value of (TDRmn) is loaded into the TCRmn register. At this point, if the MDmn0 bit of the timer mode register (TMRmn) is "0", intTMmn is not output and TOMn does not have alternating outputs; If the MDmn0 bit of the TMRmn register is "1", intTMmn is output and TOMn is alternately output.

The TCRmn registers are then decremented through the effective edges of the TI mn pin input. If the TCRmn becomes "0000H", the TOMn is output alternately. At the same time, load the value of the TDRmn register into the TCRmn register and continue counting.

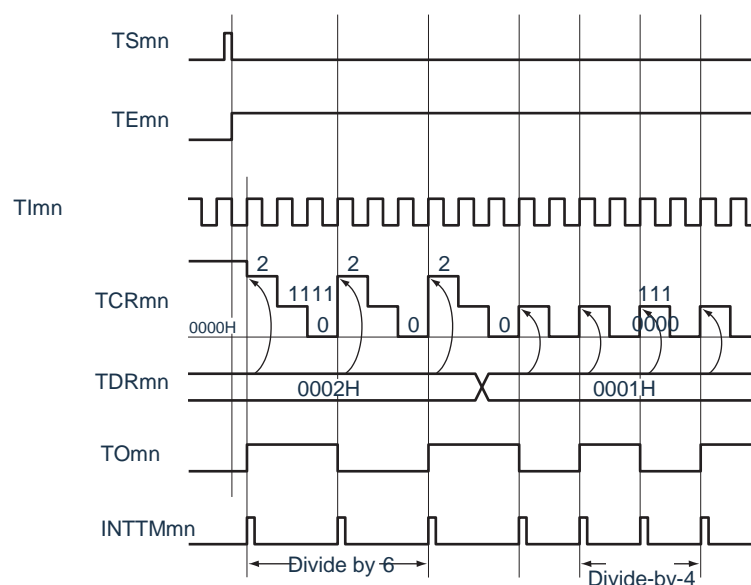
If the two-sided edge of the TI mn pin input is selected for detection, the duty cycle error of the input clock affects the divider clock period of the TOMn output.

The clock cycle of the TOMn output contains the sampling error of 1 operating clock cycle.

The clock period of the TOMn output = the supposed TOMn output clock week Period ± running clock cycle (error).

The TDRmn register can be overwritten at any time, and the value of the rewritten TDRmn register is valid for the next count period.

Figure 6-48 serves as a basic timing example of crossover operation (MDmn0=1)



Note TSmn: Bit n of the timer channel start register (TSM).

TEmn: The bit n of the timer channel that enable the status register (TEM).

TI mn: The TI mn pin input signal

TCRmn: Timer count register (TCRmn).

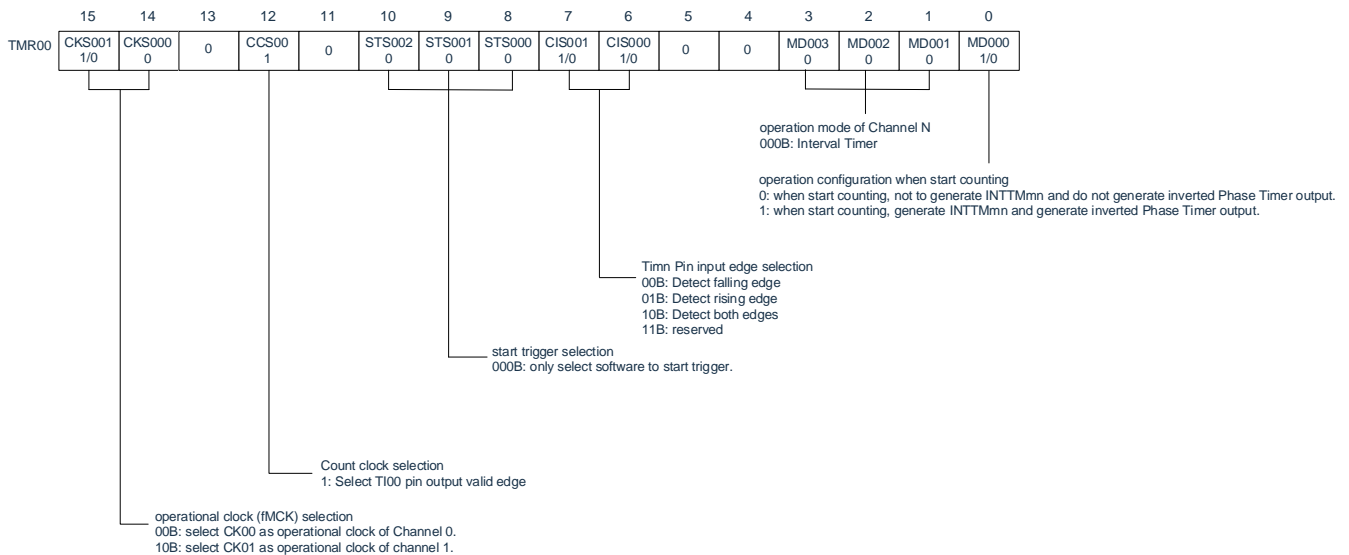
TDRmn: Timer data register (TDRmn).

TOmn: The TOmn pin output signal

m: unit number (m=0,1) n: channel number (when m=0: n=0~3, m=1: n=0~7).

Figure 6-49 an example of register settings when the divider is running (channel 0 of unit 0).

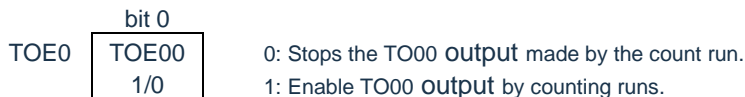
(a) Timer mode register 00 (TMR00).



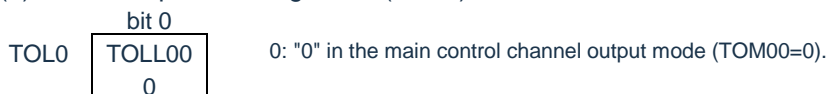
(b) Timer output register 0 (TO0).



(c) Timer output enable register 0 (TOE0).



(d) Timer output level register 0 (TOL0).



(e) Timer output mode register 0 (TOM0).

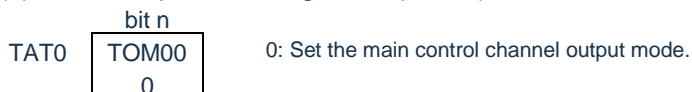


Fig. 6-50 Operational Steps for Frequency Divider (take channel 0 of unit 0 as an example)

	software operation	hardware state
Timer 4 initial configuration		Timer Unit 0 input clock is in stopped state (stop providing clock, not able to write into registers)
	set TM4mEN bit of peripheral enable register 0 (PER0) to '1' →	Timer Unit 0 input clock is in active state (start providing clock, able to write into registers)
	configure Timer clock selection register 0 (TPS0), confirm CK00~CK03 clock frequency	
Channel Initial configuration	set corresponding bit of noise filter enable register 1 (NFEN1) to '0' (OFF) or '1' (ON). Configure Timer mode register 00 (TMR00) (confirm channel operation mode, select edge detection). Configure interval(period) value of Timer data register 00 (TDR00)	channel in operation stopped state (providing clock, consume portion of power)
	set TOM00 bit of timer output mode register 0 (TOM0) to '0' (master control channel output mode). Set TOL00 bit to '0'	TO00 pin in Hi-Z output state.
	configure TO00 bit and confirm TO00 output initial voltage value. →	When port mode register set to output mode and port register as '0', output TO00 initial configured voltage level.
	Set TOE00 bit to '1', allow TO00 output. → Set port register and port mode register to '0'. →	Because channel is in operation stopped state, thus TO00 remains unchanged. TO00 pin output TO00 configured voltage level.
Start operation	set TOE00 bit to '1' (only limited to restart operation). → Set TS00 bit to '1'. Because TS00 bit is trigger bit, thus automatically return to '0'.	TE00 bit turns to '1' and start counting. Load TDR00 register value into Timer count register 00 (TCR00). When MD000 bit of TMR00 register turns into '1', generate INTTM00 and TO00 swaps output
in operation	can modify any TDR00 register configuration value. Can read TCR00 register anytime. Do not use TSR00 register. Can modify TO0 register and TOE0 register value. Forbidden modifying TMR00 register. TOM00 bit and TOL00 bit configuration value.	Counter (TCR00) performs decremental counting. When count reaches '0000H', then load TDR00 register value into TCR00 register again and continue counting. When detecting TCR00 as '0000H', generate INTTM00 and TO00 swaps output. Thereafter, repeat the operation.
stop operation	set TT00 bit to '1'. → Because TT00 bit is trigger bit, thus automatically return to '0'.	TE00 bit turns to '0' and stop counting. TCR00 register remains counted value and stop counting. TO00 output not been initialized and remain same state. TO00 pin outputs TO00 configured voltage.
	set TOE00 bit to '0' and configure value for TO00 bit. →	TO00 pin output TO00 configured voltage level.
Timer 4 stop	Scenarios to maintain TO00 pin output voltage: set TO00 bit to '0' after set hold value to port register configuration. → In case TO00 pin output voltage does not need to be held: no configuration required	hold TO00 pin output voltage level via port function.
	set TM4mEN bit of peripheral enable register 0 (PER0) to '0' →	Timer Unit 0 input clock is in stopped state. Perform initialization to all circuit and SFR of all channels. (TO00 bit turns into '0' and TO00 pin becomes port function)

Restart operation

6.8.4 Operation as input pulse interval measurements

The count value can be captured at the effective edge of TImn and the interval between TImn input pulses can be measured. During the period when the TEmn bit is "1", the software operation (TSmn=1) can also be set to capture trigger, and the capture count value can also be set.

The pulse interval can be calculated using the following calculation equation:

$$\text{TImn input pulse interval} = \text{the week of the counting clock} \times ((100\ 00\text{H} \times \text{TSRmn: OVF}) + (\text{TDRmn catch value} + 1)).$$

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 (TSm) of the timer channel start register m(TSm) is set to "1", the TCRmn register is clocked from "0000H" Start incrementing the count.

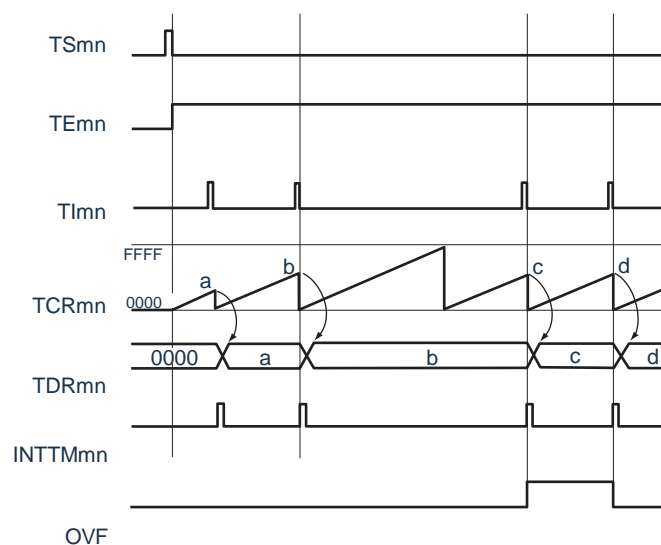
If a valid edge of the TImn pin input is detected, the count value of the TCRmn register is transmitted (captured) to the timer data register mn (TDRmn) and the TCRmn register is cleared "0000H", then output INTMmn. At this point, if the counter overflows, the OVF position of the timer status register mn (TSRmn) is "1". If the counter does not overflow, the OVF bit is cleared. After that, the same run continues.

While the count value is captured to the TDRmn register, the OVF bit of the TSRmn register is updated according to whether there is an overflow during the measurement, and the overflow status of the captured value can be confirmed.

Even if the counter performs a full count of 2 cycles or more, it is considered to have overflowed and the OVF position of the TSRmn register is considered to be "1". However, when 2 or more overflows occur, the interval value cannot be measured normally by the OVF bit.

The STSmn2~STSmn0 position of the TMRmn register is "001B", and the effective edge of TImn is used for start triggering and capture triggering.

Fig. 6-51 an example of the basic timing of the operation of the input pulse interval measurement (MDmn0=0).



Note 1.m: Unit number (m=0,1) n: Channel number (when m=0: n=0~3, m=1: n=0~7).

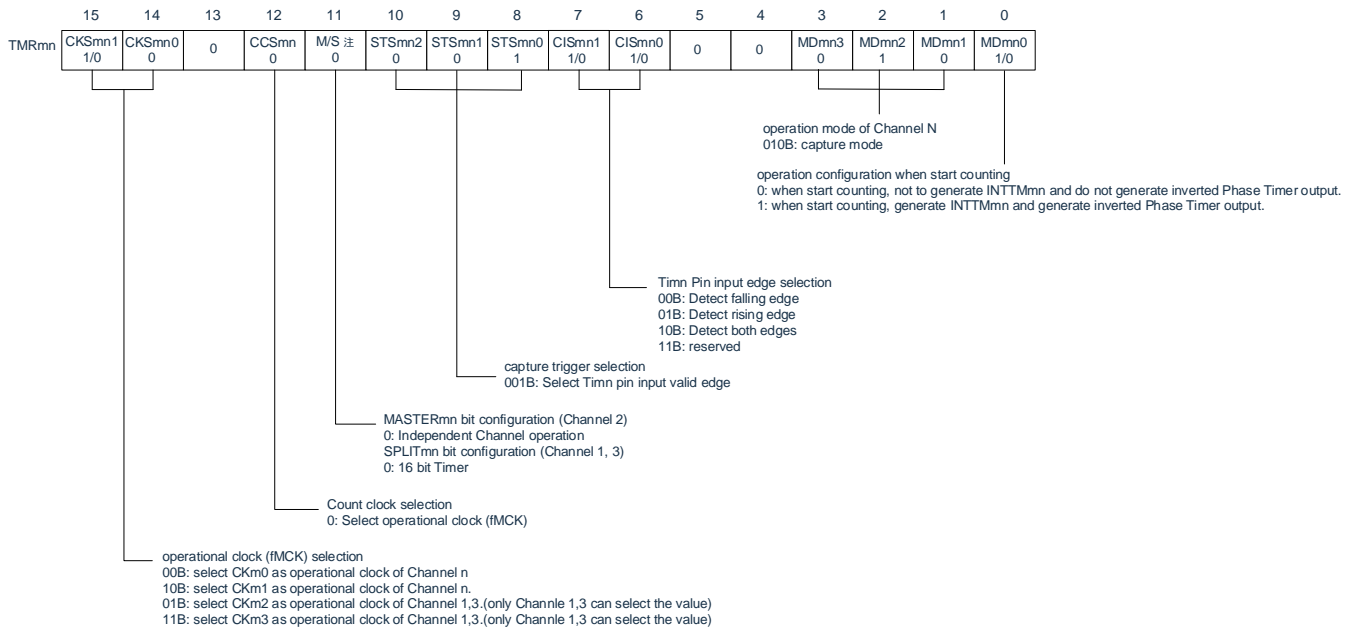
2. TSmn: Bitn of the timer channel start register m(TSm).

TEmn : The timer channel enable bitn of the status register m(TEm).

TImn : The TImn pin input signal
 TCRmn: Timer count register mn (TCRmn).
 TDRmn: Timer data register mn (TDRmn).
 OVF: Bit 0 of the timer status register mn (TSRmn).

Fig. 6-52 Example of register setting value at the measurement input pulse interval

(a) Timer mode register mn (TMRmn).



(b) The timer output register m (TOM).

bit n
 TOM

TOMn
0

 0: Outputs "0" by TOMn.

(c) The timer output enable register m (TOEm).

bit n
 TOEm

TOEmn
0

 0: Stops the TOMn output made by the count run.

(d) The timer output level register m(TOLm).

bit n
 TOLm

TOLmn
0

 0: "0" in the master channel output mode (TOMmn=0).

(e) Timer output mode register m (TOMm).

bit n
 TOMm

TOMmn
0

 0: Set the main control channel output mode.

Note: TMRm2, TMRm4, TMRm6: MASTERmn bit

TMRm1, TMRm3: SPLITmn bit

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

Remark: m: unit number (m=0,1) n: channel number (when m=0: n=0~3, m=1: n=0~7).

Fig. 6-53 Operation steps when the pulse interval measurement function is input

	software operation	hardware state
Timer 4 initial configuration		Timer Unit m input clock is in stopped state (stop providing clock, not able to write into registers)
	set TM4mEN bit of peripheral enable register 0 (PER0) to '1'	Timer Unit m input clock is in active state, all channels in operation stopped state.
	configure Timer clock selection register m(TPSm), confirm CKm0~CKm3 clock frequency	
Channel Initial configuration	set corresponding bit of noise filter enable register 1 (NFEN1) to '0' (OFF) or '1' (ON). Configure Timer mode register mn (TMRmn) (confirm channel operation mode).	channel in operation stopped state (providing clock, consume portion of power)
Start operation	set TSmn bit to '1'. Because TSmn bit is trigger bit, thus automatically return to '0'.	TEmn bit turns into '1' and start counting. Clear Timer counting register (TCRn) to "0000H". When MDmn0 bit of TMRmn register is '1', generate INTTMmn.
in operation	can only modify configure value of CISmn1 bit and CISmn0 bit of TMRmn register. Can read TDRmn register anytime. Can read TCRmn register anytime. Can read TSRmn register anytime. Forbidden modifying TOMmn bit, TOLmn bit, TOMn bit and TOEmn bit configuration.	Counter(TCRmn) start incremental counting from "0000H", if detecting TImn pin input valid edge or TSmn bit set to '1', then transfer (capture) counting value to Timer data register mn(TDRmn), at the same time, clear TCRmn to "0000H" and generate INTTMmn. At this time, if overflow occurs, then set OVF bit of Timer status register mn(TSRmn) . If overflow does not occur, then clear OVF bit. Thereafter, repeat the process.
stop operation	set TTmn bit to '1'. Because TTmn bit is trigger bit, thus automatically return to '0'.	TEmn bit turns into '0' and stop counting. TCRmn register hold counted value and stop counting. OVF bit of TSRmn register remains unchange.
timer 4 stop	set TM4mEN bit of peripheral enable register 0 (PER0) to '1'	Timer Unit m input clock is not been provided.Perform initialization to all circuit and SFR of all channels. (TO00 bit turns into '0' and TO00 pin becomes port function)

Note: m: Unit number (m=0,1) n: Channel number (when m=0: n=0~3, m=1: n=0~7).

6.8.5 Operation as input signal high and low level width measurements

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, 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 signal width of TImn can be calculated using the following equation.

$$\text{Signal width of TImn input} = \text{period of counting clocks} \times (10000\text{H} \times \text{TSRmn: OVF}) + (\text{capture value of TDRmn} + 1).$$

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 (TSm) of the timer channel start register m(TSm) is set to "1", the TEMn bit becomes "1", and enter the start edge detection wait state of the TImn pin.

If the start edge of the TImn pin input (the rising edge of the TImn pin input when measuring high level width) is detected, it is synchronized with the counting clock and the count is incremented starting at "0000H". Then, if a valid capture edge (the falling edge of the TImn pin input when measuring the high level width is measured), the intTMmn is output at the same time that the count value is passed to the timer data register mn (TDRmn). At this point, if the counter overflows, the OVF position bit of the timer status register mn (TSRmn) is placed. 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 stops counting, and enters the start edge detection wait state of the TImn pin. After that, the same run continues.

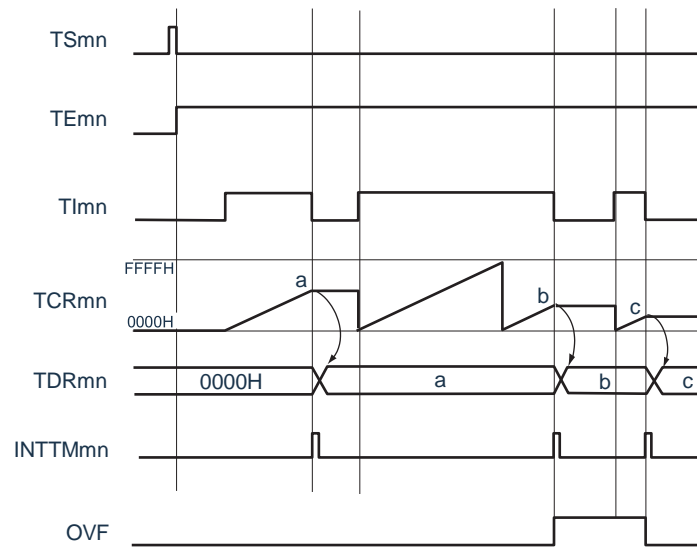
While the count value is captured to the TDRmn register, the OVF bit of the TSRmn register is updated according to whether there is an overflow during the measurement, and the overflow status of the captured value can be confirmed.

Even if the counter performs a full count of 2 cycles or more, it is considered to have overflowed and the OVF position of the TSRmn register is considered to be "1". However, when 2 or more overflows occur, the interval value cannot be measured normally by the OVF bit.

The CISmn1 and CISmn0 bits of the TMRmn registers can be used to determine whether to measure the high or low level width of the TImn pin. This function is designed to measure the input signal width of the TImn pin, so the TSmn position "1" cannot be placed during the period when the TEMn bit is "1".

CISmn1, CISmn0=10B of TMRmn registers: Measures the low level width. CISmn1, CISmn0=11B of the TMRmn register: Measures the high level width.

Figure 6-54 an example of running basic timing for input signal high and low level width measurements



Note: 1.m: unit number (m=0,1) n: channel number (m=0: n=0~3, m=1: n=0~7).

2. TSmn: Bitn of the timer channel start register m(TSm).

TEmn : The bit n of the timer channel enable the status register m(TEm).

TImn : The TImn pin input signal

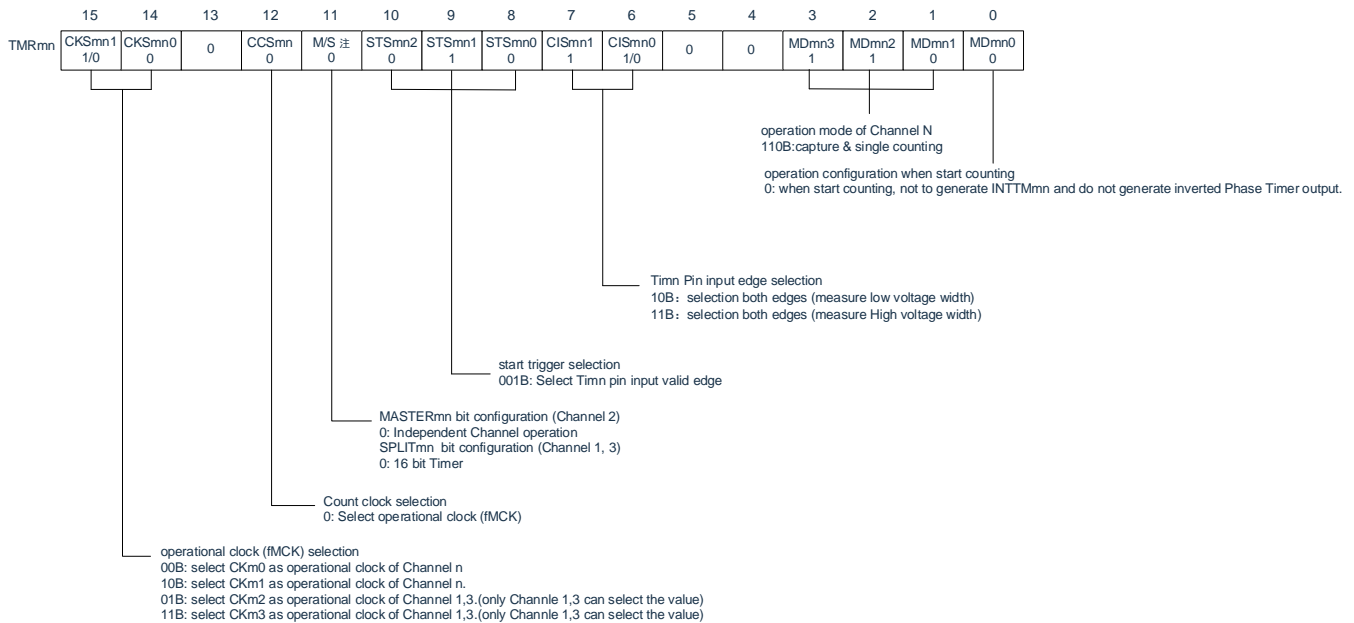
TCRmn: Timer count register mn (TCRmn).

TDRmn: Timer data register mn (TDRmn).

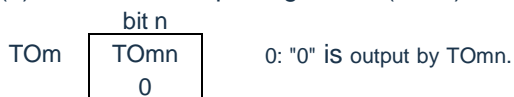
OVF : Bit0 of the timer status register mn (TSRmn).

Fig. 6-55 Example of register setting content when measuring the high and low level width of the input signal

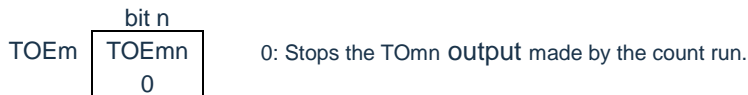
(a) Timer mode register mn (TMRmn).



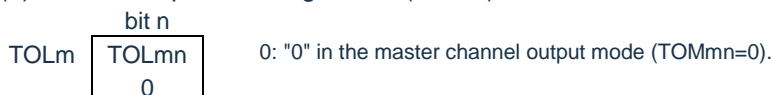
(b) The timer output register m (TOM).



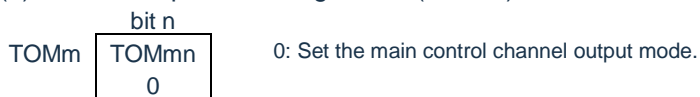
(c) The timer output enable register m (TOEm).



(d) Timer output level register m (TOLm).



(e) Timer output mode register m (TOMm).



Note: TMRm2, TMRm4, TMRm6: MASTERmn bit

TMRm1, TMRm3: SPLITmn bit

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

Remark: m: unit number (m=0,1) n: channel number (when m=0: n=0~3, m=1: n=0~7).

Fig. 6-56 input signal when measuring the high and low level width function

	software operation	hardware state
Timer 4 initial configuration		Timer Unit 0 input clock is in stopped state (stop providing clock, not able to write into registers)
	set TM4mEN bit of peripheral enable register 0 (PER0) to '1' →	Timer Unit m input clock is in active state, all channels in operation stopped state.
	configure Timer clock selection register m(TPSm), confirm CKm0~CKm3 clock frequency	
Channel Initial configuration	set corresponding bit of noise filter enable register 1 (NFEN1) to '0' (OFF) or '1' (ON). Configure Timer mode register mn (TMRmn) (confirm channel operation mode). Set T0Emn bit to '0', and stop T0mn operation.	channel in operation stopped state (providing clock, consume portion of power)
Start operation	set TSmn bit to '1'. →	TEmn bit turns into '1' and enter into start trigger (detect TImn pin input valid edge or set TSmn bit to '1') detection waiting state.
	detect TImn pin input counting start edge →	clear timer counting register mn (TCRmn) to "0000H" and start decremental counting.
in operation	can modify any TDRmn register configuration value. Can read TCRmn register anytime. Do not use TSRmn register. Forbidden modifying TMRmn register, TOMmn bit and TOLmn bit, Tomn and T0Emn bit configuration value.	while detecting TImn pin start edge, Counter(TCRmn) start incremental counting from "0000H", if detecting TImn pin input capture edge, then transfer counting value to Timer data register mn(TDRmn) and generate INTTmn. At this time, if overflow occurs, then set OVF bit of Timer status register mn(TSRmn) . If overflow does not occur, then clear OVF bit. TCRmn register stop counting before detecting next TImn pin start edge. Thereafter, repeat the process.
stop operation	set TTmn bit to '1'. →	TEmn bit turns into '0' and stop counting. TCRmn register hold counted value and stop counting. OVF bit of TSRmn register remains unchange.
timer 4 stop	set TM4mEN bit of peripheral enable register 0 (PER0) to '1' →	Timer Unit m input clock is not been provided.Perform initialization to all circuit and SFR of all channels. (TO00 bit turns into '0' and TO00 pin becomes port function)

restart operation

Note: m: unit number (m=0,1) n: channel number (when m=0: n=0~3, m=1: n=0~7).

6.8.6 Runs as a delay counter

The count can be decremented by the effective edge detection (external event) of the TImn pin input and intTMmn is generated at arbitrary set intervals (Timer interrupt).

During the period when the TEmn bit is "1", the TSmn position "1" can be changed by software, the count can be decremented, and intTMmn (timer interrupt) can be generated at any set interval.

The interrupt generation period can be calculated using the following calculation equation:

$$\text{InTT Mmn (Hours interrupt) of the production cycle} = \text{counting clocks Period} \times (\text{TDRmn set} + 1).$$

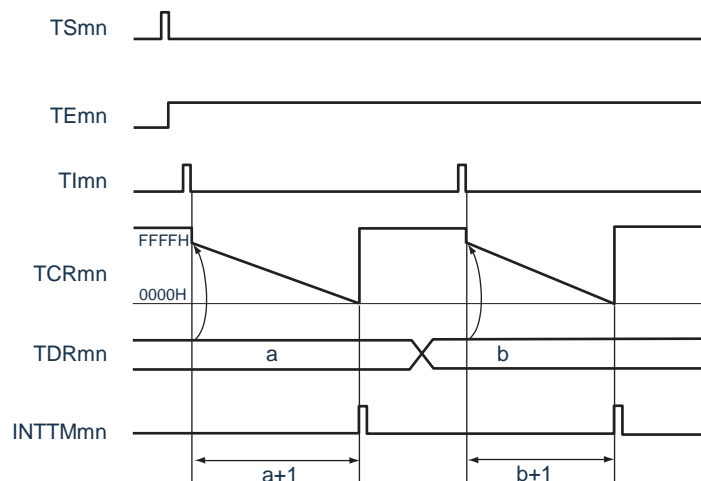
In single-count mode, the timer count register mn (TCRmn) is used as a decrement counter.

If the channel start trigger bit (TSmn, TSHm1, TSHm3) of the timer channel start register m(TSm) is set to "1" TEmn

Bits, TEHm1 bits, and TEHm3 bits become "1" and enter a valid edge detection wait state for the TImn pin. By detecting the valid edge of the TImn pin input, the operation of the TCRmn register begins and the value of the timer data register mn (TDRmn) is loaded. The TCRmn register decrements the count from the value of the loaded TDRmn register by counting the clock. If TCRmn becomes "0000H", INTMmn is output and counts are stopped before a valid edge of the next TImn pin input is detected.

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

Figure 6-57 an example of the basic timing of the operation of the delay counter



Note: 1.m: unit number (m=0,1) n: channel number (m=0: n=0~3, m=1: n=0~7).

2. TSmn : The bitn of the timer channel start register m(TSm).

TEmn : The timer channel enable bitn of the status register m(TEm).

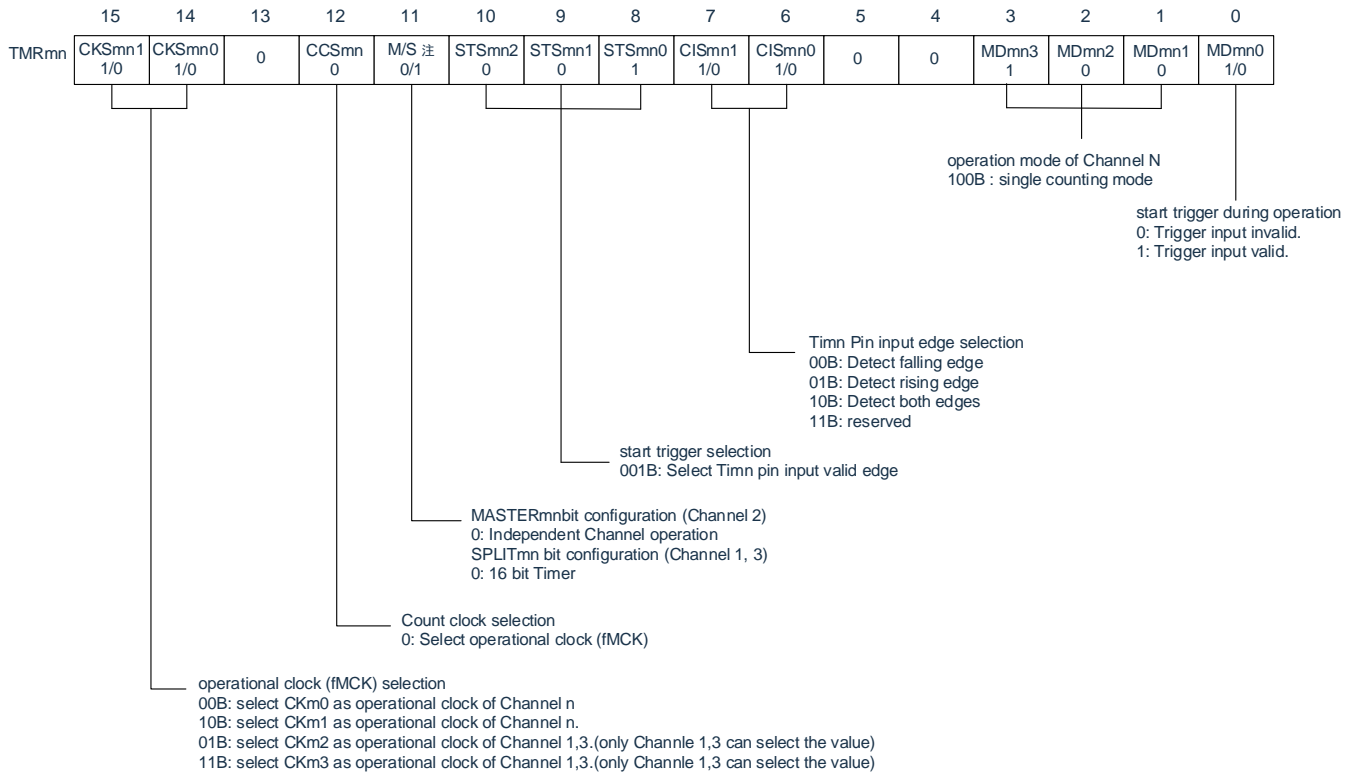
TImn : The TImn pin input signal

TCRmn: Timer count register mn (TCRmn).

TDRmn: Timer data register mn (TDRmn).

Fig. 6-58 Example of register setting value during the 58 delay counter function

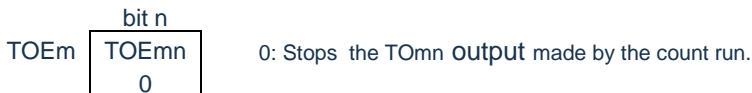
(a) Timer mode register mn (TMRmn).



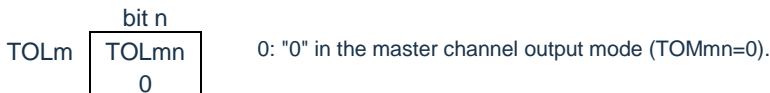
(b) The timer output register m (TOm).



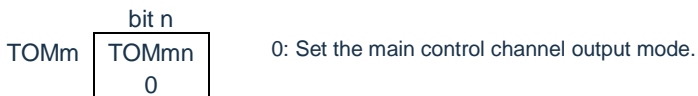
(c) The timer output enable register m (TOEm).



(d) The timer output level register m(TOLm).



(e) Timer output mode register m (TOMm).



Note: TMRm2, TMRm4, TMRm6: MASTERmn bit

TMRm1, TMRm3: SPLITmn bit

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

Remark: m: unit number (m=0,1) n: channel number (when m=0: n=0~3, m=1: n=0~7).

Figure 6-59 The operation steps when the delay counter function is performed

	software operation	hardware state
Timer 4 initial configuration		Timer Unit m input clock is in stopped state (stop providing clock, not able to write into registers)
	set TM4mEN bit of peripheral enable register 0 (PER0) to '1' →	Timer Unit m input clock is in active state, all channels in operation stopped state. (start providing clock, can write all registers)
	configure Timer clock selection register m(TPSm), confirm CKm0~CKm3 clock frequency	
Channel Initial configuration	set corresponding bit of noise filter enable register 1 (NFEN1) to '0' (OFF) or '1' (ON). Configure Timer mode register mn (TMRmn) (confirm channel operation mode). Configure output delay time via timer data register mn (TDRmn) Set T0Emn bit to '0', and stop T0mn operation.	channel in operation stopped state (providing clock, consume portion of power)
Start operation	set TSmn bit to '1'. Because TSmn bit is trigger bit, thus automatically return to '0'. →	TEmn bit turns into '1' and enter into start trigger (detect Timn pin input valid edge or set TSmn bit to '1') detection waiting state.
	start decremental counting while detecting next start trigger. • Detect Timn pin input valid edge → • set TSmn bit to "1" via software	load TDRmn register value into Timer counting register mn (TCRmn)
in operation	can modify any TDRmn register configuration value. Can read TCRmn register anytime. Do not use TSRmn register.	Counter (TCR00) performs decremental counting. When TCRmn count reaches '0000H', then generate INTTMmn and before detecting the next start trigger (detect Timn pin input valid edge or set TSmn bit to '1'), TCRmn is "0000H" and stop counting.
stop operation	set TTmn bit to '1'. Because TTmn bit is trigger bit, thus automatically return to '0'. →	TEmn bit turns into '0' and stop counting. TCRmn register hold counted value and stop counting.
Timer 4 stop	set TM4mEN bit of peripheral enable register 0 (PER0) to '0' →	Timer Unit m input clock is not been provided. Perform initialization to all circuit and SFR of all channels.

restart operation

Note: m: unit number (m=0,1) n: channel number (when m=0: n=0~3, m=1: n=0~7).

6.9 Multi-channel linkage operation function of the universal timer unit

6.9.1 Operation as a single-trigger pulse output function

Pairing the two channels enable a single-trigger pulse of any delay pulse width to be generated via the input of the TImn pin. Delay and pulse width can be calculated using the following calculations:

$$\text{Delay} = \{\text{TDRmn (master) set value} + 2\} \times \text{count clock period}$$
$$\text{Pulse width} = \{\text{TDRmp (dependent) set value}\} \times \text{counting clock periods}$$

In single count mode, the master channel runs and the delay is counted. By detecting start firing, the timer count register mn (TCRmn) of the master channel starts running and loads the value of the timer data register mn (TDRmn). The TCRmn register decrements the count from the value of the loaded TDRmn register by counting the clock. If TCRmn becomes "0000H", INTMmn is output and the count is stopped before the next start trigger is detected.

In single count mode, slave channels run and pulse widths are counted. With the INTMmn of the master channel triggered as the start, the slave channel's TCRmp register starts running and loads the value of the TDRmp register. The TCRmp register decrements the count from the loaded TDRmp register value by counting the clock. If the count value changes to "0000H", intTMmp is output and the count is stopped before the next start trigger (INTMmn of the master channel) is detected. After generating INTMmn from the master channel and passing through a count clock, the output level of TOMP becomes the effective level if TCRmp becomes "0000H", which becomes invalid.

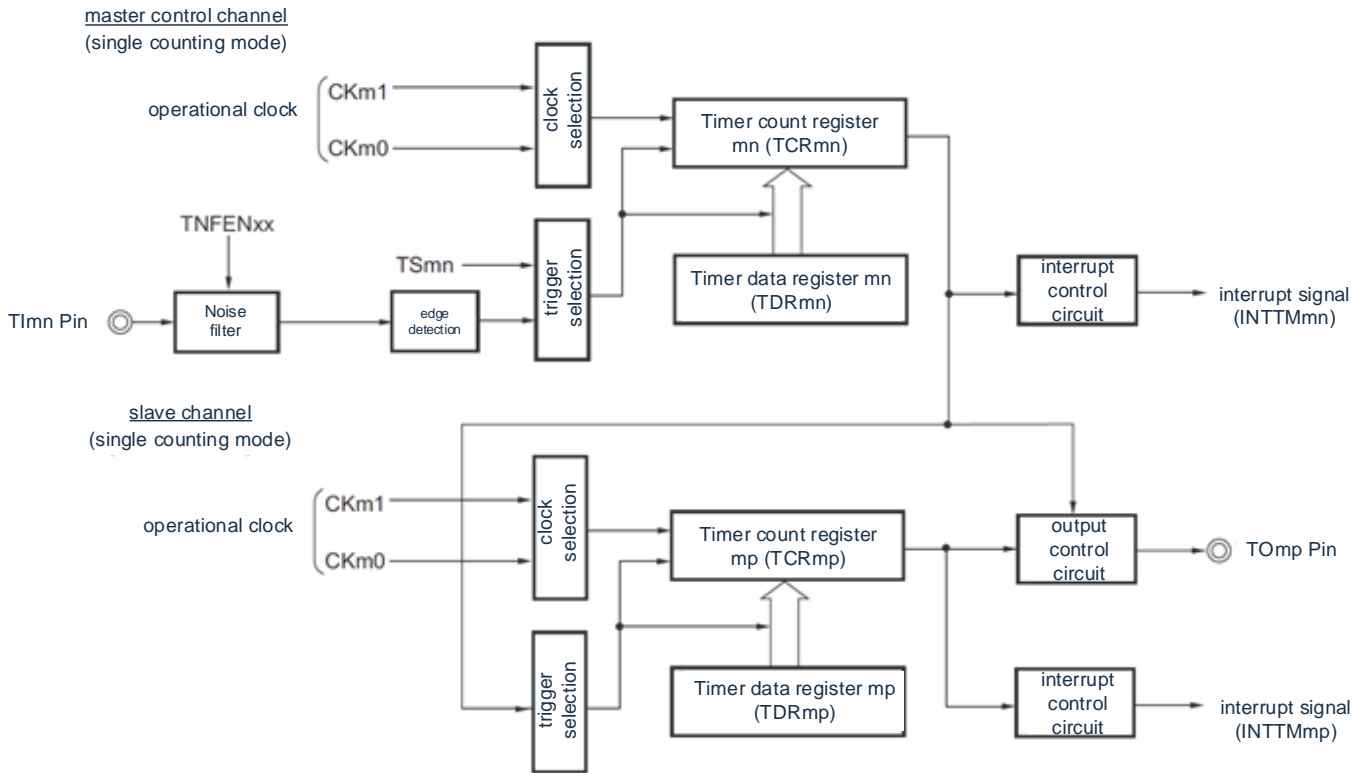
Software operation (TSmn=1) can also be used as a start trigger to output a single trigger pulse without using the TImn pin input.

Remark: Because the mount timing of the TDRmn register of the master channel and the TDRmp register of the slave channel is different, if the TDRmn register and the TDRmp register are rewritten during the counting process, it may compete with the loading timing and output an abnormal waveform. The TDRmn register must be overwritten after intTMmn is generated, and the TDRmp register must be overwritten after INTMmp is generated.

Note: m: Unit number (m=0,1)n: Master channel number (n=0, 2, 4, 6).

p: Slave channel number (m=0Time:n<p≤3, m=1Time:n<p≤7)

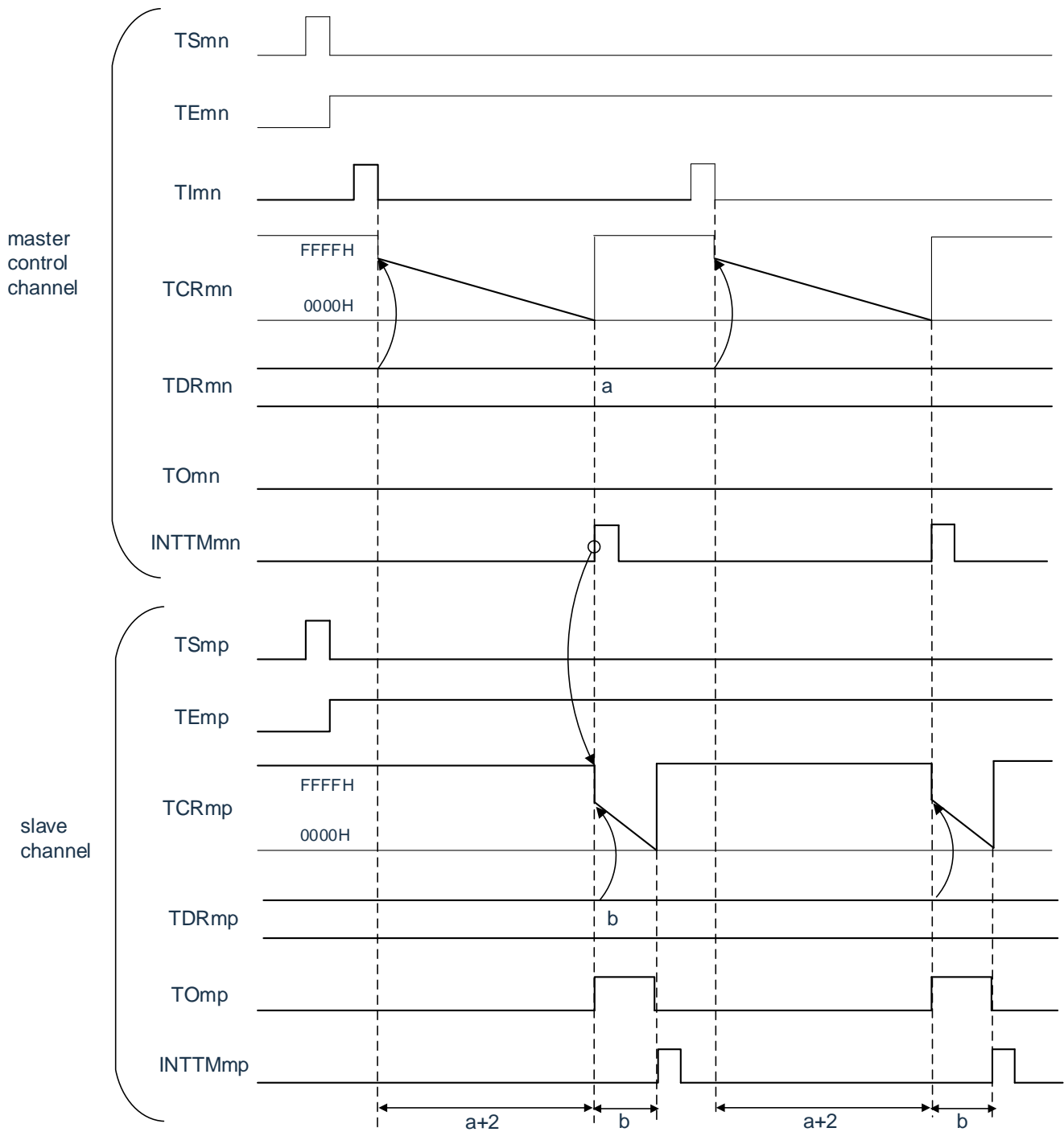
Fig6-60 operating as a single-trigger pulse output function



Note: m: Unit number (m=0,1) n: Master channel number (n=0, 2, 4, 6).

p: Slave channel number (m=0Time:n<p≤3, m=1Time:n<p≤7)

Fig6-61 an example of the basic timing of the operation of the single-trigger pulse output function



Note: 1.m: Unit number (m=0,1)n: Master channel number (n=0, 2, 4, 6).

p: Slave channel number (m=0Time:n<p≤3, m=1Time:n<p≤7)

2. TS_{mn} , TS_{mp} : the bit n, p of the timer channel start register m (TS_m).

TE_{mn} , TE_{mp} : The timer channel enable bitn, p of the status register m (TE_m).

TI_{mn} , TI_{mp} : Input signals for the TI_{mn} pin and TI_{mp} pin

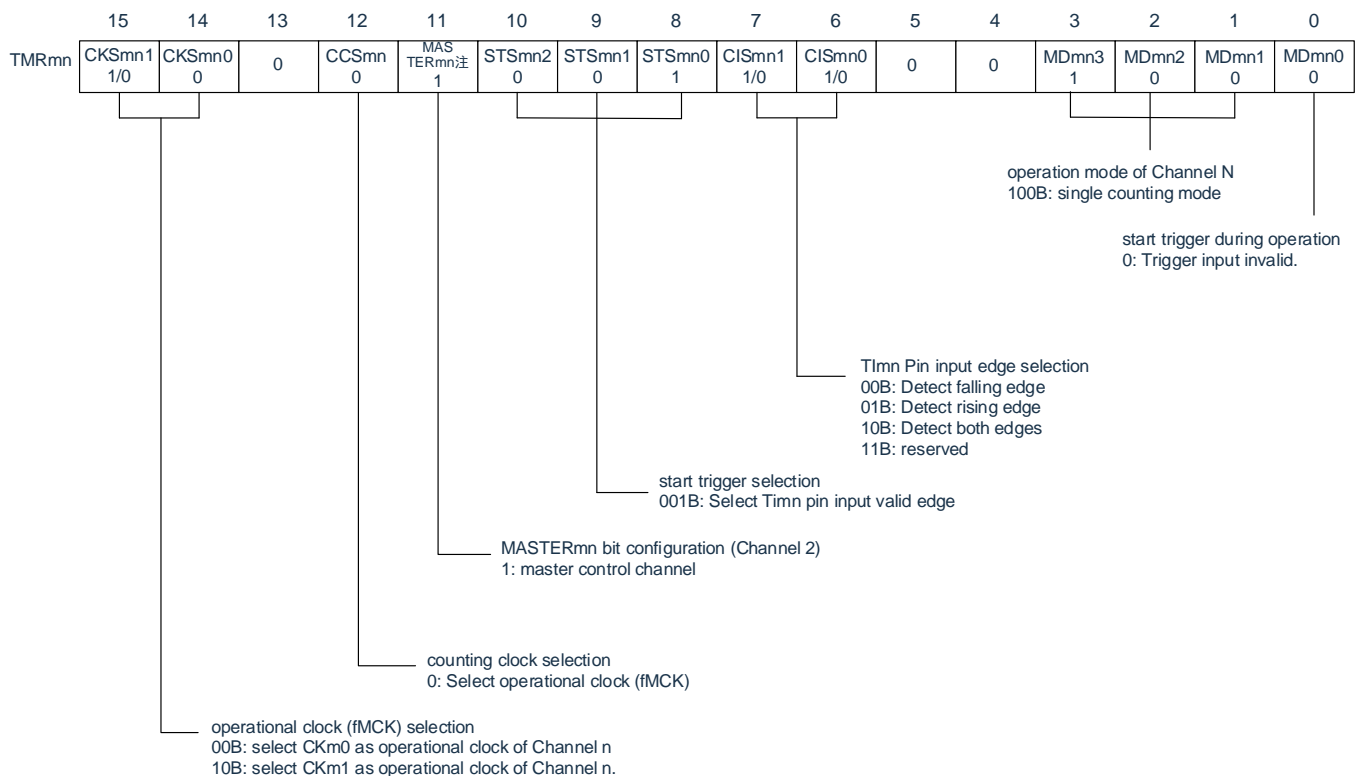
TCR_{mn} , TCR_{mp} : Timer count registers mn, mp (TCR_{mn} , TCR_{mp}).

TDR_{mn} , TDR_{mp} : timer data register mn, mp (TDR_{mn} , TDR_{mp}).

TO_{mn} , TO_{mp} : Output signals for TO_{mn} pins and TO_{mp} pins

6-62 Example of register setting content when the single trigger pulse output function (master channel) is used

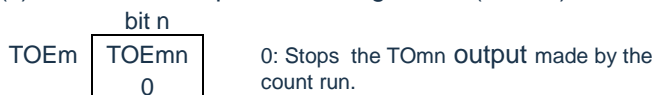
(a) Timer mode register mn (TMRmn).



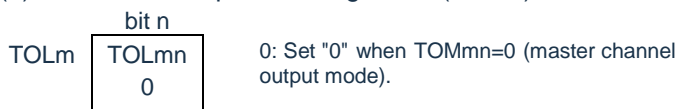
(b) The timer output register m (TOM).



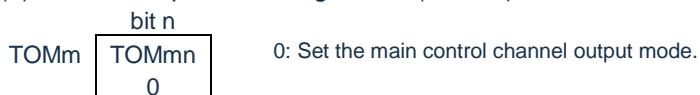
(c) The timer output enable register m (TOEm).



(d) The timer output level register m(TOLm).



(e) Timer output mode register m (TOMm).

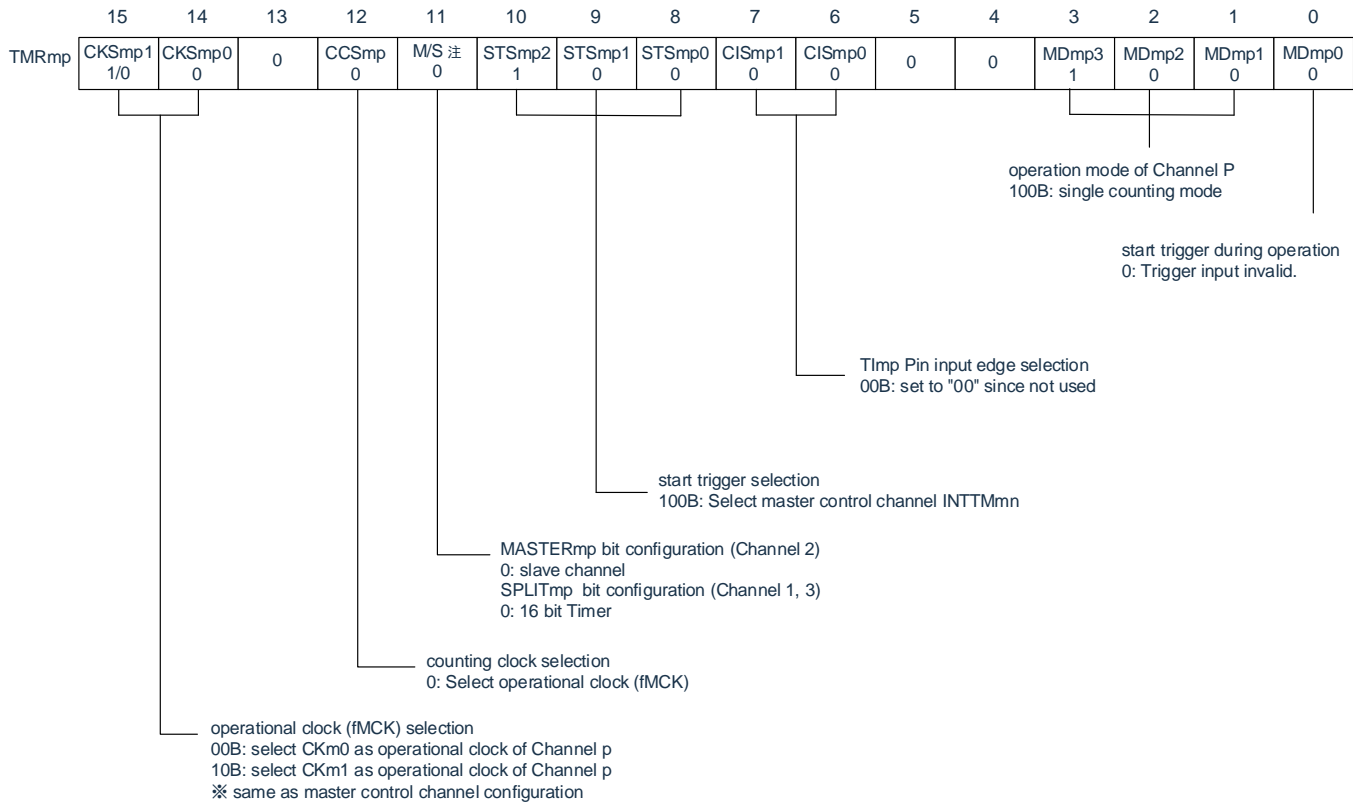


Remar TMRm2, TMRm4, TMRm6 : MASTERmn=1
TMRm0, TMRm5, TMRm7 : Fixed to "0".

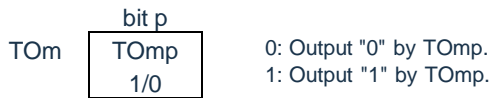
Note: m: Unit number (m=0,1) n: Master channel number (n=0, 2, 4, 6).

6-63 Example of register settings when the pulse output function is triggered alone (slave channel).

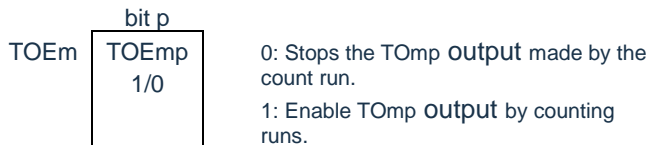
(a) Timer mode register mp (TMRmp).



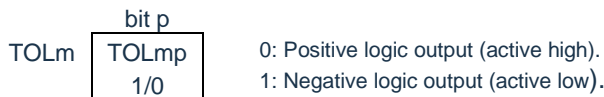
(b) The timer output register m (TOm).



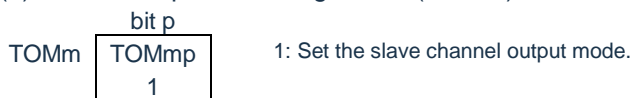
(c) The timer output enable register m (TOEm).



(d) The timer output level register m (TOLm).



(e) Timer output mode register m (TOMm).



Remark: TMRm2, TMRm4, TMRm6: MASTERmp bit

TMRm1, TMRm3: SPLITmp bit

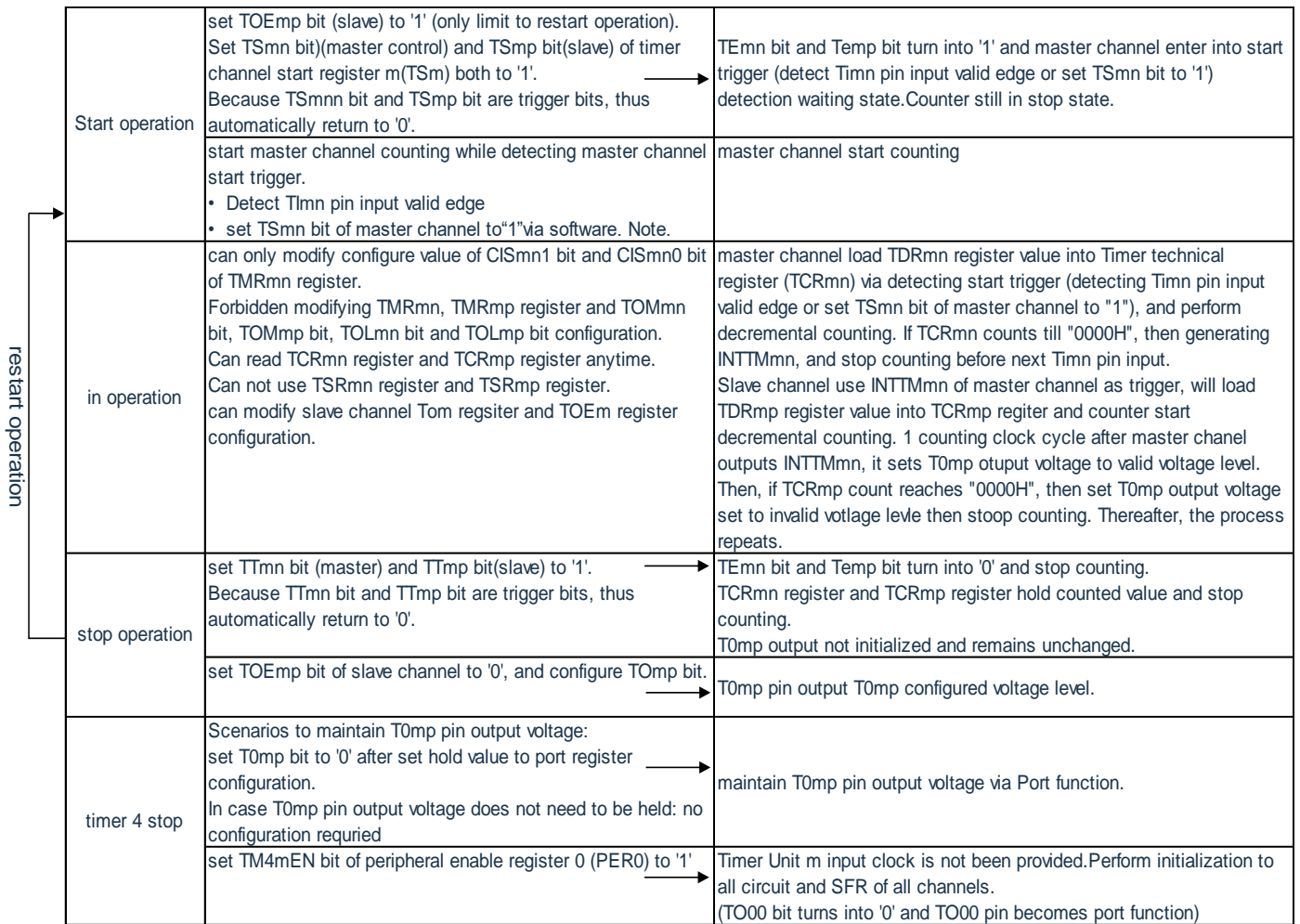
Note: m: Unit number (m=0,1) n: Master channel number (n=0, 2, 4, 6).

p: Slave channel number (m=0Time:n<p≤3, m=1Time:n<p≤7)

Fig6-64 Operating steps for single trigger pulse output function (1/2).

	software operation	hardware state
Timer 4 initial configuration		Timer Unit m input clock is in stopped state (stop providing clock, not able to write into registers)
	set TM4mEN bit of peripheral enable register 0 (PER0) to '1' →	Timer Unit m input clock is in active state, all channels in operation stopped state. (start providing clock, Start to provide clock, can write to each register)
	configure Timer clock selection register m(TPSm), confirm CKm0~CKm3 clock frequency	
Channel Initial configuration	set corresponding bit of noise filter enable register 1 (NFEN1) to '1'. Configure Timer mode registers mn,mp of 2 channels (TMRmn, TMRmp) (confirm channel operation mode). Set master control channel Timer data register mn (TDRmn) configure output delay time, and set slave channel TDRmp register pulse width.	channel in operation stopped state (providing clock, consume portion of power)
	slave channel configuration set TOMmp bit of timer output mode register m(TOMm) to '1' (slave channel output mode). Configure TOLmp bit. Configure TOmp bit and confirm TOmp otuput initial voltage. Set TOEmp bit to '1', enable TOmp output. Set port regsiter and port mode regsiter to '0'.	T0mp pin in Hi-Z output state. When port mode register set to output mode and port register as '0', output T0mp initial configured voltage level. Because channel is in operation stopped state, thus T0mp remains unchange. T0mp pin output T0mp configured voltage level.

Fig6-65 Operating steps when the pulse output function is triggered single (2/2).



Note: m: Unit number (m=0,1) n: Master channel number (n=0, 2, 4, 6).

p: Slave channel number (m=0Time:n<p≤3, m=1Time:n<p≤7)

6.9.2 Operates as a PWM function

Pairing 2 channels generates pulses of any period and duty cycle. The period and duty cycle of the output pulse can be calculated using the following calculation equation:

$$\begin{aligned} \text{Pulse period} &= \{\text{TDRmn (master control) of the set value} + 1\} \times \text{counting time Clock cycle} \\ \text{Duty cycle}[\%] &= \{\text{TDRmp (dependent) of the set value}\} / \{\text{The setting of TDRmn (master) is} + 1\} \times 100 \\ \text{0\% output} &: \text{TDRmp (dependent) set} = 0000\text{H} \\ \text{100\% output} &: \text{TDRmp (dependent) set value} \geq \text{the setting value of } \{\text{TDRmn (master)} + 1\} \end{aligned}$$

Note: When the setpoint of TDRmp (slave) > the setpoint of {TDRmn (master) + 1}, the duty cycle exceeds 100%, but it is 100% output.

The master channel is used as an interval timer mode. If the channel start trigger bit (TSMn) of the timer channel start register m(TSm) is placed "1", just output interrupt (INTTMmn), and then load the setpoint of the timer data register mn (TDRmn) into the timer count register mn(TCRmn), and the count is decremented by the counting 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. This run is then repeated before the channel stop trigger bit (TTmn) of the timer channel stop register m(TTm) is set to "1".

When used as a PWM function, the master channel counts down and counts as the PWM output (TOmp) cycle during the period of counting up to "0000H". The slave channel is used as a single count pattern. Starting with the INTMmn of the master channel, the value of the TDRmp register is loaded into the TCRmp register and counted down until "0000H". When "0000H" is counted, INTMmp is output and waits for the next start to trigger (INTMmn of the master channel).

When used as a PWM function, the slave channel counts down to the duty cycle of the PWM output (TOmp) during the count until "0000H".

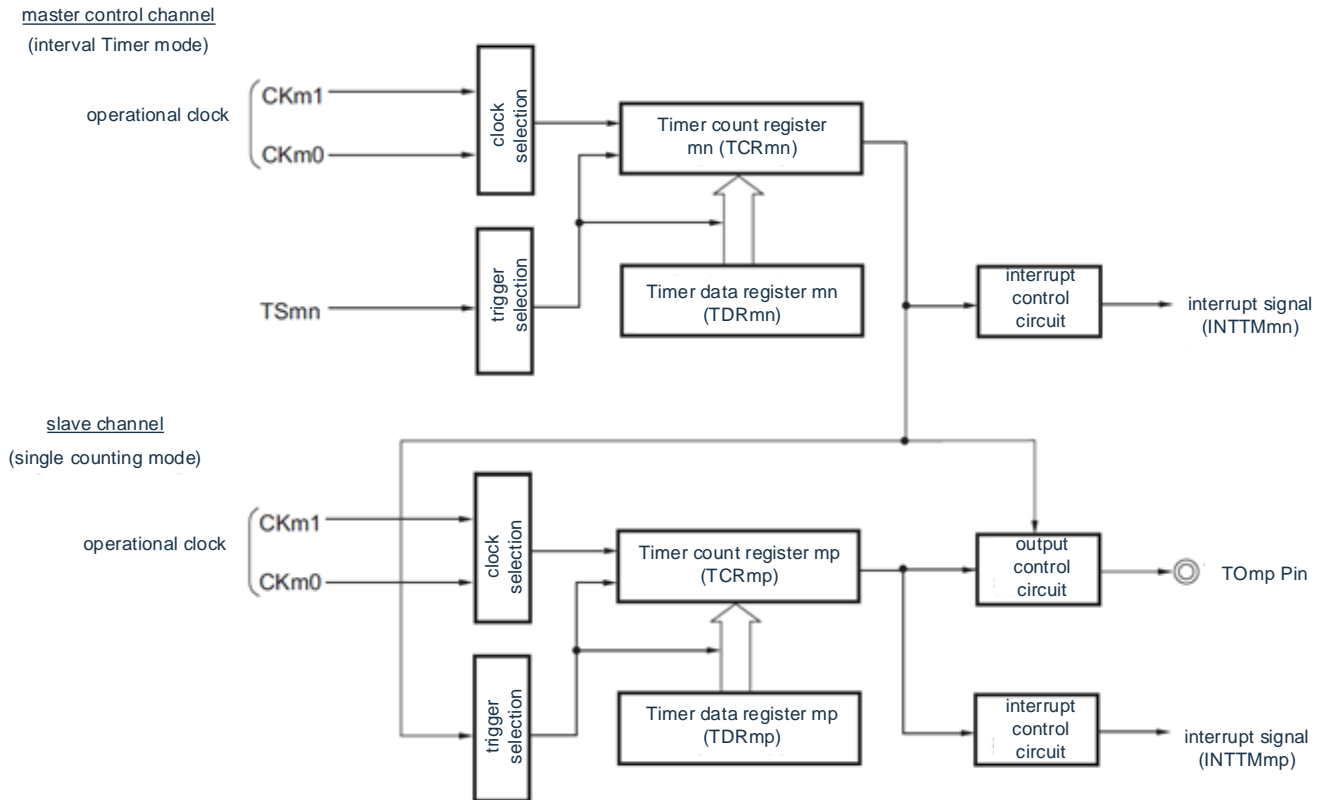
After generating INTMmn from the master channel and passing 1 clock, the PWM output (TOmp) becomes the effective level, and the value of the TCRmp register in the slave channel is "0000H" becomes invalid.

Remark: To overwrite both the timer data register mn (TDRmn) of the master channel and the TDRmp register of the slave channel, 2 write accesses are required. Because the values of the TDRmn registers and TDRmp registers are loaded into the TCRmn registers and TCRmp registers when the master channel generates INTTMmn, Therefore, if you rewrite it before and after the main control channel generates INTMmn, the TOmp pin cannot output the expected waveform. Therefore, to rewrite both the master's TDRmn registers and the slave's TDRmp registers, the two registers must be rewritten immediately after the master channel generates INTMmn.

Note: m: Unit number (m=0,1)n: Master channel number (n=0, 2, 4, 6).

p: Slave channel number (m=0Time:n<p≤3, m=1Time:n<p≤7)

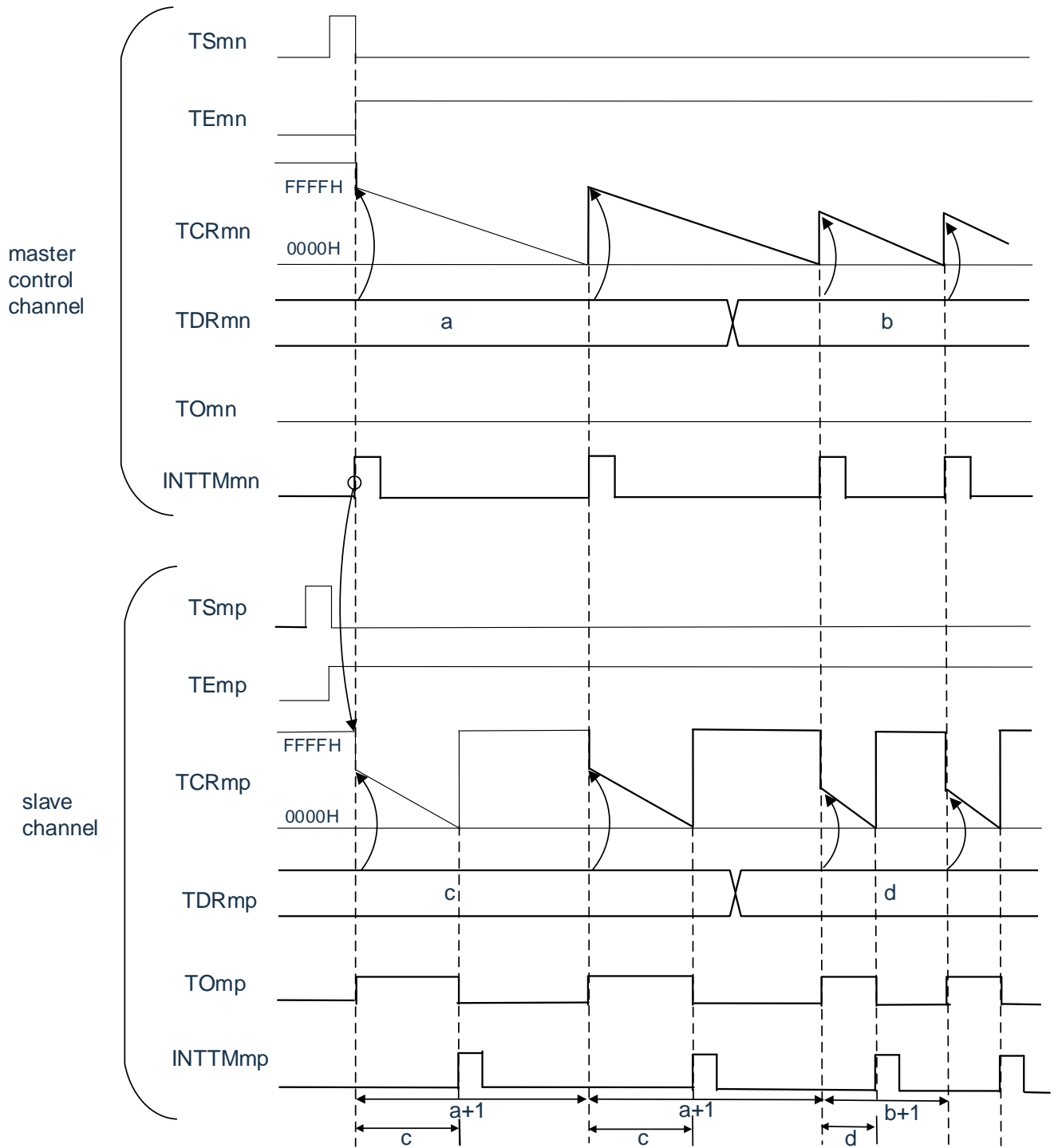
Figure 6-66 operating as a PWM function



Note: m: Unit number (m=0,1)n: Master channel number (n=0, 2, 4, 6).

p: Slave channel number (m=0Time:n<p≤3, m=1Time:n<p≤7)

Figure6-67 an example of the basic timing of the operation of the PWM function



Remarks: 1. m: Unit number (m=0,1)n: Master channel number (n=0, 2, 4, 6).

p: Slave channel number (m=0Time:n<p≤3, m=1Time:n<p≤7)

2. TSmn, TSmp: the timer channel starts the bitn, p of the register m (TSm).

TEmn, TEmmp: The timer channel enable bitn, p of the status register m (TEmp).

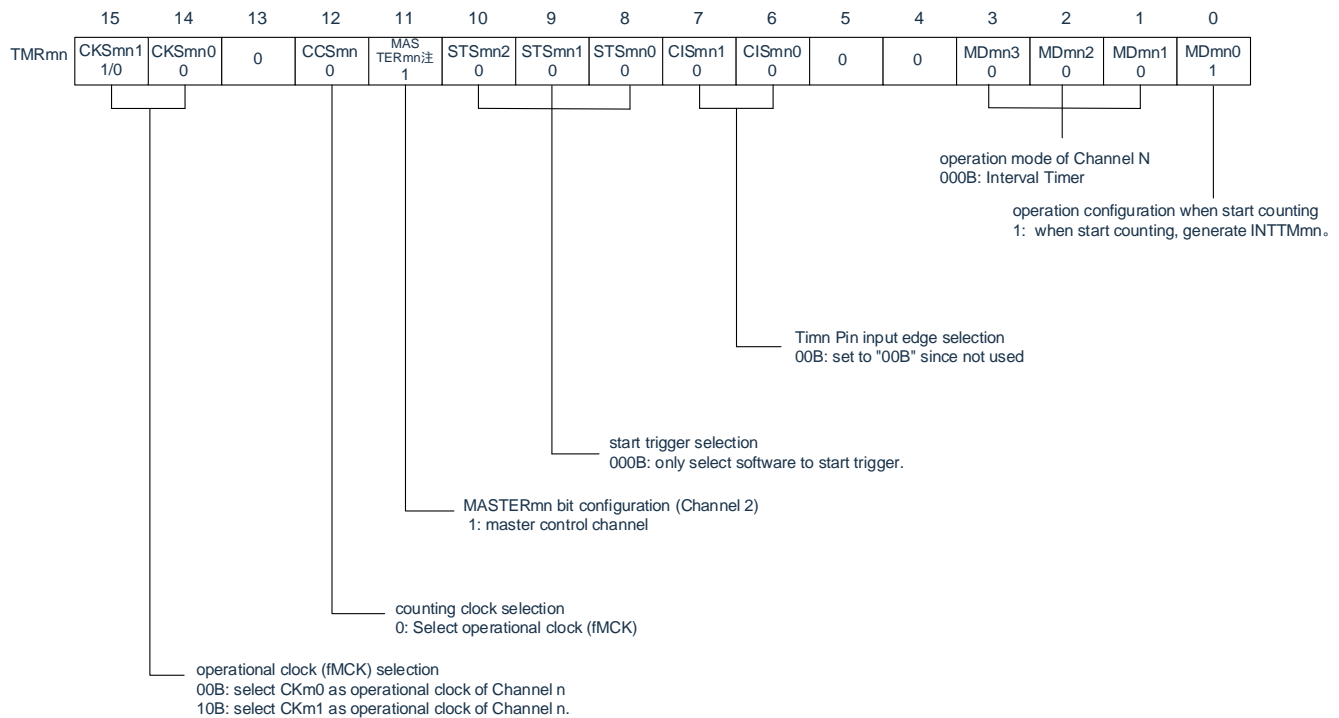
TCRmn, TCRmp: Timer count registers mn, mp (TCRmn, TCRmp).

TDRmn, TDRmp: timer data register mn, mp (TDRmn, TDRmp).

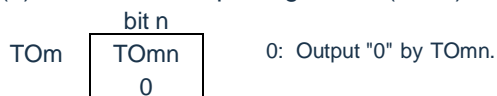
TOMn, TOMP: Output signals for TOMn pins and TOMP pins

Fig6-68 Example of register setting content during PWM function (master channel).

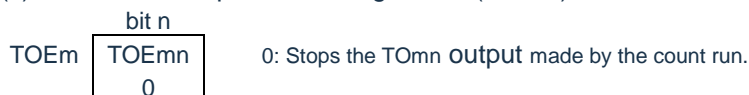
(a) Timer mode register mn (TMRmn).



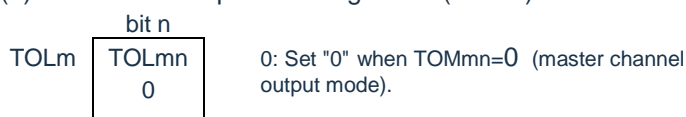
(b) The timer output register m (TOM).



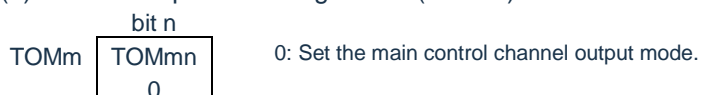
(c) The timer output enable register m (TOEm).



(d) The timer output level register m(TOLm).



(e) Timer output mode register m (TOMm).



Conce
ntrate

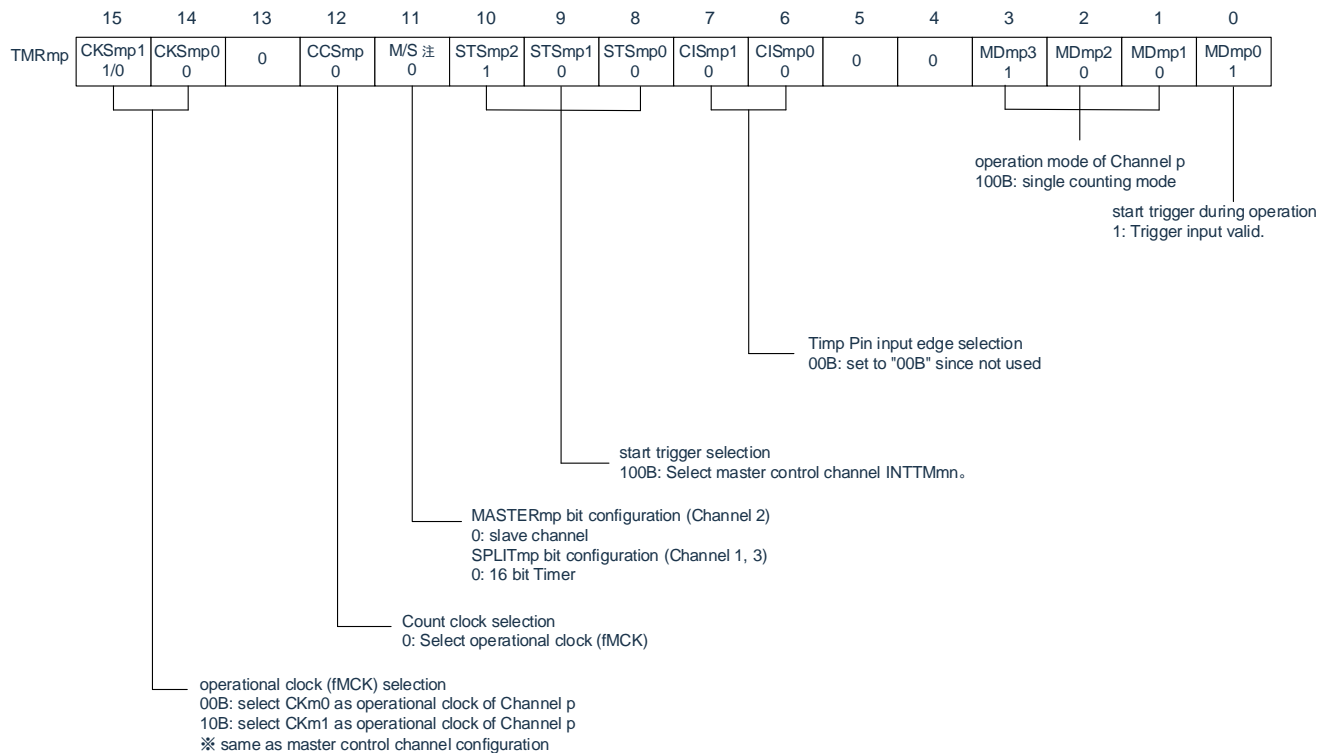
TMRm2 : MASTERmn=1

TMRm0 : Fixed to "0".

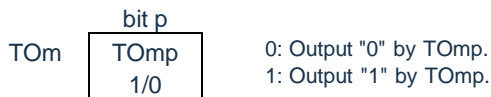
Note: m: Unit number (m=0,1) n: Master channel number (n=0, 2, 4, 6).

Fig 6-69 Example of register setting content during PWM function (slave channel).

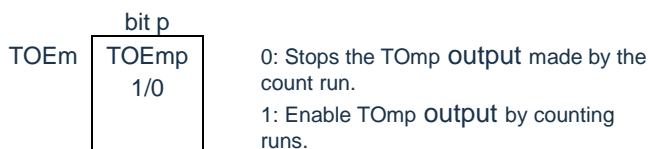
(a) Timer mode register mp (TMRmp).



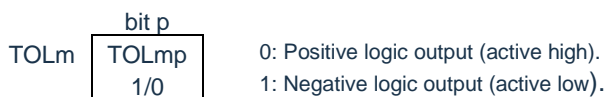
(b) The timer output register m (TOM).



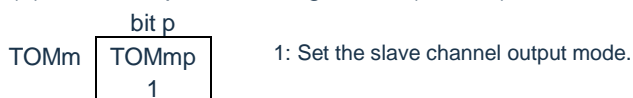
(c) The timer output enable register m (TOEm).



(d) The timer output level register m(TOLm).



(e) Timer output mode register m (TOMm).



Remark: TMRm2, TMRm4, TMRm6 : MASTERmp bit

TMRm1, TMRm3: SPLITmp bit

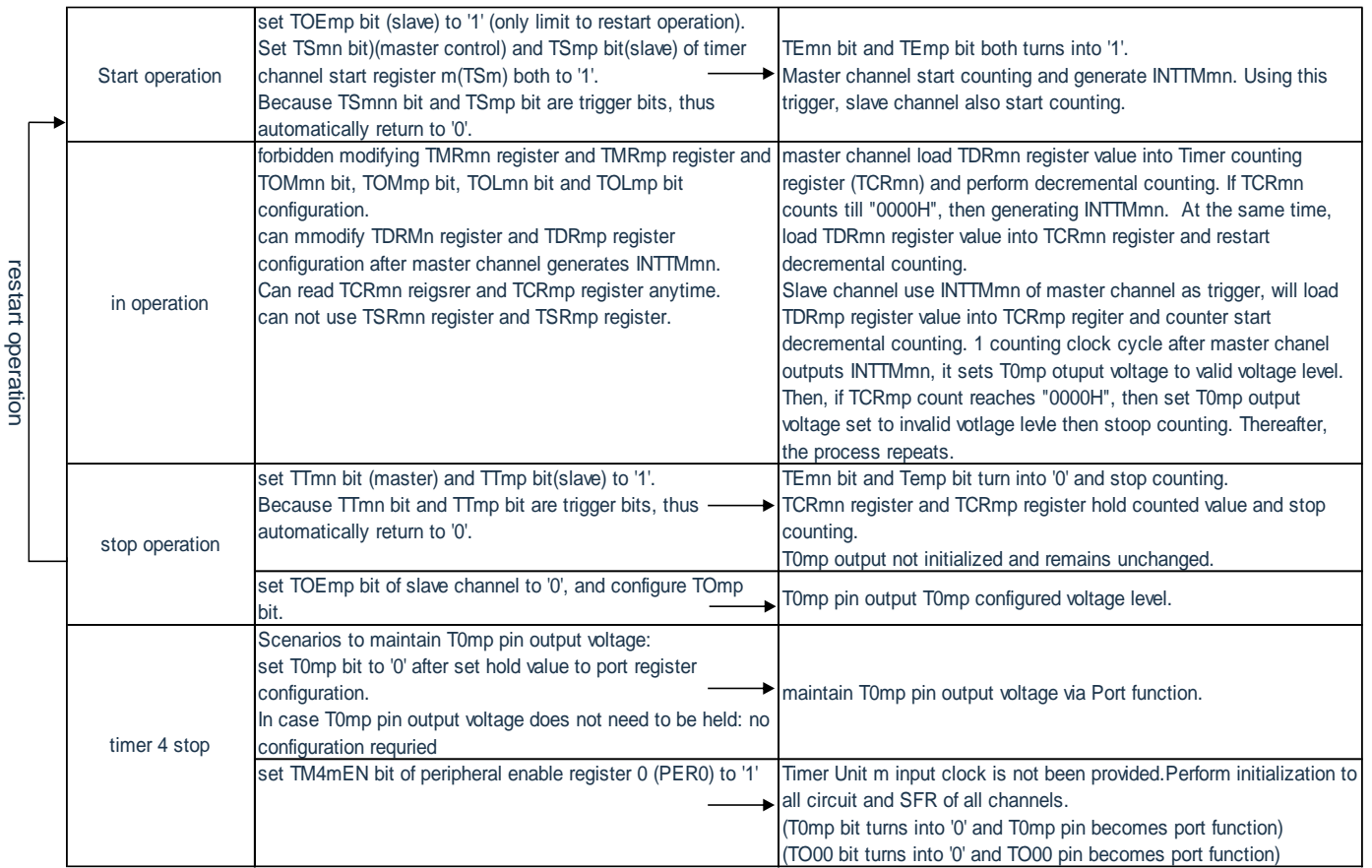
Note: m: Unit number (m=0,1) n: Master channel number (n=0, 2, 4, 6).

p: Slave channel number (m=0Time:n<p≤3, m=1Time:n<p≤7)

Figure 6-70 Operating steps when the PWM function is performed (1/2).

Timer 4 initial configuration		Timer Unit m input clock is in stopped state (stop providing clock, not able to write into registers)
	set TM4mEN bit of peripheral enable register 0 (PER0) to '1' →	Timer Unit m input clock is in active state, all channels in operation stopped state.
	configure Timer clock selection register m(TPSm), confirm CKm0~CKm3 clock frequency	
Channel Initial configuration	configure using timer mode register mn,mp (TMRmn,TMRmp) of 2 channels (confirm channel operation mode). Configure interval(period) value of Timer data register mn (TDRmn) of master control channel, and configure duty-cycle of slave channel TDRmp.	channel in operation stopped state (providing clock, consume portion of power)
	slave channel configuration set TOMmp bit of timer output mode register m(TOMm) to '1' (slave channel output mode). Configure TOLmp bit. Configure TOmp bit and confirm TOmp output initial voltage. Set TOEmp bit to '1', enable TOmp output. Set port register and port mode register to '0'.	T0mp pin in Hi-Z output state. When port mode register set to output mode and port register as '0', output T0mp initial configured voltage level. Because channel is in operation stopped state, thus T0mp remains unchange. T0mp pin output T0mp configured voltage level.

Figure 6-71 Operation steps while the PWM function is performed (2/2).



Note: m: Unit number (m=0,1) n: Master channel number (n=0, 2, 4, 6).

p: Slave channel number (m=0Time:n<p≤3, m=1Time:n<p≤7)

6.9.3 Operation as a multi-PWM output function

This is the function of extending the PWM function and using multiple slave channels for multiple PWM outputs with different duty cycles.

For example, when 2 slave channels are used in pairs, the period and duty cycle of the output pulse can be calculated using the following equation:

$$\begin{aligned} \text{Pulse period} &= \{\text{TDRmn (master control) of the set value} + 1\} \times \text{counting time Clock cycle} \\ \text{Duty cycle 1[\%]} &= \text{setpoint for } \{\text{TDRmp (slave 1)}\} / \{\text{TDRmn (master Control) of the set value} + 1\} \times 100 \\ \text{Duty cycle 2[\%]} &= \text{setpoint for } \{\text{TDRmq (slave 2)}\} / \{\text{TDRmn (primary Control) of the set value} + 1\} \times 100 \end{aligned}$$

Note: When $\text{TDRmp}(\text{Subordinate1}) \text{ setpoint} > \{\text{TDRmn The setpoint of (master).} + 1\}$ or $\{\text{TDRmq}(\text{Subordinate2}), \text{ the setpoint}\} > \{\text{TDRmn The setpoint of (master).} + 1\}$ when the duty cycle is exceeded 100%, but for 100% Output.

In interval timer mode, the timer count register mn (TCRmn) of the master channel runs and counts the cycles. In single-pass count mode, the TCRmp register of slave channel 1 runs and counts the duty cycle and outputs the PWM waveform from the TOmp pin. Starting with the INTMmn of the master channel, the value of the timer data register mp (TDRmp) is loaded into the TCRmp register and decremented. If TCRmp becomes "0000H", INTMmp is output and counts are stopped before the input next starts triggering (INTMmn of the master channel). After generating INTMmn from the master channel and passing through a count clock, the output level of TOmp becomes the effective level if TCRmp becomes "0000H", which becomes invalid.

Like the TCRmp register for slave channel 1, in single count mode, the TCRmq register of slave channel 2 runs and counts the duty cycle and outputs PWM from the TOmq pin Waveform. Starting with the INTMmn of the master channel, the value of the TDRmq register is loaded into the TCRmq register and the count is decremented. If TCRmq becomes "0000H", INTMmq is output and counts are stopped before the input next starts triggering (INTMmn of the master channel). After generating INTMmn from the master channel and going through a count clock, the output level of TOmq becomes effective if TCRmq becomes "0000H", which becomes invalid.

When channel 0 is used as the master channel by such operation, up to three PWM signals can be output simultaneously.

Note: To rewrite both the timer data register mn (TDRmn) of the master channel and the TDRmp register of slave channel 1 at the same time, at least 2 write accesses are required. Because the values of the TDRmn registers and TDRmp registers are loaded into the TCRmn registers and TCRmp registers when the master channel generates INTTMmn, Therefore, if the intTMmn is rewritten before and after the main control channel is generated, the TOmp pin cannot output the expected waveform. Therefore, to rewrite both the master's TDRmn register and the slave's TDRmp register at the same time, the two registers must be rewritten immediately after the master channel generates intTMmn (the same applies to the slave channel). 2 TDRmq registers).

Note: m: Unit number (m=0,1)n: Master channel number (n=0, 2, 4).

p: Slave channel number q: Slave channel number

m=0Time: $n < p < q \leq 3$ (p和q is greater than n integer)

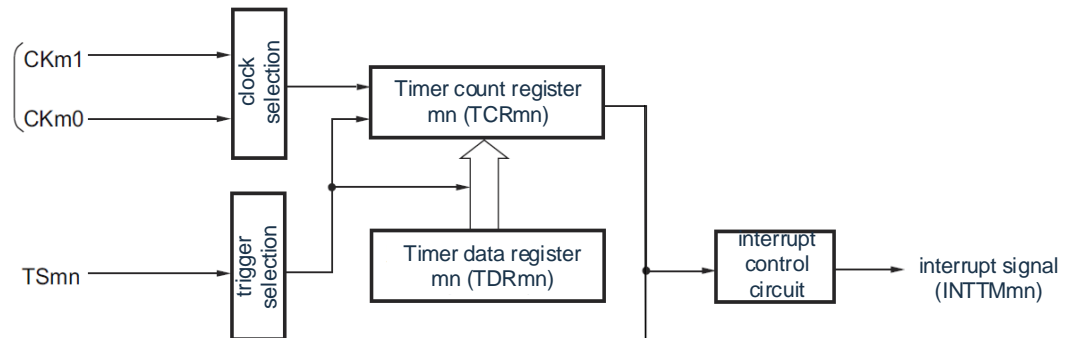
m=1Time: $n < p < q \leq 7$ (p和q is greater than n integer)

Figure 6-72 Block diagram of operation as a multi-PWM output function (in the case of outputting two types of PWM).

master control channel

(interval Timer mode)

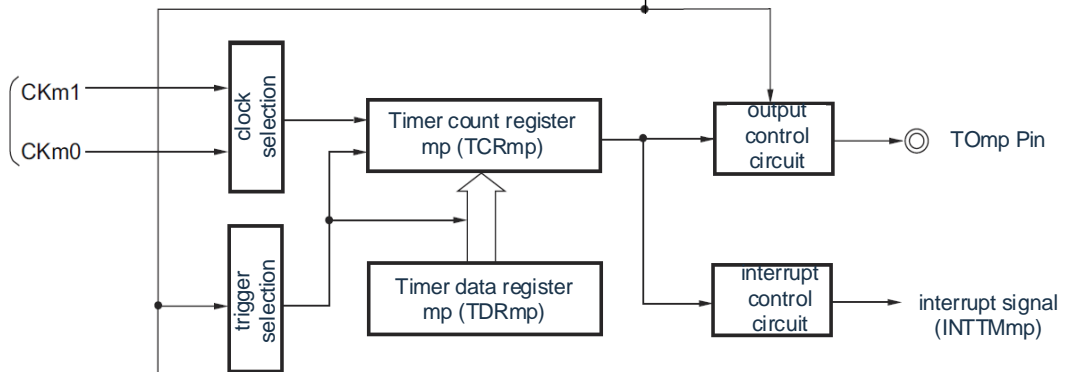
operational clock



slave channel1

(single counting mode)

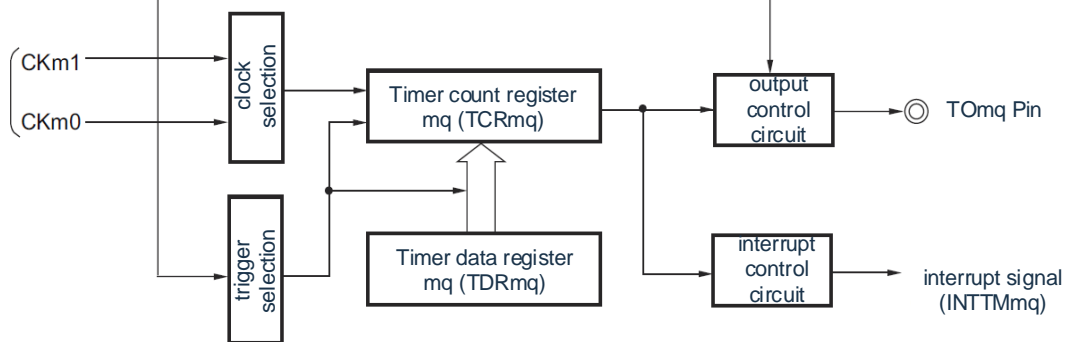
operational clock



slave channel2

(single counting mode)

operational clock



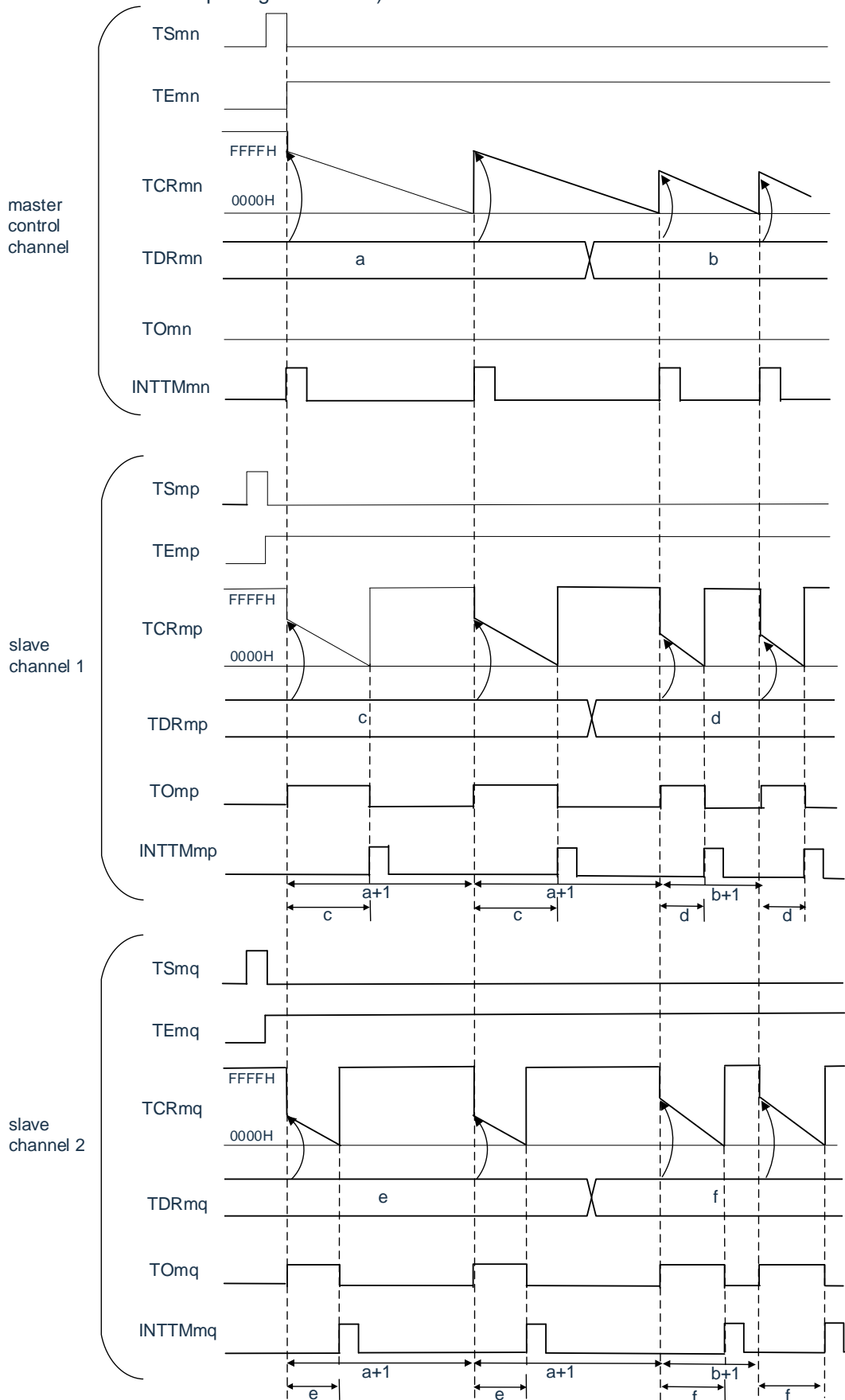
Note: m: Unit number (m=0,1) n: Master channel number (n=0, 2, 4).

p: Slave channel number q: Slave channel number

m=0Time: $n < p < q \leq 3$ (p和q is greater than n integer)

m=1Time: $n < p < q \leq 7$ (p和q is greater than n integer)

Figure 6-73 an example of running a basic timing function for multiple PWM output functions (in the case of outputting two PWMs).



Note: 1, m: unit number (m=0,1)n: master channel number (n=0, 2, 4).

p: Slave channel number q: Slave channel number

m=0Time: $n < p < q \leq 3$ (p和q is greater than integer)

m=1Time: $n < p < q \leq 7$ (p和q is greater than integer)

2, TSmn, TSmp, TSmq: timer channel start register m (TSm) of bitn, p, q

TEmn, TEmq: Timer channels enable bitn, p, and bitn of the status register

m(TEm). q

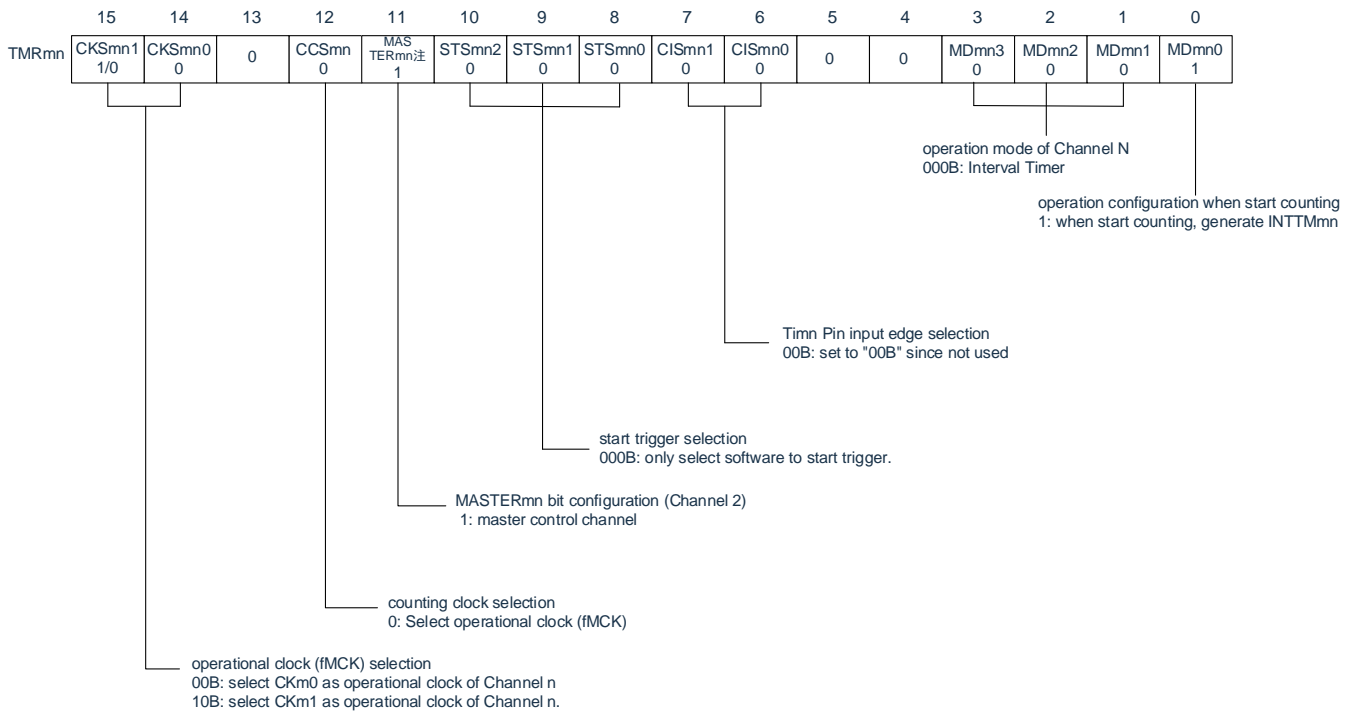
TCRmn, TCRmp, TCRmq: Timer count registers mn, mp, mq (TCRmn, TCRmp, TCRmq)

TDRmn, TDRmp, TDRmq: Timer data registers mn, mp, mq (TDRmn, TDRmp, TDRmq)

TOmn, TOmp, TOmq: Output signals from the TOmn, TOmp, TOmq pins

Fig. 6-74 multi-PWM output function (master channel).

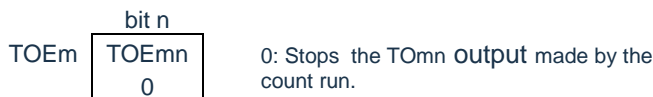
(a) Timer mode register mn (TMRmn).



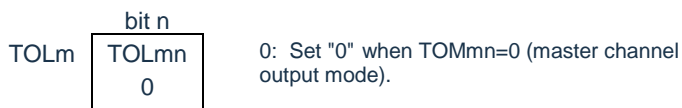
(b) The timer output register m (TOM).



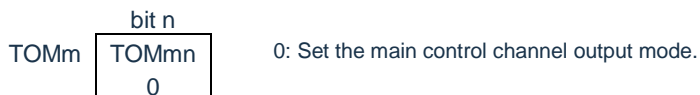
(c) The timer output enable register m (TOEm).



(d) The timer output level register m(TOLm).



(e) Timer output mode register m (TOMm).

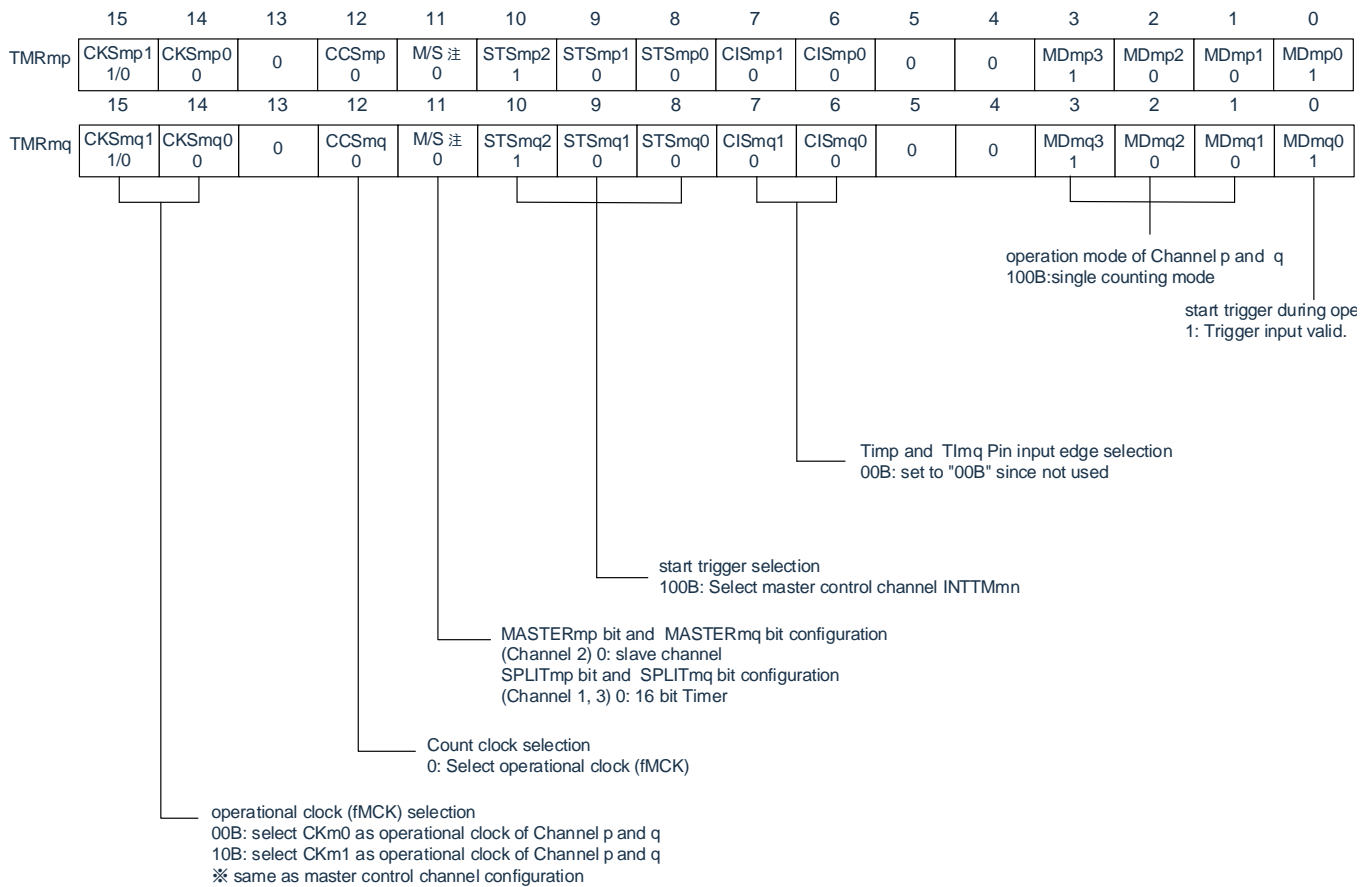


conce TMRm2, TMRm4 : MASTERmn=1
TMRm0, TMRm5, TMRm7 : Fixed to "0".

Note: m: Unit number (m=0,1) n: Master channel number (n=0, 2, 4).

Fig. 6-75 multiple PWM output function (slave channel) (in the case of outputting two PWMs).

(a) Timer mode registers mp, mq (TMRmp, TMRmq).



(b) The timer output register m (TOM).

TOM	bit q	bit p	
	TOmq 1/0	TOmp 1/0	
	0: Output "0" by TOmp and TOmq. 1: Output "1" by TOmp and TOmq.		

(c) The timer output enable register m (TOEm).

TOEm	bit q	bit p	
	TOEmq 1/0	TOEmp 1/0	
	0: Stops the TOmp and TOmq outputs performed by the count run. 1: Enable TOmp and TOmq output by counting runs.		

(d) The timer output level register m(TOLm).

TOLm	bit q	bit p	
	TOELq 1/0	TOELp 1/0	
	0: Positive logic output (active-high). 1: Negative logic output (active low).		

(e) Timer output mode register m (TOMm).

TOMm	bit q	bit p	
	TOMLq 1	TOMLp 1	
	1: Set the slave channel output mode.		

Note: TMRm2, TMRm4 : MASTERmp bit, MASTERmq bit

TMRm1, TMRm3: SPLITmp bit, SPLITmq bit

Note: m: Unit number (m=0,1) n: Master channel number (n=0, 2, 4).

p: Slave channel number q: Slave channel number

m=0Time:n < p < q ≤ 3 (p和q is greater thanninteger)

m=1Time:n < p < q ≤ 7(p和q is greater thanninteger)

Fig. 6-76 the multiple PWM output function (in the case of outputting 2 PWMs) (1/2).

	software operation	hardware state
Timer 4 initial configuration		Timer Unit m input clock is in stopped state (stop providing clock, not able to write into registers)
	set TM4mEN bit of peripheral enable register 0 (PER0) to '1' →	Timer Unit m input clock is in active state, all channels in operation stopped state.
	configure Timer clock selection register m(TPSm), confirm CKm0~CKm3 clock frequency	
Channel Initial configuration	configure using timer mode register mn,mp (TMRmn,TMRmp) of 2 channels (confirm channel operation mode). Configure interal(period) value of Timer data register mn (TDRmn) of master control channel, and configure duty-cycle of slave channel TDRmp.	channel in operation stopped state (providing clock, consume portion of power)
	slave channel configuration set TOMmp bit and TOLmq bit of timer output mode register m(TOMm) to '1' (slave channel output mode). Configure TOLmp and Tomq bit to '0'. →	T0mp pin in Hi-Z output state.
	Configure TOmp bit and Tomq bit, confirm TOmp and Tomq output initial voltage. →	When port mode register set to output mode and port register as '0', output T0mp and T0mq initial configured voltage level.
	Set TOEmp bit and TOEmq to '1', enable TOmp and Tomq output. →	Because channel is in operation stopped state, thus T0mp and T0mq remains unchange. T0mp pin and T0mq pin output T0mp and T0mq configured voltage level.
	Set port regsiter and port mode regsiter to '0'. →	

Fig. 6-77 multiple PWM output functions (in the case of outputting two kinds of PWM) (2/2).

restart operation	Start operation	(only during restart operation, TOEmp bit and TOEmq bit (slave) will set to '1'). Set TSmn bit(master), TSmp bit and TSmq bit (slave) of timer channel start register m(TSm) all set to '1' at the same time. Because TSmn bit, TSmp and TSmq bit are all trigger bits, thus automatically return to '0'.	TEmn bit and Temp bit both turns into '1'. Master channel start counting and generate INTTMmn. Using this trigger, slave channel also start counting.
	in operation	forbidden modifying TMRmn register and TMRmp register and TOMmn bit, TOMmp bit, TOLmn bit and TOLmp bit configuration. can mmodify TDRMn register and TDRmp register configuration after master channel generates INTTMmn. Can read TCRmn reigsrer and TCRmp register anytime. can not use TSRmn register and TSRmp register.	master channel load TDRmn register value into Timer counting register (TCRmn) and perform decremental counting. If TCRmn counts till "0000H", then generating INTTMmn. At the same time, load TDRmn register value into TCRmn register and restart decremental counting. Slave channel 1 use INTTMmn of master channel as trigger, will load TDRmp register value into TCRmp register and counter start decremental counting. 1 counting clock cycle after master chanel outputs INTTMmn, it sets T0mp otuput voltage to valid voltage level. Then, if TCRmp count reaches "0000H", then set T0mp output voltage set to invalid vottage levele then stoop counting. Slave channel 2 use INTTMmn of master channel as trigger, will load TDRmq register value into TCRmq regiter and counter start decremental counting. 1 counting clock cycle after master chanel outputs INTTMmn, it sets T0mq otuput voltage to valid voltage level. Then, if TCRmq count reaches "0000H", then set T0mq output voltage set to invalid vottage levele then stoop counting. Thereafter, the process repeats.
	stop operation	set TTmn bit (master), TTmp bit and TTmq bit(slave) to '1'. Because TTmn bit, TTmp bit, TTmq bit are trigger bits, thus automatically return to '0'.	TEmn bit, Temp bit and Temq turn into '0' and stop counting. TCRmn, TCRmp TCRmq registers hold counted value and stop counting. T0mp and T0mq output not initialized and remains unchanged.
		set TOEmp bit and TOEmq bit of slave channel to '0', and configure T0mp and T0mq bit.	T0mp pin and T0mq pin output T0mp and T0mq configured voltage level.
	timer 4 stop	Scenarios to maintain T0mp pin and Tomq pin output voltage: set T0mp bit and Tomq bit to '0'. In case T0mp pin and Tomq output voltage does not need to be held: no configuration requiried	maintain T0mp pin and Tomq output voltage via Port function.
		set TM4mEN bit of peripheral enable register 0 (PER0) to '1'	Timer Unit m input clock is not been provided.Perform initialization to all circuit and SFR of all channels. (T0mp bit and T0mq bit turn into '0' and T0mp pin and Tomq becomes port function) (TO00 bit turns into '0' and TO00 pin becomes port function)

Note: m: Unit number (m=0,1) n: Master channel number (n=0, 2, 4).
p: Slave channel number q: Slave channel number
m=0Time:n < p < q ≤ 3 (p和qis greater thanninteger)
m=1Time:n < p < q ≤ 7(p和qis greater thanninteger)

6.10 Considerations when using a universal timer unit

6.10.1 Considerations when using timer outputs

Depending on the product, pins assigned the timer output function may also be assigned to the output of other multiplexing functions. When using the timer output in this case, the output of the other multiplexing function needs to be set at the initial value.

For details, please refer to "Chapter 2 Pin Functions".

Chapter 7 Timer A

7.1 Function of timer A

Timer A is a 16-bit timer capable of measuring pulse outputs, pulse widths and cycles of external inputs, and counting external events.

The 16-bit timer consists of a reload register and a decrement counter, which are assigned to the same address. If you access the TA0 register, you can access the reload registers and counters.

The specifications and block diagrams of Timer A are shown in Table 7-1 and Figure 7-1, respectively.

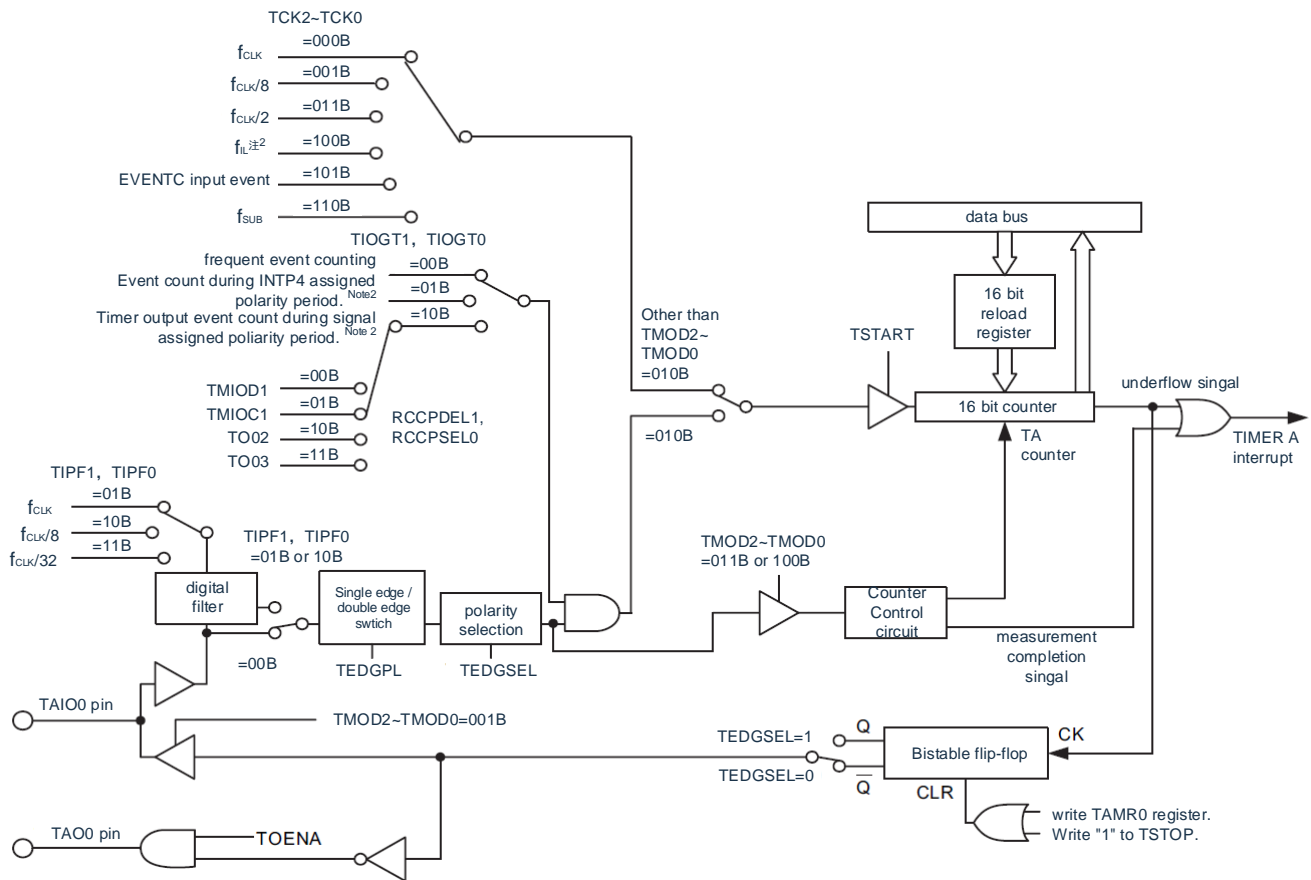
Table 7-1 Specifications for Timer A

Item		content
Operating mode	Timer mode	Count the count sources.
	Pulse output mode	The counting source is counted, and a pulse with opposite polarity is output when the timer underflow occurs.
	Event counter pattern	Count external events. It can also be operated 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 (Running Clock)		F_{CLK} , $f_{CLK}/2$, $f_{CLK}/8$, f_{IL} , f_{SUB} or EVENTC input events.
interrupt		<ul style="list-style-type: none"> • When the counter underflows • End of effective width measurement with external input (TAIO) in pulse width measurement mode • When entering the setting edge of the external input (TAIO) in pulse period measurement mode
Select Features		<ul style="list-style-type: none"> • Collaboration with EVENTC: Events entered by EVENTC can be selected as the count source.

7.2 Structure of timer A

The block diagram and pin structure of Timer A are shown in Figure 7-1 and Table 7-2, respectively.

Figure 7-1 Block diagram of Timer A



- Note 1 To select f_{IL} as the counting source, the subsystem clock must be supplied with the mode control register (OSMC) at WUTMMCK0 position "1". However, when f_{SUB} is selected as the counting source for the real-time clock or the 15-bit interval timer, f_{IL} cannot be selected as the timer A The count source.
2. RCCPSEL2 bit selectable polarity through the TAISR0 register.

Table 7-2 Timer A Pin Structure

Pin name	Input/Output	function
INTP4	input	Event counter mode control for timer A
TAIO Note	Input/Output	External event input and pulse output for timer A
TAO Note	output	Pulse output of timer A

Note the configuration of the TAO pin can be selected by the PIOR12 bit and the PIOR13 bit of the PIOR1 register, and the PIOR10 can be passed by the PIOR1 register Bits and PIOR11 bits select the configuration of the TAIO pins. For details, please refer to "Chapter 2 Pin Functions".

7.3 Controls the registers of timer A

The registers that control timer A are shown in Table 7-3.

Table 7-3 control registers of timer A

Register name	symbol
Peripheral I/O redirect register 1	PIOR1
Peripheral enable register 1	PER1
The subsystem clock provides a mode control register	OSMC
Timer A counts register 0 ^{note}	TA0
Timer A controls register 0	TACR0
Timer AI/O controls register 0	TAIOC0
Timer A mode register 0	TAMR0
Timer A event pin selects register 0	TAISR0
Port register x	Px
Port mode register x	PMx

Note When accessing the TA0 register, the CPU does not enter the processing of the next instruction but is in the waiting state of CPU processing. Therefore, when this wait occurs, the number of clocks executed by the instruction increases the number of clocks waited. The number of read and write wait clocks when accessing the TA0 register is 1 clock.

7.3.1 Peripheral enable register 1 (PER1).

The PER1 register is a register that is set to allow or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use.

To use timer A, bit0 (TMAEN) must be set to "1".

The PER1 register is set via the 8-bit memory operation instruction.

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

Figure 7-2 the format of peripheral enable register 1 (PER1)

Address: 4002081AH after reset: 00HR/W

symbol

PER1

7	6	5	4	3	2	1	0
DACEN	TMBAndN	PGACMPEN	TMMEN	DMAIn	PWMOPEN	TMCEN	TSTONE

TMAEN	Provides control of the input clock of timer A
0	Stop providing the input clock. <ul style="list-style-type: none"> • Cannot write SFR for timer A. • Timer A is reset.
1	An input clock is provided. <ul style="list-style-type: none"> • Can read and write the SFR used by timer A.

Note 1 To set timer A, you must first place the TMAEN position "1". When the TMAEN bit is "0", the write operation of the control register of timer A is ignored, and the read values are the initial values (port mode register PM x and port register Px).

7.3.2 The subsystem clock provides a mode control register (OSMC).

The operating clock of timer A can be selected by WUTMMCK0 bits.

The RTCLPC bit is a bit that reduces power consumption by stopping unwanted clock functions. For the setting of the RTCLPC bit, refer to "Chapter 4 Clock Generation Circuit".

Set the OSMC registers via the 8-bit memory operation instructions. After generating a reset signal, the value of this register changes to "00H".

Figure 7-3 the format of the mode control register (OSMC) provides for the sub-system clock

Address: 40020423H after reset: 00HR/W

Symbol	7	6	5	4	3	2	1	0
OSMC	RTCLPC	0	0	WUTMMCK0	0	0	0	0

WUTMMCK0	Choice of real-time clock, 1 5-bit interval timer operating clock (f_{RTC}) and timer A operating clock
0	Subsystem clock (f_{SUB}). <ul style="list-style-type: none"> The secondary system clock is a real-time clock and a 1 5-bit interval timer for the operating clock. The low-speed internal oscillator cannot be selected as the counting source for timer A.
1	Low-speed internal oscillator clock (f_{LL}). <ul style="list-style-type: none"> The low-speed internal oscillator clock is a real-time clock and a 15-bit interval timer for the operating clock. Low-speed internal oscillator or subsystem clock can be selected as the counting source for timer A.

7.3.3 Timer A counts register 0 (TA0).

This is the 16-bit register. If you write this register, you write the data to the reload register. If you read this register, you read the count value. The state of reloading registers and counters varies depending on the value of the TSTART bit of the TARR0 register. For details, please refer to "Rewriting of Registers and Counters in 7.4.1".

The TA0 register is set via the 16-bit memory operation instruction. After generating the reset signal, the value of the TA0 register changes to "FFFFH".

Figure 7-4 Format of timer A count register 0 (TA0).

Address: 40042300H reset: FFFFHR/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TA0																

—	function	Set the scope
bit15~0	16-bit counter <small>Notes 1, 2</small>	0000H~FFFFH

Note 1 If you write "1" to the TSTOP bit of the TARR0 register, the count of the 16-bit counter is forcibly stopped and the count value becomes "FFFFH".

2. If the setting value of the TCK2~TCK0 bit of the TAMR0 register is not "001B" ($f_{CLK}/8$) and "011B" ($f_{CLK}/2$) and the value of the TA0 register is "0000H", which is only oriented immediately after the start of counting DMA and EVENTC generate 1 request signal. However, TAO and TAO perform alternating outputs.

In event counter mode, regardless of the value of the TCK2~TCK0 bit, if the value of the TA0 register is "0000H", a 1 is generated to the DMA and EVENTC immediately after the start of counting The signal is requested, and the TAO outputs alternately even if the specified period is not counted.

If the value of the TA0 register is greater than or equal to "0001H", a request signal is generated each time the TA underflow occurs.

Note that when the TA0 register is accessed, the CPU does not enter the processing of the next instruction but is in a waiting state of CPU processing. Therefore, when this wait occurs, the number of clocks executed by the instruction increases the number of clocks waited. The number of read and write wait clocks when accessing the TA0 register is 1 clock.

7.3.4 Timer A controls register 0 (TACR0).

The TACR0 register is the register that controls the count and stop of register A and the state of timer A.

Set the TARR0 register via the 8-bit memory operation instruction.

After generating a reset signal, the value of the TACR0 register changes to "00H".

Figure 7-5 Timer A controls the format of register 0 (TACR0).

Address: 40042240H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
TACR0	0	0	TUNDF	TEDGF	0	TSTOP	TCSTF	TSTART

TUNDF	Underflow flag for timer A
0	No underflow occurred.
1	Underflow occurs.
[condition for "0"]. • When this bit is written "0" through the program [condition for "1"]. • When the counter underflows	

TEDGF	Detection flag for a valid edge
0	There are no valid edges.
1	There are valid edges.
[condition for "0"]. • When this bit is written "0" through the program [condition for "1"]. • End of effective width measurement with external input (TAIO) in pulse width measurement mode • When entering the setting edge of the external input (TAIO) in pulse period measurement mode	

TSTOP	The count of timer A forces the stop ^{note 1}
If you write "1" to this bit, you force the count to stop. The read value is "0".	

TCSTF	The count status flag for timer A ^{is noted 2}
0	Stop count.
1	Counting.
[condition for "0"]. • When writing "0" to the TSTART bit (which becomes "0" synchronously with the count source). • When writing "1" to the TSTOP bit [condition for "1"]. • When the TSTART bit is written "1" (which becomes "1" synchronously with the count source).	

TSTART	The count of timer A starts with ^{Note 2}
0	Stop count.
1	Start counting.
Start counting by writing "1" to the TSTART bit; Stop counting by writing "0" to the TSTART bit. If you place the TSTART position "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 "Start and Stop Control of 7.5.1 Count".	

Note: 1 If you write "1" (force stop count) to the TSTOP bit, both the TSTART bit and the TSTF bit are initialized at the same time, and the pulse output level is initialized.

2. For precautions when using TSTART bits and TCSTF bits, please refer to "Start and Stop Control of 7.5.1 Count".

7.3.5 Timer AI/O control register 0 (TAIOC0).

The TAIOC0 register is the register that sets the input/output of timer A. The TAIOC0 register is set via the 8-bit memory operation instruction.

After generating a reset signal, the value of the TAIOC0 register changes to "00H".

Figure 7-6 Format of timer AI/O control register 0 (TAIOC0).

Address: 40042After 241H

reset: 00HR/W

Symbol 76543210

TAIOC0	TIOGT1	TIOGT0	TIPF1	TIPF0	0	TOENA	0	TEDGSEL
--------	--------	--------	-------	-------	---	-------	---	---------

TIOGT1	TIOGT0	Taio's count control notes 1 and 2
0	0	Events are always counted.
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.
Other than the above		Prohibit settings.

TIPF1	TIPF0	Selection of TAIO input filters
0	0	There is no filter.
0	1	There is a filter that samples via the f_{CLK} .
1	0	There is a filter that samples via $f_{CLK}/8$.
1	1	There is a filter that samples 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	ALLOW FOR TAO output
0	Disable TAO output (port).
1	Allow TAO output.

TEDGSEL	Polarity switching of input/output
Functions vary depending on the mode of operation (see Tables 7-4 and 7-5).	

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.

2. TiogT0 bits and TIOGT1 bits are only valid in event counter mode.

Table 7-4 Edge and polarity switching of TAIO input/output

Operating mode	function
Timer mode	Not used (input/output ports).
Pulse output mode	0: Output from the "H" level (initial level: "H"). 1: Output from the "L" level (initial level: "L").
Event counter pattern	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 7-5 Polarity switching of TAO output

Operating mode	function
All mode	0: Output from the "L" level (initial level: "L"). 1: Output from the "H" level (initial level: "H").

7.3.6 Timer A controls register 0 (TAMR0).

The TAMR0 register is the register that sets the operating mode of register A. The TAMR0 register is set via the 8-bit memory operation instruction.

After generating the reset signal, the value of the TAMR0 register changes to "00H".

Figure 7-7 Timer A controls the format of register 0 (TAMR0).

Address: 40042After 242H

reset: 00H R/W

Symbol 76543210

TAMR0	0	TCK2	TCK1	TCK0	TEDGPL	TMOD2	TMOD1	TMOD0
-------	---	------	------	------	--------	-------	-------	-------

TCK2	TCK1	TCK0	The counting source for timer A is selected ^{note 1 and 2}
0	0	0	f_{CLK}
0	0	1	$f_{CLK}/8$
0	1	1	$f_{CLK}/2$
1	0	0	f_{IL}
1	0	1	EventC input event
1	1	0	f_{SUB}
Other than the above			Prohibit settings.

TEDGPL	TAIO edge polarity selection ^{note 5}
0	One edge
1	Bilateral along

TMOD2	TMOD1	TMOD0	Timer A operating mode select ^{Note 3}
0	0	0	Timer mode
0	0	1	Pulse output mode
0	1	0	Event counter pattern
0	1	1	Pulse width measurement mode
1	0	0	Pulse period measurement mode
Other than the above			Prohibit settings.

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.

- You cannot switch the counting source during the counting process. If you want to switch the count source, you must have "0" in both the TSTART bit and the TSTF bit in the TACR0 register (Stop Counting) when toggling.
- The operating mode can only be changed when the stop count (both the TSTART bit and the TSTF bit of the TACR0 register are "0" (stop count)), and cannot be changed during the counting process.
- To select f_{IL} as the counting source, the subsystem clock must be supplied with the mode control register (OSMC) of the WUTMMCK0 position "1". However, when f_{SUB} is selected as the counting source for the real-time clock or the 12-bit interval timer, f_{IL} cannot be selected as the counting source for timer A
- The TEDGPL bit is only valid in event counter mode.
- Initialize the output of the TAO pin and the TAIO pin of timer A by writing the TAMR0 register. For output levels at initialization, refer to the format of figure 7-6 timer A/I/O control register 0 (TAIOC0). " description.

7.3.7 Timer A event pin select register 0 (TAISR0).

The TAISR0 register is a register that selects the timer that controls the event count in event counter mode and sets the polarity. The TAISR0 register is set via the 8-bit memory operation instruction.

After generating a reset signal, the value of the TAISR0 register changes to "00H".

Figure 7-8 Timer A event pin selects the format of register 0 (TAISR0).

Address: 40042After 243H

reset: 00HR/W

Symbol 76543210

TAISR0	0	0	0	0	0	RCCPSEL2 ^{Note}	RCCPSEL1 ^{Note}	RCCPSEL0 ^{Note}
--------	---	---	---	---	---	--------------------------	--------------------------	--------------------------

RCCPSEL2 ^{Note}	Timing output signal and selection of 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.

7.3.8 Port mode register x (PMx).

This is the register that sets the port input/output.

To use the multiplex port of the timer output pin (TAIO, TAO, etc.) as the output of the timer, the bit of the port mode register (PMxx) corresponding to each port and the position of the port register (Pxx) must be "0".

(Example) The case where P01 is used as a timer output TAIO, the PM01 position of the port mode register 0 is "0". Place the P01 position of port register 0 at "0".

To use the multiplexing port (P01/TAIO, etc.) of the timer input pin as the input to the timer, the position of the port mode register (PMxx) corresponding to each port must be "1". At this point, the bit of the port register (Pxx) can be "0" or "1".

(Example) The case where P01 is used as a timer input TAIO will place the PM01 position "1" of port mode register 0. Place the P01 position of port register 0 at "0" or "1".

Set the PMxx register via the 8-bit memory operation instruction. After generating a reset signal, the values of these registers become "FFH".

For the format of the port mode registers, refer to "PMxx, Pxx, Puxx, PIMxx, POMxx, PMxx, PMxx, PPMxx, PPMxx, PXX, PMCxx registers and their bits".

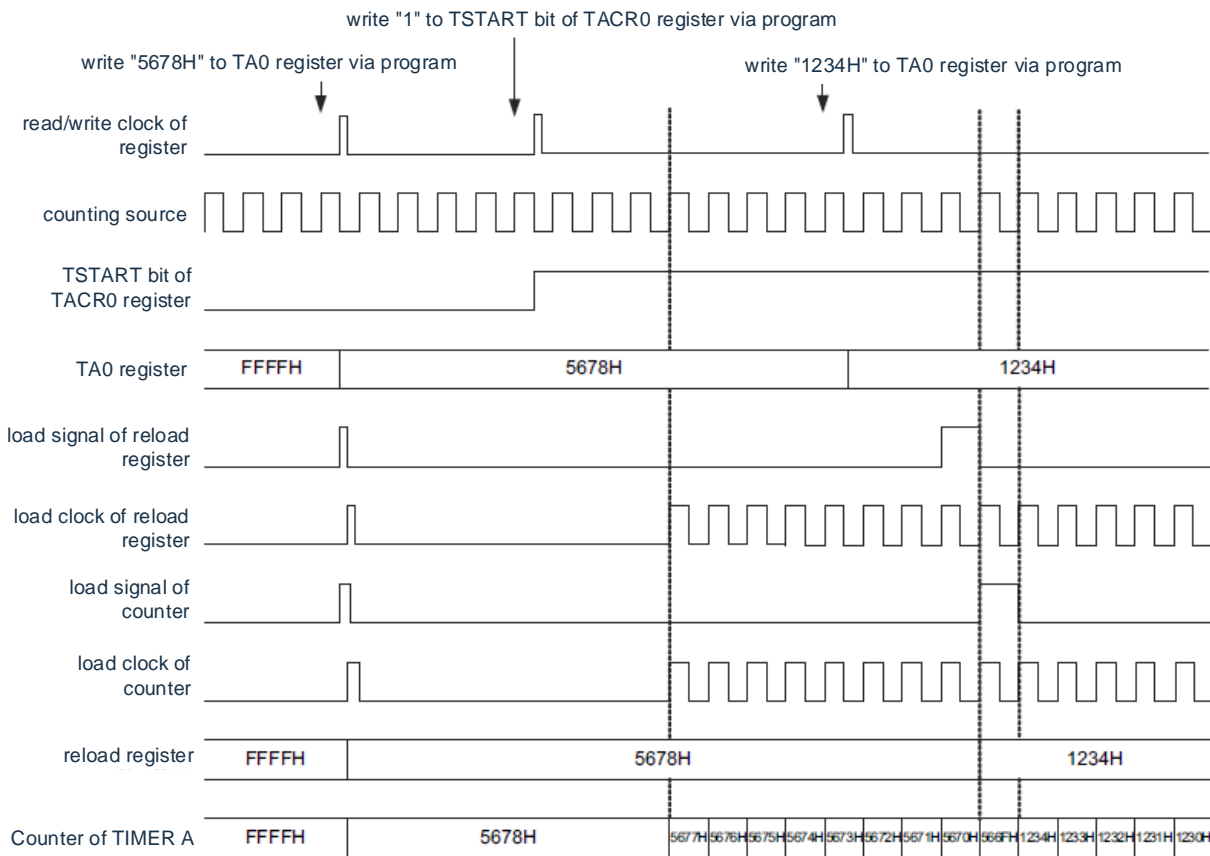
7.4 Operation of timer A

7.4.1 Overrides to reload registers and counters

Independent of the operating mode, the rewriting timing of reload registers and counters varies depending on the value of the TSTART bit of the TARR0 register. When the TSTART bit is "0" (stop count), the registers and counters are reloaded directly; When the TSTART bit is "1" (start counting), after the registers are reloaded synchronously with the count source, the counter is written synchronously with the next count source.

The rewrite timing diagram, determined by the value of the TSTART bit, is shown in Figure 7-9.

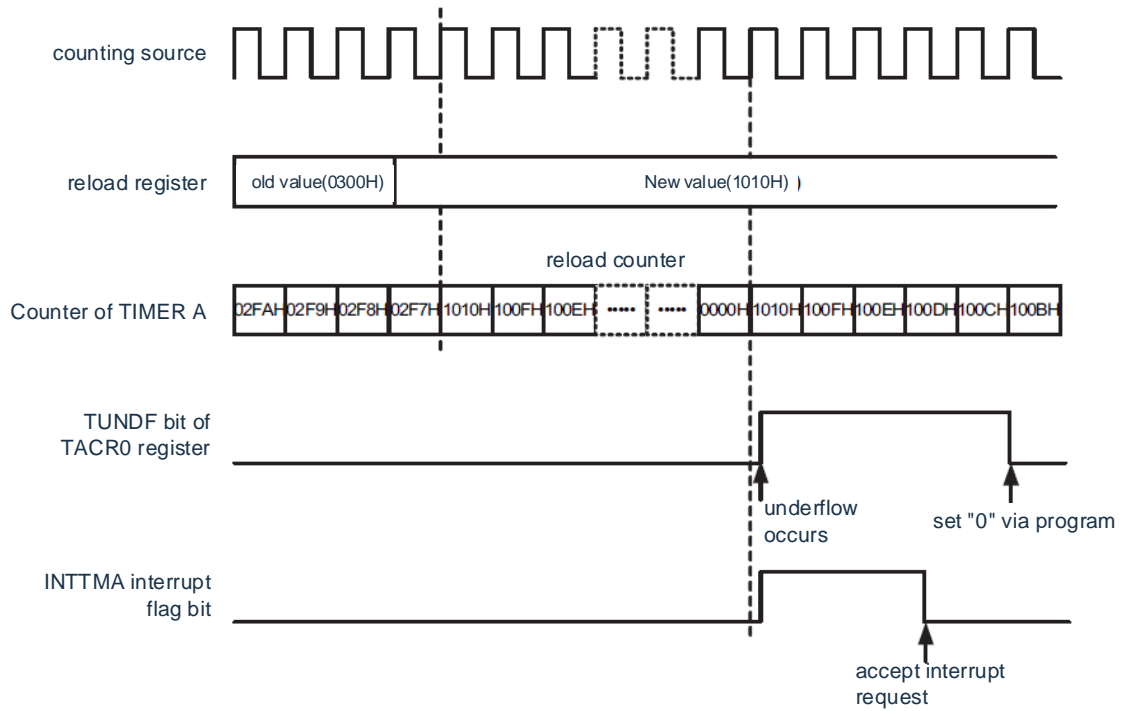
Figure 7-9 Rewritten timing diagram determined by the value of the TSTART bit



7.4.2 Timer mode

This is the mode of counting down by selecting a count source from TCK0 to TCK2 bits of the TAMR0 register. In timer mode, the count value is decremented by 1 whenever a count source is entered, and if the count value becomes "0000H" and the next count source is entered, an underflow occurs, and an interrupt request is generated. An example of the timer mode is shown in Figure 7-10.

Figure 7-10 An example of a timer mode



7.4.3 Pulse output mode

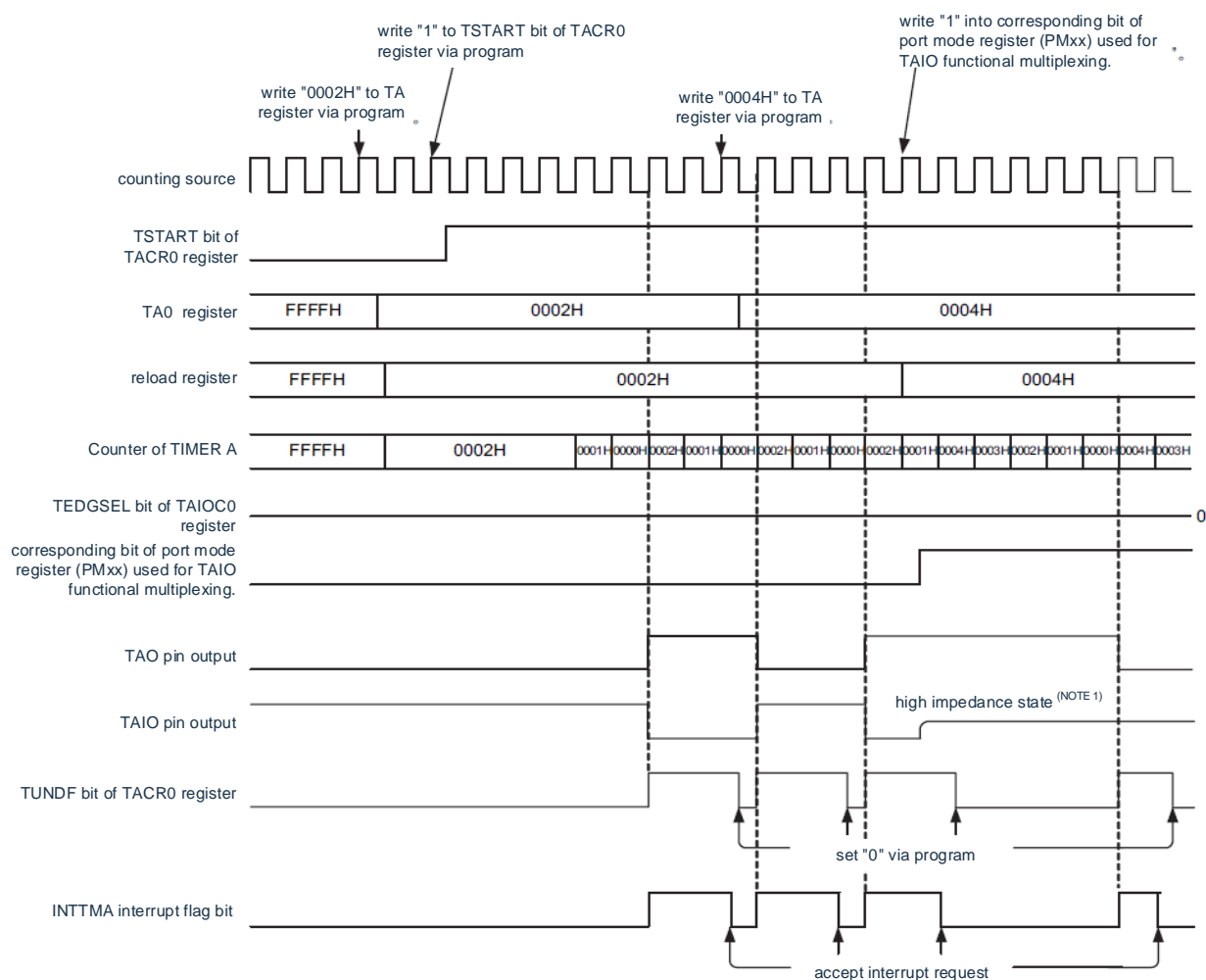
In this mode, the count is decremented through the TCK0~TCK2 bit selected count source of the TAMR0 register, and whenever an underflow occurs, the output level of the TAIO pin and the TAO pin is inverted.

In pulse output mode, the count value is decremented by 1 whenever the count source is entered, and if the count value becomes "0000H" and the next count source is entered, an underflow occurs, and an interrupt request is generated.

Pulses can be output from the TAO 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 TAI0C0 register. An example of the pulse output mode is shown in Figure 7-1.

Fig. 7-11 Example of operation of pulse output mode



NOTE 1: configure to high impedance state via port output enable control of the selected TAIO function.

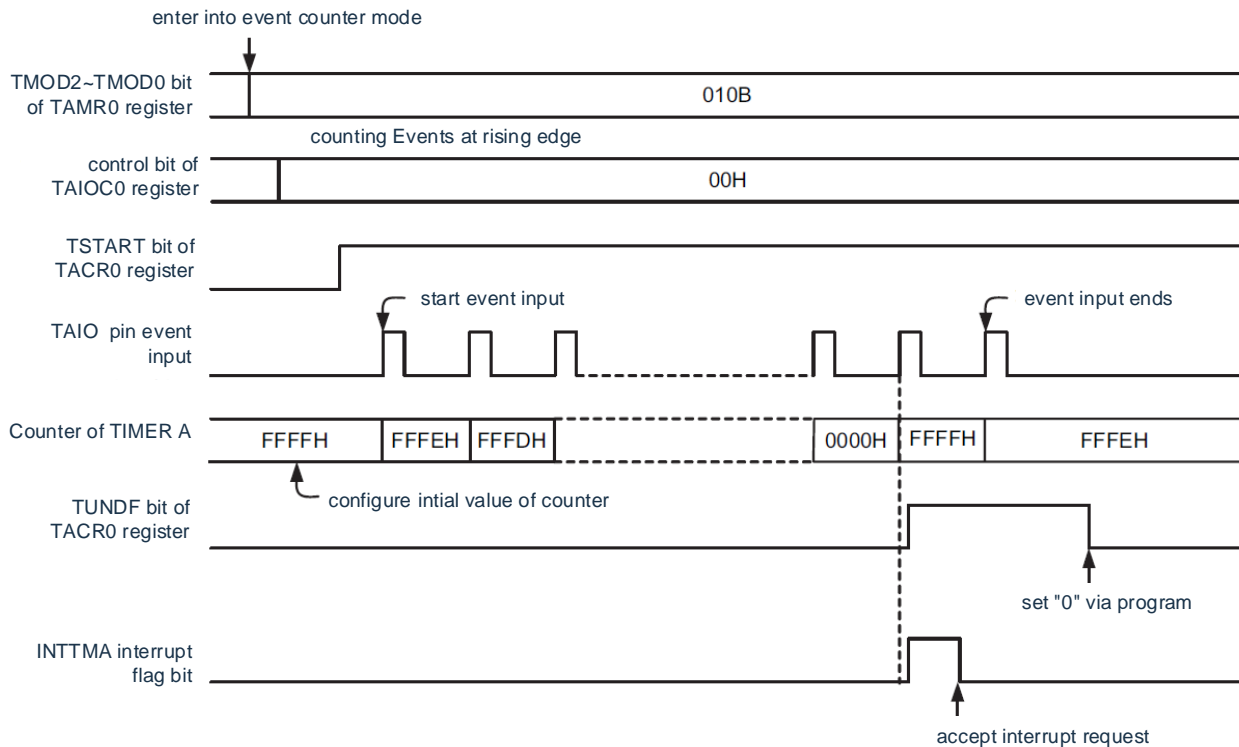
7.4.4 Event counter pattern

This is a mode of decrementing counting via an external event signal (count source) input by the TAIO pin.

I can pass tiPF0 of the TAIOC0 register through the TIOGT0 to TIOGT1 bit and TAISR0 registers for various settings during event counting and can pass the TIPF0 of the TAIOC0 register The \sim TIPF1 bit specifies the filter function of the TAIO input.

Alternating outputs can be performed even in the TAO pin in event counter mode. To use the event counter mode, refer to the "Setup Steps for 7.5.5 TAO Pins and TAIO Pins". Example 1 of the operation of the event counter pattern is shown in Figure 7-12.

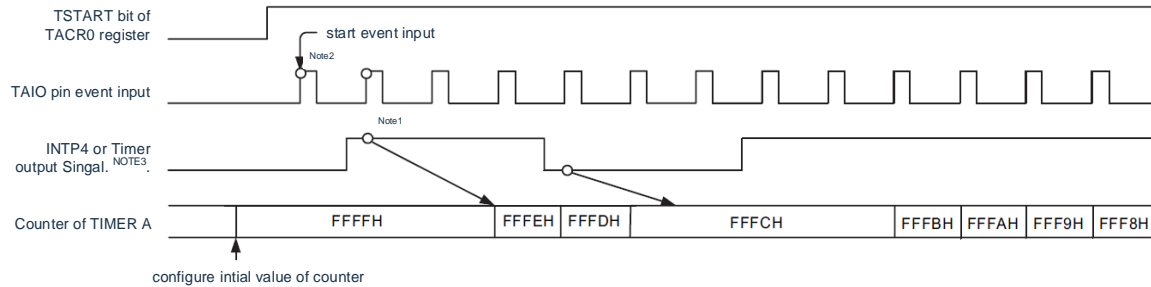
Figure 7-12 Example of running the event counter pattern 1



An example of a specified time count in event counter mode (TIOGT1 bits for the TAIOC0 register and "01B" or "10B" for TIOGT0 bits for the TAIOC0 registers) is shown in the figure 7-13.

Figure 7-13 Example of operation of the event counter pattern 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 singal 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.

7.4.5 Pulse width measurement mode

This is the mode for measuring the width of the external signal pulse at the input of the TAIO pin.

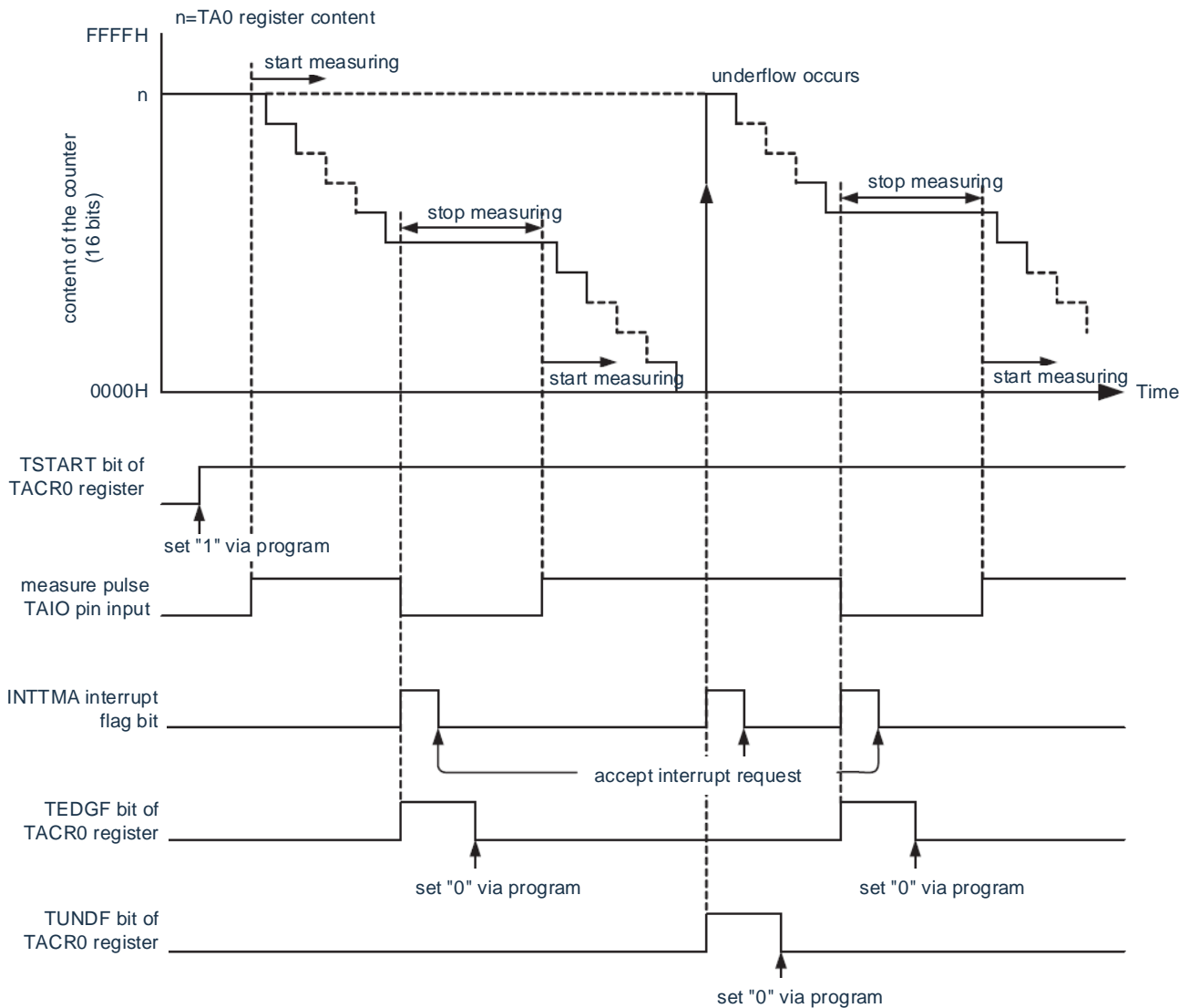
In pulse width measurement mode, if the level specified by the TEDGSEL bit of the TAIOC0 register is input to the TAIO pin, the count is decremented by the selected counting source. If the specified level of the TAIO pin input ends, the counter stops counting, the TEDGF bit of the TACR0 register becomes "1" (with a valid edge) and an interrupt request is generated. 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 7-14.

To access the TEDGF bits and TUNDF bits of the TACR0 register, refer to the "7.5.2 Flag Access" (TEDGF of the TACR0 register bits and TUNDF bits)".

Fig. 7-14 Example of operation of 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).



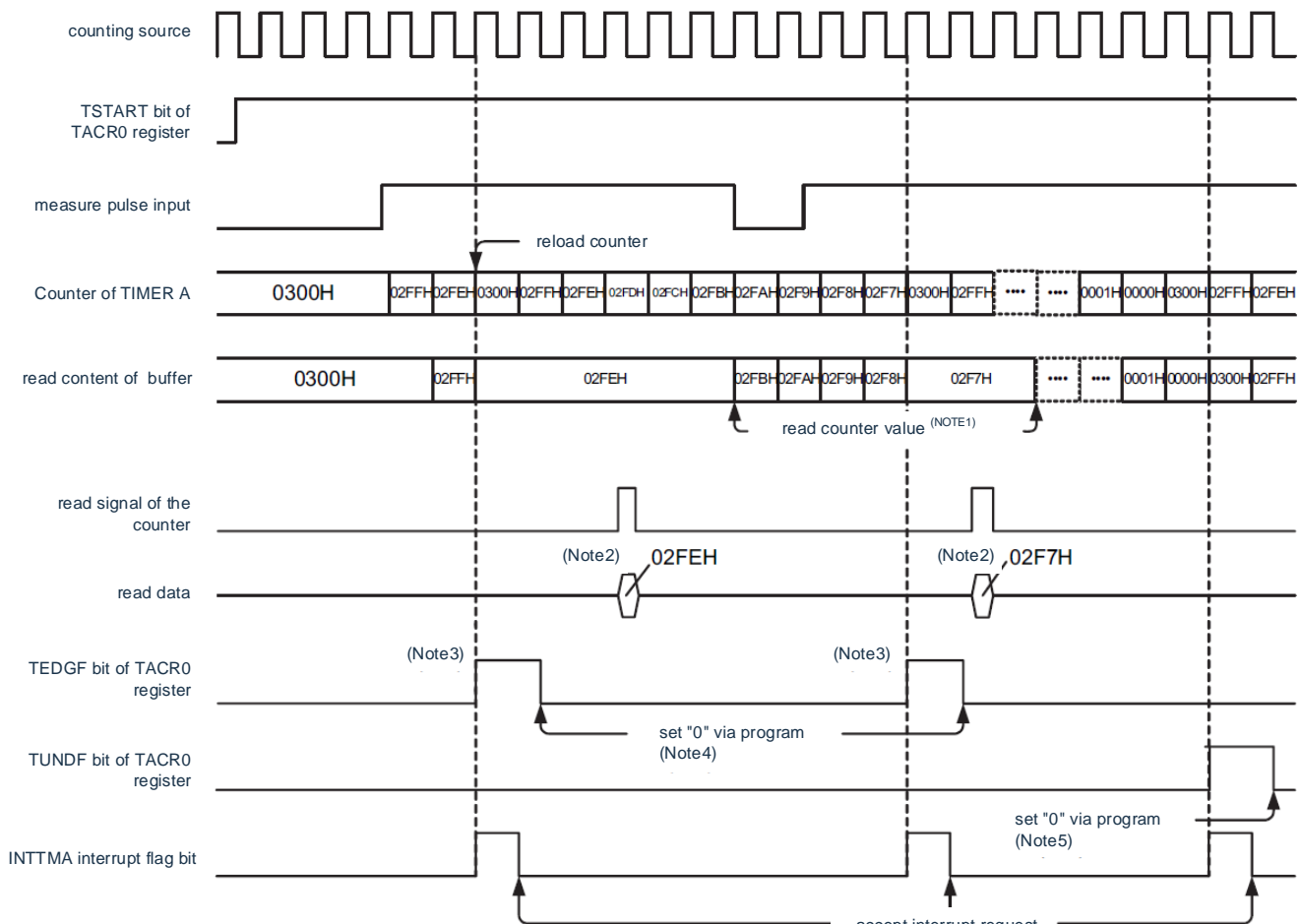
7.4.6 Pulse period measurement mode

This is the mode for measuring the pulse period of the external signal at the input of the TAI0 pin.

The counter counts down by selecting the count source from TCK0 to TCK2 bits of the TAMR0 register. If the TAI0 pin is given a pulse for the specified period of the TEDGSEL bit of the TAI0C0 register, the count value is transmitted to the read buffer on the rising edge of the counting source, and the value of the reload register is loaded to the counter on the next rising edge, while TACR0 The TEDGF bit of the register becomes "1" (with a valid edge) and the interrupt request is generated. At this point, the TA0 register (read buffer) is read, and the difference between the read value and the reload value is the periodic data of the input pulse. The period data is held until the buffer is read. If the counter underflows, the TUNDF bit of the TACR0 register becomes "1" (underflow occurs) and an interrupt request is generated. An example of the operation of the pulse-period measurement mode is shown in Fig. 7-15.

A pulse must be input that is 2 times the period of the counting source, and both the input "L" level and the width of the "H" level must be greater than the pulse of the counting source period. If the input pulse period and width do not meet these conditions, the input pulse may be overlooked.

Fig. 7-15 An example of the operation of the pulse period measurement mode



This is the scenario done while TA0 register initial value as "0300H" and TEDGSEL bit of TAI0C0 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.

7.4.7 Collaboration with EVENTC

By working with EVENTC, events entered by EVENTC can be set as the count source.

Through the TCK0~TCK2 bits of the TAMR0 register, the count is made on the rising edge of the event at the ELC input. However, the EVENTC input does not work in event counter mode.

The EVENTC setup steps are as follows.

- Steps to get started
 - (1) Sets event output destination selection registers (ELSELRn) for EVENTC.
 - (2) Sets the operating mode of the source where the event occurred.
 - (3) Sets the mode of timer A.
 - (4) Start counting of timer A.
 - (5) Start the run of the event occurrence source.
- Stop the running step
 - (1) Stops the running of the event occurrence source.
 - (2) Stop the count of timer A.
 - (3) Set the EVENTC's Event Output Destination Selection Register (ELSELRn) to "0".

7.4.8 Output settings for each mode

The status of the TAO pins and TAIO pins in each mode is shown in Table 7-6 and Table 7-7.

Table 7-6 TAO pin settings

Operating mode	TAIOC0 register		The output of the TAO pin
	TOENA bit	TEDGSEL bit	
All mode	1	1	Inverting output
		0	Normal phase output
	0	0 or 1	Disable output

Table 7-7 TAIO pin settings

Operating mode	TAIOC0 register		Input/output of the TAIO pin
	PMXX bit <small>note</small>	TEDGSEL bit	
Timer mode	0 or 1	0 or 1	Input (not used)
Pulse output mode	1	0 or 1	Disable output (Hi-Z output).
		1	Normal phase output
	0	0	Inverting output
Event counter pattern	1	0 or 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.

7.5 Considerations when using Timer A

7.5.1 Start and stop control of counting

- Event counting mode or when the counting source is set to a non-EVENTC

If you write "1" (start counting) to the TSTART bit of the TACR0 register during the counting stop, the TCSTF of the TACR0 register is counted within 3 source cycles. The bit is "0" (stop count). With the exception of the TCSTF bit, the associated register ^{note} for timer A 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 TSTF bit is "1" for 3 counting source cycles. Stop counting when the TCSTF bit changes to "0". With the exception of the TCSTF bit, the associated register ^{note} for timer A 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 25 Interrupt Functions".

Note: Timer A's relevant registers: TA0, TACR0, TAIOC0, TAMR0, TAISR0

- Event counting mode or when the counting source is set to EVENTC

If you write "1" (start counting) to the TSTART bit of the TARR0 register during the counting stop, there are 2 CPU clock cycles for the TARR0 register. The TCSTF bit is "0" (stop count). With the exception of the TCSTF bit, the associated register ^{note} for timer A 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" for 2 CPU clock cycles. Stop counting when the TCSTF bit changes to "0". With the exception of the TCSTF bit, the associated register ^{note} for timer A 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 25 Interrupt Function".

Note: Timer A's relevant registers: TA0, TACR0, TAIOC0, TAMR0, TAISR0

7.5.2 Access to flags (TEDGF bits and TUNDF bits of the TACR0 register).

If you programmatically write "0" to the TEDGF bits and TUNDF bits of the TARR0 register, these bits become "0". However, even if you write the "1" value, it does not change. If you use a read-modify-write instruction on the TARR0 register, even if the TEDGF bit becomes "1" (with a valid edge) and during instruction execution the TUNDF bit becomes "1" (underflow occurs) or the TEDGF bit and TUNDF position "0" may be mistaken for timing. The TACR0 register must be accessed via the 8-bit memory manipulation instruction.

7.5.3 Access to the Counting register

When writing the TA0 registers continuously with both the TSTART bit and the TSTF bit of the TAC0 register being "1" (counting), there must be at least 3 intervals between the respective write operations counts the source clock cycles.

7.5.4 Change in Operational mode

The operating mode correlation register (TAIOC0) of timer A can only be changed if the stop count (both the TSTART bit and the TSTF bit of the TACR0 register are "0" (stop count)). , TAMR0, TAISR0), cannot be changed during the counting process.

When changing the operating mode correlation registers of timer A, the values of the TEDGF and TUNDF bits are indefinite. The count must begin after writing "0" to the TEDGF bit (no valid edge) and "0" to the TUNDF bit (no underflow occurred).

7.5.5 Setup steps for TAO pins and TAIO pins

After reset, the multiplexed I/O ports of the TAO pin and the TAIO pin are input ports. When you want to output from the TAO pin and the TAIO pin, you must follow the steps below to set it.

Change the step

- (1) Set the mode.
- (2) Sets the initial value to allow output.
- (3) Place the TAO pin and the TAIO pin at the position "0" of the port register.
- (4) Set the bit of the port mode register corresponding to the TAO pin and the TAIO pin to output mode.
(Starting with the TAO pin and taio pin) output
- (5) Start counting (TSTART=1 for TACR0 registers).

When entering from the TAIO pin, you must follow the steps below to set it up.

- (1) Set the mode.
- (2) Set the initial value and select Edge.
- (3) Set the bit of the port mode register corresponding to the TAIO pin to input mode.
(input starting from the TAIO pin).
- (4) Start counting (TSTART=1 for TAMR0 registers).
- (5) Wait until the TCSTF bit of the TACR0 register becomes "1" (counting).
(Event counter mode only)
- (6) Input external events 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)

7.5.6 When timer A is not used

When timer A is not used, the TAMR0 register must be placed at the TMOD2~TMOD0 position "000B" (timer mode) and will

TOENA position "0" of the TAIOC0 register (TAO output is disabled).

7.5.7 Timer A runs the stop of the clock

The provision or stop of the timer A clock can be controlled via the TMAEN bit of the PER1 register. However, the following SFR cannot be accessed when timer A's clock stops, but must be accessed in the state that provides timer A clock.

TA0 registers, TACR0 registers, TAMR0 registers, TAIOC0 registers, and TAISR0 registers

7.5.8 Setup steps for deep sleep mode (event counter mode).

To make event counter mode run in deep sleep mode, you must follow the steps below to move to deep sleep mode after providing the clock for Timer A

Setup steps

- (1) Set the operating mode.
- (2) Start count (TSTART=1, TCSTF=1). (3) Stop providing the clock for timer A.

To stop event counter mode in deep sleep mode, you must follow these steps to run stop processing.

- (1) Provides a clock for timer A.
- (2) Stop count (TSTART=0, TCSTF=0).

7.5.9 Functional limitations in deep sleep mode (event counter mode only).

To make event counter mode run in deep sleep mode, you cannot use the digital filter function.

7.5.10 Forced count stop via the TSTOP bit

The following SFR cannot be taken in memory after 1 count cycle of the counter's count after the TSTOP bit of the TSTOP bit of the TAC0 register is forced to be stopped. TA0 registers, TACR0 registers, and TAMR0 registers

7.5.11 Digital filters

When using a digital filter, the timer operation cannot be started within 5 digital filter clock cycles after the TIPF1 bit and the TIPP0 bit of the TAI0C register are set.

In addition, even if the TEDGSEL bit of the TAI0C register is changed in the state of using a digital filter, the timer operation cannot be started within five digital filter clock cycles.

7.5.12 The case where f_{IL} is selected as the count source

To select f_{IL} as the counting source, the subsystem clock must be supplied with the mode control register (OSMC) at WUTMMCK0 position "1". However, when selecting f_{SUB} as the real-time clock or the counting source for the 15-bit interval timer, f_{IL} cannot be selected as the timer A The count source.

Chapter 8 Timer B

8.1 Function of timer B

Timer B has the following 3 modes:

- Timer Mode:
 - The input capturing function counts on the bilateral edges of the rising, falling, or rising/falling edges.
 - Output comparison function "L" level output, "H" level output or alternating output
- PWM mode: PWM output with arbitrary duty cycle.
- Phase count mode: Automatically measures the count value of a 2-phase encoder.

8.2 Structure of timer B

The block diagram and pin structure of timer B are shown in Figure 8-1 and Table 8-1, respectively.

Figure 8-1 Block diagram of Timer B

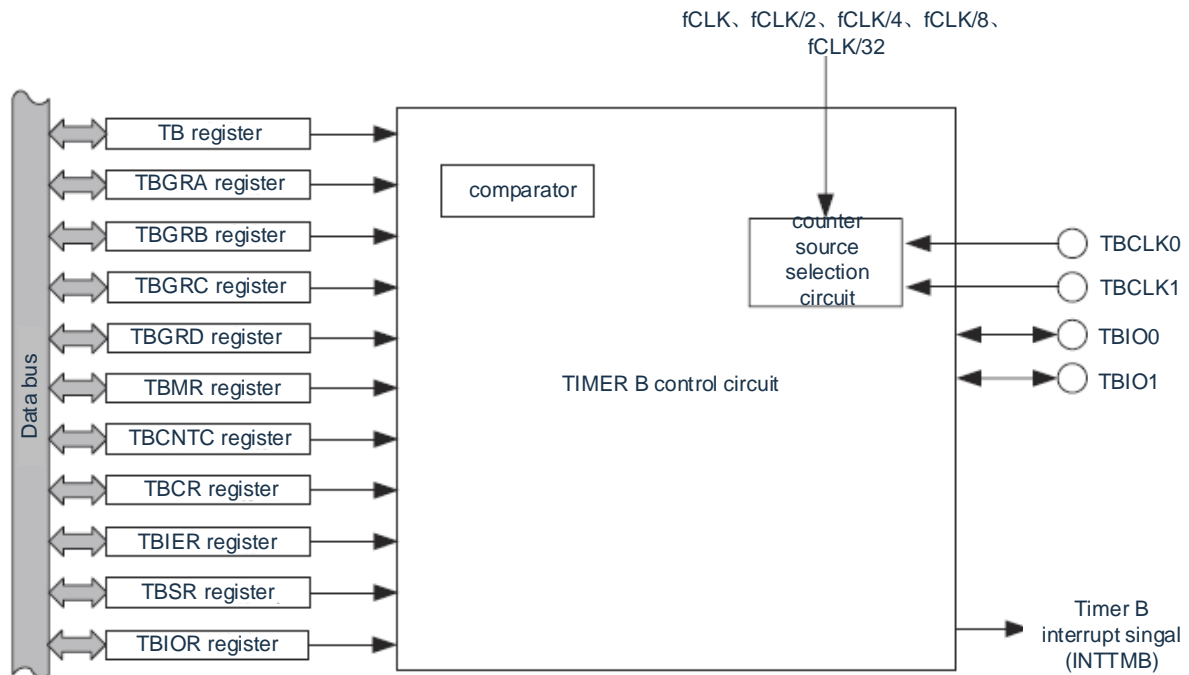


Table 8-1: Timer B Pin Structure

Pin name	The port name of the multiplex	Input/Output	function
TBCLK0	P00	input	<ul style="list-style-type: none"> Phase count mode A phase input Input of external clock 0 in non-phase counting mode
TBCLK1	P01	input	<ul style="list-style-type: none"> Phase count mode B-phase input Input of external clock 1 in non-phase counting mode
TBIO0	P50	Input/Output	<ul style="list-style-type: none"> Timer mode (output comparison function). The output of the TBGRA output comparison Timer mode (input capture function). TBGRA input captures the input PWM mode PWM output
TBIO1	P51	Input/Output	<ul style="list-style-type: none"> Timer mode (output comparison function). The output of the TBGRB output comparison Timer mode (input capture function). TBGRB input captures the input

8.3 Control registers of timer B

The registers that control timer B are shown in Table 8-2.

Table 8-2 control registers of timer B

Register name	symbol
Peripheral enable register 1	PER1
Timer B mode register	TBMR
Timer B counts control registers	TBCNTC
Timer B controls the register	TBCR
Timer B interrupt enables registers	TBIER
Timer B status register	TBSR
Timer BI/O control registers	TBIOR
Timer B counter	TB
Timer B universal register A	TBGRA
Timer B universal register B	TBGRB
Timer B universal register C	TBGRC
Timer B universal register D	TBGRD
Port registers	Pxx
Port mode registers	PMxx

8.3.1 Peripheral enables register 1 (PER1).

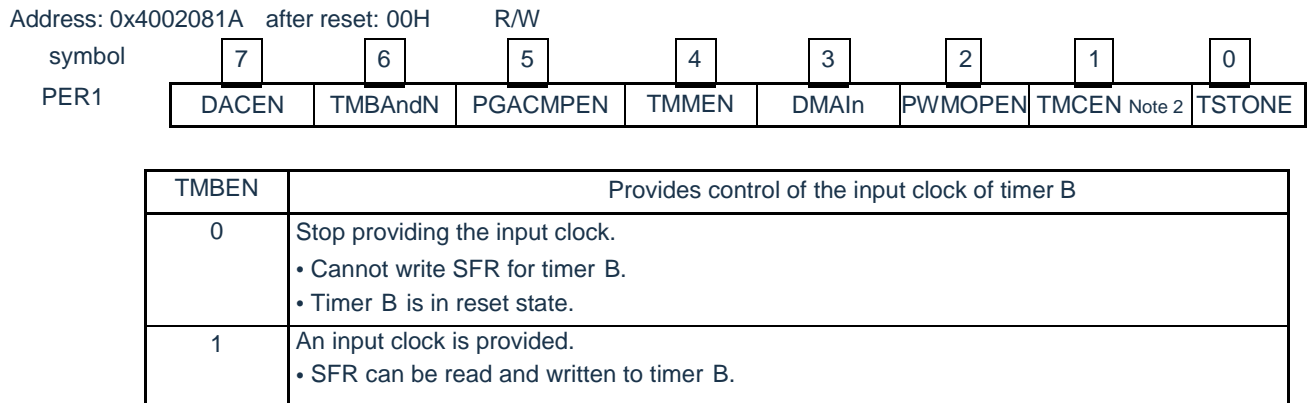
The PER1 register is a register that is set to enable or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use.

To use timer B, bit6 (TMBEN) must be set to "1".

The PER1 register is set via the 8-bit memory operation instruction.

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

Figure 8-2 format of Peripheral enable register 1 (PER1)



Note: To set timer B, you must first place the TMBEN position "1". When the TMBEN bit is "0", the write operation of the control register of timer B is ignored, and the read values are both the initial values (port mode register (PMxx) and port register (P xx) except).

8.3.2 Timer B mode register (TBMR).

Figure 8-3 Format of the timer B mode register (TBMR).

Address: 40042650H After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TBMR	TBSTART	TBELCICE	TBDFCK1	TBDFCK0	TBDFB	TBDFA	TBMDF	TBPINM

TBSTART	The beginning of the TB count
0	Stop counting, and initialize the PWM output signal (TBIO0 pin) (PWM mode).
1	Start counting.

TBELCICE	EVENTC enters the selection notes 1 and 2 for the capture request
0	Select an external input signal 1/digital filter through signal 1.
1	Select event for EVENTC input (Input Capture).

TBDFCK1	TBDFCK0	Select note 1 for the clock used in the digital filter function
0	0	$f_{CLK}/32$
0	1	$f_{CLK}/8$
1	0	f_{CLK}
1	1	TBCR register of TBTCCK0~TBTCCK2 bit select clock

TBDFB	TBIO1 pin digital filter function selection
0	There is no digital filter function.
1	There is a digital filter function.

When the digital filter function is available, up to 5 sampling clock cycles of the digital filter are required for

TBDFA	TBIO0 pin digital filter function selection
0	There is no digital filter function.
1	There is a digital filter function.

When the digital filter function is available, up to 5 sampling clock cycles of the digital filter are required for

TBMDF	Selection of phase count mode
0	Increment the count
1	Phase count mode

When the TBMDF bit is "0", the counter counts the counting source set by the TBTCCK0~TBTCCK2 bit of the TBCR register; When the TBMDF bit is "1", the counter pair is shown in "Table 8-15TB Registers Plus and

TBPWM	Choice of PWM mode
0	Timer mode
1	PWM mode

Note 1 This bit cannot be set when the TBSTART bit is "0" (stop count).

- For event (input capture) of the EVENTC input to be valid, the TBIO12 position of the TBIOR register must be "1", and the TBIO11 bit must be combined TBIO10 position "00B" (rising edge).

8.3.3 Timer B counts the control register (TBCNTC).

Use the TBCNTC register in phase count mode to set the counting conditions for phase count mode.

Figure 8-4 Format of timer B count control register (TBCNTC).

Address: 40042651H after reset: 00H R/W

symbol	<div>7</div>	<div>6</div>	<div>5</div>	<div>4</div>	<div>3</div>	<div>2</div>	<div>1</div>	<div>0</div>
TBCNTC	CNTEN7	CNTEN6	CNTEN5	CNTEN4	CNTEN3	CNTEN2	CNTEN1	CNTEN0
	CNTEN7	Enable for the count 7						
	0	void						
	1	Increment the count When the TBCLK0 input is the "L" level and the rising edge of the TBCLK1 input						
	CNTEN6	Enable for the count 6						
	0	void						
	1	Increment the count When the TBCLK1 input is the "H" level and the rising edge of the TBCLK0 input						
	CNTEN5	Enable for the count 5						
	0	void						
	1	Increment the count When the TBCLK0 input is at the "H" level and the falling edge of the TBCLK1 input						
	CNTEN4	Enable for the count 4						
	0	void						
	1	Increment the count When the TBCLK1 input is the "L" level and the falling edge of the TBCLK0 input						
	CNTEN3	Enable for the count 3						
	0	void						
	1	Decrement the count When the TBCLK1 input is at the "H" level and the falling edge of the TBCLK0 input						
	CNTEN2	Enable for the count 2						
	0	void						
	1	Decrement the count When the TBCLK0 input is the "L" level and the falling edge of the TBCLK1 input						
	CNTEN1	Enable for the count 1						
	0	void						
	1	Decrement the count When the TBCLK1 input is at the "L" level and the rising edge of the TBCLK0 input						
	CNTEN0	Enable for the count 0						
	0	void						
	1	Decrement the count When the TBCLK0 input is at the "H" level and the rising edge of the TBCLK1 input						

8.3.4 Timer B Control Register (TBCR).

The TBCR register must be written in a state where the TBSTART bit of the TBMR register is "0" (stop count).

Figure 8-5 Format of timer B control register (TBCR).

Address: 40042652H after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TBCR	0	TBCCLR1	TBCCLR0	TBCKEG1	TBCKEG0	TBTCK2	TBTCK1	TBTCK0

TBCCLR1	TBCCLR0	Clear source selection for TB registers
0	0	Not allowed to clear.
0	1	Clears when TBGRA's input captures or compares matches.
1	0	Clears when the input of the TBGRB is captured or compared to match.
Other than the above		Prohibit settings.

TBCKEG1	TBCKEG0	Select for the valid edges of the external clock <small>notes 1 and 2</small>
0	0	Count on the rising edge.
0	1	Count on the falling edge.
1	0	Count on the bilateral edges of the rising/falling edges.
Other than the above		Prohibit settings.

TBTCK2	TBTCK1	TBTCK0	Select for the count source <small>Note 1</small>
0	0	0	f_{CLK}
0	0	1	$f_{CLK}/2$
0	1	0	$f_{CLK}/4$
0	1	1	$f_{CLK}/8$
1	0	0	$f_{CLK}/32$
1	0	1	Input to TBCLK0
1	1	1	Input to TBCLK1
Other than the above			Prohibit settings.

Note 1 In the phase count mode, the settings of TBTCK0~TBTCK2 bit, TBCKEG0 bit and TBCKEG 1 bit are invalid, and the phase count mode is preferred.

2. When TBCKEG0 bits and TBCKEG1 bits are set to an external clock (TBCLK0, TBCLK1) in TBTCK0~TBTCK2 bits valid, otherwise it is not valid.

8.3.5 Timer B interrupt enable register (TBIER).

Figure 8-6 Timer B interrupt enable the format of the register (TBIER).

Address: 40042653H after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TBIER	0	0	0	0	TBOVIE	TBUDIE	TBIMIEB	TBIMIEA

TBOVIE	Overflow interrupt Enable
0	Interrupts due to TBOVF bits are prohibited.
1	Interrupts due to TBOVF bits are valid.

TBUDIE	Underflow interrupts are Enable
0	Interrupts due to TBUDF bits are prohibited.
1	Interrupts due to TBUDF bits are valid.

TBIMIEB	Input capture/compare matching interrupts allow B
0	Interrupts due to TBIMFB bits are prohibited.
1	Interrupts due to TBIMFB bits are valid.

TBIMIEA	Input capture/compare matching interrupts enable A
0	Interrupts due to TBIMFA bits are prohibited.
1	Interrupts due to TBIMFA bits are valid.

Note TBIMFA, TBIMFB, TBUDF, TBOVF: Bits of TBSR registers

8.3.6 Timer B status register (TBSR).

Figure 8-7 Format of the timer B status register (TBSR).

Address: 40042654H after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TBSR	0	0	0	TBDIRF	TBOVF	TBUDF	TBIMFB	TBIMFA

TBDIRF	Count direction flags
0	The TB register counts down.
1	The TB registers increment the count.

TBOVF	Overflow flag ^{note 1}
[condition for "0"]. After reading write "0" ^{Note 2} . [condition for "1"]. Refer to "Conditions for each flag of Table 8-3 as "1"".	

TBUDF	Underflow flag
[condition for "0"]. After reading write "0" ^{Note 2} . [condition for "1"]. Refer to "Conditions for each flag of Table 8-3 as "1"".	

TBIMFB	Enter Capture/Compare Match Flag B
[condition for "0"]. After reading, write "0" ^{Notes 2 and 3} . [condition for "1"]. Refer to "Conditions for each flag of Table 8-3 as "1"".	

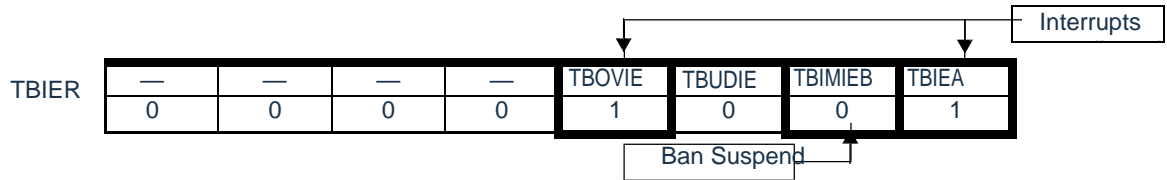
TBIMFA	Enter capture/compare match flag A
[condition for "0"]. After reading, write "0" ^{Notes 2 and 3} . [condition for "1"]. Refer to "Conditions for each flag of Table 8-3 as "1"".	

Note 1. When the count value of timer B changes from "FFFFH" to "0000H", the TBOVF bit changes to "1". In addition, according to the setting of the TBCCLR bits of the TBCR register and the TBCCLR 1 bits, if the count value of timer B is changed from "FFFFH" becomes "0000H" and the TBOVF bit becomes "1".

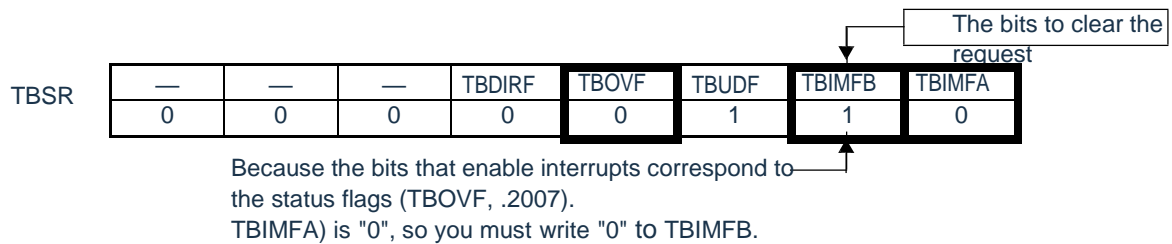
2. Write the result as follows:

- When writing "1", this bit does not change.
- In the case of reading a value of "0", it does not change even if "0" is written to the same bit (in the case of changing from "0" to "1" after reading, it remains "1" even if "0" is written status).
- In the case where the read value is "1", if you write "0" to the same bit, this bit becomes "0".
However, when you want to set the state flag (hereinafter referred to as the "object status flag") of one of the interrupt sources of timer B to "0" if the interrupt is timered
If the B interrupt enable register (TBIER) is set to disable interrupts, it must be placed "0" in any of the following methods (a)~(c).

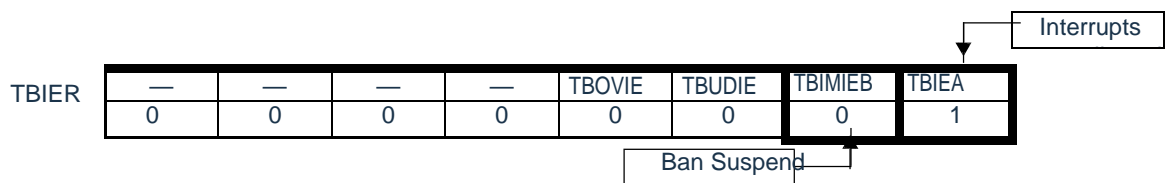
- (a) The object status flag must be written "0" after setting the timer B interrupt enable register (TBIER) to "00H" (disable all interrupts).
- (b) When the timer B interrupt enable register (TBIER) has a bit of "1" (Enable) set and the bit Enable interrupt source status is flagged as "0"
- , you must write "0" to the object status flag.
- (e.g.) clearing TBIMFB in a state where TBIMIEA and TBOVIE enable interrupts and TBIMIEB prohibits interrupts
- Timer B interrupt enables the state of the register (TBIER).



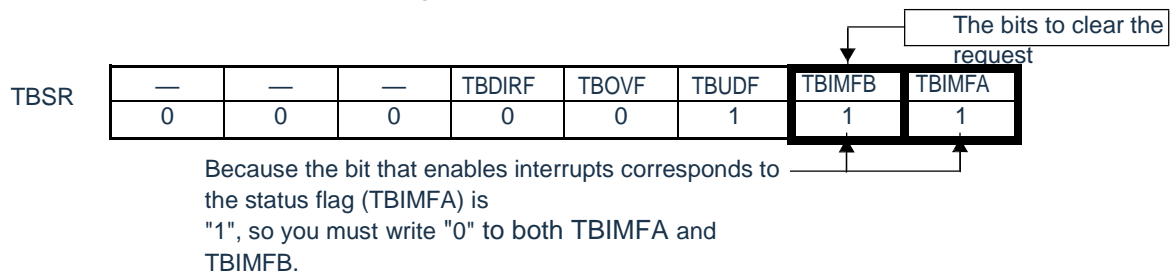
- The status of the timer B status register (TBSR).



- (c) When timer B interrupt enables a bit in the interrupt enable register (TBIER) that is set "1" (Enable) and the interrupt source status flag for that bit is marked as "1"
- , you must write "0" to both this status flag and the object status flag.
- (e.g.) when TBIMIEA clears TBIMFB in a state where interrupts are Enable and TBIMIEB is prohibited
- Timer B interrupt enables the state of the register (TBIER).



- The status of the timer B status register (TBSR).



- When using DMA, the TBIMFA bits and TBIMFB bits become "1" after the DMA transfer ends.

Table 8-3 condition for flag to be marked "1"

Bit symbol	Timer mode ^{Note 1}		PWM mode
	Enter the capture function	Output comparison function	
TBOVF	When tb overflows occur		
TBUDF	When a TB underflow occurs (limited to phase count mode only).		
TBIMFB	Note 2 to the input edge of the TBIO1 pin	When the values of TB and TBGRB are the same	
TBIMFA	The input edge of the TBIO0 pin is ^{note 2}	When the values of TB and TBGRA are the same	

Note: 1 Phase count mode is a counting method of the timer B count register, which can be set to use the above timer mode and PWM mode.

2. This is the edge selected by the TBIOj0 bit and TBIOj1 bit (j=0, 1) of the TBIOR register.

8.3.7 Timer BI/O Control Register (TBIOR).

Figure 8-8 Format of the timer BI/O Control Register (TBIOR).

Address: 40042655H After reset: 00H R/W

	7	6	5	4	3	2	1	0
TBIOR	TBBUFB	TBIOB2	TBIOB1	TBIOB0	TBBUFA	TBIOA2	TBIOA1	TBIOA0

TBBUFB	Selection of TBGRD register functions
0	A buffer register that is not used as a TBGRB register.
1	A buffer register that serves as a TBGRB register.

TBIOB2	TBGRB mode selection notes 1 and 2
0	Output comparison function
1	Enter the capture function

TBIOB1	TBIOB0	TBGRB control
0	0	Disables comparison of matching pin outputs.
0	1	Output "L" level.
1	0	Outputs the "H" level.
1	1	Alternate outputs.
The output of the TB register and tbGRB registers is compared and matched by the output comparison function.		

TBIOB1	TBIOB0	TBGRB control
0	0	Rising edge of TBIO1
0	1	The descending edge of TBIO1
1	0	Bilateral edge of TBIO1
Other than the above		Prohibit settings.
Capture the contents of tb registers to TBGRBs via the input capture function.		

TBBUFA	Selection of TBGRC register functions
0	A buffer register that is not used as a TBGRA register.
1	A buffer register that is used as a TBGRA register.

TBIOA2	TBGRA mode selection Notes 1 and 2
0	Output comparison function
1	Enter the capture function

TBIOA1	TBIOA0	TBGRA control
0	0	Disables comparison of matching pin outputs.
0	1	Output "L" level.
1	0	Outputs the "H" level.
1	1	Alternate outputs.
The output of the TB register and TBGRA registers is compared and matched by the output comparison function.		

TBIOA1	TBIOA0	TBGRA control
0	0	The rising edge of TBIO0
0	1	The descending edge of TBIO0
1	0	Bilateral edge of TBIO0
Other than the above		Prohibit settings.
Capture the contents of the TB register to TBGRA via the input capture function.		

Note: 1 When TBIOj2 bits (j=A, B) are "1" (input capture function), the TBGRj register is used as the input capture register.

2. When the TBIOj2 bit (j=A, B) is "0" (output comparison function), the TBGRj register is used as a comparison match register. Set the TBIOj0 bits and TBIOj1 bits after reset, and output the following levels from the TBIOj pin before the first comparison match occurs:
 - When TBIOj1, TBIOj0=01B, output "H" level.
 - When TBIOj1, TBIOj0=10B, the output "L" level.
 - When TBIOj1, TBIOj0=11B, output "L" level.

This TBIOR register controls the input/output pins in timer mode. Not valid in PWM mode. The TBIOR register must be set in the state of stop count (TBSTART=0 for the TBMR register).

8.3.8 Timer B counter (TB).

Tb registers are connected via a 16-bit internal bus and CPU, so they must be accessed in 16-bit units. TB registers can be incremented, free-running, cycle counting, or external event counting. Tb can be captured by matching with the corresponding TBGRA registers, TBGRB registers, or inputs to TBGRA registers and TBGRB registers Register clear "0000H" (counter clear function).

When the TB register overflows ("FFFFH" "0000H"), the TBOVF bit of the TBSR register becomes "1"; When the TB register underflows ("0000H" "FFFFH"), the TBUDF bit of the TBSR register becomes "1".

Figure 8-9 Format of the timer B counter (TB).

Address: 40042656H		After reset: 0000H		R/W													
Symbol		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TB																	
—		function													Set the scope		
bit15~0		Increment/decrement count in phase count mode and increment count in other modes.													0000H~FFFFH		

8.3.9 Timer B universal registers A, B, C, D (TBGRA, TBGRB, TBGRC, TBGRD)

The TBGRA registers and TBGRB registers are 16-bit read-write registers with the function of output comparison registers and input capture registers. Function conversion via TBIOR registers.

When used as output comparison registers, the values of tbGRA registers and TBGRB registers are always compared to the values of TB registers. If the values are the same (the comparison matches), the TBSR register's TBIMFA bit or TBIMFB bit becomes "1". Matching outputs can be set through TBIOR registers.

When used as an input capture register, the value of the TB register is saved after an external input capture signal is detected. At this point, the TBSR register

The TBIMFA bit or TBIMFB bit becomes "1". The TBIOR register selects the detection edge of the input capture signal.

TbGRC registers and TBGRD registers can also be used as buffer registers for TBGRA registers and TBGRB registers, respectively, and can pass through the TBBUFA bit sum of the TBIBOR registers The TBBUFB bit selects this feature.

For example, if you set the TBGRA register as the output comparison register and the TBGRC register as the buffer register of the TBGRA register, the value of the TBGRC register is passed to each time the comparison match A occurs TBGRA registers.

If the TBGRA register is set as the input capture register and the TBGRC register is set as the buffer register of the TBGRA register, the values of the TB register and the TBGRA register are passed to each other when the input capture occurs TBGRA registers and TBGRC registers.

Can read and write TBGRA, TBGRB, TBGRC, TBGRD registers in 16-bit increments.

Fig. 8-10 Timer B universal registers A, B, C, D (TBGRA, TBGRB, TBGRC, TBGRD) format

Address: 40042658H (TBGRA), 4004265AH (TBGRB), 40042660H (TBGRC), 40042662H (TBGRD) After Reset: FFFFH R/W

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

—	function
bit15~0	Varies by timer mode and function. TbGRA, TBGRB, TBGRC, TBGRD register functions are shown in Table 8-4.

Remark:i=A, B, C, D

Table 8-4TBGRA, TBGRB, TBGRC, TBGRD register functions

Modes and features	register	Set up	function
Input capture	TBGRA	TBIOR (TBIOA2=1) TBMR (TBPWM=0)	Enter the capture register (which holds the value of the TB register).
	TBGRB	TBIOR (TBIOB2=1) TBMR (TBPWM=0)	Enter the capture register (which holds the value of the TB register).
Output comparison	TBGRA	TBIOR (TBIOA2=0) TBMR (TBPWM=0)	Output comparison register (saves and compares the value of the TB register when the comparison matches TBIO0 output settings).
	TBGRB	TBIOR (TBIOB2=0) TBMR (TBPWM=0)	Output comparison register (saves and compares the value of the TB register when the comparison matches TBIO1 output settings).
PWM	TBGRA	TBMR (TBPWM=1)	Output comparison register (TBIO0 output "H" level when comparing matches).
	TBGRB		Output comparison register (TBIO0 output "L" level when comparing matches).
common	TBGRC	TBIOR (TBBUFA=0)	Not used.
	TBGRD	TBIOR (TBBUFB=0)	Not used.
	TBGRC	TBIOR (TBBUFA=1)	TBGRA's buffer registers (and TBGRA are transmitted). • When TBIOA2=1 By entering the capture signal, the previous input capture value is taken from TBGRA. • When TBIOA2=0 Through the comparison matching of TB and TBGRA, the next comparison expectation is transmitted to TBGRA.
	TBGRD	TBIOR (TBBUFB=1)	The BUFFER register of the TBGRB (and the TBGRB are transmitted). • When TBIOB2=1 By entering the capture signal, the previous input capture value is taken from the TBGRB. • When TBIOB2=0 By comparing and matching TB and TBGRB, the next comparison expectation is transmitted to TBGRB.

Note: If you place the TBTCK2~TBTCK0 position of the TBCR register "000B" (f_{CLK}) and set the comparison value to "0000H" , generate 1 request signal to DMA and EVENTC immediately after starting counting. If the comparison value is greater than or equal to "0001H", a request signal is generated at each time the comparison matches.

8.3.10 Port registers and port mode registers

When using the multiplex port of the timer output pin as the output of the timer, the bit of the corresponding port mode register (PMxx) and the position of the port register (Pxx) of each port must be "0".

(Example) The P50/TBIO0 is used as a timer output in the case of the PM50 position "0" of the port mode register 5. Place the P50 position of port register 5 "0".

When using the multiplex port of the timer input pin as the input to the timer, the position of the port mode register (PMxx) corresponding to each port must be "1". At this point, the bit of the port register (Pxx) can be "0" or "1".

(Example) P50/TBIO0 as a timer input

Place the PORT mode register 5 at PM50 position "1".

Place the P50 position of port register 5 "0" or "1".

For details, please refer to "2.3.1 Port Mode Register (PMxx)", "2.3.2 Port Register (Pxx)" and "2.3.6 Port Mode Control Register (PMCxx)".

The set port mode registers (PMxx), port registers (Pxx), and port mode control registers (PMCxx) vary by product. For details, please refer to "Register Settings when Using the Multiplexing Function in 2.5".

8.4 Operation of timer B

8.4.1 Common things about multiple patterns and features

(1) Count the sources

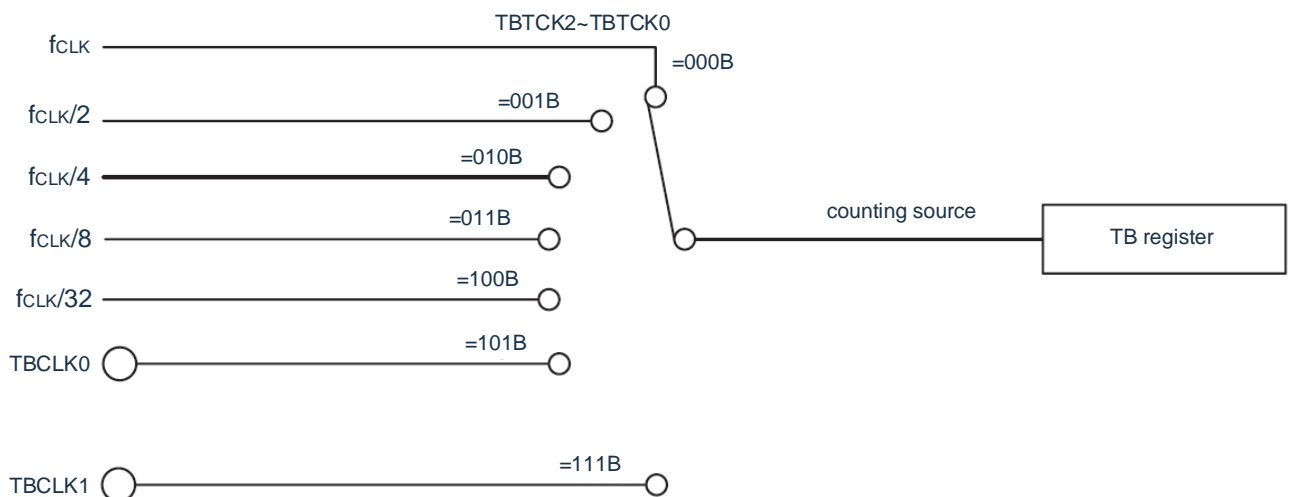
The selection of the counting source and the block diagram are shown in Table 8-5 and Figure 8-12, respectively.

When selecting the phase count mode, the settings of TBTCCK0~TBTCCK2 bits, TBCKEG0 bits, and TBCKEG 1 bits of tbCR registers are invalid.

Table 8-5 Counting Source Selection

Count the sources	Select a method
f_{CLK} , $f_{CLK}/2$, $f_{CLK}/4$, $f_{CLK}/8$, $f_{CLK}/32$	Select the counting source by the TBTCCK0~TBTCCK2 bit of the TBCR register.
TBCLK0 pin and TBCLK1 External input signal for pins	The TBCR register has bits "101B" (TBCLK0 input) or "111B" in the TBCR register (TBCLK1 input). Valid edges are selected by the TBCKEG0 bits and TBCKEG1 bits of the TBBCR registers. The corresponding bit of the port

Figure 8-12 Block diagram of the counting source



Remark: TBTCCK2~TBTCCK0: TBCR register bits

The pulse width of the external clock input to the TBCLKj pin (j=0, 1) must be at least 3 timer B operating clock (f_{CLK}) cycles.

(2) The buffer runs

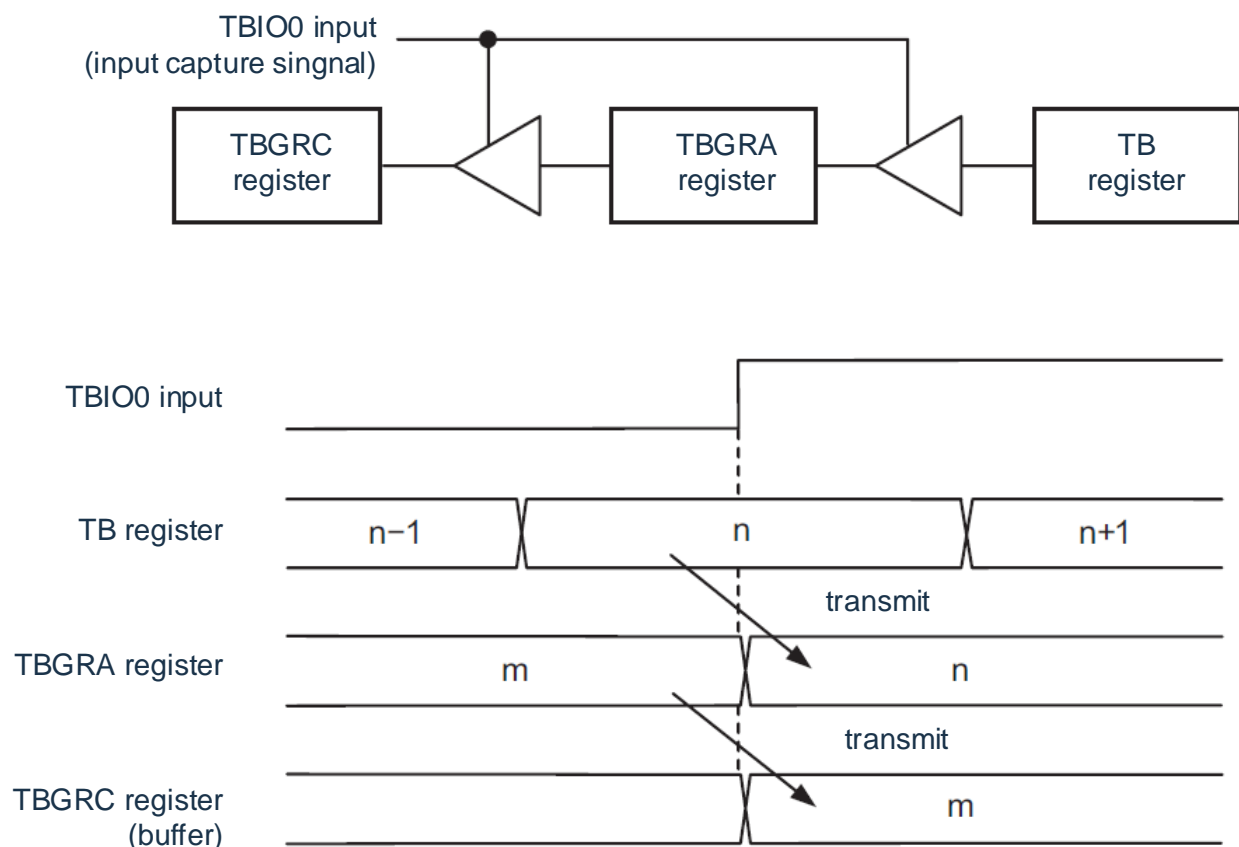
The TBGRC register and TBGRD register can be set to the TBGRA register and the TBGRA register and the TBGRD register respectively through the TBBUFA bit and TBBUFB bit of the TBIBOR register Buffer registers for TBGRB registers.

- TBGRA's buffer register: TBGRC register
- TBGRB's buffer registers: TBGRD register buffer operation varies depending on the timer mode. The buffer operation of each mode is shown in Table 8-6, and the buffer operation of the input capture function and the output comparison function is shown in Figure 8-13 and Figure 8-14, respectively.

Table 8-6 Buffer operation for each mode

Features and modes	Transmission timing	Registers for transfer
Enter the capture function	Enter the input of the capture signal	Transfers the contents of the TBGRA (TBGRB) register to the buffer register.
Output comparison function	TB registers and TBGRA (TBGRB).	Transmits the contents of the buffer register to TBGRA
PWM mode	Comparison matching of registers	(TBGRB) registers.

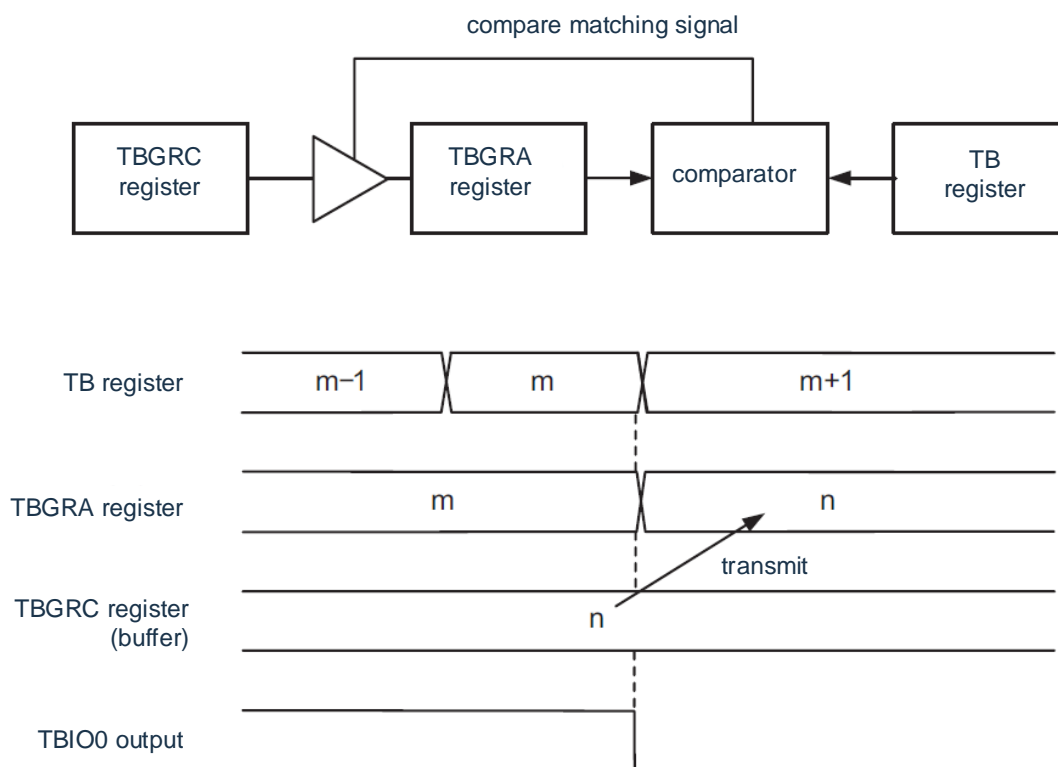
Figure 8-13 The buffer of the input capture function is running



above diagram condition as following:

- TBBUFA bit of TBIBOR register is 1 (TBGRC register is the buffer register of TBGRA)
- TBIOA~TBIOA0 bit of TBIBOR register is "100B" (input capture at rising edge)

Figure 8-14 The buffer of the output comparison function is running



above diagram condition as following:

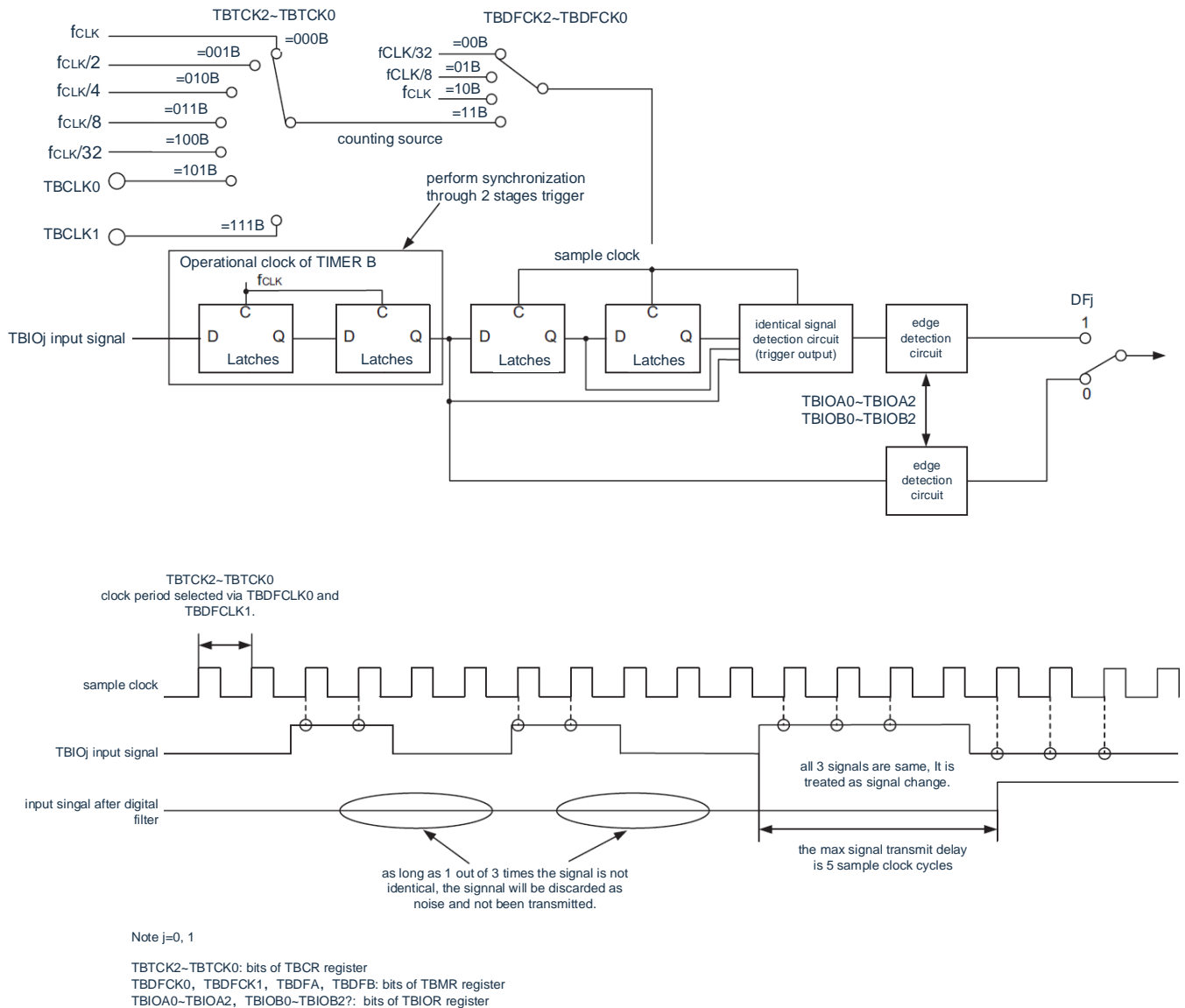
- TBBUFA bit of TBIOR register is 1 (TBGRC register is the buffer register of TBGRA)
- TBIOA2~TBIOA0 bit of TBIOR register as "001B" (while compare matching, output "L" voltage level).

(3) Digital filters

The TBIOj input (j=0, 1) is sampled and if the signals are the same three times, the level is considered determined. The function and sample clock of the digital filter must be selected through the TBMR registers.

The block diagram of the digital filter is shown in Figure 8-15.

Figure 8-15 Block diagram of a digital filter



(4) Enter events from EVENTC

Over event events entered by EVENTC, Timer B performs an input capture run B. At this point, the TBSR register has a TBIMFB bit of "1". To use this function, the input capture function of timer mode/phase count mode must be selected, and the TBELCICE position of the TBMR register must be "1". In other modes (output comparison function in timer mode/phase count mode and PWM mode), this feature is not valid.

Setup steps

- (a) Set the EVENTC event link target to Timer B.
- (b) Place the TBELCICE position of the TBMR register "1".

(5) Events output to EVENTC

The events output to EVENTC via the TBIMFA bit and the TBIMFB bit are shown in Table 8-7 and Table 8-8, respectively.

Tables 8-7 Events output through TBIMFA bits to EVENTC

Features and modes	EVENTC source
Enter the capture function (TBPWM=0, TBIO02=1)	TBIO0 edge detection through TBIOA0 bit and TBIOA1 bit settings
Output comparison function (TBPWM=0, TBIO02=0)	Comparison matching of TB registers and TBGRA registers
PWM mode (TBPWM=1).	Comparison matching of TB registers and TBGRA registers

Note TBPWM: The bit of the TBMR register
TBIOA0, TBIOA1, TBIOA2: Bits of the TBIOR register

Table 8-8 Events output to EVENTC via TBIMFB bits

Features and modes	EVENTC source
Enter the capture function (TBPWM=0, TBIO12=1)	TBIO1 edge detection by TBIO0 bit and TBIO1 bit settings
Output comparison function (TBPWM=0, TBIO12=0)	Comparison matching of TB registers and TBGRB registers
PWM mode (TBPWM=1).	Comparison matching of TB registers and TBGRB registers

Note TBPWM: The bit of the TBMR register
TBIOB0, TBIOB1, TBIOB2: Bits of the TBIOR register

8.4.2 Timer mode (input capture function)

Capable of passing the value of tb registers to TBGRA registers and TBGRBs after detecting the input edge of the input capture/output comparison pins (TBIO0, TBIO1). Register. Detection edges can be selected from rising, falling, and bilateral edges. Pulse width and pulse period can be measured by using the input capture function.

The specifications of the input capture function are shown in Table 8-9.

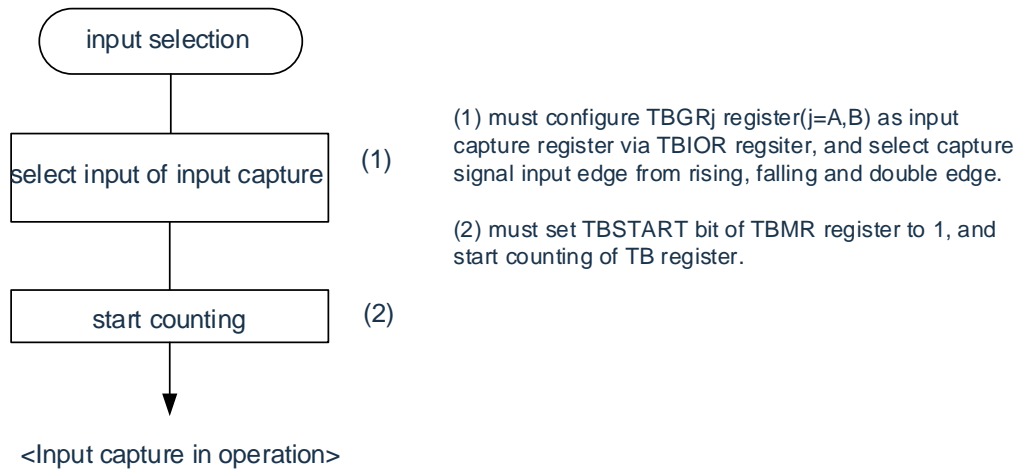
Table 8-9 Specifications for input capture function

project	specification
Count the sources	f_{CLK} , $f_{CLK}/2$, $f_{CLK}/4$, $f_{CLK}/8$, $f_{CLK}/32$ External input signals from the TBCLK0 and TBCLK1 pins (valid edges are selected programmatically).
count	Increment the count
Count cycles	When the TBCLR bits 1 and TBCCLR0 bits of the TBBCR register are "00B" (free running). $1/f_k$ 65536fk: The frequency of the count source
Count start criteria	Write "1" (start counting) to the TBSTART bit of the TBBR register.
Count stop conditions	Write "0" (stop count) to the TBSTART bit of the TBMR register.
Timing of the generation of interrupt requests	<ul style="list-style-type: none"> Input capture (effective edges of TBIO0 pin and TBIO1 pin inputs). Overflow of TB registers
TBIO0 pin and TBIO1 The function of the pin	I/O ports or inputs captured by pins
TBCLK0 pin and Features of the TBCLK1 pin	I/O port or external clock input
Read timer	If you read the TB register, you can read the count value.
Write timer	Can write TB registers.
Select Features	<ul style="list-style-type: none"> Input capture of the selection of input pins 1 or 2 of the TBIO0 pins and TBIO1 pins Input snaps the selection of valid edges of input Rising edge, falling edge, or bilateral edge Set the TB register to the "0000H" timing Spillage or input capture Buffer operation (see "8.4.1(2) Buffer Operation"). Digital filters (see "8.4.1(3) Digital Filters"). Input capture operation via EVENTC's event input signal (input capture).

(1) Enter an example of a setup step for the capture run

An example of the setup steps for entering a capture run is shown in Figure 8-16.

Figure 8-16 Example of setup steps for input capture runs

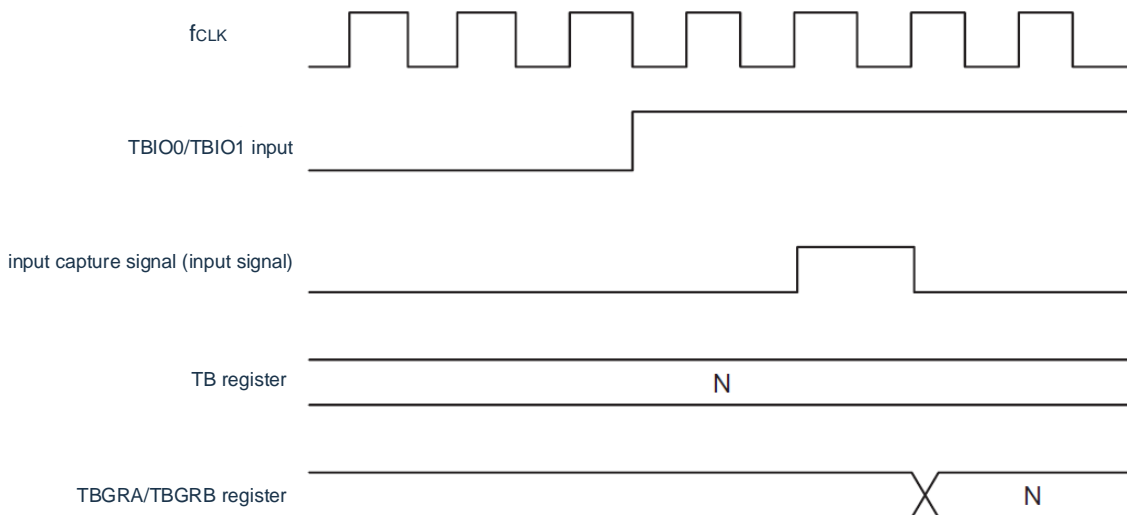


(2) Enter the timing of the capture signal

Input capture inputs can be selected by setting the TBIOR registers on the rising, falling, or bilateral edges. The input signal timing of the input capture is shown in Figure 8-17.

In the case of one-sided edges, the input signal pulse width captured by the input must be at least $1.5 f_{CLK}$; In the case of bilateral edges, the input signal pulse width captured by the input must be at least $2.5 f_{CLKs}$.

Figure 8-17 Input signal timing for input capture



(3) Run the example

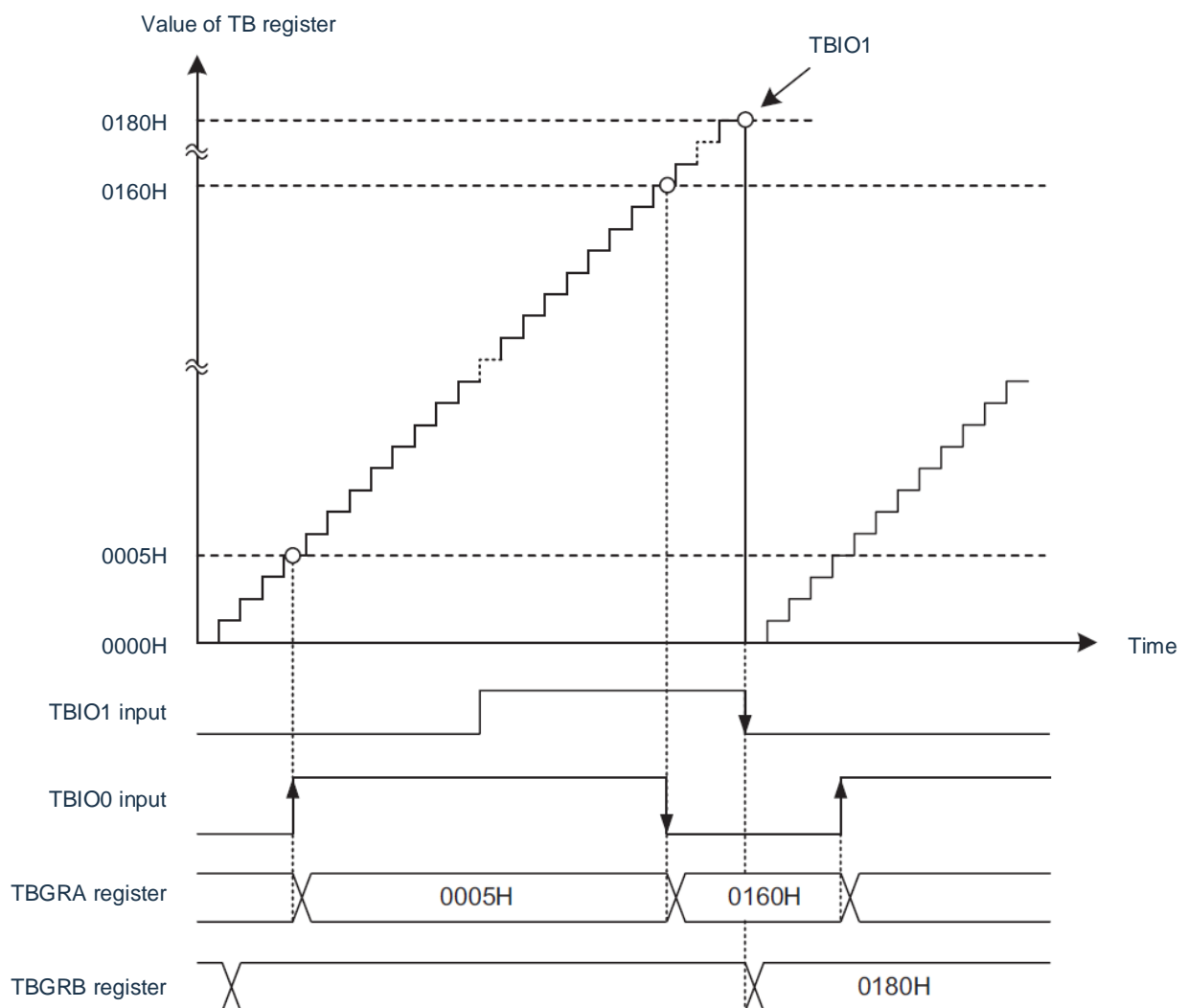
An example of an input capture operation is shown in Figure 8-18.

In this example, select the double edge of the rising/falling edge as the input edge of the input snap of the TBIO0 pin, select the falling edge as the input edge of the input capture of the TBIO1 pin, and clear the TB when the input snap of the TBGRB register Counters for registers.

(a) The TBGRA registers and TBGRB registers must be set as input capture registers through the TBIOR registers, and the input edges of the input capture signal must be selected from the rising, falling, and bilateral edges.

(b) The TBSTART position of the TBBMR register must be placed "1" to start counting the TB registers.

Figure 8-18 an example of an input capture



By setting the TBCLR bits 0 and TBCCLR 1 bits of the TBBCR registers, the count can be cleared when input capture A or input capture B occurs. Fig. 8-18 is an example of a run when the TBCCLR1 bit and the TBCCLR0 position "10B" are used. If set to count clearing by input capture during operation and input capture when the timer's count value is "FFFFH", the interrupt flags for TBIMFA bits and TBIMFB bits, etc., are set to the runtime order of the count source and input capture. The TBOVF bit may also change to "1".

8.4.3 Timer mode (output comparison function)

This is the mode that detects (relatively matches) whether the contents of the TB register and the contents of the TBGRA register or TBGRB register are the same. If the content is the same, output any level from the TBIO0 pin or the TBIO1 pin.

The specifications of the output comparison function are shown in Table 8-10.

Table 8-10 Specifications of the output comparison function

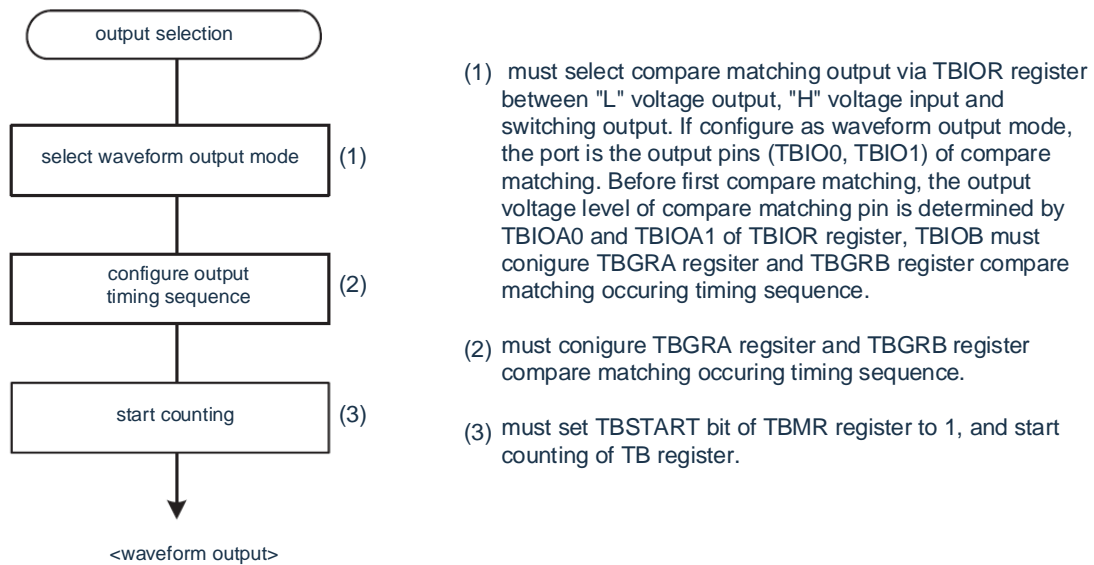
project	specification
Count the sources	f_{CLK} , $f_{CLK}/2$, $f_{CLK}/4$, $f_{CLK}/8$, $f_{CLK}/32$ An external input signal from the TBCLKj pin that programmatically selects a valid edge
count	Increment the count
Count cycles	<ul style="list-style-type: none"> When the TBCLR 1 bit and TBCCLR 0 bits of the TBCR register are "00B" (free running). 1/fk 65536fk: The frequency of the count source When the TBCCR registers the TBCCLR1 bit and the TBCCLR0 bit are "01B" or "10B" (in TBGRj Compare the setting value of the TBGRj register when TB is set to "0000H") when the match is set to 1/fk (n+1)n: TBGRj register
Waveform output timing	Comparison matching (the contents of the TB register and the TBGRj register are the same).
Count start criteria	Write "1" (start counting) to the TBSTART bit of the TBBR register.
Count stop conditions	Write "0" (stop count) to the TBSTART bit of the TBMR register.
Timing of the generation of interrupt requests	<ul style="list-style-type: none"> Comparison matching (the contents of the TB register and the TBGRj register are the same). Overflow of TB registers
TBIO0 pin and TBIO1 The function of the pin	I/O ports or outputs for output comparison (selected by pin).
TBCLK0 pin and Features of the TBCLK1 pin	I/O port or external clock input
Read timer	If you read the TB register, you can read the count value.
Write timer	Can write TB registers.
Select Features	<ul style="list-style-type: none"> Output comparison of the selection of output pins 1 or 2 of the TBIO0 pins and TBIO1 pins Compare the selection of output levels when matching "L" level output, "H" level output, or level inversion output Set the TB register to the "0000H" timing Comparison matching of overflow or TBGRj registers Buffer operation (see "8.4.1(2) Buffer Operation").

Remark: j=A, B

(1) Example of setup steps to compare matching waveform outputs

The setup steps for comparing the matching waveform outputs are shown in Figure 8-19.

Figure 8-19 compares the setup steps for matching waveform outputs

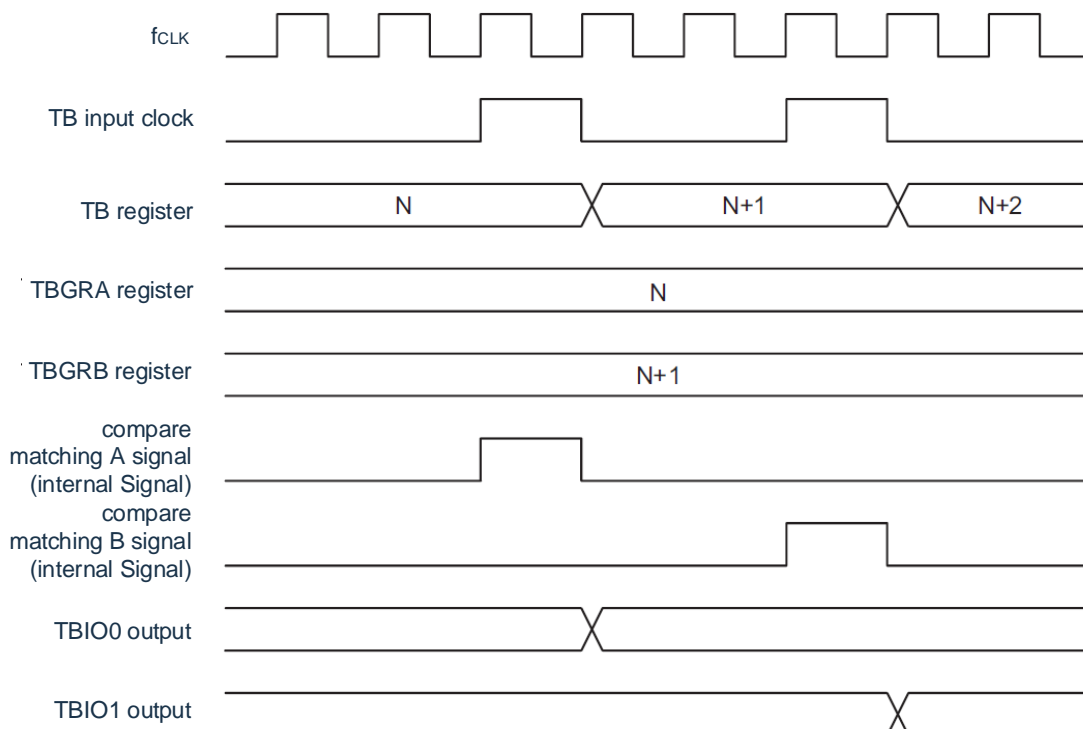


(2) Output timing for output comparison

A comparison match signal is generated when the contents of the TB registers are the same as the contents of the TBGRA registers or the TBGRB registers (when the count value after the same TB registers is updated). After generating a comparison match signal, output the output setting value of the TBIOR register from the output pins of the output comparison (TBIO0, TBIO1). From the contents of the TB registers that are identical to the contents of the TBGRA registers or TBGRB registers to the input clock that generates the TB registers, no comparison match signal is generated.

The output timing for the output comparison is shown in Figure 8-20.

Figure 8-20 Output Compares the Output Timing

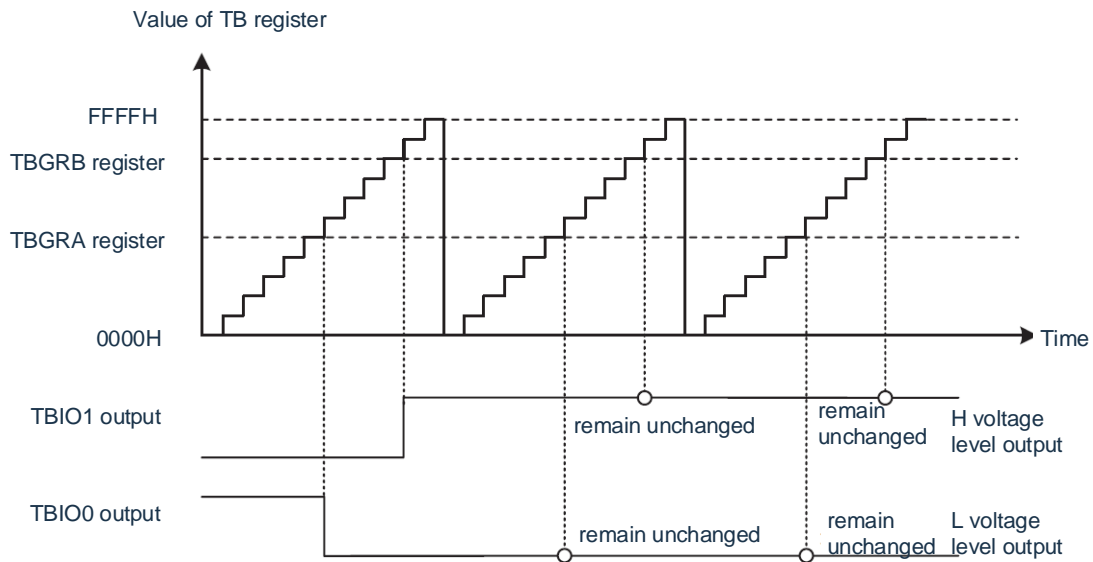


(3) Run the example

An example of the operation of the "L" level output and the "H" level output is shown in Figure 8-21.

In this example, the TB register is set to run freely and outputs the "L" level when comparing match A and the "H" level when comparing match B. If the set level is the same as the level of the pin, the level of the pin is unchanged.

Figure 8-21 "L" level output and "H" level output operation example



An example of alternating outputs is shown in Figure 8-22. In this example, the TB register is set to cycle count operation (the counter is cleared when the comparison matches B) and alternate outputs occur when either match A or match B is compared.

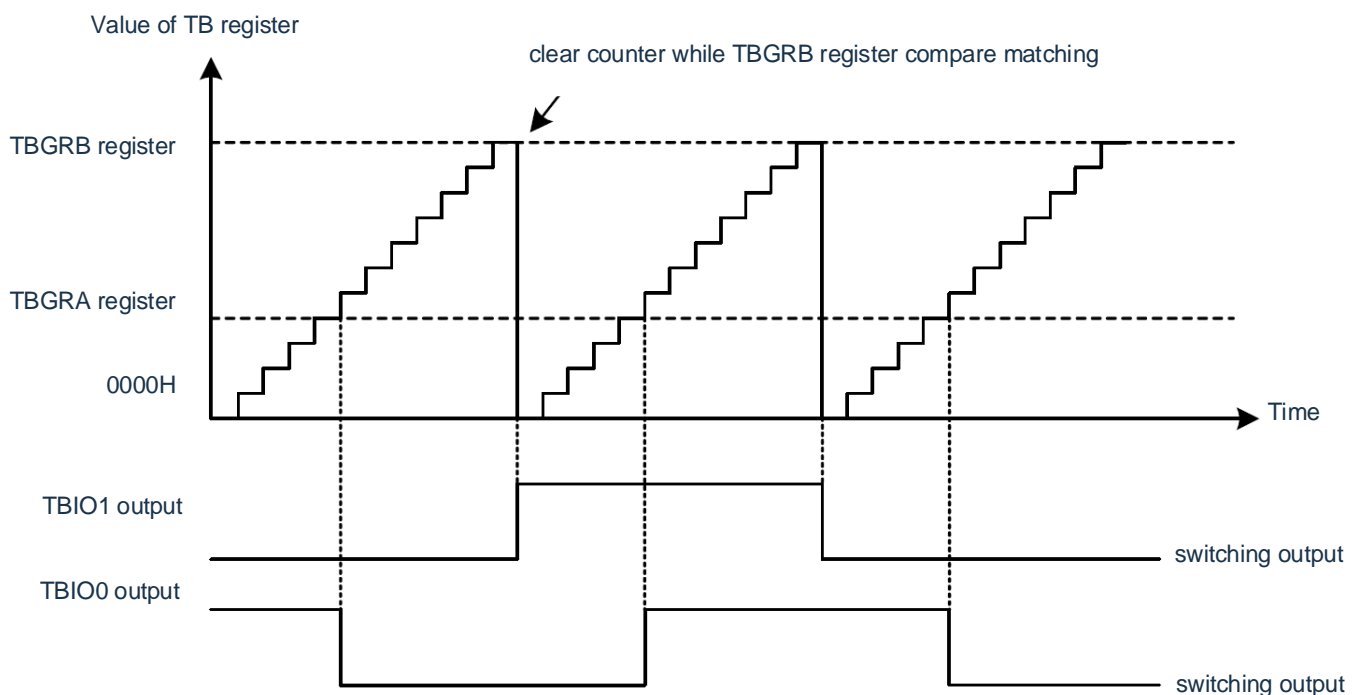
(a) The comparison matching output must be selected from the "L" level output, the "H" level output, and the alternating output through the TBIOR register. If set to waveform output mode, the ports are relatively matched output pins (TBIO0, TBIO1).

(b) The TBGRA register and TBGRB registers must be set to match the occurrence timing.

(c) The TBSTART position of the TBBMR register must be placed "1" to start counting the TB registers.

Even if the TBSTART position is "0" during operation, the output pins that match the comparison (TBIO0, TBIO1) are not initialized. To return to the initial value, the output is initialized by writing the TBIOR register (however, only the TBIO00 bits, TBIO01 bits, TBIO10 through the TBIOR registers Bits and TBIO11 bits are initialized when they are set to the "L" level output or the "H" level output). By setting the TBCCLR registers at TBCCLR bits 0 and TBCCLR 1 bits, the input capture/comparison matches (and TBGRA.) Resets the count value of timer B when the registers or TBGRB registers are the same. At this point, if the comparison expected value is "FFFFH", it changes from "FFFFH" to "0000H" in the same way as the overflow, and the TBOVF bit is "1". The same is true in the mode of output comparison using timer B and comparing expected values.

Figure 8-22 is an example of a run with alternating outputs



8.4.4 PWM mode

PWM mode pairing uses TBGRA registers and TBGRB registers to output the PWM waveform from the TBIO0 output pin. For output pins set to PWM mode, the output setting of the TBIOR register is invalid. Set the "H" level output timing of the PWM waveform to the TBGRA register and the "H" of the PWM waveform to the TBGRB register L" level output timing.

The source can be cleared by setting the comparison match of the TBGRA register or the TBGRB register to the counter of the TB register, and the output is 0 to 100% from the TBIO0 pin PWM waveform for duty cycle.

The specifications of the PWM mode and the combination of PWM output pins and registers are shown in Table 8-11 and Table 8-12, respectively.

When the setting values of the TBGRA registers and the TBGRB registers are the same, the output values remain the same even if a comparison match occurs.

Table 8-11 Specifications for PWM mode

project	specification
Count the sources	fCLK, fCLK/2, fCLK/4, fCLK/8, fCLK/32 TBCLK0, an external input signal from the TBCLK1 pin (valid edges are selected programmatically).
count	Increment the count
PWM waveform	<ul style="list-style-type: none"> Set the "H" level output timing of the PWM waveform to the TBGRA registers. Set the "L" level output timing of the PWM waveform to the TBGRB registers.
Count start criteria	Write "1" (start counting) to the TBSTART bit of the TBBR register.
Count stop conditions	Write "0" (stop count) to the TBSTART bit of the TBMR register.
Timing of the generation of interrupt requests	<ul style="list-style-type: none"> Comparison matching (the contents of the TB register and the TBGRj register are the same). Overflow of TB registers
TBIO0 pin function	PWM output
TBIO1 pin function	I/O port
TBCLK0 pin and Features of the TBCLK1 pin	I/O port or external clock input
Read timer	If you read the TB register, you can read the count value.
Write timer	Can write TB registers.
Select Features	<ul style="list-style-type: none"> Timing overflow of TB registers at "0000H" or comparison matching of TBGRj registers Buffer operation (see "8.4.1(2) Buffer Operation").

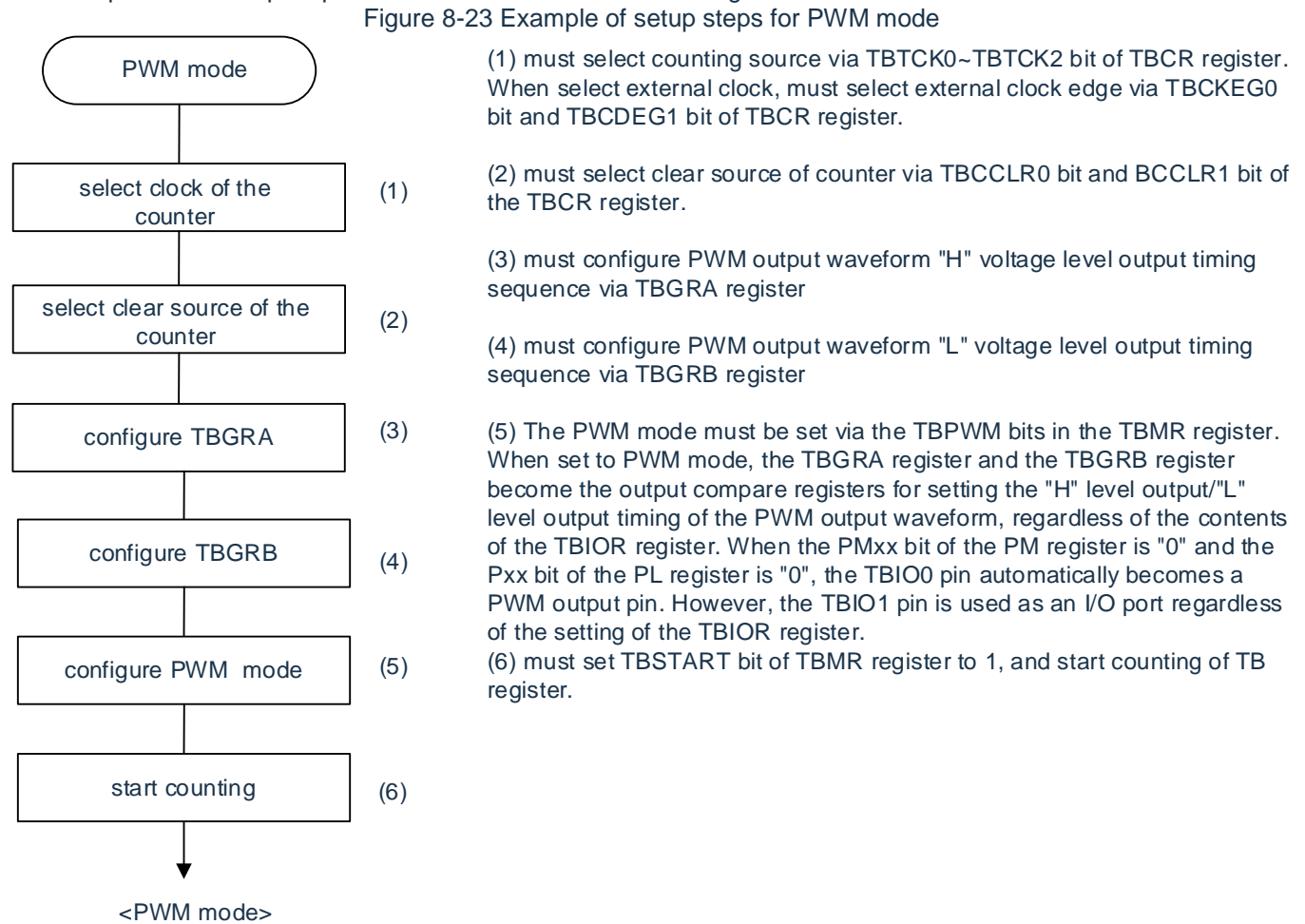
Remark j=A, B

Table 8-12 PWM output pin and register combination

Output pins	"H" level output	"L" level output
TBIO0	TBGRA	TBGRB
TBIO1	Used as an I/O port.	

(1) Example of setup steps for PWM mode

An example of the setup steps for the PWM mode is shown in Figure 8-23.



(2) Run the example

An example of operation in PWM mode (1) is shown in Figure 8-24.

When the PM xx bit of the PM register is "0" and the Pxx bit of the PL register is "0", If set to PWM mode, the TBIO0 pin automatically becomes an output pin and outputs the "H" level at TBGRB when the comparison of TBGRA registers matches The comparison of registers matches the output "L" level. However, the TBIO1 pin is independent of the setting of the TBIOR register and is used as an I/O port.

In this example, the comparison match of the TBGRA register and the TBGRB register is set to the counter clear source of the TB register. The initial state of the TBIO0 pin depends only on the clear source of the counter, as shown in Table 8-13.

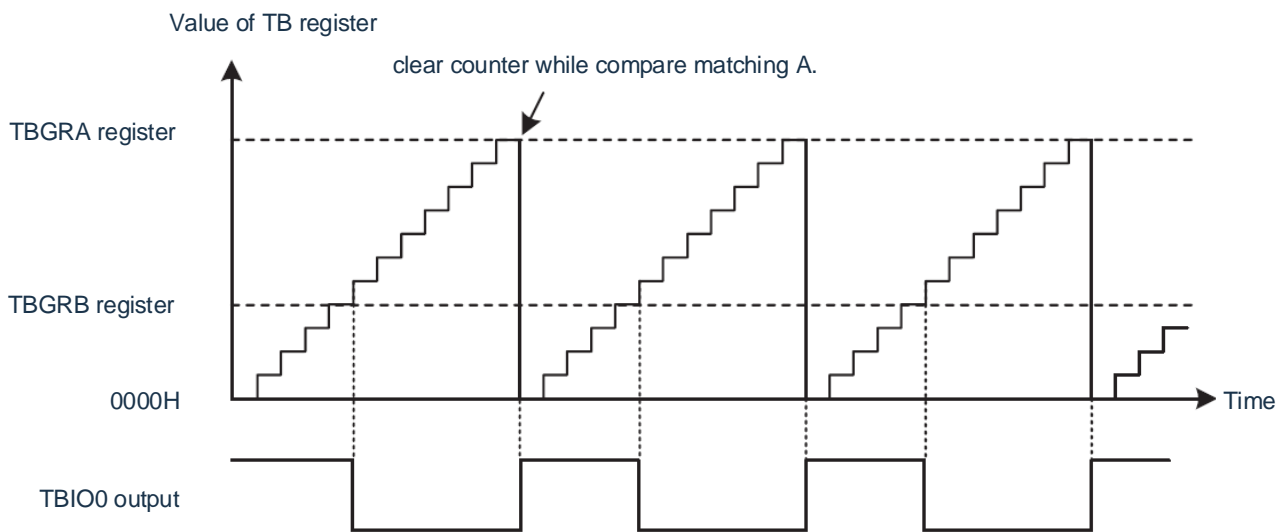
This initialization occurs when the TBSTART bit of the TBBTR register is "0" (stop count).

Table 8-13 The initial state of the TBIO0 pin and the correspondence of the counter cleanup source

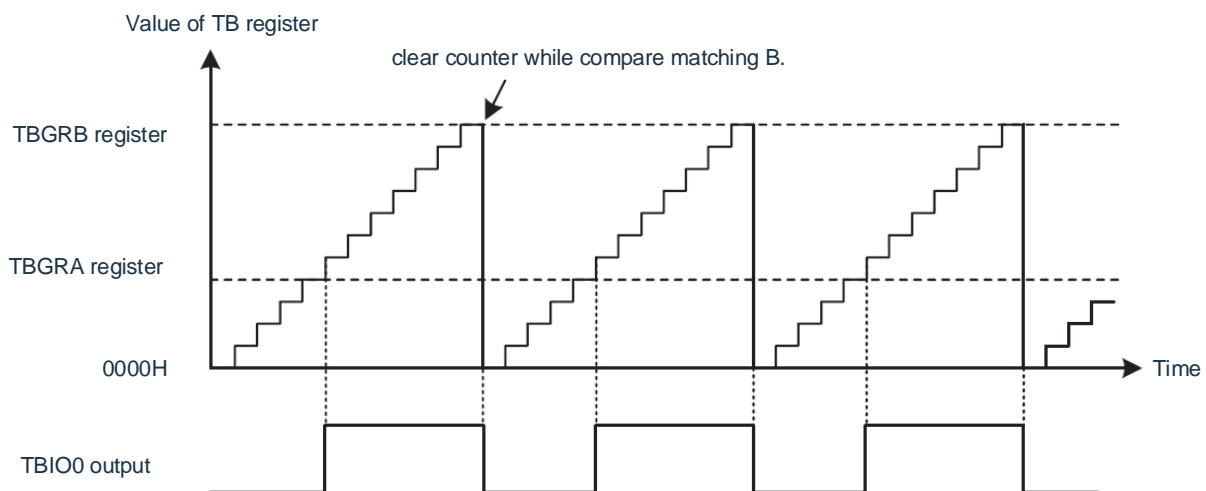
Counter clear source	The initial state of the TBIO0 pin
Comparison matching of TBGRA registers	"H" level
Comparison matching of TBGRB registers	"L" level

When the TBCLR 1 bit and TBCCLR 0 bit of the TBCR register are "00B" (disable clearing), the initial state of the TBIO0 pin is the "H" level.

Figure 8-24 Operation example of PWM mode (1).



(a) clear counter while TBGRA register compare matching.



(b) clear counter while TBGRB register compare matching

An example of a PWM waveform outputting 0% and 100% duty cycle in PWM mode is shown in Figure 8-25. When the comparison match of the TBGRB register is set to the clear source of the counter and the following conditions are met, the duty cycle of the PWM waveform is 0%.

- The setting value of the TBGRA register > the setting value of the TBGRB register

When the comparison match of the TBGRA register is set to the clear source of the counter and the following conditions are met, the duty cycle of the PWM waveform is 100%.

- The setting value of the TBGRB register > the setting value of the TBGRA register

When the following conditions are met, the output value is unchanged even if a comparison match occurs.

- Setting value of TBGRA register = setting value of TBGRB register

[illegible]

8.4.5 Phase count mode

The phase count mode detects the phase difference between the external input signals of the two TBCLK0 pins and the TBCLK1 pin, and the TB registers increment/decrement count.

When the PM xx bit of the PM register is "1", if set to phase count mode, the TBCLK0 pin and the TBCLK1 pin are automatically used as external clock input pins, and TB registers are added and subtracted according to the setting of CNTEN0 to CNTEN7 bits of the TBCNTC registers, and the TB REGISTER is counted. The settings of TBTCK0~TBTCK2 bit, TBCKEG0 bit, and TBCKEG 1 bit are not related. However, because the TBCCR registers TBCCLR bits 0 and TBCCLR 1 bits, TBIOR, TBIER, TBIR, TBIER The TBSR, TBGRA, TBGRB registers are valid, so the input capture/output comparison function, PWM output function, and interrupt source can be used.









Depending on the setting of CNTEN0 to CNTEN7 bits, the TB registers are counted on the bilateral edge of the rising/falling edge of the TBCLK0 pin and the TBCLK1 pin.

The specifications of the phase count mode and the conditions for adding and subtracting TB registers are shown in Table 8-14 and Table 8-15, respectively.

Table 8-14 Specifications of Phase Count Mode

Item	specification
Count the sources	External input signal from the TBCLK0/TBCLK1 pins
count	Increment/decrement count
Count start criteria	Write "1" (start counting) to the TBSTART bit of the TBBR register.
Count stop conditions	Write "0" (stop count) to the TBSTART bit of the TBMR register.
Timing of the generation of interrupt requests	<ul style="list-style-type: none"> • Input capture (effective edges of TBIO0/TBIO1 input). • Comparison matching (TB registers and TBGRA/TBGRB registers have the same content). • Overflow of TB registers • Underflow of TB registers
TBIO0 pin function	I/O port, input capture input, output comparison output, or PWM output
TBIO1 pin function	I/O ports, inputs that are captured, or outputs that are compared
TBCLK0 pin and Features of the TBCLK1 pin	External clock input
Read timer	If you read the TB register, you can read the count value.
Write timer	Can write TB registers.
Select Features	<ul style="list-style-type: none"> • Selection of counters for addition and subtraction conditions Selection is made by CNTEN0~CNTEN7 bits of the TBCNTC register. • Can use input capture/output comparison function and PWM function.

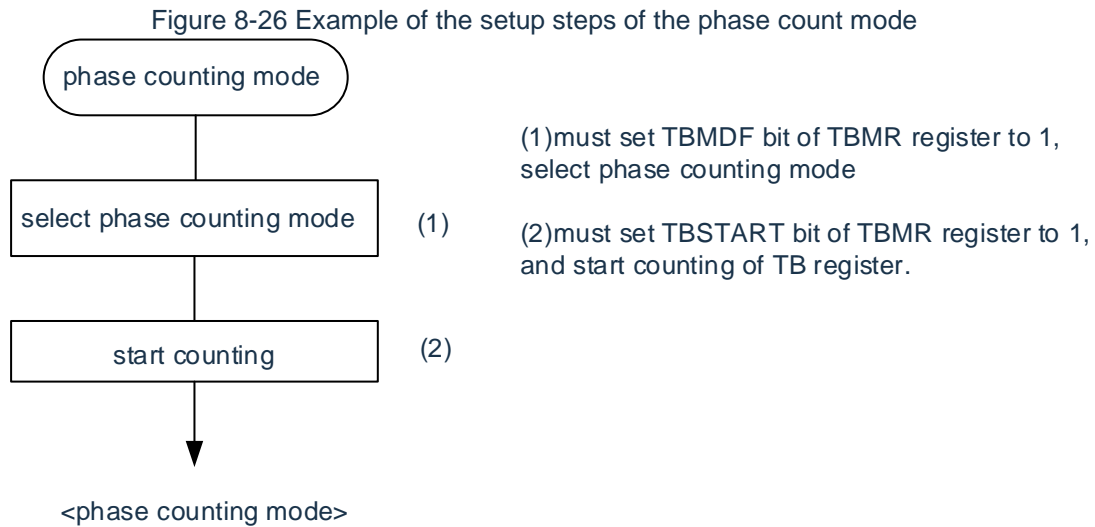
Table 8-15 TB register addition and subtraction conditions

TBCLK1 pin		"H"		"L"	"H"		"L"	
TBCLK0 pin	"L"		"H"			"L"		"H"
TBCNTC registers CNTEN0~CNTEN7 bit	CNTEN7	CNTEN6	CNTEN5	CNTEN4	CNTEN3	CNTEN2	CNTEN1	CNTEN0
Count the directional note	+1	+1	+1	+1	-1	-1	-1	-1

Note indicates the count direction of the TBCNTC register when the number is "1" (incrementing or decreasing count). Is not counted when "0" (invalid).

(1) Example of a step setup step for the phase count mode

An example of the setup steps for the phase count mode is shown in Figure 8-26.



(2) Run the example

An example of the phase count mode is shown in Figures 8-27 to 8-30.

In phase count mode, the TBBCLK0 pin is summed up according to the setting of the CNTEN0~CNTEN7 bits of the TBCNTC register

The TBCLK1 pin is counted on the double side of the rising edge () / falling edge ().

Figure 8-27 Example of the phase count mode 1

· while TBCNTC register value as "FFH"

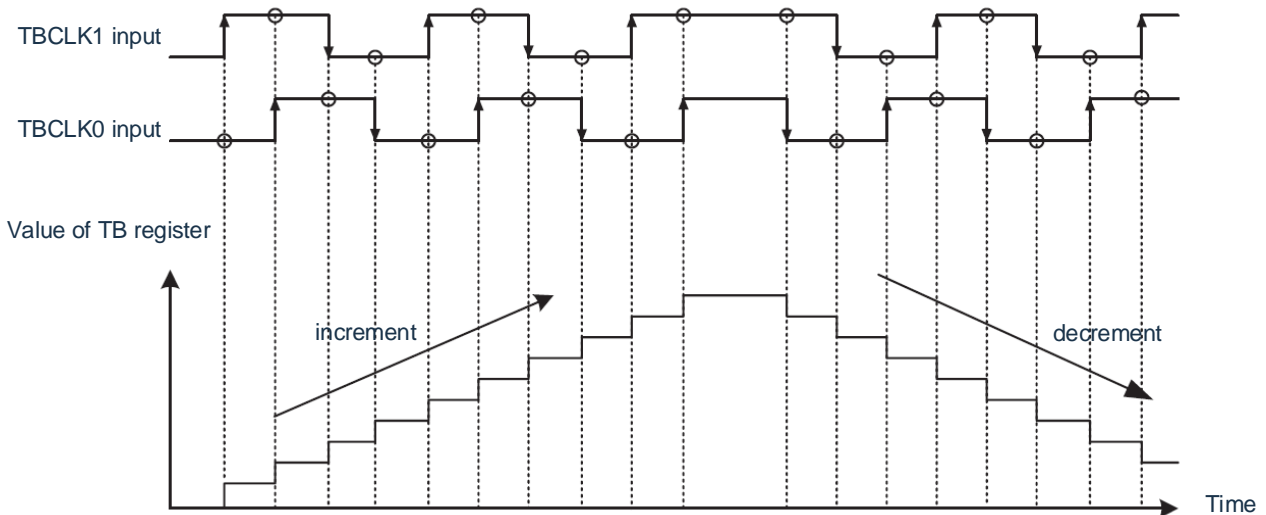


Figure 8-28 Example of the phase count mode 2

• while TBCNTC register value as "24H"

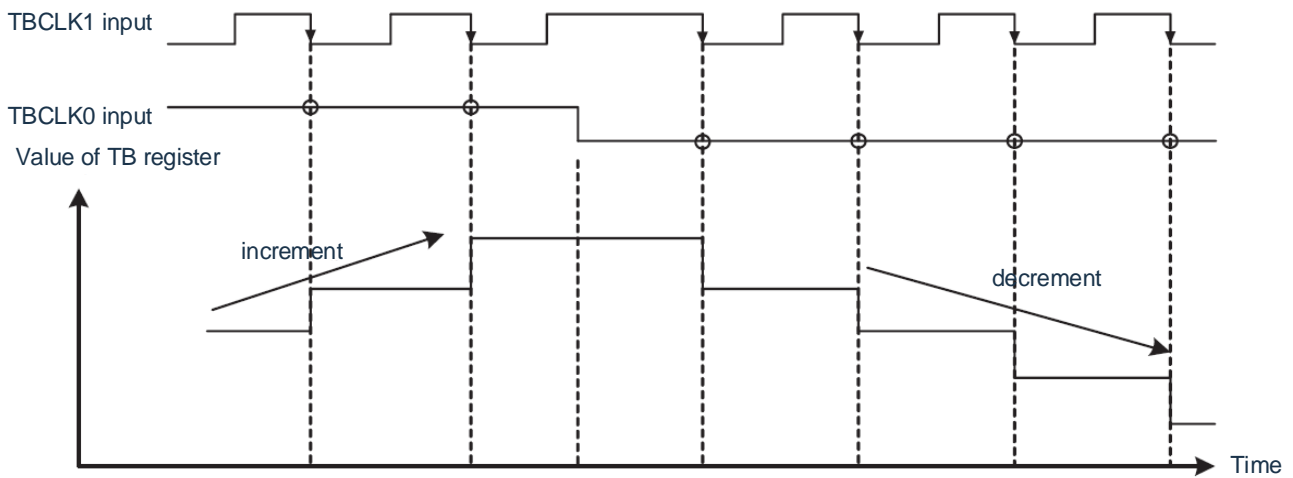


Figure 8-29 Example of the phase count mode 3

• while TBCNTC register value as "28H"

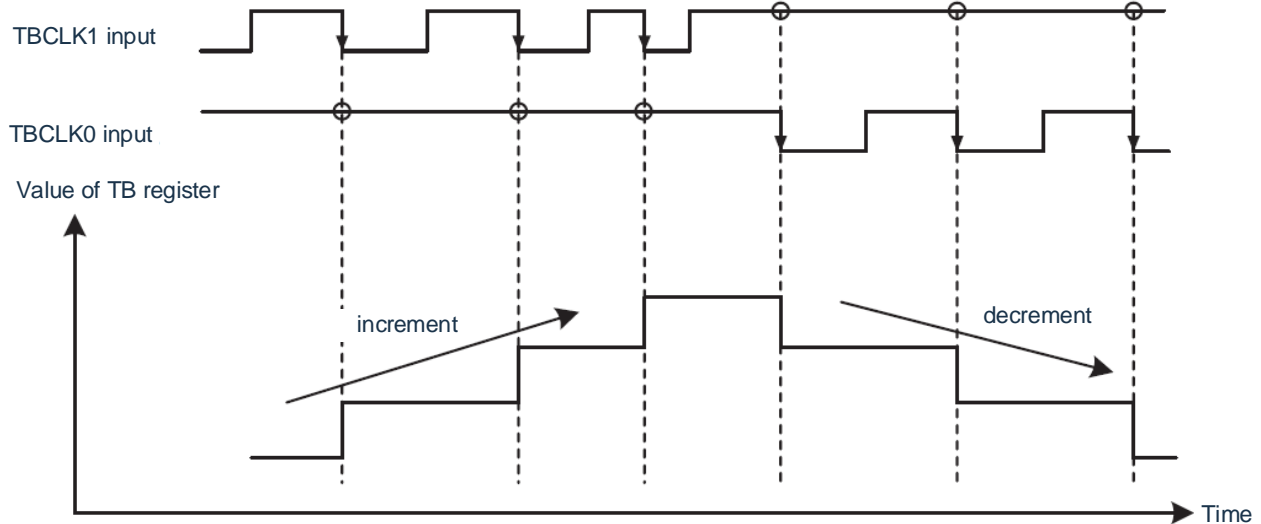
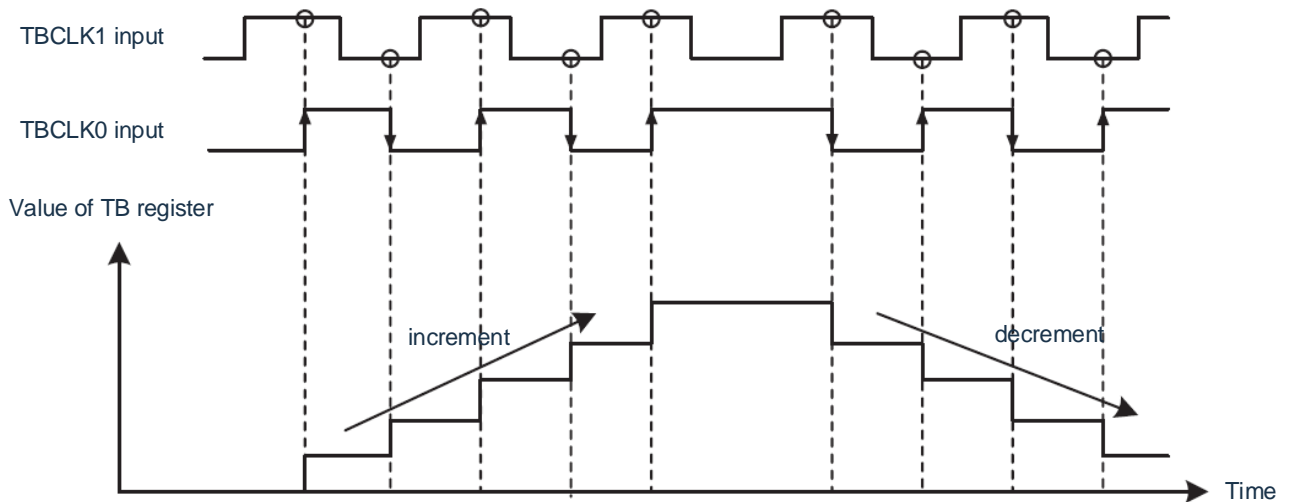


Figure 8-30 Example of the phase count mode 4

• while TBCNTC register value as "5AH"



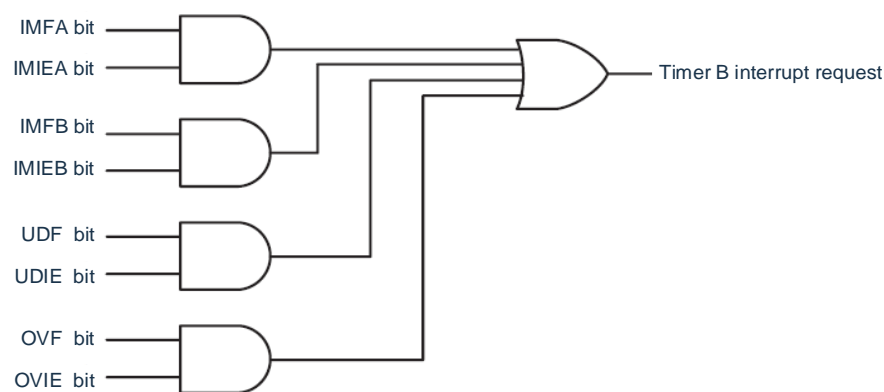
8.5 Timer B interrupt

Timer B generates timer B interrupt requests from 4 interrupt sources. The associated registers for the timer B interrupt are shown in Table 8-16, and the block diagram of the timer B interrupt is shown in Figure 8-31.

Table 8-16 Timer B interrupt related registers

	Status register for timer B	Interrupts of timer B enable registers	Interrupt request flag (Register)	Interrupt mask flag (Register)	The priority specifies the flag (Register)
Timer B	TBSR	TBIER	TBIF (IF2H)	TBMK (MK2H)	TBPR0 (PR02H) TBPR1 (PR12H)

Figure 8-31 Block diagram of timer B interrupt

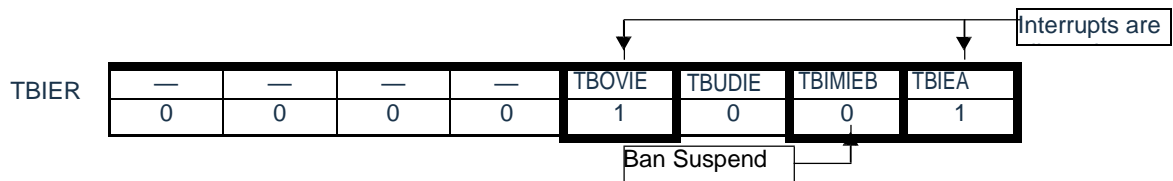


IMFA, IMFB, UDF, OVF: TBSR register Bits
IMIEA, IMIEB, UDIE, OVIE: TBIER register Bits

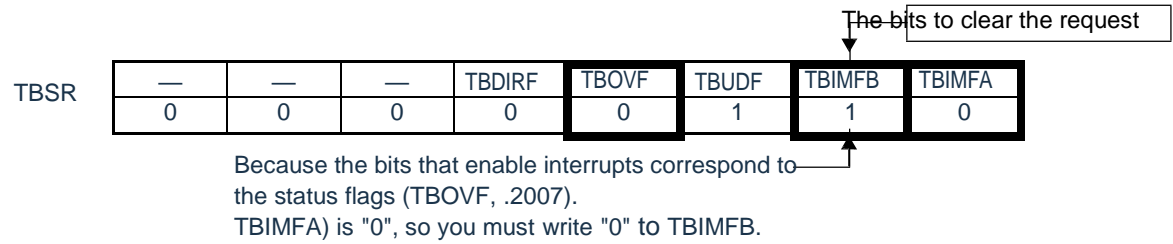
Because timer B generates 1 interrupt request from multiple interrupt request sources (timer B interrupt), in addition to the timer M interrupt, there are the following differences between other maskable interrupts:

- Bit0 of the IF1D register when the bit of the TBSR register is "1" and the bit of the corresponding TBIER register is "1" (interrupt Enable). The bit becomes "1" (with interrupt request).
- When multiple bits of the TBIER register are "1", the TBSR register must be used to determine which request source generated the interrupt.
- Because each of the TBSR registers does not automatically change to "0" even if an interrupt is accepted, these positions must be "0" in the interrupt program.
- When you want to set the status flag (hereinafter referred to as the "object status flag") of one of the interrupt sources of timer B to "0", if the interrupt is interrupted by timer B interrupt enable register (TBIER) to disable interrupts, you must use any of the following methods (a) ~ (c) to set "0".
 - a) The object status flag must be written "0" after setting the timer B interrupt enable register (TBIER) to "00H" (disable all interrupts).
 - b) When the timer B interrupt enable register (TBIER) has a bit of "1" (Enable) set and the bit Enable interrupt source status is flagged as "0", you must write "0" to the object status flag. (e.g.) clearing TBIMFB in a state where TBIMIEA and TBOVIE enable interrupts and TBIMIEB prohibits interrupts

- Timer B interrupt enables the state of the register (TBIER).



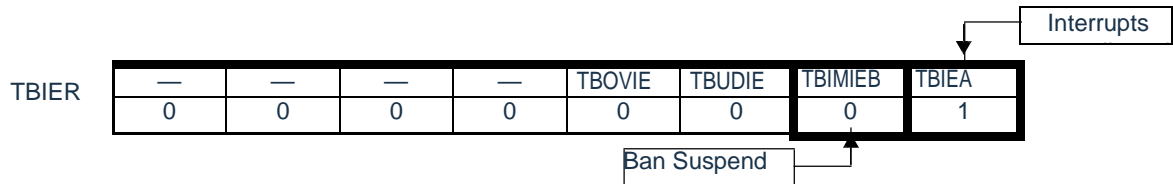
- The status of the timer B status register (TBSR).



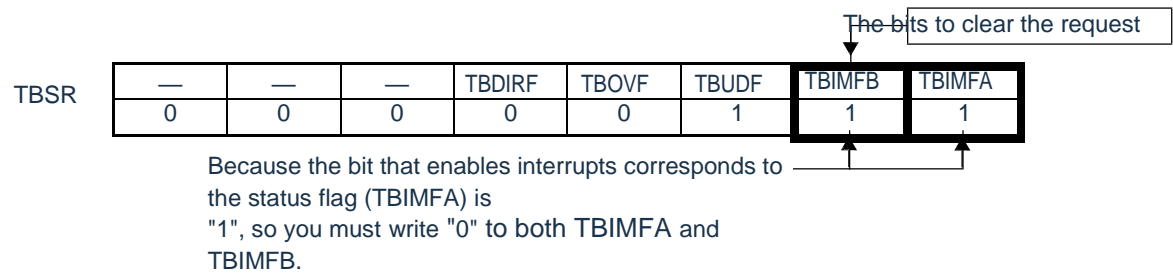
(c) When timer B interrupt enables a bit in the interrupt enable register (TBIER) that is set "1" (Enable) and the interrupt source status flag for that bit is marked as "1", you must write "0" to both this status flag and the object status flag.

(e.g.) when TBIMIEA clears TBIMFB in a state where interrupts are Enable and TBIMIEB is prohibited

- Timer B interrupt enables the state of the register (TBIER).



- The status of the timer B status register (TBSR).

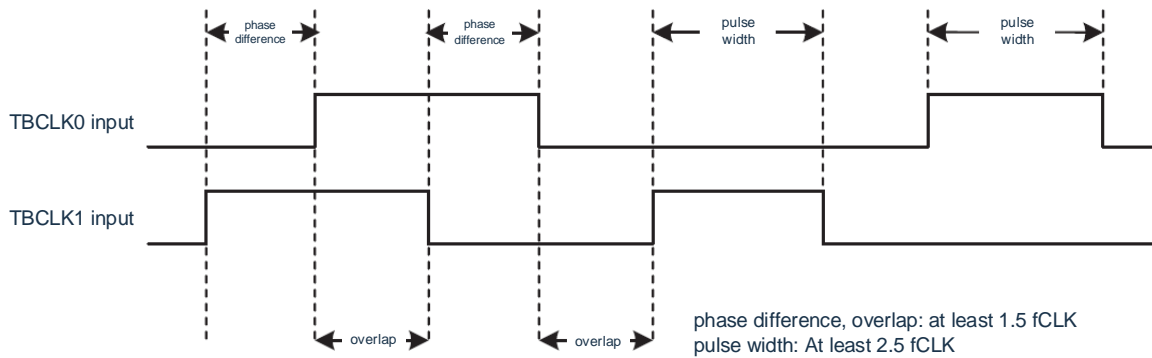


8.6 Considerations when using timer B

8.6.1 Phase difference, overlap, and pulse width in phase count mode

The phase difference and overlap of the external input signals on the TBCLK0 pin and TBCLK1 pin must be at least $1.5 f_{CLK}$ and a pulse width of at least $2.5 f_{CLK}$, respectively. The phase difference, overlap, and pulse width in phase count mode are shown in Figure 8-32.

Figure 8-32 Phase difference, overlap, and pulse width in Phase Counting mode



8.6.2 Switching modes

- To switch modes during operation, it must be done after the TBSTART position of the TBMR register is "0" (stop counting).
- The bit0 of the IF1D register must be set to "0" after switching modes and before starting running. For details, please refer to "Chapter 25 Interrupt Functions".

8.6.3 Count the switching of the source

- To switch the counting source, you must switch the note after stopping counting. Change steps
 - Place the TBSTART position of the TBMR register "0" (stop count).
 - Change the TBTCK0~TBTCK2 bit of the TBCR register.

Note The registers and bits that are forbidden to be overwritten during the counting process are as follows:

- All bits except the TBSTART bit of the TBMR register
- TBCNTC registers
- TBCR registers
- TBIOR registers

8.6.4 Setup steps for TBIO0 pins and TBIO1 pins

After reset, the multiplexed I/O ports on the TBIO0 pin and TBIO1 pins are used as input ports.

- When you want to output from the TBIO0 pin and TBIO1 pins, you must follow the steps below to set it up.

Change steps

- (1) Make the settings allowed for the mode, initial value, and output (because both the initial value and the allowed setting are made via SFR). (2) Place the position of the port register corresponding to the TBIO0 pin and the TBIO1 pin "0".
 - (3) Set the bits of the port mode registers corresponding to the TBIO0 pins and TBIO1 pins to output mode (starting with the TBIO0 pins and TBIO1 pins).
 - (4) Start counting (TBMR registers have a TBSTART bit of "1").
- To change the bit of the port mode register corresponding to the TBIO0 pin and TBIO1 pin from output mode to input mode, you must follow the steps below to set it.

Change steps

- (1) Set the bit of the port mode register corresponding to the TBIO0 pin and the TBIO1 pin to input mode (starting with the TBIO0 pin and TBIO1 pin).
 - (2) Set to input capture function.
 - (3) Start counting (TBMR registers have a TBSTART bit of "1").
- When switching the TBIO0 pin and TBIO1 pin from output mode to input mode, input capture operation may be performed depending on the state of the pin. Edge detection after at least 2 CPU clock cycles when no digital filter is used; When using digital filters, up to 5 sampling clock cycles of digital filters are required for edge detection.

8.6.5 External clocks TBBCLK0 and TBBCLK1

The pulse width of the external clock input to the TBCLKj pin (j=0, 1) must be at least 3 timer B operating clock (f_{CLK}) cycles.

8.6.6 Read and write access to SFR

To set timer B, you must first place the TMBEN position of the PER1 register "1". When the TMBEN bit is "0", the write operation of the control register of timer B is ignored, and the read values are initial values (except for the port registers and port mode registers).

(1) TBMR registers

When switching the clock of a digital filter, you must follow these steps to set it up:

- (a) Set the TBDFA bits and TBDFB bits (TBIO0 pins and) of the TBBMR registers in the state where the TBSTART bit is "0" (stop count). Digital filter function select bit on the TBIO1 pin), TBDFCK0 bit of TBMR register, and TBDFCK1 bit of TBMR register (clock select bit of digital filter function).
- (b) Place TBSTART at position "1".

However, if the digital filter is not set and the TBDFCK1 and TBDFCK0 bits that are reset to "00B" after reset are not changed, one-time setting can be made.

In addition to the external input pins (TBIO0, TBIO1), event events for the EVENTC input can be selected as the operating source for input capture. When this feature is not used, the TBELCICE position of the TBMR register must be set to "1" and the input snap function must be set (the effective edge of the input snap is the rising edge (TBIOB2~ TBIOB0=100B)). This function is invalid when using the output comparison function in PWM mode or timer mode (TBPWM=1, TBIOB2=0).

(2) TB registers

- The write operation priority of the TBMR registers is timer B operating condition resulting in a count reset.

8.6.7 Stop counting when the input snap runs

In input capture mode, input is given to the TBIO0/TBIO1 pin if the TBSTART bit of the TBBR register is "0" (stop count).

The TBIOj0 bit and TBIOj1 bits of the TBIOR registers are selected at the edges that generate an input capture interrupt request at the effective edge of the TBIO0/TBIO1 input (j=A, B).

Chapter 9 Timer C

9.1 Function of timer C

Timer C is a timer that can trigger the input capture function via software, comparator 1, and timer M.

The actions are as follows:

Count Start: The count action is triggered by software or timer M

Count Stop: The count stop is triggered by the software or the output of Comparator 1

Input capture: When an interrupt from comparator 1 occurs, the count value is transferred to the buffer

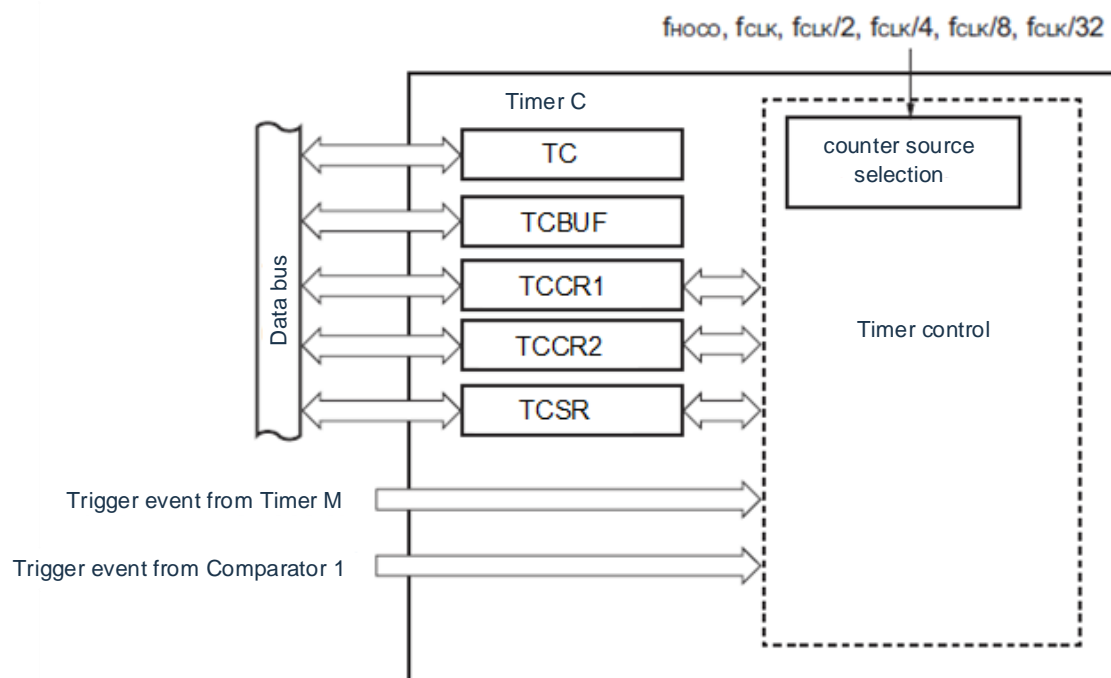
Count Reset: The count reset is triggered by timer M or comparator 1

The action clock of Timer C is f_{CLK} .

9.2 Structure of timer C

The block diagram of timer C is shown in Figure 9-1.

Figure 9-1 Block diagram of timer C



9.3 Control registers of timer C

The registers that control timer C are shown in Table 9-1.

Table 9-1 control registers of timer C

Register name	symbol
Peripheral enable register 1	PER1
Timer C count register	TC
Timer C counts buffer registers	TCBUF0
Timer C controls register 1	TCCR1
Timer C controls register 2	TCCR2
Timer C status register	TCSR

9.3.1 Peripheral enable register 1 (PER1).

The PER1 register is a register that is set to enable or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use.

To use timer C, bit1 (TMCEN) must be set to "1". The PER1 register is set via the 8-bit memory operation instruction. After generating a reset signal, the value of this register changes to "00H".

Figure 9-2 format of Peripheral enable register 1 (PER1)

Address: 0x4002081A after reset: 00H R/W

symbol

PER1

7	6	5	4	3	2	1	0
DACEN	TMBEN	PGACMPEN	TMMEN	DButIn	PWMOPEN	TMCEN	TButIn

TMCEN	Provides control of the input clock of timer C
0	Stop providing the input clock. • SFR cannot be written to timer C. • Timer A is reset.
1	An input clock is provided. • SFR can be read and written to timer C.

Note 1 To set the timer C, you must first place the TMCEN position "1". When the TMCEN bit is "0", the write operation of the control register of timer C is ignored, and the read values are the initial values.

9.3.2 Timer C count register (TC).

This is the 16-bit register. If you write this register, you write the data to the reload register. If you read this register, you read the count value.

Figure 9-3 Format of the timer C count register (TC).

Address: 0x40042C50 After reset: 0000H R/W

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

TC															
----	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

—	function	Set the scope
bit15~0	Increment the count, and the TCOVF bit of the TCSR is set to 1 when the overflow occurs	0000H~FFFFH

9.3.3 Timer C count buffer register (TCBUF).

Figure 9-4 Format of timer C count buffer register (TCBUF).

Address: 0x40042C52 After reset: 0000H R

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

TCBUF															
-------	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

—	function	Set the scope
bit15~0	When an interrupt of comparator 1 occurs, the value of the T C register is transferred to the buffer register	0000H~FFFFH

9.3.4 Timer C controls register 1 (TCCR1).

Figure 9-5 Timer C controls the format of register 1 (TCCR1).

Address: 0x40042C54 After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
TCCR1	TCK2	TCK1	TCK0	START_MD	TRIG_MD_SW	TRIG_MD_HW	TM_TRIG	OVIE

TCK2	TCK1	TCK0	Count source select Note
0	0	0	fCLK
0	0	1	fCLK/2
0	1	0	fCLK/4
0	1	1	fCLK/8
1	0	0	fCLK/32
Other than the above			Prohibit settings

START_MD	Count the selection of the starting source
0	The software sets the Timer C count to start
1	The output signal of Timer M triggers the start of the Timer C count
When START_MD=1, the count begins after the TSTART bit of TCCR2 is set to 1	

TRIG_MD_SW	The software resets the valid signal of Timer C
0	Disables software resetting the count counter of Timer C
1	Enables the software to reset the count counter of Timer C
Invalid when START_MD=1	

TRIG_MD_HW	Action selection when Timer C is triggered via the output of Timer M
0	Start counting after resetting Timer C
1	Timer C starts counting
Invalid when START_MD=0	

TM_TRIG	Hardware-triggered selection from Timer M
0	The count start action of TM0 (TSTART0 is set to 1) triggers the Timer C start count
1	The count start action of TM1 (TSTART1 is set to 1) triggers the Timer C start count
Invalid when START_MD=0	

OVIE	Enables overflow interrupt signals to be generated
0	Disables interrupts when the TC register overflows
1	Enables an interrupt to occur when the TC register overflows

Note: When working with Timer M, the action clock of Timer C must be set to match the frequency of timer M's action clock.

9.3.5 Timer C controls register 1 (TCCR2).

Figure 9-6 Timer C controls the format of register 1 (TCCR2).

Address: 0x40042C55 After reset: 00H R/W

Symbol 7 6 5 4 3 2 1 0

TCCR2	0	0	0	0	0	CMP_TCR1	CMP_TCR0	TSART
-------	---	---	---	---	---	----------	----------	-------

CMP1_TCR1	CMP1_TCR0	The action selection when Timer C is triggered by the output of comparator 1
0	0	The Timer C count stops
0	1	The Timer C count value is passed to the buffer register and the count continues
1	0	The Timer C count value becomes 0000H and the count continues
1	1	The Timer C count is passed to the buffer register, the Timer C count value becomes 0000H, and the count continues

TSTART	The Timer C action begins to control the note
0	The T-C count stops
1	TC count starts

Note: If the stop signal control from CMP1 and the T START control compete, the stop signal from comparator 1 has a higher priority.

9.3.6 Timer C status register (TCSR).

Figure 9-7 Format of timer C control register 1 (TCSR).

Address: 0x40042C56 After reset: 00H R/W

Symbol 7 6 5 4 3 2 1 0

TCSR	0	0	0	0	0	0	TCSB	TCOVF
------	---	---	---	---	---	---	------	-------

TCSB	Timer C counter status flag bit Note 1
0	Count stops
1	Count

TCOVF	The Overflow Status Flag bit of the Timer C counter is noted in Note 2, Note 3
0	No overflow occurred
1	An overflow occurred

Note 1: Read-only, not writable.

Note 2: Only 0 can be written, and 1 is invalid.

Note 3: When the Timer C counter overflow and TCOVF write 0 occur at the same time, the overflow has a higher priority.

9.4 Operation of timer C

Timer C can start with the signal trigger count of timer M, and the signal trigger count of comparator 1 stops.

9.4.1 Count the sources

The action clock of timer C is determined by the dividing setting of timer C.

(1) Clock source for timer C

The clock source of timer C is the CPU action clock fclk.

(2) Timer C counts the clock source

Use TCCR1 to set the counting frequency.

If you use a signal from Timer M to trigger the start of the Timer C count, you must set the action clock of Timer C to match the action clock frequency of Timer M.

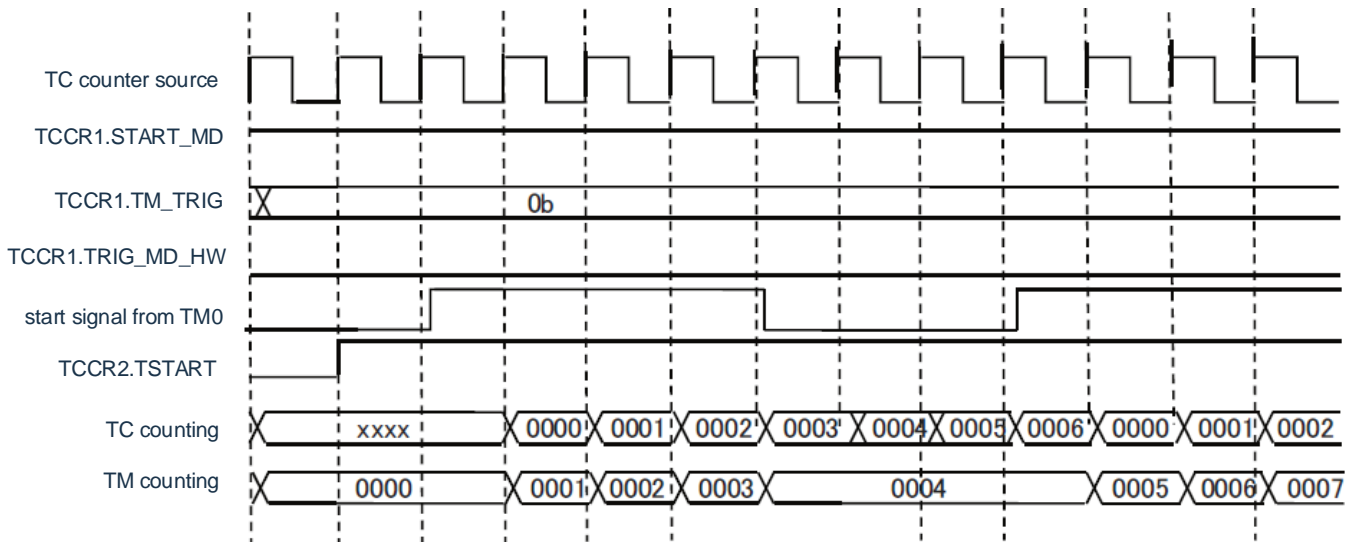
9.4.2 Timer C starts counting the actions

Through Timer M or software settings, start the timer C counting action.

9.4.2.1 Select the signal of the Timer M as the setting and action when triggered

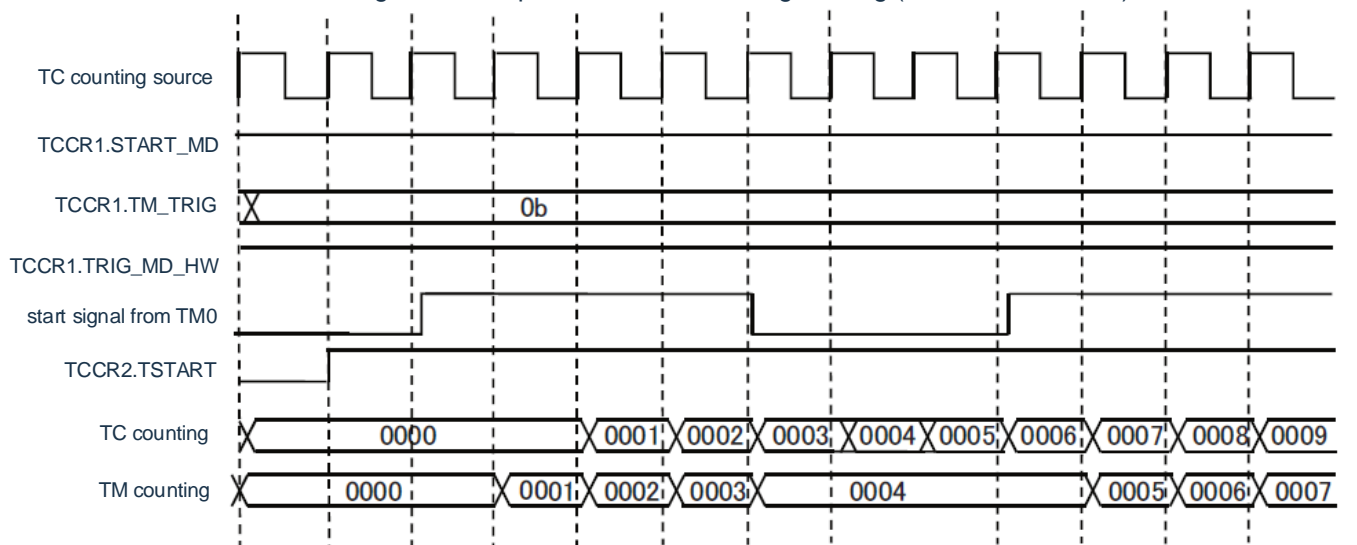
- 1) Reset and start setup steps for the Timer C count when TRIG_MD_HW=0:
 - a) Select the Timer M output signal as the trigger source for counting start: TCCR1. START_MD=1
 - b) Select the trigger function of Timer C: TCCR1. TRIG_MD_HW=0
 - c) Select Timer M_0/1 for the trigger signal: TCCR1. TC_TRIG=1/0
 - d) Timer C count start: TCCR2. TSTART=1

Fig. 9-8 Example of Timer C count reset and start (TRIG_MD_HW=0).



- 2) When TRIG_MD_HW=1, the start setup step for the Timer C count:
 - a) Select the Timer M output signal as the trigger source for counting start: TCCR1. START_MD=1
- 1.
2. Select the trigger function of Timer C: TCCR1. TRIG_MD_HW=1
3. Select Timer M_0/1 for the trigger signal: TCCR1. TC_TRIG=1/0
4. Timer C count start: TCCR2. TSTART=1

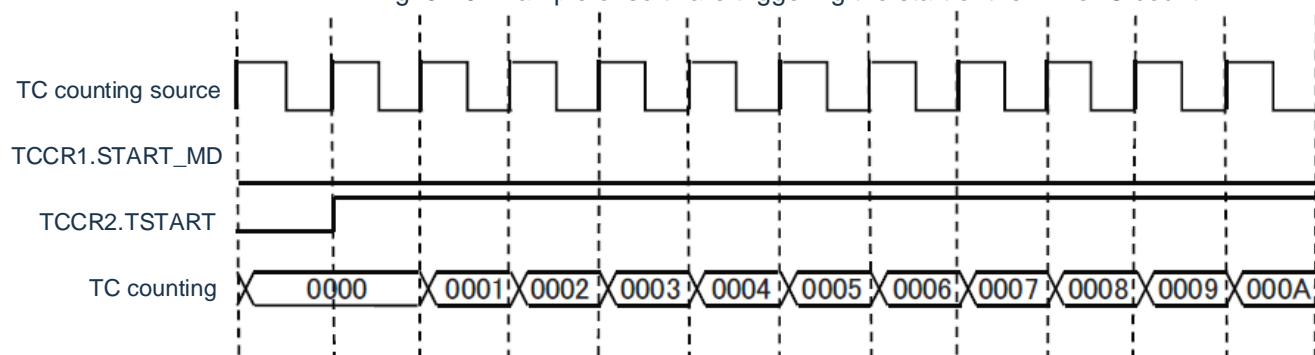
Fig. 9-9 Example of Timer C counting starting (TRIG_MD_HW =1).



9.4.2.2 Select the settings and actions when the software triggers

1. Count Start Source Selection Software Trigger: TCCR1. START_MD=0
2. Timer C count start: TCCR2. TSTART=1

Fig. 9-10 Example of software triggering the start of the Timer C count



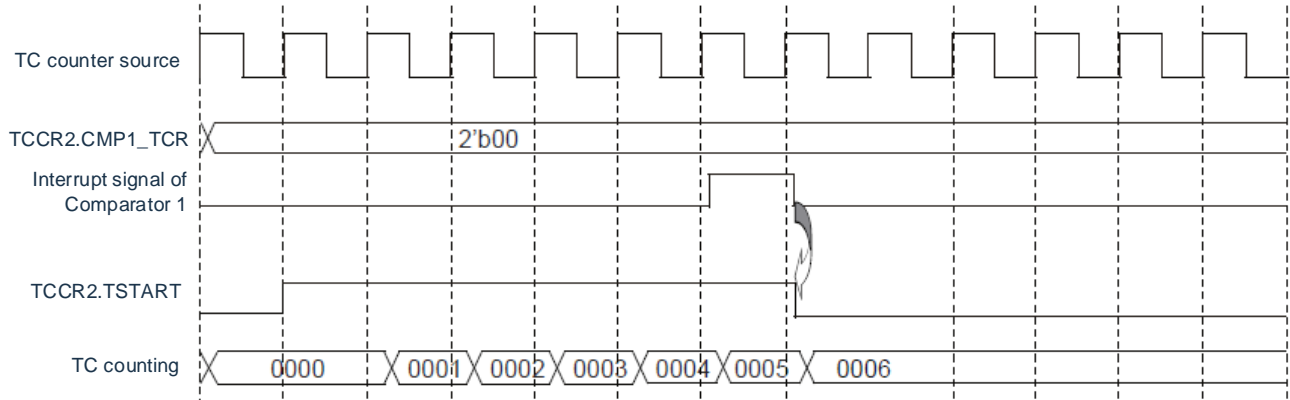
9.4.3 Timer C counts stopped actions

Timer C is in the counting and can be stopped by triggering comparator 1 or by setting the software.

9.4.3.1 Select Comparator 1 as the setting and action when triggered

1. Select Comparator 1 as the trigger: TCCR2.CMP1_TCR=00
2. Timer C count starts: TCCR2.TSTART=1

Fig. 9-11: Example of selecting comparator 1 triggering a stop in Timer C



9.4.3.2 Settings and actions when the software is triggered

1. Timer C count starts: TCCR2.TSTART=0
2. Software Settings TCCR2. The TSTART is 0 and the Timer C count stops

9.4.4 Enter the capture motion

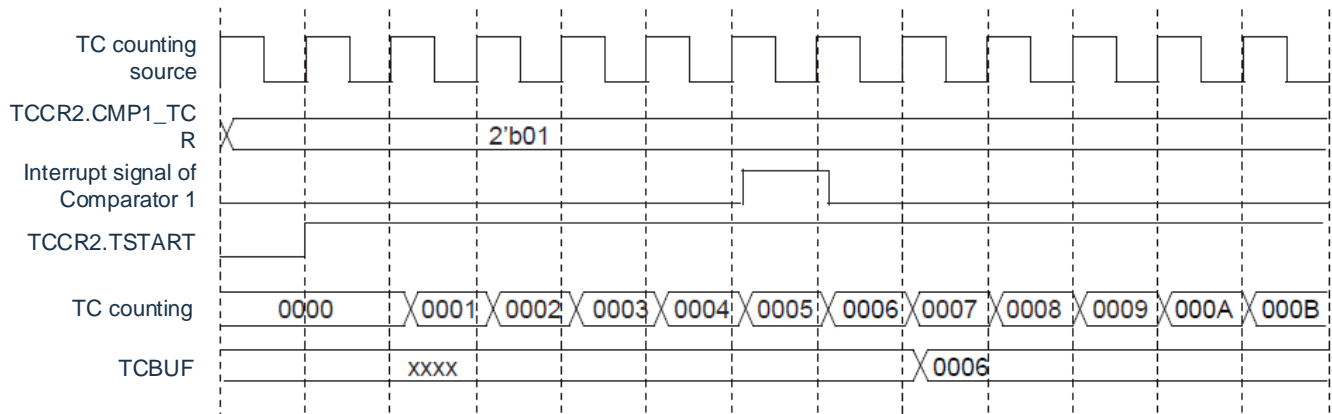
If comparator 1 produces an interrupt during the Timer C action, the Timer C action changes.

(1) Case 1:

TCCR2.CMP1_TCR=01, the count value of Timer C is transferred to the count buffer.

- TCCR2.CMP1_TCR=01 (select input capture function).
- TCCR2.TSTART=1 (Timer C count starts).

Fig. 9-12: An example of an input captured motion

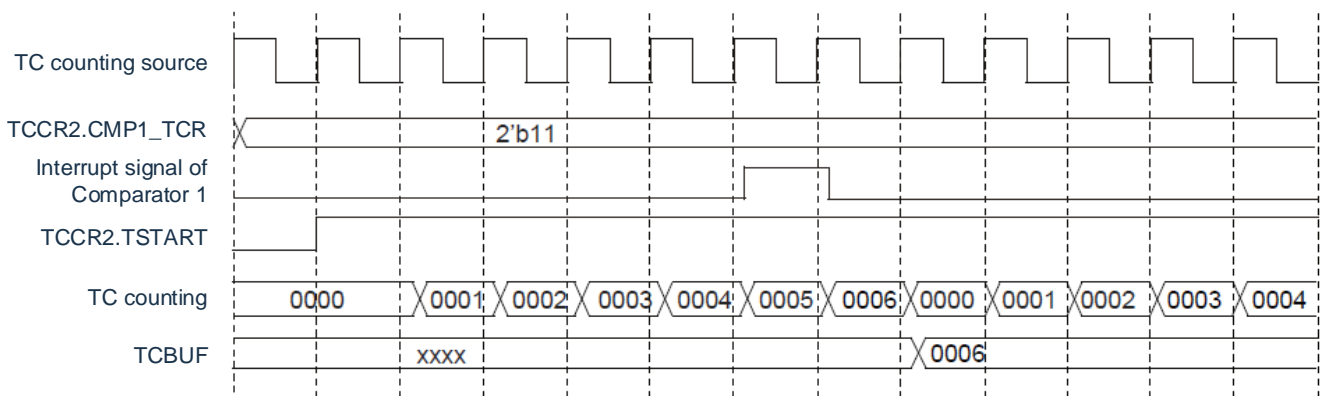


(2) Case 2:

TCCR2.CMP1_TCR=11, the count value of Timer C is transmitted to the count buffer, and the count value of Timer C is reset.

- TCCR2.CMP1_TCR=11 (select Input Capture and Reset Function).
- TCCR2.TSTART=1 (Timer C count starts).

Fig. 9-13: Example of input captured motion (reset count value at the same time)

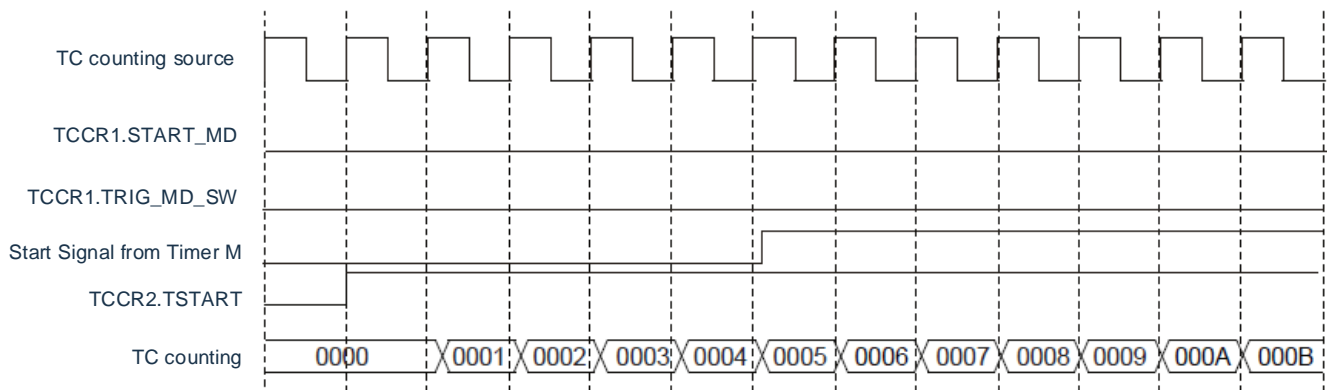


9.4.5 Timer C counts the reset action

When the Timer C action is set to begin using the software, the output signal of Timer M and the output signal of comparator 1 can reset the counter of Timer C.

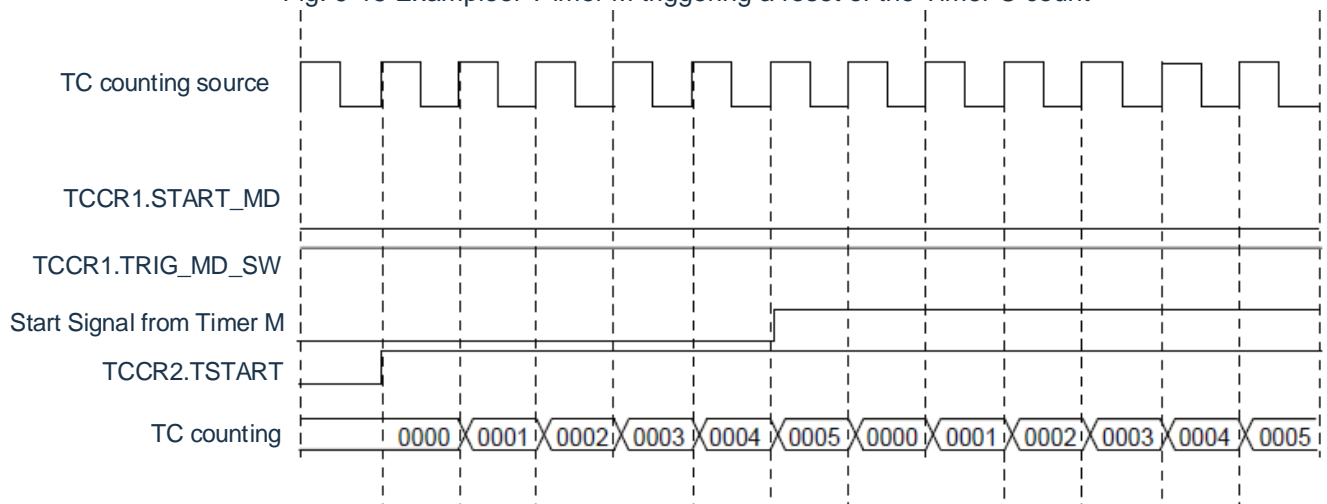
1. TRIG_MD_SW=0, the output signal of Timer M cannot reset the count value when the software triggers the count starts.
 - 1) Software Trigger Counting Begins: TCCR1. START_MD=0
 - 2) Allow software reset counter: TCCR1. TRIG_MD_SW=0
 - 3) Timer C count starts: TCCR2. TSTART=1

Fig. 9-14: Example of Timer M triggering a reset of the Timer C count



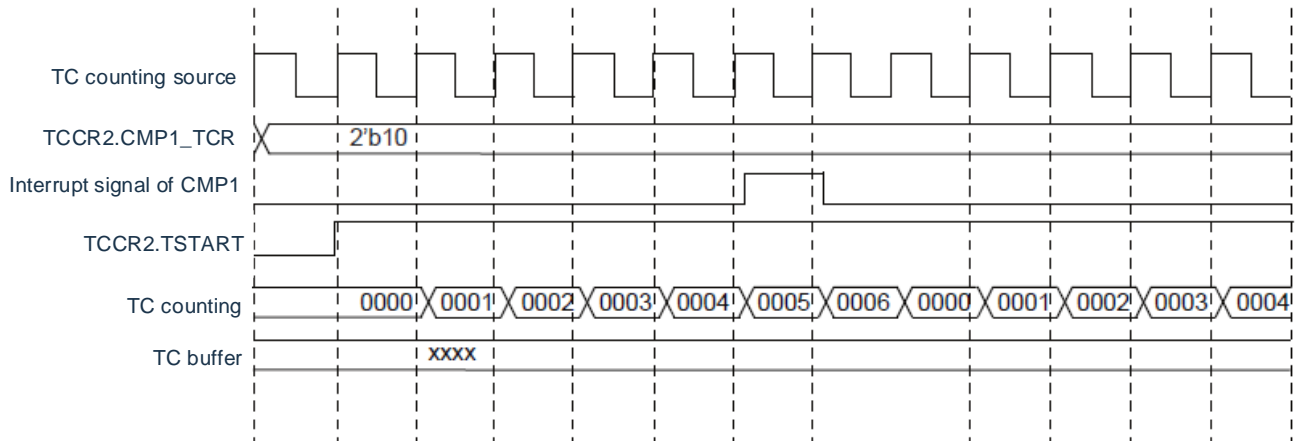
2. TRIG_MD_SW=1, the output signal of Timer M resets the count value when the software triggers the count starts.
 - 1) Software Trigger Counting Begins: TCCR1. START_MD=0
 - 2) Disable Software Reset Counter: TCCR1. TRIG_MD_SW=1
 - 3) Timer C count starts: TCCR2. TSTART=1

Fig. 9-15 Example of Timer M triggering a reset of the Timer C count



3. When TCCR2.CMP1_TCR=10, the output signal of CMP1 resets the count value.
 - 1) The count value is reset and the count action continues: TCCR2.CMP1_TCR=10 (input capture function cannot be used).
 - 2) Timer C count starts: TCCR2.TSTART=1

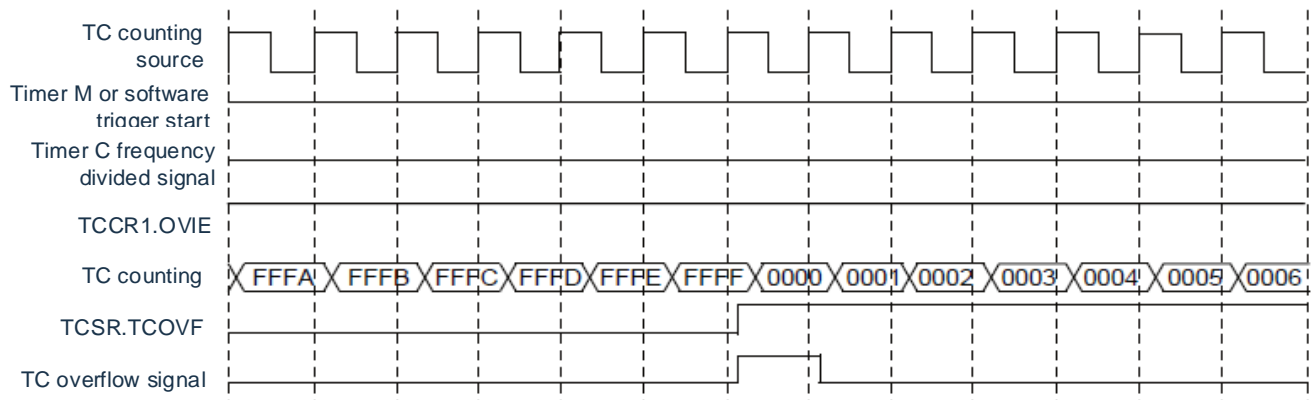
Fig. 9-16: Example of CMP1 triggering a Timer C count reset



9.4.6 Interrupt of timer C

Timer C counter overflows if TCCR1 is set. OVIE=1, which produces an overflow interrupt signal.

Fig. 9-17: Example of interrupt generation when the Timer C overflows



9.5 Precautions when using Timer C

9.5.1 Read and write registers

To set the timer C, you must first place the TMCEN position "1" of P ER1. When the TMCEN bit is "0", the write operation of the control register of timer C is ignored, and the read values are the initial values.

When the clock stops, registers other than TCBUF cannot be written except for T C, which cannot be written.

Note The following registers in the count action cannot be overwritten.

- TCCR1
- TCCR2

9.5.2 Overflow interruption

When the count value of Timer C is FFFFH, an overflow interrupt does not occur if the trigger event of the external input before the overflow reaches 0 000H resets the counter.

9.5.3 Input capture and timer C count reset action

Even if you set up TCSR. TCSB = 0 (count stop), timer M and CMP1 input signals can still trigger Timer C input capture motion and count reset action.

9.5.4 Timer C and Timer M, comparator 1 are linked

When timer C and timer M, comparator 1 are linked, the setup steps are as follows:

1. Provides a clock input for comparator 1: PGACMPEN=1
2. Enable Comparator 1 interrupt generation and output: See Chapter 17 Comparator CMP for details.
3. Clock input for timer C is provided: TMCEN=1
4. Set up TCCR1
5. Set up TCCR2
6. Set the timer M, and the timer M count starts: TMOEN=1
7. Timer C count start: TCCR2. START=1

Note 1: When Timer C is linked to Timer M and CMP1, the action clock of Timer C must be set to match the action clock frequency of Timer M.

Note 2: When setting the register, you must first set the control register TCCR1 and then set the TCCR2. TSTART

Chapter 10 Timer M

10.1 Function of timer M

Timer M has the following 4 modes:

- Timer mode
 - The input capture function is triggered by an external signal to take the count value to the register.
 - The output comparison function detects whether the count value and the register value are the same (the output of the pin can be changed at the time of detection).
 - The PWM function continuously outputs any pulse width.

The following 3 modes use the PWM function:

- Reset Synchronous PWM Mode This is a mode that outputs sawtooth wave modulation with no dead time three-phase waveforms (6).
- Complementary PWM mode This is a mode of output triangular wave modulation with dead time three-phase waveforms (6).
- PWM3 mode This is the mode of outputting the same-period PWM waveform (2).

In timer mode, timer M0 and timer M1 have equal input capture functions, output comparison functions, and PWM functions, which can be selected at each pin and can be used in combination in timer M0 and timer M1.

Reset synchronous PWM mode, complementary PWM mode, and PWM3 mode output waveforms via a combination of counters and registers from timer M0 and timer M1, with pin function depending on the operating mode.

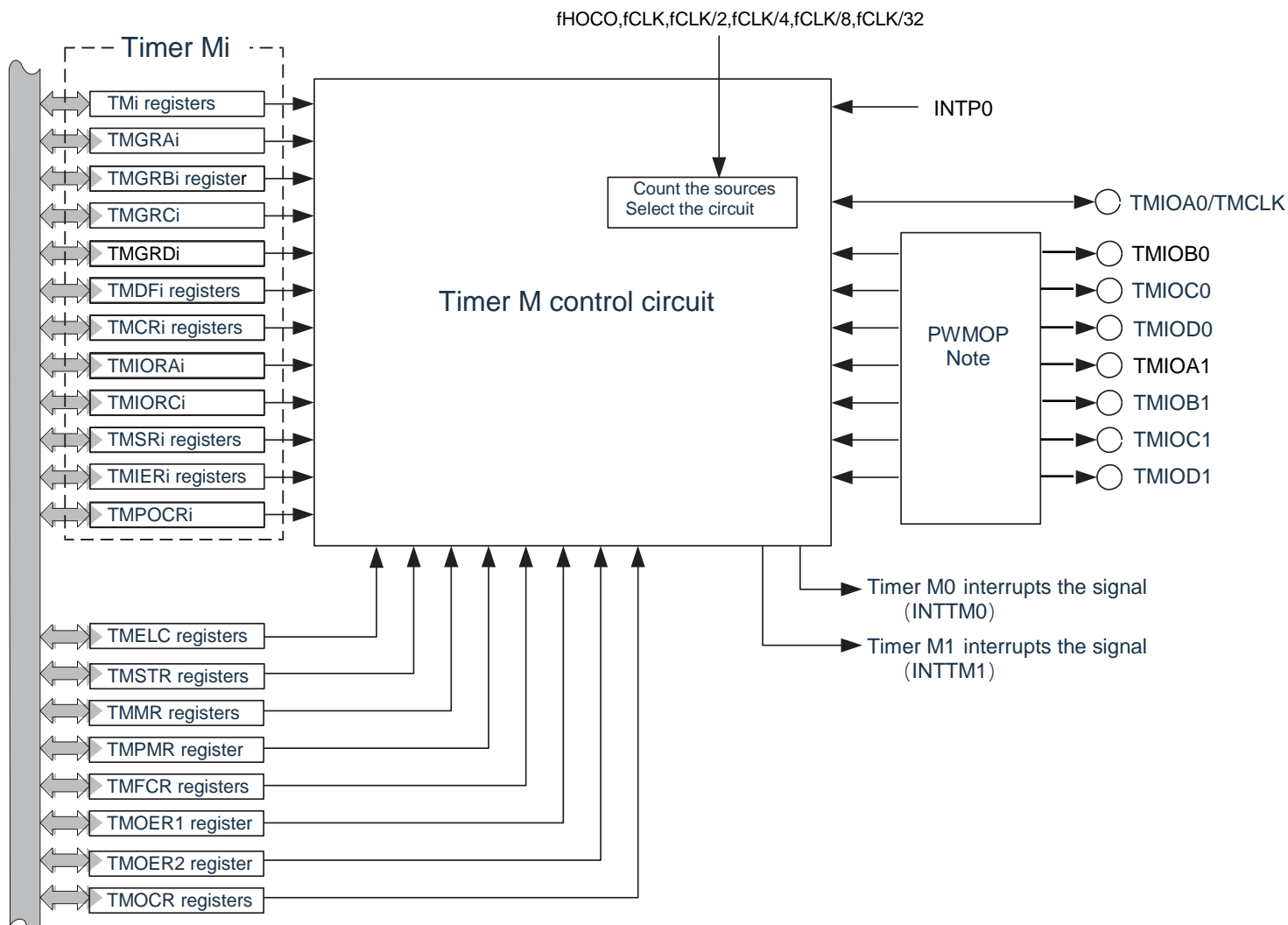
Timer M has 4 input/output pins.

The operating clock of timer M is f_{CLK} or f_{HOCO} .

10.2 Structure of timer M

The block diagram and pin structure of timer M are shown in Figure 10-1 and Table 10-1, respectively.

Figure 10-1 Block diagram of timer M



Note: Only the output can be cut-off

Note: $i=0,1$

Table 10-1 Pin structure of timer M

Pin name	The port name of the multiplex	Input/Output	function
TMIOA0/TMCLK	P17	Input/Output	Functions vary depending on the operating mode, please refer to each mode for details.
TMIOB0	P14	Input/Output	
TMIOC0	P16	Input/Output	
TMIOD0	P15	Input/Output	
TMIOA1	P12	Input/Output	
TMIOB1	P10	Input/Output	
TMIOC1	P13	Input/Output	
TMIOD1	P11	Input/Output	

10.3 Control register of timer M

The registers that control timer M are shown in Table 10-2.

Table 10-2 control registers of timer M

Register name	symbol
Peripheral enable register 1	PER1
Timer M EVENTC register	TMELC
Timer M starts the register	TMSTR
Timer M mode register	TMMR
Timer MPWM function selects registers	TMPMR
Timer M function control register	TMFCR
Timer M outputs the main enable register 1	TMOER1
Timer M outputs the main enable register 2	TMOER2
Timer M outputs the control register	TMOCR
Timer M digital filter function selects register 0	TMDF0
Timer M digital filter function select register 1	TMDF1
Timer M control register 0	TMCR0
Timer M I/O control register A0	TMIORA0
Timer M I/O control register C0	TMIORC0
Timer M-state register 0	TMSR0
Timer M interrupt enable register 0	TMIER0
The timer M PWM function outputs level control register 0	TMPOCR0
Timer M meter 0	TM0
Timer M uses register A0	TMGRA0
Timer M uses register B0	TMGRB0
Timer M general purpose register C0	TMGRC0
Timer M general purpose register D0	TMGRD0
Timer M control register 1	TMCR1
Timer M I/O controls register A1	TMIORA1
Timer M I/O control register C1	TMIORC1
Timer M status register 1	TMSR1
Timer M interrupt enable register 1	TMIER1
Timer M PWM functions output level control register 1	TMPOCR1
Timer M Meter 1	TM1
Timer M general purpose register A1	TMGRA1
Timer M general purpose register B1	TMGRB1
Timer M general purpose register C1	TMGRC1
Timer M general purpose register D1	TMGRD1
Port registers	Pxx
Port mode registers	PMx,PMCx

10.3.1 Peripheral enable register 1 (PER1).

Per1 registers are registers that are set to allow or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use.

To use timer M, bit4 (TMMEN) must be set to "1". The PER1 register is set via the 8-bit memory operation instruction. After generating a reset signal, the value of this register changes to "00H".

Figure 10-2 Format of Peripheral enable register 1 (PER1)

Address: 0x4002081A after reset: 00H R/W

symbol	7	6	5	4	3	21	0	
PER1	DACEN	TMBAndN	PGACMPEN ^{note}	TMMEN	DMAEN	PWMOPEN	TMCEN	TMAEN

TMMEN	Provides control of the input clock of timer M
0	Stop providing the input clock. • SFR used by timer M cannot be written. • Timer M is reset.
1	An input clock is provided. • SFR that can be used in the read-write timer M.

PWMOPEN	PWM cut-off control circuitry for control of the input clock
0	Stop providing the input clock. • Do not write the SFR used in the PWM cut-off circuit. • The PWM cut-off circuit is reset.
1	An input clock is provided. • Can read and write SFR used in PWM cut-off circuits.

Note 1 To set the timer M, you must first place the TMMEN position "1". When the TMMEN bit is "0", the write operation of the control register of timer M is ignored, and the read values are both the initial values (port mode register 1 (PM1) and port register 1). (P1)).

- To select fHOCO as the counting source for timer M, you must set the bit4 (TMMEN) of peripheral enable register 1 (PER1) before setting it fCLK is set to fIH. If you want to change the fCLK to a clock other than fIH, you must set bit4 of register 1 (PER1) in the peripheral to be cleared (TMMEN) after making changes.
- To set PWMOP, you must first place the PWMOPEN position "1". When the PWMOPEN bit is "0", the write operation of the control register of the PWMOP is ignored, and the read values are the initial values. See "10.8 PWMOP".

10.3.2 Timer M EVENTC register (TMELC).

Figure 10-3 Format of the timer M EVENTC register (TMELC).

Address: 0x40042A60 after reset: 00H R/W

symbol	7	6	5	4	3	21	0
TMELC	0	0	ELCOBE1	ELCICE1	0	0	ANDLCOBE0 ELCICE0

ELCOBE1	Permissibility of EVENTC event input 1 (pulse output for forced cut-off timer)
0	Mandatory cut-off is prohibited.
1	Forced cutoffs are allowed.

ELCICE1	Selection of EVENTC event input 1 (input capture D1 for timer M).
0	Select Input Capture TMIOD1.
1	Select Event Input 1 from the Event Linkage Controller (EVENTC).

ELCOBE0	Permissibility of EVENTC event input 0 (pulse output for forced cut-off timer)
0	Mandatory cut-off is prohibited.
1	Forced cutoffs are allowed.

ELCICE0	Selection of EVENTC event input 0 (input capture D0 for timer M).
0	Select Input Capture TMIOD0.
1	Select event input 0 from the Event Linkage Controller (EVENTC).

10.3.3 Timer M Start Register (TMSTR).

The TMSTR register can be set via an 8-bit memory operation instruction. Please refer to "10.7.1(1) TMSTR Registers" for Precautions When Using Timer M.

Figure 10-4 Format of the Start Register (TMSTR) for timer M

Address: 0x40042A63 reset: 0CH R/W

symbol	7	6	5	4	3	21	0
TMSTR	0	0	0	0	CSEL1	CSEL0	TSTART1 TSTART0

CSEL1	TM1 counts execution selection ^{note 1}
0	Stop counting when matching TMGRA1 register comparison.
1	The count continues after matching with the TMGRA1 register comparison ^{note 2} .

CSEL0	TM0 counts execution selection
0	Stops counting when matching TMGRA0 register comparison.
1	Continue counting ^{note 2} after matching with the TMGRA0 register comparison.

TSTART1	The start flags for the TM1 count ^{note 3, 4}
0	Stop count.
1	Start counting.

TSTART0	The start flags of the TM0 count ^{note 5, 6}
0	Stop count.
1	Start counting.

Note 1 PWM3 mode cannot be used.

2. When using the input capture feature, this position must be "1".
3. When the CSEL1 bit is "1", the TSTART1 bit must be written "0".
4. When the CSEL1 bit is "0" and produces a comparison match signal (TMIOA1), this flag is "0" (stop count).
5. When the CSEL0 bit is "1", you must write "0" to the TSTART0 bit.
6. When the CSEL0 bit is "0" and a comparison match signal (TMIOA0) is generated, this flag is "0" (stop count).

10.3.4 Timer M mode register (TMMR).

Figure 10-5 Format of the mode register (TMMR) for timer M

Address: 0x40042A64 reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TMMR	TMBFD1	TMBFC1	TMBFD0	TMBFC0	0	0	0	TMSYNC

TMBFD1	TMGRD1 register function selection ^{note 1}
0	General Purpose registers
1	Buffer register for the TMGRB1 register

TMBFC1	Select ^{note 1} for TMGRC1 register function
0	General Purpose registers
1	Buffer register for the TMGRA1 register

TMBFD0	Option ^{note 1} for TMGRD0 register function
0	General Purpose registers
1	Buffer register for the TMGRB0 register

TMBFC0	TMGRC0 register function selection ^{notes 1, 2}
0	General Purpose registers
1	Buffer register for TMGRA0 registers

TMSYNC	Synchronized ^{note 3} for timer M
0	TM0 and TM1 operate independently.
1	TM0 and TM1 run synchronously.

- Note 1 When using the output comparison function, if you pass the TMIORCi register (i=0, 1) IOj3 (j=C or D) bit to select "0" (change the output pin of the TMGRji register), you must change the TMBFji position of the TMMR register to "0".
2. In the complementary PWM mode, this position must be "0" (general purpose register).
3. In reset synchronous PWM mode, complementary PWM mode, or PWM3 mode, this position must be "0" (TM0 and TM1 Stand-alone operation).

10.3.5 Timer M PWM function select register (TMPMR).

Figure 10-6 Format of the timer M PWM function selection register (TMPMR) [Timer Mode].

Address: 0x40042A65 reset: 00HR/W

symbol	7	6	5	4	3	2	1	0
TMPMR	0	TMPWMD1	TMPWMC1	TMPWMB1	0	TMPWMD0	TMPWMC0	TMPWMB0

TMPWMD1	TMIOD1's PWM feature selection
0	Input capture function or output comparison function
1	PWM function

TMPWMC1	TMIOC1's PWM feature selection
0	Input capture function or output comparison function
1	PWM function

TMPWMB1	TMIOB1's PWM feature selection
0	Input capture function or output comparison function
1	PWM function

TMPWMD0	TMIOD0's PWM feature selection
0	Input capture function or output comparison function
1	PWM function

TMPWMC0	TMIOC0's PWM feature selection
0	Input capture function or output comparison function
1	PWM function

TMPWMB0	TMIOB0's PWM feature selection
0	Input capture function or output comparison function
1	PWM function

10.3.6 Timer M function control register (TMFCR).

Figure 10-7 Timer M function control register (TMFCR) format

Address: 0x40042A66 Reset: 80H R/W

symbol	7	6	5	4	3	2	1	0
TMFCR	HPM3	STCLK	0	0	The LS1	OLS0	CMD1	CMD0

PWM3	PWM3 mode selection ^{note 1}
<ul style="list-style-type: none"> In timer mode, "1" (not PWM3 mode) must be set. In PWM3 mode, "0" (PWM3 mode) must be set. Not valid in reset synchronous PWM mode and complementary PWM mode. 	

STCLK	Selection of external clock inputs
<ul style="list-style-type: none"> Timer mode, reset synchronous PWM mode, complementary PWM mode 0: The external clock input is invalid 1: The external clock input is valid In PWM3 mode, "0" must be set (the external clock input is invalid). 	

OLS1	Selection of inverting output levels (reset synchronous PWM mode or complementary PWM mode).
<ul style="list-style-type: none"> Reset synchronous PWM mode, complementary PWM mode 0: Initial output "H" level, "L" level is valid. 1: Initial output "L" level, "H" level is active. Not valid in timer mode and PWM3 mode. 	

OLS0	Selection of normal-phase output levels (reset synchronous PWM mode or complementary PWM mode).
<ul style="list-style-type: none"> Reset synchronous PWM mode, complementary PWM mode 0: Initial output "H" level, "L" level is valid. 1: Initial output "L" level, "H" level is active. Not valid in timer mode and PWM3 mode. 	

CMD1	CMD0	Select ^{Notes 2 and 3} for the combination mode
<ul style="list-style-type: none"> In timer mode and PWM3 mode, "00B" (timer mode or PWM3 mode) must be set. In reset synchronous PWM mode, "01B" (Reset synchronous PWM mode) must be set. Complementary PWM models 		
CMD1 CMD0 10 : Complementary PWM mode (data is transferred from the buffer register to the general purpose register in the event of a underflow in TM1). 11 : Complementary PWM mode (data is transferred from the buffer register to the general-purpose register when the TM0 and TMGRA0 registers are relatively matched). Other than the above: Prohibit settings.		

Note 1 When cmd1 bit and CMD0 bit are "00B" (timer mode or PWM3 mode), the PWM3 bit setting is valid.

2. CMD0 bits and CMD1 must be written when both TSTART0 bits and TSTART1 bits of the TMSTR register are "0" (stop count). Bit.

3. When the CMD1 bit and CMD0 bits are "01B", "10B", or "11B", it is independent of the setting of the TMPMR register, for reset synchronous PWM mode or complementary PWM mode.

10.3.7 Timer M output master enable register 1 (TMOER1).

Figure 10-8 Timer M outputs the format of the main enable register 1 (TMOER1).
[Output Comparison Function, PWM Function, Reset Synchronous PWM Mode, Complementary PWM Mode, and PWM3 Mode].

Address: 0x40042A67 reset: FFH R/W

symbol	7	6	5	4	3	2	1	0
TMOER1	ED1	EC1	EB1	EA1	ED0	EC0	EB0	EA0

ED1	Disable ^{note1} for TMIOD1 output
0	Output is allowed.
1	Disable output (TMIOD1 pin is an I/O port).

EC1	Disable ^{note1} for TMIOC1 output
0	Output is allowed.
1	Disable output (TMIOC1 pin is an I/O port).

EB1	Disable ^{note1} for TMIOB1 output
0	Output is allowed.
1	Disable output (TMIOB1 pin is an I/O port).

EA1	Disable ^{note1 and 2} for TMIOA1 output
0	Output is allowed.
1	Disable output (TMIOA1 pin is an I/O port).

ED0	Disable ^{note 1} for TMIOD0 output
0	Output is allowed.
1	Disable output (TMIOD0 pin is an I/O port).

EC0	Disable ^{note 1} for TMIOC0 output
0	Output is allowed.
1	Disable output (TMIOC0 pin is an I/O port).

EB0	TMIOB0 output is prohibited
0	Output is allowed.
1	Disable output (TMIOB0 pin is an I/O port).

EA0	Disable ^{notes 2 and 3} for TMIOA0 output
0	Output is allowed.
1	Disable output (TMIOA0 pin is an I/O port).

- Note
- 1 In PWM3 mode, this position must be "1".
 2. When using the PWM function, this position must be "1".
 3. In reset synchronous PWM mode and complementary PWM mode, this position must be "1".

10.3.8 Timer M output main enable register 2 (TMOER2).

Figure 10-9 Timer M outputs the format of the main enable register 2 (TMOER2).
[PWM function, reset synchronous PWM mode, complementary PWM mode, and PWM3 mode].

Address: 0x40042A68 reset: 00HR/W

symbol	7	6	5	4	3	2	1	0
TMOER2	TMPTO	0	0	0	0	0	0	TMSHUTS

TMPTO	The pulse output forces the cutoff signal in the INTP0 pin input valid
0	The pulse output force cutoff input is invalid.
1	The pulse output force cutoff input is valid (if the "L" level is input to the INTP0 pin, the TMSHUTS bit is "1").

TMSHUTS	Force cutoff flags
0	No cutoff is enforced.
1	Is in a forced cutoff.
When the cutoff pulse is forced by the INTP0 pin or an ELC input event, this bit becomes "1" and is not automatically cleared. Therefore, to stop the forced cutoff pulse, you must write "0" to this bit during the stop count (TSTARTi=0). Even writing "1" to the TMSHUTS bit in active mode forces the cutoff pulse.	

Note 1 Please refer to "Forced Cutoff of 10.4.4 Pulse Output".

10.3.9 Timer M output control register (TMOCR).

The TMOCR register must be written when both the TSTART0 bit and the TSTART1 bit of the TMSTR register are "0" (stop count).

Figure 10-10 Format of the M Output Control Register (TMOCR) of the timer [Output Comparison Function].

Address: 0x40042A69 reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TMOCR	TOD1	TOC1	TOB1	TOA1	TOD0	TOC0	TOB0	TOA0

TOD1	TMIOD1 Initial Output Level Selection ^{Note 1}
0	Initial output "L" level.
1	Initial output "H" level.

TOC1	TMIOC1 Initial Output Level Selection ^{Note 1}
0	Initial output "L" level.
1	Initial output "H" level.

TOB1	TMIOB1 Initial Output Level Selection ^{Note 1}
0	Initial output "L" level.
1	Initial output "H" level.

TOA1	Selection of the initial output level of TMIOA1
0	Initial output "L" level.
1	Initial output "H" level.

TOD0	Select ^{Note 1} for the initial output level of TMIOD0
0	Initial output "L" level.
1	Initial output "H" level.

TOC0	TMIOC0 Initial Output Level Selection ^{Note 1}
0	Initial output "L" level.
1	Initial output "H" level.

TOB0	TMIOB0 Initial Output Level Selection ^{Note 1}
0	Initial output "L" level.
1	Initial output "H" level.

TOA0	Selection of the initial output level of TMIOA0
0	Initial output "L" level.
1	Initial output "H" level.

Note 1 When the TMOCR register is set when the pin function of the TMOCR register is a waveform output, the initial output level is output.

Figure 10-11 : Format of the output control register (TMOCR) of the timer M [PWM function].

Address: 0x40042A69 after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TMOCR	TOD1	TOC1	TOB1	TOA1	TOD0	TOC0	TOB0	TOA0

TOD1	TMIOD1 Initial Output Level Selection ^{Note 1}
0	The initial output is an invalid level.
1	The initial output is the effective level.

TOC1	TMIOC1 Initial Output Level Selection ^{Note 1}
0	The initial output is an invalid level.
1	The initial output is the effective level.

TOB1	TMIOB1 Initial Output Level Selection ^{Note 1}
0	The initial output is an invalid level.
1	The initial output is the effective level.

TOA1	Selection of the initial output level of TMIOA1
must set to "0".	

TOD0	Select ^{Note 1} for the initial output level of TMIOD0
0	The initial output is an invalid level.
1	The initial output is the effective level.

TOC0	TMIOC0 Initial Output Level Selection ^{Note 1}
0	The initial output is an invalid level.
1	The initial output is the effective level.
Valid in reset synchronous PWM mode and complementary PWM mode.	

TOB0	TMIOB0 Initial Output Level Selection ^{Note 1}
0	The initial output is an invalid level.
1	The initial output is the effective level.

TOA0	Selection of the initial output level of TMIOA0
must set to "0".	

Note 1 When the TMOCR register is set when the pin function of the TMOCR register is a waveform output, the initial output level is output.

Figure 10-12 Format of the output control register (TMOCR) of the timer M [PWM3 mode].

Address: 0x40042A69 after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TMOCR	TOD1	TOC1	TOB1	TOA1	TOD0	TOC0	TOB0	TOA0

TOD1	The choice of the initial output level of TMIOD1
Not valid in PWM3 mode.	

TOC1	Selection of the initial output level of TMIOC1
Not valid in PWM3 mode.	

TOB1	TMIOB1 initial output level selection
Not valid in PWM3 mode.	

TOA1	Selection of the initial output level of TMIOA1
Not valid in PWM3 mode.	

TOD0	Selection of the initial output level of TMIOD0
Not valid in PWM3 mode.	

TOC0	Selection of the initial output level of TMIOC0
Not valid in PWM3 mode.	

TOB0	TMIOB0 Initial Output Level Selection ^{Note 1}
0	The initial output "L" level, the "H" level is valid. Outputs the "H" level when TBGRB1 compares the match, and the "L" level when the TBGRB0 compares the match.
1	The initial output "H" level, the "L" level is valid. Outputs the "L" level when THE TCPRB1 compares the match, and the "H" level when the TBGRB0 compares the match.

TOA0	Selection of the initial output level of TMIOA0
0	The initial output "L" level, the "H" level is valid. Outputs the "H" level when TMGRA1 compares matches and the "L" level when TMGRA0 compares matches.
1	The initial output "H" level, the "L" level is valid. Outputs the "L" level when TMGRA1 compares matches and the "H" level when TMGRA0 compares matches.

Note 1. When the TMOCR register is set when the pin function of the TMOCR register is a waveform output, the initial output level is output.

10.3.10 The timer M digital filter function selects register i (TMDFi) (i=0, 1).

Fig. 10-13 Format of timer M digital filter function selection register i (TMDFi) (i=0, 1) [Input Capture Function]

Address: 0x40042A6A (TMDF0), 0x40042A6B (TMDF1) after reset:00H R/W

symbol	7	6	5	4	3	2	1	0
TMDFi	DFCK1	DFCK0	PENB1	PENB0	DFD	DFC	DFB	DFA

DFCK1	DFCK0	The digital filter function clock selection ^{Note 1}
0	0	f _{CLK} /32
0	1	f _{CLK} /8
1	0	f _{CLK}
1	1	Count source (clock selected from TCK0 to TCK2 bits of TMCRI registers).

PENB1	PENB0	Control of pulse forced cutoff at the TMIOB pin
0	0	must set "00B".

DFD	TMIODi pins for the selection of digital filter functions
0	There is no digital filter function.
1	There is a digital filter function.

When the digital filter function is available, up to 5 sampling clock cycles of the digital filter are required for

DFC	TMIOCi pin of digital filter function selection
0	There is no digital filter function.
1	There is a digital filter function.

When the digital filter function is available, up to 5 sampling clock cycles of the digital filter are required for

DFB	TMIOBi pin digital filter function selection
0	There is no digital filter function.
1	There is a digital filter function.

When the digital filter function is available, up to 5 sampling clock cycles of the digital filter are required for

DFA	TMIOAi pin of digital filter function selection
0	There is no digital filter function.
1	There is a digital filter function.

When the digital filter function is available, up to 5 sampling clock cycles of the digital filter are required for

Note 1 Counting must begin after setting the DFCK0 and DFCK1 bits.

Fig. 10-14 Timer M digital filter function selection register i (TMDFi) (i=0, 1)
format [PWM function, reset synchronization PWM mode, complementary PWM mode, and PWM3 mode].

Address: 0x40042A6A (TMDf0), 0x40042A6B (TMDf1) after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TMDFi	DFCK1	DFCK0	PENB1	PENB0	DFD	DFC	DFB	DFA

DFCK1	DFCK0	Control of pulse forced cutoff at the TMIOA pin
0	0	Mandatory cut-off is prohibited.
0	1	High impedance output
1	0	"L" level output
1	1	"H" level output
If the corresponding pin is not used as the output port for timer M in these modes, it must be set to "00B" (forced cutoff is prohibited). Also, these bits must be set during the stop count process.		

PENB1	PENB0	Control of pulse forced cutoff at the TMIOB pin
0	0	Mandatory cut-off is prohibited.
0	1	High impedance output
1	0	"L" level output
1	1	"H" level output
If the corresponding pin is not used as the output port for timer M in these modes, it must be set to "00B" (forced cutoff is prohibited). Also, these bits must be set during the stop count process.		

DFD	DFC	Control of pulse forced cutoff at the TMIOC pin
0	0	Mandatory cut-off is prohibited.
0	1	High impedance output
1	0	"L" level output
1	1	"H" level output
If the corresponding pin is not used as the output port for timer M in these modes, it must be set to "00B" (forced cutoff is prohibited). Also, these bits must be set during the stop count process.		

DFB	DFA	Control of pulse force cutoff at the TMIOD pin
0	0	Mandatory cut-off is prohibited.
0	1	High impedance output
1	0	"L" level output
1	1	"H" level output
If the corresponding pin is not used as the output port for timer M in these modes, it must be set to "00B" (forced cutoff is prohibited). Also, these bits must be set during the stop count process.		

10.3.11 Timer M controls register i (TMCRI) (i=0, 1).

The TMCRI register is not used in reset synchronous PWM mode and PWM3 mode.

Fig. 10-15 Format of timer M control register i (TMCRI) (i=0, 1) [input capture function and output comparison function]

Address: 0x40042A70 (TMCRI0), 0x40042A80 (TMCRI1) after reset:

00H R/W

symbol	7	6	5	4	3	2	1	0
TMCRI	CCLR2	CcLR1	CCLR0	CKEG1	CKEG0	TCK2	TCK1	TCK0

CCLR2	CCLR1	CCLR0	Clear selection for the TMi counter
0	0	0	Purge is prohibited (free-running).
0	0	1	Clear when TMGRAi's input capture/comparison matches.
0	1	0	Clear when input capture/comparison matches for TMGRBi.
0	1	1	Synchronous clearing (and clearing at the same time as counters of other timers Mi) Note 1
1	0	1	Clears when input capture/compare matches for TMGRCi.
1	1	0	Clears when TMGRDi's input capture/compare matches.
Other than the above			Prohibit settings.

CKEG1	CKEG0	Select ^{note 2} for the external clock edge
0	0	Count on the rising edge.
0	1	Count on the falling edge.
1	0	Count on the bilateral edge.
Other than the above		Prohibit settings.

TCK2	TCK1	TCK0	Count the selection of sources
0	0	0	f_{CLK}
0	0	1	$f_{CLK}/2$
0	1	0	$f_{CLK}/4$
0	1	1	$f_{CLK}/8$
1	0	0	$f_{CLK}/32$
1	0	1	Input ^{note 3} for TMCLK
Other than the above			Prohibit settings.

Note 1. Valid when the TMSYNC bit of the TMMR register is "1" (TM0 and TM1 run synchronously).

2. The TCK2~TCK0 bit is "101B" (the input to TMCLK) and the STCLK bit is "1" Valid when (external clock input valid).

3. Valid when the STCLK bit of the TMFCR register is "1" (the external clock input is valid).

Fig. 10-16 Format of timer M control register i (TMCRI) (i=0, 1) [PWM function].

Address: 0x40042A70 (TMCRI0), 0x40042A80 (TMCRI1) after reset:

00H R/W

symbol	7	6	5	4	3	2	1	0
TMCRI	CCLR2	CcLR1	CCLR0	CKEG1	CKEG0	TCK2	TCK1	TCK0

CCLR2	CCLR1	CCLR0	Clear selection for the TMI counter
"001B" must be set (clear the TMI register when matching the TMGRAi register).			

CKEG1	CKEG0	Select Note 1 for the external clock edge
0	0	Count on the rising edge.
0	1	Count on the falling edge.
1	0	Count on the bilateral edge.
Other than the above		Prohibit settings.

TCK2	TCK1	TCK0	Count the selection of sources
0	0	0	f_{CLK}
0	0	1	$f_{CLK}/2$
0	1	0	$f_{CLK}/4$
0	1	1	$f_{CLK}/8$
1	0	0	$f_{CLK}/32$
1	0	1	Input note 2 for TMCLK
Other than the above			Prohibit settings.

Note 1. The TCK2~TCK0 bit is "101B" (the input to TMCLK) and the STCLK bit is "1" Valid when (external clock input valid).

2. Valid when the STCLK bit of the TMFCR register is "1" (the external clock input is valid).

Figure 10-17 Format of timer M control register 0 (TMCRO) [Reset Synchronous PWM Mode].

Address: 0x40042A70 after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TMCRO	CCLR2	CcLR1	CCLR0	CKEG1	CKEG0	TCK2	TCK1	TCK0

CCLR2	CCLR1	CCLR0	Clear selection for the TM0 counter
"001B" must be set (clear the TM0 register when matching the TMGRA0 register).			

CKEG1	CKEG0	Select ^{Note 1} for the external clock edge
0	0	Count on the rising edge.
0	1	Count on the falling edge.
1	0	Count on the bilateral edge.
Other than the above		Prohibit settings.

TCK2	TCK1	TCK0	Count the selection of sources
0	0	0	f_{CLK}
0	0	1	$f_{CLK}/2$
0	1	0	$f_{CLK}/4$
0	1	1	$f_{CLK}/8$
1	0	0	$f_{CLK}/32$
1	0	1	Input ^{note 2} for TMCLK
Other than the above			Prohibit settings.

Note 1. The TCK2~TCK0 bit is "101B" (the input to TMCLK) and the STCLK bit is "1" Valid when (external clock input valid).

2. Valid when the STCLK bit of the TMFCR register is "1" (the external clock input is valid).

Figure 10-18 Format of Timer M Control Register 0 (TMCR0) [Complementary PWM Mode].

Address: 0x40042A70 after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TMCR0	CCLR2	CcLR1	CCLR0	CKEG1	CKEG0	TCK2	TCK1	TCK0

CCLR2	CCLR1	CCLR0	Clear selection for the TM0 counter
Must be set to "000B" (No Purge (Free Run)).			

CKEG1	CKEG0	Select Notes 1 and 2 for the external clock edges
0	0	Count on the rising edge.
0	1	Count on the falling edge.
1	0	Count on the bilateral edge.
Other than the above		Prohibit settings.

TCK2	TCK1	TCK0	Count the selection of sources
0	0	0	f_{CLK}
0	0	1	$f_{CLK}/2$
0	1	0	$f_{CLK}/4$
0	1	1	$f_{CLK}/8$
1	0	0	$f_{CLK}/32$
1	0	1	Input note 3 for TMCLK
Other than the above			Prohibit settings.

- Note 1. The TCK2~TCK0 bit is "101B" (the input to TMCLK) and the STCLK bit is "1" Valid when (external clock input valid).
2. TCK0 to TCK2 bits, CKEG0 bits, CKEG0 bits, and CKEG1 registers must be given to TMCR0 registers and TCKCR1 registers The bits set the same value.
3. Valid when the STCLK bit of the TMFCR register is "1" (the external clock input is valid).

Figure 10-19 Format of timer M control register 0 (TMCR0) [PWM3 mode].

Address: 0x40042A70 after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TMCR0	CCLR2	CcLR1	CCLR0	CKEG1	CKEG0	TCK2	TCK1	TCK0

CCLR2	CCLR1	CCLR0	Clear selection for the TM0 counter
"001B" must be set (clear the TM0 register when matching the TMGRA0 register).			

CKEG1	CKEG0	Selection of external clock edges
Not valid in PWM3 mode.		

TCK2	TCK1	TCK0	Count the selection of sources
0	0	0	f_{CLK}
0	0	1	$f_{CLK}/2$
0	1	0	$f_{CLK}/4$
0	1	1	$f_{CLK}/8$
1	0	0	$f_{CLK}/32$
Other than the above			Prohibit settings.

10.3.12 Timer M I/O control register Ai (TMIORAi) (i =0, 1).

Figure 10-20 Format of the I/O control register Ai (TMIORAi) (i=0, 1) of the timer M [Input Capture Function].

Address: 0x40042A71 (TMIORA0), 0x40042A81H (TMIORA1) After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TMIORAi	0	IOB2	IOB1	IOB0	0	IOA2	IOA1	IA0

IOB2	TMGRB mode selection ^{note 1}
When using the input capture function, you must set "1" (input capture).	

IOB1	IOB0	TMGRB control
0	0	The input is captured to the TMGRBi on the rising edge.
0	1	The input is captured to the TMGRBi on the falling edge.
1	0	The input is captured on both sides to the TMGRBi.
Other than the above		Prohibit settings.

IOA2	TMGRA mode selection ^{note 2}
When using the input capture function, you must set "1" (input capture).	

IOA1	IOA0	TMGRA control
0	0	The input is captured to the THEMGRAi on the rising edge.
0	1	The input is captured to the THEMGRAi on the falling edge.
1	0	The input is captured on both sides to the THEMGRAi.
Other than the above		Prohibit settings.

- Note
1. If you select "1" (buffer register of the TMGRBi register) by the TMBFDi bit of the TMMR register, you must give the TMIORAi register The IOB2 bit and the IOD2 bit of the TMIORCi register are set to the same value.
 2. If you select "1" (the buffer register of the TMGRAi register) by the TMBFCi bit of the TMMR register, you must give the TMIORAi register The IOA2 bit and the IOC2 bit of the TMIORCi register are set to the same value.

Fig. 10-21 Format of the I/O control register Ai (TMIORAi) (i=0, 1) of the timer M[Output comparison function].

Address: 0x40042A71 (TMIORA0), 0x40042A81H (TMIORA1) After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TMIORAi	0	IOB2	IWhether	IOB0	0	IOA2	IOA1	IA0

IOB2	TMGRB mode selection ^{note 1}
When using the output comparison function, you must set "0" (output comparison).	

IOB1	IOB0	TMGRB control
0	0	Disables comparison of matching pin outputs (TMIOBi pins are I/O ports).
0	1	Outputs the "L" level when the TMGRBi comparison matches.
1	0	Outputs the "H" level when the TMGRBi comparison matches.
1	1	Alternate outputs are performed when the TMGRBi comparison matches.

IOA2	TMGRA mode selection ^{note 2}
When using the output comparison function, you must set "0" (output comparison).	

IOA1	IOA0	TMGRA control
0	0	Disables comparison of matching pin outputs (TMIOAi pins are I/O ports).
0	1	Outputs the "L" level when the TMGRAi comparison matches.
1	0	Outputs the "H" level when the TMGRAi comparison matches.
1	1	Alternate outputs are performed when the TMGRAi comparison matches.

- Note
1. If you select "1" (buffer register of the TMGRBi register) by the TMBFDi bit of the TMMR register, you must give the TMIORAi register The IOB2 bit and the IOD2 bit of the TMIORCi register are set to the same value.
 2. If you select "1" (the buffer register of the TMGRAi register) by the TMBFCi bit of the TMMR register, you must give the TMIORAi register The IOA2 bit and the IOC2 bit of the TMIORCi register are set to the same value.

10.3.13 Timer M I/O control register Ci (TMIORCi) (i=0, 1).

Figure 10-22 Format of the timer M I/O control register Ci (TMIORCi) (i=0, 1) [Input Capture Function].

Address: 0x40042A72 (TMIORC0), 0x40042A82 (TMIORC1) after reset:

88H R/W

symbol	7	6	5	4	3	2	1	0
TMIORCi	ID3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0

IOD3	Selection of TMGRD register function
When using the input capture function, a "1" (general purpose register or buffer register) must be set.	

IOD2	Choice of TMGRD mode ^{Note 1}
When using the input capture function, you must set "1" (input capture).	

IOD1	IOD0	TMGRD control
0	0	The input is captured to the TMGRDi on the rising edge.
0	1	The input is captured to the TMGRDi on the falling edge.
1	0	The input is captured on both sides to the TMGRDi.
Other than the above		Prohibit settings.

IOC3	Selection of TMGRC register function
When using the input capture function, a "1" (general purpose register or buffer register) must be set.	

IOC2	TMGRC mode selection ^{note 2}
When using the input capture function, you must set "1" (input capture).	

IOC1	IOC0	TMGRC control
0	0	The input is captured to the TMGRCi on the rising edge.
0	1	The input is captured to the TMGRCi on the falling edge.
1	0	The input is captured on both sides to the TMGRCi.
Other than the above		Prohibit settings.

- Note
1. If you select "1" (buffer register of the TMGRBi register) by the TMBFDi bit of the TMMR register, you must give the TMIORAi register The IOB2 bit and the IOD2 bit of the TMIORCi register are set to the same value.
 2. If you select "1" (the buffer register of the TMGRAi register) by the TMBFCi bit of the TMMR register, you must give the TMIORAi register The IOA2 bit and the IOC2 bit of the TMIORCi register are set to the same value.

Fig. 10-23 Format of timer M I/O control register Ci (TMIORCi) (i=0, 1) [Output comparison function].

Address: 0x40042A72 (TMIORC0), 0x40042A82 (TMIORC1) after reset:						88H R/W		
symbol	7	6	5	4	3	2	1	0
TMIORCi	ID3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0

IOD3	Selection of TMGRD register function
0	TMIOB output register (Refer to "Changes to the Output Pins of the 10.5.2(2)TMGRCi Register and the TMGRDi Register (i=0, 1)")
1	General Purpose registers or buffer registers

IOD2	Choice of TMGRD mode ^{Note 1}
When using the output comparison function, you must set "0" (output comparison).	

IOD1	IOD0	TMGRD control
0	0	Disables comparison of matching pin outputs.
0	1	Outputs the "L" level when the TMGRDi comparison matches.
1	0	Outputs the "H" level when the TMGRDi comparison matches.
1	1	Alternate outputs are performed when the TMGRDi comparison matches.

IOC3	Selection of TMGRC register function
0	TMIOA output registers
1	General Purpose registers or buffer registers

IOC2	TMGRC mode selection ^{note 2}
When using the output comparison function, you must set "0" (output comparison).	

IOC1	IOC0	TMGRC controlled
0	0	Disables comparison of matching pin outputs.
0	1	Outputs the "L" level when the TMGRCi comparison matches.
1	0	Outputs the "H" level when the TMGRCi comparison matches.
1	1	Alternate outputs are performed when the TMGRCi comparison matches.

- Note
1. If you select "1" (buffer register of the TMGRBi register) by the TMBFDi bit of the TMMR register, you must give the TMIORAi register The IOB2 bit and the IOD2 bit of the TMIORCi register are set to the same value.
 2. If you select "1" (the buffer register of the TMGRAi register) by the TMBFCi bit of the TMMR register, you must give the TMIORAi register The IOA2 bit and the IOC2 bit of the TMIORCi register are set to the same value.

10.3.14 Timer M status register 0 (TMSR0).

Figure 10-24 Format of timer M status register 0 (TMSR0) [Input Capture Function].

Address: 0x40042A73 after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TMSR0	0	0	0	AVF	IMFD	IMFC	IMFB	IMFA

OVF	Overflow flag ^{note 1}
[condition for "0"]. After reading write "0" ^{Note 2} . [condition for "1"]. When an overflow occurs in TM0	

IMFD	Enter the capture/compare match flag D ^{Note 5}
[condition for "0"]. After reading write "0" ^{Note 2} . [condition for "1"]. The input edge of the TMIOD0 pin ^{is note 3}	

IMFC	Enter the capture/compare match flag C ^{Note 5}
[condition for "0"]. After reading write "0" ^{Note 2} . [condition for "1"]. The input edge of the TMIOC0 pin ^{is note 3}	

IMFB	Enter the capture/compare match flag B ^{Note 5}
[condition for "0"]. After reading write "0" ^{Note 2} . [condition for "1"]. ^{Note 4} on the input edge of the TMIOB0 pin	

IMFA	Enter the capture/compare match flag A ^{note 5}
[condition for "0"]. After reading write "0" ^{Note 2} . [condition for "1"]. The input edge of the TMIOA0 pin ^{is note 4}	

Note 1 When the count value of timer M0 changes from "FFFFH" to "0000H", the overflow flag changes to "1". In addition, according to the setting of the CCLR0~CCLR2 bit of the TMCR0 register, if the input capture or comparison match occurs during operation, the count value of timer M0 is changed from "FFFFH" becomes "0000H" and the overflow sign becomes "1".

Note 2 Write the result as follows:

- When you write "1", this bit does not change.
- In the case of reading a value of "0", it does not change even if "0" is written to the same bit (in the case of changing from "0" to "1" after reading, it remains "1" even if "0" is written status).
- If the read value is "1", if you write "0" to the same bit, the bit becomes "0".

However, when you want to set the state flag (hereinafter referred to as the "object status flag") of one of the interrupt sources of timer M to "0" if the interrupt is timer

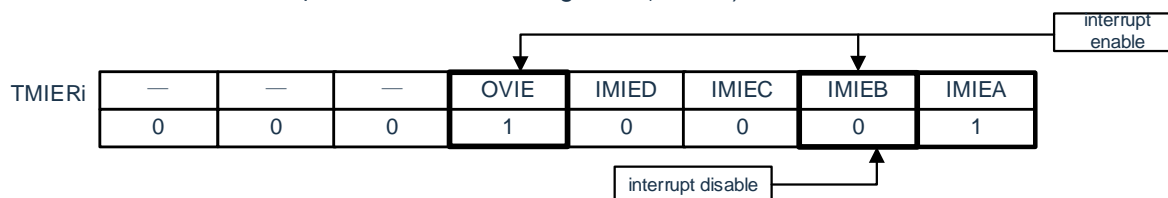
M interrupt enable register i (TMIERi) is set to disable interrupt, and it must be set to "0" in any of the following methods (a)~(c).

(a) The object status flag must be written "0" after setting the timer M interrupt enable register i (TMIERi) to "00H" (disable all interrupts).

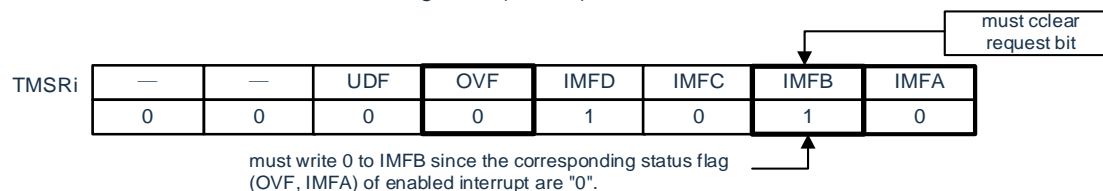
(b) When the timer M interrupt enable register i (TMIERi) has a bit placed "1" (allowed) and the interrupt source status flag allowed by that bit is "0", the object status flag must be written "0".

(e.g.) in the case where IMIEA and OVIE clear IMFB in a state where interrupts are allowed and IMIEB is prohibited

- Timer M interrupt enable the state of register i (TMIERi).



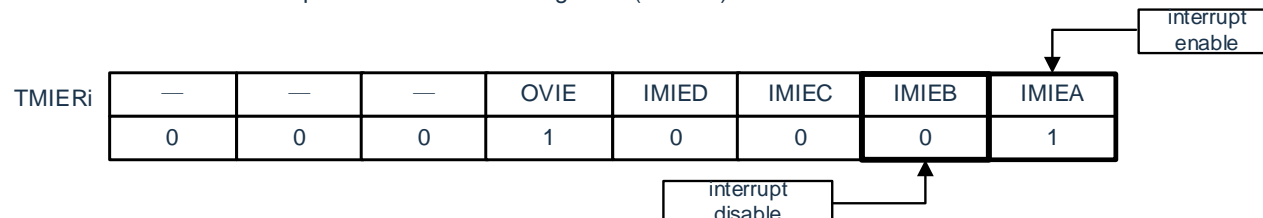
- Status of timer M status register i (TMSRi).



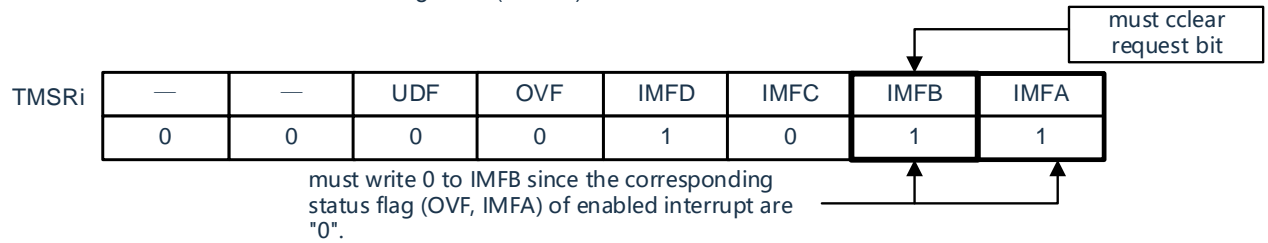
(c) When the timer M interrupt allows a bit of "1" (allowed) in the enable register i (TMIERi) and the bit enable the interrupt source status flag is "1", this state flag and the object status flag must be written "0" at the same time .

(e.g.) when IMIEA clears the IMFB in a state where interrupts are allowed and IMIEB is prohibited

- Timer M interrupt enable the state of register i (TMIERi).



- Status of timer M status register i (TMSRi).



- This is the edge of the IOk1 bit and IOk0 bit (k=C or D) selected for the TMIORC0 register.
Includes the case where the TMBFk0 bit of the TMMR register is "1" (TMGRk0 is the buffer register).
- This is the edge of the IOj1 bit and IOj0 bit (j=A or B) selected for the TMIORA0 register.
- When using DMA, IMFA bits, IMFB bits, IMFC bits, and IMFD bits become after DMA transfers end "1".

Figure 10-25 Format of timer M status register 0 (TMSR0) [functions other than input capture].

Address: 0x40042A73 after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TMSR0	0	0	0	AVF	IMFD	IMFC	IMFB	IMFA

OVF	Overflow flag ^{note 2}
[condition for "0"]. Read "0" ^{note 1} . [condition for "1"]. When an overflow occurs in TM0	

IMFD	Enter capture/compare match flag D ^{Note 4}
[condition for "0"]. Read "0" ^{note 1} . [condition for "1"]. Note ³ when the values of TM0 and TMGRD0 are the same	

IMFC	Enter the capture/compare match flag C ^{Note 4}
[condition for "0"]. Read "0" ^{note 1} . [condition for "1"]. Note ³ when the values of TM0 and TMGRC0 are the same	

IMFB	Enter the capture/compare match flag B ^{Note 4}
[condition for "0"]. Read "0" ^{note 1} . [condition for "1"]. When the values of TM0 and TMGRB0 are the same	

IMFA	Enter capture/compare match flag A ^{note 4}
[condition for "0"]. Read "0" ^{note 1} . [condition for "1"]. When the values of TM0 and TMGRA0 are the same	

Note 1 Write the result as follows:

- When you write "1", this bit does not change.
- In the case of reading a value of "0", it does not change even if "0" is written to the same bit (in the case of changing from "0" to "1" after reading, it remains "1" even if "0" is written status).
- If the read value is "1", if you write "0" to the same bit, the bit becomes "0".

However, when you want to set the state flag (hereinafter referred to as the "object status flag") of one of the interrupt sources of timer M to "0" if the interrupt is timer

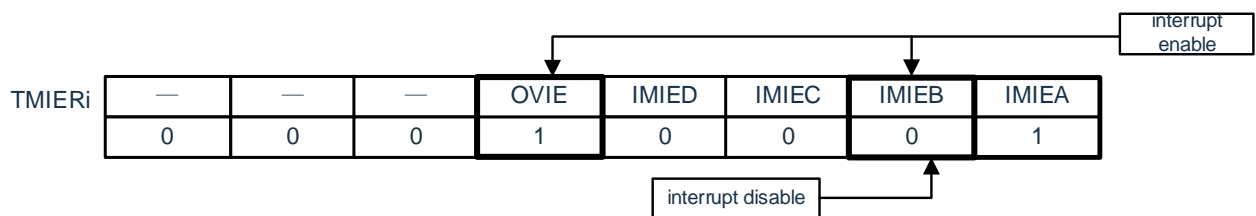
M interrupt enable register i (TMIERi) is set to disable interrupt, and it must be set to "0" in any of the following methods (a)~(c).

(a) The object status flag must be written "0" after setting the timer M interrupt enable register i (TMIERi) to "00H" (disable all interrupts).

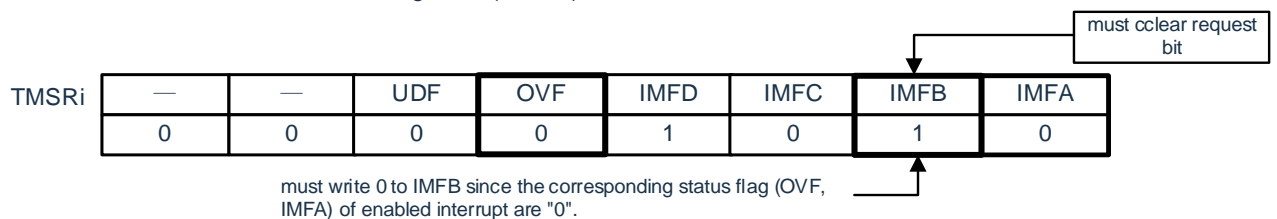
(b) When the timer M interrupt enable register i (TMIERi) has a bit placed "1" (allowed) and the interrupt source status flag allowed by that bit is "0", the object status flag must be written "0".

(e.g.) in the case where IMIEA and OVIE clear IMFB in a state where interrupts are allowed and IMIEB is prohibited

- Timer M interrupt enable the state of register i (TMIERi).



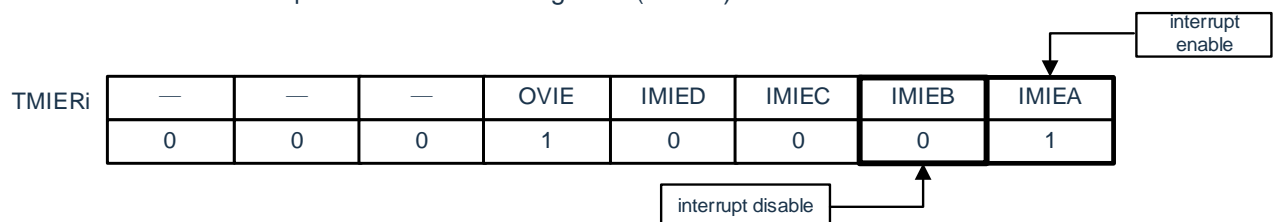
- Status of timer M status register i (TMSRi).



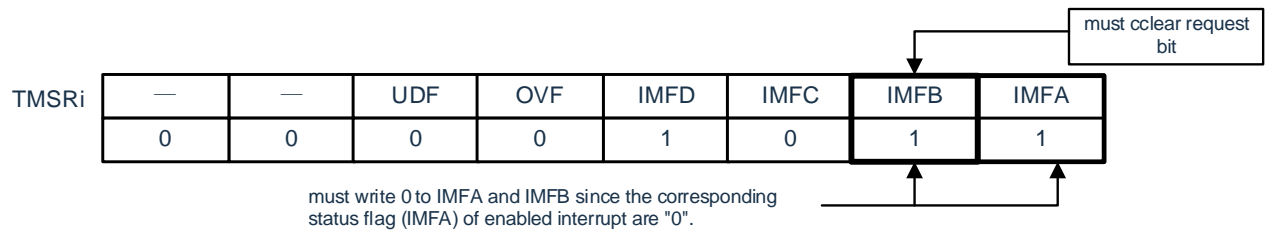
(c) When the timer M interrupt enable a bit of "1" (allowed) in the enable register i (TMIERi) and the bit enable the interrupt source status flag is "1", this state flag and the object status flag must be written "0" at the same time .

(e.g.) when IMIEA clears the IMFB in a state where interrupts are allowed and IMIEB is prohibited

- Timer M interrupt enable the state of register i (TMIERi).



- Status of timer M status register i (TMSRi).



2. When the count value of timer M0 changes from "FFFFH" to "0000H", the overflow flag changes to "1". In addition, according to the setting of the CCLR0~CCLR2 bit of the TMCR0 register, if the input capture or comparison match occurs during operation, the count value of timer M0 is changed from "FFFFH" becomes "0000H" and the overflow sign becomes "1".
3. This includes cases where the TMBFk0 bit (k=C or D) of the TMMR register is "1" (TMGRk0 is the buffer register).
4. When using DMA, IMFA bits, IMFB bits, IMFC bits, and IMFD bits become after DMA transfers end "1".

10.3.15 Timer M status register 1 (TMSR1).

Figure 10-26 Format of timer M status register 1 (TMSR1) [Input Capture Function].

Address: 0x40042A83 reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TMSR1	0	0	UDF	OVF	IMFD	IMFC	IMFB	IMFA

UDF	Underflow flag
Not valid when using the input capture feature.	

OVF	Overflow flag ^{note 1}
[condition for "0"]. After reading write "0" ^{Note 2} . [condition for "1"]. When an overflow occurs in TM1	

IMFD	Enter the capture/compare match flag D ^{Note 5}
[condition for "0"]. After reading write "0" ^{Note 2} . [condition for "1"]. ^{Note 3} on the input edge of the TMIOD1 pin	

IMFC	Enter the capture/compare match flag C ^{Note 5}
[condition for "0"]. After reading write "0" ^{Note 2} . [condition for "1"]. The input edge of the TMIOC1 pin is ^{note 3}	

IMFB	Enter the capture/compare match flag B ^{Note 5}
[condition for "0"]. After reading write "0" ^{Note 2} . [condition for "1"]. The input edge of the TMIOB1 pin is ^{note 4}	

IMFA	Enter the capture/compare match flag A ^{note 5}
[condition for "0"]. After reading write "0" ^{Note 2} . [condition for "1"]. The input edge of the TMIOA1 pin is ^{note 4}	

Note 1 When the count value of timerM1 changes from "FFFFH" to "0000H", the overflow flag changes to "1". In addition, according to the setting of the CCLR0~CCLR2 bit of the TMCr1 register, if the count value of timer M1 is changed from "FFFFH" becomes "0000H" and the overflow sign becomes "1".

Note 2 Write the result as follows:

- When writing "1", this bit does not change.
- In the case of reading a value of "0", it does not change even if "0" is written to the same bit (in the case of changing from "0" to "1" after reading, it remains "1" even if "0" is written status).
- In the case where the read value is "1", if you write "0" to the same bit, this bit becomes "0".

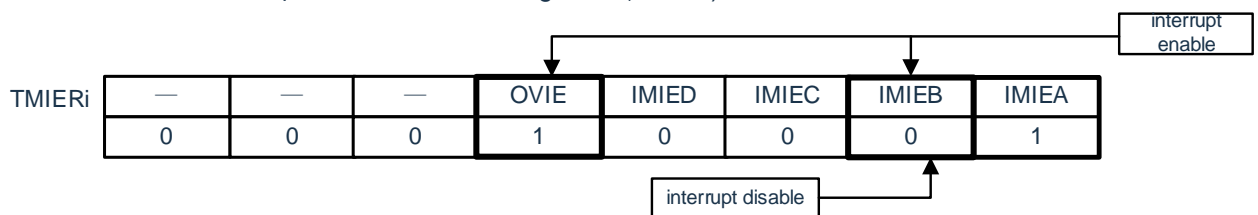
However, when you want to set the status flag of one of the interrupt sources of timer M (hereinafter referred to as the "object status flag") to "0", if the interrupt is allowed by timer M interrupt register i (TMIERi) to disable interrupts, you must use any of the following methods (a) ~ (c) to set "0".

(a) The object status flag must be written "0" after setting the timer M interrupt enable register i (TMIERi) to "00H" (disable all interrupts).

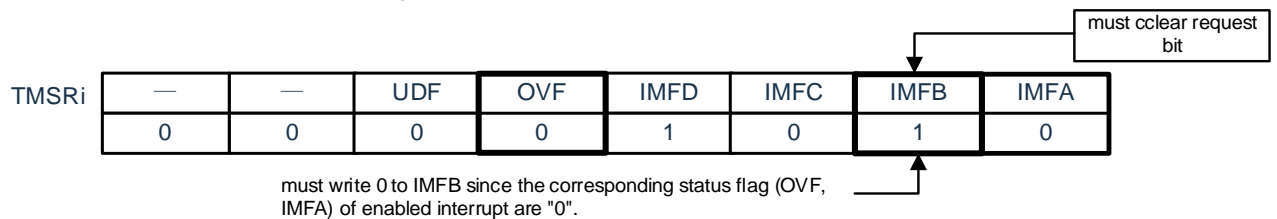
(b) When the timer M interrupt enable register i (TMIERi) has a bit placed "1" (allowed) and the interrupt source status flag allowed by that bit is "0", the object status flag must be written "0".

(e.g.) in the case where IMIEA and OVIE clear IMFB in a state where interrupts are allowed and IMIEB is prohibited

- Timer M interrupt enable the state of register i (TMIERi).



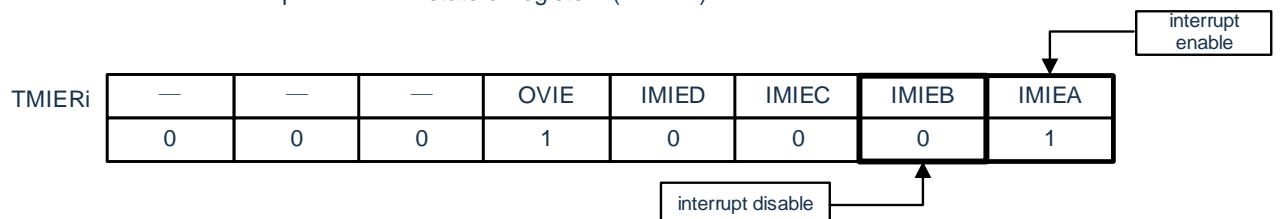
- Status of timer M status register i (TMSRi).



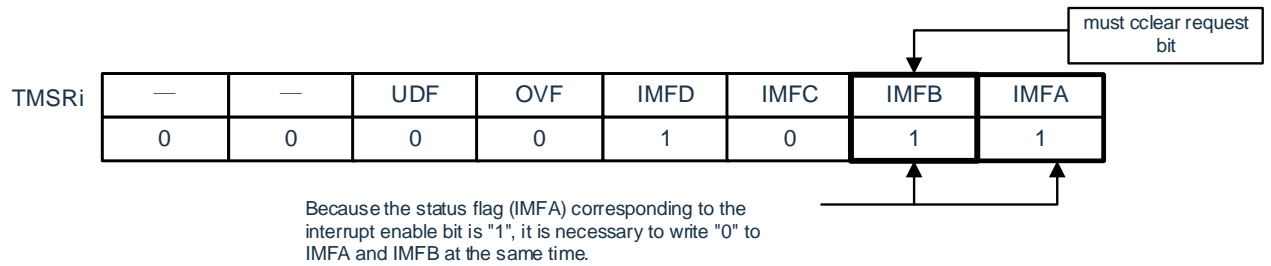
(c) When the timer M interrupt enable a bit of "1" (allowed) in the enable register i (TMIERi) and the bit enable the interrupt source status flag is "1", this state flag and the object status flag must be written "0" at the same time .

(e.g.) when IMIEA clears the IMFB in a state where interrupts are allowed and IMIEB is prohibited

- Timer M interrupt enable the state of register i (TMIERi).



- Status of timer M status register i (TMSRi).



- This is the edge of the IOk1 bit and IOk0 bit (k=C or D) selected for the TMIORC1 register.
This includes cases where the TMBFk1 bit of the TMMR register is "1" (TMGRk1 is the buffer register).
- This is the edge of the IOj1 bit and IOj0 bit (j=A or B) selected for the TMIORA1 register.
- When using DMA, IMFA bits, IMFB bits, IMFC bits, and IMFD bits become after DMA transfers end "1".

Figure 10-27 Format of timer M status register 1 (TMSR1) [functions other than input capture].

Address: 0x40042A83 reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TMSR1	0	0	UDF	OVF	IMFD	IMFC	IMFB	IMFA

UDF	Underflow flag
Case of complementary PWM mode [condition for "0"]. Read "0" note 1. [condition for "1"]. It is not valid in non-complementary PWM mode when TM1 underflow occurs.	

OVF	Overflow flag note 2
[condition for "0"]. Read "0" note 1. [condition for "1"]. When an overflow occurs in TM1	

IMFD	Enter capture/compare match flag D Note 4
[condition for "0"]. Read "0" note 1. [condition for "1"]. Note 3 when the values of TM1 and TMGRD1 are the same	

IMFC	Enter the capture/compare match flag C Note 4
[condition for "0"]. Read "0" note 1. [condition for "1"]. Note 3 when the values of TM1 and TMGRC1 are the same	

IMFB	Enter the capture/compare match flag B Note 4
[condition for "0"]. Read "0" note 1. [condition for "1"]. When the values of TM1 and TMGRB1 are the same	

IMFA	Enter capture/compare match flag A note 4
[condition for "0"]. Read "0" note 1. [condition for "1"]. When the values of TM1 and TMGRA1 are the same	

Note 1 Write the result as follows:

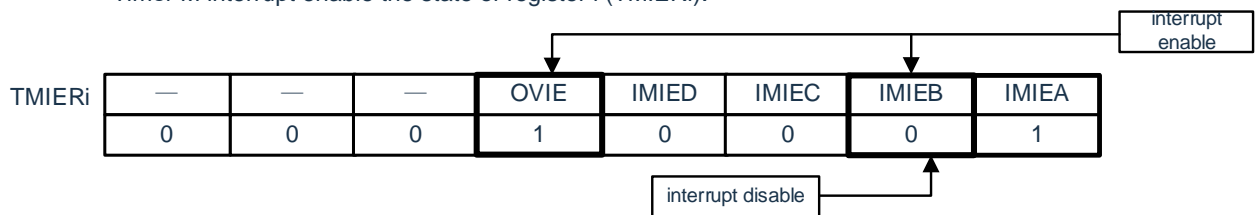
- When writing "1", this bit does not change.
 - In the case of reading a value of "0", it does not change even if "0" is written to the same bit (in the case of changing from "0" to "1" after reading, it remains "1" even if "0" is written status).
 - In the case where the read value is "1", if you write "0" to the same bit, this bit becomes "0".
- However, when you want to set the status flag of one of the interrupt sources of timer M (hereinafter referred to as the "object status flag") to "0", if the interrupt is allowed by timer M interrupt register i (TMIERi) to disable interrupts, you must use any of the following methods (a) ~ (c) to set "0".

(a) The object status flag must be written "0" after setting the timer M interrupt enable register i (TMIERi) to "00H" (disable all interrupts).

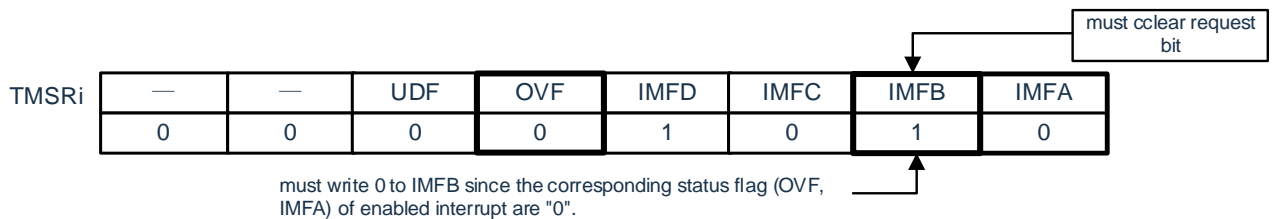
(b) When the timer M interrupt enable register i (TMIERi) has a bit placed "1" (allowed) and the interrupt source status flag allowed by that bit is "0", the object status flag must be written "0".

(e.g.) in the case where IMIEA and OVIE clear IMFB in a state where interrupts are allowed and IMIEB is prohibited

- Timer M interrupt enable the state of register i (TMIERi).



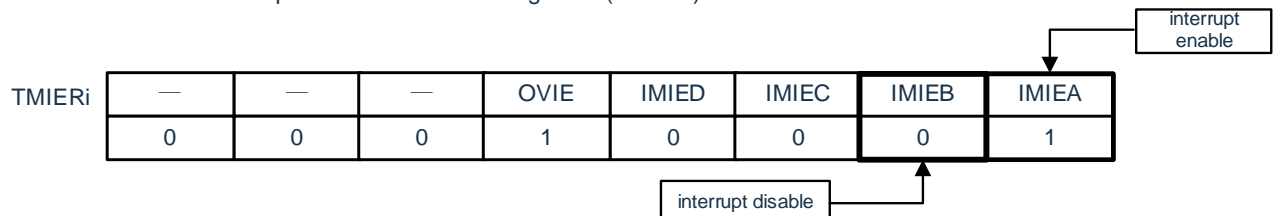
- Status of timer M status register i (TMSRi).



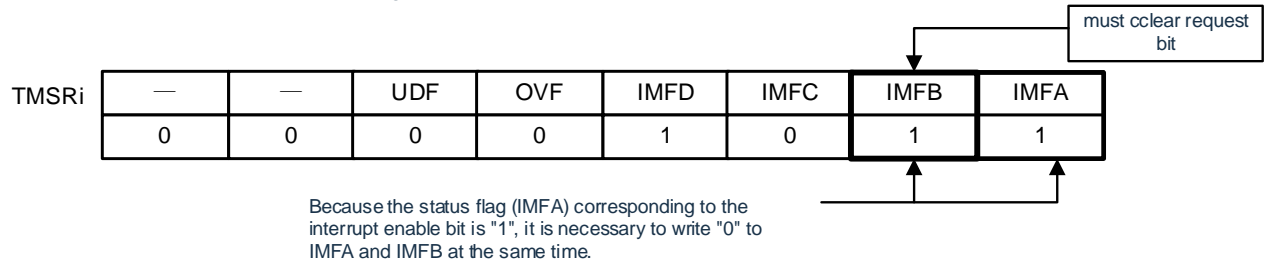
(c) When the timer M interrupt enable a bit of "1" (allowed) in the enable register i (TMIERi) and the bit enable the interrupt source status flag is "1", this state flag and the object status flag must be written "0" at the same time .

(e.g.) when IMIEA clears the IMFB in a state where interrupts are allowed and IMIEB is prohibited

- Timer M interrupt enable the state of register i (TMIERi).



- Status of timer M status register i (TMSRi).



2. When the count value of timerM1 changes from "FFFFH" to "0000H", the overflow flag changes to "1". In addition, according to the setting of the CCLR0~CCLR2 bit of the TMCR1 register, if the count value of timer M1 is changed from " FFFFH" becomes "0000H" and the overflow sign becomes "1".
3. This includes cases where the TMBFk1 bit (k=C or D) of the TMMR register is "1" (TMGRk1 is the buffer register).
4. When using DMA, IMFA bits, IMFB bits, IMFC bits, and IMFD bits become after DMA transfers end "1".

10.3.16 Timer M interrupt enable register i (TMIERi) (i= 0, 1).

Figure 10-28 the format of Timer M interrupt enable register i (TMIERi) (i=0, 1).

Address: 0x40042A74 (TMIER0), 0x40042A84 (TMIER1) after reset:

00HR/W

symbol	7	6	5	4	3	21	0
TMIERi	0	0	0	OLife	InIED	IMIEC	InIEB IMIE EA

OVIE	Allowed overflow/underflow interrupts
0	Interrupts due to OVF bits and UDF bits (OVI) are prohibited.
1	Interrupts due to OVF bits and UDF bits (OVI) are allowed.

IMIED	Input capture/comparison matching interrupts allow D
0	Interrupts (IMID) due to IMFD bits are prohibited.
1	Interrupts (IMIDs) due to IMFD bits are allowed.

IMIEC	Input capture/compare matching interrupts allow C
0	Interrupts due to IMFC bits (IMIC) are prohibited.
1	Interrupts due to IMFC bits (IMCs) are allowed.

IMIEB	Input capture/compare matching interrupts allow B
0	Interrupts due to IMFB bits (IMIB) are prohibited.
1	Interrupts due to IMFB bits (IMIB) are allowed.

IMIEA	Input capture/compare matching interrupts allow A
0	Interrupts due to IMFA bits (IMIA) are prohibited.
1	Interrupts due to IMFA bits (IMIA) are allowed.

10.3.17 The timer MPWM function outputs level control register i (TMPOCRi) (i=0, 1).

The setting of the TMPOCRi register is valid only when the PWM function is used, otherwise the setting of the TMPOCRi register is invalid.

Figure 10-29 Format of the output level control register i (TMPOCRi) (i=0, 1) of the MPWM function of the timer [PWM]. Function].

Address: 0x40042A75 (TMPOCR0), 0x40042A85 (TMPOCR1) after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TMPOCRi	0	0	0	0	0	POLD	POLC	POLB

BOLT	The output level control of the PWM function D
0	The TMIODi output level is active at the "L" level.
1	The TMIODi output level is active at "H" level.

SHELF	The output level control of the PWM function C
0	The TMIOCI output level is active at "L" level.
1	The TMIOCI output level is active at "H" level.

POLB	The output level control of the PWM function B
0	The TMIOBi output level is active at "L" level.
1	The TMIOBi output level is active at "H" level.

10.3.18 Timer M counter i(TM_i) (i=0, 1).

[Timer mode].

TM_i registers must be accessed in 16 bits, not 8 bits.

[Reset Synchronous PWM Mode and PWM3 Mode].

The TM0 register must be accessed in 16 bits, not 8 bits. In reset synchronous PWM mode and PWM3 mode, TM1 is not used Register.

[Complementary PWM mode (TM0)].

The TM0 register must be accessed in 16 bits, not 8 bits.

[Complementary PWM mode (TM1)].

The TM1 register must be accessed in 16 bits, not 8 bits.

Figure 10-30 Format of timer M counter i(TM_i) (i=0, 1) [timer mode].

Address: 0x40042A76 (TM0), 0x40042A86 (TM1) after reset:																0000H R/W	
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
TMi																	

—	function	Configure range
bit15~0	Increment the count on the count source. If an overflow occurs, the OVF bit of the TMSRi register becomes "1".	0000H~FFFFH

Figure 10-31 Format of the timer M counter i(TM_i) (i=0, 1) [Reset Synchronous PWM mode and PWM3 mode].

Address: 0x40042A76 (TM0), 0x40042A86 (TM1) after reset:													0000H R/W			
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMi																
—		function												Configure		
bit15~0		Increment the count on the count source. If an overflow occurs, the OVF bit of the TMSR0 register becomes "1".												0000H~FFFFH		

Figure 10-32 Format of the timer M counter i(TMi) (i=0, 1) [Complementary PWM Mode (TM0)]

Address: 0x40042A76 (TM0), 0x40042A86 (TM1) after reset: 0000H R/W																
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMi																
—		function												Configure range		
bit15~0		The dead time must be set. Increment or decrement counts on the count source. If an overflow occurs, the OVF bit of the TMSR0 register becomes "1".												0001H~FFFFH		

Figure 10-33 Format of the timer M counter i(TMi) (i=0, 1) [Complementary PWM Mode (TM1)]

Address: 0x40042A76 (TM0), 0x40042A86 (TM1) after reset: 0000H R/W																
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMi																
—		function												Configure range		
bit15~0		Must be set to "0000H". Increment or decrement counts on the count source. If an overflow occurs, the UDF bit of the TMSR1 register becomes "1".												0000H~FFFFH		

[Input capture function].

The TMGRAi~TMGRDi registers must be accessed in 16 bits, not 8 bits.

When using the input capture function, the following registers are invalid:

TMOER1, TMOER2, TMOCR, TMPOCR0, TMPOCR1

When no digital filter is used (the DFj bit of the TMDFi register is "0"), the pulse width of the capture signal input to the TMIOj pin must be at least 3 timer M's operating clock (f_{CLK}) cycle.

[Output comparison function].

The TMGRAi~TMGRDi registers must be accessed in 16 bits, not 8 bits.

When using the output comparison function, the following registers are invalid:

TMDF0, TMDF1, TMPOCR0, TMPOCR1

[PWM function].

The TMGRAi~TMGRDi registers must be accessed in 16 bits, not 8 bits.

When using the PWM function, the following registers are invalid:

TMDF0, TMDF1, TMIORA0, TMIORC0, TMIORA1, TMIORC1

[Reset Synchronous PWM Mode].

The TMGRAi~TMGRDi registers must be accessed in 16 bits, not 8 bits.

In reset synchronous PWM mode, the following registers are invalid:

TMPMR, TMOCR^{note}, TMDF0, TMDF1, TMIORA0, TMIORC0, TMPOCR0, TMIORA1, TMIORC1, TMPOCR1

Note As the initial output setting of TMIOC0 in reset synchronous PWM mode and complementary PWM mode, only the TOC0 bit of the TMOCR register is valid.

[Complementary PWM mode].

The TMGRAi~TMGRDi registers must be accessed in 16 bits, not 8 bits.

In complementary PWM mode, the TMGRC0 register is not used.

In complementary PWM mode, the following registers are invalid:

TMPMR, TMOCR^注, TMDF0, TMDF1, TMIORA0, TMIORC0, TMPOCR0, TMIORA1, TMIORC1, TMPOCR1

Note As the initial output setting of TMIOC0 in reset synchronous PWM mode and complementary PWM mode, only the TOC0 bit of the TMOCR register is valid.

Since you cannot write the TMGRB0, TMGRA1, TMGRB1 registers (prohibited) directly after starting counting, you must add TMGRD0, TMGRC1. The TMGRD1 register is used as a buffer register.

However, when writing the TMGRD0, TMGRC1, TMGRD1 registers, place the TMBFD0 bits, TMBFC1 bits, and TMBFD1 positions Write these registers after "0" (General Purpose registers). Thereafter, the TMBFD0 bit, TMBFC1 bit, and TMBFD1 position "1" (slow charge register) can be changed.

[PWM3 mode].

The TMGRAi~TMGRDi registers must be accessed in 16 bits, not 8 bits.

In PWM3 mode, the following registers are invalid:

TMPMR, TMDf0, TMDf1, TMIOA0, TMIORC0, TMPOCR0, TMIOA1, TMIORC1, TMPOCR1

In PWM3 mode, the TMGRC0, TMGRC1, TMGRD0, TMGRD1 registers are not used. However, when using these registers as buffer registers, place the TMBFC0 bits, TMBFC1 bits, TMBFD0 bits, and TMBFD1 positions "0" (General purpose register) after the TMGRC0, TMGRC1, TMGRD0, TMGRD1 registers write values. Thereafter, TMBFC0 bits, TMBFC 1 bits, TMBFD0 bits, and TMBFD1 positions "1" (buffer registers) can be converted.

Fig. 10-34 Format of Timer M General Purpose registers Ai, Bi, Ci, Di
(TMGRAi, TMGRBi, TMGRCi, TMGRDi) (i=0, 1) [Input Capture Function].

Address: 0x40042A78 (TMGRA0), 0x40042A7A (TMGRB0), After reset: FFFFH R/W

0x40042B58 (TMGRC0) , 0x40042B5A (TMGRD0) ,

0x40042A88 (TMGRA1) , 0x40042A8A (TMGRB1) ,

0x40042B5 C(TMGRCl), 0x40042B5E(TMGRDl)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMGRAi																
TMGRBi																
TMGRCi																
TMGRDi																

—	function
bit15~0	Refer to "Table 10-3 TMGRji Register Function When Using the Input Capture

Table 10-3 Uses the TMGRji register function when using the input capture function

register	Set up	Register function	Input pins for input capture
TMGRAi	—	A general purpose register that reads the value of the TMi register at the time of input capture.	TMIOAi
TMGRBi			TMIOBi
TMGRCi	TMBFCi=0	A general purpose register that reads the value of the TMi register at the time of input capture.	TMIOCi
TMGRDi	TMBFDi=0		TMIODi
TMGRCi	TMBFCi=1	General purpose register that reads the value of the TMi register at the time of input capture (see "10.4.2." Buffer run").	TMIOAi
TMGRDi	TMBFDi=1		TMIOBi

Remark i=0, 1, j=A, B, C, D

TMBFCi, TMBFDi: Bits of the TMMR register

Fig. 10-35 Timer M General Purpose registers Ai, Bi, Ci, Di
Format of (TMGRAi, TMGRBi, TMGRCi, TMGRDi) (i=0, 1) [Output comparison function].

Address: 0x40042A78 (TMGRA0), 0x40042A7A (TMGRB0), After Reset: FFFFH R/W
0x40042B58 (TMGRC0) , 0x40042B5A (TMGRD0) ,
0x40042A88 (TMGRA1) , 0x40042A8A (TMGRB1) ,
0x40042B5C(TMGRCl), 0x40042B5E(TMGRD1)

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

TMGRAi

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

TMGRBi

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

TMGRCi

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

TMGRDi

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

—	function
bit15~0	Refer to "Table 10-4 TMGRji Register Function When Using the Output Comparison Function".

Table 10-4 The TMGRji register function when using the output comparison function

register	Set up		Register function	Output pin for output comparison
	TMBFki	IOj3		
TMGRAi	—	—	A general purpose register that must write the comparison value.	TMIOAi
TMGRBi				TMIOBi
TMGRCi	0	1	A general purpose register that must write the comparison value.	TMIOCi
TMGRDi				TMIODi
TMGRCi	1	1	Buffer registers, which must write down the next comparison value (See "10.4.2 Buffer Run").	TMIOAi
TMGRDi				TMIOBi
TMGRCi	0	0	TMIOAi output control	TMIOAi
TMGRDi			TMIOBi output control	TMIOBi

Note if you put the TCK2~TCK0 position of the TMCRI register "000B" (f_{CLK} , f_{HOCO} And set the comparison value to "0000H" to generate a request signal to the DMA and ELC only after starting the count. If the comparison value is greater than or equal to "0001H", a request signal is generated at each time the comparison matches.

Remark i=0, 1,j=A, B, C, D,k=C, D
TMBFki: Bits of TMMR registers, IOj3: Bits of TMIORCi registers

Fig. 10-36 Timer M General Purpose registers Ai, Bi, Ci, Di
(TMGRAi, TMGRBi, TMGRCi, TMGRDi) (i=0, 1) format [PWM]. Function].

Address: 0x40042A78 (TMGRA0), 0x40042A7A (TMGRB0), After reset: FFFFH R/W
0x40042B58 (TMGRC0), 0x40042B5A (TMGRD0),
0x40042A88 (TMGRA1), 0x40042A8A (TMGRB1),
0x40042B5C (TMGRC1), 0x40042B5E (TMGRD1)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMGRAi																
TMGRBi																
TMGRCi																
TMGRDi																

—	function
bit15~0	Refer to "Table 10-5 TMGRji Register Function When Using the PWM Function".

Table 10-5 TMGRji register function when using the PWM function

register	Set up	Register function	PWM output pin
TMGRAi	—	General purpose registers, the PWM period must be set.	—
TMGRBi	—	A general purpose register where the change point of the PWM output must be set.	TMIOBi
TMGRCi	TMBFCi=0	A general purpose register where the change point of the PWM output must be set.	TMIOCi
TMGRDi	TMBFDi=0		TMIODi
TMGRCi	TMBFCi=1	Buffer register, the next PWM period must be set (refer to "10.4.2 Buffer Run").	—
TMGRDi	TMBFDi=1	Buffer register, the change point of the next PWM output must be set (see "10.4.2 Buffer Run").	TMIOBi

Note if you put the TCK2~TCK0 position of the TMCri register "000B" (f_{CLK} , f_{HOCO} And set the comparison value to "0000H" to generate a request signal to the DMA and ELC only after starting the count. If the comparison value is greater than or equal to "0001H", a request signal is generated at each time the comparison matches.

Remark i=0, 1, j=A, B, C, D
TMBFCi, TMBFDi: Bits of the TMMR register

Fig. 10-37 Timer M General Purpose registers Ai, Bi, Ci, Di
Format of (TMGRAi, TMGRBi, TMGRCi, TMGRDi) (i=0, 1) [Reset synchronous PWM mode].

Address: 0x40042A78 (TMGRA0), 0x40042A7A (TMGRB0), After reset: FFFFH R/W

0x40042B58 (TMGRC0), 0x40042B5A (TMGRD0),

0x40042A88 (TMGRA1), 0x40042A8A (TMGRB1),

0x40042B5C (TMGRC1), 0x40042B5E (TMGRD1)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMGRAi																
TMGRBi																
TMGRCi																
TMGRDi																

—	function
bit15~0	Refer to "Table 10-6 Reset TMGRji Register Function in Synchronous PWM Mode".

Table 10-6 Reset TMGRji register function in synchronous PWM mode

register	Set up	Register function	PWM output pin
TMGRA0	—	General Purpose registers, the PWM period must be set.	(TMIOC0 is per PWM.) Period for inverting output)
TMGRB0	—	A general purpose register where the change point of the PWM1 output must be set.	TMIOB0T MIOD0
TMGRC0	TMBFC0=0	(Not used in reset synchronous PWM mode).	—
TMGRD0	TMBFD0=0		
TMGRA1	—	A general purpose register where the change point of the PWM2 output must be set.	TMIOA1T MIOC1
TMGRB1	—	A general purpose register where the change point of the PWM3 output must be set.	TMIOB1T MIOD1
TMGRC1	TMBFC1=0	(Not used in reset synchronous PWM mode).	—
TMGRD1	TMBFD1=0		
TMGRC0	TMBFC0=1	Buffer register, the next PWM period must be set (refer to "10.4.2 Buffer Run").	(TMIOC0 is per PWM.) Period for inverting output)
TMGRD0	TMBFD0=1	Buffer register, which must be set to the next PWM1 output change point (reference "10.4.2 Buffer Run").	TMIOB0T MIOD0
TMGRC1	TMBFC1=1	Buffer register, which must be set to the next PWM2 output change point (reference "10.4.2 Buffer Run").	TMIOA1T MIOC1
TMGRD1	TMBFD1=1	Buffer registers, which must be set to the next PWM3 output change point (reference "10.4.2 Buffer Run").	TMIOB1T MIOD1

Note if you put the TCK2~TCK0 position of the TCK0 register "000B" (f_{CLK} , f_{HOCO} And set the comparison value to "0000H" to generate a request signal to the DMA and ELC only after starting the count. If the comparison value is greater than or equal to "0001H", a request signal is generated at each time the comparison matches.

Remark i=0, 1, j=A, B, C, D

TMBFC0, TMBFD0, TMBFC1, TMBFD1: Bits of the TMMR register

Fig. 10-38 Timer M General Purpose registers Ai, Bi, Ci, Di
Format of (TMGRAi, TMGRBi, TMGRCi, TMGRDi) (i=0, 1) [Complementary PWM mode].

Address: 0x40042A78 (TMGRA0), 0x40042A7A (TMGRB0), After reset: FFFFH R/W
0x40042B58 (TMGRC0) , 0x40042B5A (TMGRD0) ,
0x40042A88 (TMGRA1) , 0x40042A8A (TMGRB1) ,
0x40042B5C(TMGR C1), 0x40042B5E(TMGRD1)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMGRAi																
TMGRBi																
TMGRCi																
TMGRDi																

—	function
bit15~0	Refer to "TABLE 10-7 COMPLEMENTARY PWM MODE IN THE TBGRJI REGISTER

Table 10-7 Complementary TMGRji register functions in PWM mode

register	Set up	Register function	PWM output pin
TMGRA0	—	A general purpose register that must set the PWM period at the time of initial setup. Setting range: The setting value (initial value of the count) of the TM0 register \leq The setting value of TMGRA0 \leq the setting value of the FFFFH—TM0 register When the TSTART0 bit and TSTART1 bit of the TMSTR register are "1" This register cannot be written when (Start Count).	(TMIOC0 outputs inverting phase every half cycle).
TMGRB0	—	A general-purpose register where the change point of the PWM1 output must be set at the initial setup. Setting range: The setting value (initial value of the count) of the TM0 register \leq The setting value for TMGRB0 \leq Setting value of TMGRA0 – The setting value of the TM0 register When the TSTART0 bit and TSTART1 bit of the TMSTR register are "1" This register cannot be written when (Start Count).	TMIOB0T MIOD0
TMGRA1	—	A general purpose register that must set the change point of the PWM2 output at the initial setup. Setting range: The setting value (initial value of the count) of the TM0 register \leq The setting value of TMGRA1 \leq Setting value of TMGRA0 – The setting value of the TM0 register When the TSTART0 bit and TSTART1 bit of the TMSTR register are "1" This register cannot be written when (Start Count).	TMIOA1T MIOC1
TMGRB1	—	A general purpose register where the change point of the PWM3 output must be set at the initial setup. Setting range: The setting value (initial value of the count) of the TM0 register \leq The setting value for TMGRB1 \leq Setting value of TMGRA0 – The setting value of the TM0 register When the TSTART0 bit and TSTART1 bit of the TMSTR register are "1" This register cannot be written when (Start Count).	TMIOB1T MIOD1
TMGRC0	—	(Not used in complementary PWM mode).	—
TMGRD0	TMBFD0=1	Buffer register, the change point of the next PWM1 output must be set (see "10.4.2 Buffer Run"). Setting range: The setting value (initial value of the count) of the TM0 register \leq The setting value for TMGRD0 \leq Setting value of TMGRA0 – The setting value of the TM0 register The same value of the TMGRB0 register must be set at the initial setup.	TMIOB0T MIOD0
TMGRC1	TMBFC1=1	Buffer register, which must be set to change point for the next PWM2 output (see "Buffer Run 10.4.2"). Setting range: The setting value (initial value of the count) of the TM0 register \leq The setting value for TMGRC1 \leq Setting value of TMGRA0 – The setting value of the TM0 register The same value of the TMGRA1 register must be set at the initial setup.	TMIOA1T MIOC1
TMGRD1	TMBFD1=1	Buffer register, which must be set to change point for the next PWM3 output (see "Buffer Run 10.4.2"). Setting range: The setting value (initial value of the count) of the TM0 register \leq The setting value for TMGRD1 \leq Setting value of TMGRA0 – The setting value of the TM0 register The same value of the TMGRB1 register must be set at the initial setup.	TMIOB1T MIOD1

Note if you put the TCK2~TCK0 position of the TMCRI register "000B" (fCLK, fHOCO And set the comparison value to "0000H" to generate a request signal to the DMA and ELC only after starting the count. If the comparison value is greater than or equal to "0001H", a request signal is generated at each time the comparison matches.

Remark i=0, 1, j=A, B, C, D
TMBFD0, TMBFC1, TMBFD1: Bits of the TMMR register

Fig. 10-39 Timer M General Purpose registers Ai, Bi, Ci, Di
Format (TMGRAi, TMGRBi, TMGRCi, TMGRDi) (i=0, 1) format [PWM3 Mode].

Address: 0x40042A78 (TMGRA0), 0x40042A7A (TMGRB0), After reset: FFFFH R/W
0x40042B58 (TMGRC0) , 0x40042B5A (TMGRD0) ,
0x40042A88 (TMGRA1) , 0x40042A8A (TMGRB1) ,
0x40042B5C(TMGR C1), 0x40042B5E(TMGRD1)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMGRAi																
TMGRBi																
TMGRCi																
TMGRDi																

—	function
bit15~0	Refer to "TMGRji Register Functions in Table 10-8PWM3 Mode".

Table 10-8 TMGR*ji* register functions in PWM3 mode

register	Set up	Register function	PWM output pin
TMGRA0	—	General Purpose registers, the PWM period must be set. Setting range: The setting value of the TMGRA1 register \leq the setting value of TMGRA0	TMIOA0
TMGRA1		A general purpose register where the change point of the PWM output must be set (the timing of becoming the effective level). Setting range: The setting value of TMGRA1 \leq the setting value of the TMGRA0 register	
TMGRB0		A general purpose register where the point of change of the PWM output must be set (the timing of returning to the initial output level). Setting range: The setting value of the TMGRB1 register \leq the setting value of TMGRB0 \leq The setting value of the TMGRA0 register	TMIOB0
TMGRB1		A general purpose register where the change point of the PWM output must be set (the timing of becoming the effective level). Setting range: The setting value of TMGRB1 \leq the setting value of the TMGRB0 register	
TMGRC0	TMBFC0=0	(Not used in PWM3 mode).	—
TMGRC1	TMBFC1=0		
TMGRD0	TMBFD0=0		
TMGRD1	TMBFD1=0		
TMGRC0	TMBFC0=1	Buffer register, the next PWM period must be set (refer to "10.4.2 Buffer Run"). Setting range: The setting value of the TMGRC1 register \leq the setting value of TMGRC0	TMIOA0
TMGRC1	TMBFC1=1	Buffer register, the change point of the next PWM output must be set (see "10.4.2 Buffer Run"). Setting range: The setting value of TMGRC1 \leq the setting value of the TMGRC0 register	
TMGRD0	TMBFD0=1	Buffer register, the change point of the next PWM output must be set (see "10.4.2 Buffer Run"). Setting range: TMGRD1 register \leq TMGRD0 set value \leq The setting value of the TMGRC0 register	TMIOB0
TMGRD1	TMBFD1=1	Buffer register, the change point of the next PWM output must be set (see "10.4.2 Buffer Run"). Setting range: The setting value of TMGRD1 \leq the setting value of the TMGRD0 register	

Note if you put the TCK2~TCK0 position of the TCK0 register "000B" (fCLK, fHOCO And set the comparison value to "0000H" to generate a request signal to the DMA and EVENTC only after starting the count. If the comparison value is greater than or equal to "0001H", a request signal is generated at each time the comparison matches.

Remark i=0, 1, j=A, B, C, D
TMBFC0, TMBFD0, TMBFC1, TMBFD1: Bits of the TMMR register

10.3.20 Port mode registers (PMxx, PMCxx).

This is the register that sets the input/output of the port or the analog input.

When using the multiplex port of the timer output pin (Pxx/TMIOD1, Pxx/TMIOC1, etc.) as the output of the timer, the corresponding port mode register (PMxx, P) must be used for each port MCxx) bit and port register (Pxx) position "0".

(Example) P10/TMIOD1 as a timer output

Place the PM10 andPMC10 positions of port mode register 1 "0".

Place port register 1 at P10 position "0".

When using the multiplexing port of the timer input pin (P10/TMIOD1, P11/TMIOC1, etc.) as the input to the timer, the position of the port mode register (PMxx) corresponding to each port must be "1" and each port The corresponding port mode register (PMCxx) is at position "0". At this point, the bit of the port register (Pxx) can be "0" or "1".

(Example) P10/TMIOD1 as a timer input

Place the PM10 position of Port Mode Register 1 "1".

Place the PMC10 position of port mode register 1 "0".

Place the P10 position of port register 1 at "0" or "1".

Set the PM1 and PMC1 registers via the 8-bit memory operation instructions. After generating a reset signal, the value of the register changes to "FFH".

Figure 10-40 Format of Port Mode Register 1 (PM1, PMC1).

Address: 0x40040321 after reset: FFH R/W

Symbol 7 6 5 4 3 2 1 0

PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10
-----	------	------	------	------	------	------	------	------

PMmn	The input/output mode of the Pmn pin is selected (m=1, n=0 to 7).
0	Output mode (Output Buffer ON).
1	Input mode (output buffer OFF).

Address: 0x40040061 After reset: FFH R/W

Symbol 7 6 5 4 3 2 1 0

PMC1	PMC17	PMC16	PMC15	PMC14	PMC13	PMC12	PMC11	PMC10
------	-------	-------	-------	-------	-------	-------	-------	-------

PMCMn	The input/output mode of the Pmn pin is selected (m=1, n=0~7).
0	Digital inputs/outputs (multiplexing functions other than analog inputs).
1	Analog input

10.4 Common things about multiple patterns

10.4.1 Counting sources

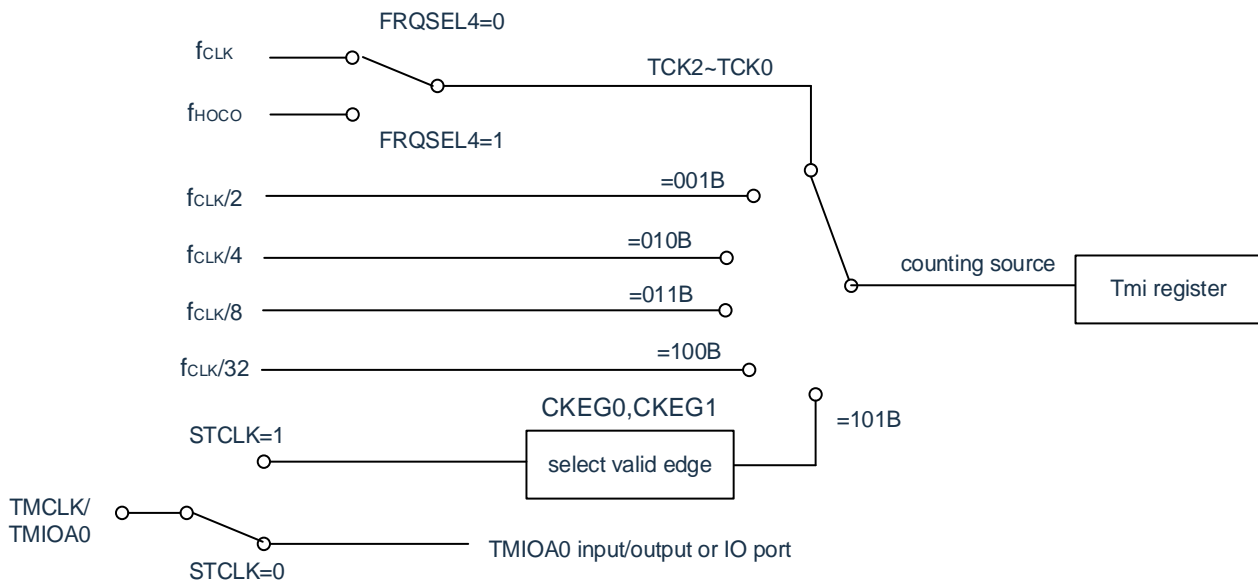
The method of selecting the counting source is the same for all patterns. However, in PWM3 mode, an external clock cannot be selected.

Table 10-9 Selection of counting sources

Counting source	Selection method
f_{CLK} , $f_{CLK}/2$, $f_{CLK}/4$, $f_{CLK}/8$, $f_{CLK}/32$	The counting source is selected by the TCK2~TCK0 bits of the TMCRI register.
External input signal of the TMCLK pin	The STCLK bit of the TMFCR register is "1" (the external clock input is valid). The TCK2~TCK0 bits of the TMCRI register are "101B" (the counting source is an external clock). The effective edge is selected by the CKEG1 to CKEG0 bits of the TMCRI register. The bit of the port mode register of the multiplexed I/O port of the TMCLK pin is "1" (input mode).

Note $i = 0, 1$

Figure 10-41 Block diagram of counting source selection



Note $i = 0, 1$

The pulse width of the external clock input to the TMCLK pin must be at least 3 timer M operating clock (f_{CLK}) cycles.

10.4.2 The buffer operation

The TMBFCi (i=0, 1) bits and TMBFDi bits of the TMMR registers can be combined with the TMBRCi registers. The TMGRDi registers are set as buffer registers for the TMGRAi registers and the TMGRBi registers, respectively.

- Buffer register for TMGRAi: TMGRCi register
- Buffer register for TMGRBi: TMGRDi register

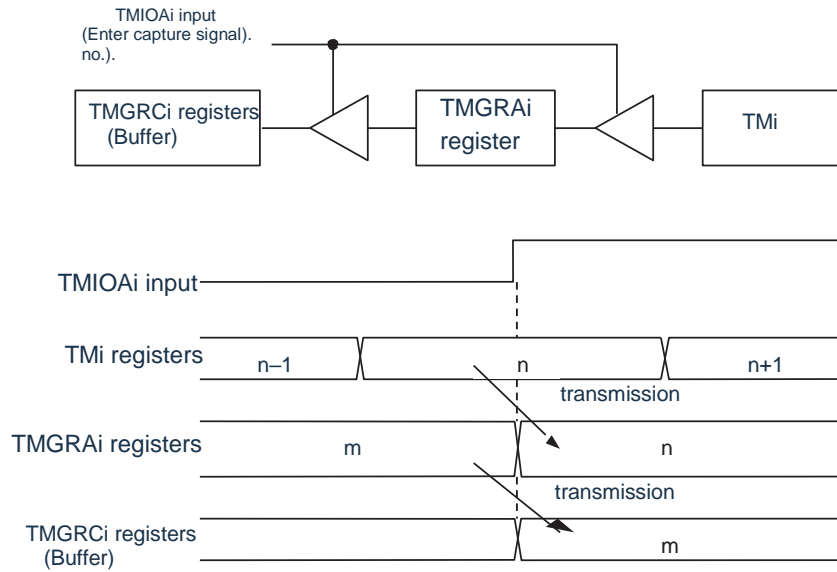
Buffer operation varies depending on the timer mode, and the buffer operation of each mode is shown in Table 10-10.

Table 10-10 Buffer operation for each mode

Features and modes		Transmission timing	Registers for transfer
Timer mode	Enter the capture function	The input signal of TMIOAi (Input to capture signal)	Transfer the contents of the TMGRAi register to the TMGRCi register (buffer register).
		The input signal of the TMIOBi (Input to capture signal)	Transfer the contents of the TMGRBi register to the TMGRDi register (buffer register).
	Output comparison function	Comparison matching of TMi registers and TMGRAi registers	Transfer the contents of the TMGRCi register (buffer register) to the TMGRAi register.
		Comparison matching of TMi registers and TMGRBi registers	Transfer the contents of the TMGRDi register (buffer register) to the TMGRBi register.
	PWM function	Comparison matching of TMi registers and TMGRAi registers	Transfer the contents of the TMGRCi register (buffer register) to the TMGRAi register.
		Comparison matching of TMi registers and TMGRBi registers	Transfer the contents of the TMGRDi register (buffer register) to the TMGRBi register.
Reset synchronous PWM mode		Comparison matching of TM0 registers and TMGRA0 registers	Transfer the contents of the TMGRCi register (buffer register) to the TMGRAi register. Transfer the contents of the TMGRDi register (buffer register) to the TMGRBi register.
Complementary PWM mode		<ul style="list-style-type: none">• When the CMD1 bits and CMD0 bits of the TMFCR register are "11B", it is the underflow of the TMFCR register.• When the CMD1 bit and CMD0 bits of the TMFCR register are "10B", the comparison match between the TMFCR register and the TMGRA0 register is matched.	Transfers the contents of the TMGRC1 register (buffer register) to the TMGGRA1 register. Transfer the contents of the TMGRDi register (buffer register) to the TMGRBi register.
PWM3 mode		Comparison matching of TM0 registers and TMGRA0 registers	Transfer the contents of the TMGRCi register (buffer register) to the TMGRAi register. Transfer the contents of the TMGRDi register (buffer register) to the TMGRBi register.

Note i = 0, 1

Figure 10-42 The buffer operation of the input capture function

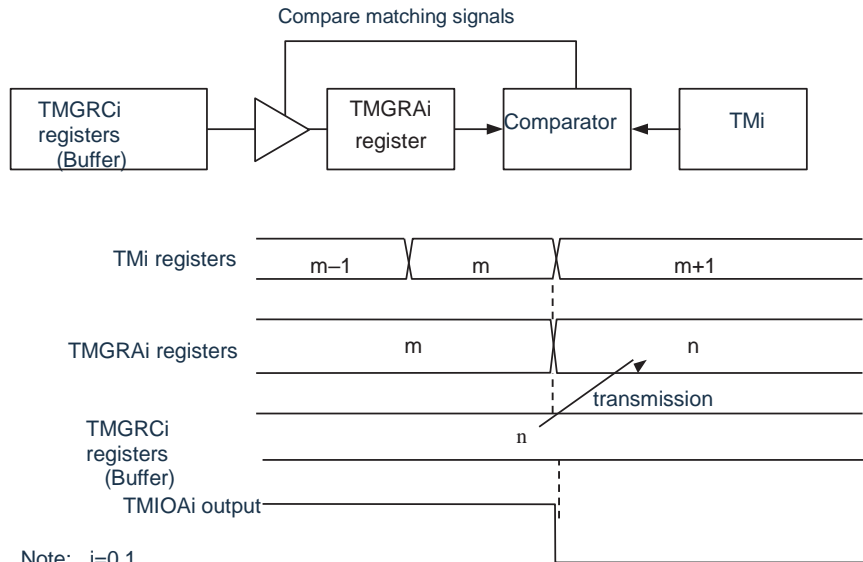


Note: $i=0,1$

The conditions in the above figure are as follows:

- The TMBFCi bit of the TMMR register is "1" (the TMGRCi register is the buffer register of the TMGRAi register).
- The IOA2IOA0 bit of the TMIOAi register is "100B" (input snapped on the rising edge).

Figure 10-43 The buffer operation of the output comparison function



Note: $i=0,1$

The conditions in the above figure are as follows:

- The TMBFCi bit of the TMMR register is "1" (the TMGRCi register is the buffer register of the TMGRAi register).
- The IOA2IOA0 bit of the TMIOAi register is "001B" (when the comparison matches, the output "L" level).

In timer mode (input capture function and output comparison function), the following settings must be made. Case of using the TMGRCi (i=0, 1) register as a buffer register for the TMGRAi register:

- The IOC3 position of the TMIORCi register must be "1" (general purpose register or buffer register).
- The IOC2 bit of the TMIORCi register must be given the same value as the IOMA2 bit of the TMIORAi register.

Cases where the TMGRDi register is used as a buffer register for the TMGRBi register:

- The IOD3 position of the TMIORCi register must be "1" (general purpose register or buffer register).
- The IOD2 bit of the TMIORCi register must be set to the same value as the IOB2 bit of the TMIORAi register.

When using the input capture function, even if the TMGRCi register and the TMGRDi register are used as buffer registers, the TMSRi registers are used on the input edge of the TMIOCi pin and the TMIODi pin. The IMFC bits and IMFD bits also become "1".

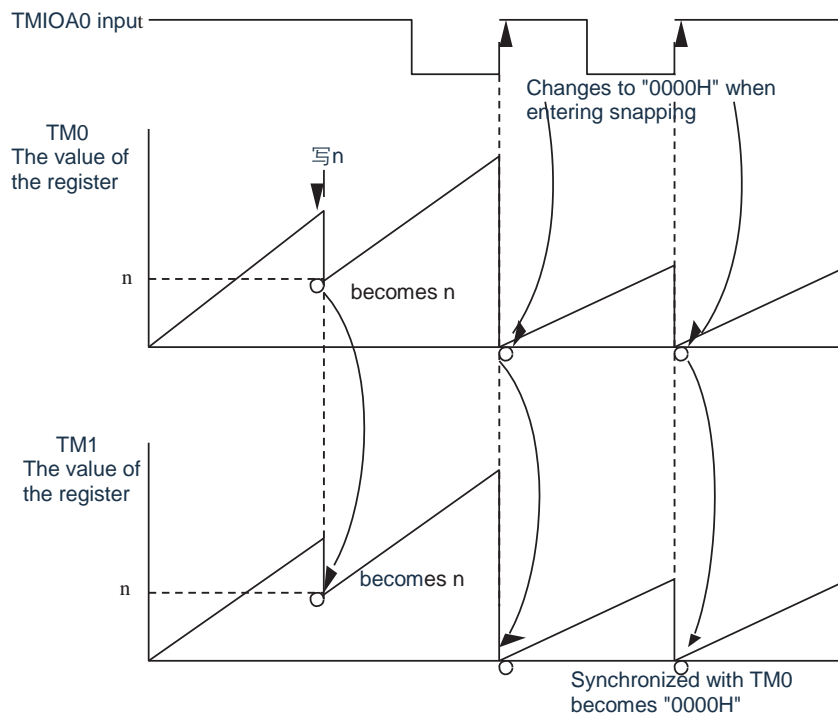
When using the output comparison function or the PWM function or in reset synchronous PWM mode, complementary PWM mode and PWM3 mode, even if the TMGRCi register and TMGRDi are used. The register is used as a buffer register, and when the TMi register is matched, the IMFC bits and IMFD bits of the TMSRi register also become "1".

10.4.3 Synchronous Operation

Synchronize the TM0 register with the TM1 register.

- Sync presets
If you write the TMi register when the TMSYNC bit of the TMMR register is "1" (synchronous operation), the data is written to both the TM0 register and the TM1 register.
- Sync clear
When the TMSYNC bit of the TMMR register is "1" and the CCLR2~CCLR0 bit of the TMCRO register is "011B." "(Synchronous clearing), the TM0 register and the TM1 register both become "0000H".
Similarly, when the TMSYNC bit of the TMMR register is "1" and the CCLR2~CCLR0 bit of the TMCRI register is "011B" (simultaneous clear), the TM1 register and TM0 registers become "0000H" at the same time.

Figure 10-44 runs synchronously



The conditions in the above figure are as follows:

- The TMSYNC bit of the TMMR register is "1" (synchronous operation).
- The CCLR2~CCLR0 bit of the TMCRO register is "001B" (when entering the capture, the TM0 is set to "0000H").
- The CCLR2~CCLR0 bit of the TMCRI register is "011B" (synchronized with TM0, TM1 is set to "0000H").
- TMIOA0RegistersIOA2~IOA0Bit is "100B"
- The CMD1 bit and CMD0 bit of the TMFCR register } (The rising edge of the TMIOA0 input is snapped at the input).
- The PWM3 bit of the TMFCR register is "1"

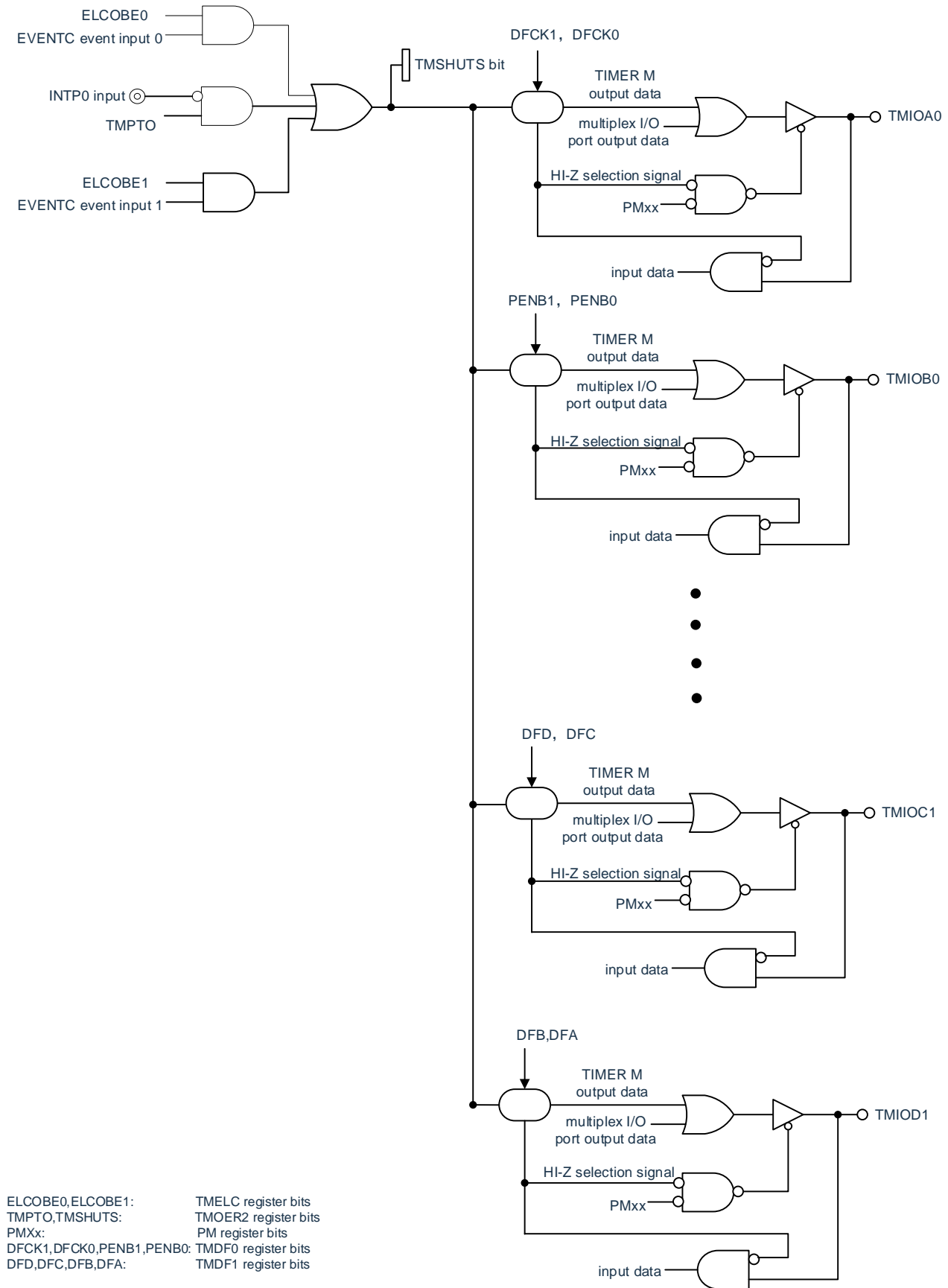
10.4.4 Forced cutoff of the pulse output

When using the PWM function or in reset synchronous PWM mode, complementary PWM mode, and PWM3 mode, TMIO_{ji} can be cut off by the input of the INTP0 pin Pulse output of the output pins (i=0, 1, j=A, B, C, D).

If the corresponding position of the TMOER1 register is "0" (allowing the output of timer M), the output pins used in these functions or modes are used as the output pins of timer M. When the TMPTO bit of the TMOER2 register is "1" (the pulse output force-cut signal INTP0 pin input is valid), the TMDF0 register is either TMDF1 registers for DFCK1, DFCK0, PENB1, PENB0, DFD, the output values of the DFC, DFB, and DFA bit settings are output from the output pins used as the M output port for the timer.

When you use this feature, you must make the following settings:

- Pin state (high impedance, "L" level output, or "H" level output) when the pulse output is forced to cut off through the TMDF_i registers.
- For pulse forced cutoff via EVENTC event input, refer to "10.4.5 Events Entered from the Event Linkage Controller (EVENTC)".
- At the output of the forced cutoff pulse, the TMOER2 register has a TMSTUTS bit of "1". To abort the output of the forced cutoff pulse, the TMSTUS position must be "0" in the process of stopping the counting (TSTART_i=0).
- Put the TMPTO position of the TMOER2 register "1" (the pulse output force cutoff signal INTP0 pin is valid).

Fig. 10-45 Forced cutoff of pulse output


10.4.5 Event inputs from the Event Linkage Controller (EVENTC).

For events entered by EVENTC, timer M performs 2 operations.

(a) Input capture for TMIOD0/TMIOD1

With event input from EVENTC, timer M performs input capture of TMIOD0/TMIOD1. At this point, the IMFD bit of the TMSRi register is "1".

To use this function, you must select the input capture function of the timer mode and change the ELCICE0 bit of the TMELC register or the ELCICE1 position "1". In other modes (output comparison function for timer mode, PWM function, reset synchronous PWM mode, complementary PWM mode, PWM3 mode), this function is invalid.

(b) Force cutoff pulse output of the run note

The output of the forced cutoff pulse is forced through the event input of the EVENTC. To use this function, the pulse output mode (PWM function, reset synchronous PWM mode, complementary PWM mode, PWM3 mode) must be selected and ELCOBE 0 bit or ELCOBE1 position "1". This feature has no effect when using the input capture feature in timer mode.

Note The forced cutoff function of the INTP0 pin cuts off the pulse output during the "L" level input, but the forced cutoff function for event 1 for event of event CSV input, cutoff 1 Secondary pulse output.

Setup steps

- (1) Set the EVENTC event linkage target to timer M.
- (2) Combine the ELCICEi bits of the TMELC registers (i=0, 1) and the ELCOBEi bits (i=0, 1) Set "1".

10.4.6 Event output to the Event Linkage Controller (EVENTC)/Data Transfer Controller (DMA).

The mode of timer M and the events output to EVENTC/DMA are shown in Table 10-11.

Table 10-11 Timer M Modes and Events Output to ELC/DMA

Usage patterns	Output source	EVENTC	DMA
Enter the capture function	TMIOA0 edge detection through IOA1 bit and IOA0 bit settings of TMIOA0 registers	○	○
	TMIOB0 edge detection through IOB1 bit and IOB0 bit settings of TMIOA0 registers	○	○
	TMIOC0 edge detection through IOC1 bit and IOC0 bit settings of TIORC0 registers	—	○
	TMIOD0 edge detection by IOD1 bit and IOD0 bit settings of TMIORC0 registers	—	○
	TMIOA1 edge detection through TMIOA1 registerSO1 bits and IOA0 bit settings	○	○
	TMIOB1 edge detection through IOB1 bit and IOB0 bit settings of TMIOA1 registers	○	○
	TMIOC1 edge detection by TMIOC1 edge detection of IOC1 bit and IOC0 bit settings of TIORC1 registers	—	○
	TMIOD1 edge detection by IOD1 bit and IOD0 bit settings of TMIORC1 registers	—	○
Output comparison function, PWM function, Reset synchronous PWM mode, complementary PWM mode, PWM3 mode	Comparison matching of TM0 registers and TMGRA0 registers	○	○
	Comparison matching of TM0 registers and TMGRB0 registers	○	○
	Comparison matching of TM0 registers and TMGRC0 registers	—	○
	Comparison matching of TM0 registers and TMGRD0 registers	—	○
	Comparison matching of TM1 registers and TMGRA1 registers	○	○
	Comparison matching of TM1 registers and TMGRB1 registers	○	○
	Comparison matching of TM1 registers and TMGRC1 registers	—	○
Complementary PWM mode	Comparison matching of TM1 registers and TMGRD1 registers	—	○
	Underflow of TM1 registers	○	—

10.5 Operation of timer M

10.5.1 Enter the capture function

This is a function of measuring the width and period of an external signal. Triggered with the external signal of the TMIO_{ji} pins ($i=0, 1$, $j=A, B, C, D$), will be The contents of the TM_i register (counter) are transferred to the TMGR_{ji} register (input capture). Because the TMIO_{ji} pin and the TMGR_{ji} register are used in combination, they can be pin-selected as input capture functions, or other modes and functions.

The block diagram and running example of the input capture function are shown in Figures 10-46 and 10-47, respectively, and the specifications of the input capture function are shown in Table 10-12.

[illegible]

Table 10-12 Specifications for input capture function

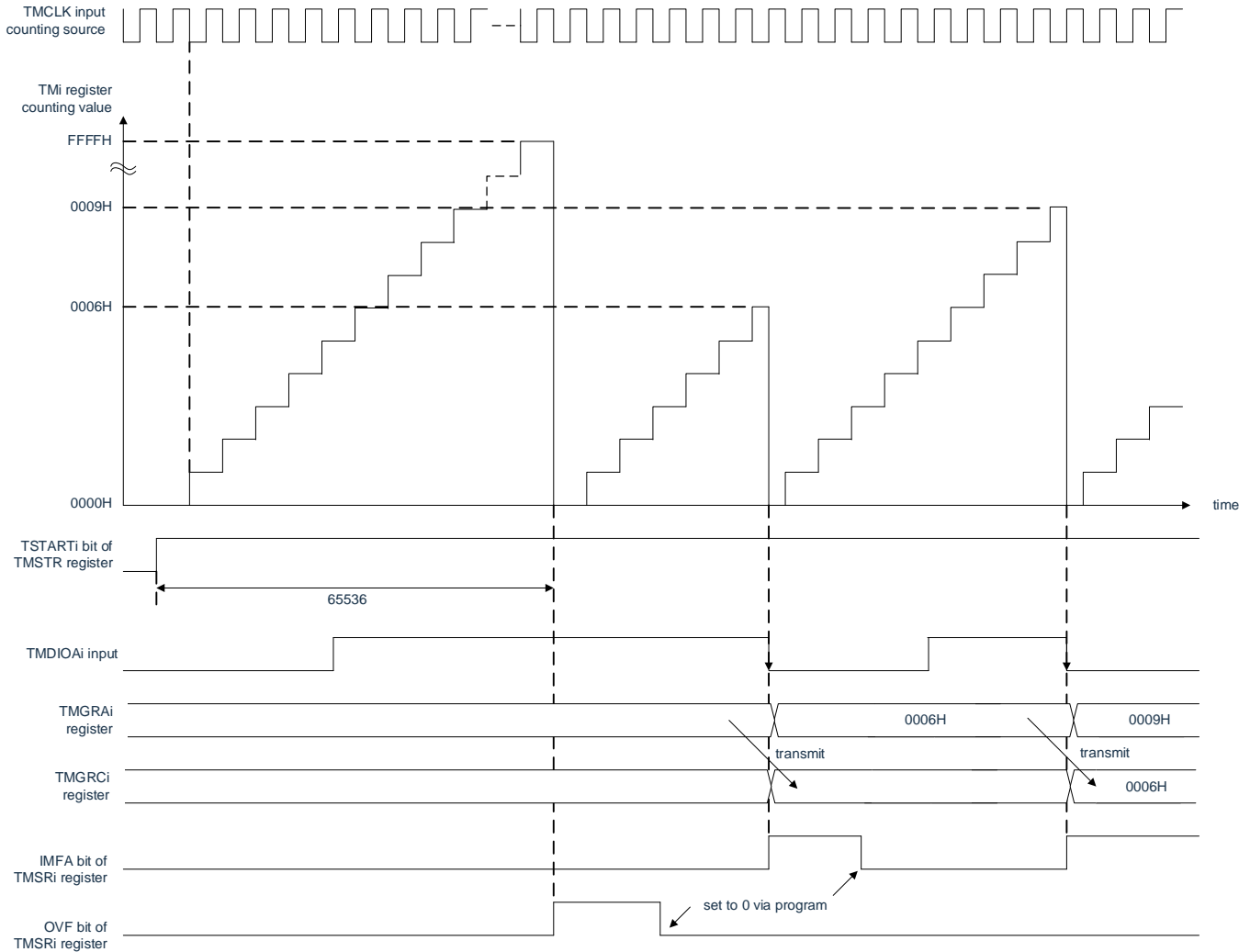
project	specification
Count the sources	f_{CLK} , $f_{CLK}/2$, $f_{CLK}/4$, $f_{CLK}/8$, $f_{CLK}/32$ An external input signal from the TMCLK pin that procedurally selects a valid edge
count	Increment the count
Count cycles	When the CCLR2~CCLR0 bits of the TMCRi register are "000B" (free-running). $1/f_k$ 65536 f_k : The frequency at which the source is counted
Count start criteria	Write "1" (start counting) to the TSTARTi bit of the TMSTR register.
Count stop conditions	When the CSELi bit of the TMSTR register is "1", write "0" (stop count) to the TSTARTi bit.
Timing of the generation of interrupt requests	• Input capture (effective edge of TMIOji input). Overflow of TMi
TMIOA0 pin function	I/O port, input capture input, or TMCLK (external clock) input
TMIOB0, TMIOC0, TMIOD0, TMIOA1~TMIOD1 pin function	I/O ports or inputs captured by pins
INTP0 pin function	Not used (input private port or INTP0 interrupt input).
Read timer	If you read the TMi register, you can read the count value.
Write timer	• Write the TMi register when the TMSYNC bit of the TMMR register is "0" (timer M0 and timer M1 operate independently). • If you write the TMi register when the TMSYNC bit of the TMMR register is "1" (timer M0 and timer M1 run simultaneously), The data is written to both the TM0 register and the TM1 register.
Select Features	• Input capture of the selection of input pins One or more pins in the TMIOAi, TMIOBi, TMIOCi, TMIODi pins Select the rising, falling, or bilateral edges of the input valid edges for input snaps Timing overflow or input capture of TMi to "0000H" • Buffer run (see "10.4.2 buffer run"). • Synchronous run (see "10.4.3 sync run"). • Digital filters The TMIOji input is sampled and if the signals are the same 3 times, the level is considered determined. • Input capture runs through ELC input events

Remark i=0, 1, j=A, B, C, D

(1) Operation example

By setting the CCLR0 to CCLR2 bits of the TMCRI registers ($i=0, 1$), the timer M i is applied when input capture or comparison matching occurs. The count value is reset. Fig. 10-47 is an example of the operation when the CCLR2~CCLR0 position "001B" is used. If it is set to count clearing by input capture during operation and input capture when the timer's count value is "FFFFH", the IMFA~IMFD of the TMSRi register depends on the runtime sequence of the counting source and input capture. The interrupt flag for both bits and OVF bits may change to "1" at the same time.

Figure 10-47 An example of an input capture function



Note $i=0, 1$

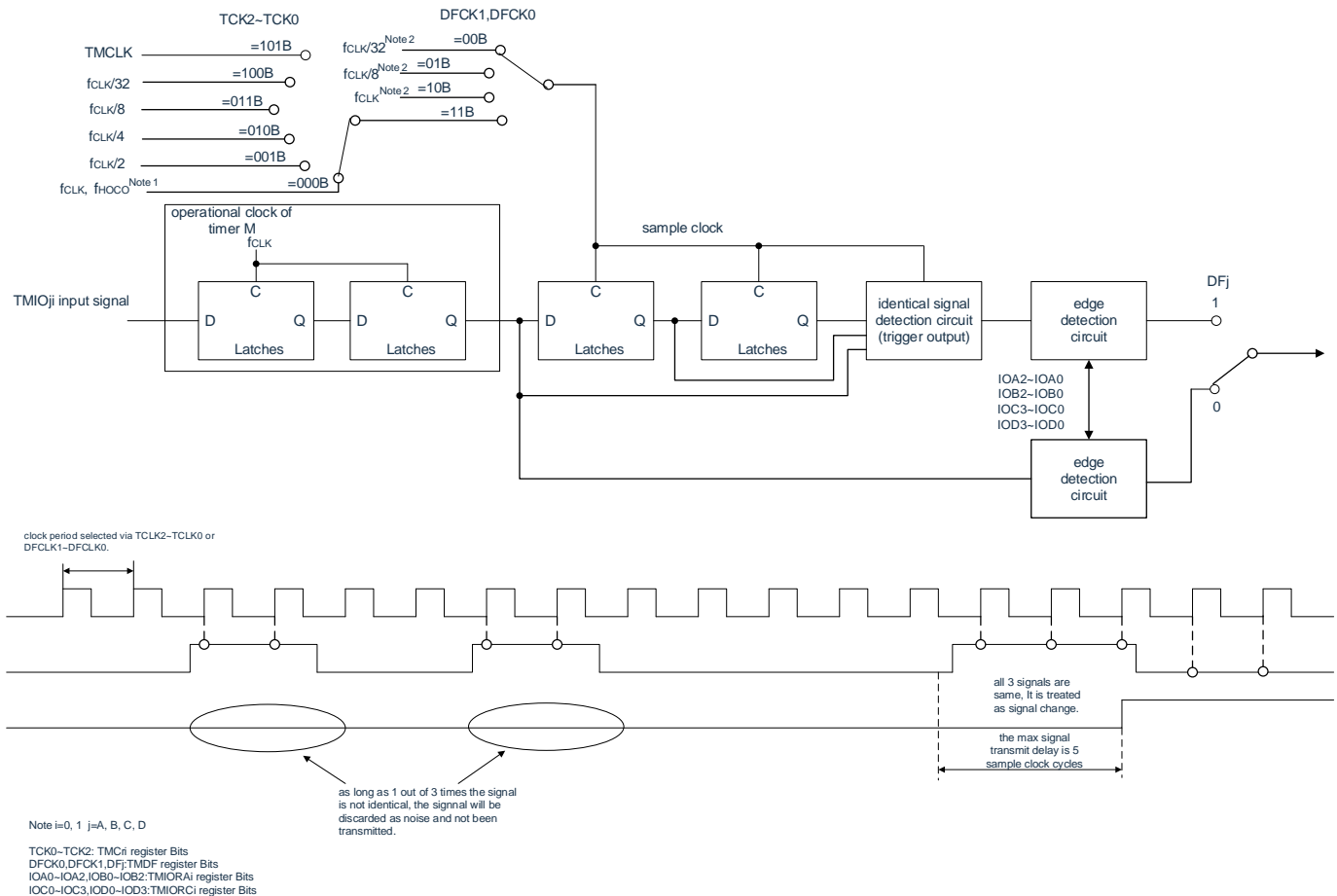
above diagram condition as following:
 CCLR2~CCLR0 bit of TMCRI register as "001B" (set Tmi bit as "0000H" while TMGRAi input capture)
 TCLK2~TCLK0 bit of TMCRI register as "101B" (Counting source as TMCLK input)
 CKEG1 bit and CKEG0 bit of TMCRI register as "01B" (counting at falling edge of counting source).
 IOA2~IOA0 bit of TMIOAi register as "101B" (input capture at falling edge of TMIOAi input)
 TMBFCi bit of TMMR register as "1" (TMGRci register is the buffer register of TMGRAi register)

(2) Digital filters

Sample the TMIOj inputs ($i=0, 1, j=A, B, C, D$) if the signal 3 times the same, the level is considered to have been determined. The function and sample clock of the digital filter must be selected through the TMDFi registers.

The block diagram of the digital filter is shown in Figure 10-48.

Figure 10-48 Block diagram of a digital filter



10.5.2 Output comparison function

This is the content of the detection TMi register (counter) (i=0, 1) and the TMGRji register (j=A, B, C, D) whether the content is the same (comparative match) pattern. If the content is the same, output any level from the TMIOji pin. Because of the TMIOji pin and the TMGRji Registers are used in combination so that pin selection can be made for output comparison functions, or other modes and functions. The block diagram and operation example of the output comparison function are shown in Figures 10-49 and 10-50, respectively, and the specifications of the output comparison function are shown in Table 10-13.

Figure 10-49 Block diagram of the output comparison function

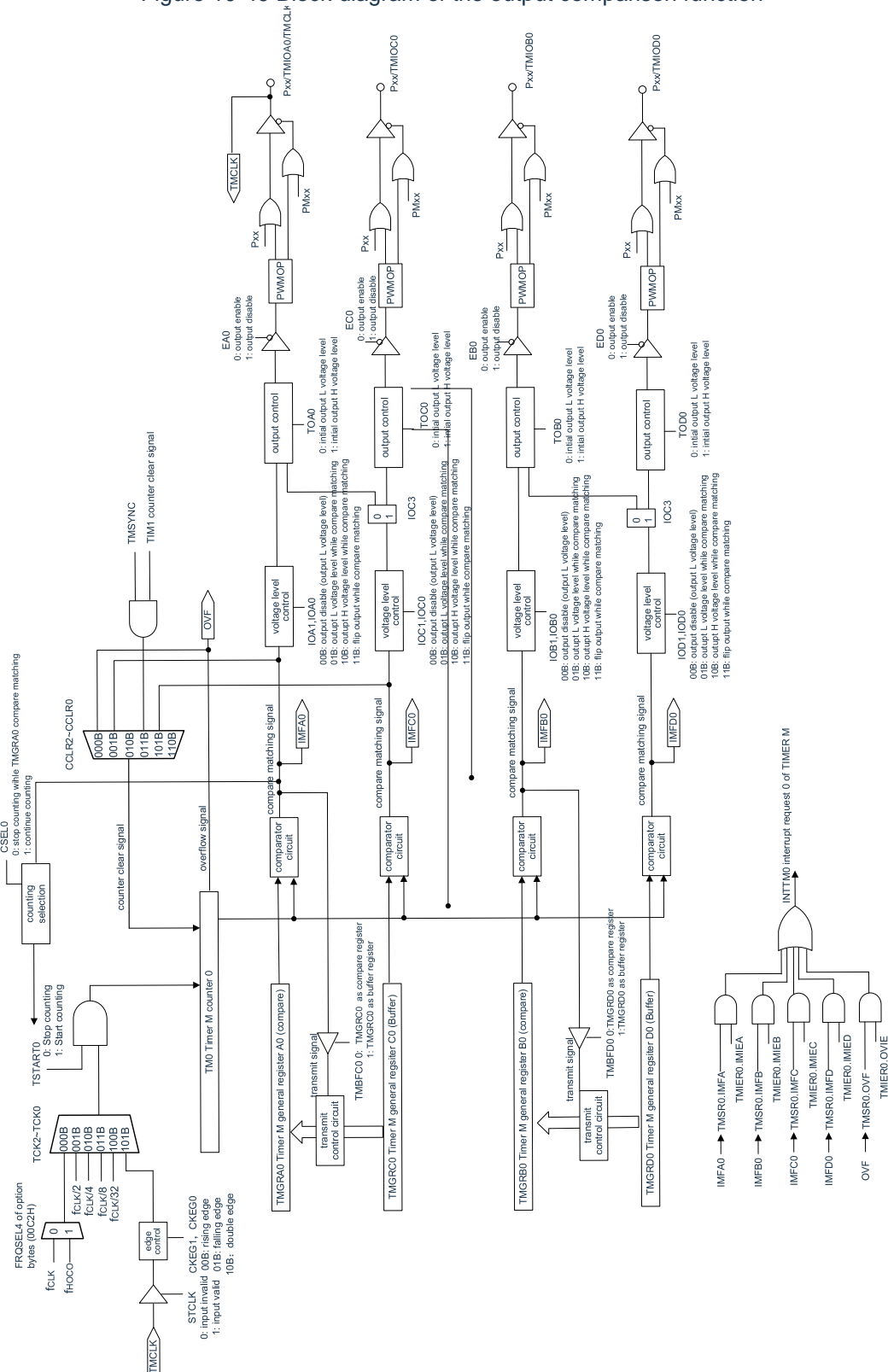


Table 10-13 Specifications of the output comparison function

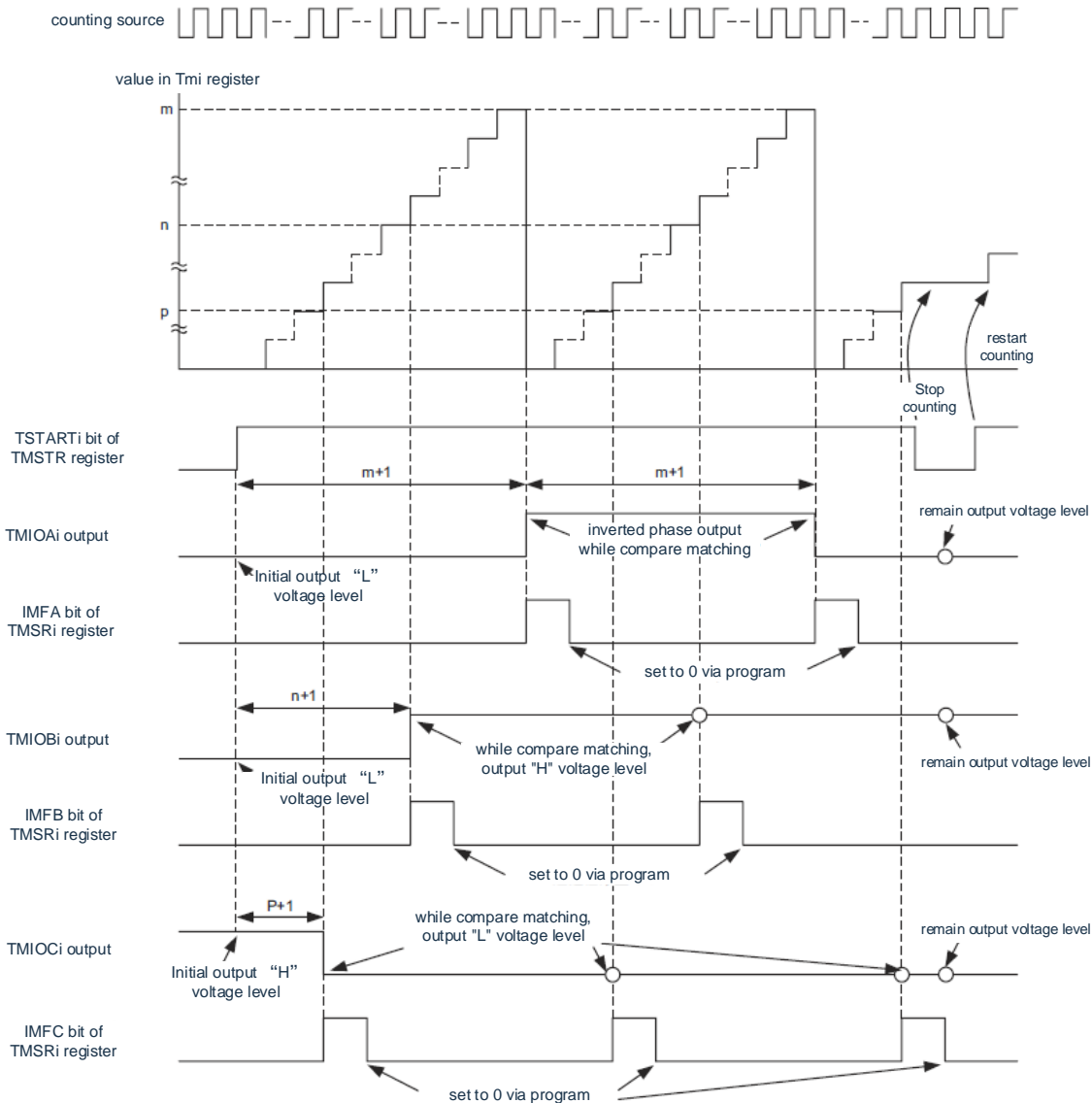
project	specification
Count the sources	f_{CLK} , $f_{CLK}/2$, $f_{CLK}/4$, $f_{CLK}/8$, $f_{CLK}/32$ An external input signal from the TMCLK pin that procedurally selects a valid edge
count	Increment the count
Count cycles	<ul style="list-style-type: none"> When the CCLR2~CCLR0 bit of the TMCRI register is "000B" (free running). 1/fk 65536 fk: The frequency at which the source is counted When the CCLR1~CCLR0 bits of the TMCRI register are "01B" or "10B" (when the TMGRji comparison matches, when the TMI is set to "0000H"). 1/fk (n+1) n: The setting value of the TMGRji register
Waveform output timing	Comparison matching (TMI registers and TMGRji registers have the same content).
Count start criteria	Write "1" (start counting) to the TSTARTi bit of the TMSTR register.
Count stop conditions	<ul style="list-style-type: none"> Write "0" (stop count) to the TSTARTi bit when the CSELi bit of the TMSTR register is "1". The output pin for output comparison maintains the output level before stopping the count. Stop counting when the CSELi bit of the TMSTR register is "0" and a comparison match of TMGRAi occurs. The output pins of the output comparison remain the level at which the comparison matches the output change.
Timing of the generation of interrupt requests	<ul style="list-style-type: none"> Comparison matching (TMI registers and TMGRji registers have the same content). Overflow of TMI
TMIOA0 pin function	I/O port, output for output comparison, or TMCLK (external clock) input
TMIOB0, TMIOC0, TMIOD0, TMIOA1~TMIOD1 pin function	I/O ports or outputs for output comparison (selected by pin).
INTP0 pin function	Not used (input private port or INTP0 interrupt input).
Read timer	If you read the TMI register, you can read the count value.
Write timer	<ul style="list-style-type: none"> Write the TMI register when the TMSYNC bit of the TMMR register is "0" (timer M0 and timer M1 operate independently). If you write the TMI register when the TMSYNC bit of the TMMR register is "1" (timer M0 and timer M1 run simultaneously), The data is written to both the TM0 register and the TM1 register.
Select Features	<ul style="list-style-type: none"> Output comparison of the selection of output pins One or more pins in the TMIOAi, TMIOBi, TMIOCi, TMIODi pins Compare the selection of output levels when matching "L" level output, "H" level output, or level inversion output The initial output level is selected to set the level from the start of counting to the time of the comparison match. Set the TMI in the timing of "0000H" Comparison matching of overflow or TMGRAi registers Buffer run (see "10.4.2 Buffer Run"). Synchronous run (see "10.4.3 Sync Run"). Changes to the output pins of TMGRCi and TMGRDi TMGRCi and TMGRDi can be used for output control of TMIOAi pin and TMIOBi pin, respectively. Timer M can be used as an internal timer without output.

Remark i=0, 1, j=A, B, C, D

(1) Operation example

By setting the CCLR0 to CCLR2 bits of the TMCRI registers ($i=0, 1$), the timer M i is applied when input capture or comparison matching occurs. The count value is reset. If the comparison expectation value is "FFFFH", it changes from "FFFFH" to "0000H" in the same way as the overflow, and the overflow flag changes to "1".

Figure 10-50 An example of an output comparison function



Note: $i=0,1$

m: The setting value of the TMGRAi register

n: The setting value of the TMGRBi register

p: The setting value of the TMGRCi register

The conditions in the above figure are as follows:

The CSELi bit of the TMSTR register is "1" (Tmi does not stop when the comparison matches).

The TMBFCi bits and TMBFDi bits of the TMMR registers are "0" (TMGRCi and TMGRDi do not operate as buffers).

The EAi bits of the TMOER1 register, the EBi bits, and the ECi bits are "0" (allowing the outputs of TMIOAi, TMIOBi, and TMIOCi).

The CCLR2~CCLR0 bits of the TMCRI register are "001B" (when the TMGRAi comparison matches, the Tmi position is "0000H").

The TMOCR registers TOAi bit and TOBi bit are "0" (the initial output "L" level is used before comparison matching) and the TOCi bit is "1" (The initial output "H" level is before the comparison matches).

The IOA2~IOA0 bits of the TMIORAi register are "011B" (TMIOAi inverts the output when the TMGRAi comparison matches).

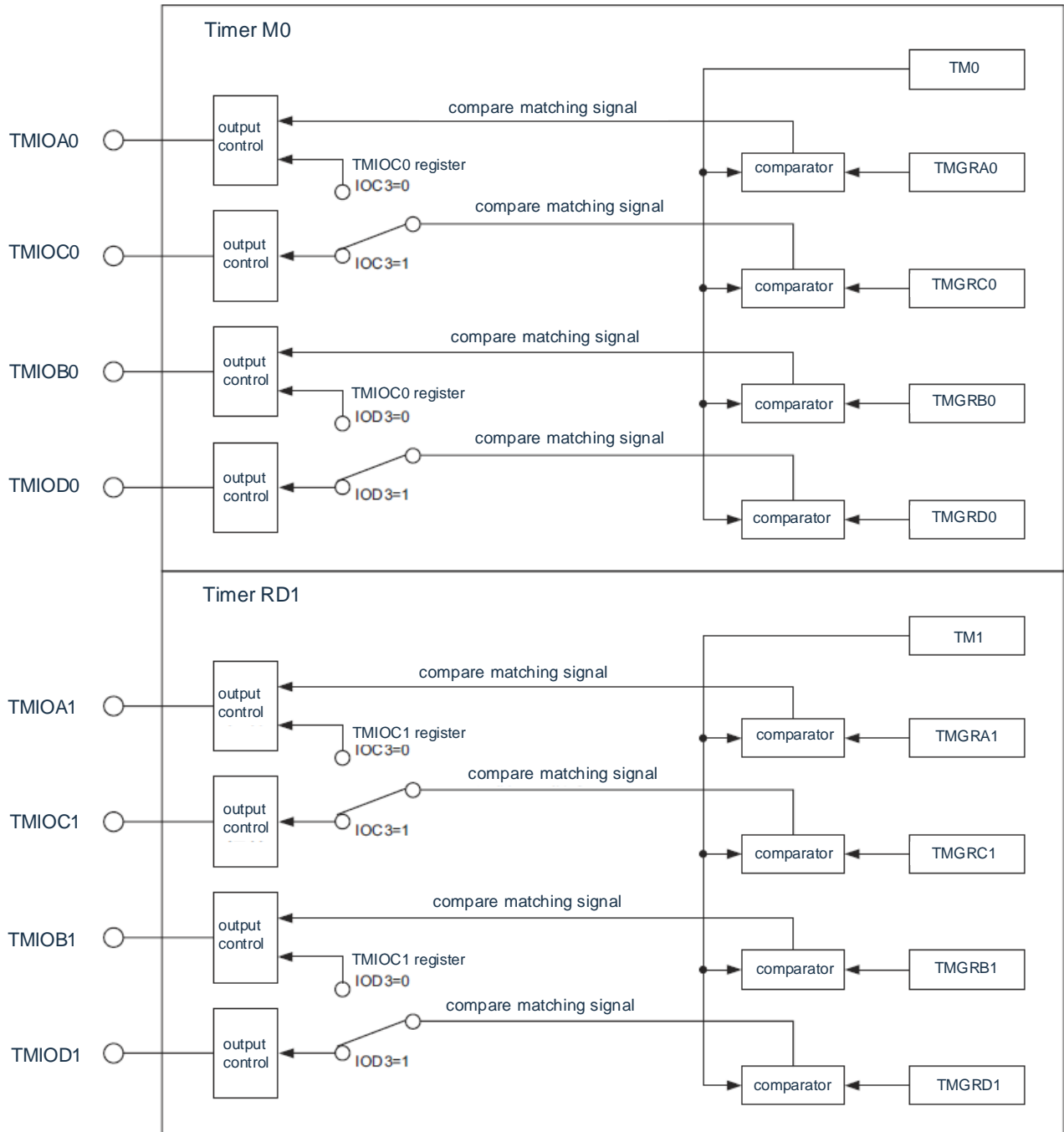
The IOB2~IOB0 of the TMIORAi register is "010B" (when the TMGRBi comparison matches, the TMIOBi outputs the "H" level).

(2) Change of output pins of the TMGR*Ci* register and the TMGRD*i* register (*i*=0, 1).

The TMGR*Ci* register and the TMGRD*i* register can be used for output control of the TMIOA*i* pin and the TMIOB*i* pin, respectively. Therefore, the output control of each pin can be performed as follows:

- The TMIOA*i* output is controlled by the value of the TMGRA*i* register and the value of the TMGR*Ci* register.
- The TMIOB*i* output is controlled by the value of the TMGRB*i* register and the value of the TMGRD*i* register.

Figure 10-51 Changes to the output pins of TMGR*Ci* and TMGRD*i*

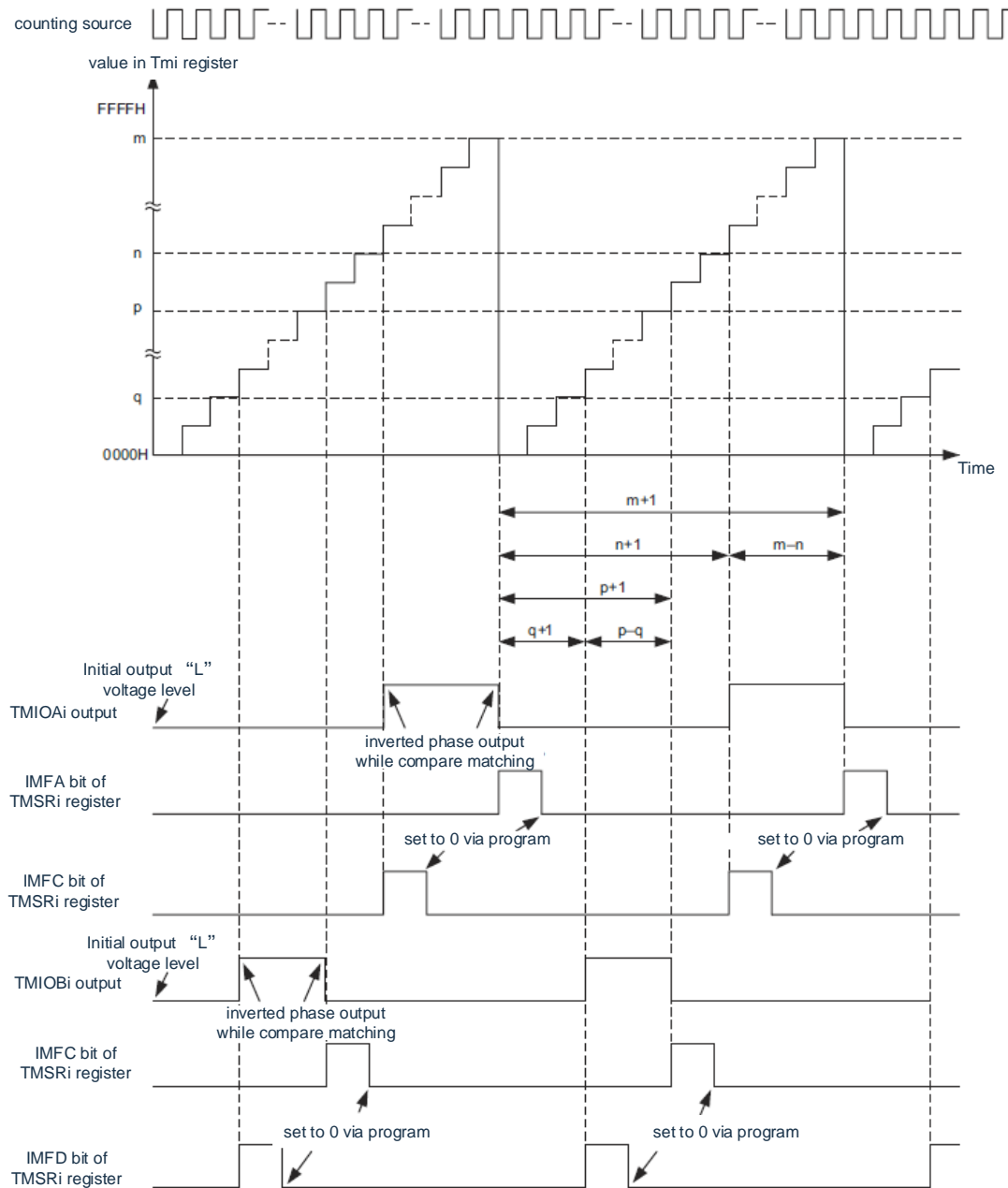


To change the output pins of the TMGR*Ci* register and the TMGRD*i* register, the following settings must be made:

- Select "0" (TMGR*ji*) via the IOj3 bit (*j*=C, D) of the TMIO*RCi* register Changes to the output pins of the register).
- Place the TMBF*ji* position of the TMMR register "0" (general purpose register).
- Set different values for the TMGRA*i* register and the TMGR*Ci* register, and set different values for the TMGRB*i* register and the TMGRD*i* register.

An example of how TBGR*i* and TMGR*i* are used for output control of the TMIOA*i* pin and TMIOB*i* pin, respectively, is shown in Figure 10-52.

Figure 10-52 uses tMGRC*i* and TMGRD*i* for examples of operation when using output control of the TMIOA*i* pin and TMIOB*i* pin, respectively



Note: $i=0,1$

m: The setting value of the TMGRA*i* register

n: The setting value of the TMGRB*i* register

p: The setting value of the TMGRC*i* register

q: The setting value of the TMGRD*i* register

The conditions in the above figure are as follows:

The CSEL*i* bit of the TMSTR register is "1" (TMi does not stop when the comparison matches).

The TMBFC*i* bits and TMBFD*i* bits of the TMMR registers are "0" (TMGRC*i* and TMGRD*i* do not operate as buffers).

The EA*i* bits of the TMOER1 register, the EB*i* bits, and the EC*i* bits are "0" (allowing the outputs of TMIOA*i*, TMIOB*i*, and TMIOCI).

The CCLR2~CCLR0 bits of the TMCRI register are "001B" (when the TMGRA*i* comparison matches, the TMi position is "0000H").

The TMOCR registers TOAi bits and TOBi bits are "0" (the initial output "L" level is before the comparison matches).
The IOA2~IOA0 bits of the TMIORAi register are "011B" (TMIOAi inverts the output when the TMGRAi comparison matches).
The IOB2~IOB0 of the TMIORAi register is "011B" (TMIOBi inverts the output when the TMGRBi comparison matches).
The IOC3~IOC0 of the TMIORCi register is "0011B" (TMIOAi inverts the output when the TMGRCi comparison matches).
The IOD3~IOD0 of the TMIORDi register is "1000B" (TMIOBi inverts the output when the T MGRCi comparison matches).

10.5.3 PWM function

This is the function of the output PWM waveform. Up to 3 PWM waveforms of the same period can be output via the timer Mi (i=0, 1). By synchronizing timer M0 with timer M1, up to six PWM waveforms of the same period can be output.

Because the TMIOj pins (j=B, C, D) and the TMGRji registers are used in combination, they can be selected by pin in the PWM function. or other modes and functions (however, the TMGRAi register is used regardless of which pin is used as the PWM function, so the TMGRAi register cannot be used for other modes).

The block diagram and specifications of the PWM function are shown in Figures 10-53 and Table 10-14, respectively, and the operating examples of the PWM functions are shown in Figures 10-54 and 10-55.

Figure 10-53 Block diagram of the PWM function

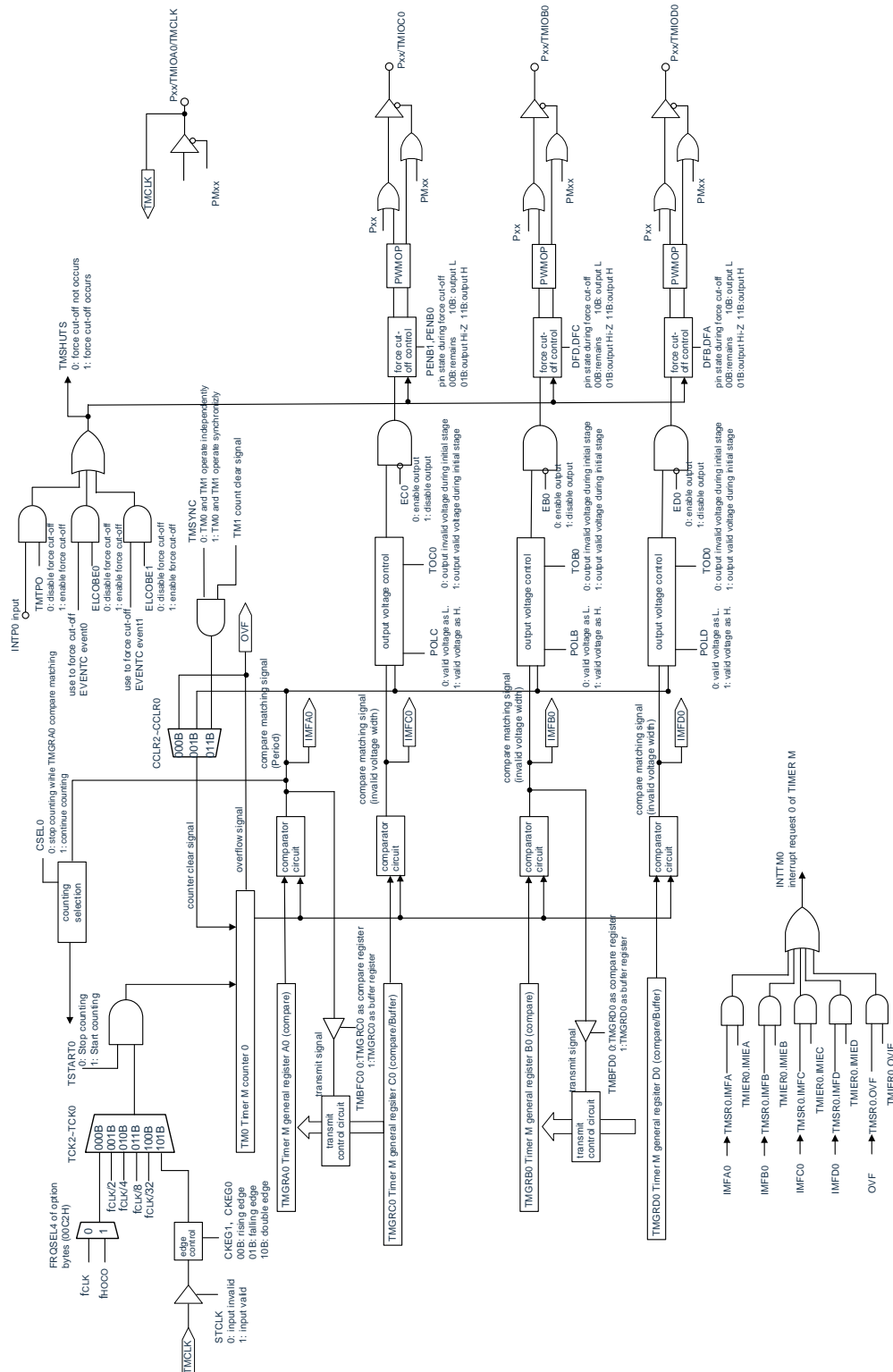
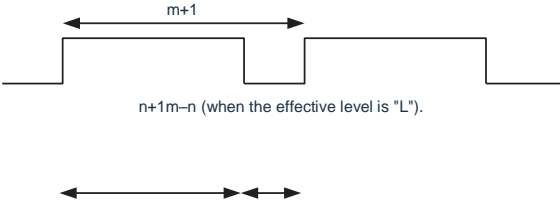


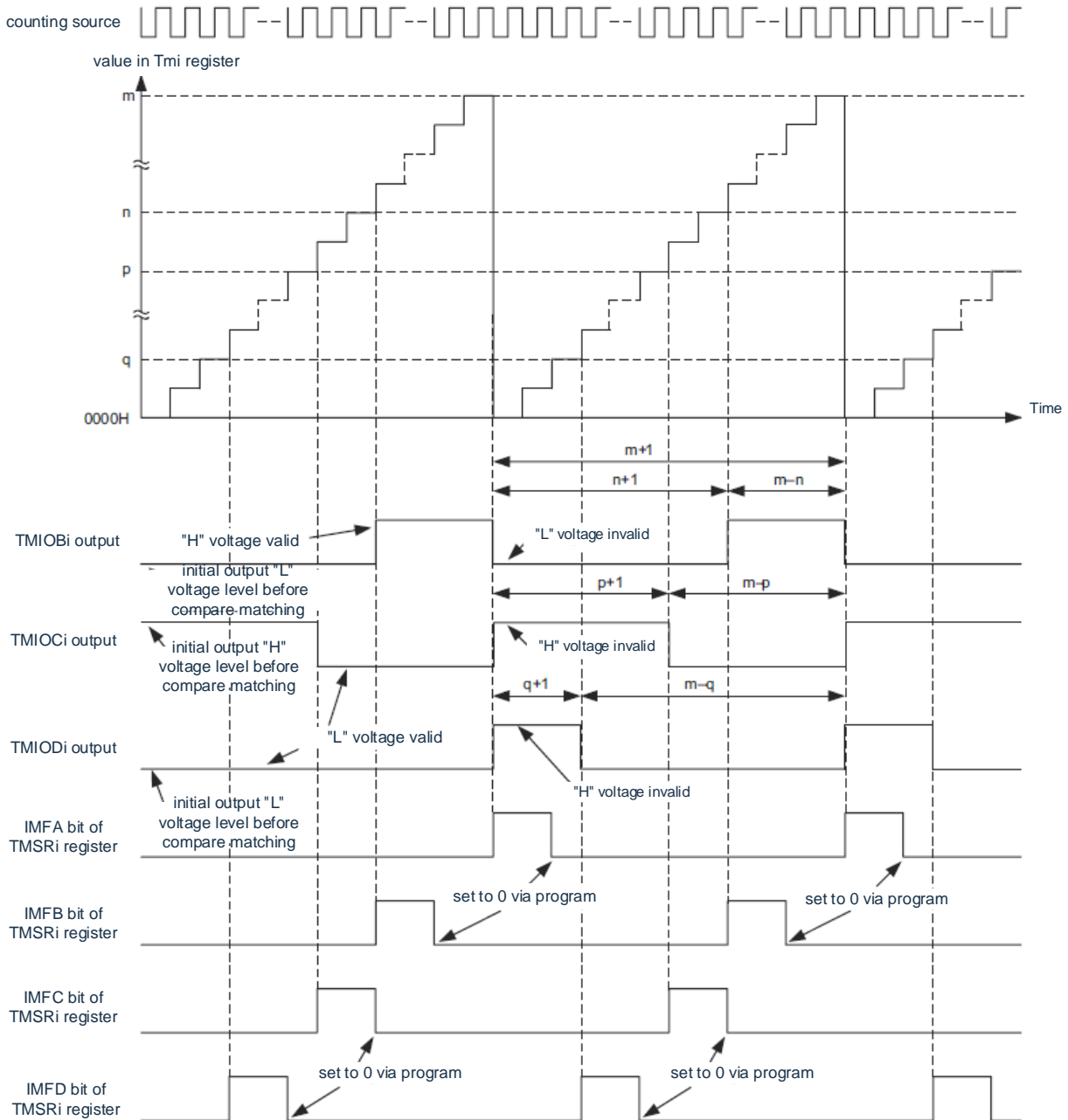
Table 10-14 Specifications for PWM functions

project	specification
Count the sources	f_{CLK} , $f_{CLK}/2$, $f_{CLK}/4$, $f_{CLK}/8$, $f_{CLK}/32$ An external input signal from the TMCLK pin that can programmatically select valid edges
count	Increment the count
PWM waveform	<p>PWM period: $1/f_k \times (m+1)$ Effective level width: $1/f_k \times (m-n)$ Invalid level width: $1/f_k \times (n+1)$. f_k: The frequency at which the source is counted m: The setting value of the TMGRAi register n: The setting value of the TMGRji register</p> 
Count start criteria	Write "1" (start counting) to the TSTARTi bit of the TMSTR register.
Count stop conditions	<ul style="list-style-type: none"> Write "0" (stop count) to the TSTARTi bit when the CSELi bit of the TMSTR register is "1". The PWM output pin holds the output level before stopping the count. Stop counting when the CSELi bit of the TMSTR register is "0" and a comparison match of TMGRAi occurs. The PWM output pins remain relatively matched after the output changes.
Timing of the generation of interrupt requests	<ul style="list-style-type: none"> Comparison matching (TMi registers and TMGRhi registers have the same content). Overflow of TMi
TMIOA0 pin function	I/O port or TMCLK (external clock) input
TMIOA1 pin function	I/O port
TMIOB0, TMIOC0, TMIOD0, TMIOB1, TMIOC1, TMIOD1 Pin function	I/O port or PWM output (select by pin).
INTP0 pin function	The pulse output is the input of the forced cutoff signal (input dedicated port or INTP0 interrupt input).
Read timer	If you read the TMi register, you can read the count value.
Write timer	Can write TMi registers.
Select Features	<ul style="list-style-type: none"> Selection of 1 to 3 PWM output pins via timer Mi One or more pins in the TMIOBi, TMIOCi, TMIODi pins Selection of effective levels for each pin Selection of initial output levels for each pin Synchronous run (see "10.4.3 sync run"). Buffer run (see "10.4.2 buffer run"). Input of the pulse output forced cutoff signal (see "Forced cutoff of 10.4.4 pulse output").

Remark $i=0, 1, j=B, C, D, h=A, B, C, D$

(1) Operation example

Figure 10-54 Example of operation of the PWM function



Note: $i=0,1$

m: The setting value of the TMGRAi register

n: The setting value of the TMGRBi register

p: The setting value of the TMGRCi register

q: The setting value of the T MGRDi register

The conditions in the above figure are as follows:

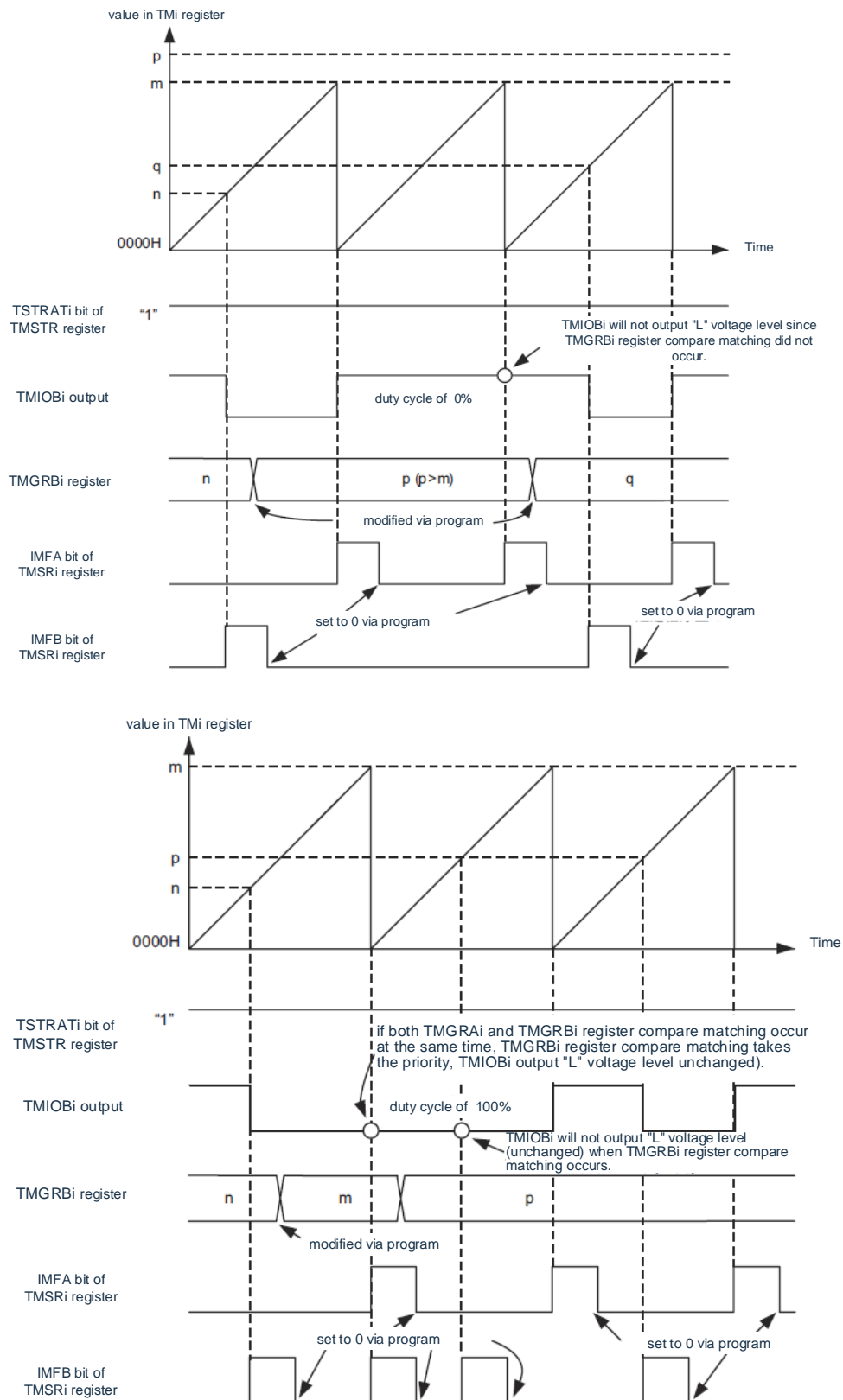
The TMBFCi bits and TMBFDi bits of the TMMR registers are "0" (TMGRCi and TMGRDi do not operate as buffers).

The EBi bit and ECi bit of the TMOER1 register are "0" (the output of TMIOBi and TMIOCi is allowed).

The TMOCR registers TOBi bits and TOCi bits are "0" (invalid level) and TODi bits are "1" (effective level).

The TMPOCRi register has a POLB bit of "1" (the "H" level is active), and the POLC bit and THE PLN bit are "0" (the "L" level is active).

Fig 10-55 PWM Example of running a feature (duty cycle 0% and 100%)



Note: i=0,1

m: The setting value of the TMGRAi register

The conditions in the above figure are as follows:

The EBi bit of the TMOER1 register is "0" (TMOBi output is allowed).

The POLB bit of the TMPOCRi register is "0" (the "L" level is active).

10.5.4 Reset synchronous PWM mode

Outputs the same-period PWM waveforms (three-phase, sawtooth wave modulation, no dead time) of 3 normal phases and 3 inverted phases (6 in total).

The block diagram and operation example of the reset synchronous PWM mode are shown in Figures 10-56 and 10-57, respectively, and the specifications of the reset synchronous PWM mode are shown in Table 10-15.

For examples of PWM operations with duty cycles of 0% and 100%, please refer to "Examples of PWM operations in Figure 10-55."

Figure 10-56 Block diagram of reset synchronous PWM mode

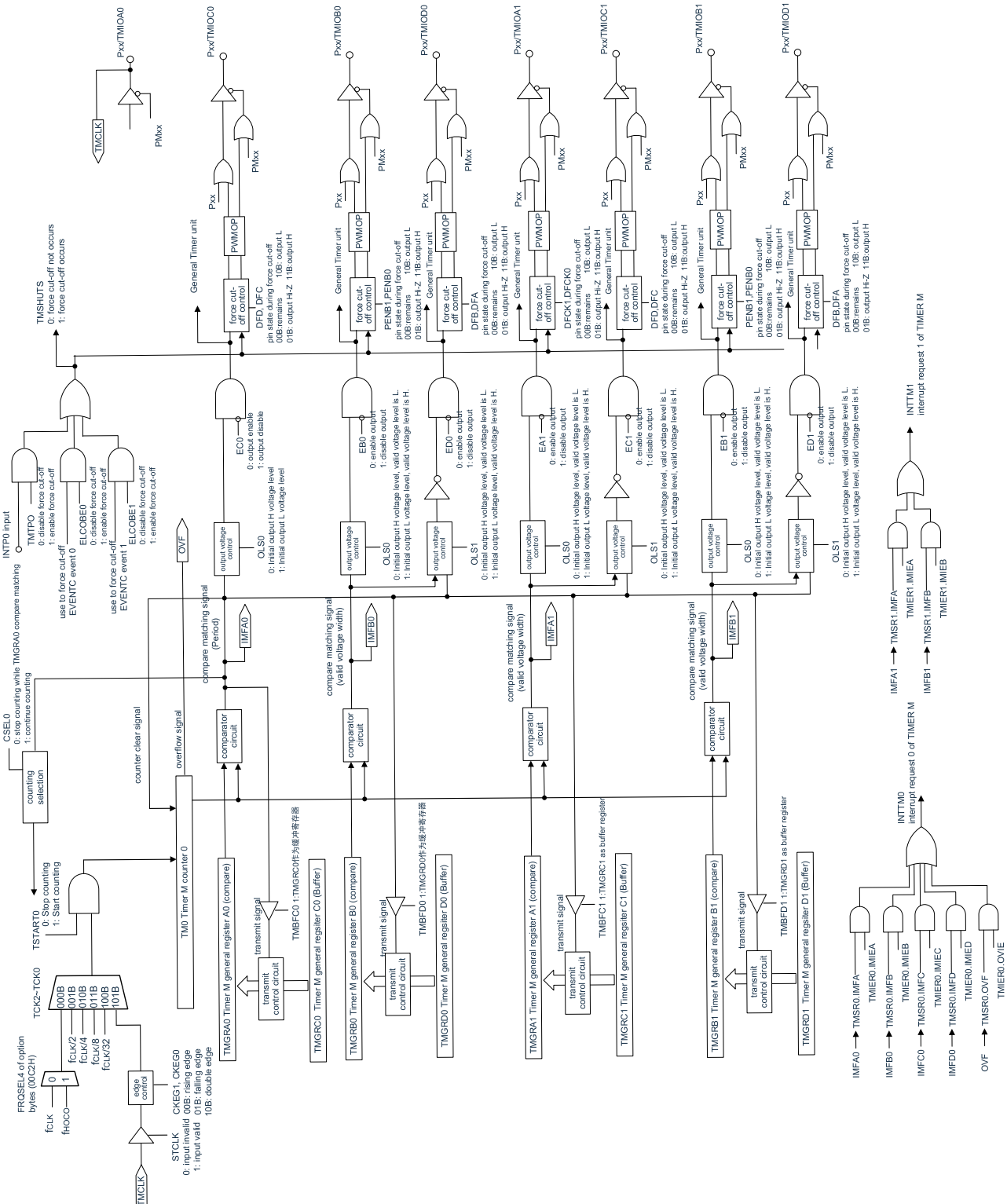
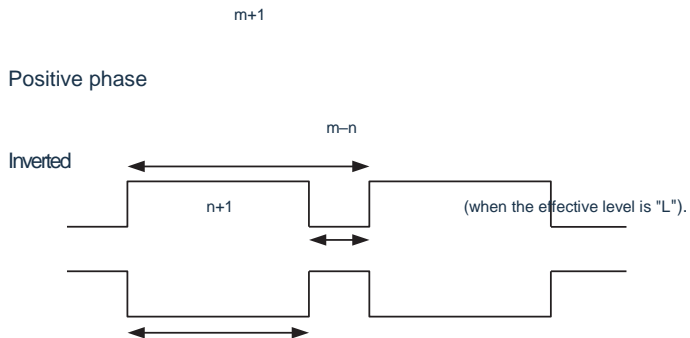


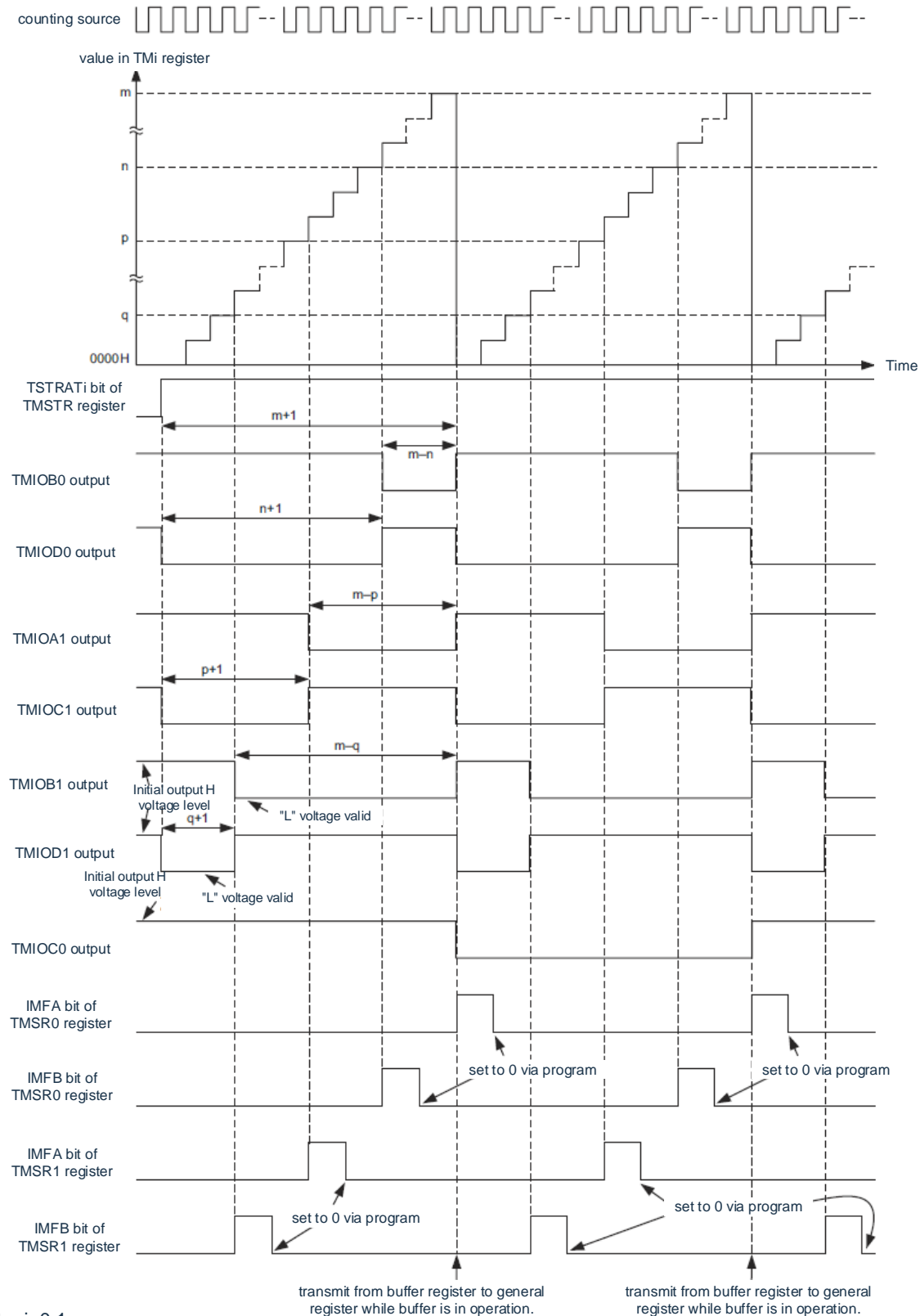
Table 10-15 Specifications for Reset Synchronous PWM Mode

project	specification
Count the sources	f_{CLK} , $f_{CLK}/2$, $f_{CLK}/4$, $f_{CLK}/8$, $f_{CLK}/32$ An external input signal from the TMCLK pin that can programmatically select valid edges
count	TM0 is an incremental count (TM1 is not used).
PWM waveform	<p>PWM period: $1/f_k \times (m+1)$ Normal phase effective level width: $1/f_k$ (m-n) Inverting effective level width: $1/f_k \times (n+1)$.</p> <p>f_k: The frequency of the counting source m:the setting value of the TMGRA0 register n: TMGRB0 register set value (PWM output 1) TMGRA1 register set value (PWM output 2) The setting value of the TMGRB1 register (PWM output 3).</p> 
Count start criteria	Write "1" (start counting) to the TSTART0 bit of the TMSTR register.
Count stop conditions	<ul style="list-style-type: none"> Write "0" (stop count) to the TSTART0 bit when the CSEL0 bit of the TMSTR register is "1". <p>The PWM output pin outputs the initial output level of the OLS0 bit and OLS1 bit selection of the TMFCR register.</p> <ul style="list-style-type: none"> Stop counting when the CSEL0 bit of the TMSTR register is "0" and a comparison match of
Timing of the generation of interrupt requests	<ul style="list-style-type: none"> Comparison matching (TM0 registers have the same content as TMGRj0, TMGRA1, TMGRB1 registers). Overflow of TM0
TMIOA0 pin function	I/O port or TMCLK (external clock) input
TMIOB0 pin function	The normal phase output of PWM output 1
TMIOD0 pin function	Inverting output of PWM output 1
TMIOA1 pin function	The normal phase output of PWM output 2
TMIOC1 pin function	Inverting output of PWM output 2
TMIOB1 pin function	The normal phase output of PWM output 3
TMIOD1 pin function	Inverting output of PWM output 3
TMIOC0 pin function	Inverting output at each PWM cycle.
INTP0 pin function	The pulse output is the input of the forced cutoff signal (input dedicated port or INTP0 interrupt input).
Read timer	If you read the TM0 register, you can read the count value.
Write timer	Can write TM0 registers.
Select Features	<ul style="list-style-type: none"> Selection of positive and negative phase effective levels and initial output levels Buffer run (see "10.4.2 buffer run"). Input of the pulse output forced cutoff signal (see "Forced cutoff of 10.4.4 pulse output").

Remark j=A, B, C, D

(1) Operation example

Figure 10-57 Operation example of reset synchronous PWM mode



Note: $i=0,1$

m: The setting value of the TMGRA0 register

n: The setting value of the TMGRB0 register

p: The setting value of the TMGRA1 register

q: The setting value of the TMGRB1 register

The conditions in the above figure are as follows:

The OLS1 bits and OLS0 bits of the TMFCR register are "0" (initial output "H" level, "L" level is active).

10.5.5 Complementary PWM mode

Outputs the same-period PWM waveforms (three-phase, triangular wave modulation, dead time) of 3 normal phases and 3 inverting phases (6 in total).

The block diagram of the complementary PWM mode is shown in Figure 10-58, the specifications of the complementary PWM mode are shown in Table 10-16, and the output model and running example of the complementary PWM mode are shown in Figure 10-59, respectively and Figure 10-60.

Figure 10-58 Block diagram of complementary PWM mode

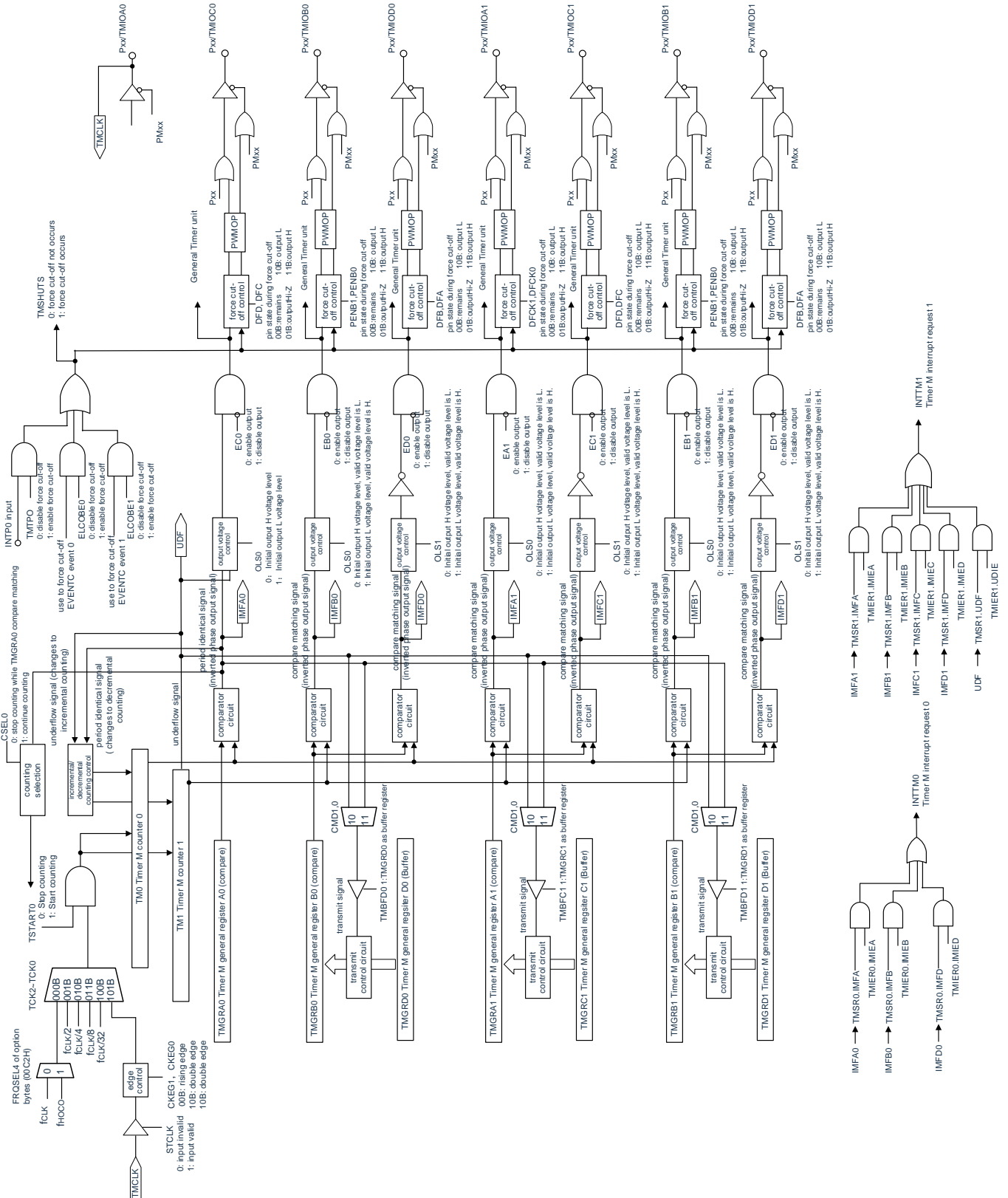
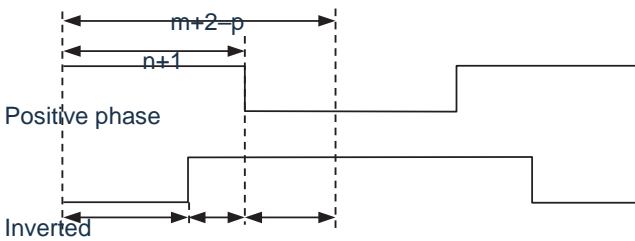


Table 10-16 Specifications for Complementary PWM Modes

project	specification
Count the sources	f_{CLK} , $f_{CLK}/2$, $f_{CLK}/4$, $f_{CLK}/8$, $f_{CLK}/32$ An external input signal from the TMCLK pin that can programmatically select valid edges The same values must be set for the TCK0~TCK2 bits of the TKG0 register and the TCK0~TCK2 bits of the TMCR1 register (same count source).
count	Increment the count or decrement the count If the TM0 register and TMGRA0 registers match during the process of increasing the count, both TM0 and TM1 become decrementing counts; If the TM1 register changes from "0000H" to "FFFFH" during the decrement of the count, both TM0 and TM1 become incrementing
PWM waveform	<p>PWM cycle: $1/f_k \times (m+2-p) \times 2$ note1 Dead time: p Normal phase effective level width: $1/f_k \times (m-n-p+1) \times 2$ Inverting effective level width: $1/f_k \times (n+1-p) \times 2$</p> <p>$f_k$: The frequency at which the source is counted m: The setting value of the TMGRA0 register n: The setting value of the TMGRB0 register (PWM output 1). The setting value of the TMGRA1 register (PWM output 2). The set value of the TMGRB1 register (PWM output 3). p: The setting value of the TM0 register</p> 
Count start criteria	Write "1" (start counting) to the TSTART0 bits and TSTART1 bits of the TMSTR register.
Count stop conditions	When the CSEL0 bit of the TMSTR register is "1", write "0" (stop count) to the TSTART0 bit and TSTART1 bit
Timing of the generation of interrupt requests	<ul style="list-style-type: none"> • Comparison matching (TMi registers and TMGRji registers have the same content). • Underflow of TM1
TMIOA0 pin function	I/O port or TMCLK (external clock) input
TMIOB0 pin function	The positive phase output pin of PWM output 1
TMIOD0 pin function	Inverting output pin of PWM output 1
TMIOA1 pin function	The positive phase output pin of PWM output 2
TMIOC1 pin function	The inverting output pin of PWM output 2
TMIOB1 pin function	The positive phase output pin of PWM output 3
TMIOD1 pin function	Inverting output pin of PWM output 3
TMIOC0 pin function	Inverting output at every 1/2 PWM cycle.
INTP0 pin function	The pulse output is the input of the forced cutoff signal (input dedicated port or INTP0
Read timer	If you read the TMi register, you can read the count value.
Write timer	Can write TMi registers.
Select Features	<ul style="list-style-type: none"> • Input of the pulse output forced cutoff signal (see "Forced cutoff of 10.4.4 pulse output"). • Selection of positive and negative phase effective levels and initial output levels • Selection of transfer timing of buffer registers

Note 1 After starting counting, the PWM cycle is fixed.

Remark i=0, 1, j=A, B, C, D

(1) Operation example

Fig. 10-59 Output model of complementary PWM mode

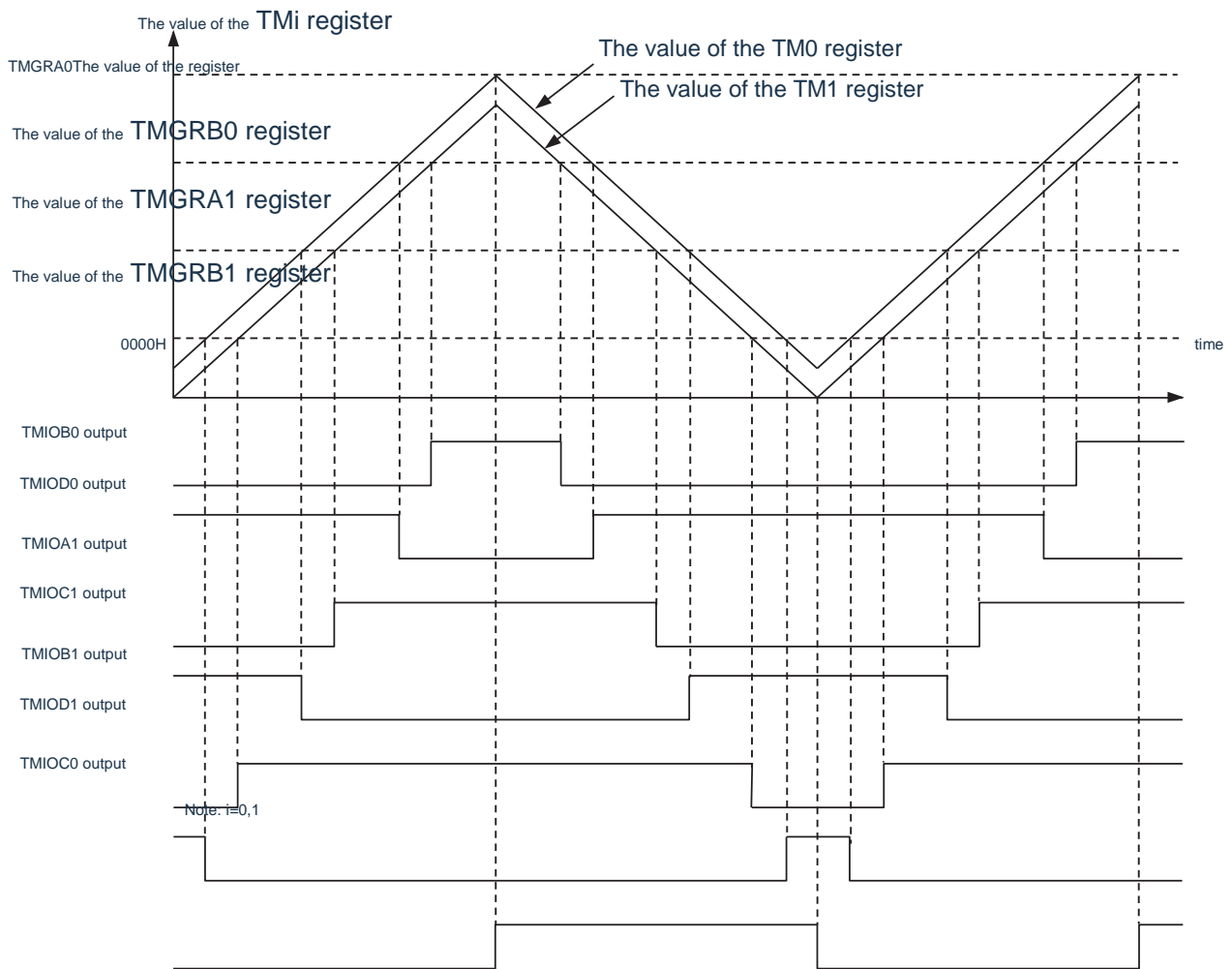
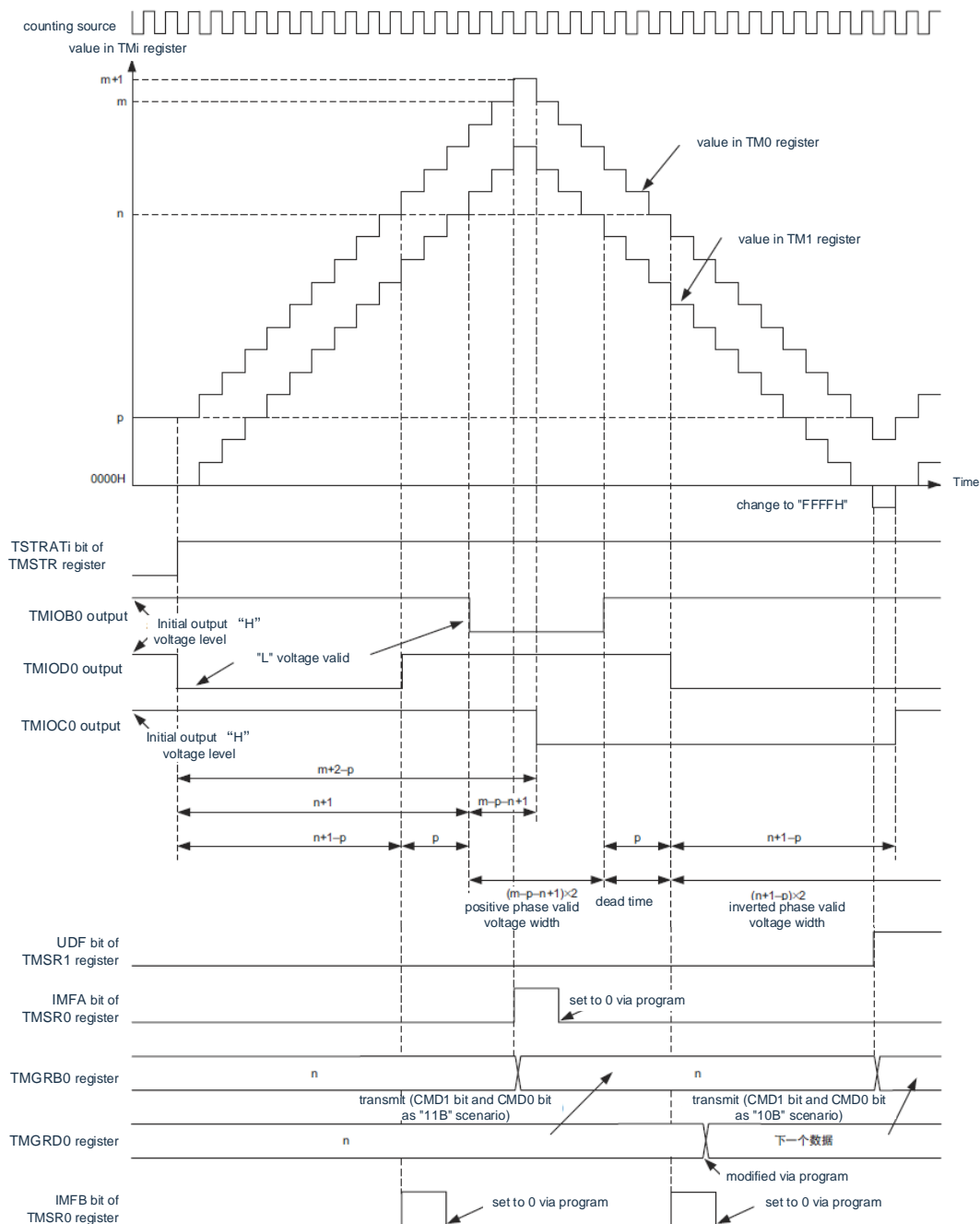


Figure 10-60 operation example of complementary PWM mode



Note: CMD0, CMD1: Bit of the TMFCR register
 $i=0, 1$

m: The setting value of the TMGRA0 register

n: The setting value of the TMGRB0 register

p: The setting value of the TM0 register

The conditions in the above figure are as follows:

The OLS1 bits and OLS0 bits of the TMFCR register are "0" (initial output "H" level, "L" level is active).

(2) Data transfer timing for buffer registers

TMGRD0, TMGRC1, TMGRD1 registers to TMGRB0, TMGRA1, TMGRB1 register data is transferred to TMFCR. The CMD1 bits and CMD0 bits of the registers are "10B" and data transfer occurs when TM1 underflow occurs.

Data transfer occurs when CMD1 and CMD0 bits are "11B" and the TM0 registers and TMGRA0 registers are relatively matched.

10.5.6 PWM3 mode

Outputs 2 PWM waveforms of the same period.

The block diagram and operation example of the PWM3 mode are shown in Figures 10-61 and 10-62, respectively, and the specifications of the PWM3 mode are shown in Table 10-17.

Figure 10-61 Block diagram of PWM3 mode

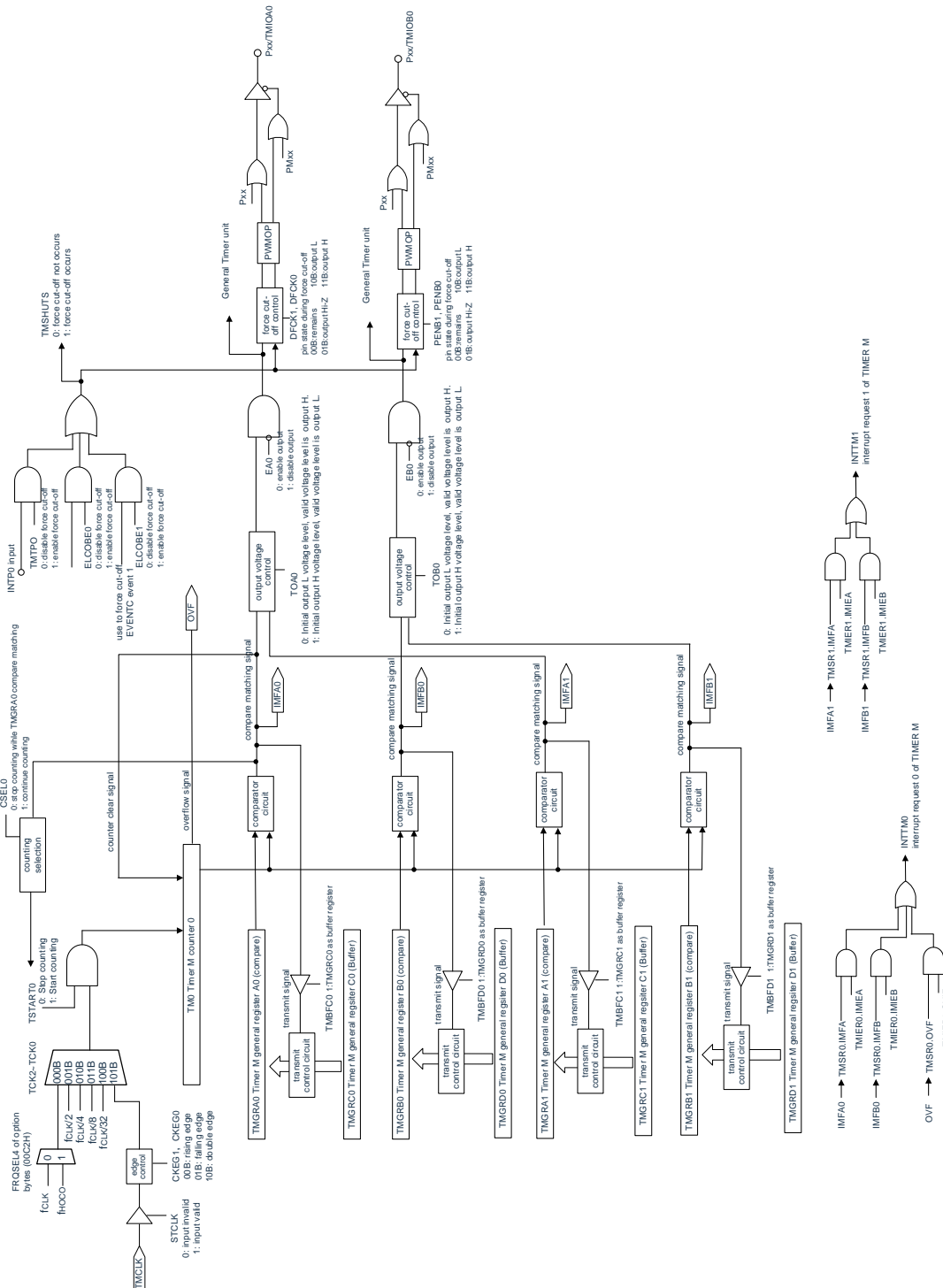
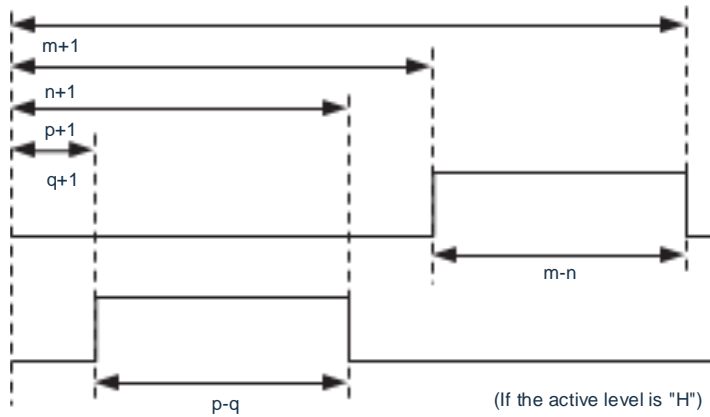


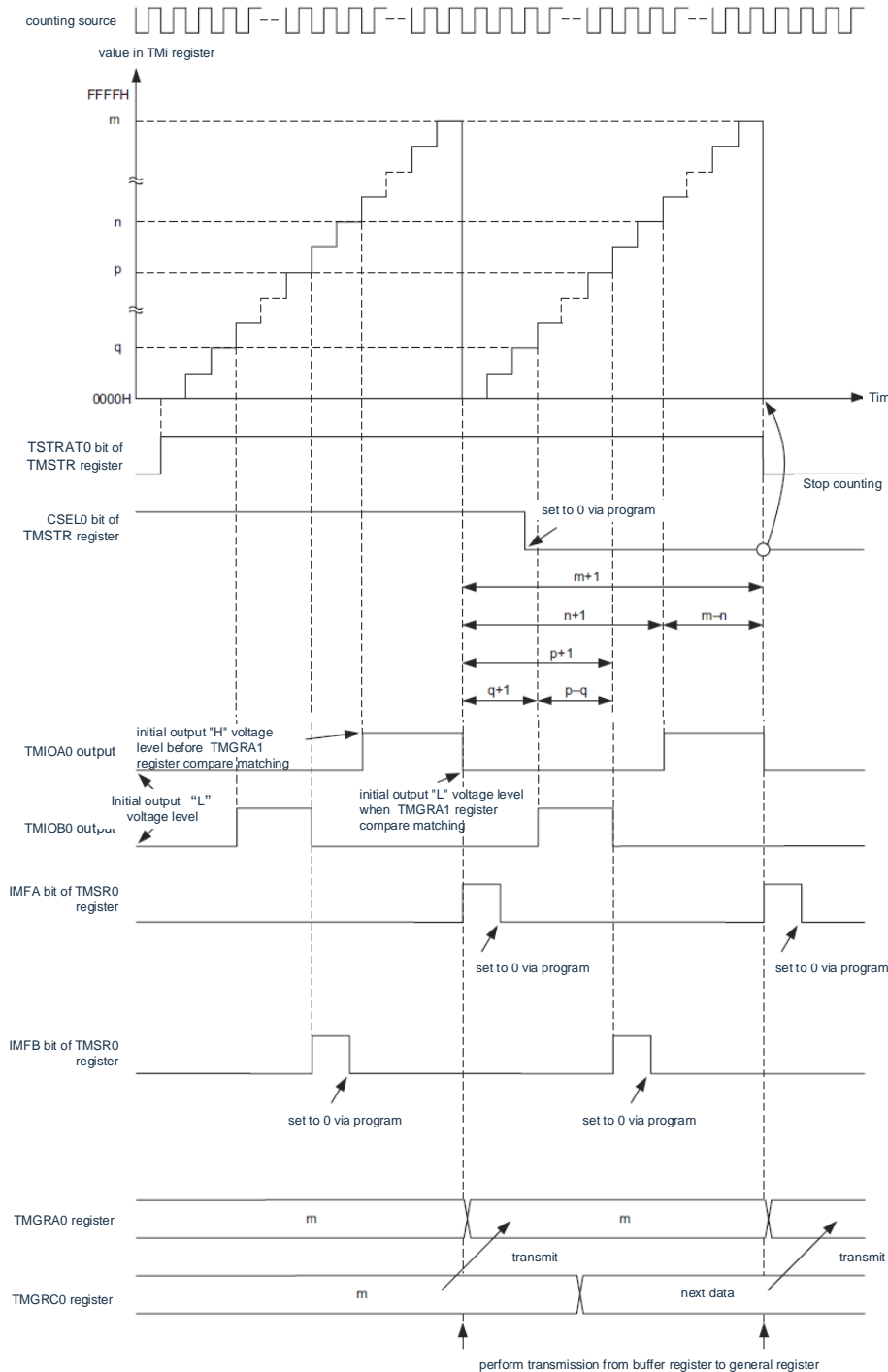
Table 10-17 Specifications for PWM3 mode

project	specification
Count the sources	f_{CLK} , $f_{CLK}/2$, $f_{CLK}/4$, $f_{CLK}/8$, $f_{CLK}/32$
count	TM0 is an incremental count (TM1 is not used).
PWM waveform	<p>PWM period: $1/f_k \times (m+1)$. Effective level width for TMIOA0 output: $1/f_k \times (m-n)$. Effective level width for TMIOB0 output: $1/f_k \times (p-q)$. f_k: The frequency at which the source is counted m: The setting value of the TMGRA0 register n: The setting value of the TMGRA1 register p: The setting value of the TMGRB0 register q: The setting value of the TMGRB1 register</p>  <p>(If the active level is "H")</p>
Count start criteria	Write "1" (start counting) to the TSTART0 bit of the TMSTR register.
Count stop conditions	<ul style="list-style-type: none"> Write "0" (stop count) to the TSTART0 bit when the CSEL0 bit of the TMSTR register is "1". The PWM output pin holds the output level before stopping the count. Stop counting when the CSEL0 bit of the TMSTR register is "0" and a comparison match of TMGRA0 occurs. The PWM output pins remain relatively matched after the output changes.
Timing of the generation of interrupt requests	<ul style="list-style-type: none"> Comparison matching (TMi registers and TMGRji registers have the same content). Overflow of TM0
TMIOA0 pin and The function of the TMIOB0 pin	PWM output
TMIOC0, TMIOD0, TMIOA1~TMIOD1 Pin function	I/O port
INTP0 pin function	The pulse output is the input of the forced cutoff signal (input dedicated port or INTP0 interrupt input).
Read timer	If you read the TM0 register, you can read the count value.
Write timer	Can write TM0 registers.
Select Features	<ul style="list-style-type: none"> Input of the pulse output forced cutoff signal (see "Forced cutoff of 10.4.4 pulse output"). Selection of effective levels for each pin Buffer run (see "10.4.2 buffer run").

Remark i=0, 1,j=A, B, C, D

(1) Operation example

Figure 10-62 Example of operation of the PWM3 mode



Remark:j=A, B

m: The setting value of the TMGRA0 register

n: The setting value of the TMGRA1 register

p: The setting value of the TMGRB0 register

q: The setting value of the T MGRB1 register

The conditions in the above figure are as follows:

TMOCR registers TOA0 and TOB0 are "0" (initial output "L" level, output "H" level when TMGRj1 register comparison matches, output "L" level when TMGRj0 register comparison matches).

The TMBFC0 bit of the TMMR register is "1" (the TMGRC0 register is the buffer register of the TMGRA0 register).

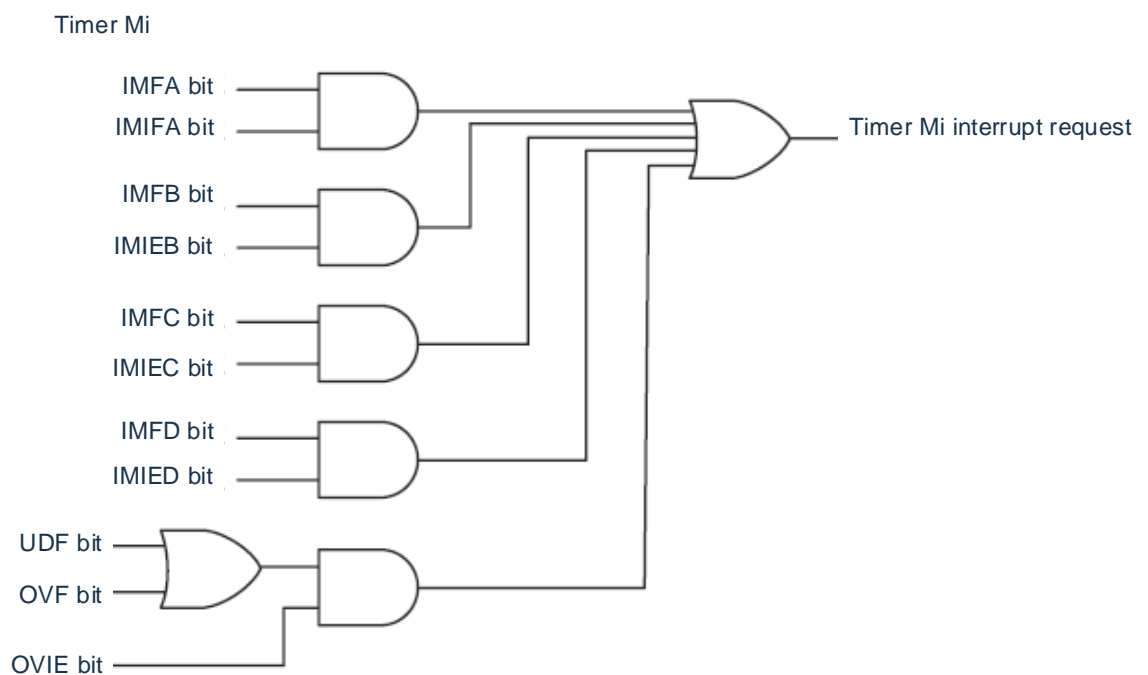
10.6 Timer M interrupt

Timer M generates timer M i ($i=0, 1$) interrupt requests from each of the six interrupt sources of timer M 0 and timer M1. The associated registers for the timer M interrupt are shown in Table 10-18, and the block diagram of the timer M interrupt is shown in Figure 10-63.

Table 10-18 Timer M interrupt related registers

	Status register for timer M	Interrupts of timer M enable registers	Interrupt request flag (Register)	Interrupt mask flag (Register)
Timer M0	TMSR0	TMIER0	TMIF0 (IF2H)	TMMK0 (MK2H)
Timer M1	TMSR1	TMIER1	TMIF1 (IF2H)	TMMK1 (MK2H)

Figure 10-63 Block diagram of timer M interrupt



$i = 0, 1$

IMFA、IMFB、IMFC、IMFD、OVF、UDF: TMSri register bits

IMIEA、IMIEB、IMIEC、IMIED、OVIE: TMIERi register bits

Because timer M generates 1 interrupt request from multiple interrupt request sources (timer M interrupt), there is the following difference between timer RG interrupts and other maskable interrupts:

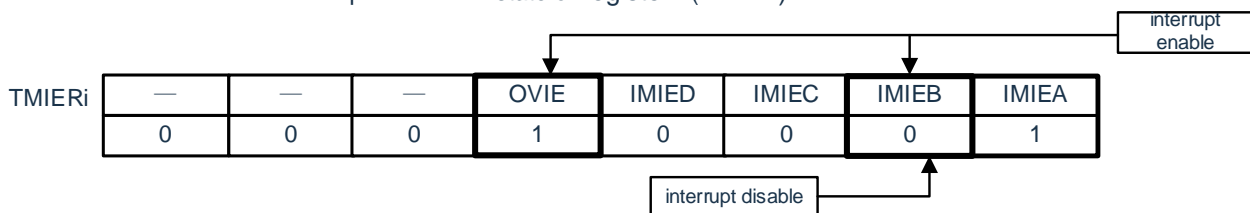
- TMIFi of the IF2H register when the bit of the TMSRi register is "1" and the bit of the corresponding TMEEi register is "1" (interrupt allowed). The bit becomes "1" (with interrupt request).
- When multiple bits of the TMEEi register are "1", the TMSRi register must be used to determine which request source generated the interrupt.
- Because each of the TMSRi registers does not automatically change to "0" even if an interrupt is accepted, these positions must be "0" in the interrupt program.
- When you want to set the status flag (hereinafter referred to as the "object status flag") of one of the interrupt sources of timer M to "0", if the interrupt is allowed by timer M interrupt register i (TMIERi) to disable interrupts, you must use any of the following methods (a) ~ (c) to set "0".

(a) The object status flag must be written "0" after setting the timer M interrupt enable register i (TMIERi) to "00H" (disable all interrupts).

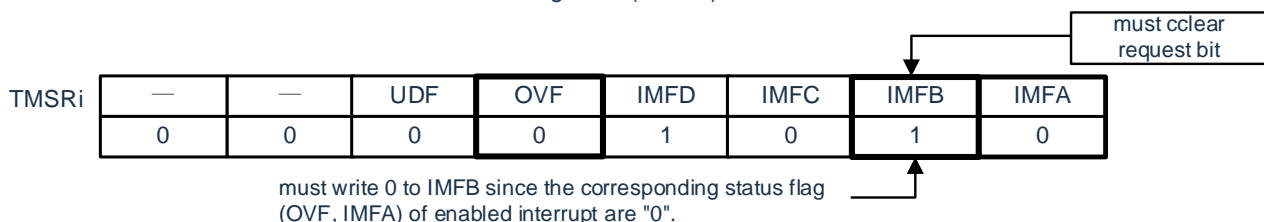
(b) When the timer M interrupt enable register i (TMIERi) has a bit placed "1" (allowed) and the interrupt source status flag allowed by that bit is "0", the object status flag must be written "0".

(e.g.) in the case where IMIEA and OVIE clear IMFB in a state where interrupts are allowed and IMIEB is prohibited

- Timer M interrupt enable the state of register i (TMIERi).



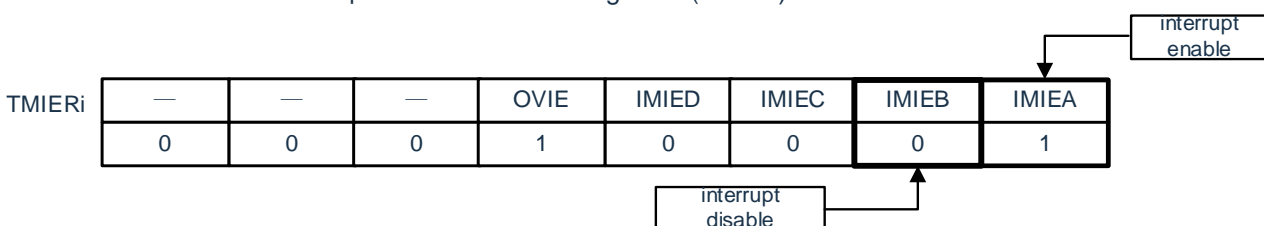
- The status of the timer M status register i (TMSRi).



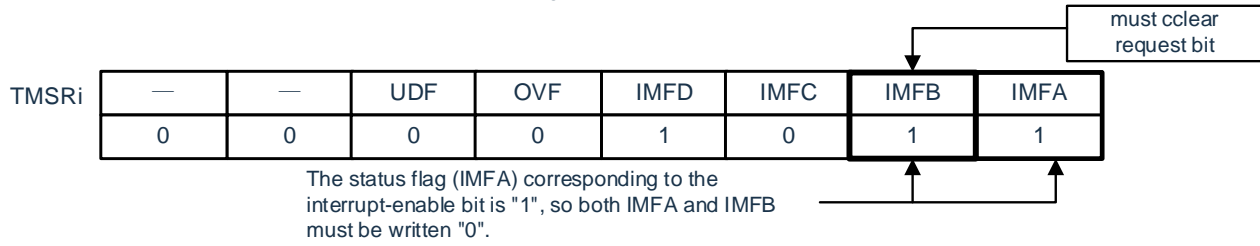
(c) When the timer M interrupt enable a bit of "1" (allowed) in the enable register i (TMIERi) and the bit enable the interrupt source status flag is "1", this state flag and the object status flag must be written "0" at the same time .

(e.g.) when IMIEA clears the IMFB in a state where interrupts are allowed and IMIEB is prohibited

- Timer M interrupt enable the state of register i (TMIERi).



- The status of the timer M status register i (TMSRi).



10.7 Considerations when using timer M

10.7.1 Read and write access to SFR

To set the timer M, you must first place the TMMEN position of the PER1 register "1". When the TMMEN bit is "0", the write operation of the control register of timer M is ignored, and the read values are initial values (except for port registers and port mode registers).

The following registers are prohibited during the counting process:

TMELC registers, TMMR registers, TMPMR registers, TMFCR registers, TMOER1 registers, TMPTO bits, TMDFi registers, TMCRI registers, TMIOARi registers, TMOERARi registers, TMOER2 registers, TMDFi registers, TMD, TM, TMD, TMD, TMIORCi registers, TMPOCRi registers

(1) TMSTR registers

- Ability to set TMSTR registers via 8-bit memory manipulation instructions.
- When the CSELi bits of the TMSTR register (i=0, 1) are "0" (in the Tmi register and TMGRAi register comparison stop count when matching), even if you write "0" (stop count) to the TSTARTi bit, the stop count is not stopped and the TSTARTi bit does not change. The TSTARTi bit becomes "0" (stop count) only when it matches the TMGRAi register.

When rewriting the TMSTR register, if you want to change the CSELi bit to "1" without affecting the count with the CSELi bit being "0", you must give it to TSTARTi Bit writes "0".

If you write "1" to the TSTARTi bit while the counter stops counting, you may start counting. To stop counting programmatically, you must write "0" to the TSTARTi bit after the CSELi position "1". Even if you write "1" and "0" to the CSELi bit and the TSTARTi bit at the same time (using 1 instruction), the count cannot be stopped.

- The output level during the count stop when the TMIOji pin (j=A, B, C, D) is used for the timer M output as shown in Table 10-19 as shown.

The output levels of the TMIOji pins (j=A, B, C, D) when Table 10-19 stop counts

The method that stops counting	Stop counting the output level of the TMIOji pin
When the CSELi bit is "1", stop counting by writing "0" to the TSTARTi bit.	Maintain the output level before stopping counting (in complementary PWM mode of timer M or reset synchronous PWM mode, the initial output level of the OLS0 bit of the output TMFCR register and the OLS1 bit of
When the CSELi bit is "0", the count stops when the Tmi registers and the TMGRAi registers are relatively matched.	Keep the comparison matching level after the output change is caused (in the complementary PWM mode of timer M or the reset synchronous PWM mode, the initial output level of the OLS0 bit of the output TMFCR register

Remark i=0, 1, j=A, B, C, D

(2) TMDFi registers (i=0, 1).

Counting must begin after setting the DFCK0 and DFCK1 bits of the TMDFi register.

(3) Tmi registers (i=0, 1).

If the value of the Tmi register changes to "0000H" and the timing of the write Tmi register overlaps, the register is preferred.

10.7.2 Switching modes

- To switch modes during operation, it must be switched after entering the count stop state (TSTART0 bit and TSTART1 position "0").
- Before changing the TSTART0 bit and TSTART1 bit from "0" to "1", the TMIF0 bit and TIF1 bits must be in the position "0". For details, please refer to "Chapter 25 Interrupt Function".

10.7.3 Counting sources

- To switch the counting source, you must switch after stopping counting.
[Change Step].
 - (1) Place the TSTARTi bits (i=0, 1) of the TMSTR register to "0" (stop count).
 - (2) Change the TCK0~TCK2 bit of the TMCRi register.
- To select f_{HOCO} (64MHz or 48MHz) as the counting source for timer M, register 1 (PER1) must be allowed on the Peripheral Set f_{CLK} to f_{IH} before bit4 (TMMEN) is set. If you want to change the f_{CLK} to a clock other than f_{IH} , you must allow bit4 of register 1 (PER1) in the Peripheral to be cleared (TMMEN) after making changes.

10.7.4 Enter the capture function

- The pulse width of the input capture signal must be at least 3 timer M operating clock cycles.
- After the input capture signal is input from the TMIOji pin (j=A, B, C, D), you need to wait 2~ The operating clock (f_{CLK}) cycle of the 3 timers M is then passed to the TMGRji register (in the absence of a digital filter).
- In input capture mode, if the TMTSTARTi bit of the TMSTR register is "0" (stop count), the TMIOji pin is input to TMIORki. The IOj0 bit of the register and the edge of the IOj1 bit selection generate an input capture interrupt request (i=0, 1) at the valid edge of the TMIOji input, j=A, B, C, D, k=A, C).

10.7.5 Configuration steps (i=0, 1) for TMIOAi, TMIOBi, TMIOCi, TMIODi pins

After reset, the multiplexed I/O ports of the TMIOAi, TMIOBi, TMIOCi, TMIODi pins are used as input ports.

- To output from the TMIOAi, TMIOBi, TMIOCi, TMIODi pins, you must follow the steps below to set them.
[Change Step].
 - (2) Set the mode and initial values.
 - (3) Place the TMUAI, TMIOBi, TMIOCi, TMIODi pins as allowed outputs (TMOER1 registers).
 - (4) Place the port registers at "0" corresponding to the TMIOAi, TMIOBi, TMIOCi, and TMIODi pins.
 - (5) Set the bit of the port mode register corresponding to the TMIOAi, TMIOBi, TMIOCi, TMIODi pins to output mode (from TMIOAi, TMIODi, TMIOBi, TMIOCi, TMIODi pins start output).
 - (6) Start counting (place TSTART0 bits and TSTART1 at position "1").
- To change the bits of the port mode registers corresponding to the TMUAI, TMIOBi, TMIOCi, and TMIODi pins from output mode to input mode, you must follow the steps below to set them.
[Change Step].
 - (1) Set the bit of the port mode register corresponding to the TMIOAi, TMIOBi, TMIOCi, TMIODi pins to input mode (from TMIOAi, TMIOBi, TMIOCi, TMIODi pin start input).
 - (2) Set to input capture function.
 - (3) Start counting (place TSTART0 bits and TSTART1 at position "1").
- When switching the TMIOAi, TMIOBi, TMIOCi, TMIODi pins from output mode to input mode, input capture operation may be performed depending on the state of the pins. Edge detection after at least 2 CPU clock cycles when no digital filter is used; When using digital filters, up to 5 sampling clock cycles of digital filters are required for edge detection.

10.7.6 External clock TMCLK

The pulse width of the external clock input to the TMCLK pin must be at least 3 timer M operating clock cycles.

Reset synchronous PWM mode

- When this mode is used for motor control, it must be used under the condition of OLS0=OLS1.
- To set the reset sync PWM mode, you must follow the steps below to set it up. [Change Step].
 - (1) Place the TSTART0 position of the TMSTR register "0" (stop count).
 - (2) Place the CMD1 bit and CMD0 position "00B" of the TMFCR register (timer mode, PWM mode, and PWM3 mode).
 - (3) Place CMD1 bit and CMD0 position "01B" (reset synchronous PWM mode).
 - (4) Reset the other associated registers for timer M.

10.7.7 Complementary PWM mode

- When this mode is used for motor control, it must be used under the condition of OLS0=OLS1.
- To change the CMD0 bits and CMD1 bits of the TMFCR registers, you must follow the steps below to make the changes.

[Change step: when set to complementary PWM mode (including resetting), or change the data transfer timing of buffer registers to General Purpose registers in complementary PWM mode].

- (1) Both the TSTART0 bits and the TSTART1 bits of the TMSTR register are set to "0" (stop count).
- (2) Place the CMD1 bit and CMD0 position "00B" of the TMFCR register (timer mode, PWM mode, and PWM3 mode).
- (3) Place CMD1 bit and CMD0 at "10B" or "11B" (complementary PWM mode).
- (4) Reset the other associated registers for timer M.

[Change Step: Case of Discontinuation of Complementary PWM Mode].

- (1) Both the TSTART0 bits and the TSTART1 bits of the TMSTR register are set to "0" (stop count).
- (2) Place cmd1 bit and CMD0 position "00B" (timer mode, PWM mode, and PWM3 mode).

- TMGRA0, TMGRB0, TMGRA1, TMGRB1 registers cannot be written during operation. To change the PWM waveform, the write values of the TBMGRD0, TMGRC1, and TMGRD1 registers must be transmitted to TMGRB0 via buffer operation TMGRA1, TMGRB1 registers. However, when writing TMGRD0, TMGRC1, and TMGRD1, you must first change the TMBFD0 bit, TMBFC1 Bits and TMBFD1 position "0" (General Purpose registers) and then write data to these registers. Thereafter, TMBFD 0 bits, TMBFC1 bits, and TMBFD1 positions "1" (buffer registers) can be changed. The PWM period cannot be changed.
- Assuming that the setting value of the TMGRA0 register is m, the TM0 register performs $m-1 \rightarrow m \rightarrow$ when changing from an increasing count to a decrement count
 Count of $m+1 \rightarrow m \rightarrow m-1$.
 When performing an increment count of $m \rightarrow m+1$, the IMFA bit of the TMSRi register becomes "1". When the CMD1 bit and CMD0 bits of the TMFCR register are "11B" (complementary PWM mode, at TM0 When the buffer data is transmitted when matching the TMGRA0 register), the contents of the buffer register (TMGRD0, TMGRC1, TMGRD1) are transmitted to the general-purpose register (TMGRD0, TMGRD1, TMGRD1). TMGRB0, TMGRA1, TMGRB1) .
 When performing a decrement count of $m+1 \rightarrow m \rightarrow m-1$, the IMFA bits do not change and data is not transmitted to registers such as TMGRA0.

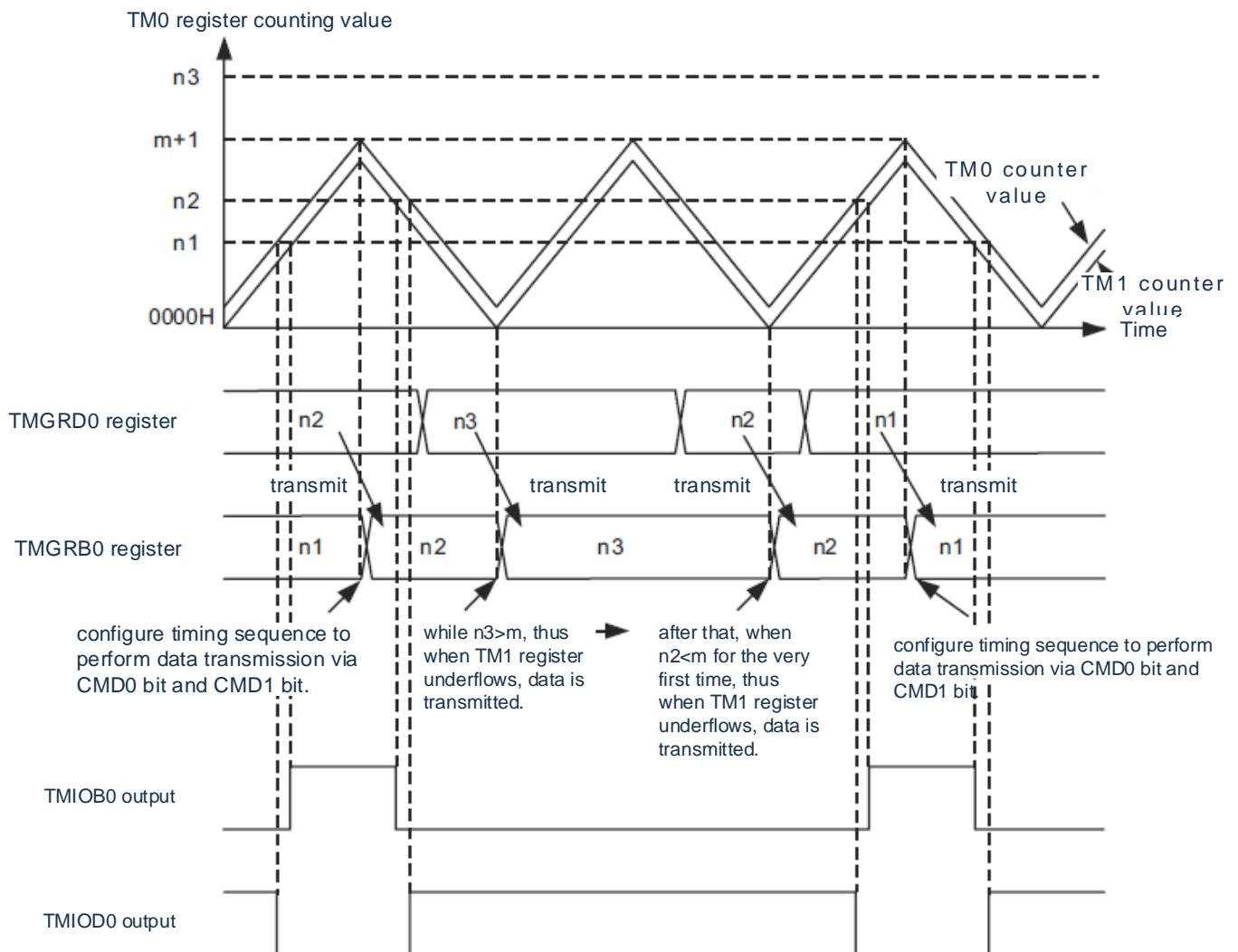
- The data transfer timing from the buffer register to the general-purpose register must be selected by the CMD0 bit and CMD1 bit of the TMFCR register. However, in the case of 0% duty cycle and 100% duty cycle, regardless of the values of CMD0 bits and CMD1 bits, the following transmission timing is the following.

The value of the buffer register \geq the value of the TMGRA0 register (0% duty cycle) data transfer occurs when the TM1 register underflow occurs.

Thereafter, if you set a value for the buffer register ($0001H \leq \text{set the value} < \text{TMGRA0Register value}$), just after setting TM1Register occurrence 1 Transmits data to a general purpose register on the second underflow. Then, pass CMD0Bit sum CMD1The bit selects the time series for data transfer.

However, a waveform with a 0% duty cycle cannot be generated when the initial value of the buffer register is "FFFFH". To generate a waveform with a 0% duty cycle, the value of the buffer register must be \geq the value of the TMGRA0 register by writing the buffer register.

Figure 10-66 Example of operation when the value of the buffer register in the complementary PWM mode \geq the value of the TMGRA0 register



If you set a value for the buffer register (the setting value \geq the value of the TMGRA0 register), the value of the buffer register is passed to the general purpose register just when the TM1 counter underflows, and it is fixed to the output level of 100% positive phase duty cycle and inverting 0% duty cycle The setting of cmd0 bits is independent.

To desetting the output level, the buffer register must be set (the value of the TM0 register \leq the value of the \leq (the value of the TMGRA0 – the value of the TM0 register)). After writing the buffer, regardless of the setting of the CMD0 bit, the value of the buffer register is transferred to the general-purpose

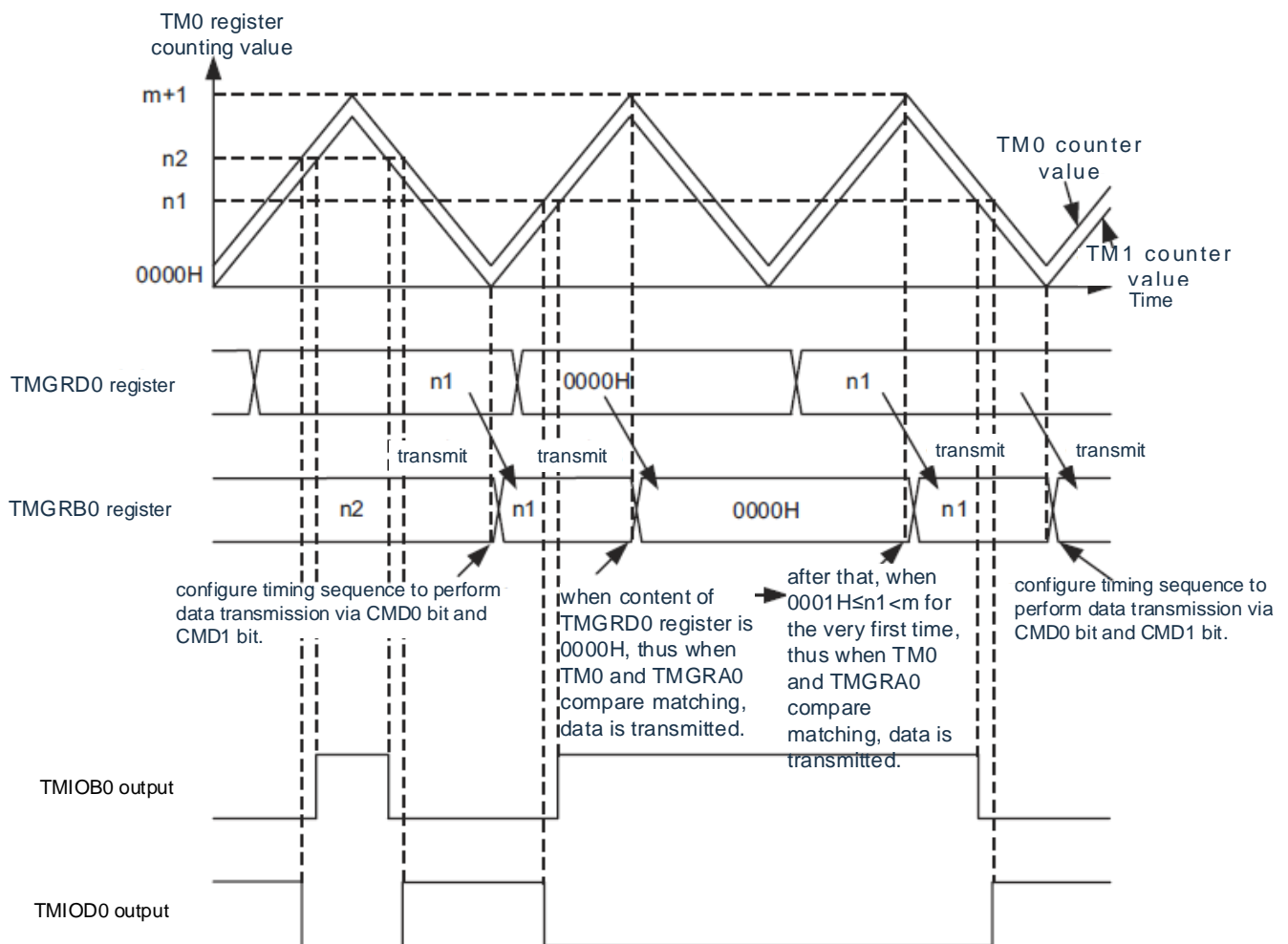
register and the PWM waveform is output when the TM1 counter underflow occurs. After the PWM waveform is output, the value of the buffer register is transmitted to the general-purpose register via the timing set by CMD0 bits.

However, the output of 100% positive phase and 0% duty cycle of inverting phase cannot be set with the initial value of the buffer register "FFFFH". Nor can it be directly changed from the output of 100% positive phase duty cycle and inverting 0% duty cycle to the output of positive phase 0% duty cycle and inverting phase 100% duty cycle.

The case where the value of the buffer register is "0000H" (100% duty cycle) data transfer occurs when the TM0 and TMGRA0 registers are relatively matched.

Thereafter, if you set a value for the buffer register ($0001H \leq \text{set the value} < \text{TMGRA0Register value}$), just after setting TM0 and TMGRA0 Register occurrence 1 When the second comparison matches, the data is transferred to a general purpose register. Then, pass CMD0Bit sum CMD1The bit selects the time series for data transfer.

Figure 10-67 Example of operation when the buffer register in complementary PWM mode has a value of "0000H"



If you write "0000H" to the buffer register, the value of the buffer register is passed to the general-purpose register when the TM0 register and the TMGRA0 registers are relatively matched, and fixed to a positive phase 0% duty cycle and an inverting 100%. The output level of the duty cycle, independent of the setting of the CMD0 bit.

To deserring the output level, the buffer register must be set (the value of the TM0 register \leq the value of the \leq (the value of the TMGRA0 – the value of the TM0 register)). After writing the buffer, regardless of the setting of the CMD0 bit, the value of the buffer register is transferred to the general-purpose register and the PWM waveform is output when the TM1 counter underflow occurs. After the PWM

waveform is output, the value of the buffer register is transmitted to the general-purpose register via the timing set by CMD0 bits.

You cannot directly change from the output of 0% positive phase duty cycle and 100% inverting phase to the output of 100% duty cycle of positive phase and 0% duty cycle of inverting phase.

10.8 PWMOP

The PWMOP unit can implement the output forced cutoff function of the Timer M. The cutoff source can be selected from CMP0, INTP0, and EVENT. This is different from timer M's own pulse forced cutoff function.

Table 10-20 Forced cutoff of pulse output and functional comparison of PWMOP

	Forced cutoff of the TimerM pulse output	The output of the PWMOP is forced to cut off
Supports a pattern of forced cutoffs	PWM function reset synchronous PWM mode complementary PWM mode PWM3 mode	All mode port outputs that support TimerM can also be forced to cut off
The source of the forced cutoff	EventC event input INTP0 input	Event input INTP0 for EVENTC is input to the output of comparator 0
Force the source of the cutoff to be released	Stop the counter and release it by software	Hardware de-software release (no need to stop counter)
Pins that can be forced to cut off	Pxx/TMIOA0,Pxx/TMIOB0,Pxx/TMIOC0,Pxx/TMIOD0,Pxx/TMIOA1,Pxx/TMIOB1,Pxx/TMIOC1,Pxx/TMIOD1. (Based on TimerM output settings)	Pxx/TMIOA0,Pxx/TMIOB0,Pxx/TMIOC0,Pxx/TMIOD0,Pxx/TMIOA1,Pxx/TMIOB1,Pxx/TMIOC1,Pxx/TMIOD1. The output of a PORT can also be forced to cut off
The PORT status at the time of the forced cutoff	HI-ZL level H level	HI-ZL level H If you select forced cutoff of the PORT output, only HI-Z can be output

Note: When using both pulse force cutoff and output force cut off function, you cannot select the same source.

10.8.1 Features of PWMOP

PWMOP can implement the following functions:

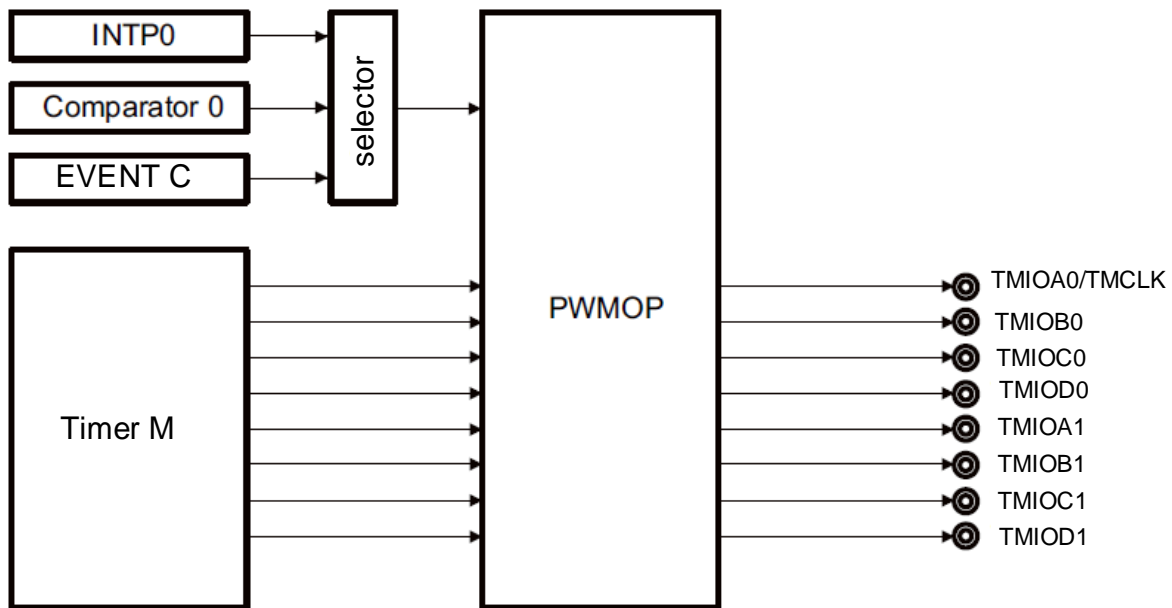
The output of comparator 0, the INTP0 input, and the event input of EVENTC can be selected as output force cutoff sources.

When selecting the output of comparator 0 and the input of INTP0 as the source, you can select the edge checkout.

The output can be depressed by software or hardware.

When forced cutoff, the output level can be selected for "H" level, "L" level, and Hi-Z.

Figure 10-68 Block diagram of PWMOP



10.8.2 Registers for PWMOP

The registers of the PWMOP are shown in Table 10-21.

Table 10-21 Control registers for PWMOP

Register name	symbol
PWMOP control register 0	OPCTL0
PWMOP forces a cut-off control register of 0	OPDF0
PWMOP forces a cutoff control register 1	OPDF1
PWMOP along the selection register	OPEDGE
PWMOP status register	OPSR

(1) PWMOP control register 0 (OPCTL0).

Figure 10-69 Format of PWMOP control register 0

Address: 0x40043C58 after reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
OPCTL0	0	HAZAD_SET	IN_EG	IN_SEL1	IN_SEL0	ACT	HZ_REL	HS_SEL

HAZAD_SET	Output forced cut-off hazard control Note 1
0	Prohibition of hazard countermeasures
1	Enable for hazard countermeasures

IN_EG	Selection of source/output force cutoff release source edge Note 2,3
0	Rising Edge: Output Forced Cutoff Descending Edge: Output Forced Cutoff Is Lifted
1	Rising Edge: Output Forced Cutoff Release Falling Edge: Output Force Cutoff

IN_SEL1	IN_SEL0	Choice of cut-off source Note 2, 4, 5
0	0	Do not select
0	1	The output of comparator 0
1	0	INTP0 input
1	1	Event input from EVENTC

ACT	Software deserialization selection when software is released
0	When the HZ_SEL is set to 1 by software, the forced cutoff is lifted and the pulse output is restored
1	When the HZ_SEL is 1, the forced cutoff is released, and the timing of the pulse output recovery is as follows: Timer M Complementary PWM mode: The edge of TMIOC0 selected by OPEDGE is deforced cutoff, the pulse output recovery timer M reset synchronous PWM mode: When the count value of TM0 is 0000H, the pulse output recovers the above mode: When the count value of TM0 is 0000H, TMIOj0 (j= A, B, C, D) When the count value of the forced cut-off TM1 is 0000H, TMIOj1 (j=A, B, C, D) forced cut-off release. Note 6

HZ_REL	When the software is dismissed: Output forced cut-off to de-control Note 7
0	Keep output force cutoff (when output force cutoff is dismissed, HZ_REL bit becomes 0)
1	Forced cutoff is lifted, pulse output resumes Note 8

The readout value of HZ_REL bit varies depending on the state: usually when the state: 1/0 is written, and when the 0 readout output is forced to cut off: only 1 can be written and 1 can be read out.

HS_REL	Outputs a mode selection for forced cutoff
0	Hardware release: When using hardware to deactivate the output, the timing is different depending on the action mode of timer M. Timer M complementary PWM mode: After monitoring the release source, the edge of TMIOC0 selected according to OPEDGE is decommissioned. Timer M reset synchronous PWM mode: when the count value of TM0 is 0000H, when the count value of pulse output recovery TM0 is 0000H, TMIOj0 (j=A, B, C, D) forced cut-off off TM1 is 0000H, TMIOj1 (j=A, B, C, D) forced cut-off release. <small>Note 9</small>
1	Software dismissal

Note 1: Timer M cannot be rewritten during the M action

Note 2: Set the IN_EG at least three clocks apart, and then set the IN_SEL1 and the IN_SEL0.

Note 3: Enabled when comparator 0 output and INTP0 input are selected.

Note 4: When using EVENTC to desist the deadline, software decommission must be selected (HS_SEL set to 1). There are no limitations when using comparator 0 output and INTP0 input.

Note 5: Selecting the effective width of the comparator 0 output and the INTP0 input must be greater than one clock cycle.

Note 6: The count value of TM0 and TM1 = 0000H refers to the moment when the counter bit15~bit0 is all 0 during the operation of TM0 and TM1.

Note 7: If timer M is operating in output comparison function, PWM function, or PWM3 mode, using 2 channels and using 1 channel, the deactivation action of the cutoff output is different.

When using 2 channels:

If the HZ_REL is set to 1 by software, the output cutoff bit (HZOF0, HZOF1) all becomes 0, and the HZ_REL bit also becomes 0.

When using 1 channel:

If the HZ_REL is set to 1 by software, the output cutoff of the currently used channel bit (HZOF0 or HZOF1) becomes 0, and the HZ_REL bit also becomes 0.

Note 8: If no mandatory cutoff occurs, it cannot be set to 1

Note 9: If timer M is operating in output comparison function, PWM function, or PWM3 mode, when the forced cutoff is lifted, the channel without action cannot cancel the forced output cutoff state. (HZOF0 and HZOF1 do not become 0).

(2) PWMOP Forced CutOff Control Register 0 (OPDF0).

Figure 10-70 PWMOP Forced Cutoff Control Register 0 Format

Address: 0x40043C59 after reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
OPDF0	DFD01	DFD00	DFC01	DFC00	DFB01	DFB00	DFA01	DFA00

DFD01	DFD00	TMIOD0 pin output forced cutoff control
0	0	Disables the forced cutoff feature
0	1	Output Hi-Z
1	0	Output L level
1	1	Output H level

DFC01	DFC00	TMIOC0 pin output forced cutoff control
0	0	Disables the forced cutoff feature
0	1	Output Hi-Z
1	0	Output L level
1	1	Output H level

DFB01	DFB00	TMIOB0 pin output forced cutoff control
0	0	Disables the forced cutoff feature
0	1	Output Hi-Z
1	0	Output L level
1	1	Output H level

DFA01	DFA00	TMIOA0 pin output forced cutoff control
0	0	Disables the forced cutoff feature
0	1	Output Hi-Z
1	0	Output L level
1	1	Output H level

Note 1: The Hi-Z output must be selected when the TMIOj0 (j=A, B, C, D) pins are used as PORT outputs and the forced cutoff function is enabled.

Note 2: The value of the register cannot be changed in the forced cutoff state.

Note 3: When using PIOR to redirect TMIOji (j=A, B, C, D; i=0,1) pin, only a single setting can be made for the same pin.

Example: When using PIOR2 to select P17 as the TMIOD0 pin and allow the TMIOD0 output, DFA0n (n=0,1) must be set to 0 (disables the forced cutoff function).

(3) PWMOP Forced Cutoff Control Register 1 (OPDF1).

Figure 10-71 Format of PWMOP Forced Cutoff Control Register 1

Address: 0x40043C5A after reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
OPDF1	DFD11	DFD10	DFC11	DFC10	DFB11	DFB10	DFA11	DFA10

DFD11	DFD10	TMIOD1 pin output forced cutoff control
0	0	Disables the forced cutoff feature
0	1	Output Hi-Z
1	0	Output L level
1	1	Output H level

DFC11	DFC10	TMIOC1 pin output forced cutoff control
0	0	Disables the forced cutoff feature
0	1	Output Hi-Z
1	0	Output L level
1	1	Output H level

DFB11	DFB10	TMIOB1 pin output forced cutoff control
0	0	Disables the forced cutoff feature
0	1	Output Hi-Z
1	0	Output L level
1	1	Output H level

DFA11	DFA10	TMIOA1 pin output forced cutoff control
0	0	Disables the forced cutoff feature
0	1	Output Hi-Z
1	0	Output L level
1	1	Output H level

Note 1: The Hi-Z output must be selected when the TMIOj1 (j=A, B, C, D) pin is used as a PORT output and the forced cutoff function is enabled.

Note 2: The value of the register cannot be changed in the forced cutoff state.

Note 3: When using PIOR to redirect TMIOj1 (j=A, B, C, D; i=0,1) pin, only a single setting can be made for the same pin.

Example: When using PIOR2 to select P16 as the TMIOA1 pin and allow TMIOA1 output, DFC1n (n=0,1) must be set to 0 (forced cutoff function is disabled).

(4) PWMOP edge selection register (OPEDGE).

When timer M is operating in complementary PWM mode and a hardware de-output is forced to cut off, the deregistration point can be set via the OPEGE register.

Figure 10-72 the format of PWMOP edge selection register

Address: 0x40043C5B after reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
OPEGE	-	-	-	-	-	-	-	EG1 EG0

EG1	EG0	Output force cutoff to dismiss the edge selection
0	0	The rising edge of TMIOC0 is lifted
0	1	The descending edge of TMIOC0 is lifted
1	0	The rising or falling edge of TMIOC0 is lifted
1	1	Do not enable along the selection, keep the cutoff

(5) PWMOP Status Register (OPSR).

Figure 10-73 Format of the PWMOP status register

Address: 0x40043C5C after reset: 00H R

Symbol	7	6	5	4	3	2	1	0	
OnSR	0	0	0	0	0	0	0	0	<div> <div> HZOF1 </div> <div> HZOF0 </div> <div> HZIF0 </div> </div>

HZOF1	Mandatory cut-off status
0	Normal Timer output (TMIOA1, TMIOB1, TMIOC1, TMIOD1)
1	Force cutoff state (TMIOA1,TMIOB1,TMIOC1,TMIOD1)

HZOF0	Mandatory cut-off status
0	Normal Timer output (TMIOA0, TMIOB0, TMIOC0, TMIOD0)
1	Force cutoff state (TMIOA0,TMIOB0,TMIOC0,TMIOD0)

HZIF0	Output Force Cutoff Source Status Note 1,2
0	The output force cutoff source is within the threshold range
1	The output force cutoff source is out of the threshold range

Note 1: Before selecting the INTP0 input by setting the IN_SEL1, IN_SEL0, comparator 0 output as the forced cutoff factor, if the output force cutoff source is outside the threshold range, N_SEL1 and IN_SEL0 When set, HZIF0 bit will be set to 1, while HZOF0 and HZOF1 will not be set.

Note 2: Valid only when the INTP0 input and comparator 0 output are selected.

10.8.3 Operation of PWMOP

The output of comparator 0, the INTP0 input, and the event input of EVENTC can be selected as output force cutoff sources.

When selecting the output of comparator 0 and the input of INTP0 as the source, you can select the edge checkout.

10.8.3.1 Output forces cutoff

The output of comparator 0, the INTP0 input, and the event input of EVENTC can be selected as trigger events to force the cutoff TMIOj(j=A, B, C, D; i=0,1) of the output.

When a trigger event is detected, the output of timer M is forced to cut off and the setting value of register OPDF0/OPDF1 is output. For detailed actions, see Figure 10-7 5.

The HS_SEL bit of the OPCTL0 registers allows you to choose to remove the forced cutoff function by software or hardware.

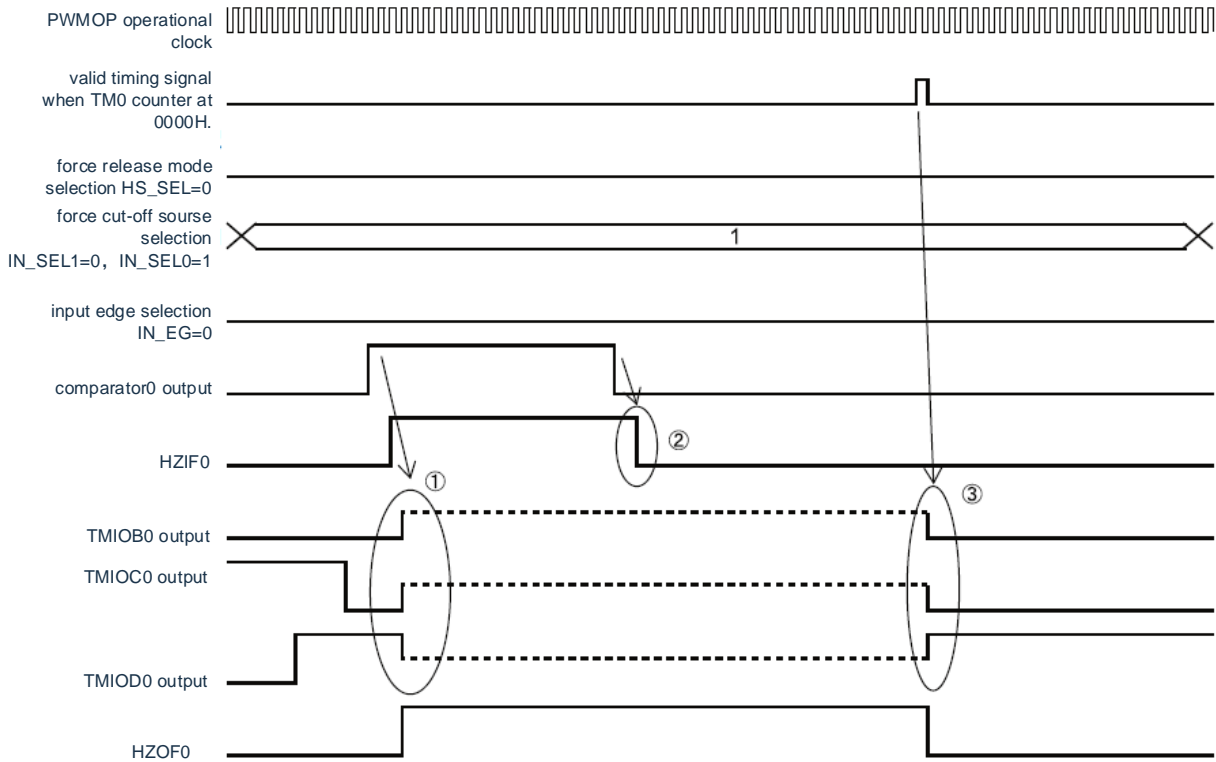
10.8.3.2 Hardware release (HS_SEL=0).

When timer M works in different modes, the de-timing is different:

(1) Outputs outside of complementary PWM functions

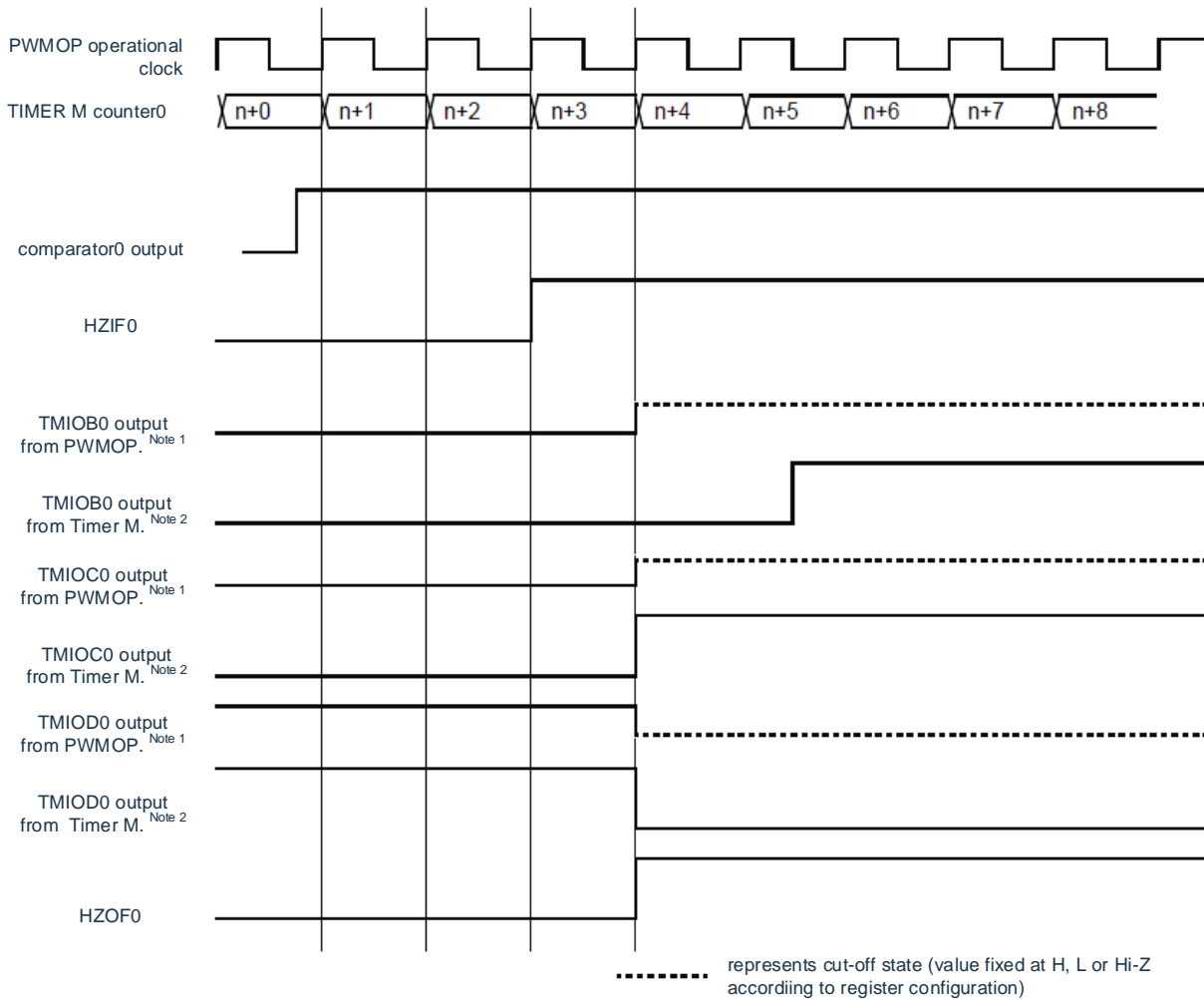
- When timer M operates in output comparison function, PWM function, or PWM3 mode, after the deserialization event occurs, when the count value of TM0 is 0000H, the forced cut-off of TMIOj0 (j=A, B, C, D) is lifted. When the count value of TM0 is 0000H, the forced cutoff of TMIOj1 (j=A, B, C, D) is lifted.
- When timer M is operating in reset PWM mode during the same period, after the de-event occurs, when the count value of TM0 is 0000H, the forced cutoff of all T MIO pins is completely released.

Figure 10–74 Example of output force cutoff/hardware de-output force cutoff
(Example of TMIOB0, TMIOC0, TMIOD0 pin being forced to cut off).



- (1) When the rising edge of the comparator output signal is detected, the TMIOB0, TMIOC0, TMIOD0 pin outputs are forced to cut off.
- (2) After the falling edge of the comparator output signal is detected, the HZIF0 bit is cleared.
- (3) At a TDi count value of 0000H, the cutoff state is forced to be dismissed.

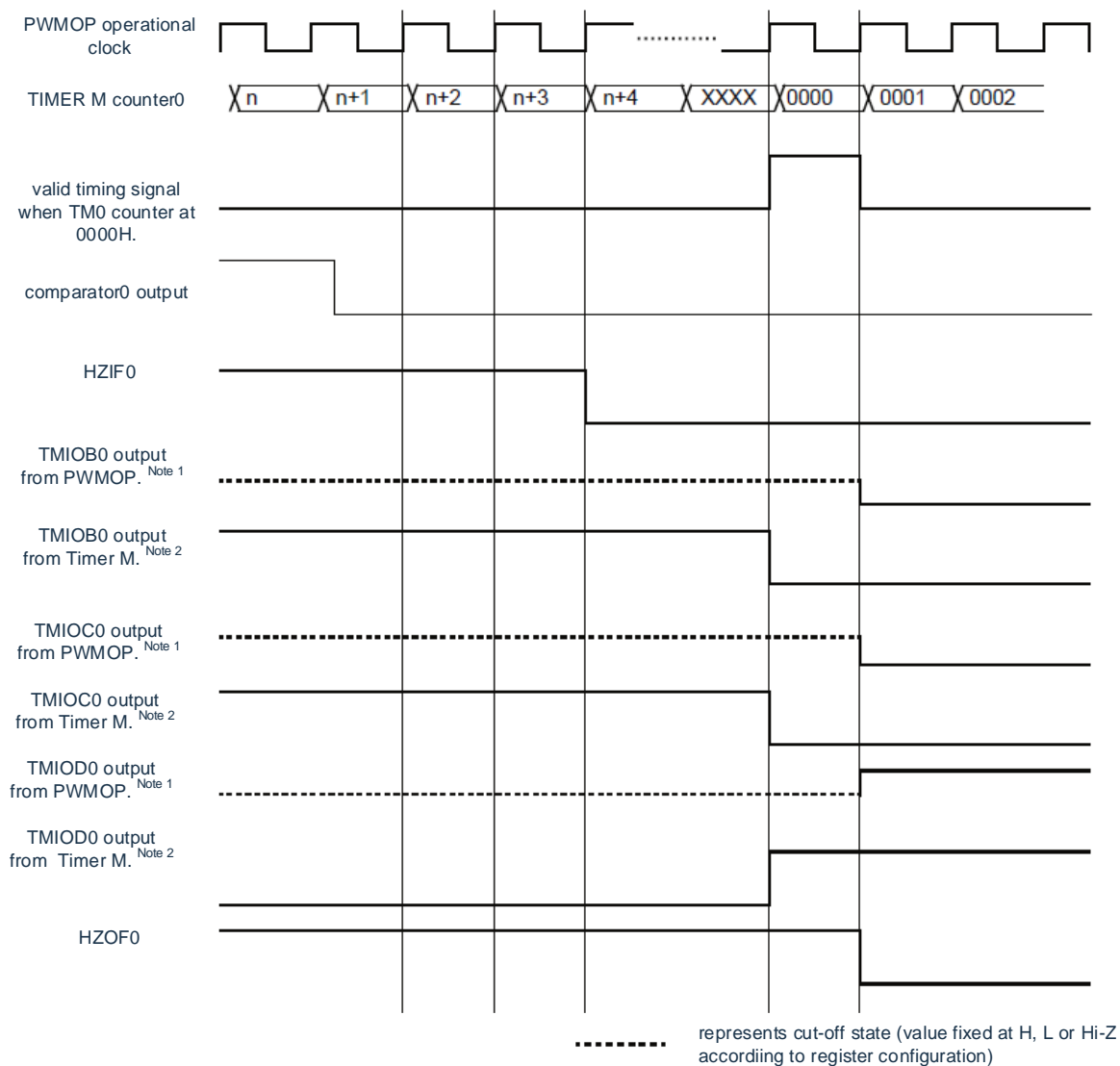
Figure 10–75 Detailed timing diagram of the forced cutoff



Note 1: The TMIO* (*=B~D) output from PWMOP indicates the state of the pin that reuses the M function of the timer.

Note 2: TMIO* (*=B~D) output from Timer M indicates the input signal state from the timer M output to PWMOP.

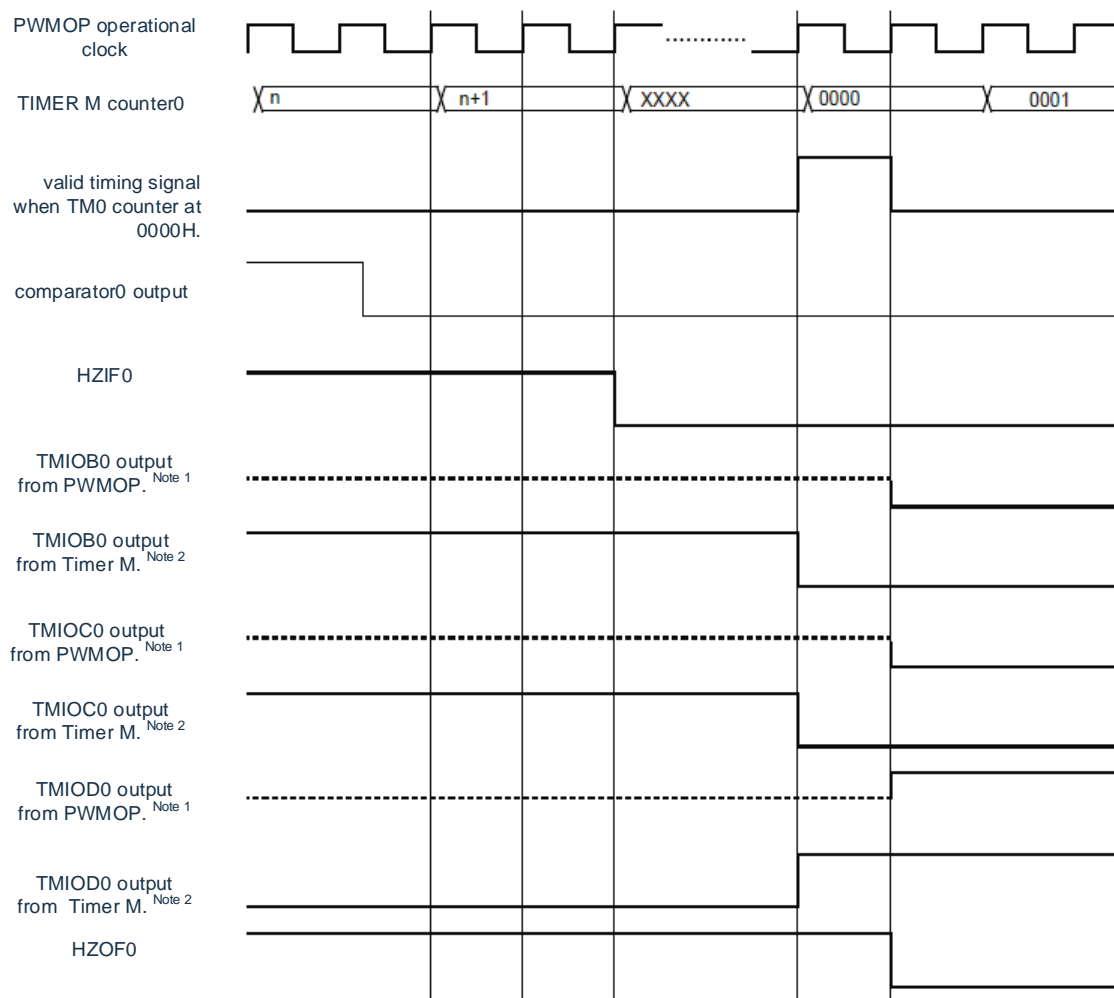
Figures 10–76: Detailed timing diagram of forced cutoff release (count source for timer M = Fclk).



Note 1: The TMIO* (*=B~D) output from PWMOP indicates the state of the pin that reuses the M function of the timer.

Note 2: TMIO* (*=B~D) output from Timer M indicates the input signal state from the timer M output to PWMOP.

Figure 10–77: Detailed timing diagram of forced cut-off release (count source of timer M = Fclk/2).



..... represents cut-off state (value fixed at H, L or Hi-Z according to register configuration)

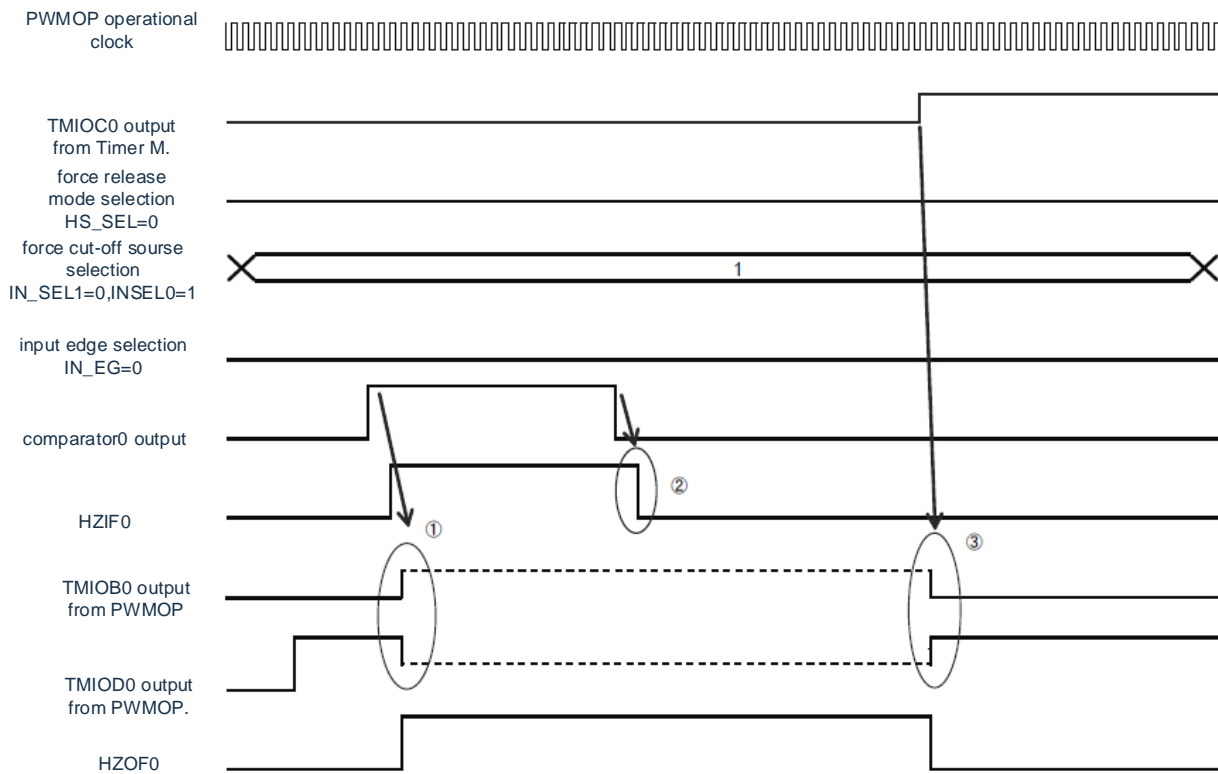
Note 1: The TMIO* (*=B~D) output from PWMOP indicates the state of the pin that reuses the M function of the timer.

Note 2: TMIO* (*=B~D) output from Timer M indicates the input signal state from the timer M output to PWMOP.

(2) The occasion of complementary PWM function output

After the source is detected, the edge of TMIOC0 selected according to OPEDGE is decommissioned.

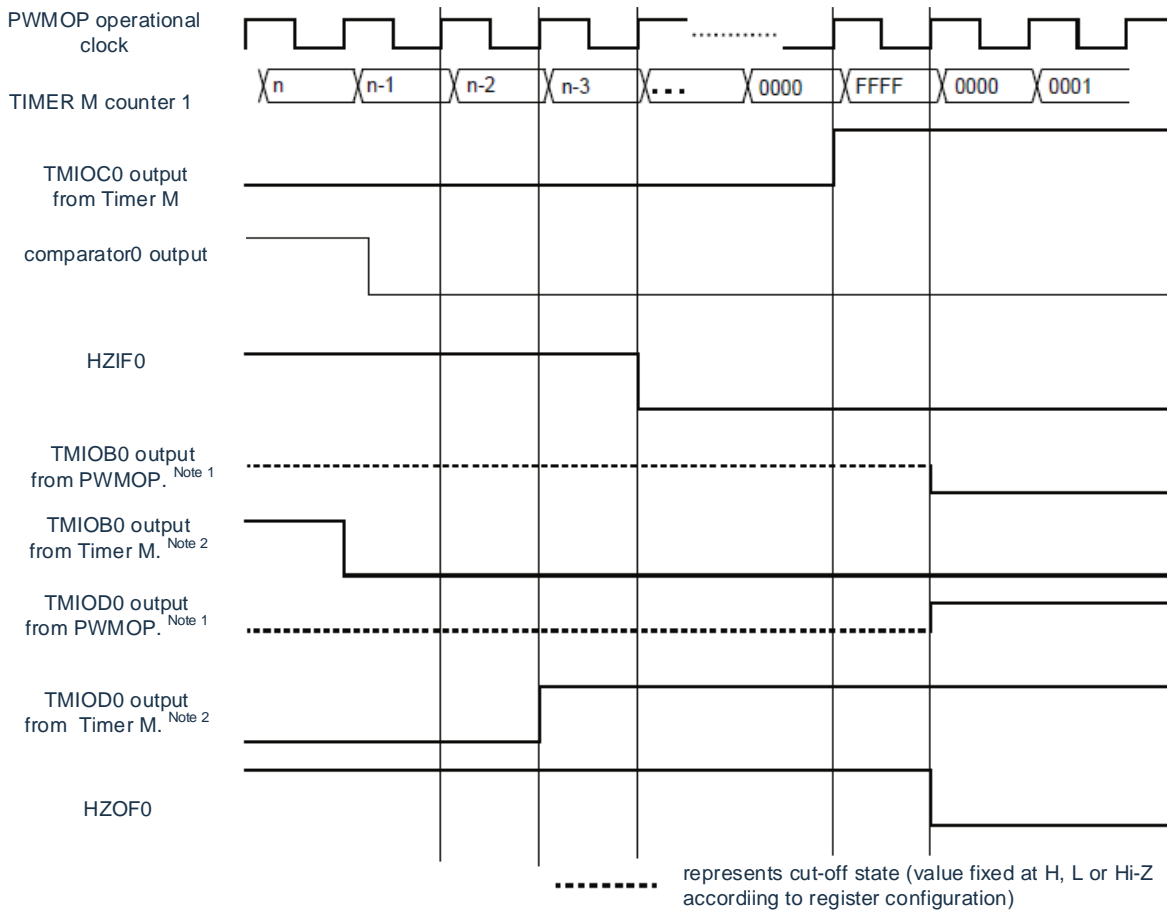
Figure 10-78 Example of hardware deforce cutoff (take TMIOB0, TMIOD0 as an example).



- (1) When the rising edge of the comparator 0 output signal is detected, the output of the T MIOB0, TMIOD0 pin is forced to cut off.
- (2) When the falling edge of the comparator 0 output signal is detected, the HZIF0 bit is cleared.
- (3) The forced cutoff state is lifted when the rising edge of T MIOC0 occurs.

For detailed timing of the mandatory cutoff, refer to Figure 10-75.

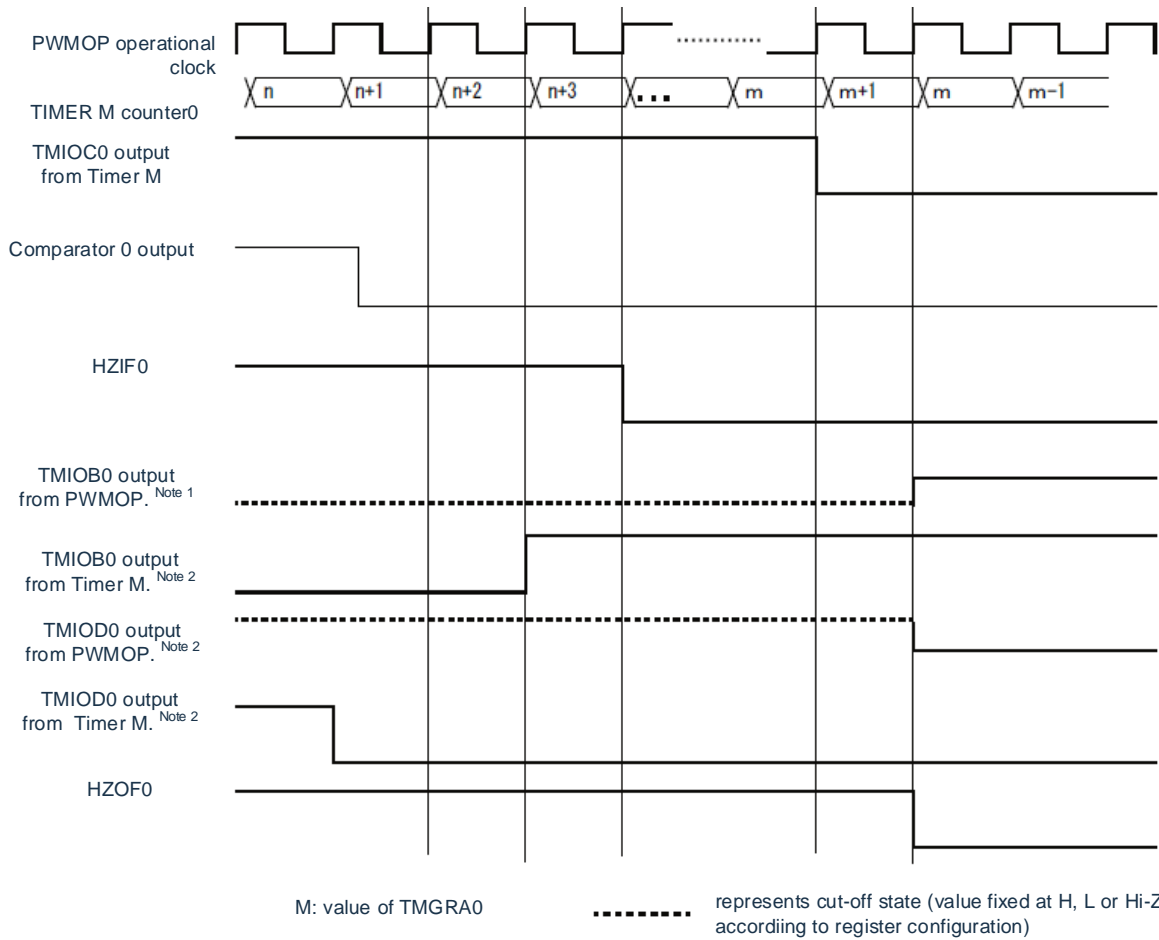
Fig. 10-79 Detailed timing diagram of forced cut-off release (count source of timer M = Fclk, decrement count).



Note 1: The TMIO* (*=B~D) output from PWMOP indicates the state of the pin that reuses the M function of the timer.

Note 2: TMIO* (*=B~D) output from Timer M indicates the input signal state from the timer M output to PWMOP.

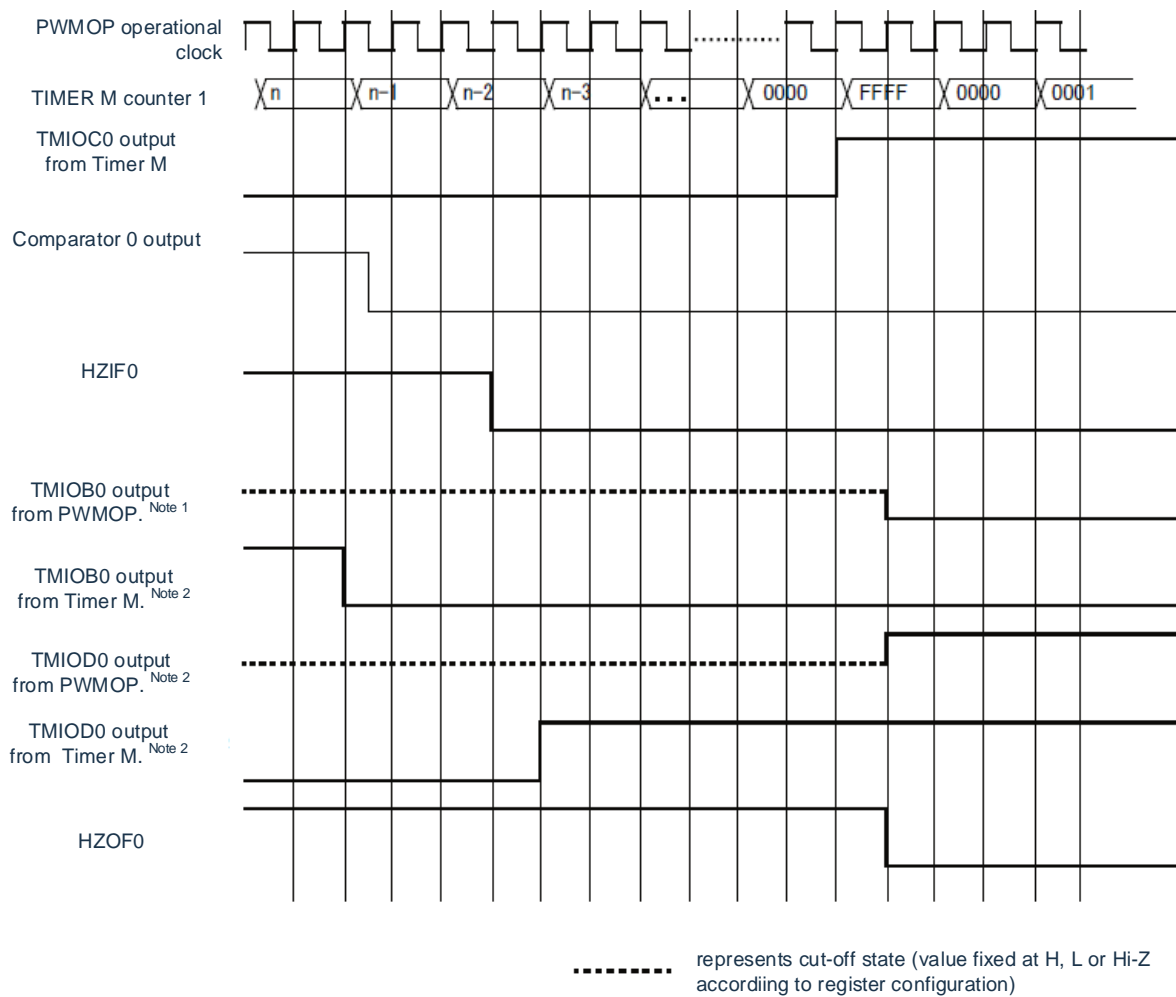
Fig. 10-80 Detailed timing diagram of forced cutoff release (count source for timer M = Fclk, counter = TMGRA0).



Note 1: The TMIO* (*=B~D) output from PWMOP indicates the state of the pin that reuses the M function of the timer.

Note 2: TMIO* (*=B~D) output from Timer M indicates the input signal state from the timer M output to PWMOP.

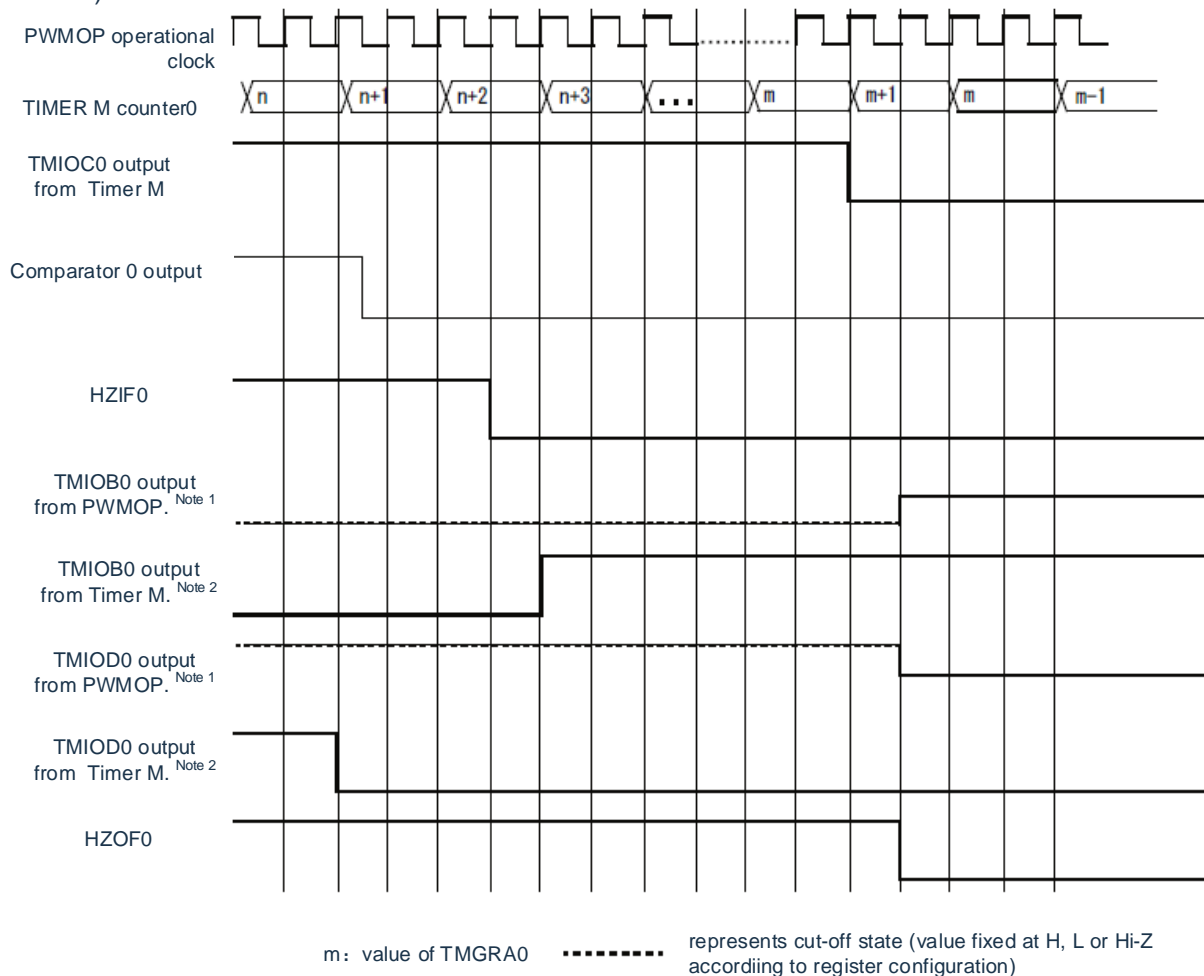
Fig. 10-81 Detailed timing diagram of forced cut-off release (count source of timer M = Fclk/2, decrement count).



Note 1: The TMIO* (*=B~D) output from PWMOP indicates the state of the pin that reuses the M function of the timer.

Note 2: TMIO* (*=B~D) output from Timer M indicates the input signal state from the timer M output to PWMOP.

Fig. 10-82 Detailed timing diagram of forced cut-off release (count source of timer M = $F_{clk}/2$, counter = TMGRA0).



Note 1: The TMIO* (*=B~D) output from PWMOP indicates the state of the pin that reuses the M function of the timer.

Note 2: TMIO* (*=B~D) output from Timer M indicates the input signal state from the timer M output to PWMOP.

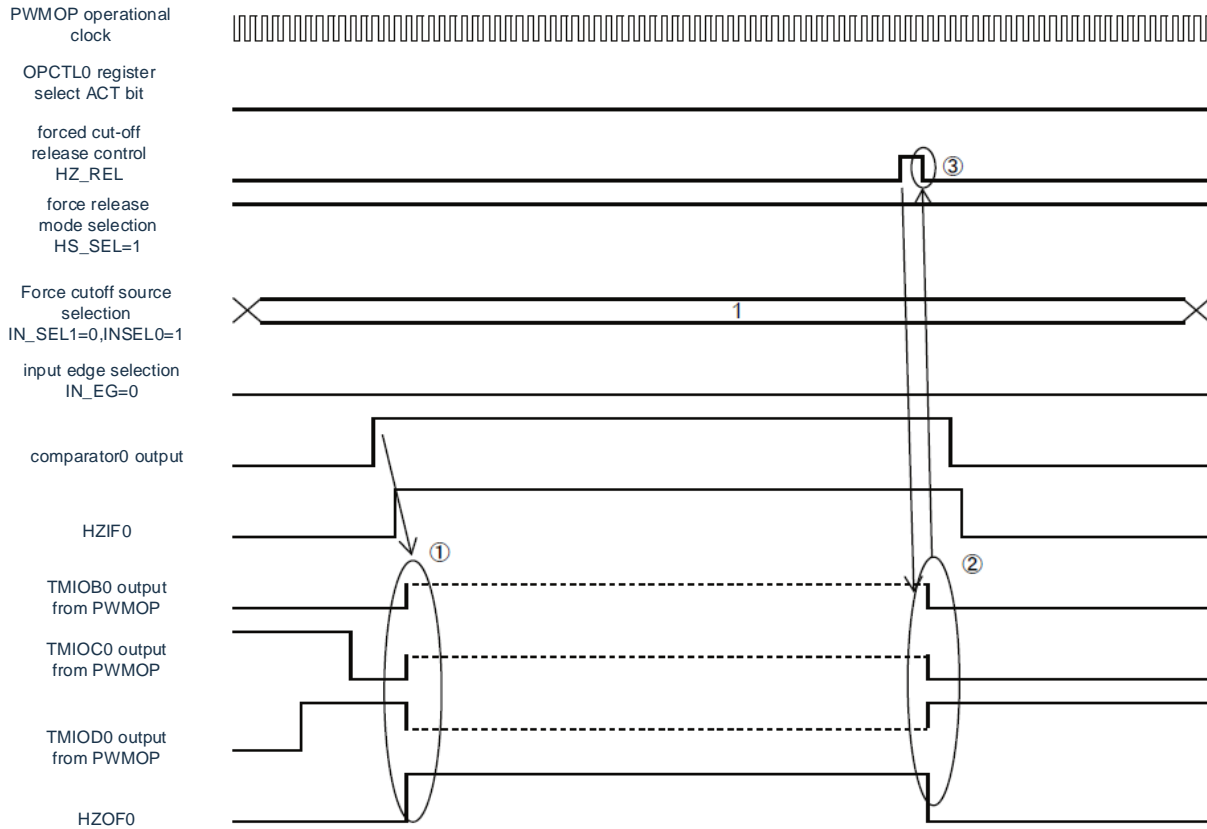
10.8.3.3 Software release (HS_SEL=1).

The ACT settings of OPCTL0 are different, and the forced cutoff is released in different timings.

(1) Immediately unblock (ACT=0) using the software

If ACT=0 is set, once the HZ_REL bit of the OPCTL0 register is set to 1, the enforcement cutoff is immediately lifted. After de-opening HZ_REL bit is automatically set to 0.

Fig. 10-83 Example of software releasing cutoff (in the case of TMIOB0, TMIOC0, TMIOD0).



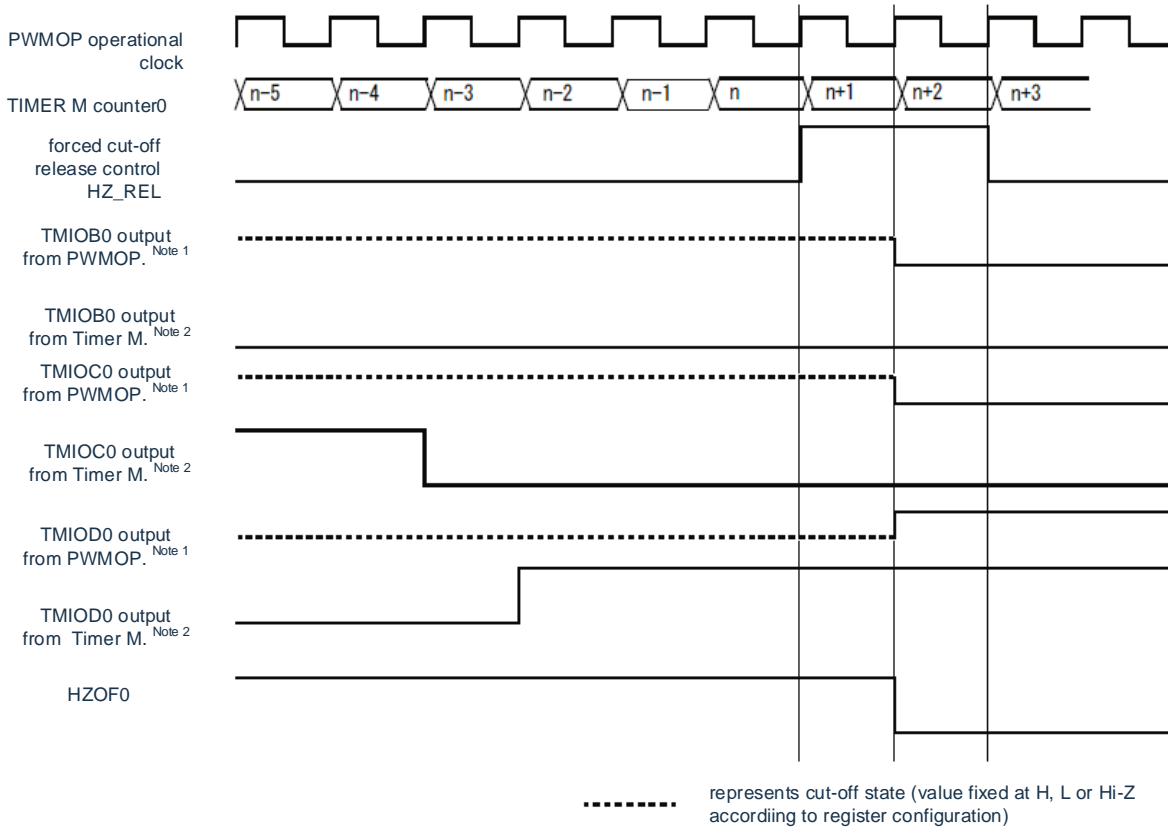
(1) When the rising edge of the comparator 0 output signal is detected, the outputs of the T MIOB0, TMIOC0, TMIOD0 pins are forced to cut off.

(2) Set the HZ_SEL to 1 to force the cutoff to be lifted immediately.

(3) After the force cutoff is lifted, HZ_REL bit becomes 0.

For detailed timing of the mandatory cutoff, refer to Figure 10-75.

Fig. 10-84 Detailed timing diagram of forced cutoff release



(2) When the software is released (ACT=1).

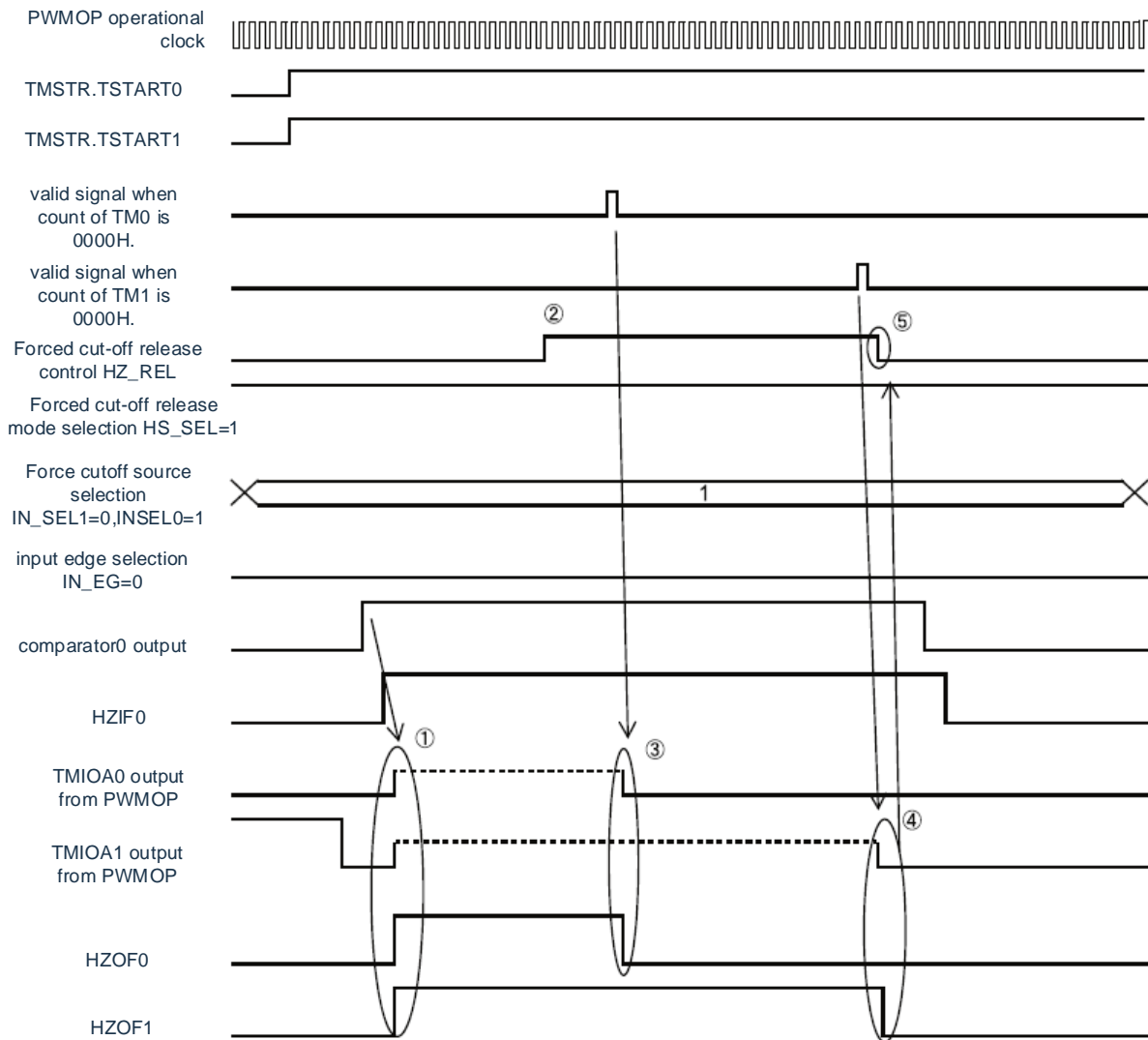
If ACT is set to 1, after setting the HZ_REL of OPCTL0 to 1, the forced cutoff state can be lifted by signaling from timer M. After deregistration, HZ_REL is automatically set to 0

When the hardware de-cutoff is detected, the release is triggered by a signal from timer M and the output is restored. When the software is decommissioned, the HZ_REL is set to 1 and the release is triggered by the signal of timer M and the output is restored. The release timing is the same.

(a) Timer M operates in the case of output comparison function, PWM function, PWM3 mode:

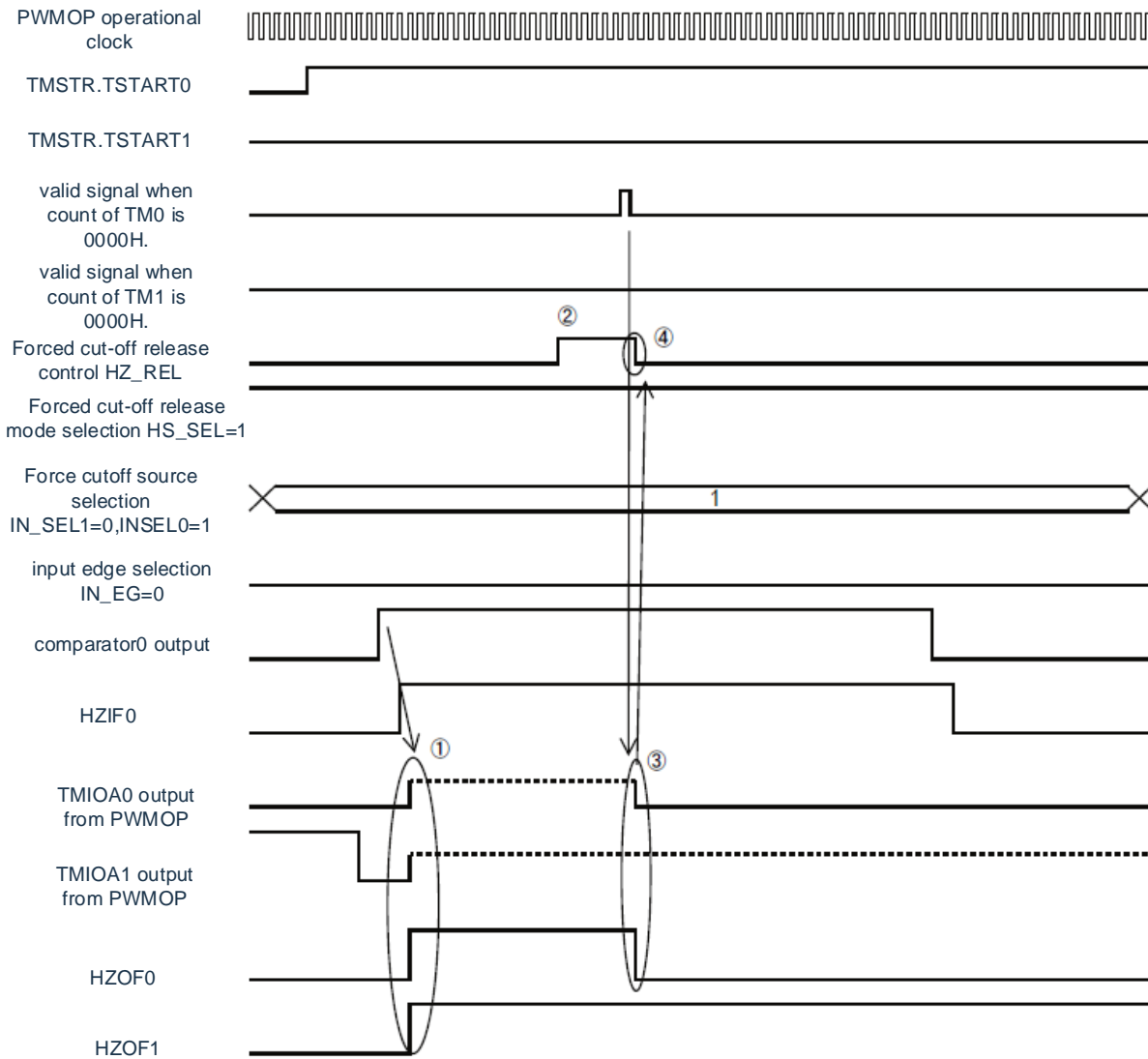
After the HZ_REL is set to 1, when the count value of T M0 becomes 0 000H, the output forced cut-off state of T MIOA0, TMIOB0, TMIOC0, TMIOD0 is lifted. When the count value of T M1 becomes 0 000H, TMIOA1, TMIOB1, TMIOC1, TMIOD1 The output forced cutoff state of 1 is dismissed.

Fig. 10-85 Example of software release forced cutoff (timer M, 2-channel count).



- (1) When the rising edge of the comparator 0 output signal is detected, the output of the TMIOA0, TMIOA1 pin is forced to cut off.
- (2) Set HZ_REL to 1 and wait for each count count value to change to 0000H.
- (3) When the count value of TM0 reaches 0 000H, the forced cutoff state of T MIOA0 is lifted.
- (4) TM1 When the count value reaches 0000H, theforced cutoff state of T MIOA 1 is lifted.
- (5) After the force cutoff is lifted, HZ_REL bit becomes 0.

Fig. 10-86 Example of software release cutoff (timer M, 1 channel count).



(1) When the rising edge of the comparator 0 output signal is detected, the output of the TMIOA0, TMIOA1 pin is forced to cut off.

(2) Set HZ_REL to 1 and wait for each count count value to change to 0000H.

(3) When the count value of TM0 reaches 0000H, the forced cutoff state of T MIOA0 is lifted.

(4) After the force cutoff is lifted, HZ_REL bit becomes 0.

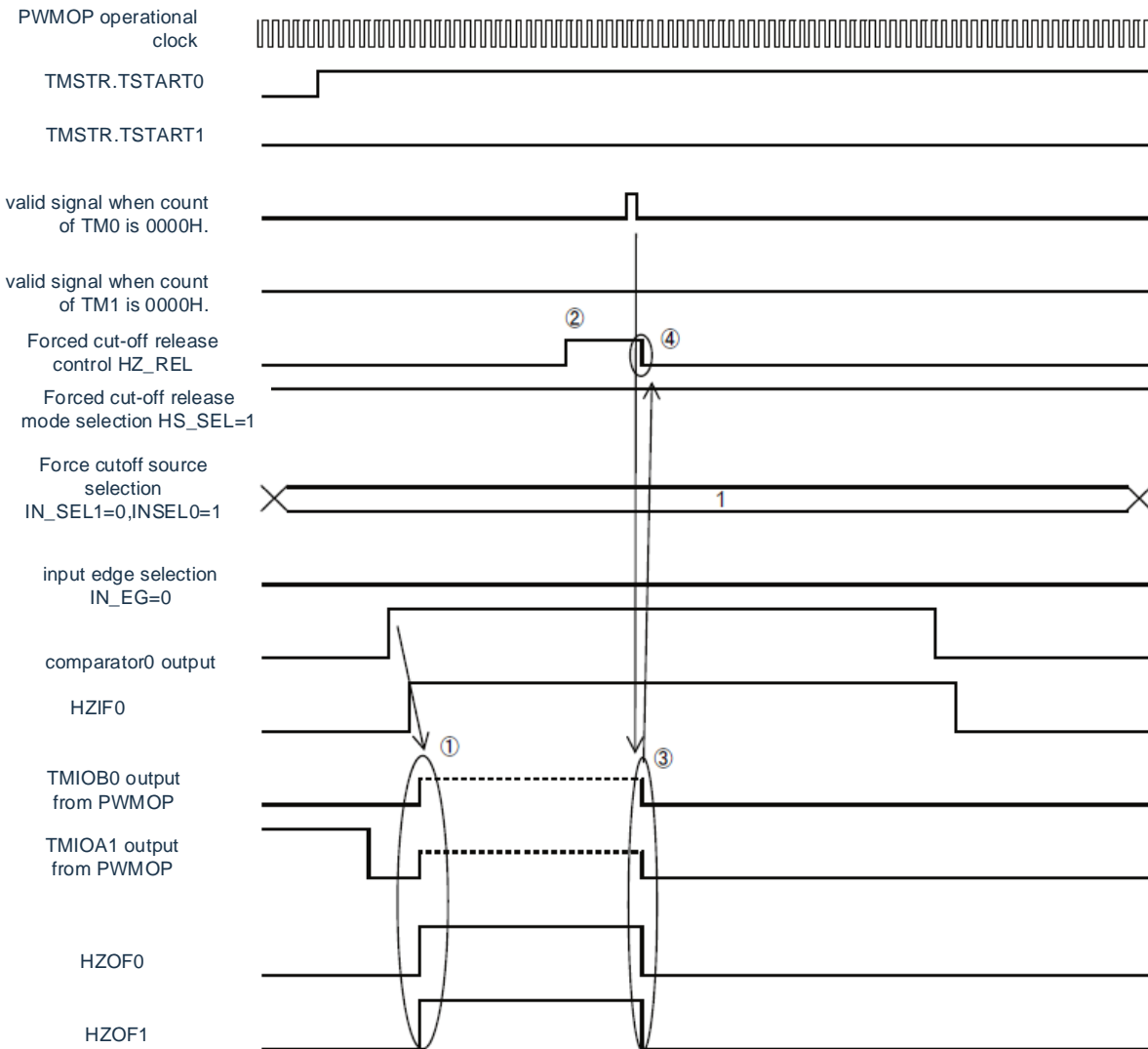
For detailed timing of the mandatory cutoff, refer to Figure 10-75.

For the detailed timing of the forced cut-off, refer to Fig. 10-76, Fig. 10-77.

For HZ_REL the timing of automatic zeroing, refer to FIG. 10-84.

(b) Timer M operates in reset synchronous PWM mode
Set the HZ_REL to 1, and at a T M0 count value of 00 000H, the output force cutoff of all T MIO pins is removed.

Figure 10-87 Example of software release forced cutoff



- (1) When the rising edge of the comparator 0 output signal is detected, the output of the TMIOA0, TMIOA1 pin is forced to cut off.
- (2) Set HZ_REL to 1 and wait for channel 0 count value of timer M to change to 0000H.
- (3) When the count value of TM0 reaches 0000H, the forced cut-off state of T MIOB0 and TMIOA1 is lifted (the action of channel 1 is not affected).
- (4) After the force cutoff is lifted, HZ_REL bit becomes 0.

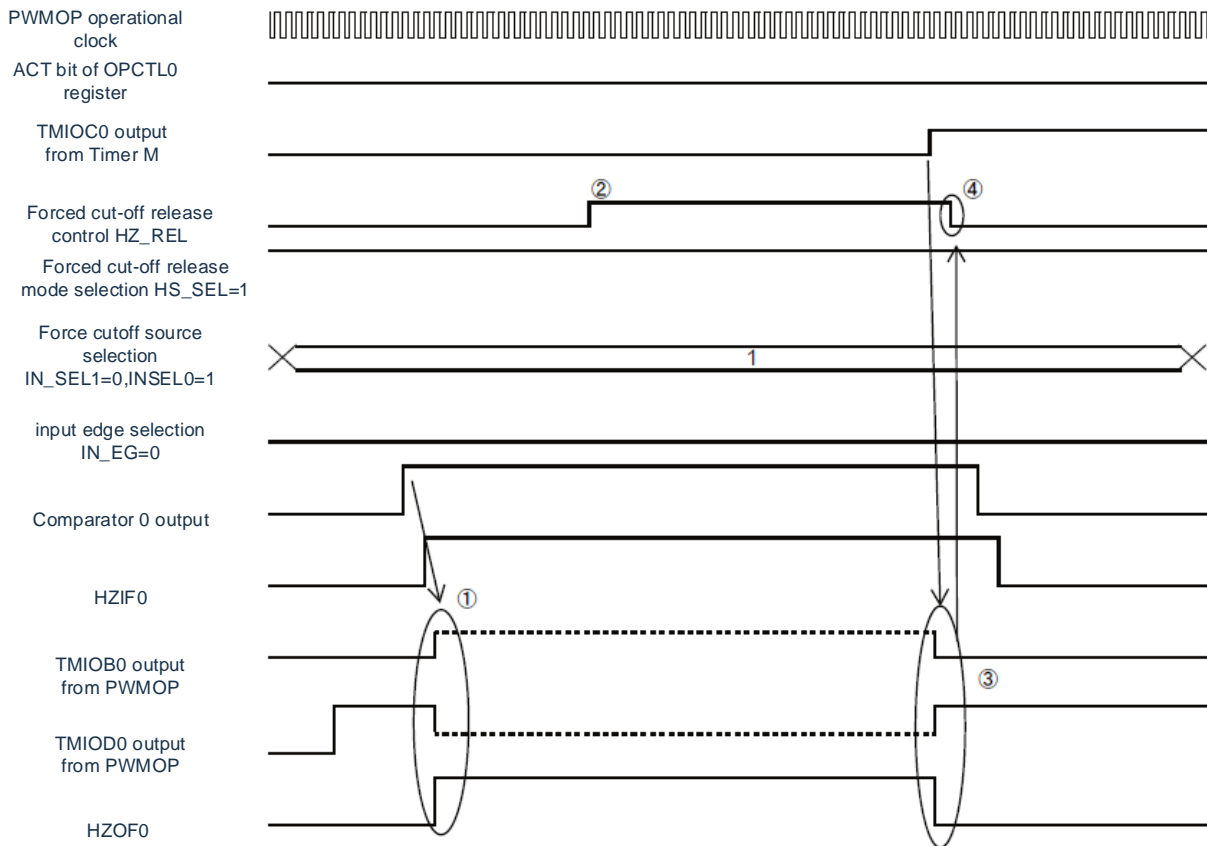
For detailed timing of the mandatory cutoff, refer to Figure 10-75.

For the detailed timing of the forced cut-off, refer to Fig. 10-76, Fig. 10-77.

For HZ_REL the timing of automatic zeroing, refer to FIG. 10-84.

(c) The timer operates in complementary PWM modes
After the HZ_REL is set to 1, the edge of TMIOC0 selected by OPEDGE is lifted from the forced cutoff.

Figure 10-88 Example of software release forced cutoff (take TMIOB0, TMIOD0 as an example).



(1) When the rising edge of the comparator 0 output signal is detected, the output of the TMIOB0, TMIOD0 pins is forced to cut off.

(2) Set the HZ_REL to 1 and wait for the rising edge of TMIOC0.

(3) After the rising edge of TMIOC0 is detected, the forced cut-off state is lifted.

(4) After the force cutoff is lifted, HZ_REL bit becomes 0.

For detailed timing of the mandatory cutoff, refer to Figure 10-75.

For the detailed timing of the forced cut-off, refer to Fig. 10-79, FIG. 10-80, FIG. 10-81, FIG. 10-82.

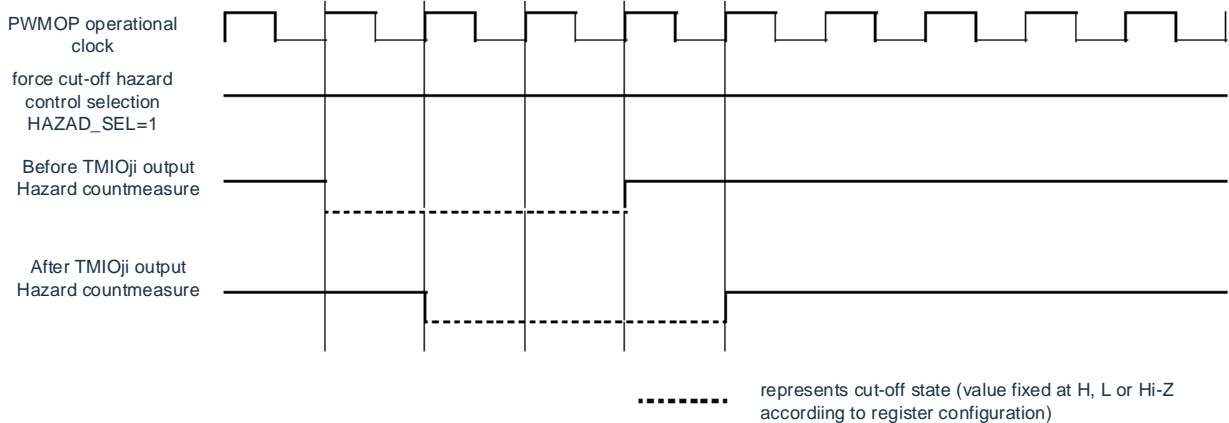
For HZ_REL the timing of automatic zeroing, refer to FIG. 10-84.

10.8.3.4 Hazard countermeasures

If the TMIO pin switches between the multiplexing function and the PORT function in the forced cut-off state/forced cut-off state/timer M action, there is a risk of malfunction. The H AZAD_SET can be set to 1 to allow hazard countermeasures to avoid this risk.

Note that when using hazard countermeasures, the output of timer M is delayed by a clock cycle than when the hazard countermeasure is disabled.

Figure 10-89 Hazard countermeasure timing diagram



Note: j=A,B,C,D; i=0,1

10.8.3.5 Outputs force-cut source checkout and unchecked out states

Whether the output forced cutoff source is checked out is selected by the cutoff source bit (OPCTL0, IN_SEL1, OPCTL0, IN_SEL0). The level of the selected signal (INTP0, CMP0) is determined.

If you set OPCTL0, the IN_EG is 0, the high level is the source detected state, and the low level is the undetected state.

If you set OPCTL0, the IN_EG is 1, the low level is the source detected state, and the high level is the undetected state.

Note: Before selecting the INTP0 input by setting the IN_SEL1, IN_SEL0, comparator 0 output as the forced cutoff factor, if the output force cutoff source is outside the threshold range, the IN_SEL1 and IN_ When SEL0 is set, HZIF0 bit is set to 1, while HZOF0 and HZOF1 are not set.

10.8.3.6 Timing plot of the counter of timer M reaches 0000H

When the hardware releases the output force cutoff, the force cutoff condition varies depending on the action mode of timer M.

- (1) Timer M works in the time series of 0000H when the output comparison function is reached
 - Count value = 0000H, timer M count starts: forced cutoff cannot be lifted
 - Timer M in the counting, using software to assign the counter value of 0 000H, to remove the forced cutoff state
 - The timer overflows, the count value becomes 0000H, and the forced cutoff state is lifted
 - In line with the value of TMGRA0, the count value becomes 0000H, and the forced cutoff state is lifted
- (2) Timer M works in the PWM function when the count value reaches the timing of 0 000H
 - Count value = 0000H, timer M count starts: forced cutoff cannot be lifted
 - Timer M in the counting, using software to assign the counter value of 0 000H, to remove the forced cutoff state
 - In line with the value of TMGRA0, the count value becomes 0000H, and the forced cutoff state is lifted
- (3) Timer M works in reset synchronous PWM mode when the count value reaches the timing of 0 000H
 - Count value = 0000H, timer M count starts: forced cutoff cannot be lifted
 - Timer M in the counting, using software to assign the counter value of 0 000H, to remove the forced cutoff state
 - In line with the value of TMGRA0, the count value becomes 0000H, and the forced cutoff state is lifted
- (4) Timer M works in PWM3 mode when the count value reaches the timing of 0 000H
 - Count value = 0000H, timer M count starts: forced cutoff cannot be lifted
 - Timer M in the counting, using software to assign the counter value of 0 000H, to remove the forced cutoff state
 - In line with the value of TMGRA0, the count value becomes 0000H, and the forced cutoff state is lifted
- (5) When the timer M count value reaches 0000H and timer M stops working
If the timer M stops working while the count value reaches 0 000H, the forced cutoff cannot be lifted.

Fig. 10-90 timing sequence when counting value = 0000H
(Count source = Fclk/2, the counter stops when the timer M count value reaches 0 000H).

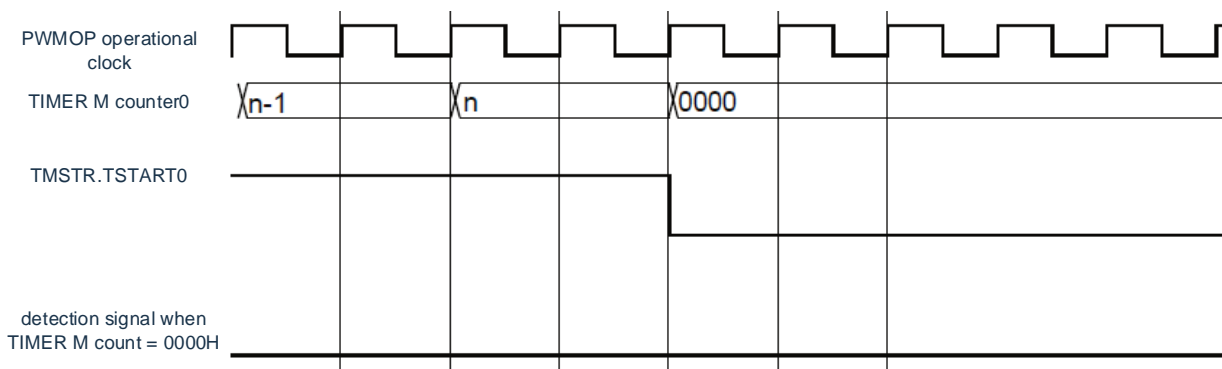
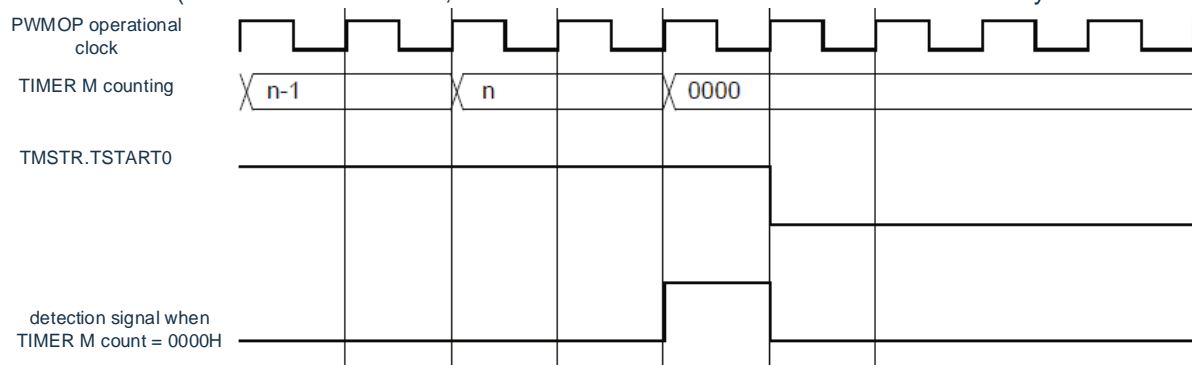


Fig. 10-91 timing sequence when count value = 0000H
(Count source = $F_{clk}/2$, timer M count value reaches 0000H at the next cycle counter stop).



10.8.3.7 Setup steps

The PWMOP may be associate with that timer M, and the setting of the PWMOP may be appended to the setting of the timer M. The steps are as follows:

After Timer M's clock and mode are set

- 1) Set PWMOPEN to 1
- 2) Set up OPCTL0 register
- 3) Set OPEDGE register
- 4) Set OPDF0, OPDF1 register, timer M action begins
- 5) Wait for the forced cutoff state shown by HZOF1, HZOF0
- 6) force termination

Note: The PWMOP implements the function of using comparator 0 output, external interrupt input INTP0 and Coordination Controllerto force cut off timer M output. Therefore, the PWMOP must be used when the timer M is active. If only timer M is used, the relevant registers of PWMOP need to be kept in reset state.

10.8.4 Precautions

(1) When the pulse output forced cutoff of timer M works at the same time as the output force cutoff of PWMOP, the priority is as follows:

Table 10-22 Priority at the time of enforcement deadline

		Pin status at PWMOP forced cutoff			
		forbid	Hi-Z	L level	H level
Pin state when timer M forces off	forbid	forbid	Hi-Z	L level	H level
	Hi-Z	Hi-Z	Hi-Z	L level	H level
	L level	L level	Hi-Z	L level	H level
	H level	H level	Hi-Z	L level	H level

- (2) In the complementary PWM mode, if the PWMOP is in the output forced cut-off state, the timer M also enters the pulse output forced cut-off state, which may produce an edge of the de-cut off for the PWMOP.
- (3) When using EVENTC as the cutoff source, you must use software to deactivate the cutoff state (HS_SEL=1).
- (4) When using the output cut-off hazard countermeasure, the output of the timer M via PWMOP is delayed by one clock cycle.
- (5) When using the output cut-off hazard countermeasure, the multiplexing function and the PORT function can be switched between the timer count operation through the output pin of the timer M of the PWMOP.
- (6) Select the effective width of the comparator 0 output and the INTP0 input must be greater than one clock cycle.

Chapter 11 Real-time clock

11.1 The function of a real-time clock

The real-time clock has the following functions.

- Holds a counter of years, months, days, hours, minutes, and seconds that can be counted up to 99 years.
- Fixed cycle interrupt function (period: 0.5 seconds, 1 second, 1 minute, 1 hour, 1 day, 1 month).
- Alarm interrupt function (alarm clock: week, hour, minute).
- 1Hz pinout function

11.2 The structure of the real-time clock

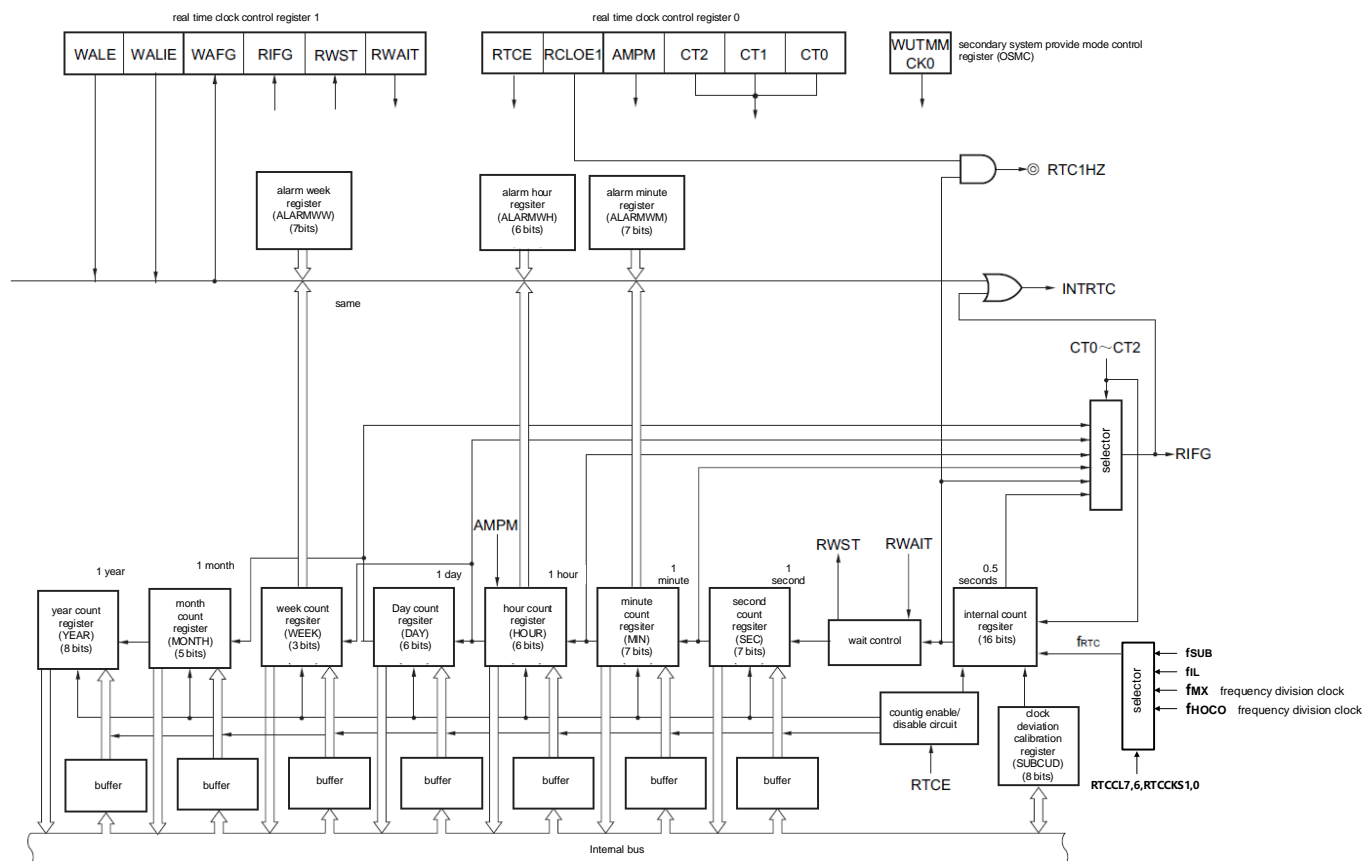
A real-time clock consists of the following hardware.

Table 11-1 Structure of real-time clocks

Item	structure
counter	Internal counter (16 bits).
Control registers	Perimeter allow register 0 (PER0. bit7)
	Real-time clock selection register (RTCCCL).
	Real-time clock control register 0 (RTCC0).
	Real-time clock control register 1 (RTCC1).
	Seconds Count Register (SEC).
	Minute Count Register (MIN).
	Hour Count Register (HOUR).
	Day count register (DAY).
	Week Count Register (WEEK).
	Month count register (MONTH).
	Year Count Register (YEAR).
	Clock Error Correction Register (SUBCUD).
	Alarm clock minute register (ALARMWM).
	Alarm hour register (ALARMWH).
	Alarm Clock Week Register (ALARMWW).

Note: The reset of the above RTC control register is controlled only by the POR reset.

Figure 11-1 Block diagram of real-time clock



Note that only the f_{MX}/f_{HOCO} divider clock (after the week $\approx 32,768$ KHz) or the subsystem clock ($f_{SUB}=32.768$ kHz) is selected. As a real-time clock, the year, month, day, day, hour, minute, and second counts can be made. When selecting a low-speed internal oscillator clock ($f_{IL}=15$ kHz), only the fixed-cycle interrupt function can be used. The fixed-period interrupt interval when f_{IL} is selected is calculated using the following calculation: fixed period (value selected by the RTCC0 register) $\times f_{SUB}/f_{IL}$.

11.3 Control Registers the 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).
- Clock Error Correction Register (SUBCUD).
- Alarm Clock Minute Register (ALARMWM).
- Alarm Clock Hour Register (ALARMWH).
- Alarm Clock Week Register (ALARMWW).
- Port Mode Register (PMx).
- Port register (Px).

11.3.1 Peripheral enable register 0 (PER0).

The PER0 register is a register that is set to allow or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use.

To use a real-time clock, bit7 (RTCEN) must be set to "1". The PER0 register is set via the 8-bit memory operation instructions. After generating a reset signal, the value of this register changes to "00H".

Fig.11-2 The format of Peripheral enable register 0 (PER0)

Address: 0x40020420		After reset: 00H				R/W		
symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	-	ADCEN	IICA0EN	SCI1EN	SCI0EN	CAN0IN	TM40EN

RTCEN	Provides control of the input clock of the real-time clock (RTC) and a 15-bit interval timer
0	Stop providing the input clock. • You cannot write SFR for real-time clocks (RTCs) and 15-bit interval timers. • The real-time clock (RTC) and 15-bit interval timer are reset.
1	An input clock is provided. • Read and write SFR for real-time clock (RTC) and 15-bit interval timers.

Note 1 If you want to use a real-time clock, you must first place the RTCEN position "1" in the oscillation stable state of the counting clock (f_{RTC}), 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).
- Clock Error Correction Register (SUBCUD).
- Alarm Clock Minute Register (ALARMWM).
- Alarm Clock Hour Register (ALARMWH).
- Alarm Clock Week Register (ALARMWW).

2. Can be stopped to the real-time clock and 15 in deep sleep mode or sleep mode running in the subsystem clock by providing the RTCLPC position "1" of the mode control register (OSMC) of the subsystem clock. Peripheral functions other than bit-interval timers provide the subsystem clock.

11.3.2 Real-time clock selection register (RTCCL).

The count clock (fRTC) of the real-time clock and the 15-bit interval timer can be selected via RTCCL.

Figure 11-3 Format of the real-time clock selection register (RTCCL).

Address: 0x4004047C after reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
RTCCL	RTCCL7	RTCCL6	RTCCL5	0	0	0	RTCKS1	RTCKS0

RTCCL7	Selection of clock source for real-time clocks, 15-bit interval timers for counting clocks
0	Select the high-speed system clock (fMX).
1	Choose the high-speed internal oscillator (fhoco).

RTCKS1	RTCKS0	RTCCL6	RTCCL5	Selection of operating clocks for real-time clocks, 15-bit interval timers for counting clocks
0	0	x	x	Subsystem clock (fSUB).
0	1			Low-speed internal oscillator clock (f _{IL}) (WUTMMCK0 must be set to 1).
1	0	0	1	Master clock fmax/fhoco (selected via RTCCL7)/1952
1	0	0	0	Master clock fmax/fhoco (selected via RTCCL7)/1464
1	0	1	0	Master clock fmax/fhoco (selected via RTCCL7)/976
1	1	0	0	Master clock fmax/fhoco (selected via RTCCL7)/488
1	1	1	0	Master clock fmax/fhoco (selected via RTCCL7)/244

11.3.3 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 via the 8-bit memory operation instruction. After generating a reset signal, the value of this register changes to "00H".

Figure 11-4 Format of real-time clock control register 0 (RTCC0).

Address: 0x40044F5D after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
RTCC0	RTCE	0	RCLOE1 ^{note}	0	AMPM	CT2	CT1	CT0

RTCE	Operational control of the real-time clock
0	Stop the counter from running.
1	Start the run of the counter.

RCLOE1	Output control of the RTC1HZ pin
0	Disable the output of the RTC1HZ pin (1Hz).
1	The output of the RTC1HZ pin is allowed (1Hz).

AMPM	Choice of 12-hour system/24-hour system
0	12-hour system (for morning or afternoon).
1	24-hour system
<ul style="list-style-type: none"> To change the value of the AMPM bit, the RWAIT bit (bit0 of the real-time clock control register 1 (RTCC1)) must be set to "1" It is rewritten later. If you change the value of the AMPM bit, the value of the hour count register (HOUR) becomes the corresponding value of the set time system. The representation of the time bits is shown in Table 11-2. 	

CT2	CT1	CT0	Choice of Fixed Cycle Interrupt (INTRTC).
0	0	0	The fixed-cycle interrupt feature is not used.
0	0	1	0.5 seconds at a time (synchronized with seconds accumulation).
0	1	0	Once per second (at the same time as the seconds are accumulated).
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	x	Once a month (1st of each month at 00:00:00 a.m.).
To change the value of the CT2 to CT0 bits during counter operation (RTCE=1), it must be overwritten after the INRTC is set to disable interrupt processing through the interrupt mask flag register, and the RIFG must be cleared after the override Flags and RTCIF flags, and then set to Allow Interrupt Handling.			

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

2. When the RTCE bit is "0", even if the RCLOE1 position "1" is not output 1Hz.

Note x: Ignored

11.3.4 Real-time clock control register 1 (RTCC1).

This is an 8-bit register that controls the alarm interrupt function and counter waits. The RTCC1 register is set via the 8-bit memory operation instruction. After generating a reset signal, the value of this register changes to "00H".

Fig. 11-5 Format of real-time clock control register 1 (RTCC1) (1/2).

Address: 0x40044F5E After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
RTCC1	WALIE	WALIE	0	WAFG	RIFG	0	RWST	RWAIT

WALIE	Run control of the alarm
0	Consistent operation is invalid.
1	Consistent operation works.
To set the WALL bit in counter operation (RTCE=1) and the WALLE bit is "1", it must be rewritten after the INRTC is set to disable interrupt processing through the interrupt mask flag register. And the WAFG flag and RTCIF flag must be cleared after rewriting. To set each alarm register (RTCC1 register's WALIE flag, alarm clock minute register (ALARMWM), alarm hour register (ALARMWH) and alarm clock week register (ALARMWW)), the WALL position must be "0" (invalid for consistent operation).	

WALIE	Operational control of the Alarm Clock Interrupt (INTRTC) function
0	No consistent alarm interruptions.
1	Produces a consistent alarm.

WAFG	Alarm detects status flags
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. Valid only when the WALL bit is "1", it becomes "1" after detecting the alarm clock is consistent and passes through 1 fRTC clock. Clear this flag by writing "0" to it. The operation to write "1" is invalid.	

Figure 11-5 Format of real-time clock control register 1 (RTCC1) (2/2).

RIFG	Fixed-cycle interrupt status flag
0	There are no fixed cycle interruptions.
1	Generates a fixed cycle interrupt.
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. The operation to write "1" is invalid.	

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.
<p>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".</p> <p>The time required from the RWAIT position "1" to the time the count value can be read and written (RWST=1) is up to 1 fRTC clock.</p> <p>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".</p> <p>However, when the second count register is written, the overflow state that occurs is not maintained.</p>	

Note 1 Fixed-cycle interrupts and alarm-consistent interrupts use the same interrupt source (INTRTC). In the case of using these two interrupts at the same time, when INTRTC occurs, you can determine which interrupt occurred by acknowledging the fixed-cycle interrupt status flag (RIFG) and the alarm detection status flag (WAFG).

2. If the Write Second Count Register (SEC), clear the internal counter (16 bits).

11.3.5 Clock Error Correction Register (SUBCUD).

This is a register that can correct the clock speed with high accuracy by changing the overflow value (reference value: 7FFFH) from the internal counter (16 bits) to the second count register (SEC).

Set the SUBCUD registers via 16-bit memory manipulation instructions. After generating a reset signal, the value of this register becomes "0000H".

Figure 11-13 Format of the Clock Error Correction Register (SUBCUD).

[illegible]

Note: "/" indicates the value after each person is reversed.

The ranges that can be corrected by the Clock Error Correction Register (SUBCUD) are as follows.

	DEV = 0 (correction every 20 seconds)	DEV = 1 (correction every 60 seconds).
Correctable range	-12496.9ppm~12496.9ppm	-4165.6ppmto4165.6ppm
Maximum quantization error	±1.53ppm	±0.51ppm
Minimum resolution	±3.05ppm	±1.02ppm

Note When the correction range is outside the range of -4165.6ppm to 4165.6ppm , the DEV position must be "0".

11.3.6 Seconds Count Register (SEC).

This is an 8-bit register that represents the second count value from 0 to 59 (decimal). The count is incremented by the overflow of the internal counter (16 bits). At write time, the data is first written to the buffer and to the counter after passing through up to 2 fRTC clocks. The decimal value must be set to 00~59 in the BCD code.

Set the SEC registers via the 8-bit memory operation instructions. After generating a reset signal, the value of this register changes to "00H".

Figure 11-6 sec count register (SEC) format

Address: 0x40044F52 After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
SEC	0	SEC40	SEC20	SEC10	SEC8	SEC4	SEC2	SEC1

Note that to read and write this register during counter operation (RTCE=1), it must be done according to the steps described in "11.4.3 Reading and Writing of Real-Time Clock Counters". Note If the Write Second Count Register (SEC) clears the internal counter (16 bits).

11.3.7 Minute Count Register (MIN).

This is an 8-bit register that represents the minute count value in 0 to 59 (decimal). The count is incremented by the overflow of the second counter. At write time, the data is first written to the buffer and to the counter after passing through up to 2 fRTC clocks. The overflow of the second count register is ignored during the write operation and is set to the write value. The decimal value must be set to 00~59 in the BCD code.

Set the MIN register via the 8-bit memory operation instruction. After generating a reset signal, the value of this register changes to "00H".

Figure 11-7 min count register (MIN) format

Address: 0x40044F53 After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
MIN	0	MANDN40	MIN20	MANDN1	MIN8	MIN4	MIN2	MIN1

Note That to read and write this register while the counter is running (RTCE=1), it must follow the steps described in "11.4.3 Reading and Writing to a Real-Time Clock Counter".

11.3.8 Hour Count Register (HOUR).

This is 8 of the hour count value expressed in 00 to 23 or 01 to 12, 21 to 32 (decimal). Bit registers. The count is incremented by the overflow of the minute counter.

At write time, the data is first written to the buffer and to the counter after passing through up to 2 fRTC clocks. The overflow of the minute count register is ignored during the write operation and is set to the write value.

The time system must be set according to the bit3 (AMPM) of the real-time clock control register 0 (RTCC0), and the decimal 00~ in BCD code 23 or 01~12, 21~32.

If you change the value of the AMPM bit, the value of the HOUR register becomes the corresponding value of the time system set. Set the HOUR register via the 8-bit memory operation instruction. After generating a reset signal, the value of this register changes to "12H".

However, if the AMPM position is "1" after reset, the value of this register becomes "00H".

Figure 11-8 Format of the Hour Count Register (HOUR).

Address: 0x40044F54 After reset: 12H R/W

Symbol	7	6	5	4	3	2	1	0
HOUR	0	0	HOUR20	HOUR10	HOUR8	HOUR4	HOUR2	HOUR1

Note 1 When the AMPM bit is selected as "0" (12-hour system), the BIT5 (HOUR20) representation of the HOUR register is indicated AM (0) /PM (1) .

2. To read and write this register in the run of the counter (RTCE=1), it must be carried out according to the steps described in "1 1.4.3 Reading and Writing of the Real-Time Clock Counter".

The relationship between the setting value of the AMPM bit, the value of the hour count register (HOUR), and the time is shown in Table 11-2.

Table 11-2 Representation of time bits

24-hour representation (AMPM=1).		12 hours is represented (AMPM=0).	
Time	HOUR register	Time	HOUR register
0 o'clock	00H	AM12 o'clock	12H
1 o'clock	01H	AM1 o'clock	01H
2 o'clock	02H	AM2 o'clock	02H
3 o'clock	03H	AM3 o'clock	03H
4 o'clock	04H	AM4 o'clock	04H
5 o'clock	05H	AM5 o'clock	05H
6 o'clock	06H	AM6 o'clock	06H
7 o'clock	07H	AM7 o'clock	07H
8 o'clock	08H	AM8 o'clock	08H
9 o'clock	09H	AM9 o'clock	09H
10 o'clock	10H	AM10 o'clock	10H
11 o'clock	11H	AM11 o'clock	11H
12 o'clock	12H	PM12 o'clock	32H
13 o'clock	13H	PM1 o'clock	21H
14 o'clock	14H	PM2 o'clock	22H
15 o'clock	15H	PM3 o'clock	23H
16 o'clock	16H	PM4 o'clock	24H
17 o'clock	17H	PM5 o'clock	25H
18 o'clock	18H	PM6 o'clock	26H
19 o'clock	19H	PM7 o'clock	27H
20 o'clock	20H	PM8 o'clock	28H
21 o'clock	21H	PM9 o'clock	29H
22 o'clock	22H	PM10 o'clock	30H
23 o'clock	23H	PM11 o'clock	31H

When the AMPM bit is "0", the value of the HOUR register is 12 hours; When the AMPM bit is "1", the value of the HOUR register is indicated as 24 hours.

At 12 hours, bit5 of the HOUR register represents am/pm. "0" in the morning (AM) and "1" in the afternoon (PM).

11.3.9 Day count register (DAY).

This is an 8-bit register that represents the daily count value from 1 to 31 (decimal). The count is incremented by the overflow of the hour counter. Counters make the following counts.

- 01~31 (month of 1, 3, 5, 7, 8, 10, 12)
- 01~30 (month of 4, 6, 9, 11)
- 01~29 (Leap year in February).
- 01~ 28 (February normal year).

At write time, the data is first written to the buffer and to the counter after passing through up to 2 f_{RTC} clocks. The overflow of the hour count register is ignored during the write operation and is set to the write value. The decimal 01~31 must be set in the BCD code.

Set the DAY register via the 8-bit memory operation instruction. After generating a reset signal, the value of this register changes to "01H".

Figure 11-9 Format of the Day Count Register (DAY).

Address: 0x40044F56H

after reset: 01H R/W

Symbol	7	6	5	4	3	2	1	0
DAY	0	0	DAY20	DAY10	DAY8	DAY4	DAY2	DAY1

Note That to read and write this register while the counter is running (RTCE=1), it must follow the steps described in "11.4.3 Reading and Writing to a Real-Time Clock Counter".

11.3.10 Week Count Register (WEEK).

This is an 8-bit register that represents the week count value in 0 to 6 (decimal). Increment the count synchronously with the daily counter.

At write time, the data is first written to the buffer and to the counter after passing through up to 2 fRTC clocks. The decimal value must be set to 00~06 in the BCD code.

Set the WEEK register via the 8-bit memory operation instructions. After generating a reset signal, the value of this register changes to "00H".

Figure 11-10 Format of the Week Count Register (WEEK).

Address: 0x40044F55H

after reset: 00HR/W

Symbol	7	6	5	4	3	2	1	0			
WEEK	0	0	0	0	0	0	0	0	WEEK4	WEEK2	WEEK1

Note 1 The corresponding values of the month count register (MONTH) and the day count register (DAY) are not automatically saved to the week register (WEEK). The following settings must be made after the reset is lifted:

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

2. To read and write this register in the run of the counter (RTCE=1), it must be carried out according to the steps described in "1 1.4.3 Reading and Writing of the Real-Time Clock Counter".

11.3.11 Month count register (MONTH).

This is an 8-bit register that represents the monthly count value from 1 to 12 (decimal). The count is incremented by the overflow of the daily counter.

At write time, the data is first written to the buffer and to the counter after passing through up to 2 f_{RTC} clocks. The overflow of the day count register is ignored during the write operation and set to the write value. The decimal 01~12 must be set in the BCD code.

The MONTH register is set by the 8-bit memory operation instruction. After generating a reset signal, the value of this register changes to "01H".

Figure 11-11 Format of the Month Count Register (MONTH).

Address: 0x40044F57H

After reset: 01HR/W

Symbol	7	6	5	4	3	2	1	0	
MONTH	0	0	0	0	MONTH10	MONTH8	MONTH4	MYTH2	MONTH1

Note That to read and write this register while the counter is running (RTCE=1), it must follow the steps described in "11.4.3 Reading and Writing to a Real-Time Clock Counter".

11.3.12 Year Count Register (YEAR).

This is an 8-bit register representing the annual count value from 0 to 99 (decimal). The count is incremented by the overflow of the month counter (MONTH). 00, 04, 08, , 92, 96 are leap years.

At write time, the data is first written to the buffer and to the counter after passing through up to 2 f_{RTC} clocks. The overflow of the MONTH register is ignored during the write operation and is set to the write value. The decimal value must be set to 00~99 in the BCD code. Set the YEAR register with an 8-bit memory operation instruction. After generating a reset signal, the value of this register changes to "00H".

Figure 11-12 Format of the Year Count Register (YEAR).

Address: 0x40044F58H	after reset: 00H R/W							
Symbol	7	6	5	4	3	2	1	0
YEAR	YEAR80	YEAR40	ANDEAR20	YEAR10	YEAR8	ANDEAR4	YEAR2	YEAR1

Note That to read and write this register while the counter is running (RTCE=1), it must follow the steps described in "11.4.3 Reading and Writing to a Real-Time Clock Counter".

11.3.13 Alarm clock minute register (ALARMWM).

This is the register for setting the alarm minute.

The ALARMWM register is set by the 8-bit memory operation instruction. After generating a reset signal, the value of this register changes to "00H".

Note that the decimal value must be set to 00~59 in the BCD code. If you set a value outside the range, the alarm is not detected.

Figure 11-14 Format of the Alarm Clock Minute Register (ALARMWM).

Address: 0x40044F5AH	after reset: 00H R/W							
Symbol	7	6	5	4	3	2	1	0
ALARMWM	0	WM40	WM20	WM10	WM8	WM4	WM2	WM1

11.3.14 Alarm hour register (ALARMWH).

This is the register for setting the alarm hour.

The ALARMWH register is set via the 8-bit memory operation instruction. After generating a reset signal, the value of this register changes to "12H". However, if the AMPM position is "1" after reset, the value of this register becomes "00H".

Note that the decimal value of 00~23 or 01~12, 21~32 must be set in the BCD code. If you set a value outside the range, the alarm is not detected.

Figure 11-15 Format of the Alarm Clock Hour Register (ALARMWH).

Address: 0x40044F5 BH After reset: 12H R/W								
Symbol	7	6	5	4	3	2	1	0
ALARMWH	0	0	WH20	WH10	WH8	WH4	WH2	WH1

Note that when the AMPM bit is selected as "0" (12-hour system), bit5 (WH20) of the ALARMWH register represents AM (0) /PM (1) .

11.3.15 Alarm Clock Week Register (ALARMWW).

This is the register that sets the alarm for the week.

The ALARMWW register is set via the 8-bit memory operation instruction. After generating a reset signal, the value of this register changes to "00H".

Figure 11-16 Format of alarm clock week register (ALARMWW).

Address: 0x40044F5CH	after reset: 00H R/W							
Symbol	7	6	5	4	3	2	1	0
ALARMWW	0	WW6	WW5	WW4	WW3	WW2	WW1	WW0

An example of setting the alarm time is shown below.

Alarm setting time	week							12 hours indicates				24 hours indicated			
	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday								
	W	W	W	W	W	W	W	10 o'clock	1 o'clock	10 minutes	1 minute	10 o'clock	1 o'clock	10 minutes	1 minute
	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
Monday to Friday 00:00 pm	0	1	1	1	1	1	0	3	2	0	0	1	2	0	0
Sunday 1:30 p.m	1	0	0	0	0	0	0	2	1	3	0	1	3	3	0
Monday, Wednesday, Friday 11:59 p.m	0	1	0	1	0	1	0	3	1	5	9	2	3	5	9

11.3.16 Port mode registers and port registers

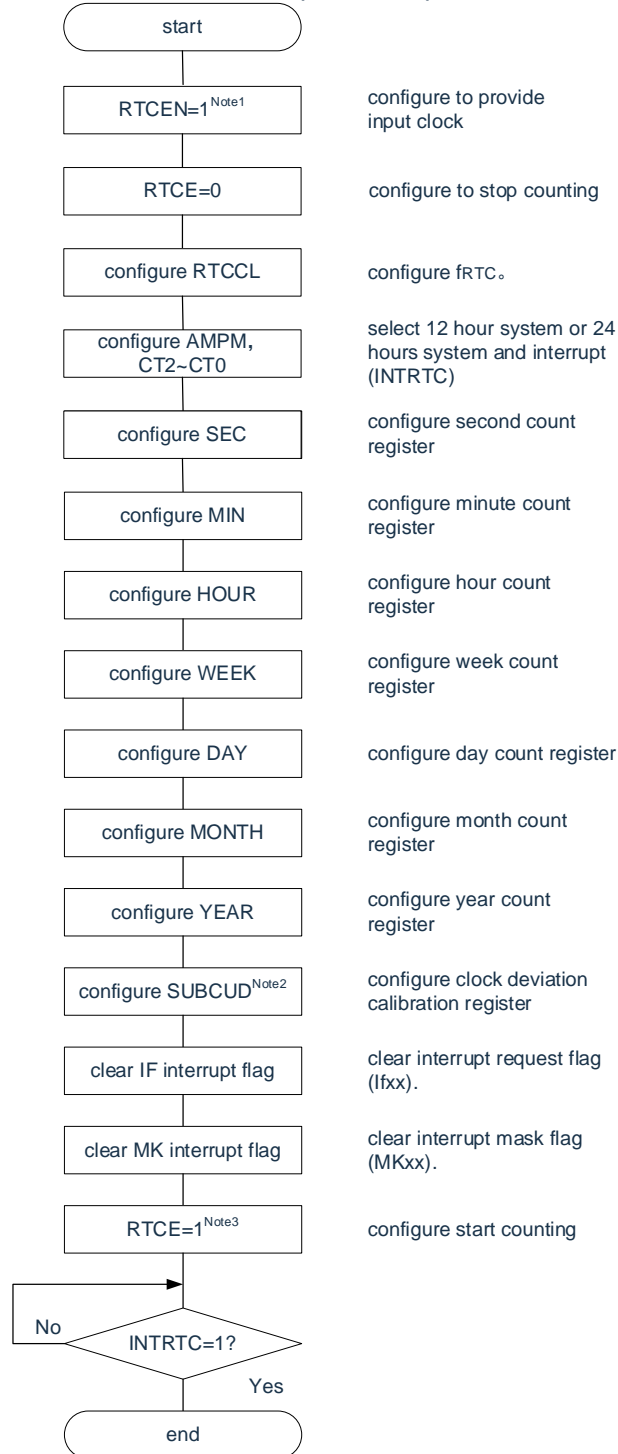
When using the multiplex port of the RTC1HZ output pin as a 1Hz output, the bits of the port mode control register (PMCxx), the bits of the port mode register (PMxx), and the port register (Pxx) corresponding to each port must be used) position "0".

The set port mode registers (PMxx), port registers (Pxx), and port mode control registers (PMCxx) vary by product. For details, please refer to "Register Settings when Using the Multiplexing Function in 2.5".

11.4 Operation of a real-time clock

11.4.1 The operation of the real-time clock begins

Figure 11-19 The start steps of the operation of the real-time clock



Concentrate:

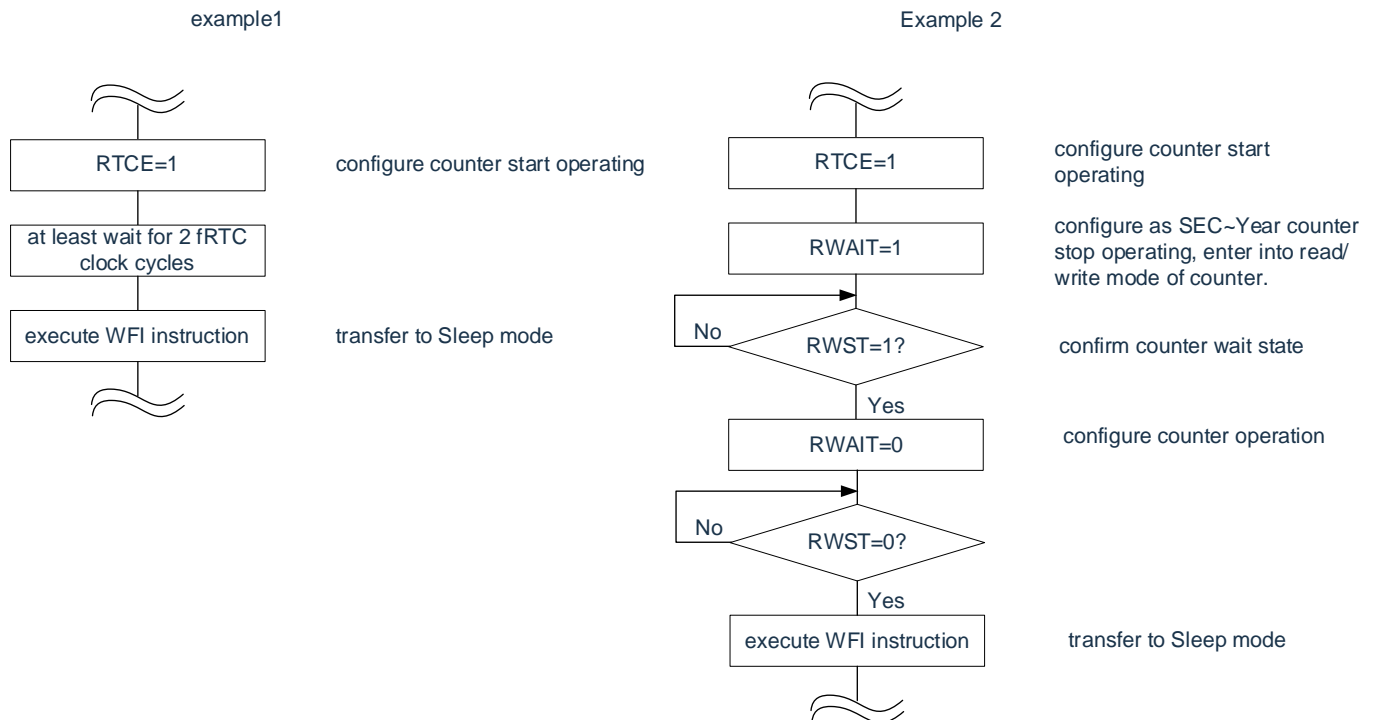
1. The RTCEN position "1" must first be placed in the oscillation stable state of the count clock (fRTC).
2. This is just a case where the clock error needs to be corrected. For how to calculate the correction value, refer to "Example of Clock Error Correction for 11.4.6 Real-Time Clocks".
3. In the case of transferring to sleep mode after the RTCE bit is "1" without waiting for the INRTC bit to change to "1", confirm "1." 1.4.2 Start running after the transfer of sleep mode" steps.

11.4.2 Start the transfer of sleep mode after running

To transfer to sleep (including deep sleep) mode immediately after the RTCE position "1", one of the following processes must be performed. However, after placing the RTCE position "1", these processes are not required if you want to move to sleep mode after an INTTTC interrupt occurs.

- Move to sleep mode after at least 2 counting clocks (f_{RTC}) of time after RTCE position "1" has elapsed (refer to Figure 11 Example 1 of -20).
- After the RWATCH position "1" is placed, the RWAIT position is changed to "1" and the RWST bit is confirmed by polling. Then, place the RWAIT position "0" and confirm again by polling that the RWST bit becomes "0" and then move to sleep mode (reference Fig. 11-20 example 2).

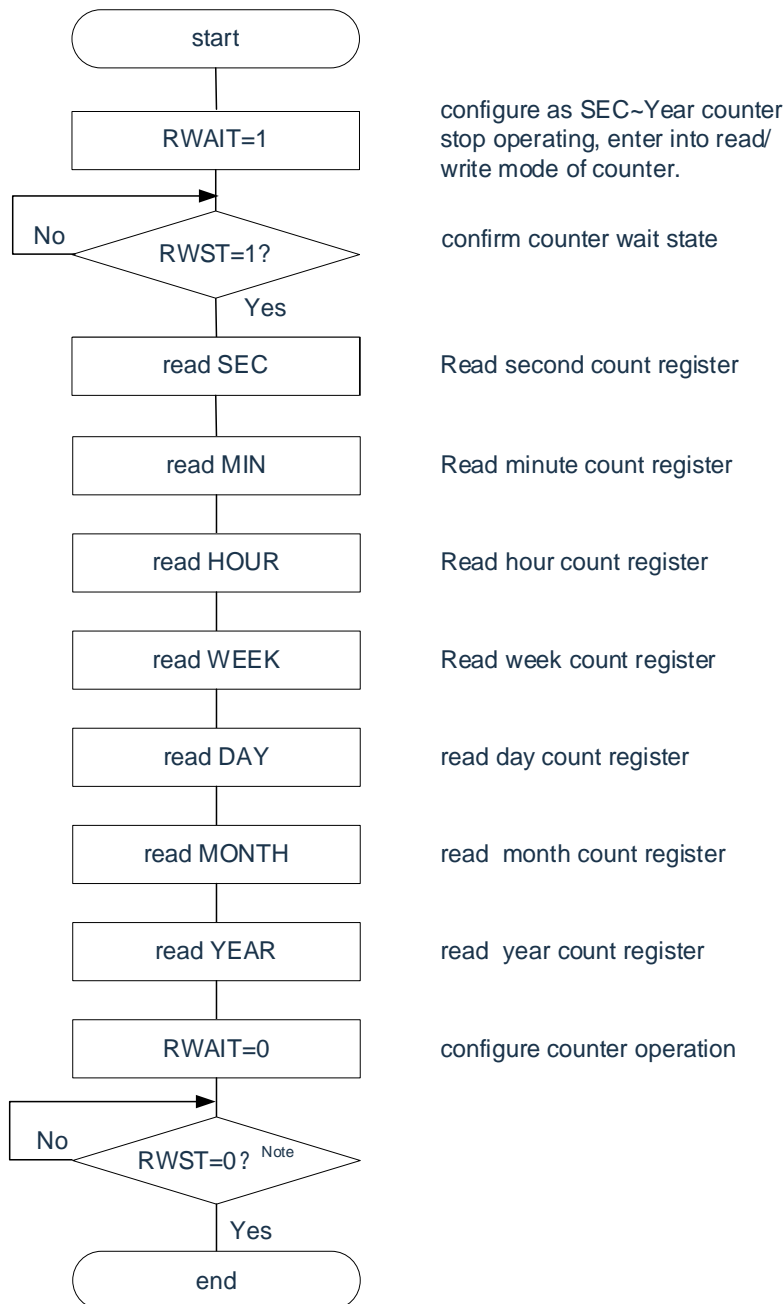
Figure 11-20 transfers the sleep/deep sleep mode steps after the RTCE set to "1"



11.4.3 Read and write to the real-time clock counter

The RWAIT must first be set to "1" and then the counter must be read and written. The RWAIT position must be "0" after reading and writing the counter.

Figure 11-21 The read operation procedure of the real-time clock counter

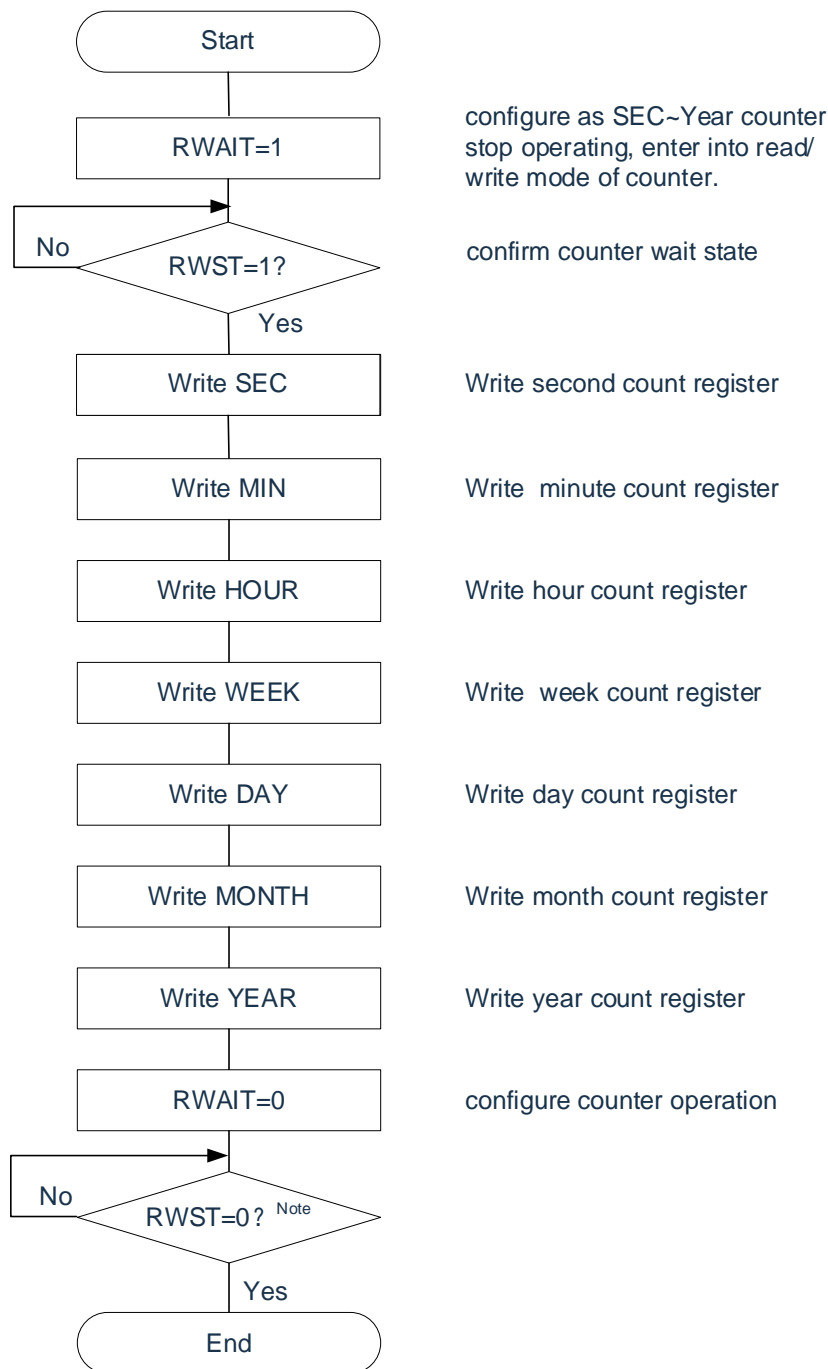


Note That the RWST bit must be confirmed to be "0" before moving to sleep mode.

Note that processing from RWAIT position "1" to RWAIT position "0" must be carried out within 1 second.

Note There is no limit to the order of read operations for seconds/minutes/hours/week/day/month/and year count registers/. It is possible to read only part of the registers without reading all the registers.

Figure 11-22: Read procedure for a real-time clock counter



Note You must confirm that the RWST bit is "0" before moving to SLEEP mode.

Note 1. Processing from RWAIT position "1" to RWAIT position "0" must be carried out within 1 second.

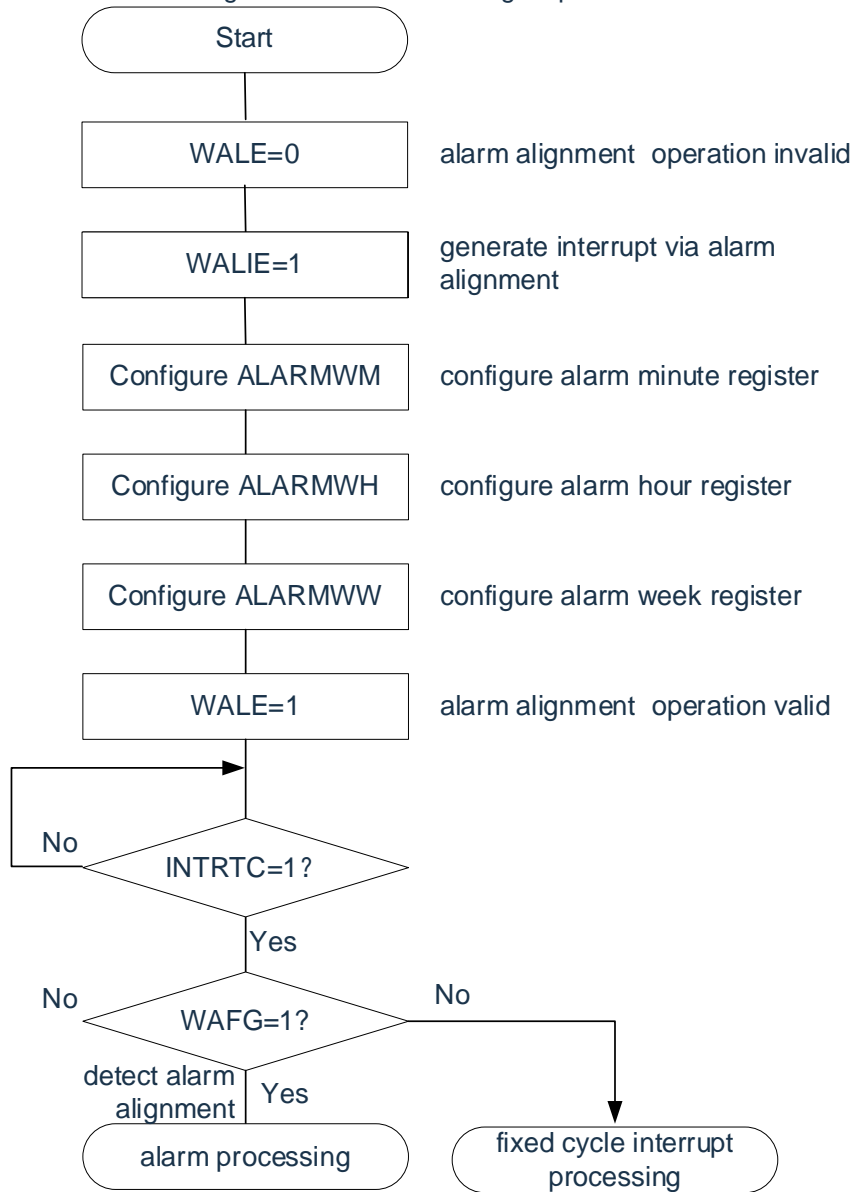
2. To rewrite SEC, MIN, HOUR, WEEK, DAY, MONTH in counter operation (RTCE=1). When the YEAR register is registered, it must be overwritten after the INRTC is set to disable interrupt processing through the interrupt mask flag register, and the WAFG flag and RIFG must be cleared after the rewrite Flag and RTCIF flag.

Note There is no limit to the order of read operations for seconds/minutes/hours/week/day/month/and year count registers/. It is possible to read only part of the registers without reading all the registers.

11.4.4 Alarm settings for the real-time clock

You must first place the WALL position "0" (the alarm does not run invalid) and then set the alarm time.

Figure 11-23 Alarm setting steps

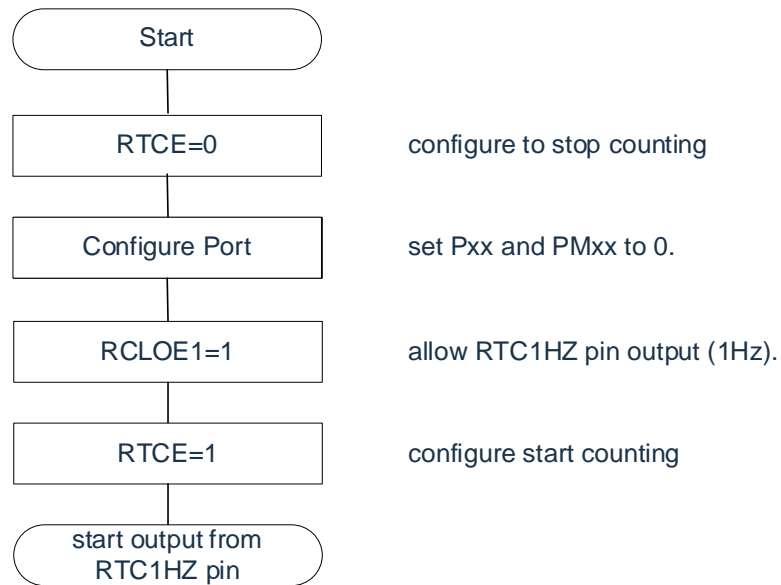


Note 1 There is no limit to the order of write operations for alarm minute registers (ALARMWWM), alarm hour registers (ALARMMWH), and alarm clock week registers (ALARMWW).

2. Fixed cycle interrupts and alarm clock consistent interrupts use the same interrupt source (INTRTC). In the case of using both interrupts, in the event of INTRTC, it is possible to determine which interrupt occurred by confirming the fixed-period interrupt status flag (RIFG) and the alarm clock detection status flag (WAFG) when intRTC occurs.

11.4.5 1Hz output of the real-time clock

Figure 11-24 Setup steps for the 1Hz output



Note 1 The RTCEN position "1" must first be placed in the oscillation stable state of the counting clock (f_{SUB}).

2. Some packages do not support the 1Hz output function of real-time clocks.

11.4.6 An example of clock error correction for a real-time clock

High-precision clock speed correction can be performed by setting the value of the clock error correction register.

Example of the method of calculating the corrected value

The correction value when correcting the count value of the internal counter (16 bits) can be calculated using the following formula. When the correction range is outside the range of -4165.6ppm to 4165.6ppm , the DEV position must be "0".

(The case of DEV=0)

$$\text{Correction Value}^{\text{Note}} = 1 \text{ minute correction count value} = (\text{oscillation frequency target frequency} - 1) \div 32768 \times 60 \times \div 3$$

(The case of DEV=1)

$$\text{Correction Value}^{\text{Note}} = 1 \text{ minute correction count value} = (\text{Oscillation Frequency Target Frequency} - 1) \times 32768 \times 60$$

Note: The correction value is a clock error correction value calculated based on the value of bit12~0 of the clock error correction register (SUBCUD).

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

(Case of F12=1) Correction = $-\{(/F11, /F10, /F9, /F8, /F7, /F6, /F5, /F4, /F3, /F2, /F1, /F0) + 1\} \times 2$

When $(F12 \sim F0) = (*, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, *)$, no correction of clock error is performed. * is "0" or "1".
/F12~F0 is the inverse of each member (when "000000000011", "111111111100").

Note: 1. The correction value is 2,4,6,8,1.....,8186, 8188 or -2,-4,-6,-8,.....,-8186,-8188.

2. The oscillation frequency is the value of the counting clock (fRTC).

Output frequency of the RTC1HZ pin 032768 when clock error correction register is initial value (00H)

3. The target frequency is the frequency corrected using the clock error correction register.

Correction example

Example of correction from 32767.4Hz to 32768Hz (32767.4Hz +18.3ppm).

【Measurement of oscillation frequency】

When the clock error correction register (SUBCUD) is the initial value ("0000H"), the oscillation frequency of each product is measured by outputting a signal of approximately 1Hz from the RTC1HZ pin.

Note For the setup steps of the RTC1Hz output, please refer to "11 1.4.5 1Hz Output of the Real-Time Clock".

【Calculation of correction values】

(When the output frequency of the RTC1HZ pin is 0.9999817Hz).

$$\text{Oscillation frequency} = 327680.9999817 \times \approx 32767.4\text{Hz}$$

Suppose the target frequency is 32768Hz (32767.4Hz+18.3ppm) and DEV=1.

A formula for calculating the correction value when the DEV bit is "1" is applied.

$$\begin{aligned} \text{Correction value} &= 1 \text{ minute correction count value} = \div (\text{oscillation frequency target} \\ &\text{frequency} - 1) \times 32768 \times 60 \\ &= (32767.4 \div 32768 - 1) \times 32768 \times 60 \\ &= -36 \end{aligned}$$

Calculation of the setting value of (F12~F0).

(In the case of a correction value =-36).

Because the correction value is less than 0 (faster), F12=1. Calculated based on correction values (F11~F0).

$$\begin{aligned} -\{(/F11 \sim /F0) + 1\} \times 2 &= -36 \\ (/F11 \sim /F0) &= 17 \\ (/F11 \sim /F0) &= (0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1) \\ (F11 \sim F0) &= (1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0) \end{aligned}$$

Therefore, the correction from 32767.4Hz to 32768Hz (32767.4Hz +18.3ppm) is as follows:

If by DEV=1 and correction value=-36 (bit12~0:1,1,1,1,1,1,1,1,1,1,1,0 by DEV=1,1,1,1,1,0 by DEV=1,1,1,1,0 by DEV=1,1,1,1,1,0.)) to set the correction register, you can correct to 32768Hz (0ppm).

Chapter 12 15-bit interval timer

12.1 Function of 5-bit interval timer

Generate interrupts (INTIT) at any pre-set interval that can be used for wake-up from deep sleep mode.

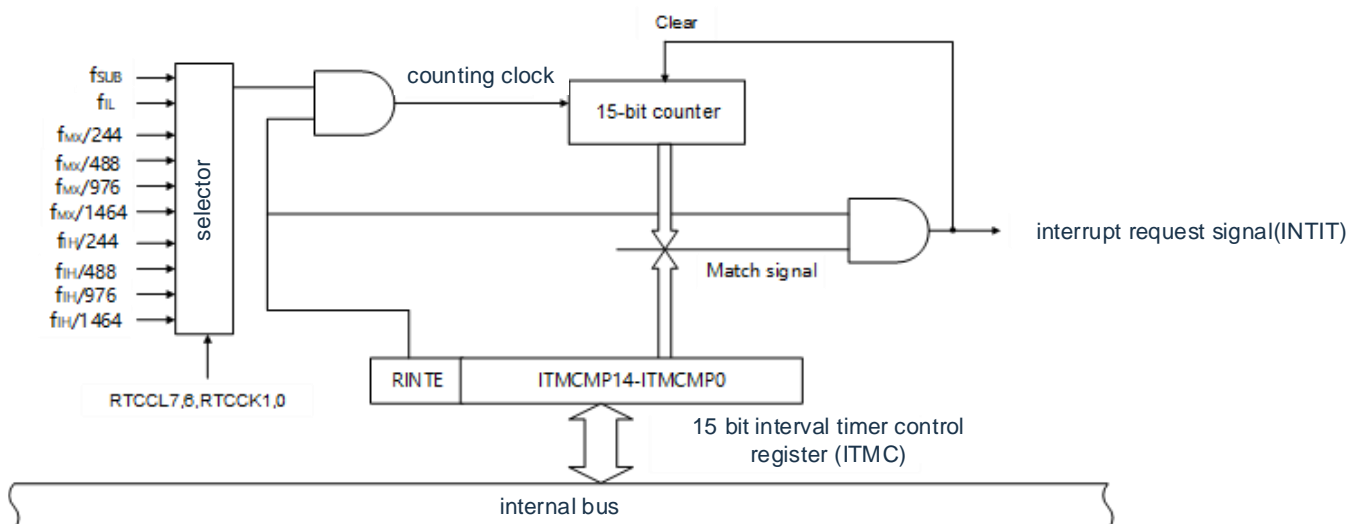
12.2 Structure of the 15-bit interval timer

The 15-bit interval timer consists of the following hardware.

Table 12-1 Structure of the 15-bit interval timer

item	structure
counter	1 5-bit counter
Control registers	Peripheral enable register 0 (PER0).
	Real-time clock selection register (RTCCL).
	1Control Register (ITMC) for 5-bit interval timers

Figure 12-1 Block diagram of a 15-bit interval timer



12.3 control Registers of the 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).
- Control Register (ITMC) for 15-bit interval timers

12.3.1 Peripheral enable register 0 (PER0).

The PER0 register is a register that is set to allow or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use .

To use the 15-bit interval timer, bit7 (RTCEN) must be set to "1". Set the PER0 register with 8-bit memory manipulation instructions . After generating a reset signal, the value of this register becomes "00H".

Figure 12-2 The format of Peripheral enable register0(PER0)

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

symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	-	ADCEN	IICA0EN	SCI1EN	SCI0EN	CAN0IN	TM40EN

RTCEN	Provides control of the input clock for a real-time clock (RTC) and a 15-bit interval timer
0	Stop providing the input clock. • SFR used by real-time clocks (RTCs) and 15-bit interval timers cannot be written. • The real-time clock (RTC) and 15-bit interval timer are reset.
1	An input clock is provided. • Read and write SFR for real-time clock (RTC) and 15-bit interval timers.

12.3.2 Real-time clock selection register (RTCCL).

The real-time clock and the count clock (fRTC) of the 15-bit interval timer can be selected via RTCCL.

Figure 12-3 Format of the Real-Time Clock Selection Register (RTCCL).

Address: 0x4002047C

after reset: 00H R/W

Symbol 7 6 5 4 3 2 1 0

RTCCL	RTCCL7	RTCCL6	RTCCL5	0	0	0	RTCKS1	RTCKS0
-------	--------	--------	--------	---	---	---	--------	--------

RTCCL7	A choice of clock sources for real-time clocks, 15-bit interval timers that count clocks
0	Select the high-speed system clock (f _{MX}).
1	Select the high-speed internal oscillator (fhoco).

RTCKS1	RTCKS0	RTCCL6	RTCCL5	Selection of operating clocks for real-time clocks, 15-bit interval timers for counting clocks
0	0	x	x	Subsystem clock (f _{SUB}).
0	1			Low-speed internal oscillator clock (f _{IL})(WUTMMCK0 must be set to 1).
1	0	0	1	Master clock fmax/fhoco (selected via RTCCL7)/1952
1	0	0	0	Master clock fmax/fhoco (selected via RTCCL7)/1464
1	0	1	0	Master clock fmax/fhoco (selected via RTCCL7)/976
1	1	0	0	Master clock fmax/fhoco (selected via RTCCL7)/488
1	1	1	0	Master clock fmax/fhoco (selected via RTCCL7)/244

12.3.3 Control Register (ITMC) for 15-bit interval timers

This is the register that sets the start and stop of the run of the 15-bit interval timer and the comparison value. Set the ITMC registers with 16-bit memory operation instructions. After generating a reset signal, the value of this register becomes "7FFFH".

Figure 12-4 Format of the 15-bit interval timer control register (ITMC).

Address: 0x40044F50

After reset: 7FFFH R/W

symbol 15

14~0

ITMC	RINTE	ITCMP14 ~ ITCMP0
------	-------	------------------

RINTE	1 5-bit interval timer for operational control
0	Stops the counter from running (clear count).
1	Start the run of the counter.

ITCMP14 ~ ITCMP0	The 15-bit interval timer compares the setting of the value
0001H	These bits produce a x fixed-cycle interrupt that counts clock cycles (ITCMP setting value +1).
• • •	
7FFFH	
0,000H	Prohibit settings.
Example of an interrupt cycle when ITCMP1 4 ~ ITCMP0 is "0 001H" or "7FFFH"	
<ul style="list-style-type: none">ITCMP14 ~ ITCMP0 = 0001H, counting clock: f_{SUB}=32.768kHz 1/32.768[kHz] (1+1) =0.06103515625[ms]x ≈ 61.03[us]ITCMP14 ~ ITCMP0 = 7FFFH, counting clock: f_{SUB}=32.768kHz 1/32.768[kHz] (x32767+1) = 1000[ms]	

note

- To change the RINTE bit from "1" to "0", it must be rewritten after the INTIT is set to disable interrupt processing through the interrupt mask flag register. To restart the run (from "0" to "1"), you must set it to allow interrupt processing after the ITIF flag is cleared.
- The read value of the RINTE bit is reflected after 1 count clock after the RINTE bit is set.
- After moving from sleep mode to normal operating mode, if you want to set the ITMC register and move to sleep mode again, you must confirm that the write value of the ITMC register is reflected or set the ITMC Registers pass through at least 1 count clock before moving to sleep mode.
- To change the settings for ITCMP14~ITCMP0 bits, you must do so in a state where the RINTE bit is "0". However, it is possible to change the ITCMP14~at the same time as changing the RINTE bit from "0" to "1" or from "1" to "0" ITCMP0 bit settings.

12.4 15-bit interval timer operation

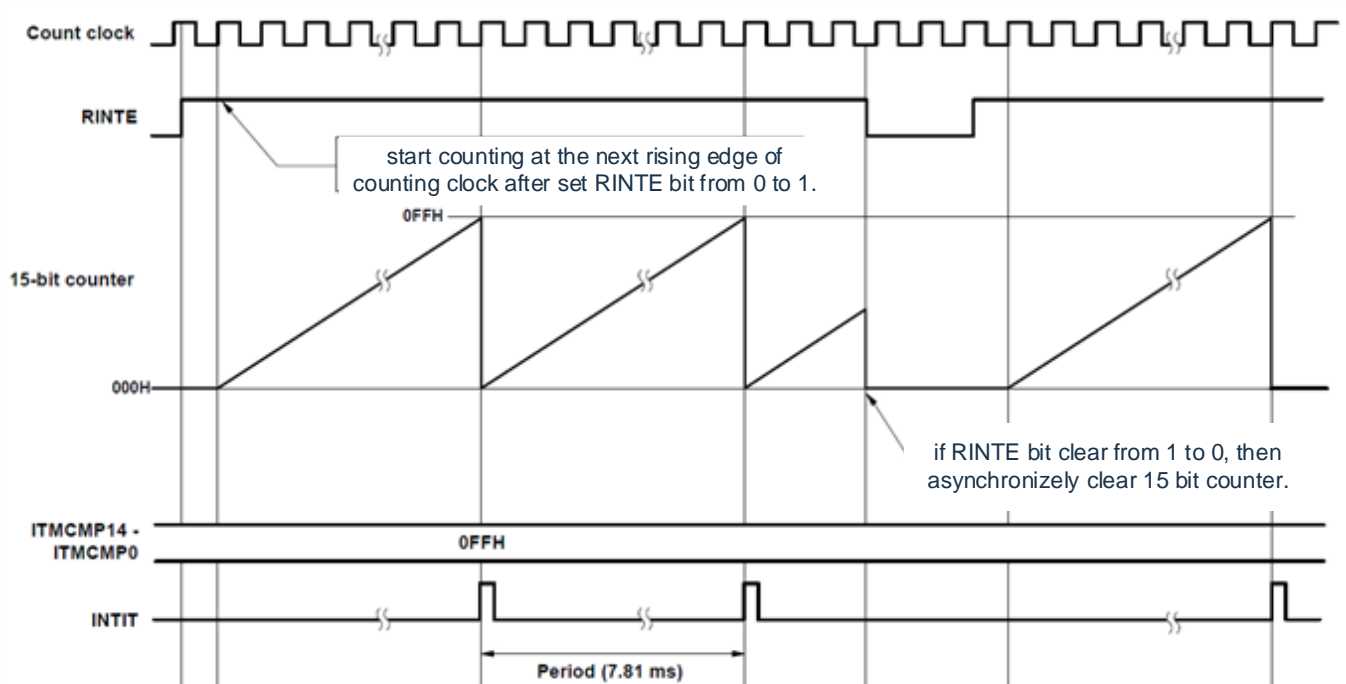
12.4.1 Operation sequence of the 15-bit interval timer

Operation as a 15-bit interval timer for repeatedly generating interrupt requests (INTIT) in intervals of the count value set by ITCMP14 ~ ITCMP0 bits. If you place the RINTE at "1", the 15-bit counter starts counting.

When the 15-bit count value is the same as the itCMP14 ~ ITCMP0 bits setting value, the 15-bit count value is cleared to "0" and the count continues, generating an interrupt request signal (INTIT).

The basic operation of the 15-bit interval timer is shown in Figure 12-5.

Figure 12-5 Operation sequence of a 15-bit interval timer
(ITCMP14 ~ ITCMP0=0FFH, count clock: $f_{SUB}=32.768\text{kHz}$).

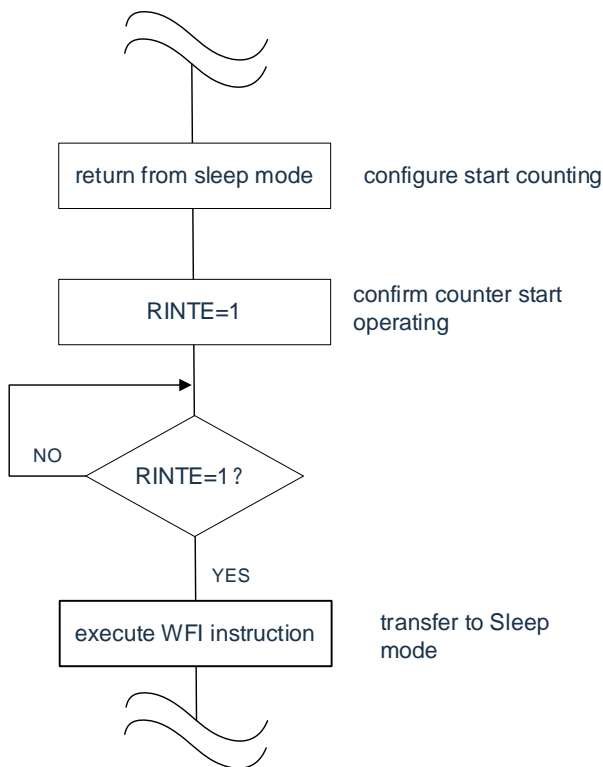


12.4.2 After returning from sleep mode, the operation of the counter begins and then transition to sleep mode again

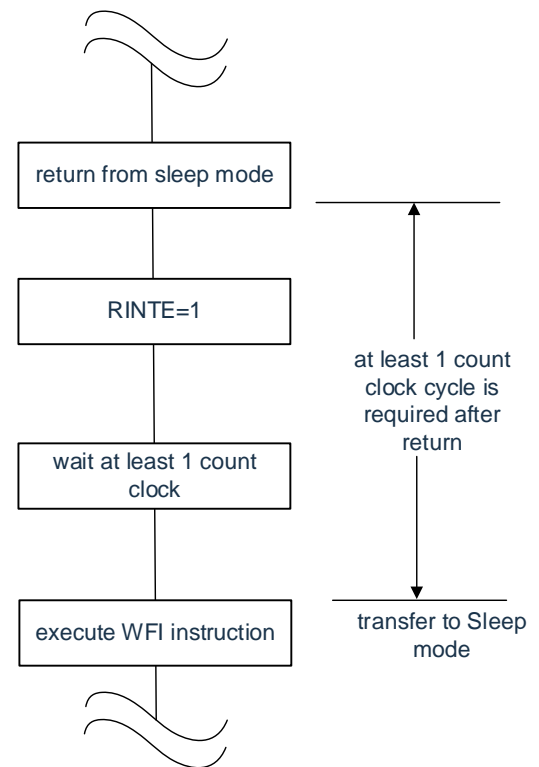
After returning from sleep mode, if you want to transfer the RINTE position "1" and to sleep mode again, you must confirm that the write value of the RINTE bit is reflected after the RINTE position "1" is returned, or at least passed after returning the time of 1 count clock is then transferred to sleep mode.

- After the RINTE position "1", confirm that the RINTE bit changes to "1" by polling and then move to sleep mode (refer to example1 in the figure below).
- Move to sleep mode after at least 1 counting clock time has elapsed since the RINTE set to "1" (refer to Example 2 in the figure below).

Example 1



Example 2



Chapter 13 Clock output/buzzer output control circuitry

13.1 the function of controls circuitry of Clock output/buzzer output

The clock output is the function of outputting the clock to the peripheral IC, and the buzzer output is the function of outputting the buzzer frequency square wave.

It can be selected as a clock output or buzzer output with 1 pin.

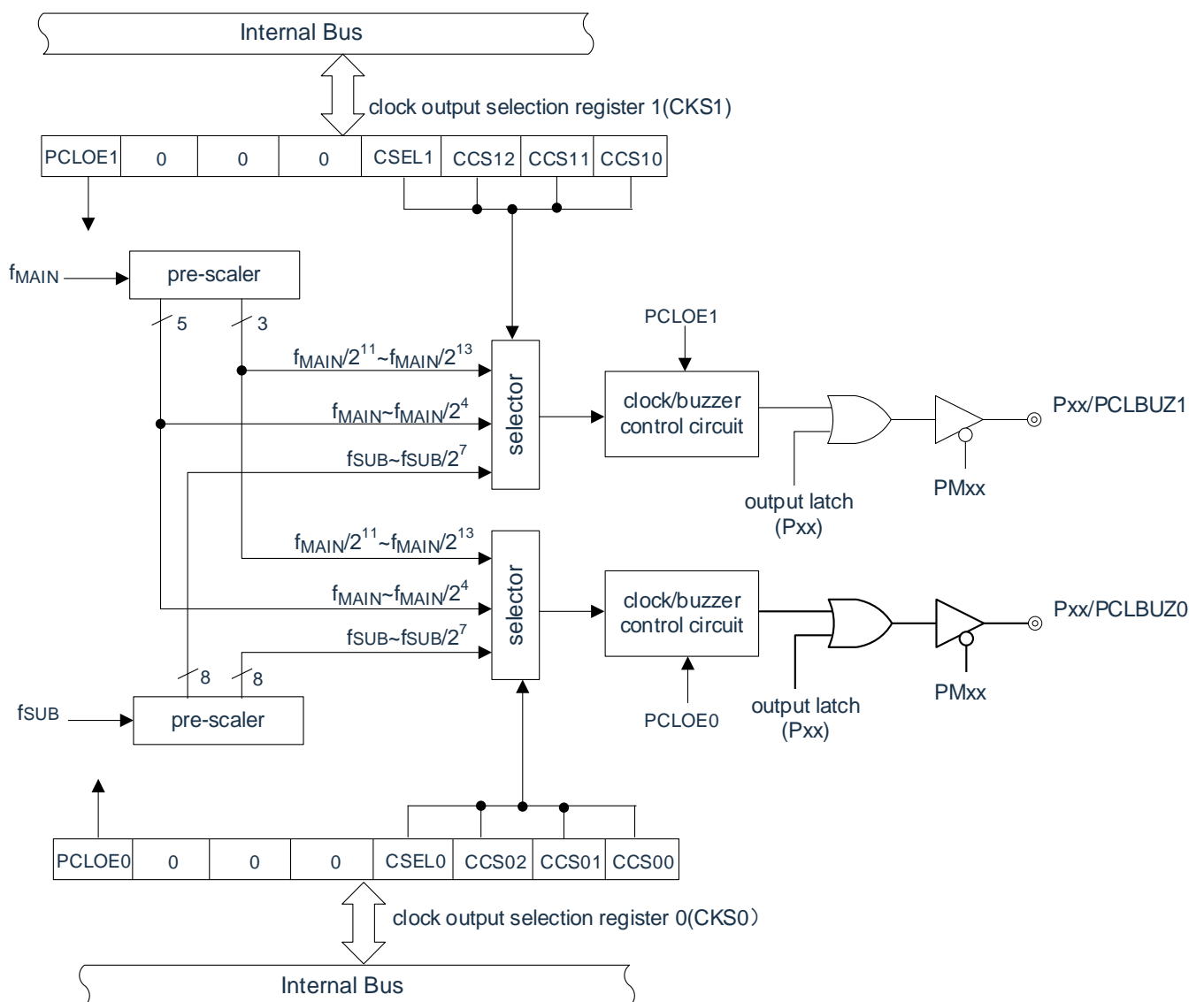
The CLKBUZn pin outputs a clock selected by the clock output select register n (CKSn).

A block diagram of the clock output buzzer output control circuit is shown in Figure 13-1.

Note that the RTCLPC bit of the mode control register (OSMC) provided on the subsystem clock is "1" and the CPU is running on the subsystem clock (f_{SUB}). In SLEEP mode, the subsystem clock (f_{SUB}) cannot be output from the CLKBUZn pin.

Note: n=0, 1

Figure 13-1 Block diagram of the clock output/buzzer output control circuit



Note For the frequencies that can be output from the CLKBUZ0 pin and the CLKBUZ1 pin, please refer to "AC Characteristics".

13.2 Structure of the clock output/buzzer output control circuit

The clock output/buzzer output control circuit consists of the following hardware.

Table 13-1 Structure of the clock output / buzzer output control circuit

Item	structure
Control registers	Clock output select register n (CKSn). Port Mode Register (PMmn) Port Register Pmn).

13.3 control Registers of the clock output/buzzer output control circuitry

13.3.1 Clock output select register n (CKSn).

This is a register that enable or disables the output of the clock output pin or buzzer frequency output pin (CLKBUZn) and sets the output clock.

The clock output of the CLKBUZn pin is selected through the CKSn register. The CKSn register is set via the 8-bit memory operation instruction. After generating a reset signal, the value of this register changes to "00H".

Figure 13-2 The clock output selects the format of register n (CKSn).

Address: 0x40040FA5 (CKS0), 0x40040FA6 (CKS1) after reset: 00H R/W

Symbol 7 6 5 4 3 2 1 0

CKSn	PCLOEn	0	0	0	0	CSELn	CCSn2	CCSn1	CCSn0
------	--------	---	---	---	---	-------	-------	-------	-------

PCLOEn	CLKBUZn pin output enable/disables the designation
0	Disable output (default).
1	Output is allowed.

CSELn	CCSn2	CCSn1	CCSn0		CLKBUZn pin output clock selection			
					f _{MAIN} = 10MHz	f _{MAIN} = 20MHz	f _{MAIN} = 32MHz	f _{MAIN} = 48MHz
0	0	0	0	f _{MAIN}	10MHz ^{Note}	Disable settings of notes	Disable settings of notes	Disable settings of notes
0	0	0	1	f _{MAIN} /2	5MHz	10MHz ^{Note}	16MHz ^{Note}	Prohibit settings ^{Note}
0	0	1	0	f _{MAIN} /2 ²	2.5MHz	5MHz	8MHz	12MHz
0	0	1	1	f _{MAIN} /2 ³	1.25MHz	2.5MHz	4MHz	6MHz
0	1	0	0	f _{MAIN} /2 ⁴	625kHz	1.25MHz	2MHz	3MHz
0	1	0	1	f _{MAIN} /2 ¹¹	4.88kHz	9.77kHz	15.63kHz	23.44kHz
0	1	1	0	f _{MAIN} /2 ¹²	2.44kHz	4.88kHz	7.81kHz	11.72kHz
0	1	1	1	f _{MAIN} /2 ¹³	1.22kHz	2.44kHz	3.91kHz	5.86kHz
1	0	0	0	f _{SUB}	32.768kHz			
1	0	0	1	f _{SUB} /2	16.384kHz			
1	0	1	0	f _{SUB} /2 ²	8.192kHz			
1	0	1	1	f _{SUB} /2 ³	4.096kHz			
1	1	0	0	f _{SUB} /2 ⁴	2.048kHz			
1	1	0	1	f _{SUB} /2 ⁵	1.024kHz			
1	1	1	0	f _{SUB} /2 ⁶	512Hz			
1	1	1	1	f _{SUB} /2 ⁷	256Hz			

Note The output clock must be used within the range of 16MHz. For details, please refer to "AC Features".

Note 1 Switching of the output clock must be performed after the output is set to Disable Output (PCLOEn=0).

- When selecting the main system clock (CSELn=0), if you want to move to deep sleep mode, you must set the PCLOEn to "0" before executing the WFI instruction; When selecting a subsystem clock (CSELn=1), the RTCLPC bit of the mode control register (OSMC) that can be provided on the subsystem clock is "0" and is in deep sleep The clock is output in the mode, so the PCLOEn can be set to "1".
- When the RTCLPC bit of the mode control register (OSMC) provided by the sub-system clock is "1" and the CPU is running in the sub-system clock (f_{SUB}). In sleep mode, the subsystem clock (f_{SUB}) cannot be output from the CLKBUZn pin.

Note 1. n=0, 1

- f_{MAIN} : The main system clock frequency
- f_{SUB} : Subsystem clock frequency

13.3.2 Control Registers of the clock output/buzzer output pin port function

When used as a clock output/buzzer output function, the control registers (port mode register (PMxx) and port register (Pxx)) that are reused with the port function of the object channel must be set. For details, please refer to "2.3.1 Port Mode Register (PMxx)" and "2.3.2 Port Register" (Pxx) ”.

When using the multiplex port of the clock output/buzzer output pin as the clock output/buzzer output, the bit of the port mode register (PMxx) and the position of the port register (Pxx) corresponding to each port must be used “0”.

(Example) When the P140/INTP6/CLKBUZ0 is used as the clock output/buzzer output

Place the PORT mode register 14 at PM140 position "0".

Place the P140 position "0" of port register 14.

13.4 the operation of Clock output/buzzer output controls circuitry

It can be selected as a clock output or buzzer output with 1 pin.

The CLKBUZ0 pin outputs a clock/buzzer selected by clock output select register 0 (CKS0).

The CLKBUZ1 pin outputs a clock/buzzer selected by clock output select register 1 (CKS1).

13.4.1 Operation of the output pins

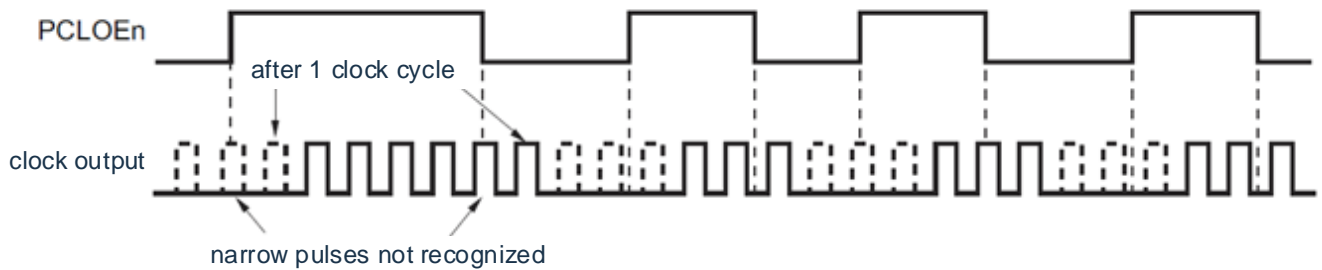
The CLKBUZn pin is output as follows:

- ① The position "0" of the port mode register (PMxx) and the port register (Pxx) that will be used as the port of the CLKBUZ0 pin.
- ② Select bit0 to 3 (CCSn0 ~CCSn2), register select register (CKSn) through the clock output of the CLKBUZn pin CSELn) selects the output frequency (output is disabled).
- ③ Place bit7 (PCLOEn) of the CKSn register at "1" to allow the output of the clock/buzzer.

Note 1 When used as a clock output, the control circuit starts or stops the clock output after allowing or disabling one clock after the clock output (PCLOEn bit). Pulses with narrow widths are not output at this time. The timing of the outputs allowed or stopped by the PCLOEn bits and the timing of the clock outputs is shown in Figure 13-3.

2. n=0, 1

Figure 13-3 clock output timing of the CLKBUZn pin



13.5 Considerations for clock output/buzzer output control circuitry

When the main system clock is selected as the CLKBUZn output (CSELn=0), if 1.5 after the stop output (PCLOEn=0) is set

The output clock of the CLKBUZn pin is shifted to deep sleep mode, and the output width of CLKBUZn is narrowed.

Chapter 14 Watchdog timer

14.1 The function of the watchdog timer

The watchdog timer operates with an option byte (000C0H) to set the count. The watchdog timer operates with a low-speed internal oscillator clock (f_{IL}). Watchdog timers are used to detect out-of-control programs. When a program is detected out of control, an internal reset signal is generated.

The following situations are judged to be out of control of the program.

- When the watchdog timer counter overflows
- When a bit manipulation instruction is performed on the allow register (WDTE) of the watchdog timer
- When writing data other than "ACH" to the WDTE register
- When writing data to the WDTE register during window closing

When reset occurs due to a watchdog timer, set the bit4 (WDTRF) of the reset control flag register (RESF) to "1". For details on RESF registers, refer to Chapter 30 Reset Functions. When 75% of the overflow time is reached +1/2f_{IL}, interval interrupts can be generated.

14.2 Structure of the watchdog timer

Watchdog timers consist of the following hardware.

Table 14-1 watchdog timer

Item	structure
counter	Internal counter (17 bits).
Control registers	The Watchdog Timer's Allowed Register (WDTE).

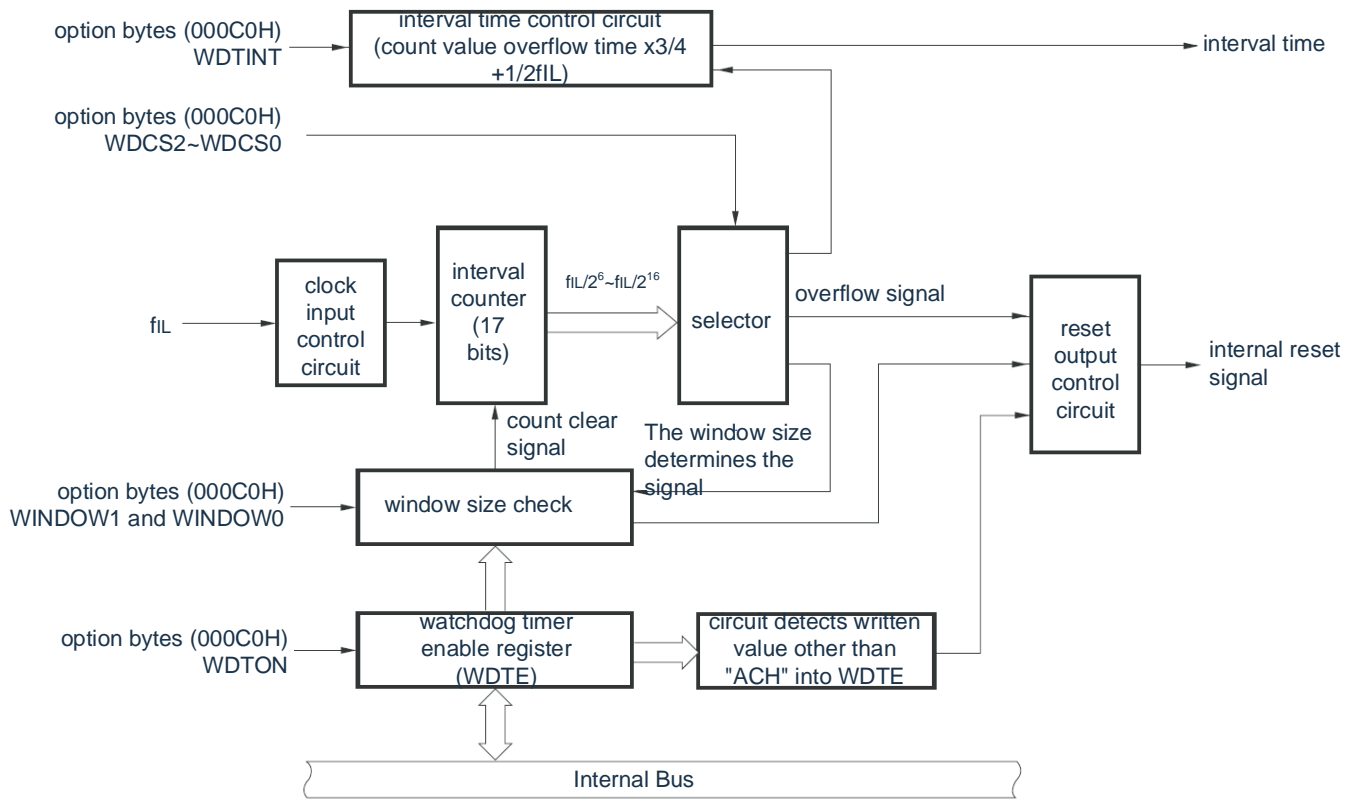
The option bytes control the operation of counters and set overflow times, window opening periods, and interval interrupts.

Table 14-2 Options Bytes and Watchdog Timer Settings

The setting content of the watchdog timer	Option byte (000C0H).
The setting of the interval interrupt of the watchdog timer	bit7 (WDTINT)
Settings during window opening	bit6 and bit5 (WINDOW1, WINDOW0)
Counter run control of the watchdog timer	bit4 (WDTON)
Setting of the overflow time of the watchdog timer	bit3~1 (WDCS2~WDCS0)
Counter run control of the watchdog timer (during sleep).	bit0 (WDSTBYON)

Note: For option bytes, refer to Chapter 33 Option Bytes.

Figure 14-1 Diagram of Watchdog Timer



Note: f_{IL} : The clock frequency of the low-speed internal oscillator

14.3 Control registers of the watchdog timer

The watchdog timer is controlled by the watchdog timer's Allow Register (WDTE).

14.3.1 The Watchdog Timer's enable Register (WDTE).

By writing "ACH" to the WDTE register, clear the watchdog timer counter and start counting again. The WDTE register is set via the 8-bit memory operation instruction. After generating a reset signal, the value of this register changes to the "9AH" or "1AH" ^{note}.

Figure 14-2 Format of the enable register (WDTE) of the watchdog timer

Address: 0x40021001	after reset: 9AH/1AH	Note	R/W								
WDTE	<table><tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>										

Note: The reset value of the WDTE register varies depending on the setpoint of the WDTON bit of the option byte (000C0H). For the watchdog timer to run, the WDTON must be placed at the position "1".

The setpoint of the WDTON bit	The reset value of the WDTE register
0 (Disables the counting run of the watchdog timer).	1AH
1 (enable count run of watchdog timer).	9AH

Note:

1. When writing a value other than "ACH" to the WDTE register, an internal reset signal is generated.
2. When the bit operation instruction is executed on the WDTE register, an internal reset signal is generated.
3. The read value of the WDTE register is "9AH/1AH" (different from the write value ("ACH")).

14.3.2 LockUP Control Register (LOCKCTL) and its Protection Register (PRCR).

The LOCKCTL register is the configuration register for whether the Cortex-M0+ LockUp function causes the watchdog timer to run, and the PRCR is its write-protect register.

Set lockCTL, PRCR registers via 8-bit memory operation instructions.

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

Figure 14-3 LOCKUP control register (LOCKCTL) and the format of 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 the LOCKUP function
0	• LOCKUP does not cause a WDT reset
1	• LOCKUP causes the WDT to reset

Figure 14-3 LOCKUP control register (LOCKCTL) and the format of its protection register (PRCR) (2/2)

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

PRCR	PRTKEY[7:1]	PRCR
------	-------------	------

PRCR	LOCKUP controls register write protection
0	• The LOCKCTL register is not writable
1	• LockCTL register is writable

PRTKEY[7:1]	Write protection for PRCR
78H	• PRCR is writable
other	• PRCR is not writable

14.3.3 WDTCFG Configuration Register (WDTCFG0/1/2/3)

The WDTCFG configuration register is a register that forces the watchdog timer to run.

The WDTCFG register is set via the 8-bit memory operation instruction.

After generating a reset signal, the value of the WDTCFG register changes to "00H".

Figure 14-4 WDTCFG configuration register (WDTCFG0/1/2/3).

Address: 40020408H	After reset: 00HR/W
WDTCFG0	WDTCFG0
Address: 40020409H	After reset: 00HR/W
WDTCFG1	WDTCFG1
Address: 4002040AH	after reset: 00HR/W
WDTCFG2	WDTCFG2
Address: 4002040BH	After reset: 00HR/W
WDTCFG3	WDTCFG3

WDTCFG0	WDTCFG1	WDTCFG2	WDTCFG3	Configuration of the watchdog timer function
0x1A	0x2B	0x3C	0x4D	• The operation of the watchdog timer after reset is determined by the option byte ^{note 1}
other				• Forced to run watchdog timer after reset

Note:

1. Refer to the 34.2 User Options Bytes section for detailed configuration.

14.4 Operation of the watchdog timer

14.4.1 Operational control of the watchdog timer

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

- The bit4 (WDTON) of the option byte (000C0H) must be set to "1" to allow the watchdog timer to run count (after the reset is lifted, the counter starts to run) (see Article 33 for details). Chapter Options Bytes).

WDTON	Watchdog timer counter
0	Disables counting runs (stops counting after de-reset).
1	Allow counting runs (start counting after de-reset).

- The overflow time must be set by bit3~1 (WDCS2~WDCS0) of the option byte (000C0H) (see 14.4.2 and for14.4.2details Chapter 33).
- The window opening period must be set by bit6 and bit5 (WINDOW1, WINDOW0) of the option bytes (000C0H) (see 14.4.2 for details).14.4.2and Chapter 33).

2) After the reset is lifted, the watchdog timer starts counting.

3) After starting counting and before the overflow time set by the option byte, if you write "ACH" to the allowed register (WDTE) of the watchdog timer, clear the watchdog timer and start 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
- When writing data other than "ACH" to the WDTE register

note

- Only when the allow register (WDTE) of the watchdog timer is written for the first time after the reset is unchecked, regardless of the window opening period, as long as the WDTE is written at any time before the overflow time, the watchdog timer is cleared and the count is restarted.
- From writing "ACH" to the WDTE register to clearing the watchdog timer counter, it is possible to generate an error of up to 2 f_{IL} clocks.
- The watchdog timer can be cleared before the count value overflows.
- As shown below, the watchdog timer operates in sleep or deep sleep mode depending on the setpoint of bit0 (WDSTBYON) of the option byte (000C0H).

	WDSTBYON=0	WDSTBYON=1
Sleep mode	Stop the watchdog timer from running.	Continue watchdog timer operation.
Deep sleep mode		

When the WDSTBYON bit is "0", restart the watchdog timer count after the sleep or deep sleep mode is lifted. At this point, the counter is cleared to "0" and the count begins.

When running on the X1 oscillating clock after deactivating deep sleep mode, the CPU starts running after the oscillation settling time has elapsed.

If the time from the time from the release of deep sleep mode to the timepiece overflow of the watchdog timer is short, a return will occur during the oscillation stabilization time. Therefore, after de-deep sleep mode by interval interrupt, if you want to run with the X1 oscillation clock and clear the watchdog timer, because the watchdog timer is cleared after the oscillation stabilization time has elapsed, you must consider this situation to set the overflow time.

14.4.2 Setting of the watchdog timer overflow timer

Set the overflow time of the watchdog timer by bit3~1 (WDCS2~WDCS0) of the option byte (000C0H).

In the event of an overflow, an internal reset signal is generated. If you write "ACH" to the watchdog timer's allow register (WDTE) during window opening before the overflow time, the count is cleared and the count is restarted. The overflow time that can be set is as follows.

Table 14-3 Watchdog timer overflow timer settings

WDCS2	WDCS1	WDCS0	Overflow time of the watchdog timer ($f_{IL}=20\text{kHz}(\text{MAX.})$ case).
0	0	0	$2^6/f_{IL}$ (3.2ms)
0	0	1	$2^7/f_{IL}$ (6.4ms)
0	1	0	$2^8/f_{IL}$ (12.8ms)
0	1	1	$2^9/f_{IL}$ (25.6ms)
1	0	0	$2^{11}/f_{IL}$ (102.4ms)
1	0	1	$2^{13}/f_{IL}$ (409.6ms)
1	1	0	$2^{14}/f_{IL}$ (819.2ms)
1	1	1	$2^{16}/f_{IL}$ (3276.8ms)

Note: f_{IL} : The clock frequency of the low-speed internal oscillator.

14.4.3 The setting during which the watchdog timer window is open

Set the window opening period of the watchdog timer by bit6 and bit5 (WINDOW1, WINDOW0) of the option bytes (000C0H). The window outline is as follows:

- If you write "ACH" to the watchdog timer's allow register (WDTE) while the window is open, clear the watchdog timer and start counting again.
- During window shutdown, even if you write "ACH" to the WDTE register, an anomaly is detected and an internal reset signal is generated.

Note: Only when writing the WDTE register for the first time after unsheathing, regardless of the window opening period, as long as the WDTE is written at any time before the overflow time, the watchdog timer is cleared and the count is restarted.

The window opening period that can be set is as follows.

Table 14-4 Settings during the opening of the watchdog timer window

WINDOW1	WINDOW0	Watchdog timer during window opening
0	-	Disable settings
1	0	75%
1	1	100%

Note: When the bit0 (WDSTBYON) of the option byte (000C0H) is "0", it is independent of the values of the WINDOW1 bit and the WINDOW0 bit, The window is 100% open.

Note: When the overflow time is set to 29/f_{IL}, the window closing time and opening time are as follows.

	Settings during window opening	
	75%	100%
Window closing time	0~12.8ms	not
Window open time	12.8~25.6ms	0~25.6ms

< is the period during which the window is open75%>

- Spillover time:
 $2^9/f_{IL}(\text{MAX.})=2^9/20\text{kHz}(\text{MAX.})=25.6\text{ms}$
- Window close time:
 $0\sim 2^9/f_{IL}(\text{MIN.})(1-0. \times 75)=0\sim 2^9/10\text{kHz}0. \times 25=0\sim 12.8\text{ms}$
- Window open time:
 $2^9/f_{IL}(\text{MIN.})(1-0. \times 75)\sim 2^9/f_{IL}(\text{MAX.})=12.8\sim 25.6\text{ms}$

14.4.4 Setting of watchdog timer interval interrupts

By setting the bit7 (WDTINT) of the option byte (000C0H), an interval interrupt (INTWDTI) can be generated when the overflow time is reached at $75\% + 1/2f_{IL}$.

Table 14-5 Watchdog timer interval interrupt settings

WDTINT	Watchdog timer interval interrupts the use/non-use
0	Interval interrupts are not used.
1	Interval interruptions occur when $75\% + 1/2f_{IL}$ of the overflow time is reached.

Note: When running on the X1 oscillating clock after deactivating deep sleep mode, the CPU starts running after the oscillation settling time has elapsed.

If the time from the time from the release of deep sleep mode to the timepiece overflow of the watchdog timer is short, a return will occur during the oscillation stabilization time. Therefore, after de-deep sleep mode by interval interrupt, if you want to run with the X1 oscillation clock and clear the watchdog timer, because the watchdog timer is cleared after the oscillation stabilization time has elapsed, you must consider this situation to set the overflow time.

Note: Continue counting even after intWDTI is generated (continue until "ACH" is written to the allowed register (WDTE) of the watchdog timer). If you do not write "ACH" to the WDTE register before the overflow time, an internal reset signal is generated.

14.4.5 Operation of the watchdog timer during LOCKUP

When the lockup control register lockcTL lockup_rst bit is set to 1, once the kernel enters the LOCKUP state, the low-speed internal oscillator begins to vibrate, the watchdog timer automatically starts running, and the overflow time control bit (WDCS2~WDCS0) is set to 3'b010, that is, the overflow time is set to 12.8ms.

14.4.6 WDTCFG is not configured when the watchdog timer is running

When WDTCFG is not configured, the watchdog timer automatically starts running, and the overflow time is determined by the overflow time control bit (WDCS2~WDCS0) in the option byte.

Chapter 15 A/D converter

15.1 Functions of the A/D converter

The A/D converter is a converter that converts analog inputs to digital values and supports A/D conversion of up to 31 analog channels (28 external pin input channels and 3 internal channels). The number of external analog input channels for the A/D converter is related to each product as shown in the following table.

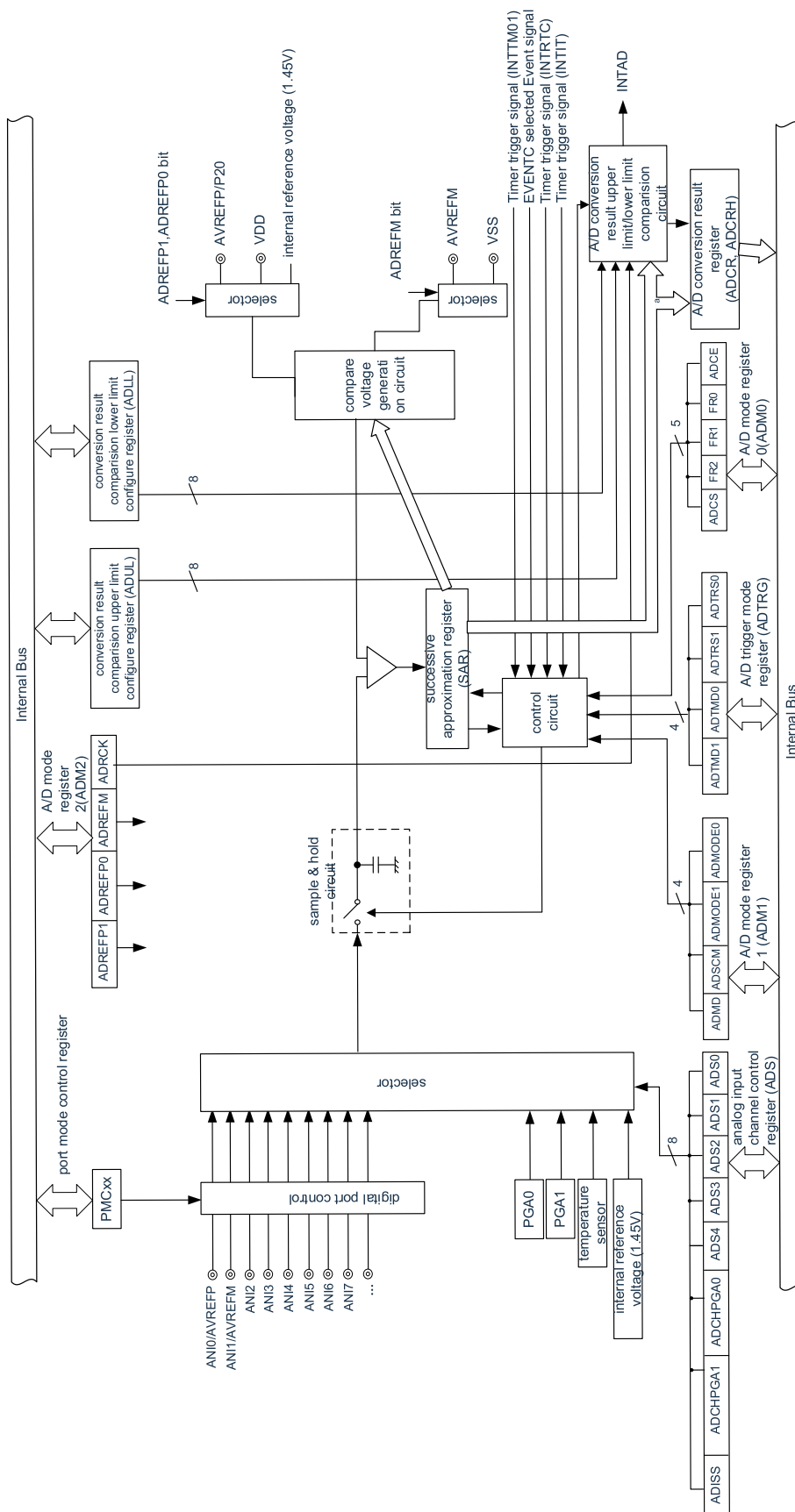
Number of pins	48 pins	64 pins	80 pins	100 pins
The number of analog input channels	15ch	16ch	21ch	28ch
	(ANI0~ANI12) (ANI14~ANI 15)	(ANI0~ANI15)	(ANI0~ANI20)	(ANI0~ANI27)

The A/D converter supports A/D conversion with 12-bit resolution. Select 1 channel of analog signal from ANIx, PGA0, PGA1 and built-in temperature sensors for A/D with 12-bit resolution Transfer. For every A/D conversion at the end of the A/D conversion, an Interrupt Request (INTAD) is generated. A/D conversion of each channel can be performed in single-shot, continuous, scan mode.

Various A/D conversion modes are set by the combination of modes described below.

Trigger mode	Software triggered	Start the conversion with software operations.
	Hardware triggers no-wait mode	Start the conversion by detecting hardware triggers.
	Hardware triggers wait mode	In the switch standby state of the power supply, the power is turned on by detecting the hardware trigger, and the conversion automatically starts after the A/D power supply stable wait time.
Channel selection mode	Select a mode	Select 1 channel of analog input for A/D conversion.
	Scan mode	A/D conversion of 2/3/4 channel analog inputs sequentially. Consecutive 2/3/4 channels from ANI0 to ANI15 can be selected as analog inputs.
Conversion mode	Single-shot conversion mode	Perform 1 A/D conversion on the selected channel.
	Continuous transition 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 (fAD) by default.

Figure15-1 Block Diagram of A/D converter



15.2 Control registers of the A/D converter

The registers that control the A/D converter are as follows:

Register base address: CSC_BASE=4002_0420H; ADC_BASE=4004_5000H; PORT_BASE=4004_0000H

Register name	Register description	R/W	Reset the value	Register address
PER0	Peripheral enable register 0	R/W	00H	CSC_BASE+20H
ADM0	The mode register of the A/D converter is 0	R/W	00H	ADC_BASE+00H
ADM1	Mode register 1 for A/D converters	R/W	00H	ADC_BASE+02H
ADM2	Mode register 2 for A/D converters	R/W	00H	ADC_BASE+04H
ADTRG	Trigger mode registers for A/D converters	R/W	00H	ADC_BASE+06AH
ADS	Analog input channels specify registers	R/W	00H	ADC_BASE+08AH
ADLL	The conversion result compares the lower limit value to set the register	R/W	00H	ADC_BASE+0AH
ADULTS	The conversion result compares the upper limit value to set the register	R/W	00H	ADC_BASE+0BH
ADNSMP	The sample time control register of the A/D converter	R/W	0dH	ADC_BASE+0CH
ADCR	1 2-bit A/D conversion result register	R	0,000H	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	Charge-discharge control registers for A/D converters	R/W	00H	ADC_BASE+11H
ADSMPWAIT	The sampling time of the A/D converter extends the control registers	R/W	00H	ADC_BASE+15H
ADFLG	A/D hardmode status register	R	00H	ADC_BASE+16H
PMC0	Pin mode control register 0	R/W	FFH	PORT_BASE+60H
PMC1	Pin mode control register 1	R/W	FFH	PORT_BASE+61H
PMC2	Pin mode control register 2	R/W	FFH	PORT_BASE+62H
PMC10	Pin mode control register 10	R/W	FFH	PORT_BASE+6AH
PMC12	Pin mode control register 12	R/W	FFH	PORT_BASE+6CH
PMC14	Pin mode control register 14	R/W	FFH	PORT_BASE+6EH
PMC15	Pin mode control registers 15	R/W	FFH	PORT_BASE+6FH

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

15.2.1 Peripheral enable register 0 (PER0).

The PER0 register is a register that is set to allow or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use.

To use the A/D converter, bit5 (ADCEN) must be set to "1".

The PER0 register is set via the 8-bit memory operation instructions.

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

Figure15-2 The format of the Peripheral enable register 0 (PER0).

Reset value: 00H

R/W

	7	6	5	4	3	2	1	0
PER0	RTCEN	-	ADCEN	IICA0EN	SCI1EN	SCI0IN	CAN0EN	TM40EN

ADCEN	Control of the input clock of the A/D converter
0	Stop providing the input clock. <ul style="list-style-type: none"> • Cannot write SFR used by A/D converters. • The A/D converter is in a reset state.
1	An input clock is provided. <ul style="list-style-type: none"> • Can read and write SFR used by A/D converters.

Note1: To set the A/D converter, you must first set the following registers in the ADCEN bit "1". When the ADCEN bit is "0", the value of the control register of the A/D converter is the initial value, ignoring the write operation (Pin Mode Control Register (PMCxx)) except).

- Mode register 0 (ADM0) for A/D converters
- Mode register 1 (ADM1) for A/D converters
- Mode register 2 (ADM2) for A/D converters
- Trigger Mode Register (ADTRG) for A/D converters
- Analog Input Channel Specified Register (ADS).
- Conversion results compare the lower limit value of the setting register (ADLL).
- Conversion results compare upper limit value set register (ADUL).
- A/D 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 Charge and Discharge Control Register (ADNDIS).
- A/D Sampling Time Extended Control Register (ADSMPWAIT).
- A/D Hard Module Status Register (ADFLG).

15.2.2 The mode register 0 (ADM0) of the A/D converter

Registers for setting the A/D conversion clock, conversion start or stop. The ADM0 register is set via the 8-bit memory operation instruction.

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

Figure15-3 Format of the mode register 0 (ADM0) of A/D converter

Reset value: 00H
R/W

	7	6	5	4	3	2	1	0
ADM0	ADCS	0	FR2	FR1	FR0	0	0	ADCE

ADCS	Control of A/D conversion operation
0	Stop the conversion from running. [When reading]. Stops the transition run/standby state
1	Enable the conversion to run. [When reading]. When the software triggers the mode: Convert the running state When the hardware triggers the wait mode: the A/D power supply waits for the steady state + transition operation state

ADCE	Operating control of the A/D voltage comparator ^{Note 2}
0	Stop the operation of the A/D voltage comparator.
1	Enable operation of the A/D voltage comparator.

Note1: For details on FR2-FR0 bit, SHT1-SHT0-bit, and A/D conversion, refer to Table15-3 Selection of A/D conversion times (1/2).".

Note2: The A/D converter takes 1us settling time to start operating. In software-triggered mode or hardware-triggered no-wait mode, at least 1us time has elapsed after placing the ADCE position "1", and then the ADCS position will be placed "1", the result of this conversion is valid. If the wait time is less than 1us and the ADCs position is "1", the result of this conversion must be ignored. In the hardware trigger wait mode, the design guarantees a wait time of 1 us.

Note:

1. The FR2~FR0 bit must be changed under the transition stop state ADCS=0.
2. Disable the setting of ADCS=1 and ADCE=0.
3. Disable the 8-bit operation instruction to set the status of ADCS=0 and ADCE=0 to ADCS=1 and ADCE=1. Setup must be made by following the steps 15.5"Setup Flowchart for the 15.5 A/D Converter".

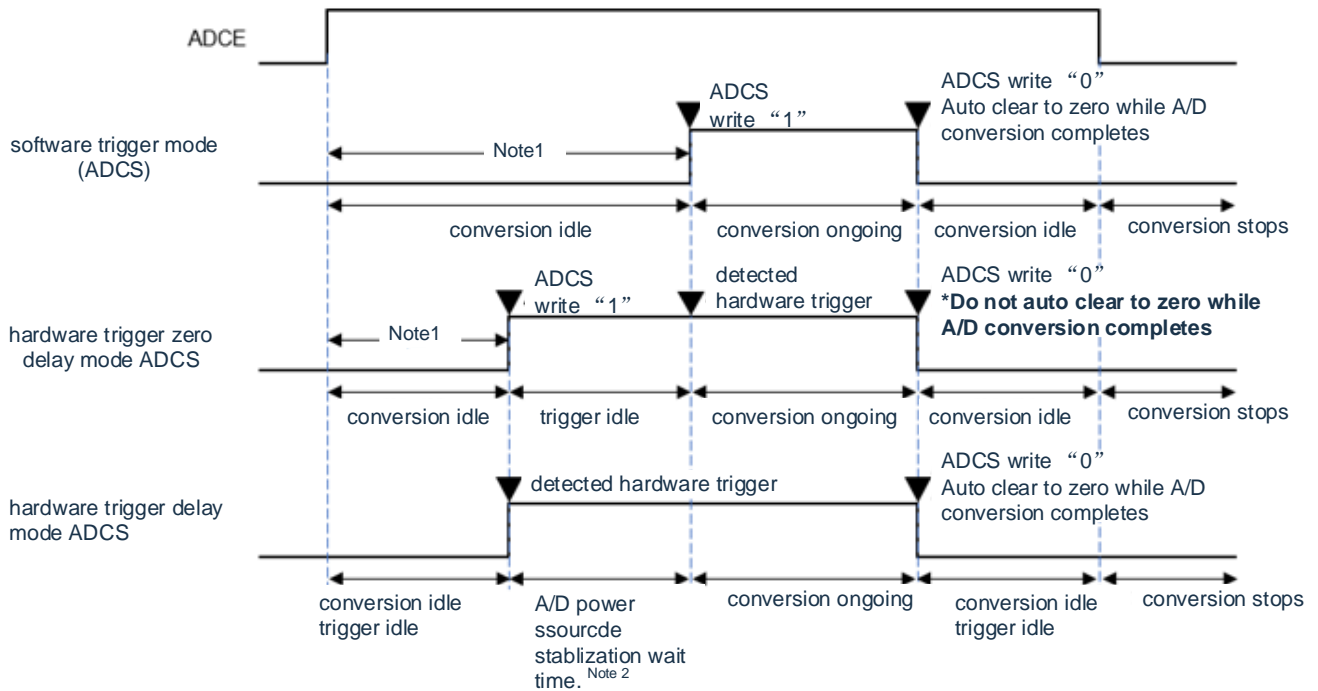
Table15-1 Settings for ADCS bits and ADCCE bits

ADCS	ADCE	The A/D conversion runs
0	0	The transition stopped state
0	1	Transition standby
1	0	Prohibit settings.
1	1	Transition run status

Table15-2 ADCS bits for positioning and clearing conditions

A/D conversion mode			Placement conditions	Clear the condition
Software triggered	Select a mode	Continuous transition mode	When writing to the ADCS bit "1"	When writing "0" to the ADCS bit
		Single-shot conversion mode		<ul style="list-style-type: none"> • When writing "0" to the ADCS bit • Automatically clear "0" at the end of A/D conversion.
	Scan mode	Continuous transition mode		When writing "0" to the ADCS bit
		Single-shot conversion mode		<ul style="list-style-type: none"> • When writing "0" to the ADCS bit • When the 4-channel conversion of the setting ends, the "0" is automatically cleared.
Hardware triggers no-wait mode	Select a mode	Continuous transition mode	When writing to the ADCS bit "1"	When writing "0" to the ADCS bit
		Single-shot conversion mode		<ul style="list-style-type: none"> • When writing "0" to the ADCS bit
	Scan mode	Continuous transition mode		When writing "0" to the ADCS bit
		Single-shot conversion mode		<ul style="list-style-type: none"> • When writing "0" to the ADCS bit
Hardware triggers wait mode	Select a mode	Continuous transition mode	When the input hardware is triggered	When writing "0" to the ADCS bit
		Single-shot conversion mode		<ul style="list-style-type: none"> • When writing "0" to the ADCS bit • Automatically clear "0" at the end of A/D conversion.
	Scan mode	Continuous transition mode		When writing "0" to the ADCS bit
		Single-shot conversion mode		<ul style="list-style-type: none"> • When writing "0" to the ADCS bit • When the 4-channel conversion of the setting ends, the "0" is automatically cleared.

Figure15-4 Motion state Diagram while using the various modes of A/D



Note1: In software-triggered mode or hardware-triggered no-wait mode, the time to rise from the ADCE bit to the ADCS bit needs to be at least 1us (TBD) to stabilize the internal circuitry.

Note2: In hardware trigger wait mode, the A/D power settling time of 1us is guaranteed by design.

Note:

- 1 When you want to use the hardware trigger wait mode, disable the ADCs position "1" (automatically switch to "1" when a hardware trigger signal is detected). However, in order to set the standby state for A/D transition, the ADCS position can be "0".
- 2 The ADCE bit must be rewritten when the ADCS bit is "0" (stop transition/transition standby).
- 3 In order to end the A/D conversion, the hardware trigger interval must be set to at least the following times:

When hardware triggers no wait mode: $2 f_{CLK} \text{ clocks} + \text{A/D conversion time}$

When hardware triggers wait mode: $2 f_{CLK} \text{ clock} + \text{A/D power supply settling wait time} + \text{A/D conversion time}$

Note f_{CLK} : Cpu/peripheral hardware clock frequency

Table15-3 Selection of A/D conversion times (1/2).

(1) No A/D power settling wait time (software trigger mode/hardware trigger no wait mode).

The mode of the A/D converter Register 0 (ADM0)			The mode of the A/D converter Register 1 (ADM1).		mode	Converts the frequency of the clock ADCLK	1 conversion time for 2-bit resolution	
FR2	FR1	FR0	ADMODE[1]	ADMOD[0]			The number of conversion clocks	Conversion time
0	0	0	0	0	High-speed transform mode	$f_{CLK}/32$	45 ADCLK (Number of sample clocks: 13.5 ADCLKs).	$1440/f_{CLK}$
0	0	1				$f_{CLK}/16$		$720/f_{CLK}$
0	1	0				$f_{CLK}/8$		$360/f_{CLK}$
0	1	1				$f_{CLK}/4$		$180/f_{CLK}$
1	0	0				$f_{CLK}/2$		$90/f_{CLK}$
1	0	1				$f_{CLK}/1$		$45/f_{CLK}$
0	0	0	1	1	Low current mode	$f_{CLK}/32$	54 ADCLK (Number of sample clocks: 13.5 ADCLKs).	$1728/f_{CLK}$
0	0	1				$f_{CLK}/16$		$864/f_{CLK}$
0	1	0				$f_{CLK}/8$		$432/f_{CLK}$
0	1	1				$f_{CLK}/4$		$216/f_{CLK}$
1	0	0				$f_{CLK}/2$		$108/f_{CLK}$
1	0	1				$f_{CLK}/1$		$54/f_{CLK}$

Note 1 When you want to rewrite the FR2~FR0 bits and the ADMODE [1:0] bits to different data, you must be in the transition stop state (ADCS=0) under .

Note f_{CLK} : Cpu/peripheral hardware clock frequency

Selection of A/D conversion time (2/2).

(2) There is an A/D power supply settling wait time (hardware trigger wait mode Note 1).

The mode of the A/D converter Register 0 (ADM0).			The mode of the A/D converter Register 1 (ADM1).		mode	The frequency of the conversion clock ADCLK (f_{AD}).	A/D power supply Wait steadily Time	The number of conversion clocks	The A/D power supply is stable Wait time + Conversion time
FR2	FR1	FR0	ADM0E[1]	ADM0D[0]					
0	0	0	0	0	High-speed transform mode	$f_{CLK}/32$	1us	45 ADCLK (Number of sample clocks: 13.5 ADCLKs).	$1\mu s + 1440/f_{CLK}$
0	0	1				$f_{CLK}/16$			$1\mu s + 720/f_{CLK}$
0	1	0				$f_{CLK}/8$			$1\mu s + 360/f_{CLK}$
0	1	1				$f_{CLK}/4$			$1\mu s + 180/f_{CLK}$
1	0	0				$f_{CLK}/2$			$1\mu s + 90/f_{CLK}$
1	0	1				$f_{CLK}/1$			$1\mu s + 45/f_{CLK}$
0	0	0	1	1	Low current mode	$f_{CLK}/32$	1us	54 ADCLK (Number of sample clocks: 13.5 ADCLKs).	$1\mu s + 1728/f_{CLK}$
0	0	1				$f_{CLK}/16$			$1\mu s + 864/f_{CLK}$
0	1	0				$f_{CLK}/8$			$1\mu s + 432/f_{CLK}$
0	1	1				$f_{CLK}/4$			$1\mu s + 216/f_{CLK}$
1	0	0				$f_{CLK}/2$			$1\mu s + 108/f_{CLK}$
1	0	1				$f_{CLK}/1$			$1\mu s + 54/f_{CLK}$

Note1: In continuous transition mode, the A/D power supply stabilization wait time occurs only after the first hardware trigger is detected.

Note:

1. To rewrite fr2~FR0 bits and ADMODE [1:0] bits into different data, it must be done in the conversion stop state (ADCS=0).
2. The conversion time in the hardware trigger wait mode includes the A/D power supply stabilization wait time after the hardware trigger is detected.

Note f_{CLK} : Cpu/peripheral hardware clock frequency

15.2.3 The mode register 1 (ADM1) of the A/D converter

This is the register that sets the A/D conversion mode.

The ADM1 register is set via the 8-bit memory operation instruction.

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

Figure15-5 mode register 1 (ADM1) of the A/D converter

Reset value: 00H

R/W

	7	6	5	4	3	2	1	0
ADM1	ADMD	SMODE1	SMODE0	0	ADSCM	0	ADMODE1	ADMODE0

ADMD	A/D conversion channel selection mode settings
0	Select a mode
1	Scan mode

SMODE1	SMODE0	The number of channels cycled when scanning the pattern
0	0	4-channel sweep
0	1	3-channel sweep
1	0	2-channel sweep
other		Set Prohibit

ADSCM	The setting of the A/D conversion mode
0	Continuous transition mode
1	Single-shot conversion mode

ADMODE1	ADMODE0	A/D conversion mode
0	0	High-speed transform mode
1	1	Low current mode
other		Prohibit settings

Note:

1. To rewrite the ADM1 register, it must be done in the transition stop state (ADCS=0, ADCE=0).
2. In order to end the A/D conversion normally, the hardware trigger interval must be set to at least the following time:
When hardware triggers no wait mode: $2 f_{CLK}$ clocks + A/D conversion time
When hardware triggers wait mode: $2 f_{CLK}$ clock + A/D power supply settling wait time + A/D conversion time

Note 1. f_{CLK} : Cpu/peripheral hardware clock frequency

15.2.4 The mode register 2 (ADM2) of the A/D converter

Set the ADM2 registers via the 8-bit memory operation instructions.

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

Figure15-6 mode register 2 (ADM2) of A/D converters (1/3).

Reset value: 00H

R/W

	7	6	5	4	3	2	1	0
ADM2	ADREFP1	HOMEPO	ADREFM	0	ADRCK	0	CHRDE	0

ADREFP1	ADREFP0	Selection of the positive (+) reference for the A/D converter
0	0	Supplied by V_{DD} .
0	1	Supplied by P20/AV _{REFP} /ANI0.
1	0	Supplied by the internal reference voltage of the A/D converter.
1	1	Prohibit settings

ADREFM	Selection of the negative (–) reference for the A/D converter
0	Supplied by V_{SS} .
1	Supplied by P21/AV _{REFM} /ANI1.

ADRCK	Checking of the upper and lower values of the conversion results
0	When the ADLL register ≤ the ADCR register ≤ the ADUL register (AREA1), an interrupt
1	When ADCRRegister < ADLLRegisters (AREA2orADULTSRegister < ADCRRegisters (AREA3), an interrupt signal is generated (INTAD) .
The range of the interrupt signal (INTAD) of AREA1 to AREA3 is shown in the following figure.	

CHRDE	The A/D converter scans the mode when the output of the channel id is enabled
0	When scanning mode, the channel number is not identified in the conversion results
1	In scan mode, the high four bits of the conversion result ([15:12] of the ADCR register) are the channel numbers for this result

value in ADCR register
(A/D conversion result)

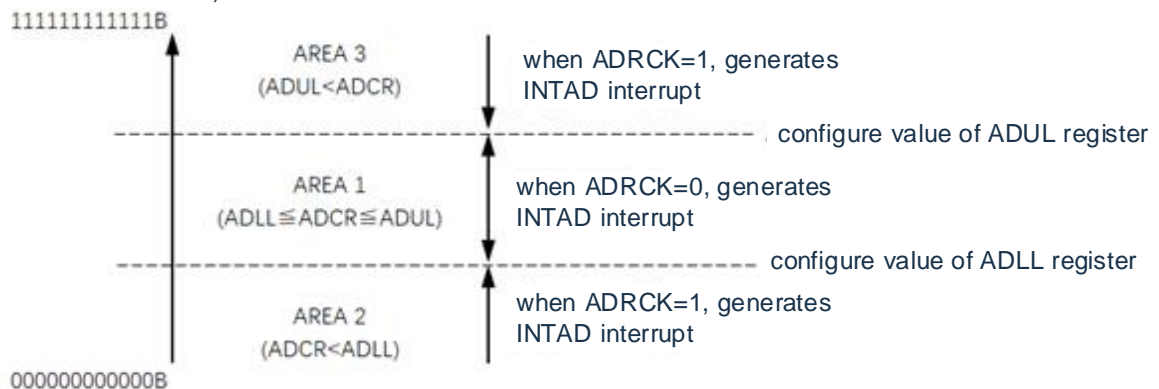


Figure15-7range of interrupt signal generation by ADRCK bit

Note:

1. To rewrite the ADM2 register, it must be done in the transition stop state (ADCS=0).

2. When using AV_{REFP} and AV_{REFM}, THE ANI0 and ANI1 must be set to analog inputs and set to input mode through pin-mode registers.

Note When IND does not occur, the A/D conversion results are not saved to the ADCR registers and ADCRH registers.

15.2.5 The A/D converter's trigger mode register (ADTRG).

This is the register that sets the A/D conversion trigger mode and the hardware trigger signal.

The ADTRG register is set via the 8-bit memory operation instruction.

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

Figure15-8 Format of the trigger mode register (ADTRG) for A/D converter

Reset value: 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 triggers no-wait mode
1	1	Hardware triggers 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).

Note:

1. To override the ADTRG register, it must be done in the transition stop state (ADCS=0, ADCE=0).

2. In order to end the A/D conversion normally, the hardware trigger interval must be set to at least the following time:

When hardware triggers no wait mode: $2 f_{CLK}$ clocks + A/D conversion time

When hardware triggers wait mode: $2 f_{CLK}$ clock + A/D power supply settling wait time + A/D conversion time

Note 1. f_{CLK} : Cpu/peripheral hardware clock frequency

15.2.6 Analog input channel specified register (ADS).

This is the register that specifies the analog voltage input channel to be converted to A/D.

Set the ADS registers via the 8-bit memory operation instructions.

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

Figure15-9 The analog input channel specifies the format of the register (ADS).

Reset value: 00H R/W

	7	6	5	4	3	2	1	0
ADS	ADISS	ADCHPGA1	ADCHPGA0	ADS4	ADS3	ADS2	ADS1	ADS0

○ Selection mode (ADM1. ADMD=0)

ADS register setting value								CH Selection
ADISS	ADCHPGA1	ADCHPGA0	ADS4	ADS3	ADS2	ADS1	ADS0	
0	0	0	0	0	0	0	0	ANI0(P20)
0	0	0	0	0	0	0	1	ANI1(P21)
0	0	0	0	0	0	1	0	ANI2(P22)
0	0	0	0	0	0	1	1	ANI3(P23)
0	0	0	0	0	1	0	0	ANI4(P24)
0	0	0	0	0	1	0	1	ANI5(P25)
0	0	0	0	0	1	1	0	ANI6(P26)
0	0	0	0	0	1	1	1	ANI7(P27)
0	0	0	0	1	0	0	0	ANI8(P11)
0	0	0	0	1	0	0	1	ANI9(P10)
0	0	0	0	1	0	1	0	ANI10(P03)
0	0	0	0	1	0	1	1	ANI11(P02)
0	0	0	0	1	1	0	0	ANI12(P147)
0	0	0	0	1	1	0	1	ANI13(P04)
0	0	0	0	1	1	1	0	ANI14(P120)
0	0	0	0	1	1	1	1	ANI15(P146)
0	0	0	1	0	0	0	0	ANI16(P100)
0	0	0	1	0	0	0	1	ANI17(P150)
0	0	0	1	0	0	1	0	ANI18(P151)
0	0	0	1	0	0	1	1	ANI19(P152)
0	0	0	1	0	1	0	0	ANI20(P153)
0	0	0	1	0	1	0	1	ANI21(P154)
0	0	0	1	0	1	1	0	ANI22(P155)
0	0	0	1	0	1	1	1	ANI23(P156)
0	0	0	1	1	0	0	0	ANI24(P101)
0	0	0	1	1	0	0	1	ANI25(P102)
0	0	0	1	1	0	1	0	ANI26(P144)
0	0	0	1	1	0	1	1	ANI27(P145)
0	0	0	1	1	1	1	1	SW ALL OFF
1	0	0	0	0	0	0	0	Output Voltage ^{Note 1} for Temperature Sensor
1	0	0	0	0	0	0	1	Internal Reference Voltage (1.45V)
0	1	0	0	0	0	0	0	PGA1
0	0	1	0	0	0	0	0	PGA0
Others disable from setting								

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

○4-channel scan mode (ADM1. ADMD=1)

ADISS	ADS4	ADS3	ADS2	ADS1	ADS0	Analog input channel			
						Scan 0	Scan 1	Scan 2	Scan 3
0	0	0	0	0	0	ANI0	ANI1	ANI2	ANI3
0	0	0	0	0	1	ANI1	ANI2	ANI3	ANI4
0	0	0	0	1	0	ANI2	ANI3	ANI4	ANI5
0	0	0	0	1	1	ANI3	ANI4	ANI5	ANI6
0	0	0	1	0	0	ANI4	ANI5	ANI6	ANI7
0	0	0	1	0	1	ANI5	ANI6	ANI7	ANI8
0	0	0	1	1	0	ANI6	ANI7	ANI8	ANI9
0	0	0	1	1	1	ANI7	ANI8	ANI9	ANI10
0	0	1	0	0	0	ANI8	ANI9	ANI10	ANI11
0	0	1	0	0	1	ANI9	ANI10	ANI11	ANI12
0	0	1	0	1	0	ANI10	ANI11	ANI12	ANI13
0	0	1	0	1	1	ANI11	ANI12	ANI13	ANI14
0	0	1	1	0	0	ANI12	ANI13	ANI14	ANI15
Other than the above						Prohibit settings.			

○3 channel scan mode (ADM1. ADMD=1)

ADISS	ADS[4:0]	Analog input channel		
		Scan 0	Scan 1	Scan 2
1'b0	5'h00	ANI0	ANI1	ANI2
1'b0	5'h01	ANI1	ANI2	ANI3
1'b0	5'h02	ANI2	ANI3	ANI4
1'b0	5'h03	ANI3	ANI4	ANI5
1'b0	5'h04	ANI4	ANI5	ANI6
1'b0	5'h05	ANI5	ANI6	ANI7
1'b0	5'h06	ANI6	ANI7	ANI8
1'b0	5'h07	ANI7	ANI8	ANI9
1'b0	5'h08	ANI8	ANI9	ANI10
1'b0	5'h09	ANI9	ANI10	ANI11
1'b0	5'h0A	ANI10	ANI11	ANI12
1'b0	5'h0B	ANI11	ANI12	ANI13
1'b0	5'h0C	ANI12	ANI13	ANI14
1'b0	5'h0D	ANI13	ANI14	ANI15
Other than the above		Prohibit settings.		

○2 channel scan mode (ADM1. ADMD=1)

ADISS	ADS[4:0]	Analog input channel	
		Scan 0	Scan 1
1'b0	5'h00	ANI0	ANI1
1'b0	5'h01	ANI1	ANI2
1'b0	5'h02	ANI2	ANI3
1'b0	5'h03	ANI3	ANI4
1'b0	5'h04	ANI4	ANI5
1'b0	5'h05	ANI5	ANI6
1'b0	5'h06	ANI6	ANI7
1'b0	5'h07	ANI7	ANI8
1'b0	5'h08	ANI8	ANI9
1'b0	5'h09	ANI9	ANI10
1'b0	5'h0A	ANI10	ANI11
1'b0	5'h0B	ANI11	ANI12
1'b0	5'h0C	ANI12	ANI13
1'b0	5'h0D	ANI13	ANI14
1'b0	5'h0E	ANI14	ANI15
Other than the above		Prohibit settings.	

Note:

1. When scanning mode, bit4, bit5 and bit6 must be set to "0".
2. In the pin set by the PMCx register as the analog input, the A/D conversion can be specified as the analog input by the ADS.
3. Pins that are set to digital inputs/outputs by pin mode control registers (PMCxx) cannot be set through ADS registers.
4. To override the ADISS bit, it must be done in the transition stop state (ADCS=0, ADCE=0).
5. When using AVREFP as the positive (+) reference voltage for an A/D converter, ANI0 cannot be selected as the A/D conversion channel.
6. When using AVREFM as the negative (–) reference voltage for an A/D converter, ANI1 cannot be selected as the A/D conversion channel.
7. After placing the ADISS position "1", the conversion result of the first time cannot be used. For a detailed setup procedure, refer to "15.5.4 Settings when selecting the output voltage/internal reference voltage of the temperature sensor".
8. When you want to move to deep sleep mode or to sleep mode while the CPU is running at the subsystem clock, you cannot place the ADISS position "1".

15.2.7 12-bit A/D conversion result register (ADCR).

This is the 16-bit register that holds the results of the A/D conversion, which is readable only. Whenever an A/D conversion ends, the conversion result note is loaded from the successive approximation register (SAR).

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.

Read the ADCR registers via 16-bit memory manipulation instructions. After generating a reset signal, the value of this register changes to "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 is not saved Conversion results.

Figure15-10 Format of the 12-bit A/D Conversion Result Register (ADCR).

Reset value: 0000H R

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCR	ADCH3	ADCH2	ADCH1	ADCH0	ADCR[11:0]											

Note:

1. If only 8 bit resolution A/D conversion results are required, the high 8 bits of the conversion result can be read through the ADCRH register.

2. When 16 bits of access are made to the ADCR register, the high 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 The relationship between the readout values of CHRDE=1 and ADCH0~3 and the conversion channel is as follows:

ADCH3	ADCH2	ADCH1	ADCH0	The conversion channel ID
0	0	0	0	ANI0(P20)
0	0	0	1	ANI1(P21)
0	0	1	0	ANI2(P22)
0	0	1	1	ANI3(P23)
0	1	0	0	ANI4(P24)
0	1	0	1	ANI5(P25)
0	1	1	0	ANI6(P26)
0	1	1	1	ANI7(P27)
1	0	0	0	ANI8(P11)
1	0	0	1	ANI9(P10)
1	0	1	0	ANI10(P03)
1	0	1	1	ANI11(P02)
1	1	0	0	ANI12(P147)
1	1	0	1	ANI13(P04)
1	1	1	0	ANI14(P120)
1	1	1	1	ANI15(P146)

15.2.8 8-bit A/D conversion result register (ADCRH).

This is an 8-bit register that holds the A/D conversion results, saving a high 8-bit ^{note} with 12-bit resolution.

Read the ADCRH registers via the 8-bit memory operation instructions.

After generating a reset signal, the value of this register changes to "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 is not saved Conversion results.

Figure15-11 8-bit A/D conversion result register (ADCRH).

Reset value: 00H R

	7	6	5	4	3	2	1	0
ADCRH								

Note: The conversion results must be read after the conversion is complete and before configuring the ADM0, ADS registers. Otherwise, you may not read the correct conversion results.

15.2.9 The conversion result compares the upper limit value of the set register (ADUL).

This is the setting register used to check the upper limit of the A/D conversion result.

Compare the A/D conversion results with the values of the ADUL registers, and the ADRCK in the mode register 2 (ADM2) of the A/D converter. The setting range of bits controls the generation of an interrupt signal (INTAD). The ADUL register is set via the 8-bit memory operation instruction.

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

Note:

1. Convert only the 12-bit A/D result register (ADCR) to the high 8-bit and ADUL registers as well as the ADLL Registers for comparison.
2. To rewrite the ADUL registers and ADLL registers, it must be done in the transition stop state (ADCS=0).
3. When setting the ADUL register and the ADLL register, the ADUL > ADLL must be made.

Figure15-12The conversion result compares the format of the upper limit setting register (ADUL).

Reset Value:FFH R/W

	7	6	5	4	3	2	1	0
ADULTS	ADUL7	AOFL6	ADUL5	ADUL4	ADUL3	ADUL2	ADUL1	ADUL0

15.2.10 The conversion results compare the lower limit value set register (ADLL).

This is the setting register used to check the lower limit value of the A/D conversion result.

The A/D conversion result is compared to the value of the ADLL register, and the ADRCK in the mode register 2 (ADM2) of the A/D converter the setting range of bits controls the generation of an interrupt signal (INTAD). Set the ADLL registers via the 8-bit memory operation instructions.

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

Figure15-13 The conversion results compare the format of the lower limit setting register (ADLL).

Reset value: 00H R/W

	7	6	5	4	3	2	1	0
ADLL	ADLL7	ADLL6	ADLL5	ADLL4	ADLL3	ADLL2	ADLL1	ADLL0

Note:

1. Convert only the 12-bit A/D result register (ADCR) to the high 8-bit and ADUL registers as well as the ADLL Registers for comparison.
2. To rewrite the ADUL registers and ADLL registers, it must be done in the transition stop state (ADCS=0).
3. When setting the ADUL register and the ADLL register, the ADUL > ADLL must be made.

15.2.11 A/D Sampling Time Control Register (ADNSMP).

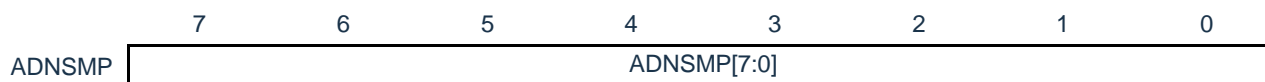
This register controls the A/D sampling time.

The ADNSMP register is set via the 8-bit memory operation instruction.

After generating a reset signal, the value of this register changes to "0dH".

Figure15-14A/D Sampling Time Control Register (ADNSMP).

Reset value: 0dH R/W



Number of sample clock settings:

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

Note: To override the ADSMP register, it must be done in the transition stop state (ADCS=0).

Under different conditions, the sampling time required for each conversion channel is guaranteed:

A/D conversion mode	AVDD[V]	AN1x[ns]	PGA0/PGA1[ns]
High-speed transformation	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

15.2.12 A/D sampling time extended register (ADSMPWAIT).

This register is used to extend the A/D sampling time.
The ADSMPWAIT register is set via the 8-bit memory operation instruction.
After generating a reset signal, the value of this register changes to "00H".

Figure15-15 A/D Sampling Time Extension Register (ADSMPWAIT).

Reset value: 00H

R/W

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

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 arbitrarily extended when "1", 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 transition stop state (ADCS=0), and ADSMPWAIT can be rewritten to "0" at (ADCS=1).

15.2.13 A/D Test Register (ADTES).

This register is used to set the test mode of the A/D converter.
The ADTES register is set via the 8-bit memory operation instruction.
After generating a reset signal, the value of this register changes to "00H".

Figure15-16 A/D Test Register (ADTES).

Reset value: 00H R/W

	7	6	5	4	3	2	1	0
ADTEST	0	0	0	0	ADTES3	ADTES2	ADTES1	ADTES0

ADTES2	ADTES1	ADTES0	A/D operating mode
0	0	0	Usually converted
0	0	1	Self-diagnostic test of 0 yards
0	1	1	Half-code self-diagnostic test
1	0	1	Full-code self-diagnostic test
Other than the above			Prohibit settings.

15.2.14 A/D status register (ADFLG).

This register represents the status of the A/D converter.
Read the ADFLG registers via the 8-bit memory operation instructions.
After generating a reset signal, the value of this register changes to "00H".

Figure15-17 A/D Status Register (ADFLG).

Reset value: 00H R

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

ADFLG4	A/D transition status
0	The A/D conversion does not end in single conversion mode
1	In single-stroke conversion mode, the conversion ends (zero is automatically cleared after 2 ADCLKs). The ADFLG4 remains at 1'b0 in continuous conversion mode

ADFLG3	A/D transition status
0	1 ADCLK before the end of the non-A/D conversion
1	1 ADCLK before the end of A/D conversion (1 ADCLK is automatically cleared after 1 ADCLK).

ADFLG2	A/D transition status
0	2 ADCLKs before the end of non-A/D conversion
1	2 ADCLKs before the end of A/D conversion (automatic zeroing after 1 ADCLK).

ADFLG1	A/D transition status
0	Non-sequential comparison period
1	Compare the periods one by one

ADFLG0	A/D transition status
0	During non-A/D sampling
1	A/D sampling period

15.2.15 A/D Charge and Discharge Control Register (ADNDIS).

This register is used to control the charge and discharge action and time of the A/D converter.
Read and write ADNDIS registers via 8-bit memory manipulation instructions.
After generating a reset signal, the value of this register changes to "00H".

Figure15-18 A/D Charge and Discharge Control Register (ADNDIS).

Reset value: 00HW

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

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

ADNDIS[3:0]	Charge and discharge time
4'b0000	No charge or discharge is
4'b0010	2 ADCLKs
4'b0011	3 ADCLKs
4'b0100	4 ADCLKs
4'b0101	5 ADCLKs
4'b0110	6 ADCLKs
.....
4'b1111	15 ADCLKs

Note: It is forbidden to set the charge-discharge time to 1 ADCLK, that is, ADNDIS [3:0] = 4'b0001

15.2.16 Registers that control the pin function of the analog input pins

Control registers (PIN Mode Control Registers (PMCxx)) for pin function multiplexing with the analog inputs of the A/D converter must be set. For details, please refer to "2.3." 8-pin mode control register (PMCxx)".

When using the ANIx pins as analog inputs to A/D converters, the corresponding pin-mode control register (PMCxx) must be placed at position "1".

15.3 Input voltage and conversion result

Analog input voltage of the analog input pin (AN_{ix}) and theoretical A/D conversion result (1 2-bit A/D conversion result register (ADCR)) has the following expression relationship.

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

or

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

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

V_{AIN} : Analog input voltage

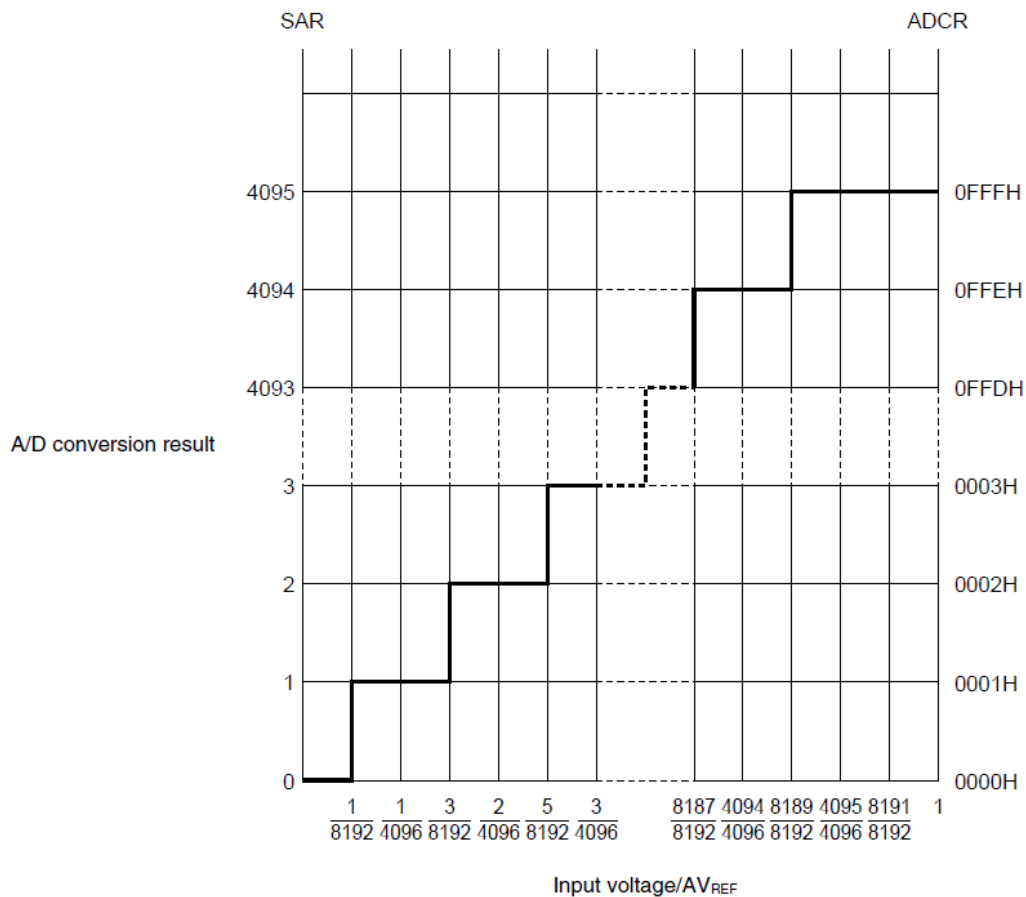
AV_{REF} : AV_{REF} pin voltage

ADCR: The value of the A/D conversion result register (ADCR).

SAR: Successive approximation registers

The analog input voltage and the A/D conversion result are shown in the following figure.

Figure15-19 Analog input voltage vs. A/D conversion result



Note: AV_{REF} is the positive (+) reference voltage of the A/D converter, and AV_{REFP} or V_{DD} can be selected .

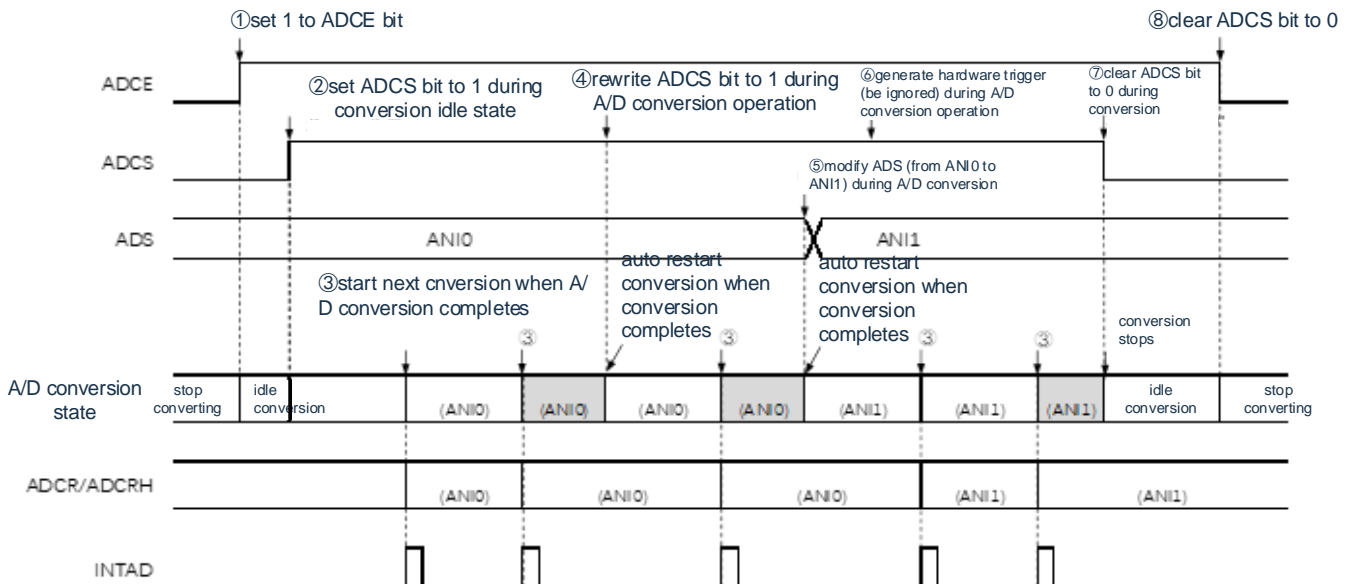
15.4 The operating mode of the A/D converter

The operation of each mode of the A/D converter is as follows. For the setup steps for each mode, refer to "15.5 A/D Converter Setup Flowchart".

15.4.1 Software-triggered mode (selection mode, continuous conversion mode)

- ① In the stopped state, enter the ADCE position "1" of the A/D converter's mode register 0 (ADM0) into the A/D transition standby state.
- ② After counting the settling wait time (1us) by software, the ADCS position of the ADM0 register is "1" to the register specified by the analog input channel (ADS The specified analog input performs an A/D conversion.
- ③ If the A/D conversion is complete, the conversion result is saved to the A/D conversion result register (ADCR, ADCRH) and an A/D transition end interrupt request signal is generated (INTAD) . Start the next A/D conversion immediately after the A/D conversion is complete.
- ④ If you rewrite "1" for the ADCS bit during the conversion process, the current A/D conversion is immediately aborted and the conversion is restarted.
- ⑤ If the ADS registers are overwritten or rewritten during the conversion process, the current A/D conversion is immediately aborted and the analog inputs reassigned by the ADS registers are A/D converted.
- ⑥ The A/D conversion does not start even if the input hardware triggers during the conversion process.
- ⑦ If the ADCS position is "0" during the conversion, the current A/D conversion is immediately aborted and then enters the A/D conversion standby state.
- ⑧ If the ADCE position is "0" in the A/D conversion standby state, the A/D converter enters the stopped state. When the ADCE bit is "0", even the ADCS position "1" is ignored and the A/D conversion is not started.

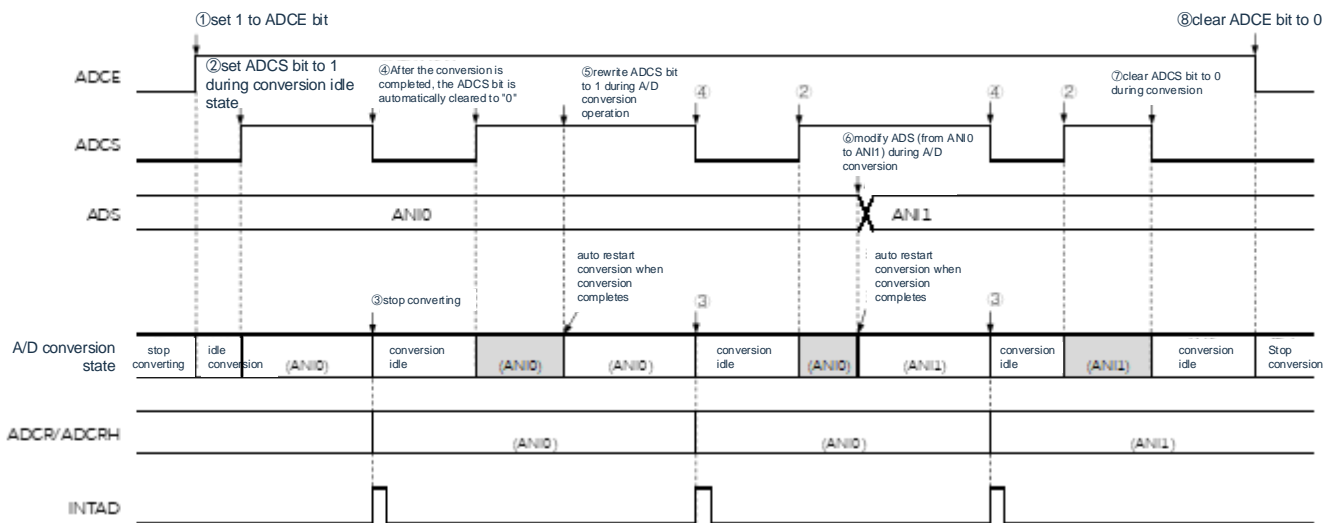
Figure15-20 Operation sequence of software trigger modes (select mode, continuous transition mode).



15.4.2 Software-triggered mode (select mode, single-shot conversion mode)

- ① In the stopped state, enter the ADCE position "1" of the A/D converter's mode register 0 (ADM0) into the A/D transition standby state.
- ② After counting the settling wait time (1us) by software, the ADCS position of the ADM0 register "1" is assigned to the analog input channel (ADS) The specified analog input performs A/D conversion.
- ③ If the A/D conversion is complete, the conversion result is saved to the A/D conversion result register (ADCR, ADCRH) and an A/D transition end interrupt request signal is generated (INTAD) .
- ④ After the A/D conversion is over, the ADCS bit automatically clears "0" and enters the A/D conversion standby state.
- ⑤ If you rewrite "1" for the ADCS bit during the conversion process, the current A/D conversion is immediately aborted and the conversion restarts.
- ⑥ If the ADS registers are overwritten or rewritten during the conversion process, the current A/D conversion is immediately aborted and the analog inputs reassigned by the ADS registers are A/D converted.
- ⑦ If the ADCS position is "0" during the conversion, the current A/D conversion is immediately aborted and then enters the A/D conversion standby state.
- ⑧ If the ADCE position is "0" in the A/D conversion standby state, the A/D converter enters the stopped state. When the ADCE bit is "0", even the ADCS position "1" is ignored and the A/D conversion is not started. The input hardware trigger does not start even in the A/D conversion standby state.

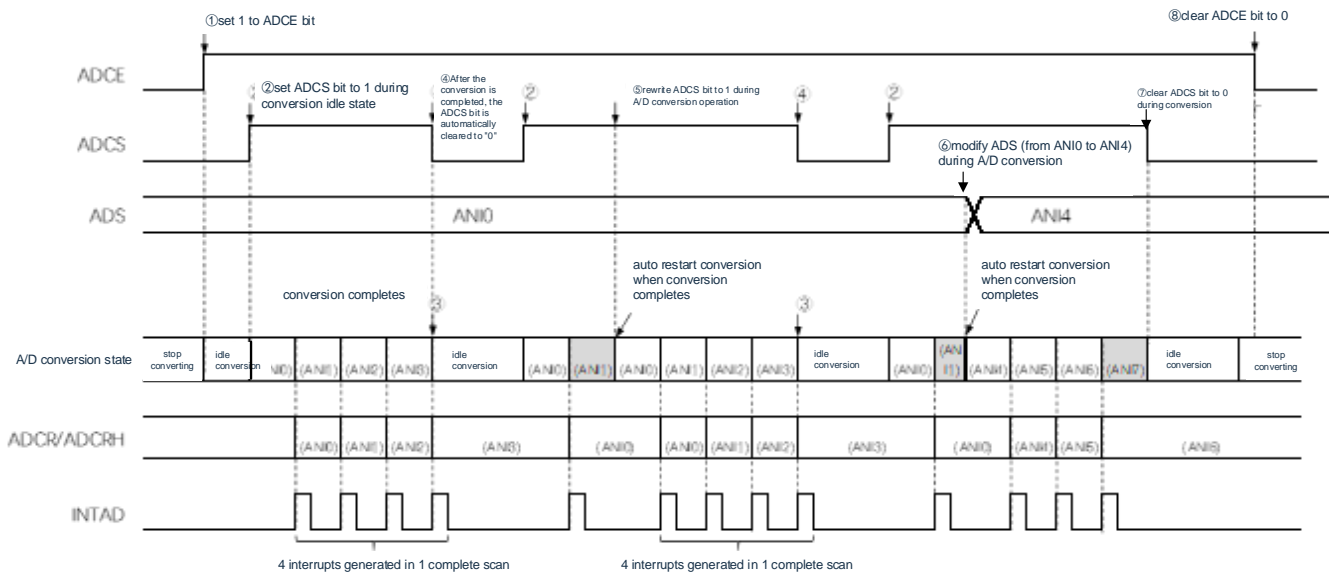
Figure15-21 Example of Operation sequence of software trigger mode (select mode, single conversion mode).



15.4.4 Software trigger mode (scan mode, single-shot conversion mode)

- ① In the stopped state, enter the ADCE position "1" of the A/D converter's mode register 0 (ADM0) into the A/D transition standby state.
- ② After counting the settling wait time (1us) by software, the ADCS position of the ADM0 register "1" is switched on by the analog input The four analog input channels specified by the Channel Specified Register (ADS) scan 0 to scan 3 perform A/D conversion. A/D conversion is performed sequentially from the simulated input channel specified by scan 0.
- ③ A/D conversion of 4 analog input channels is performed continuously. Whenever the A/D conversion ends, the conversion result is saved to the A/D conversion result register (ADCR, ADCRH) and an A/D is generated Conversion End Interrupt Request Signal (INTAD).
- ④ After the A/D conversion of 4 channels is completed, the ADCS bit is automatically cleared "0" and enters the A/D conversion standby state.
- ⑤ If you rewrite "1" for the ADCS bit during the conversion process, the current A/D conversion is immediately aborted and the conversion restarts.
- ⑥ If the ADS register is overwritten or rewritten during the conversion process, the current A/D conversion is immediately aborted and the A/D conversion is performed from the initial channel re-specified by the ADS register.
- ⑦ If the ADCS position is "0" during the conversion, the current A/D conversion is immediately aborted and then enters the A/D conversion standby state.
- ⑧ If the ADCE position is "0" in the A/D conversion standby state, the A/D converter enters the stopped state. When the ADCE bit is "0", even the ADCS position "1" is ignored and the A/D conversion is not started. The input hardware trigger does not start even in the A/D conversion standby state.

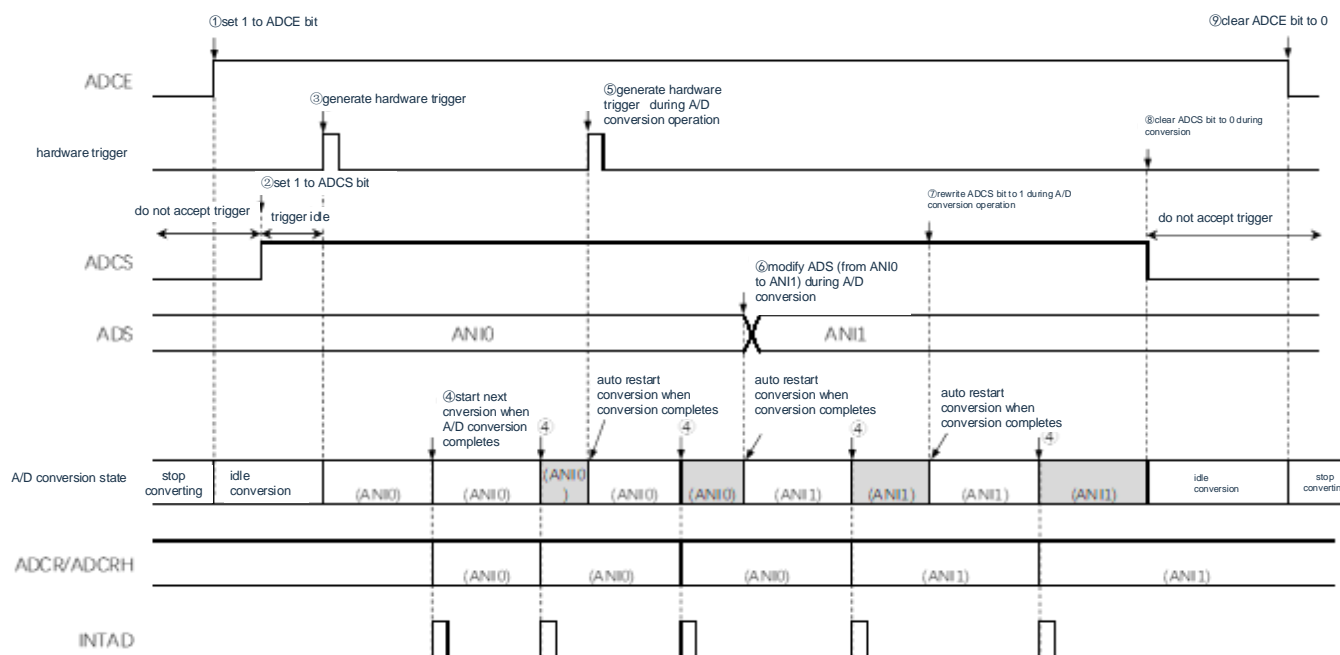
Figure15-23 Operation sequence of software trigger mode (scan mode, single-shot conversion mode).



15.4.5 Hardware-triggered no-wait mode (select mode, continuous transition mode)

- ① In the stopped state, enter the ADCE position "1" of the A/D converter's mode register 0 (ADM0) into the A/D transition standby state.
- ② After counting the settling wait time (1us) by software, the ADCS position of the ADM0 register is "1", Entering hardware triggers standby (this phase does not start the transition). When the hardware triggers standby, the A/D conversion does not start even if the ADCS position "1" is placed.
- ③ If the input hardware is triggered in a state where the ADCS bit is "1", A/D conversion is performed on the analog input specified by the analog input channel specified by the register (ADS).
- ④ If the A/D conversion is complete, the conversion result is saved to the A/D conversion result register (ADCR, ADCRH) and an A/D transition end interrupt request signal is generated (INTAD). Start the next A/D conversion immediately after the A/D conversion is complete.
- ⑤ If the input hardware is triggered during the conversion, the current A/D conversion is aborted immediately and the conversion is restarted.
- ⑥ If the ADS registers are overwritten or rewritten during the conversion process, the current A/D conversion is immediately aborted and the analog inputs reassigned by the ADS registers are A/D converted.
- ⑦ If you rewrite "1" for the ADCS bit during the conversion process, the current A/D conversion is immediately aborted and the conversion restarts.
- ⑧ If the ADCS position is "0" during the conversion, the current A/D conversion is immediately aborted and then enters the A/D conversion standby state. However, in this state the A/D converter does not enter the stopped state.
- ⑨ If the ADCE position is "0" in the A/D conversion standby state, the A/D converter enters the stopped state. When the ADCS bit is "0", even the input hardware trigger is ignored and the A/D conversion does not begin.

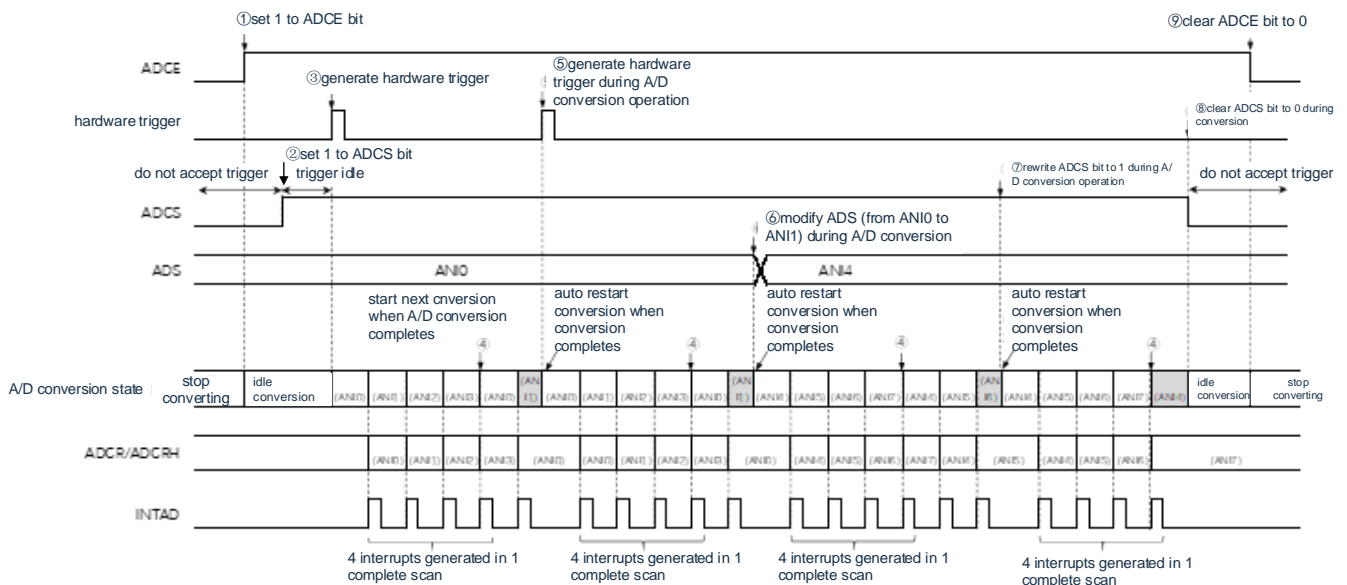
Figure15-24 Hardware triggers an example of a Operation sequence in a wait-free mode (select mode, continuous transition mode).



15.4.7 Hardware-triggered no-wait mode (scan mode, continuous transition mode)

- ① In the stopped state, enter the ADCE position "1" of the A/D converter's mode register 0 (ADM0) into the A/D transition standby state.
- ② After counting the settling wait time (1us) by software, the ADCS position of the ADM0 register is "1" and enters the hardware-triggered standby state (this stage does not start the transition). When the hardware triggers standby, the A/D conversion does not start even if the ADCS position "1" is placed.
- ③ If the input hardware is triggered in the ADCS bit "1", scan 0 to 4 of scan 3 specified by the analog input channel specified register (ADS). A/D conversion of analog input channels. A/D conversion is performed sequentially from the analog input channel specified by scan 0.
- ④ A/D conversion of 4 analog input channels is performed continuously. Whenever the A/D conversion ends, the conversion result is saved to the A/D conversion result register (ADCR, ADCRH) and an A/D is generated Conversion End Interrupt Request Signal (INTAD). The next A/D conversion automatically starts from the set channel immediately after the end of the 4-channel A/D conversion.
- ⑤ If the input hardware is triggered during the conversion, the current A/D conversion is immediately aborted and the conversion is restarted from the original channel.
- ⑥ If the ADS registers are overwritten or rewritten during the conversion process, the current A/D conversion is immediately aborted and then A/D is performed from the channel reassigned by the ADS registers.
- ⑦ If you rewrite "1" for the ADCS bit during the conversion process, the current A/D conversion is immediately aborted and the conversion starts again from the original channel.
- ⑧ If the ADCS position is "0" during the conversion, the current A/D conversion is immediately aborted and then enters the A/D conversion standby state. However, in this state the A/D converter does not enter the stopped state.
- ⑨ If the ADCE position is "0" in the A/D conversion standby state, the A/D converter enters the stopped state. When the ADCE bit is "0", even the ADCS position "1" is ignored and the A/D conversion is not started.

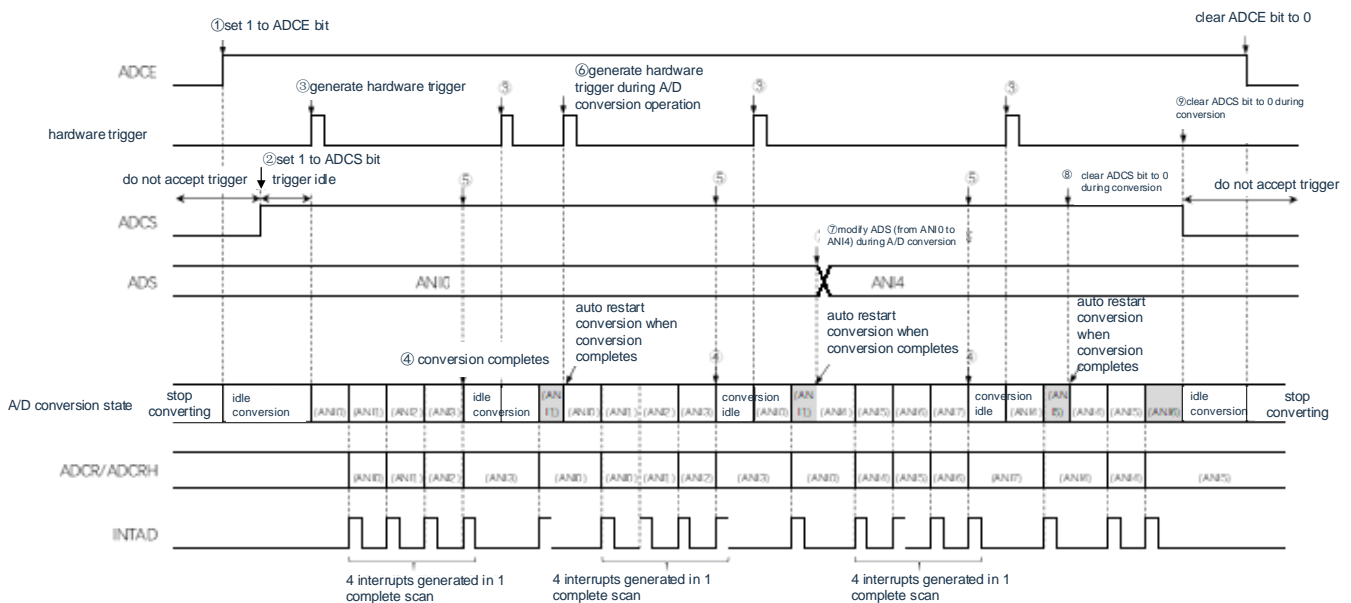
Figure15-26 Example of a Operation sequence in which hardware triggers no wait mode (scan mode, continuous transition mode).



15.4.8 Hardware-triggered no-wait mode (scan mode, single-shot conversion mode)

- ① In the stopped state, enter the ADCE position "1" of the A/D converter's mode register 0 (ADM0) into the A/D transition standby state.
- ② After counting the settling wait time (1 μ s) by software, the ADCS position of the ADM0 register "1" enters the hardware-trigger standby state (this stage does not start the transition). When the hardware triggers standby, the A/D conversion does not start even if the ADCS position "1" is placed.
- ③ If the input hardware is triggered in the ADCS bit "1", scan 0 to 4 of scan 3 specified by the analog input channel specified register (ADS). A/D conversion of analog input channels. A/D conversion is performed sequentially from the analog input channel specified by scan 0.
- ④ A/D conversion of 4 analog input channels is performed continuously. Whenever the A/D conversion ends, the conversion result is saved to the A/D conversion result register (ADCR, ADCRH) and an A/D is generated Conversion End Interrupt Request Signal (INTAD).
- ⑤ After the A/D conversion of the 4 channels is completed, the ADCS bit remains in the "1" state and enters the A/D conversion standby state.
- ⑥ If the input hardware is triggered during the conversion process, the current A/D conversion is aborted immediately and the conversion is restarted from the original channel.
- ⑦ If the ADS register is overwritten or rewritten during the conversion process, the current A/D conversion is immediately aborted and the A/D conversion is performed from the initial channel re-specified by the ADS register.
- ⑧ If you rewrite "1" for the ADCS bit during the conversion process, the current A/D conversion is immediately aborted and the conversion starts again from the original channel.
- ⑨ If the ADCS position is "0" during the conversion, the current A/D conversion is immediately aborted and then enters the A/D conversion standby state. However, in this state the A/D converter does not enter the stopped state.
- ⑩ If the ADCE position is "0" in the A/D conversion standby state, the A/D converter enters the stopped state. When the ADCS bit is "0", even the input hardware trigger is ignored and the A/D conversion does not begin.

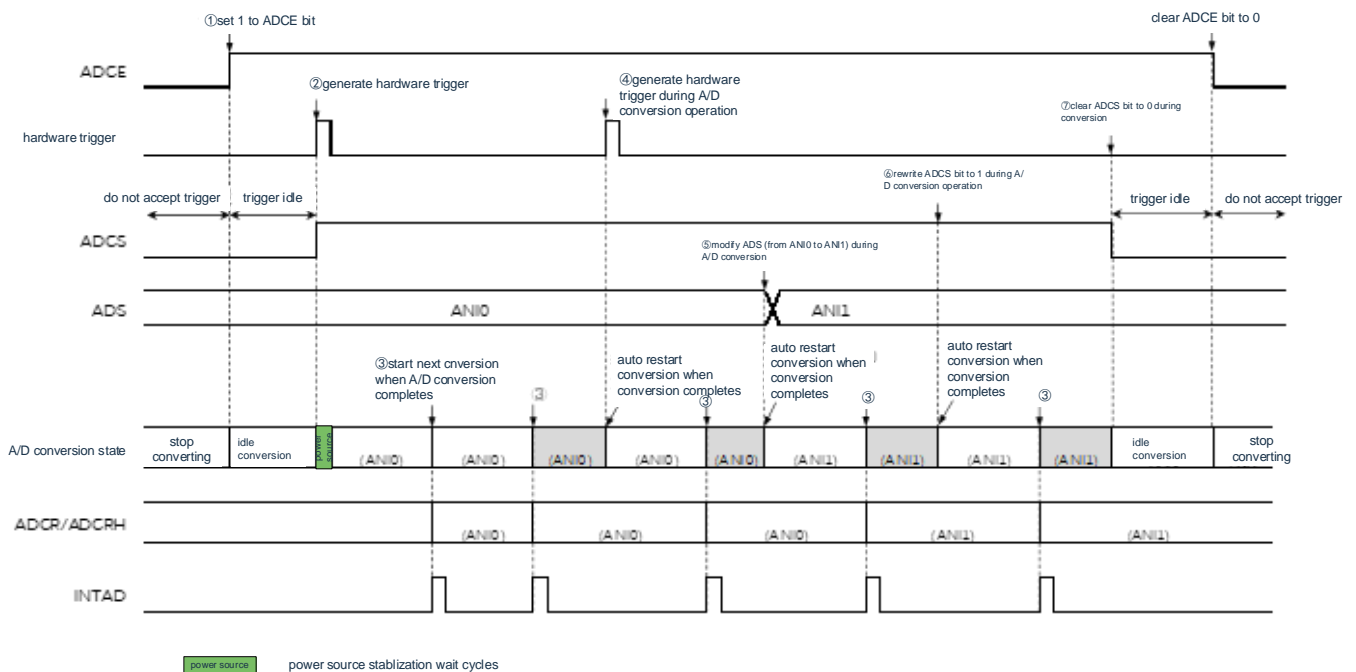
Figure15-27 Operation sequence of hardware triggering no wait mode (scan mode, single transition mode).



15.4.9 Hardware-triggered wait mode (selection mode, continuous transition mode)

- ① In the stopped state, enter the hardware-triggered standby state by placing the ADCE position "1" of the ADM0's mode register 0 (ADM0).
- ② If the input hardware trigger is in hardware-triggered standby, A/D conversion is performed on the analog input specified by the analog input channel specified by the register (ADS). The ADCS position of the ADM0 register is automatically "1" while the input hardware is triggered.
- ③ If the A/D conversion is complete, the conversion result is saved to the A/D conversion result register (ADCR, ADCRH) and an A/D transition end interrupt request signal is generated (INTAD). Start the next A/D conversion immediately after the A/D conversion (at this point, no hardware trigger is required).
- ④ If the input hardware is triggered during the conversion, the current A/D conversion is aborted immediately and the conversion is restarted.
- ⑤ If the ADS registers are overwritten or rewritten during the conversion process, the current A/D conversion is immediately aborted and the analog inputs reassigned by the ADS registers are A/D converted.
- ⑥ If you rewrite "1" for the ADCS bit during the conversion process, the current A/D conversion is immediately aborted and the conversion restarts.
- ⑦ If the ADCS position is "0" during the conversion, the current A/D conversion is aborted immediately, then the hardware triggers the standby state, and the A/D converter enters the stopped state. When the ADCE bit is "0", even the input hardware trigger is ignored and the A/D conversion does not begin.

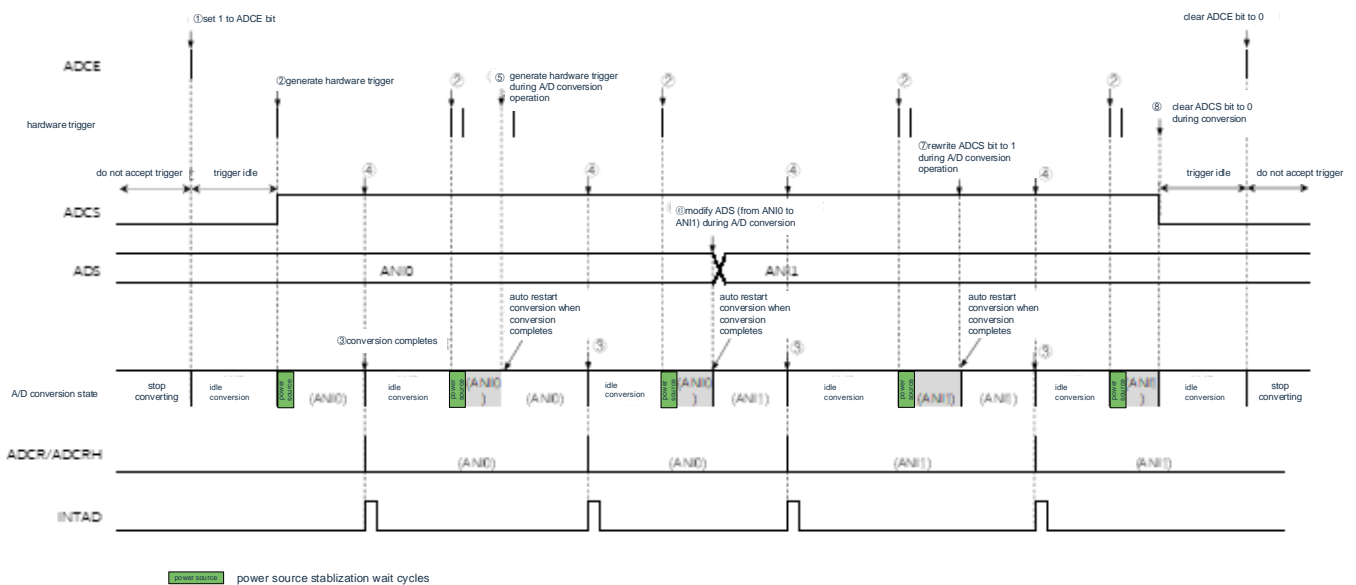
Figure15-28 Example of Operation sequence of hardware triggering wait modes (select mode, continuous transition mode).



15.4.10 Hardware-triggered wait mode (select mode, single-shot transition mode)

- ① In the stopped state, enter the hardware-triggered standby state by placing the ADCE position "1" of the ADM0's mode register 0 (ADM0).
- ② If the input hardware trigger is in hardware-triggered standby, A/D conversion is performed on the analog input specified by the analog input channel specified by the register (ADS). The ADCS position of the ADM0 register is automatically "1" while the input hardware is triggered.
- ③ If the A/D conversion is complete, the conversion result is saved to the A/D conversion result register (ADCR, ADCRH) and an A/D transition end interrupt request signal is generated (INTAD) .
- ④ After the A/D conversion is complete, the ADCS bit is automatically cleared to "0" and the A/D converter enters the stopped state.
- ⑤ If the input hardware is triggered during the conversion, the current A/D conversion is aborted immediately and the conversion is restarted.
- ⑥ If the ADS registers are overwritten or rewritten during the conversion process, the current A/D conversion is immediately aborted and the analog inputs reassigned by the ADS registers are A/D converted.
- ⑦ If you rewrite "1" for the ADCS bit during the conversion process, the current A/D conversion is immediately aborted and the conversion restarts.
- ⑧ If the ADCS position is "0" during the conversion, the current A/D conversion is aborted immediately, then the hardware triggers the standby state, and the A/D converter enters the stopped state. When the ADCE bit is "0", even the input hardware trigger is ignored and the A/D conversion does not begin.

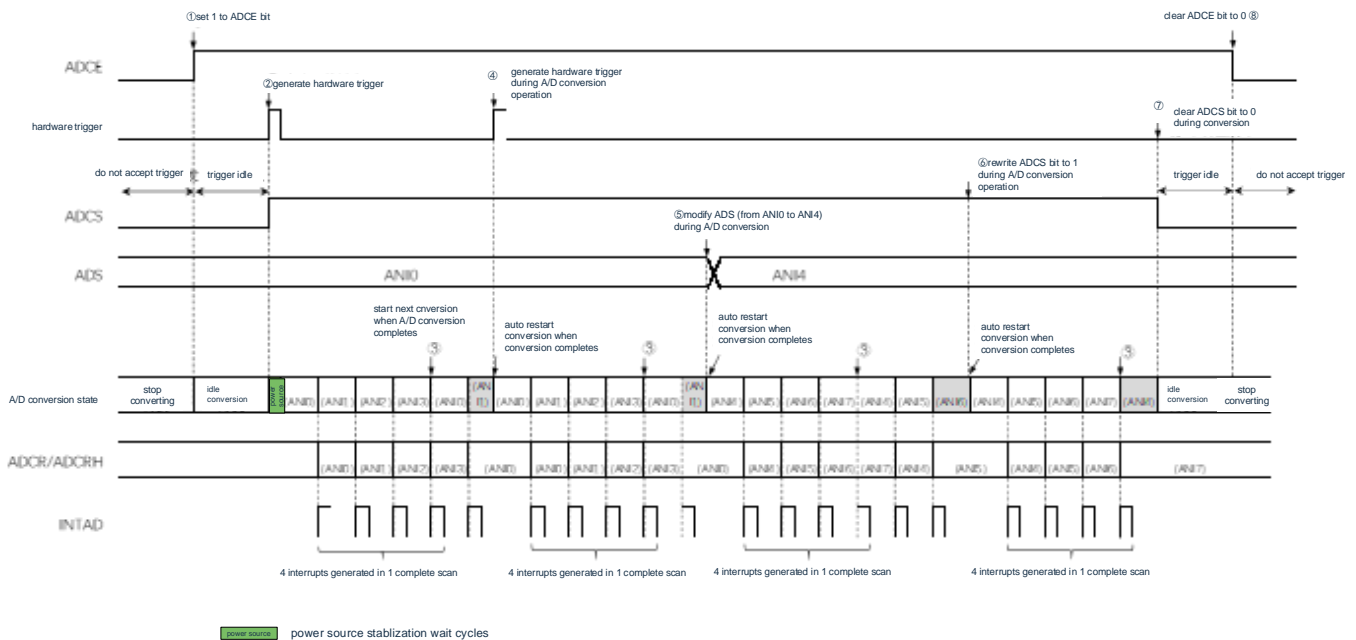
Figure15-29 Operation sequence of hardware trigger wait mode (select mode, single-shot transition mode).



15.4.11 Hardware-triggered wait mode (scan mode, continuous transition mode)

- ① In the stopped state, enter the hardware-triggered standby state by placing the ADCE position "1" of the ADM0's mode register 0 (ADM0).
- ② If the input hardware is triggered in the hardware-triggered standby state, A/D conversion is performed on the four analog input channels specified by the analog input channel specified register (ADS) scan 0 to scan 3. The ADCS position of the ADM0 register is automatically "1" while the input hardware is triggered. A/D conversion is performed sequentially from the analog input channel specified by scan 0.
- ③ A/D conversion of 4 analog input channels is performed continuously. Whenever the A/D conversion ends, the conversion result is saved to the A/D conversion result register (ADCR, ADCRH) and an A/D is generated Conversion End Interrupt Request Signal (INTAD). The next A/D conversion automatically starts from the set channel immediately after the end of the 4-channel A/D conversion.
- ④ If the input hardware is triggered during the conversion process, the current A/D conversion is aborted immediately and the conversion is restarted from the original channel.
- ⑤ If the ADS registers are overwritten or rewritten during the conversion process, the current A/D conversion is immediately aborted and the scan conversion starts with the channel re-specified by the ADS registers.
- ⑥ If you rewrite "1" for the ADCS bit during the conversion process, the current A/D conversion is immediately aborted and the conversion starts again from the original channel.
- ⑦ If the ADCS position is "0" during the conversion, the current A/D conversion is aborted immediately, then the hardware triggers the standby state, and the A/D converter enters the stopped state. When the ADCE bit is "0", even the input hardware trigger is ignored and the A/D conversion does not begin.

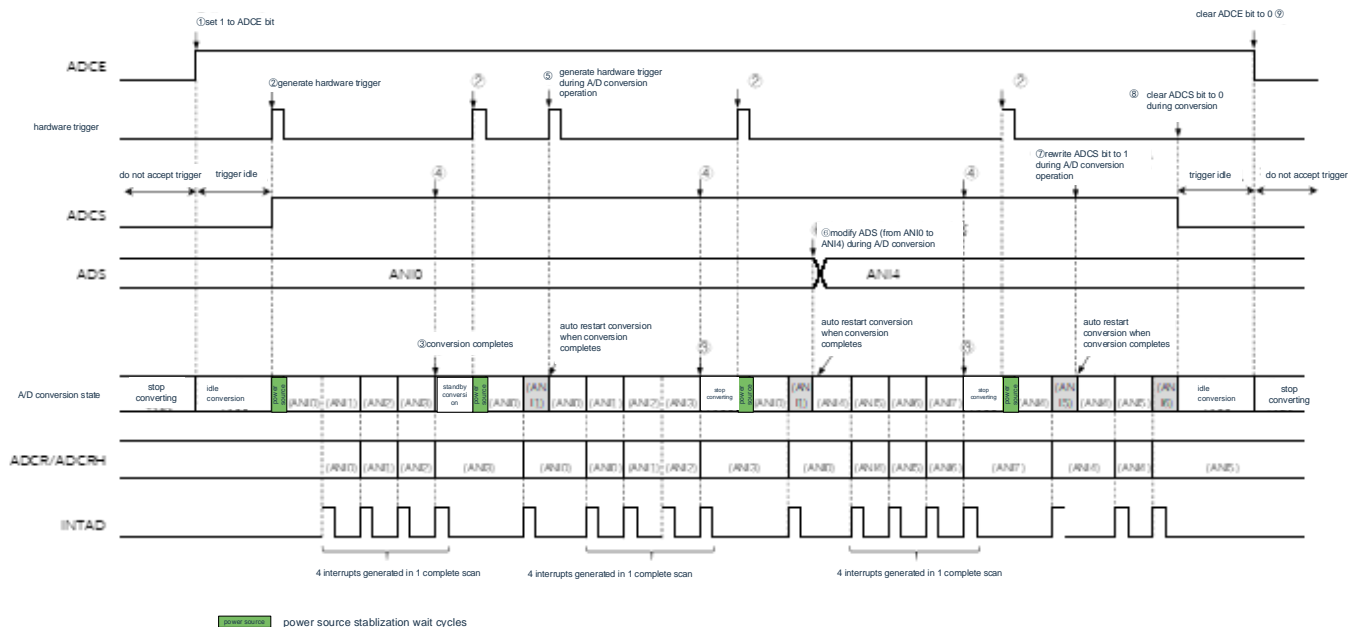
Figure15-30 Example of Operation sequence of hardware-triggered wait modes (scan mode, continuous transition mode).



15.4.12 Hardware-triggered wait mode (scan mode, single-shot transition mode)

- ① In the stopped state, enter the hardware-triggered standby state by placing the ADCE position "1" of the ADM0's mode register 0 (ADM0).
- ② If the input hardware is triggered in the hardware-triggered standby state, A/D conversion is performed on the four analog input channels specified by the analog input channel specified register (ADS) scan 0 to scan 3. The ADCS position of the ADM0 register is automatically "1" after the input hardware is triggered. A/D conversion is performed sequentially from the analog input channel specified by scan 0.
- ③ A/D conversion of 4 analog input channels is performed continuously. Whenever the A/D conversion ends, the conversion result is saved to the A/D conversion result register (ADCR, ADCRH) and an A/D generated Conversion End Interrupt Request Signal (INTAD).
- ④ After the A/D conversion is complete, the ADCS bit is automatically cleared to "0" and the A/D converter enters the stopped state.
- ⑤ If the input hardware is triggered during the conversion, the current A/D conversion is immediately aborted and the conversion is scanned again starting from the original channel.
- ⑥ If the ADS register is overwritten or rewritten during the conversion process, the current A/D conversion is immediately aborted and the scan conversion begins with the channel reassigned by the ADS register.
- ⑦ If you rewrite "1" for the ADCS bit during the conversion, the current A/D conversion is immediately aborted and the conversion is scanned starting from the initial channel.
- ⑧ If the ADCS position is "0" during the conversion, the current A/D conversion is aborted immediately, then the hardware triggers the standby state, and the A/D converter enters the stopped state. When the ADCE bit is "0", even the input hardware trigger is ignored and the A/D conversion does not begin.

Figure15-31 An example of a Operation sequence in which hardware triggers a wait mode (scan mode, single-shot conversion mode).

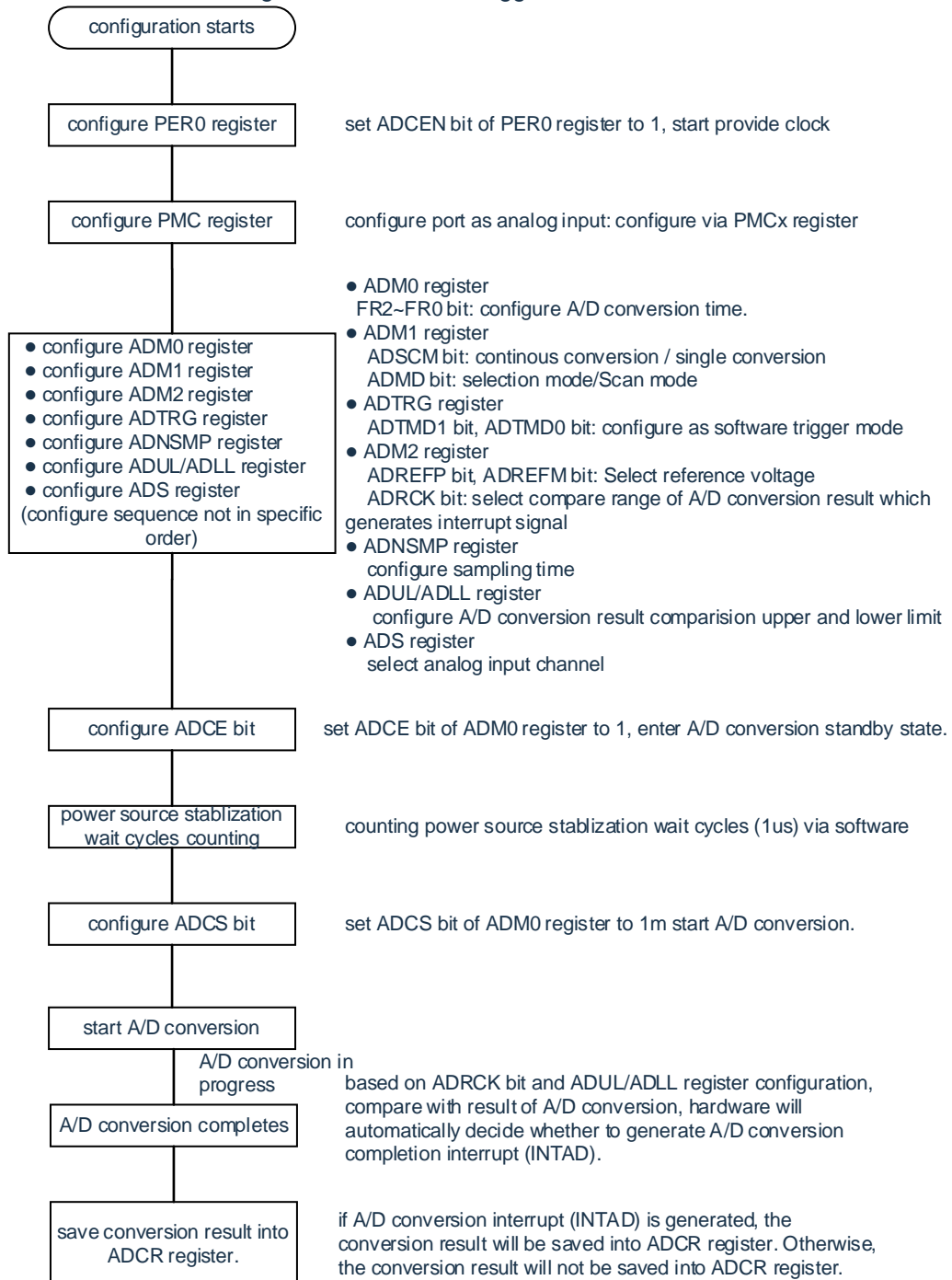


15.5 Setup flowchart for the converter

The setup flowchart of the A/D converters for each operating mode is shown below.

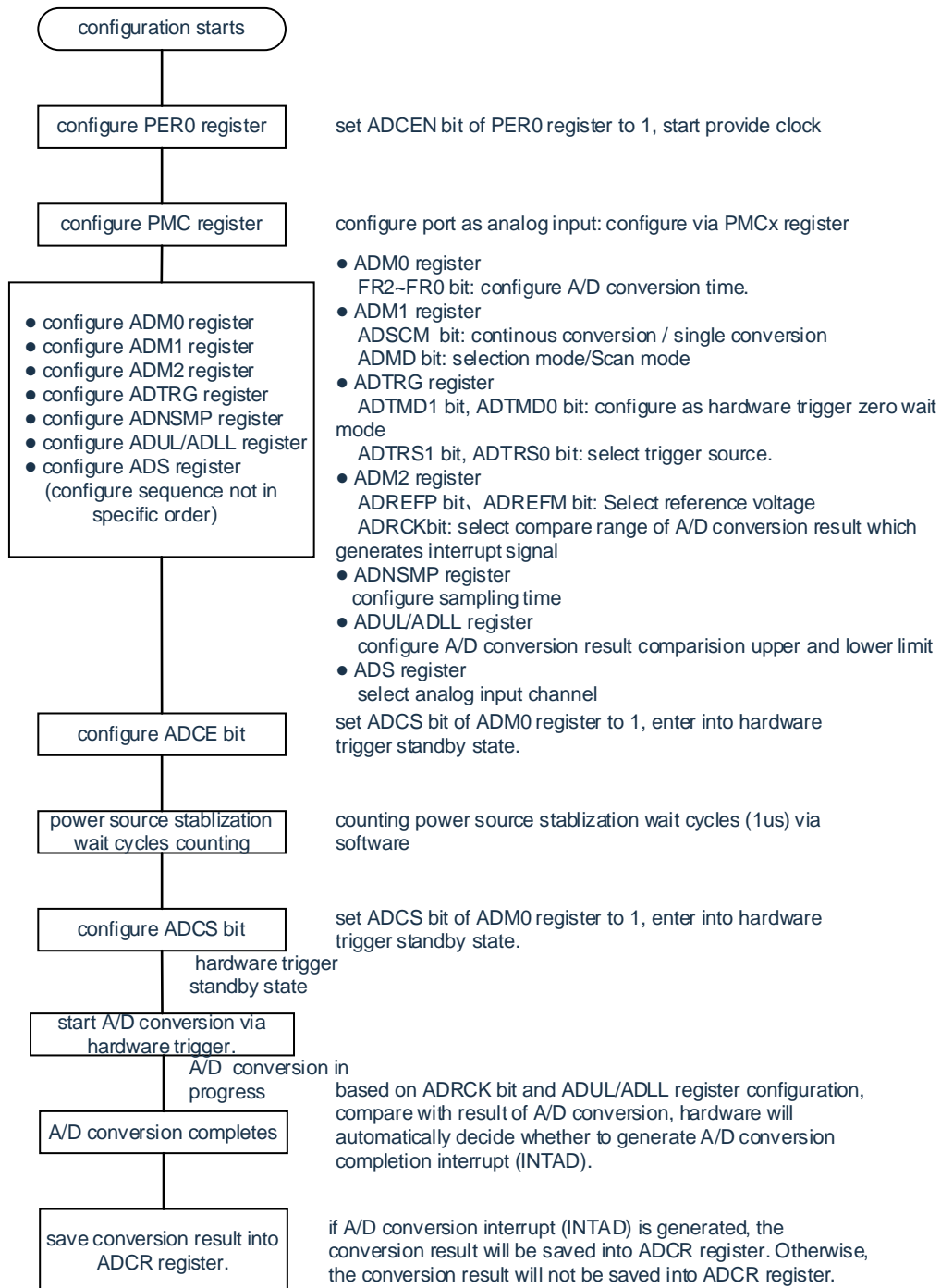
15.5.1 The setting of the software trigger mode

Figure15-32 software trigger mode



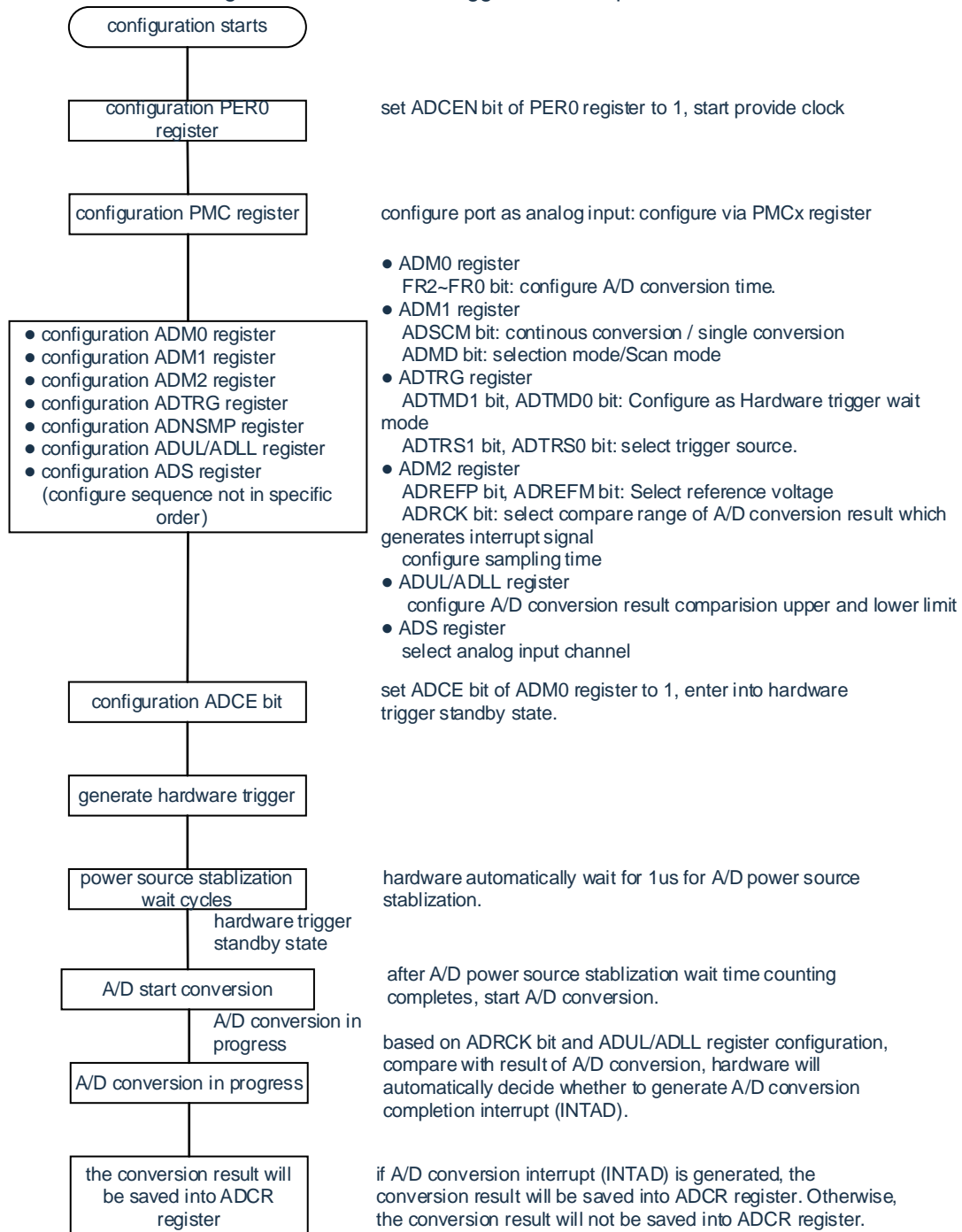
15.5.2 The hardware triggers the setting of no wait mode

Figure15-33 Hardware triggers the setup of no wait mode



15.5.3 Hardware triggers the setting of wait mode

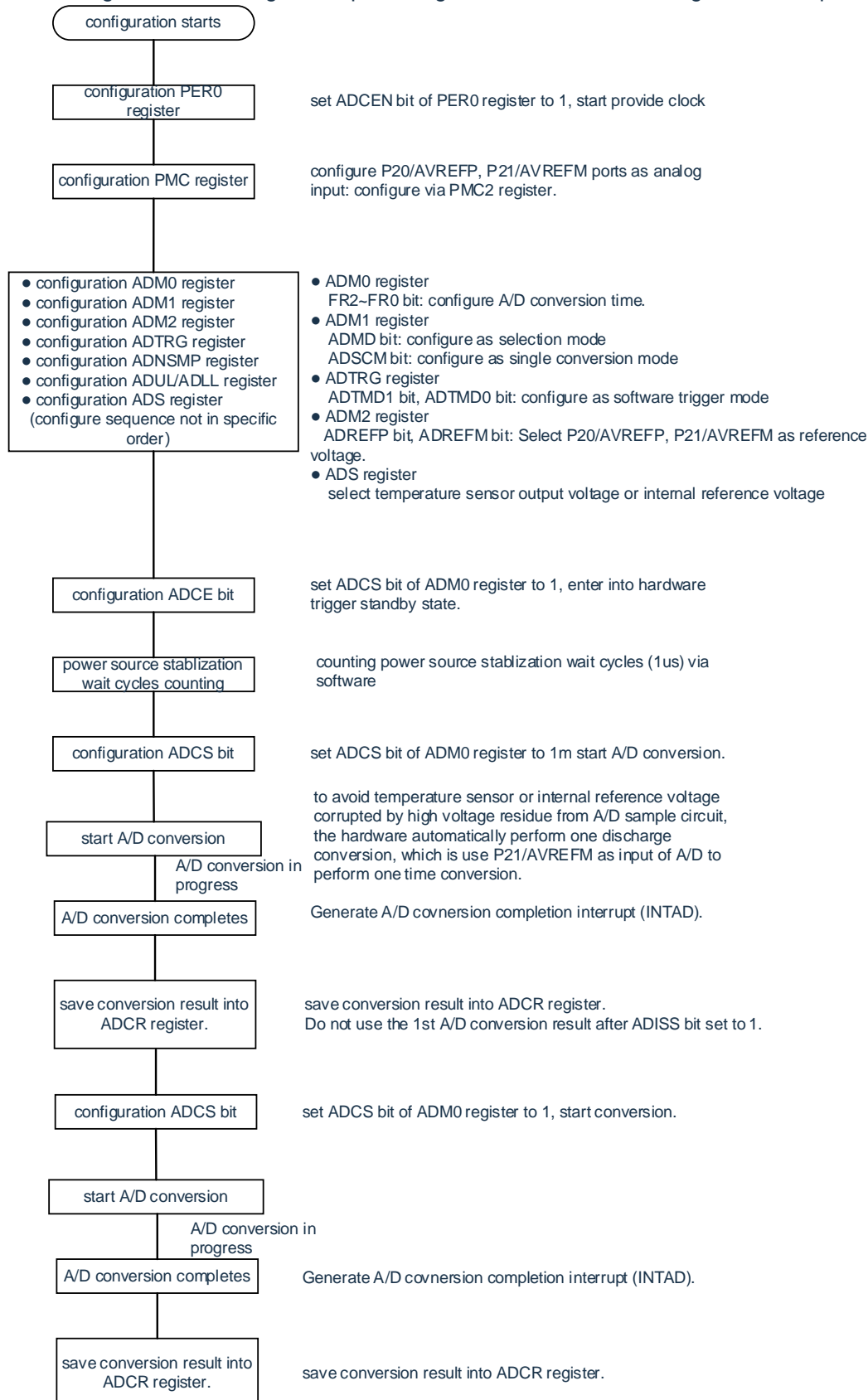
Figure15-34 Hardware triggers the setup of wait mode



15.5.4 Select the setting for the output voltage/internal reference voltage of the temperature sensor

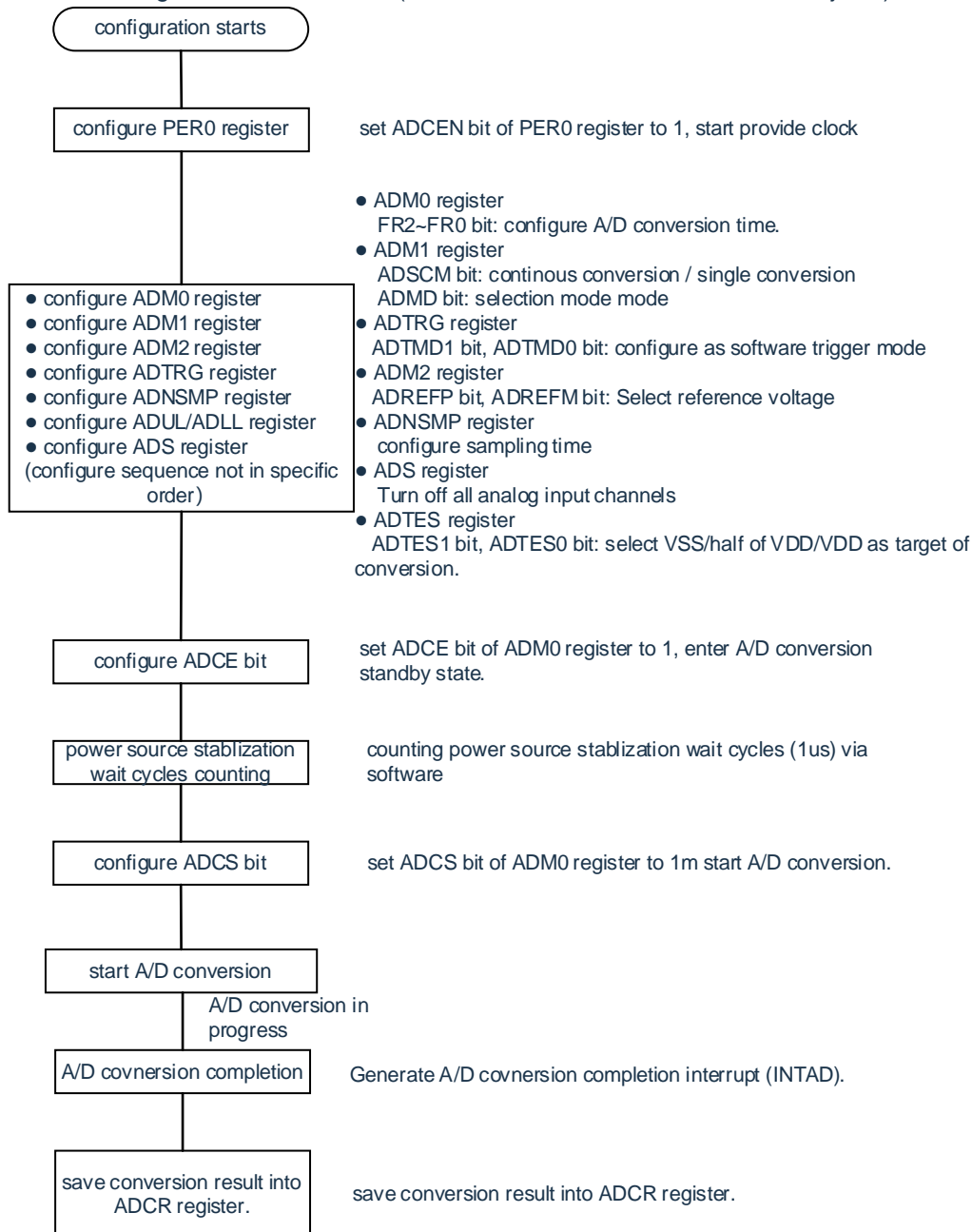
(Take software trigger mode, single conversion mode as an example)

Figure15-35 Settings when selecting the output voltage/internal reference voltage of the temperature sensor



15.5.5 The setting of the test mode

Figure15-36 test mode (VSS/half_VDD/VDD as conversion objects).



Chapter 16 D/A converter

16.1 The functionality of the D/A converter

The D/A converter is an 8-bit resolution converter that converts a digital input to an analog signal and can control the analog output of 2 channels (ANO0, ANO1).

The D/A converter has the following functions:

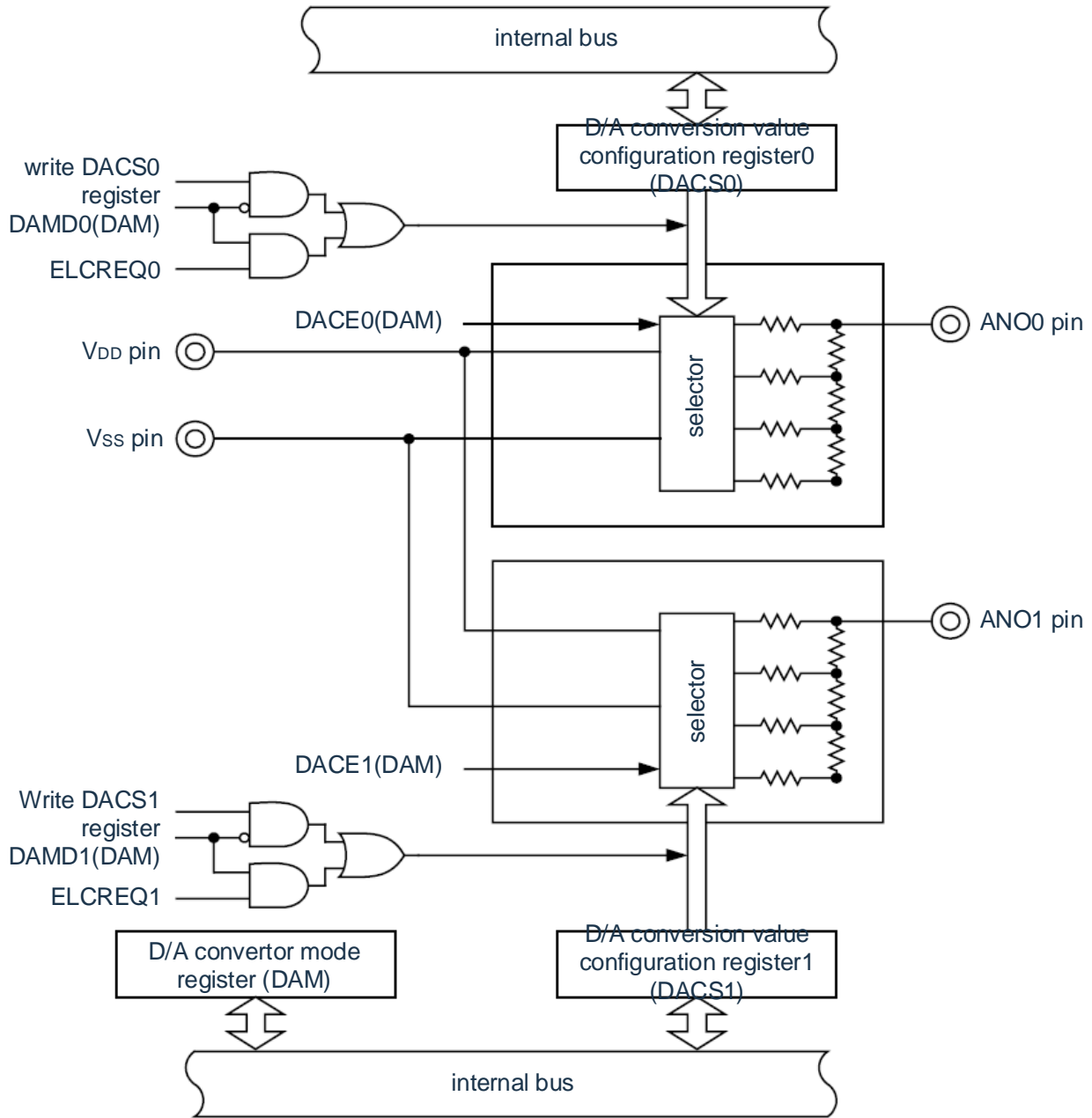
- 8-bit resolution 2ch
- R-2R trapezoidal mode
- Analog output voltage
 - 8-bit resolution: $V_{DD}m8/256$ (m8: the value set for the DACSi register).
- Operating mode
 - Usual mode
 - Real-time output mode

Note $i = 0, 1$

16.2 The structure of the D/A converter

The block diagram of the D/A converter is shown in Figure 16-1.

Fig. 16-1 Block diagram of the D/A converter



Note ELCREQ0 and ELCREQ1 are trigger signals for real-time output mode (event signals for EVENTC).

16.3 Registers that control the D/A converter

The D/A converter is controlled by the following registers.

- Peripheral enable register 1 (PER1).
- Mode register (DAM) for D/A converters
- D/A conversion values set registers 0, 1 (DACS0, DACS1).
- Event output target selection register n (ELSELRn), n=00~21

16.3.1 Peripheral enable register 1 (PER1).

The PER1 register is a register that is set to allow or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use.

To use a D/A converter, bit7 (DACEN) must be set to "1".

The PER1 register is set via the 8-bit memory operation instruction.

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

Figure 16-2 format of Peripheral enable register 1 (PER1)

Address: 0x4002081A after reset: 00H R/W

symbol

PER1

7	6	5	4	3	2	1	0
DACEN	TMBAndN	PGACMPEN	TMMEN	DMAIn	PWMOPEN	TMCEN	TSTONE

DACEN	Control of the input clock of the D/A converter
0	Stop providing the input clock. <ul style="list-style-type: none"> • Cannot write SFR used by D/A converters. • The D/A converter is in a reset state.
1	An input clock is provided. <ul style="list-style-type: none"> • Can read and write SFR used by D/A converters.

Note: To set up a D/A converter, you must first place the DACEN at "1".

When the DACEN bit is "0", the write operation of the control register of the D/A converter is ignored, and the read values are both the initial values (port mode register 2 (PM2) and port register 2). P2) except).

16.3.2 The mode register (DAM) of the D/A converter

This is the register that controls the operation of the D/A converter.

Set the DAM registers via the 8-bit memory manipulation instructions. After generating a reset signal, the value of this register changes to "00H".

Figure 16-3 Format of the mode register (DAM) of a 6-3D/A converter

Address: 40044736H After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
DAM	0	0	DACE1	DACE0	0	0	DAMD1	DAMD0

DACEi	Control of the conversion run of the D/A converter
0	Stop the D/A conversion.
1	Enable D/A conversion.

DAMDi	The choice of operating mode for the D/A converter
0	Usually operating mode
1	Real-time output mode

Note i = 0, 1

16.3.3 The D/A conversion value sets register i (DACS_i) (i=0, 1).

This is the register that sets the analog voltage value output to the ANO0 pin and the ANO1 pin when using the D/A converter. The DACSi registers are set via the 8-bit memory operation instructions.

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

Figure 16-4 The D/A conversion value sets the format of register i (DACS_i) (i=0, 1).

Address: 40044734H (DACS0), 40044735H (DACS1) after reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
DACS _i	DACS _i 7	DACS _i 6	DACS _i 5	DACS _i 4	DACS _i 3	DACS _i 2	DACS _i 1	DACS _i 0

Note The analog output voltage (VANO_i) of the D/A converter is as follows: $VANO_i = V_{DD} \cdot (DACS_i) / 256$

When the D/A converter is not in use, in order to reduce unnecessary current consumption, it is necessary to place the DACE_i position "0" (disable output) and set the DACSi register to "00H" to make R-2R. There is no current flowing through the resistor.

16.3.4 The event output target selection register n (ELSELRn), n=00~21

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 "24.3.1 Event Output Destination Selection Register n(ELSELRn)(n=00~21)".

16.3.5 Registers that control the function of the analog input pin port

Control registers (port mode register (PMxx) and port mode control register (PMC)) for port function multiplexing with the analog output of the D/A converter must be set. For details, please refer to "2.3.1 Port Mode Register (PMxx)" and "2.3" 6-Port Mode Control Register (PMCxx)".

When using the ANO0 pin and the ANO1 pin as analog outputs for the D/A converter, the position of the corresponding port mode register (PMxx) for each port must be "1", And the control register (PMC) is set to the analog output via port mode.

16.4 Operation of the D/A converter

16.4.1 Normal mode of operation

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 Take the DACEN position "1" of the PER1 register (peripheral allow register 1) and start providing the input clock to the D/A converter.
- ② Set the port to the analog pin through the PMC register (port mode control register).
- ③ Place the DAMDi position of the DAM register (the mode register of the D/A converter) "0" (usually 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) is the initial setting.

- ⑤ Place the DACEi position of the DAM register "1" (allow D/A conversion). Start the D/A conversion and output the analog voltage set by (4) to the ANOi pin after the settling time has elapsed.
- ⑥ Thereafter, when you want to perform a D/A conversion, write the DACSi register.

Hold the results of the previous D/A conversion before performing the next D/A conversion.

If the DACEi position of the DAM register is "0" (stops the D/A conversion), the D/A conversion is stopped.

Note 1 Even if the value of the DACEi bit is set to "1" "0" "1", the DACSi will be set after the settling time after the last "1" The analog voltage of the register setting is output to the ANOi pin.

2. If the DACSi register is overwritten during the stable time, the conversion is

aborted and the conversion restarts with the overwritten value. Note i = 0, 1

16.4.2 Operation of real-time output mode

Each channel of the D/A converter is triggered by the event signal of eventC and the D/A conversion is performed.

The setup method is as follows:

- ① Take the DACEN position "1" of the PER1 register (peripheral allow register 1) and start providing the input clock to the D/A converter.
- ② Set the port to the analog pin through the PMC register (port mode control register).
- ③ Place the DAMDi position of the DAM register (the mode register of the D/A converter) "0" (usually mode).
- ④ Sets the analog voltage value of the ANOi pin output to the DACSi register (D/A conversion value setting register i).
- ⑤ Place the DACEi position of the DAM register "1" (allow D/A conversion).

Start the D/A conversion and output the analog voltage set (3) to the ANOi pin after the settling time has elapsed.

- ⑥ The trigger signal for the real-time output mode is set by the event output target selection register n (ELSELRn, n=00~2 1).
- ⑦ Place the DAM Registers at THE DAMDi position "1" (real-time output mode).
- ⑧ Start the run of the event

occurrence source.

The above (1) ~ (8) is the

initial setting.

- ⑨ Thereafter, by generating a trigger signal for the real-time output mode, the D/A conversion begins, and after a 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.

The analog voltage value of the ANOi pin output must be set to the DACSi registers before the next D/A conversion, which generates a trigger signal for real-time output mode.

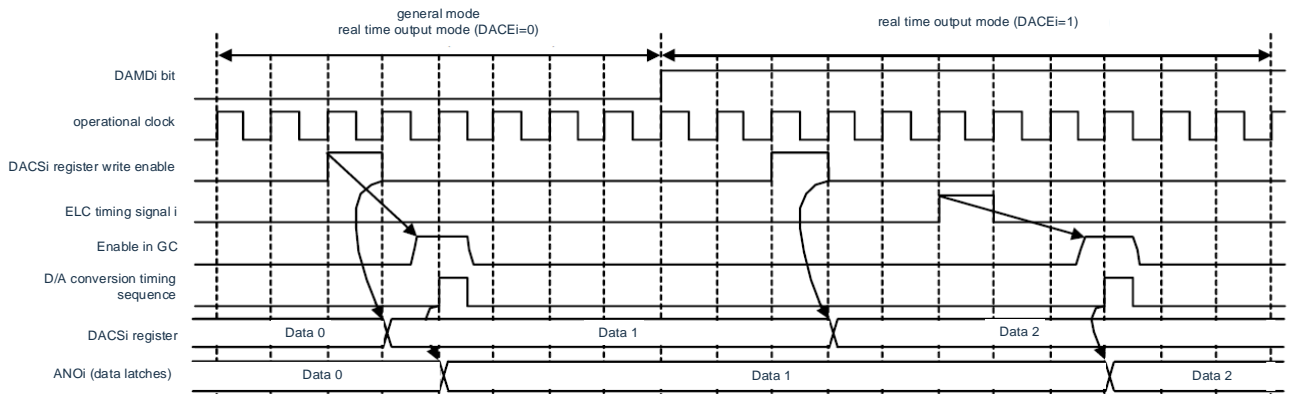
If the DACEi position of the DAM register is "0" (stops the D/A conversion), the D/A conversion is stopped.

Note 1 Even if the value of the DACEi bit is set to "1" "0" "1", the DACSi will be set after the settling time after the last "1" The analog voltage of the register setting is output to the ANOi pin.

2. The interval between generation of trigger signals for the same channel real-time output mode must be greater than the settling time. If a trigger signal for the real-time output mode is generated during the settled 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.

16.4.3 The output timing of the D/A conversion values

The output timing of the D/A conversion values is shown in Figure 16-5.



Note $i = 0, 1$

- Usual operating mode and real-time output mode (where conversion is not allowed).

Write the data latch (output from the ANO_i pin) after writing the DACSi register 1 cycle (running clock).

- Real-time output mode (where conversion is allowed).

Write the data latch (output from the ANO_i pin) after 3 cycles after accepting the EVENTC event signal (running clock).

16.5 Considerations when using D/A converters

Considerations when using D/A converters are as follows.

- (1) When the port is set to an analog pin through the PMC register (port mode control register), the input/output function of the digital ports with the AN0 pin and anO1 pin multiplexing does not work. If you read the P2 register while setting the port to the analog pin via the ADPC register, the value of "0" is read in input mode and the setting value of P2 is read in output mode. In addition, output data is not output to pins even if set to digital output mode.
- (2) In sleep mode and deep sleep mode, the D/A converter continues to run. 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.

Note i = 0, 1

- (3) When stopping the real-time output mode, including when it changes to normal mode, you need to follow these steps:

- Wait at least 3 clocks after stopping the trigger output source, then place the DACEi bit and the DAMDi position "0".
- Place the DACEN position of the PER1 register "0" (stop) after placing the DACEi bit and the DAMDi position "0" (DAC) .
If the DACEN position is "0", all registers inside the DAC are cleared.
Therefore, when running again, each SFR needs to be set.

- (4) When D/A conversion is allowed, A/D conversion cannot be performed on analog input pins that are multiplexed with the AN0 pin and the ANO1 pin.
- (5) In real-time output mode, the value of the DACSi register must be set before generating a trigger signal for real-time output mode.
The setting value of the DACSi register cannot be changed while the trigger signal is active.
- (6) Because the output impedance of the D/A converter is high, it is not possible to draw current from the ANO0 and ANO1 pins. With low input impedance to the load, a tracking amplifier must be inserted between the load and the AN0 pins and the ANO1 pins. In addition, it is important to minimize the routing between the tracking amplifier and the load (due to the high output impedance).
If the wiring is too long, it is necessary to process the grounding graph and so on around the wiring.
- (7) If you want to enter deep sleep mode when the real-time output mode is active, you must disable event links for EVENTC before entering deep sleep mode.

Chapter 17 Comparator

This product has a built-in comparator with 2 channels.

17.1 The functionality of the comparator

The comparator has the following features:

- The CMP1's input pins select external ports, internal reference voltage, and built-in DAC reference voltage.
- When the motor is stopped, the motor position can be detected by comparing the combinations of U, V and W for sensorless motor control.
- Three-phase zero crossover detection can be achieved by switching a comparator.
- Comparator 0 and Comparator 1 comparison results can be pin output (VCOUT0, VCOUT1).
- Supports comparator positive hysteresis, negative hysteresis, and bilateral hysteresis.

Table 17-1 Comparator function summary

project	content
CMP	<ul style="list-style-type: none"> • 2-channel comparators (CMP0 and CMP1) • The comparator negative terminal can select the reference voltage: Selectable external pin input on the negative side of CMP0, built-in reference voltage and internal reference voltage (1.45V) for CMP0 Selectable external pin inputs (4) on the negative side of CMP1, internal reference voltage and internal reference voltage (1.45V) for CMP1 • Programmable internal reference voltage on the negative side (256steps) • The positive end of CMP0 selects the output of the PGA • Selectable External Pin Inputs on the Positive Side of CMP1 (4) • When the positive input voltage > the negative input voltage, the output is high When the positive input voltage < the negative input voltage, the output level 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 effective edge of the comparator output and generates an interrupt signal • Combined with other functions, the initial position of the motor is detected and high/low speed rotation can be controlled The timer's 6-phase pulse-width modulated output can be set/reset to a high-impedance state in overcurrent conditions • Combine with Timer4 to output TIMER WINDOW • Supports comparator positive hysteresis, negative hysteresis, and bilateral hysteresis with 20mV, 40mV, and 60mV delay voltages available

17.2 The structure of the comparator

The block diagram of the comparator is shown in Figure 17-1.

Fig. 17-1 Block diagram of comparator 0

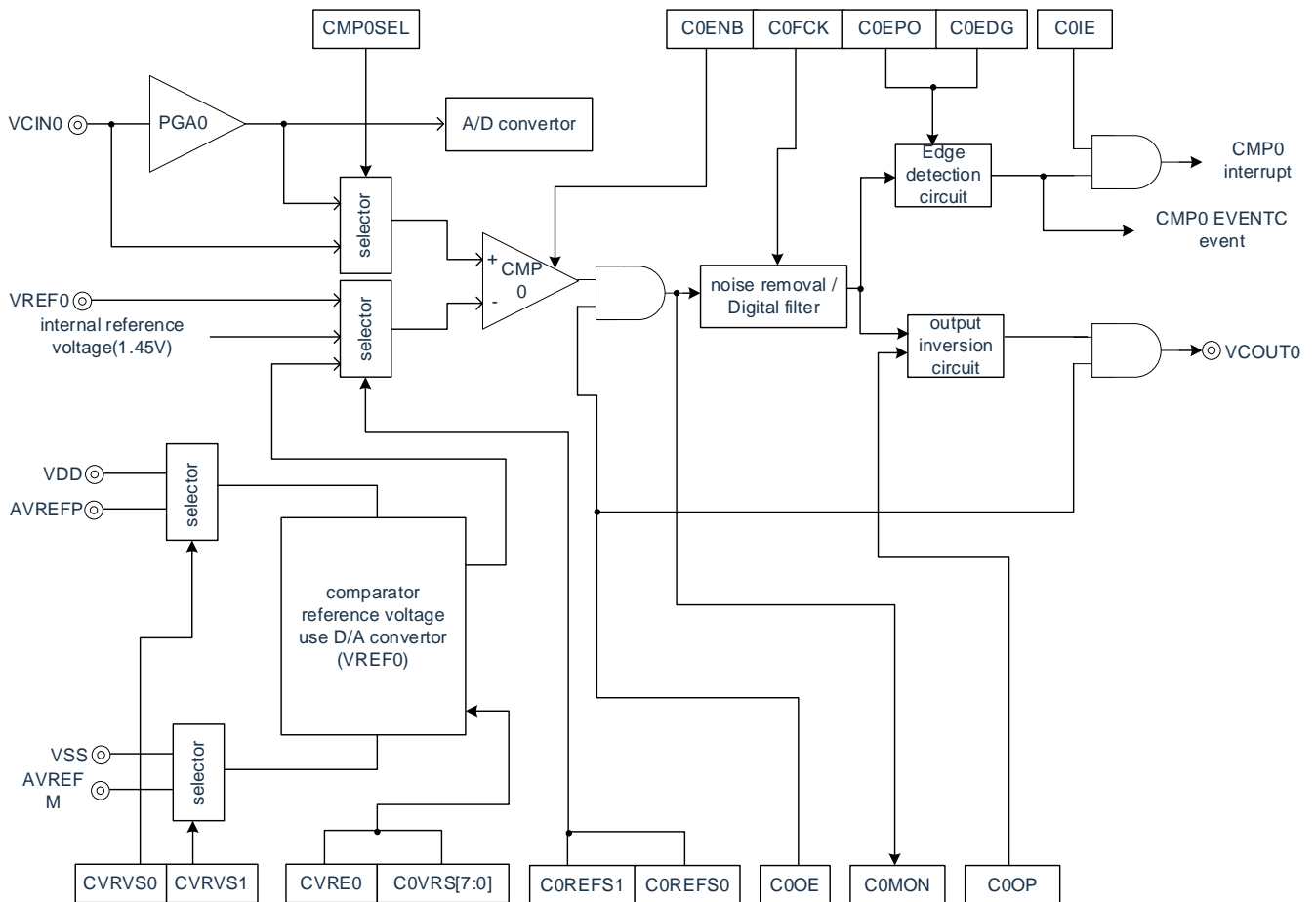
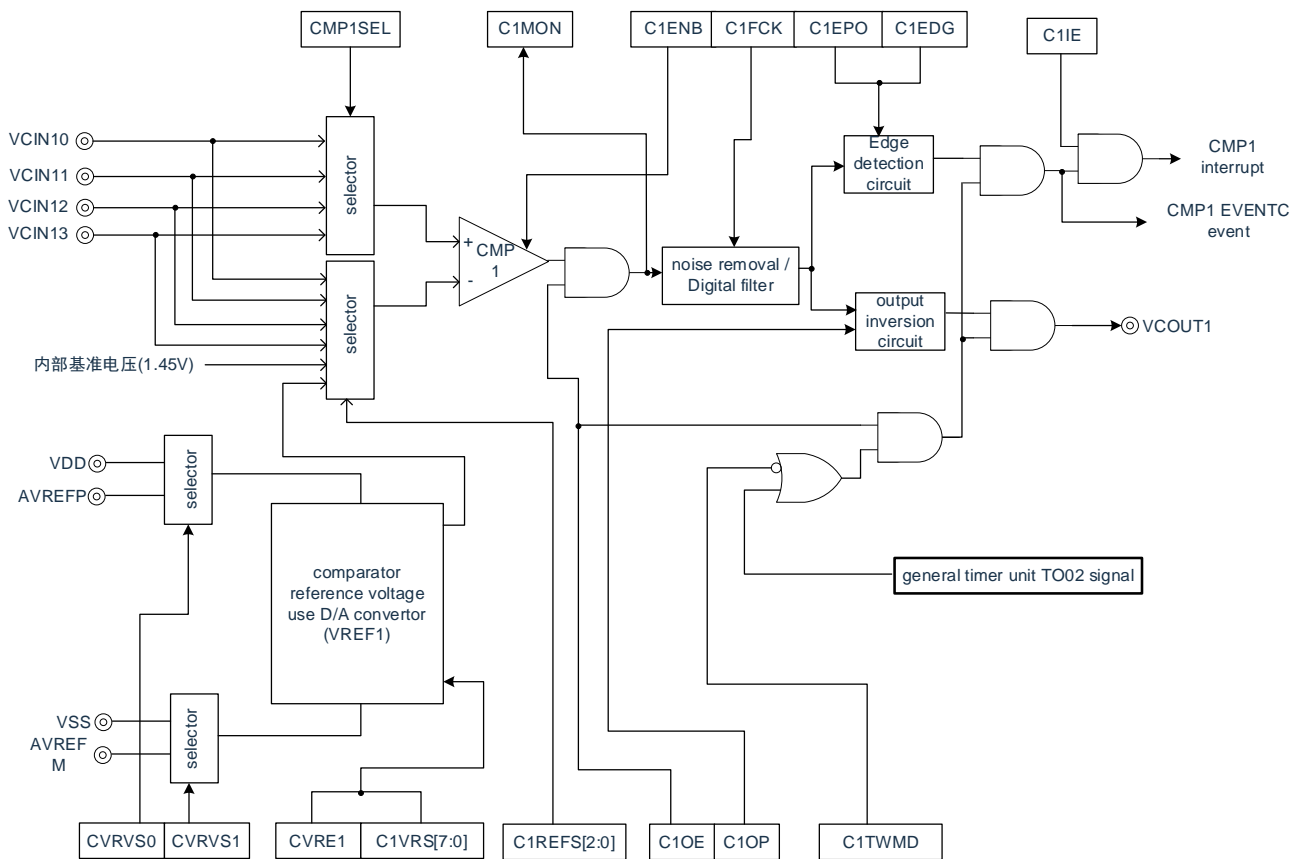


Figure 17-2 Block diagram of comparator 1



17.3 Control registers of the comparator

The registers that control the comparator are shown in Table 17-1.

Table 17-2 controls the registers of the comparator

Register name	symbol
Peripheral enable register 1	PER1
Comparator mode configure registers	COMPMDR
Comparator filter control registers	COMPFIR
Comparator output control registers	COMPOCR
The comparator built-in reference control register	CVRCTL
The comparator built-in reference voltage selection register of 0	C0RVM
Comparator built-in reference voltage selection register1	C1RVM
Comparator 0 input selection control register	CMPSEL0
The comparator 1 input selection control register	CMPSEL1
Comparator 0 hysteresis control register	CMP0HY
Comparator 1 hysteresis control register	CMP1HY
Port mode control registers	PMCxx
Port mode registers	PMxx
Port registers	Pxx

17.3.1 Peripheral enable register 1 (PER1).

The PER1 register is a register that is set to enable or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use.

To use the comparator, bit5 (PGACMPEN) must be set to "1".

The PER1 register is set via the 8-bit memory operation instruction.

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

Figure 17-3 format of the Peripheral enable register 1 (PER1)

Address: 0x4002081A after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
PER1	DACEN	TMBAndN	PGACMPEN	TMMEN	DMAIn	PWMOPEN	TMCEN	TSTONE

PGACMPEN	Control of the comparator input clock
0	Stop providing the input clock. <ul style="list-style-type: none"> • Cannot write SFR used by the comparator. • The comparator is in reset state.
1	An input clock is provided. <ul style="list-style-type: none"> • I can read and write the SFR used by the comparator.

note

To set the comparator, you must first place the PGACMPEN at "1".

When the PGACMPEN bit is "0", the write operation of the comparator's control register is ignored, and the read values are all initial values (except for the port mode register (PMxx) and the port register (Pxx)).

17.3.2 Comparator Mode Setting Register (COMPMDR).

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

The CiENB bit is disabled to "0" when the comparator output is licensed (CiOE position "1" for compoCR registers).

In the following cases, it is forbidden to place the CiENB position "1" (i=0,1):

- The CMP negative input selects the built-in reference voltage and the built-in reference stops (the CVREi bit of the CVRCTL register is "0").
- When the input of CMP0 selects the output of the PGA and the PGA action stops (the CMPSEL0 bit of the CVRCTL register is "1" and the PGAEN bit of the PGAEN register is "0").

Set the COMMDR register via the 8-bit memory operation instruction.

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

Figure 17-4 Comparator Mode Setting Register (COMPMDR) Format

Address: 400438after 40H

reset: 00H R/W

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

C1MON	Monitor flags for comparator 1, note 1, 2
0	VCIN1< comparator1of the reference voltage, or comparator1Stop running.
1	VCIN1> the reference voltage of comparator 1

C1ENB	Comparator 1 runs allowed
0	Disables the operation of comparator 1.
1	Enable comparator 1 to run.

C0MON	Monitor flags for comparator 0, notes 1, 2
0	VCIN0< comparator0of the reference voltage, or comparator0Stop running.
1	VCIN0> comparator 0 reference voltage

C0ENB	Comparator 0 is allowed to run
0	Disables the operation of comparator 0.
1	Enable the comparator 0 to run.

concentrate

1. Immediately after the reset is released, it becomes "0" (initial value), if the C0ENB bit and the C1ENB bit are set to "0" after allowing the comparator to run, it is an indefinite value.
2. Ignore the write value of this bit.

17.3.3 Comparator Filter Control Register (COMPFIR).

The COMPFIR register is the control register for a digital filter. Set the COMFIR registers via the 8-bit memory operation instructions.

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

Figure 17-5 Comparator Filter Control Register (COMPFIR) format

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

Symbol	7	6	5	4	3	2	1	0
COMPBeR	C1EDG	C1EPO	C1FCK1	C1FCK0	C0EDG	C0EPO	C0FCK1	C0FCK0
C1EDG	Comparator 1 edge detection of choice ^{note 1}							
0	Interrupt requests are generated by comparator 1's one-sided edge detection.							
1	An interrupt request is generated by the bilateral edge detection of comparator 1.							
C1EPO	Comparator 1 edge polarity switch ^{note 1}							
0	An interrupt request is generated through the rising edge of comparator 1.							
1	An interrupt request is generated through the falling edge of comparator 1.							
C1FCK1	C1FCK0	Comparator 1 filter selection ^{note 1}						
0	0	Comparator 1 does not have a filter.						
0	1	Comparator 1 has a filter that samples via f_{CLK} .						
1	0	Comparator 1 has a filter that samples by $f_{CLK}/8$.						
1	1	Comparator 1 has a filter that samples via $f_{CLK}/32$.						
C0EDG	Select ^{note 2} for comparator 0 edge detection							
0	An interrupt request is generated by the one-edge detection of comparator 0.							
1	An interrupt request is generated by the bilateral edge detection of comparator 0.							
C0EPO	Comparator 0 edge polarity toggle ^{note 2}							
0	An interrupt request is generated through the rising edge of comparator 0.							
1	An interrupt request is generated through the falling edge of comparator 0.							
C0FCK1	C0FCK0	Comparator 0 filter selection ^{note 2}						
0	0	Comparator 0 does not have a filter.						
0	1	Comparator 0 has a filter that samples via f_{CLK} .						
1	0	Comparator 0 has a filter that samples via $f_{CLK}/8$.						
1	1	Comparator 0 has a filter that samples by $f_{CLK}/32$.						

Note 1 If you change the C1FCK1~C1FCK0 bit, C1EPO bit, and C1EDG bit, you may generate an interrupt request for comparator 1 and to EVENTC The output event signal. These bits must be changed after setting the EVENTC's ELSELR21 register (which does not link the output of comparator 1) to "0". In addition, the IF of the interrupt request flag register must be cleared to "0".

If you change the C1FCK1~C1FCK0 bits from "00B" (comparator 1 without a filter) to another value (comparator 1 has a filter), you must pass before updating the output of the filter After 4 samples, use the interrupt request of comparator 1 or the event signal output to eventC.

2. If you change the C0FCK1 ~C0FCK0 bit, C0EPO bit and C0EDG bit, it may generate an interrupt request of comparator 0 and to eventC the output event signal. These bits must be changed after setting the EVENTC's ELSELR20 register (which does not link the output of comparator 0) to "0". In addition, the IF of the interrupt request flag register must be cleared to "0".

If you change the C0FCK1~C0FCK0 bits from "00B" (comparator 0 without filter) to another value (comparator 0 has a filter), you must pass before updating the output of the filter. After 4 samples, use the interrupt request of comparator 0 or the event signal output to EVENTC.

17.3.4 Comparator Output Control Register (COMPOCR).

COMPOCR registers are control registers that set the polarity of the comparator output, the permission/disable of the output, and the permission/disable of the interrupt output.

In the following cases, it is forbidden to place the CiOE position "1" of the COMCCR register (output license). (i=0,1)

- When the comparator action stops (the CiENB bit of the COMPMDR register is "0").
- The CMP negative input selects the built-in reference voltage and the built-in reference stops (the CVREi bit of the CVRCTL register is "0").
- When the input of CMP0 selects the output of the PGA and the PGA action stops (the CMPSEL0 bit of the CVRCTL register is "1" and the PGAEN bit of the PGAEN register is "0").

Set the COMNCR registers via the 8-bit memory operation instructions.

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

Figure 17-6 Comparator Output Control Register (COMPOCR) Format

Address: 40043842H after reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
COMPOCR	C1OTWMD	C1On	C1OE	C1IE	0	C0On	C0OE	C0IE

C1OTWMD	Comparator 1's TIMER WINDOW output mode controls bit ^{note1}
0	Comparator 1 normal output mode (controlled by C1OE).
1	Comparator 1TIMER WINDOW output mode (controlled by TO02 and C1OE together).

C1OP	The choice of polarity of the VCOUT1 output
0	Output from VCOUT1 comparator 1.
1	Inverting output from VCOUT1 comparator 1.

C1OE	VCOUT1 pin output is allowed in ^{Note 2, Note 3}
0	Disables the VCOUT1 pin output of comparator 1.
1	Enable the VCOUT1 pin output of comparator 1.

C1IE	Comparator 1 interrupts the request for allowed ^{note 4}
0	Suppresses interrupt requests for comparator 1.
1	Enable interrupt requests for comparator 1.

C0OP	VCOUT0 output polarity selection ^{note 5}
0	Output from VCOUT0 comparator 0
1	Inverting output from VCOUT0 comparator 0

C0OE	VCOUT0 pin output is allowed in ^{Notes 5, 6, and 7}
0	Disables the VCOUT0 pin output of comparator 0.
1	The VCOUT0 pin allowing comparator 0 outputs ^{Note 4, Note 8} .

C0IE	Comparator 0 interrupt request allowed ^{note 8}
0	Suppresses interrupt requests for comparator 0.
1	Enable interrupt requests for comparator 0.

Note1: When comparator 1 uses TIMER WINDOW mode, the bit7 (C1EDG) of register COMPFIR must be set to "1".

C1OE and C1OTWMD bits cannot be set at the same time, and the C1OTWMD bit must be set first, and then the C1OE position should be "1".

Note2: When the C1OE bit is rewritten, a comparator 1 interrupt request and an EVENTC event may be generated.

Override this bit after setting the EVENTC's ELSELR21 register to 0 (not linked to the output of comparator 1).

Also, initialize the IF bit of the interrupt request flag register (no interrupt request) after rewriting the C1OE bit.

Note3: When the result of comparator 1 is output to a pin, bit2 (PIOR32) of the PIOR3 register must be set to "1".

Note4: When the C0OE bit is rewritten, a comparator 0 interrupt request and an EVENTC event may be generated.

Override this bit after setting the EVENTC's ELSELR20 register to 0 (not linking to the output of comparator 0).

Also, initialize the IF bit of the interrupt request flag register (no interrupt request) after rewriting the C0OE bit.

Note5: Set the C0OE bit, C0OP bit, and input the result of comparator 9 into the PWM option unit to control the forced cutoff of the PWM output.

Note6: If C1IE is changed from "0" (prohibit interrupt requests) to "1" (enable interrupt requests), the I F of the interrupt request flag register may become "1" (with interrupt requests), Interrupts must therefore be used after the IFL of the interrupt request flag register is cleared "0".

Note7: When the result of comparator 0 is output to a pin, bit1 (PIOR31) of the PIOR3 register must be set to "1"

Note8: If C0IE is changed from "0" (prohibit interrupt request) to "1" (enable interrupt request), the IF of the interrupt request flag register may become "1" (with interrupt request) , so interrupts must be used after clearing the IF of the interrupt request flag register to "0".

17.3.5 The comparator has a built-in reference control register (CVRCTL).

The CVRCTL register is the register that sets the allow/stop action of the comparator's built-in reference voltage.

The CVRCTL register is set via the 8-bit memory operation instruction.

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

Note that the CVRVSi bit of the CVRCTL register is rewritten when the built-in reference stops (CVREi=0).

Figure 17-7 comparators built-in reference control register (CVRCTL) format

Address: 40043843H after reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CVRCTL	0	0	CVRE1	CVRVSN	0	0	CVRE0	CVRVSP

CVRE1	Control bit with built-in reference voltage 1
0	Disables operation of the built-in reference voltage 1
1	Enable operation of the built-in reference voltage 1

CVRVSN	The base-side selection bit of the built-in reference voltage
0	Vss is selected at the ground end of the built-in reference voltage
1	The base end of the built-in reference voltage selects AVREFM ^{Note 1}

CVRE0	Built-in control bit of reference voltage 0
0	Disables operation of the built-in reference voltage 0
1	Enable operation of the built-in reference voltage 0

CVRVSP	Supply-side select bit with built-in reference voltage
0	VDD is selected at the supply end of the built-in reference voltage
1	AvREFP ^{note 2} is selected on the supply side of the built-in reference voltage

Note1: The P21 pin uses both AVREFM and VCIN13, so when the P21 pin is used as the input signal to CMP1, the CVRVSN position "1" is prohibited.

Note2: The P20 pin uses both AVREFP and VCIN12, so when the P20 pin is used as the input signal to CMP1, the CVRVSP position "1" is prohibited

17.3.6 The comparator has a built-in reference voltage selection register (CiRVM).

The CiRVM register is the register that sets the comparator's built-in reference voltage.

When the built-in reference stop action (CVREi=0), the CiRVM register is rewritten

The CVRCTL register is set via the 8-bit memory operation instruction.

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

Figure 17-8 Comparator with the format of the built-in reference voltage selection register i (CiRVM).

Address: 400438434H (C0RVM), 400438435H (C1RVM), reset: 00H R/W

Symbol 7 6 5 4 3 2 1 0

CiRVM	CiRVS7	CiRVS6	CiRVS5	CiRVS4	CiRVS3	CiRVS2	CiRVS1	CiRVS0
-------	--------	--------	--------	--------	--------	--------	--------	--------

CiRV S7	CiRV S6	CiRV S5	CiRV S4	CiRV S3	CiRV S2	CiRV S1	CiRV S0	The comparator's built-in reference voltage is set
0	0	0	0	0	0	0	0	{{(AVREFP or VDD)/256} x0
0	0	0	0	0	0	0	1	{{(AVREFP or VDD)/256} x1
0	0	0	0	0	0	1	0	{{(AVREFP or VDD)/256} x2
0	0	0	0	0	0	1	1	{{(AVREFP or VDD)/256} x3
...								...
1	1	1	1	1	1	0	0	{{(AVREFP or VDD)/256} x252
1	1	1	1	1	1	0	1	{{(AVREFP or VDD)/256} x253
1	1	1	1	1	1	1	0	{{(AVREFP or VDD)/256} x254
1	1	1	1	1	1	1	1	{{(AVREFP or VDD)/256} x255

17.3.7 The input signal of comparator 0 selects the control register (CMPSEL0).

The CMPSEL0 register is the positive end of comparator 0 and the selection register for the negative end of the input signal.

When the comparator 0 stops (C0ENB=0), the CMPSEL0 register is overwritten.

The CMPSEL0 register is set via the 8-bit memory operation instruction.

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

Figure 17-9 The format of input signal selection of Comparator 0 control register (CMPSEL0).

Address: 4004384AH after reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CMPSEL0	CMP0SEL	0	0	0	0	0	C0REFS1	C0REFS0

CMP0SEL	The positive input signal selection bit of comparator 0
0	Select an external pin (VCIN0 pin).
1	Select the PGA output signal

C0REFS1	C0REFS0	The negative input signal selection bit of comparator 0
0	0	Select the built-in reference voltage VREF0
0	1	Select the internal reference voltage (1.45V).
1	0	Select the external pin (IVREF0 pin).
1	1	Prohibit settings

17.3.8 The input signal of comparator 1 selects the control register (CMPSEL1).

The CMPSEL1 register is the positive end of comparator 1 and the selection register for the negative input signal.

When comparator 1 stops (C1ENB=0), rewrite the CMPSEL1 register.

The CMPSEL1 register is set by the 8-bit memory operation instruction.

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

Figure 17-10 The input signal selection of comparator 1 control register (CMPSEL1).

Address: 4004384BH after reset: 00H R/W

Symbol 7 6 5 4 3 2 1 0

CMPSEL1	CMP1SEL1	CMP1SEL0	0	0	0	C0REFS1	C0REFS1	C0REFS0
---------	----------	----------	---	---	---	---------	---------	---------

CMP1SEL1	CMP1SEL0	The positive input signal selection bit of comparator 1
0	0	Select an external pin (VCIN10 pin).
0	1	Select the external pin (VCIN11 pin).
1	0	Select the external pin (VCIN12 pin).
1	1	Select the external pin (VCIN13 pin).

C0REFS2	C0REFS1	C0REFS0	The negative input signal selection bit of comparator 1
0	0	0	Select the built-in reference voltage VREF1
0	0	1	Select the internal reference voltage (1.45V).
0	1	0	Select an external pin (VCIN10 pin).
0	1	1	Select the external pin (VCIN11 pin).
1	0	0	Select the external pin (VCIN12 pin).
1	0	1	Select the external pin (VCIN13 pin).
1	1	0	Prohibit settings
1	1	1	Prohibit settings

Note that when switching the analog input of CMP1, in order to prevent the through current before the two input signals, the switching interval must be above 3us.

17.3.9 Hysteresis control register (CMP0HY) for comparator 0

The CMP0HY register is the hysteresis function control register of comparator 0.
When the comparator 0 stops (C0ENB=0), the CMP0HY register is rewritten.
The CMP0HY register is set via the 8-bit memory operation instruction.
After generating a reset signal, the value of this register changes to "00H".

Figure 17-1 Format of the hysteresis control register (CMP0HY) of comparator 0

Address: 4004384EH after reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CMP0HY	0	0	C0HYSVS1	C0HYSVS0	0	0	C0HYSLS1	C0HYSLS0

C0HYSVS1	C0HYSVS0	The hysteresis voltage selection bit of comparator 0
0	0	No lag
0	1	20mV
1	0	40mV
1	1	60mV

C0HYSLS1	C0HYSLS0	The hysteresis mode of comparator 0 selects the bit
0	0	No lag
0	1	Positive hysteresis
1	0	Negative hysteresis
1	1	Bilateral delay

17.3.10 Hysteresis control register (CMP1HY) for comparator 1

The CMP1HY register is the hysteresis function control register of comparator 1.

When comparator 1 stops (C1ENB=0), rewrite the CMP1HY register.

The CMP1HY register is set via the 8-bit memory operation instruction.

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

Figure 17-2 Format of the hysteresis control register (CMP1HY) of the 2 comparator 1

Address: 4004384FH after reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CMP1HY	0	0	C1HYSVS1	C1HYSVS0	0	0	C1HYSLS1	C1HYSLS0

C1HYSVS1	C1HYSVS0	The hysteresis voltage selection bit of comparator 1
0	0	No lag
0	1	20mV
1	0	40mV
1	1	60mV

C1HYSLS1	C1HYSLS0	Comparator 1's hysteresis mode selects bits
0	0	No lag
0	1	Positive hysteresis
1	0	Negative hysteresis
1	1	Bilateral delay

17.3.11 Registers that control the function of the analog input pin port

When using the VCIN0 pin, VCIN10-VCIN13 pin, and VREF0 pin as analog inputs to the comparator, the corresponding port mode register (PMxx) must be used for each port and the position of the port control register (PMCxx) is "1".

When using the VCOOUT0 and VCOOUT1 functions, control registers for port functions that are multiplexed with the object channel (port mode registers (PMxx) and port registers (Pxx)) and peripheral IO redirect registers (PIOR2, PIOR3). For details, please refer to "2.3.1 Port Mode Register (PMxx)", "2.3.2" Port Register (Pxx)", "2.3.9 Peripheral IO redirect registers (PIOR2) and 2.3.10 Peripheral IO Redirect Register (PIOR3)".

17.4 Run the instructions

Comparators 0 and comparator 1 can operate independently of each other. The setup method is the same as running

CMP0 and PGA can be combined to link.

The setup steps for the independent operation and linkage of the comparator are shown in Table 17-3.

Table 17-3 Comparator-related register setup steps

steps	register	bit	Set the value
1	PGACTL	PGAVG0/1/2	Select buff ^{Note 3}
2	PGACTL	PVRVS	0 (Vss pin selection) ^{Note 3}
3	PGACTL	PGAEN	1 (allowed to run) ^{Note 3}
4	Wait for the PGA to settle down (minimum 10μs).		
5	COMPSELi	CMP0SEL/CMP1SELi1	Comparator i positive input selection
6	COMPSELi	CiREFS	Comparator i negative input selection
7	CiRVM	CiRVSn	Sets the value of the built-in reference voltage
8	CVRCTL	CVRVSi	Select the power supply and ground for the built-in reference voltage
9	CVRCTL	CVREi	1 (built-in reference voltage i enable operation)
10	Wait for the reference voltage to settle (min. 20μs).		
11	Set VCIN0, VCIN1x, IVREF0 pins (input), PGAI (input) ^{Note 3} as analog input functions. PMCxx The VCIN0 pin, VCi and IVREFi pins feature selection with PMCxx position "1" (analog input). Place PMxx at "1" (input mode).		
12	COMPMDR	CiENB	1 (allowed to run)
13	Wait for the settling time of the comparator (min. 3μs).		
14	COMPFIR	CiFCK	With or without a digital filter, select the sample clock.
		CiEOP, CiEDG	Select the edge detection condition for the interrupt request (rising, falling, or bilateral edge).
15	COMPOCR	CiOP, CiOE	Set the output of the VCOUTi (select Polarity, set Enable or Disable Output). Refer to "Output of 17.4.4 Comparator i (i=0, 1)".
		CiIE	Sets the output that enable or disables interrupt requests. Refer to "17.4.4 Comparator i output (i=0, 1)."
		C1OTWMD	Set the TIMER WINDOW output license/disable for comparator 1
16	MKxx ^{Note1}	MKL	When using interrupts: Select Mask interrupts.
17	IFxx ^{Note 1}	IFL	When using interrupts: 0 (no interrupt request: initialization) ^{Note 2}

Note1: MKxx, IFxx is the interrupt control register of the comparator, for details, please refer to "Chapter 25 Interrupt Function".

Note2: Once the comparator is set, by the time of stable operation, unwanted interrupt requests may be generated, and the interrupt request flag bits must be initialized.

Note3: The comparator 0 must be set when the PGA is linked

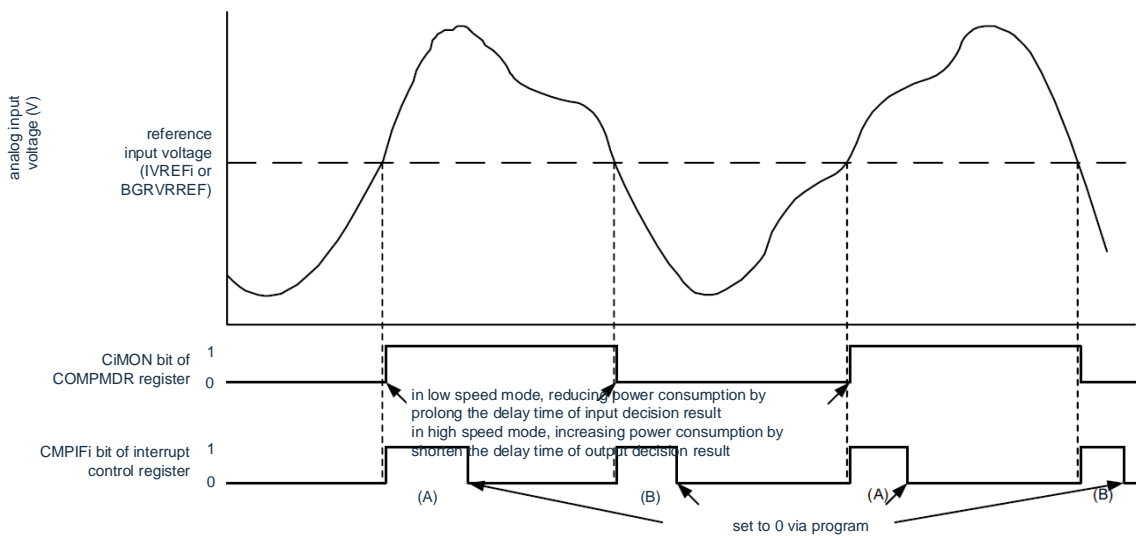
Remark i=0, 1, n=0-7, x=0-3

An example of the operation of comparators $i(i=0, 1)$ is shown in Figure 17-11. In basic mode, when the analog input voltage is higher than the reference input voltage, the CiMON bit of the COMPMDR register is "1"; When the analog input voltage is lower than the reference input voltage, the CiMON bit is "0".

To use the comparator i interrupt, the CiIE position of the COMNOCR register must be "1" (enable interrupt requests). At this point, if the comparison result changes, an interrupt request for comparator i is generated. For details on interrupt requests, refer to "17.4.2 Comparator i Interrupt ($i=0, 1$)".

Figure 17-11 Comparator i ($i=0, 1$) is an example of operation (basic mode).

• basic mode operation example

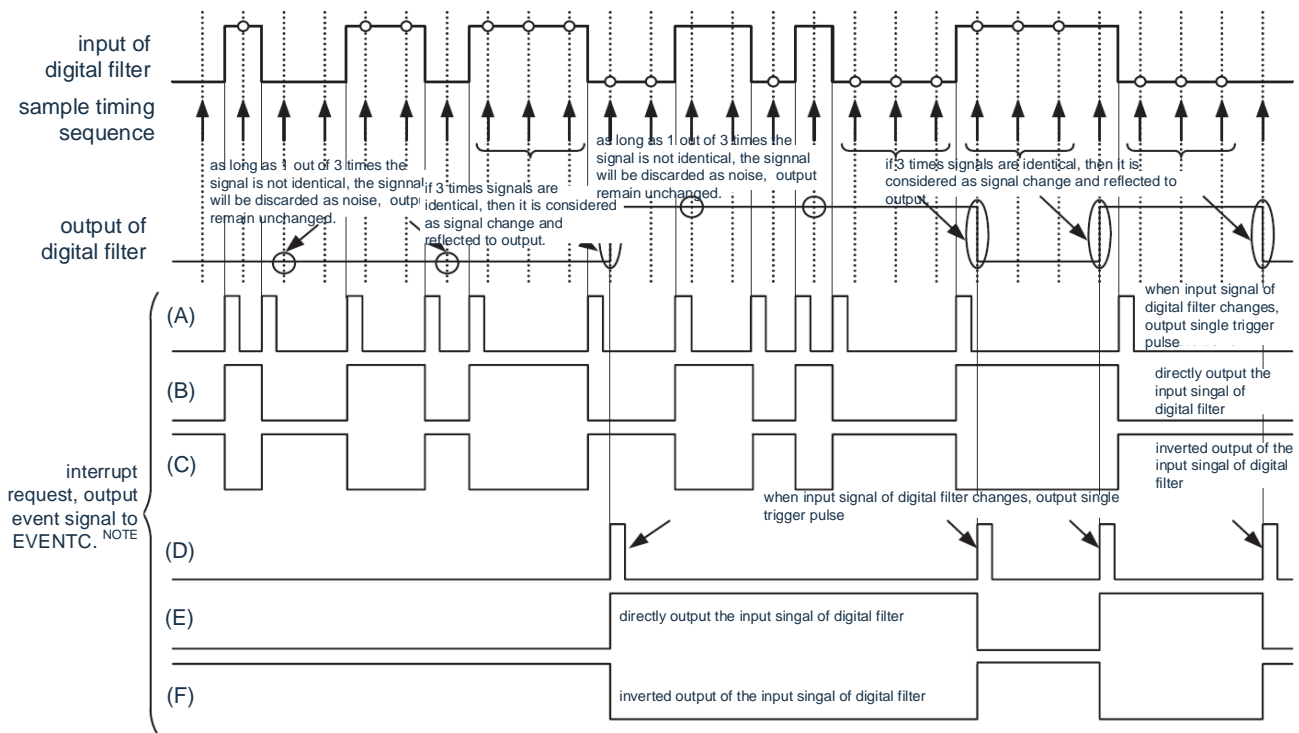


Note The CiFCK1~CiFCK0 bits of the COMFIR register above are "00B" (filterless) and the CiEDG bits are "1". CMPIFi is limited to the case (where the CiEDG bit is "0" and the CiEPO bit is "0" (rising edge). (A) changes the CMPIFi when the CiEDG bit is "0" and the CiEPO bit is "1" (falling edge). Limited to changes in (B).)

17.4.12 The event signal output to the linkage controller (EVENTC).

In the same way that interrupt requests are generated, an event signal is generated to eventc by detecting the output edge of the digital filter set by the COMFIR register. However, unlike interrupt requests, it is independent of the CiIE bit of the COMCCR register and always outputs an event signal to EVENTC. The selection of event output targets and the stop of event links must be set through EVENTC's ELSELR 20 registers and ELSELR21 registers.

Figure 17-14 Operation of a digital filter, interrupt request, and output event signal to EVENTC



Note That when the CiIE bit ($i=0, 1$) is "1", the interrupt request and the event signal output to EVENTC are the same waveform. When the CiIE bit ($i=0, 1$) is "0", only the interrupt request is fixed to "0".

The waveform of (A), (B), (C) is the CiFCK bit of the COMFIR register ($i = 0, 1$) is "1". In the case of 00B" (without a digital filter), the waveforms of (D), (E), (F) are the CiFCK of the COMFIR registers. When the bits ($i=0, 1$) are "01B", "10B", or "11B" (with digital filter). (A) and (D) are the cases where the CiEDG bit is "1" (bilateral edge), (B), (E). Is the case where the CiEDG bit is "0" and the CiEPO bit is "0" (rising edge), (C), (F). Is the case where the CiEDG bit is "0" and the CiEPO bit is "1" (falling edge).

17.4.13 The output of comparator i (i=0, 1).

The comparison results of the comparator can be output to an external pin, and the output polarity (positive or inverting output) can be set through the CiOP and CiOE bits of the COMLOCR register and whether the output is allowed. For register settings and comparator output correspondence, refer to "17.3.4 Comparator Output Control Register (COMPOCR)".

To output the comparison result of the comparator to the output pin of the VCOUTi, you must follow the steps below to set the port (after reset, the port is in the input state):

- ① Set the mode of the comparator (Steps 2 to 5 of "Setting Registers Related to Table 17-3 Comparators").
- ② Sets the VCOUTi output of the comparator (sets the COMCCR register, selects the polarity and enable the output).
- ③ Place the VCOUTi's output pin corresponding to the port mode control register position "0".
- ④ Place the output pin of the VCOUTi at the position "0" of the port register corresponding to it.
- ⑤ Set the port direction register corresponding to the output pin of the VCOUTi to the output (starting with the pin output).

17.4.14 Stop and provision of the comparator clock

In the case of stopping the comparator clock by setting the peripheral enable register 1 (PER1), it must be set as follows:

- ① Place the CiENB position of the COMMDR register "0" (stop the comparison from running).
- ② Place the interrupt request flag register at IF position "0" (clear the unneeded interrupt before the comparator stops).
- ③ Place the PGACMPEN position of the PER1 register "0".

If the clock is stopped by setting the PER1 register, the internal registers of the comparator are all initialized, so to use the comparator again, the registers must be set according to the steps in Table 17-3.

Note 1 If you set the comparator n reference voltage selection bit (CnVRF) of the comparator mode setting register (COMPMDR) to "1" (the comparator n reference voltage is the internal reference voltage (1.45V)), the output of the temperature sensor cannot be A/D converted by an A/D converter.

2. If DMA startup is allowed in one of the following states, the DMA transfer starts and an interruption occurs after the transfer ends. Therefore, it is necessary to enable DMA to start after the monitor flag (CnMON) of the acknowledging comparator.

- Set to generate an interrupt request (CnEDG=0) through the single edge detection of the comparator and an interrupt request (CnEPO=0) through the rising edge of the comparator and $VCIN > VREF$ (or internal reference voltage). 1.45V)。
- Set to produce an interrupt request through the comparator's one-edge detection (CnEDG=0) and an interrupt request is generated through the falling edge of the comparator (CnEPO=1) And $VCIN < VREF$ (Or internal reference voltage.)1.45V)。

(n=0, 1)

Chapter 18 Programmable Gain Amplifier (PGA)

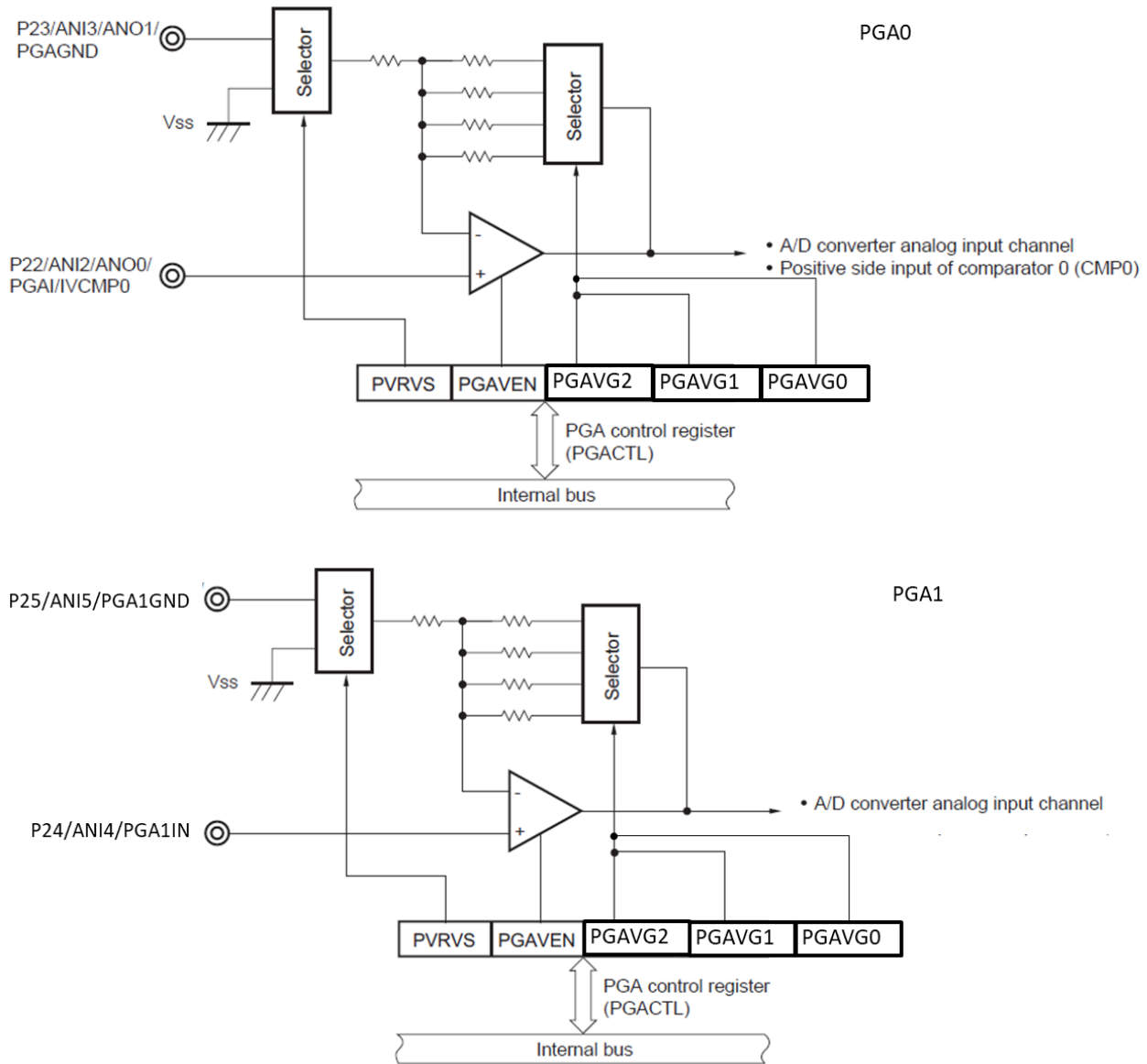
18.1 Programmable gain amplifier function

This product contains two programmable gain amplifiers (PGA0 and PGA1) with the following functions:

- There are 7 options for amplification gain per PGA: 4x, 8x, 10x, 12x, 14x, 16x, 32x
- An external pin can be selected as ground for the PGA negative-side feedback resistor, with PGA0 defaulting to differential mode and PGA1 defaulting to single-ended mode.
- The output of PGA 0 can be selected as an analog input for the A/D converter or as an analog input on the positive end of comparator 0 (CMP0).
- The output of PGA 1 can be selected as the analog input for the A/D converter

18.2 Structure of a programmable gain amplifier

Figure18-1 diagram of a programmable gain amplifier



18.3 Register of a programmable gain amplifier

Table18-1 control registers of the programmable gain amplifier

Register Name	Symbol
Peripheral enable register 1	PER1
PGA control register	PGACTL
Port mode control register 2	PMC2
Port mode register 2	PM2

18.3.1 Peripheral enable register 1 (PER1).

The PER1 register is a register that is set to enable or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use.

To use a programmable gain amplifier, the bit5 (PGACMPEN) of this register must be placed at "1".

The PER1 register is set via 1-bit or 8-bit memory manipulation instructions.

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

Figure18-2 The format of the Peripheral enable register 1 (PER1).

Reset value: 00H R/W

	7	6	5	4	3	2	1	0
PER1	DACEN	TIMERBEN	PGACMPEN	TIMERMEN	DTCEN	PWMOPEN	TIMERCEN	TIMERAEN

PGACMPEN	Control of the input clock of the comparator/programmable gain amplifier
0	Stop providing the input clock. The registers of the comparator or programmable gain amplifier are not writable comparators or the programmable gain amplifier is in a reset state
1	An input clock is provided. Registers of the comparator or programmable gain amplifier are read-write

Note:

Before configuring the registers of the comparator or programmable gain amplifier, confirm that the bit of PGACMPEN is set to 1

If PGACMPEN = 0, writing to the control registers of the comparator or programmable gain amplifier is invalid, and all readout values are default. (Except port mode register 2 (PM2) and port register P2).)

18.3.2 Programmable Gain Amplifier Control Register (PGAnCTL)

The PGA0CTL and PGA1CTL registers are used to control the programmable gain amplifier to start, stop, and amplify.

The PGA0CTL and PGA 1 CTL registers can be set via 1-bit or 8-bit memory manipulation instructions. After generating a reset signal, this register resets to a value of 00H.

Figure18-3 Format of the PGA Control Register (PGAnCTL).

Reset value: 00H R/W

	7	6	5	4	3	2	1	0
PGAnCTL	PGAnEN	-	-	-	PVRVS ^{Note}	PGAnV G2	PGAnV G1	PGAnV G0

n=0.1

PGAnEN	Programmable gain amplifier operation control
0	The amplifier stops working
1	Enable the amplifier to work

	The choice of feedback resistor ground	
PVRVS ^{Note}	PGA0	PGA1
0	Select the PGAnGND pin (default).	Select Vss(default).
1	Select Vss	Select the PGAnGND pin

PGAnVG2	PGAnVG1	PGAnVG0	PGAn gain
0	0	0	4 times
0	0	1	8 times
0	1	0	10 times
0	1	1	12 times
1	0	0	14 times
1	0	1	16 times
1	1	0	32 times
other			Set Prohibit

Note: PGA0 defaults to select the PGA0GND pin as the feedback resistor ground, which is differential mode, and PGA1 selects VSS as the feedback resistor ground, that is, single-ended mode.

Note that when PGAnEN is set to 1, the programmable gain amplifier runs with settling time of 10 us

18.3.3 Registers that control the function of the analog input pin port

When using the PGA0IN pin, PGA1IN pin, PGA0GND pin, and PGA1GND pin as analog inputs to a programmable gain amplifier, the bits of the corresponding port mode register (PMxx) and the port digital/analog control register (PMCxx) must be used for each port location "1".

18.4 Operation of a programmable gain amplifier

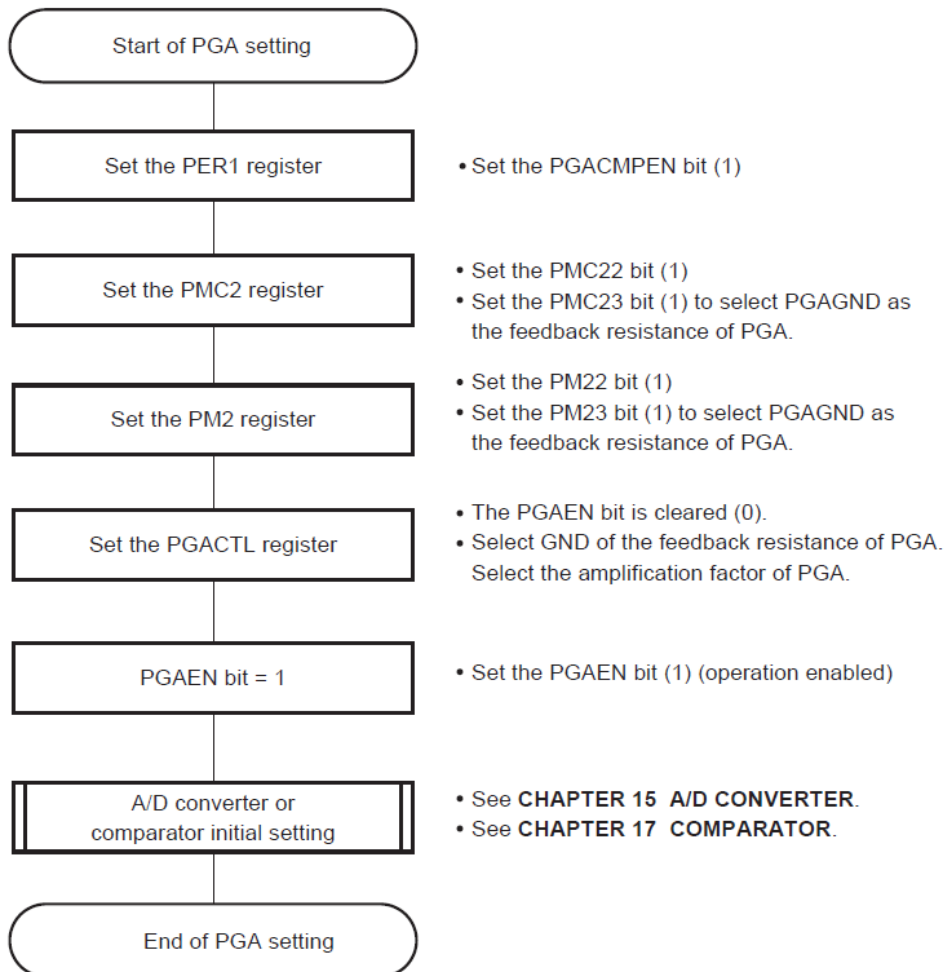
Amplifying the analog voltage of the PGAIN pin input, there are 7 options for amplification gain: 4x, 8x, 10x, 12x, 14x, 16x, 32x.

The amplified voltage can be used for the analog input of the A/D converter and the positive input signal of comparator 0 (CMP 0).

The steps to start and stop the programmable gain amplifier are as follows.

18.4.1 The start-up procedure for the programmable gain amplifier

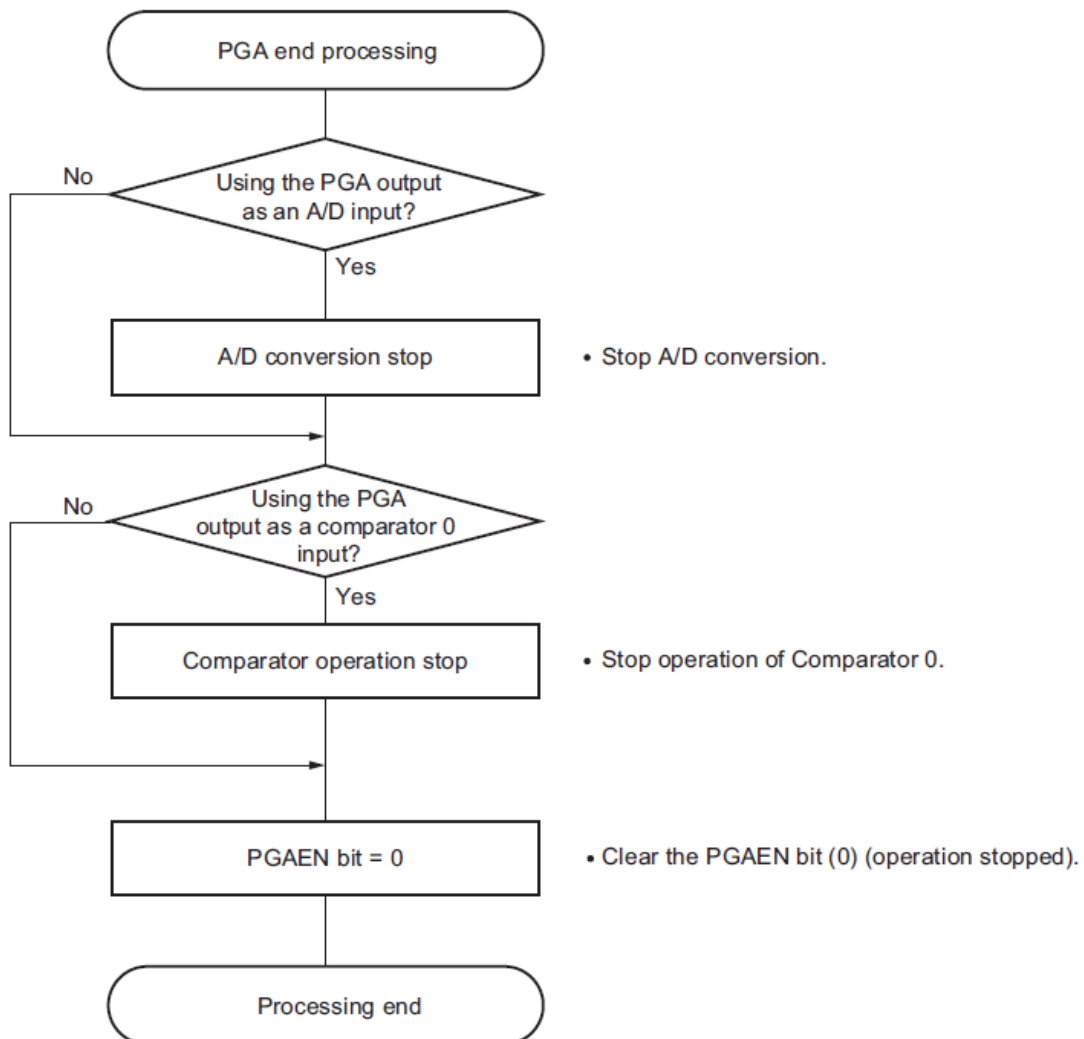
Taking PGA0 as an example, the setup steps are as follows:



Note 1: Setting the PGAEN bit to 1 requires a PGA settling time of 10 μs. Then start the A/D conversion.

18.4.2 The stop-run step of the programmable gain amplifier

Taking PGA0 as an example, the setup steps are as follows:



Note 1: When restarting PGA and A/D conversion or amplifier, a PGA settling time of 10 μ s is required after setting the PGAEN bit to 1.

2. Even if the PGA operation is stopped, the straight-through pin can be used for A/D conversion and comparator action.

Chapter 19 Universal serial communication unit

Unit 0 of the Universal Serial Communication Unit has 4 serial channels, Unit 1 has 2 serial channels, and Unit 2 has 2 serial channels, each channel can achieve 3-wire serial SSPI, UART and simple I²C Communication features. The function assignments for each channel supported by this product are as follows:

unit	passage	Used as SSPI	Used as a UART	Used as a simple I ² C
0	0	SSPI00 (Support slave selection input function)	UART0	IIC00
	1	SSPI01		IIC01
	2	SSPI10	UART1	IIC10
	3	SSPI11		IIC11
1	0	SSPI20	UART2	IIC20
	1	SSPI21		IIC21
2	0	SSPI30	UART3	IIC30
	1	SSPI31		IIC31

When using UART0 for channels 0 and 1 of unit 0, SSPI00 and SSPI01 cannot be used, but channel 2 can be used and SSPI10, UART1 and IIC10 for channel 3.

Note the following in this chapter primarily describes the cell and channel structure of a 10-pin product.

19.1 Functions of the Universal Serial Communication Unit

The characteristics of each serial interface supported by this product are as follows.

19.1.1 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21, SSPI30, SSPI31)

Data is sent and received synchronously with the serial clock (SCLK) output of the master device.

This is a total of 1 Serial Clock (SCLK), 1 Transmit Serial Data (SDO), and 1 Receive Serial Data (SDI).

Clock synchronization communication function for communication by 3 communication lines.

For specific setup examples, refer to "19.53-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI20, SSPI20, SSPI20, Operation of SSPI21, SSPI30, SSPI31) communication".

[Sending and receiving data].

- 7-bit or 8-bit data length (SCI0).
- Data length of 7-16 bits (SCI1 and SCI2).
- Phase control of sending and receiving data
- MSB/LSB preferred

[Clock Control]

- Master or Slave selection
- Phase control of input/output clocks
- Set the transmission period generated by the prescaler and the internal counter of the channel.
- Maximum transfer rate ^{note}
Master communication: $\text{Max.f}_{\text{CLK}}/2$
Slave communication:
 $\text{Max.f}_{\text{MCK}}/6$

[Interrupt function].

- Transmit end interrupt, buffer empty interrupt

[Error Detection Flag].

- Overflow error

Note: Must be used within the range that satisfies the SCLK Cycle Time (t_{KCY}) characteristic. Please refer to the data sheet for details.

19.1.2 UART (UART0~UART3)

This is a function of asynchronous communication over two lines: Serial Data Transmission (TxD) and Serial Data Receiving (RxD). Using these two communication lines, data is sent and received asynchronously (using the internal baud rate) by data frame (consisting of start bits, data, parity bits, and stop bits). Full-duplex UART communication can be achieved by using two channels dedicated for transmit (even channel) and receive dedicated (odd channel).

For a specific setup example, please refer to "19.7 Operation of UART (UART0~UART3) communication".

[Sending and receiving data].

- 7-bit, 8-bit, or 9-bit data length (SCI0).
7-bit, 8-bit, 9-bit, or 16-bit data lengths (SCI1 and SCI2).
- MSB/LSB preferred
- Level setting of sending and receiving data, selection of inverted phase
- Additional, parity function of parity bits
- Additional stop bits

[Interrupt function].

- Transmit end interrupt, buffer empty interrupt
- Error interrupts caused by frame errors, parity errors, or overflow errors

[Error Detection Flag].

- Frame error, parity error, overflow error

19.1.3 Simple I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21, IIC30, IIC31)

This is the function of clock synchronization communication with multiple devices through two lines of serial clock (SCL) and serial data (SDA). Because this simple I²C is designed for single communication with devices such as EEPROM, flash memory, A/D converters, etc., it is only used as a master device.

For start and stop conditions, AC specifications must be adhered to and processed by software while operating the control registers. For specific setup examples, refer to "19.9 Simple I²C (IIC00, IIC01, IIC10, IIC11, IIC20, Operation of IIC21, IIC30, IIC31) communication communications.

[Sending and receiving data].

- Master send, master receive (limited to single master function).
- ACK output function ^{note}, ACK detection function
- 8 bits data length (when sending an address, specify the address with a high 7 bits, and R/W control with the lowest bit).
- Manual generation of start conditions and stop conditions

[Interrupt function].

- End of transfer interrupted

[Error Detection Flag].

- ACK error, overflow error

※[Function not supported by Simple I²C].

- Slave sending, Slave receiving
- Quorum failure detection feature
- Wait for detection feature

Note When receiving the last data, if you write "0" to the SOEmn bit (serial output allow register m(SOEm)) to stop the output of the serial communication data, the ACK is not output. For details, please refer to "19.9.3(2) Process".

Note When using the fully functional I²C C-bus, please refer to "Chapter 20 Serial Interface IICA".

19.2 The structure of a universal serial communication unit

The Universal Serial Communication Unit consists of the following hardware.

Table 19-1 Structure of the Universal Serial Communication Unit

project	structure
Shift register	SCI0: 8 digits or 9 bits ^{note 1} SCI1/SCI2: 16 bits
Buffer registers	SCI0: Low 8 bits of serial data register mn (SDRmn) OR 9 bits ^{note 1, 2} SCI1/SCI2: Serial data register mn (SDRmn) ^{Note 3}
Serial clock input/output	SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, SCLK21, SCLK30, SCLK31 pin (for 3-wire serial I/O), SCL00, SCL01, SCL10, SCL11, SCL20, SCL21, SCL30, SCL31 pin (for simple I ² C)
Serial data input	SDI00, SDI01, SDI10, SDI11, SDI20, SDI21, SDI30, SDI31 pin (for 3-wire serial I/O), Rx0, Rx1, Rx2, Rx3 pins (for UART).
Serial data output	SDO00, SDO01, SDO10, SDO11, SDO20, SDO21, SDO30, SDO31 pin (for 3-wire serial I/O), Tx0, Tx1, Tx2, Tx3 pins (for UART).
Serial data input/output	SDA00, SDA01, SDA10, SDA11, SDA20, SDA21, SDA30, SDA31 pin (for simple I ² C)
Slave selection inputs	SS00 pin for slave selection input function
Control registers	< Module configuration Register > • Peripheral enable register 0/2 (PER0/2). • Serial clock selection register m (SPSm). • Serial channel enable status register m(SEm). • Serial channel start register m(SSm). • Serial channel stop register m(STm). • Serial output enable register m (SOEm). • Serial output register m(SOm). • Serial output level register m(SOLm). • Input Switch Control Register (ISC). • Noise filter enable register 0 (NFEN0).
	<register for each channel section > • Serial data register mn (SDRmn). • Serial mode register mn (SMRmn). • Serial communication runs set register mn (SCRmn). • Serial status register mn (SSRmn). • Serial flag clears the trigger register mn (SIRmn).
	•Port multiplexing function configuration register (PxxcFG). •Port Output Mode Register (POMxx). •Port mode register (PMxx). •Port register (Pxx).

Note 1 The number of bits used as shift registers and buffer registers varies depending on the cell and channel.

- mn=00, 01: 9 bits lower
- mn= 02, 03: 8 bits lower

2. Depending on the communication mode, the low 8 bits of the serial data register mn (SDRmn) can be read and written with the following SFR name.

- SSPIp communication... SIOp (SSPIp Data Register).
- UARTq receives... RXDq (UARTq Receive Data Register).
- UARTq sends... TXDq (UARTq Transmit Data Register).
- IICr communication... SIOr (IICr Data Register).

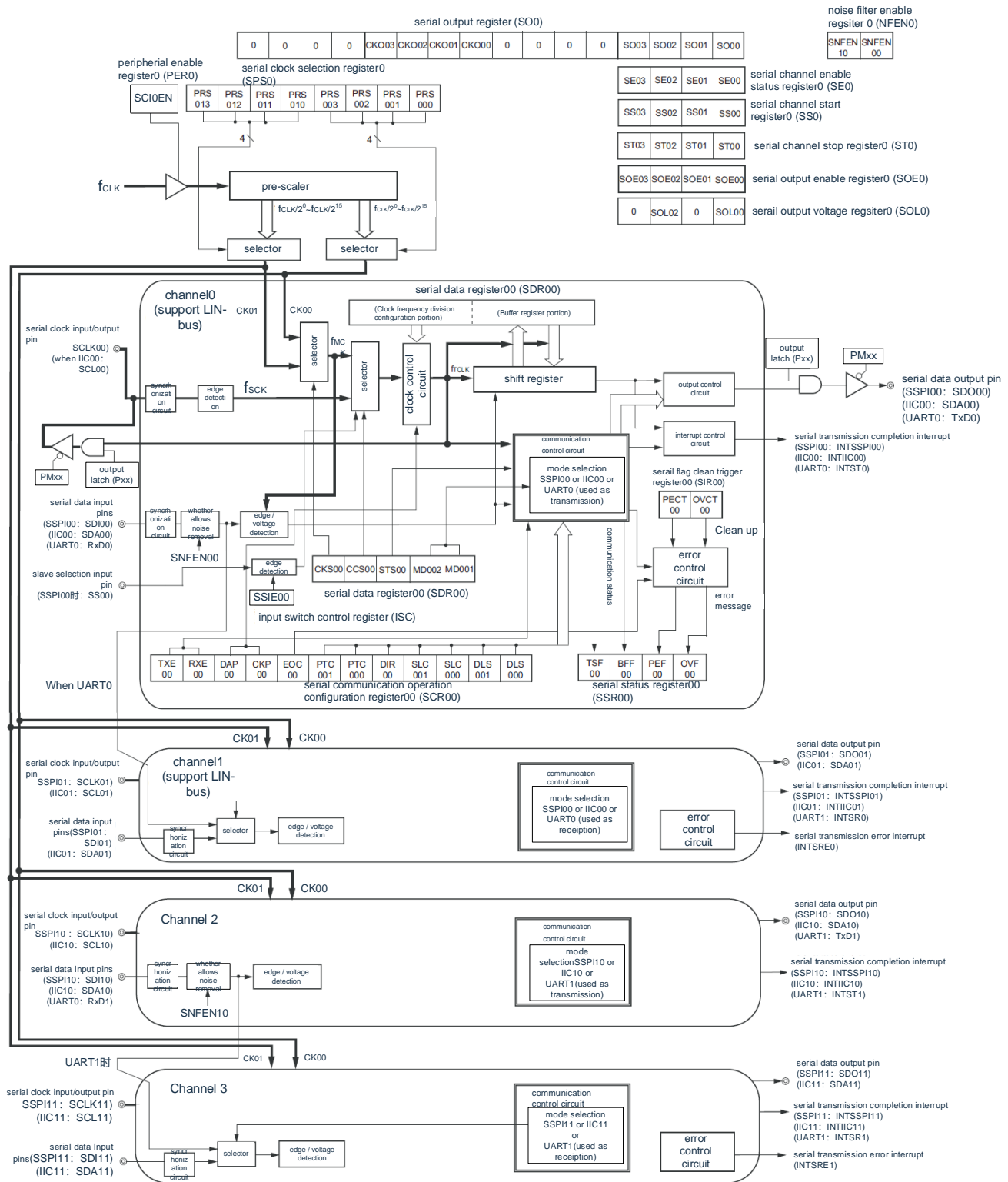
3. During the action of SEMn=1.

Remark m: unit number (m=0, 1, 2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)

q: UART number (q=0~3) r: IIC number (r=00, 01, 10, 11, 20, 21, 30, 31)

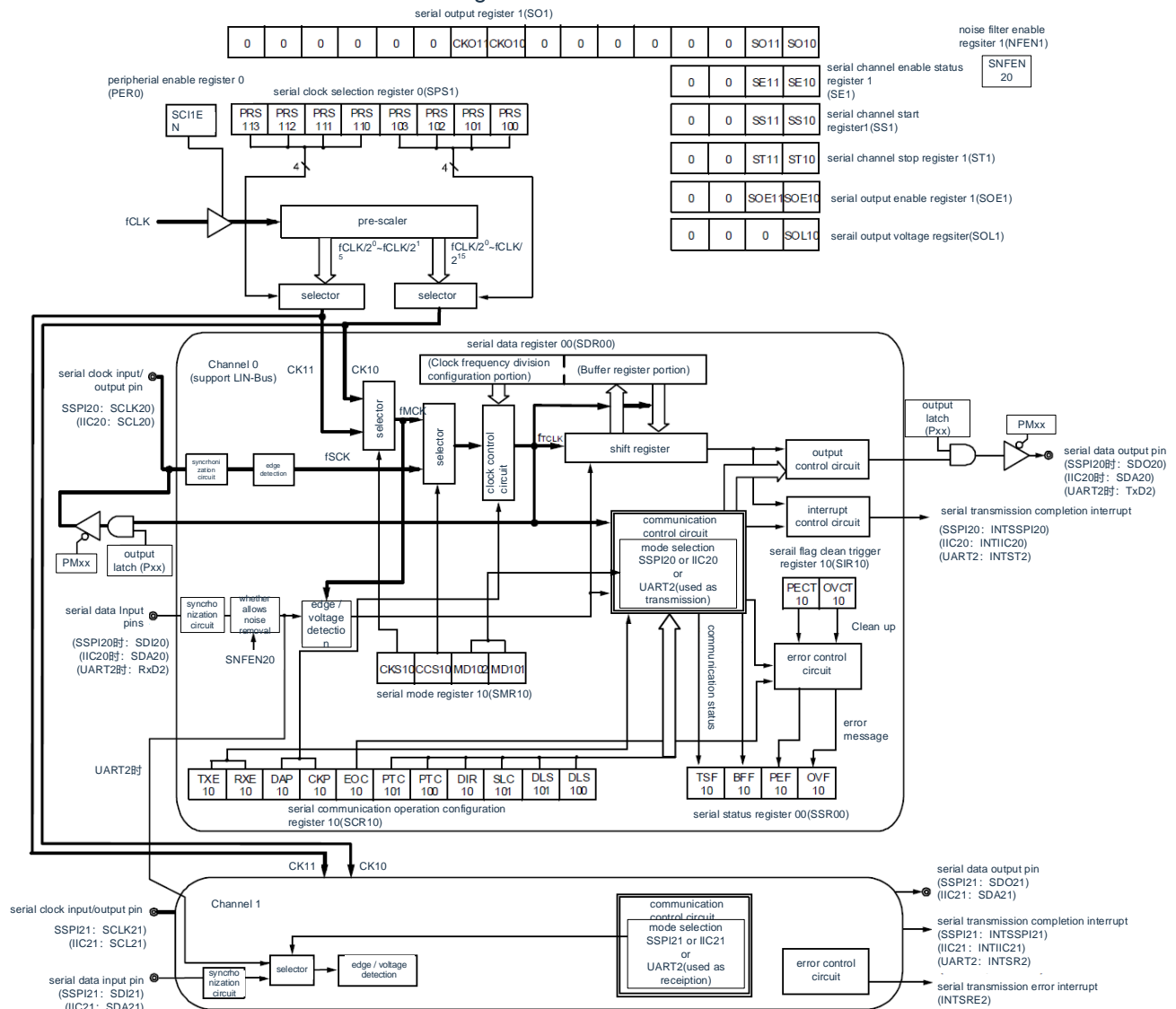
Block diagram of universal serial communication unit 0 is shown in Figure 19-1.

Figure 19-1 Block diagram of the Universal Serial Communication Unit 0



Block diagram of the universal serial communication unit 1 is shown in FIG 19-2.

FIG 19-2 Block diagram of the universal serial communication unit 1



The structure of the universal serial communication unit 2 is the same as that of the universal serial communication unit 1.

19.2.1 Shift Register (SCI0)

This is a 9-bit register that converts parallel and serial to and from each other.

For UART communication at 9 bits of data length, use 9 bits (bit0 to 8) ^{Note 1}。 Convert the input data of the serial input pins into parallel data when receiving data; When data is sent, the value that will be transferred to this register is output as serial data from the serial output pin ^{note 1}. Shift registers cannot be manipulated directly through the program.

To read and write data from the shift register, use the low 8 bits or 9 bits low of the serial data register mn (SDRmn).



19.2.2 The serial data register mn (SDRmn) is either 8 bits low or 9 bits low (SCI0).

The SDRmn register is the transmit and receive data registers (16 bits) of channel n.

Bit8~0 (low 9 bits) ^{note} or bit7~0 (low 8 bits) is used as transmit and receive buffer registers Bit15~9 is used as a crossover setting register for the operating clock (f_{MCK}).

When receiving data, save parallel data converted by shift registers to a low 8 bit or a low 9 bit; When sending data, the transmit data transmitted to the shift register is set to 8 bits low or 9 bits low.

Regardless of the output order of the data, bit0 and bit1 (DLSmn0, DLSmn1) of the set register mn (SCRmn) are run according to serial communication) settings, save to the low 8 bits or low 9 bits data as follows:

- 7 bits of data length (bit0~6 saved in the SDRmn register).
- 8 bits of data length (bit0~7 saved in the SDRmn register).
- 9 bits of data length (bit0~8 saved in the SDRmn register) ^{Note 1}

SDRmn registers can be read and written in 16-bit increments.

Depending on the communication mode, the lower 8 bits of the SDRmn register or the low 9 bits of the SDRmn register can be read and written in 8 bits by the following SFR name.

- SSPIp communication... SDIOp (SSPIp Data Register).
- UARTq receives... RXDq (UARTq Receive Data Register).
- UARTq sends... TXDq (UARTq Transmit Data Register).
- IICr communication... SDIO r (IICr Data Register).

After generating the reset signal, the value of the SDRmn register changes to "0000H".

Note 1 Only UART0 supports 9 bits of data length.

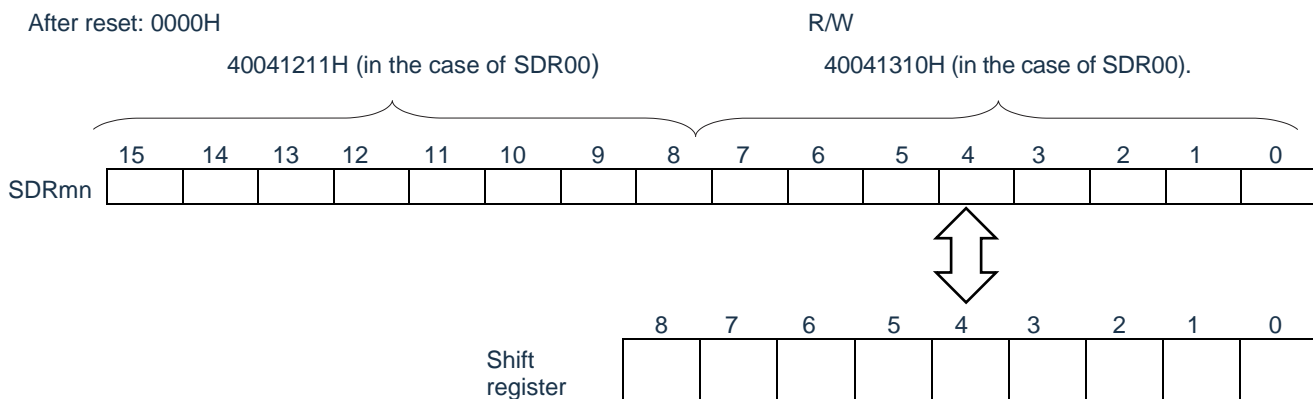
2. When the operation stops ($SEmn=0$), it is forbidden to rewrite SDRmn [7:0] through the 8-bit memory operation instruction (otherwise, SDRmn [15:9] is all cleared "0").

Note 1 At the end of the reception, bits from bit0 to 8 that exceed the length of the data are bits "0".

2.m: unit number ($m=0$) n: channel number ($n=0, 1$) p: SSPI number ($p=00, 01, 10, 11$)

q: UART number ($q=0, 1$) r: IIC number ($r=00, 01, 10, 11$)

Figure 19-3 the format of Serial data register mn (SDRmn) (mn = 00, 01, 10, 11),



Note For the high 7-bit function of the SDRmn register, refer to "19.3 Registers for Controlling Universal Serial Communication Units".

19.2.3 Shift Register (SCI1/SCI2)

This is a 16-bit register that converts parallelly and serially to and from each other.

Convert the input data of the serial input pins into parallel data when receiving data; When data is sent, the value that will be passed to this register is output from the serial output pin as serial data. Shift registers cannot be manipulated directly through the program.

To read and write data from the shift register, use the serial data register mn (SDRmn) during operation (SEmn=1).

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

19.2.4 Serial data register mn(SDRmn) (SCI1/SCI2).

The SDRmn register is the transmit and receive data registers (16 bits) of channel n.

When the operation stops (SEmn=0), bit15~9 is used as a crossover setting register for the operating clock (f_{MCK}). During operation (SEmn=1) bit15~9 is used as a transmit and receive buffer register.

When receiving data, save parallel data converted by shift registers to serial data register SDRmn; When sending data, the transmit data that will be transferred to the shift register is set to the serial data register SDRmn.

Regardless of the output order of the data, bit3 to bit 0 (DLSmn3~) of the set register mn (SCRmn) is run according to the serial communication DLSmn0 settings, the data saved to the SDRmn register is as follows:

- 7 bits of data length (bit0~6 saved in the SDRmn register).
- 8 bits of data length (bit0~7 saved in the SDRmn register).
- :
- 16-bit data length (bit0~15 saved in the SDRmn register).

SDRmn registers can be read and written in 16-bit increments.

At SEmn=1, the lower 8 bits of the SDRmn register can be read and written as SDRmnL in 8 bits.

Depending on the communication mode, the SDRmnL register can be read and written with the following SFR name.

- SSPIp communication... SDIOpL
- UARTq receives... RXDq (UARTq Receive Data Register).
- UARTq sends... TXDq (UARTq Transmit Data Register).
- IICr communication... SDIO r (IICr Data Register).

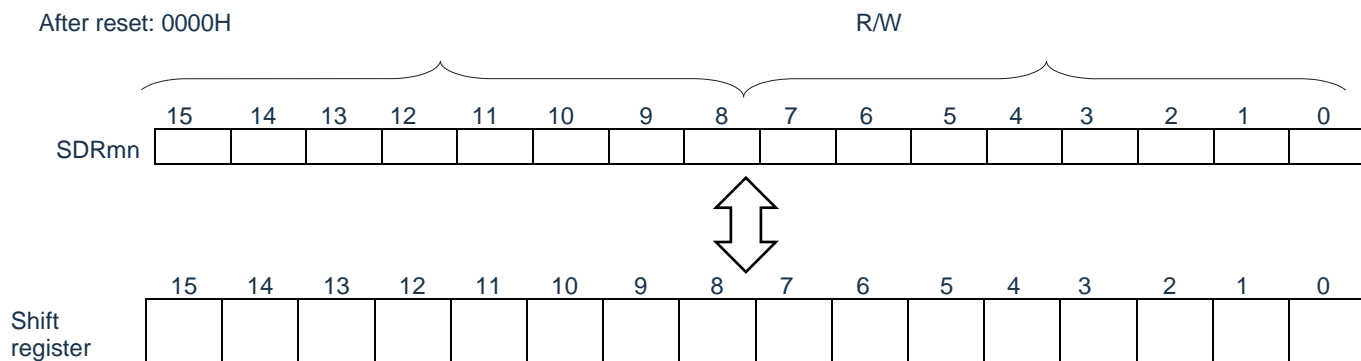
After generating the reset signal, the value of the SDRmn register changes to "0000H".

Note 1 At run stop (SEmn=0), it is forbidden to rewrite SDRmn [7:0] via the 8-bit memory operation instruction (otherwise, SDRmn[15:9] is all cleared "0").

Note 1.m: Unit number (m=1, 2) n: Channel number (n=2, 3). p: SSPI number (p=20, 21, 30, 31)

q: UART number (q=2, 3) r: IIC number (r=20, 21, 30, 31)

Figure 19-4 Serial data register mn (SDRmn) (mn = 20, 21, 30, 31) format



Note For the high 7-bit function of the SDRmn register, refer to "19.3 Registers for Controlling Universal Serial Communication Units".

19.3 Control registers of the universal serial communication unit

The registers that control the universal serial communication unit are as follows:

Register base address: SCI0=4004_1100H; SCI1=4004_1400H; SCI2=4004_1600H;

Register name	Register description	R/W	Reset value	Register address
PER0	Peripheral enable register 0	R/W	00H	0x40020420H
PER2	Peripheral enable register 2	R/W	00H	0x40020422H
SSR00/01/02/03	Serial status register 00/01/02/03	R/W	0000H	SCI0+00H/02H/04H/06H
SIR00/01/02/03	The serial flag clears the trigger registers 00/01/02/03	R/W	0000H	SCI0+08H/0AH/0CH/0EH
SMR00/01/02/03	Serial mode register 00/01/02/03	R/W	0020H	SCI0+10H/12H/14H/16H
SCR00/01/02/03	Serial communication runs set register 00/01/02/03	R/W	0087H	SCI0+18H/1AH/1CH/1EH
SE0	The serial channel enable status register 0	R/W	0000H	SCI0+20H
SS0	Serial channel start register 0	R/W	0000H	SCI0+22H
ST0	Serial channel stop register 0	R/W	0000H	SCI0+24H
SPS0	Serial clock selection register 0	R/W	0000H	SCI0+26H
SO0	Serial output register 0	R/W	0F0FH	SCI0+28H
SOE0	The serial output enable register 0	R/W	0000H	SCI0+2AH
SOLO	Serial output level register 0	R/W	0000H	SCI0+34H
SDR00/01/02/03	Serial data register 00/01/02/03	R/W	0000H	SCI0+210H/212H/214H/216H
SSR10/11	Serial status register 10/11	R/W	0000H	SCI1+00H/02H
SIR10/11	The serial flag clears trigger registers 10/11	R/W	0000H	SCI1+04H/06H
SMR10/11	Serial mode registers 10/11	R/W	0020H	SCI1+08H/0AH
SCR10/11	Serial communication runs set register 10/11	R/W	0087H	SCI1+0CH/0EH
SE1	The serial channel enable status register 1	R/W	0000H	SCI1+10H
SS1	Serial channel start register 1	R/W	0000H	ICS1+12H
ST1	Serial channel stop register 1	R/W	0000H	ICS1+14H
SPS1	Serial clock selection register 1	R/W	0000H	ICS1+16H
SO1	Serial output register 1	R/W	0303O'CLOCK	SCI1+18H
SOE1	The serial output enable register 1	R/W	0000H	SCI1+1AH
SOL1	Serial output level register 1	R/W	0000H	ICS1+20H
SDR10/11	Serial data registers 10/11	R/W	0000H	SCI1+110H/112H

Register base address: SCIO=4004_1100H; SCI1=4004_1400H; SCI2=4004_1600H;

Register name	Register description	R/W	Reset the value	Register address
SSR20/21	Serial status register 20/21	R/W	0000H	SCI2+00H/02H
SIR20/21	The serial flag clears the trigger register 20/21	R/W	0000H	SCI2+04H/06H
SMR20/21	Serial mode register 20/21	R/W	0020H	SCI2+08H/0AH
SCR20/21	Serial communication runs set register 20/21	R/W	0087H	SCI2+0CH/0EH
SE2	The serial channel enable status register 2	R/W	0000H	SCI2+10H
SS2	Serial channel start register 2	R/W	0000H	SCI2+12H
ST2	Serial channel stop register 2	R/W	0000H	SCI2+14H
SPS2	Serial clock selection register 2	R/W	0000H	ICS2+16H
SO2	Serial output register 2	R/W	0303O'CLOCK	SCI2+18H
SOE2	The serial output enable register 2	R/W	0000H	SCI2+1AH
SOL2	Serial output level register 2	R/W	0000H	SCI2+20H
SDR20/21	Serial data register 20/21	R/W	0000H	SCI2+110H/112H
ISC	The input switches the control registers	R/W	0000H	0x40040473
NFEN0	Noise filter enable registers	R/W	0000H	0x40040470
POMx	Port output mode registers	R/W	0000H	0x4000004x
PIMx	Port input mode registers	R/W	0000H	0x4000005x
PMx	Port mode registers	R/W	0000H	0x4000032x
Px	Port registers	R/W	0000H	0x4000030x

19.3.1 Peripheral enable register 0/2 (PER0/PER2).

Per0/2 registers are registers that are set to allow or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use.

To use Universal Serial Communication Unit 0, the bit2 (SCI0EN) of PER0 must be placed on "1".

To use Universal Serial Communication Unit 1, the bit3 (SCI1EN) of PER0 must be placed on "1".

To use Universal Serial Communication Unit 2, the bit3 (SCI2 EN) of PER2 must be placed on "1".

For details, refer to "4.3.7 Peripheral Allow Registers 0, 1, 2 (PER0, PER1, P ER2) "

Note 1 To set the universal serial communication unit m, the following registers must first be set in the SCImEN bit "1".

When the SCImEN bit is "0", the write operation of the control register of the universal serial communication unit m is ignored, and the read values are initial values (input switching control register (ISC), Noise filter enable register 0 (NFEN0), port input mode register (PIMx), port output mode register (POMx), Except for the port mode registers (PMx), port mode control registers (PMCx), and port registers (Px).

- Serial clock selection register m (SPSm).
- Serial mode register mn (SMRmn).
- Serial communication runs set register mn (SCRmn).
- Serial data register mn (SDRmn).
- Serial flag clears the trigger register mn (SIRmn).
- Serial status register mn (SSRmn).
- Serial channel start register m(SSm).
- Serial channel stop register m(STm).
- Serial channel enable status register m(SEm).
- Serial output enable register m (SOEm).
- Serial output level register m(SOLm).
- Serial output register m(SOm).

19.3.2 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 operation instruction.

I can set the low 8 bits of the SPSm register with SPSmL and through the 8-bit memory operation instruction.

After generating a reset signal, the value of the SPSm register changes to "0000H".

Figure 19-5 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 ^{note} for the running clock (CKmk).
0	0	0	0	f_{CLK}
0	0	0	1	$f_{CLK}/2$
0	0	1	0	$f_{CLK}/2^2$
0	0	1	1	$f_{CLK}/2^3$
0	1	0	0	$f_{CLK}/2^4$
0	1	0	1	$f_{CLK}/2^5$
0	1	1	0	$f_{CLK}/2^6$
0	1	1	1	$f_{CLK}/2^7$
1	0	0	0	$f_{CLK}/2^8$
1	0	0	1	$f_{CLK}/2^9$
1	0	1	0	$f_{CLK}/2^{10}$
1	0	1	1	$f_{CLK}/2^{11}$
1	1	0	0	$f_{CLK}/2^{12}$
1	1	0	1	$f_{CLK}/2^{13}$
1	1	1	0	$f_{CLK}/2^{14}$
1	1	1	1	$f_{CLK}/2^{15}$

Note When you change the clock selected as f_{CLK} (change the value of the system clock control register (CKC)) during the operation of the Universal Serial Communication Unit (SCI), you must stop the operation of the SCI (serial channel stop register m). (STm)=000FH) after making changes.

Note that bit15~8 must be set to "0".

Note 1. f_{CLK} : Cpu/peripheral hardware clock frequency

2.m: Unit number (m=0~2).

3.k=0, 1

Figure 19-7 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	STSm n ^{note1}	0	SISmn 0 ^{note1}	1	0	0	MD mn2	MD mn1	MD mn0
-------	-----------	-----------	---	---	---	---	---	----------------------------	---	-----------------------------	---	---	---	-----------	-----------	-----------

SISmn0 ^{Note 4}	Channel n receives data in UART mode with level inversion control
0	Detect the falling edge as the starting position. 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 operating mode
0	0	SSPI mode
0	1	UART mode
1	0	Simple I ² C mode
1	1	Disable settings.

MDmn0	Channel n interrupt source selection
0	The end of transfer is interrupted
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, SMR03, SMR11, registers.

Note that bit13~9, 7, 4, 3 (SMR00, SMR02, MUST BE APPLIED TO BIT13,7,4,3.) The SMR10 register is bit13~6, 4, 3) set "0" and will bit5 to 1".

Remark m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)
q: UART number (q=0~3) r: IIC number (r=00, 01, 10, 11, 20, 21, 30, 31)

19.3.4 Serial communication runs the set 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).

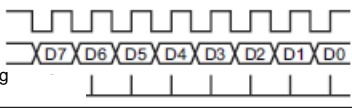
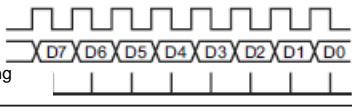
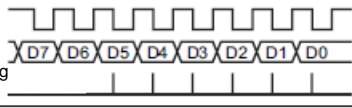
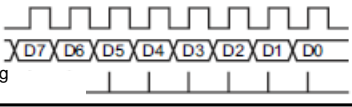
The SCRmn register is set by means of a 16-bit memory operation instruction.

After generating a reset signal, the value of the SCRmn register changes to "0087H".

Figure 19-7 Format of serial communication operation set 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	TXE mn	RXE mn	Dap mn	CKP mn	0	EOC mn Note 4	PTC mn1	PTC mn0	You mn	0	SLCm n1note1	SLC mn0	DLSm n3note3	DLSm n2note3	DLSm n1note2	DLS mn0

TXEmn	RXEmn	Setting of channel n operating 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.

EOCmn	Mask control of error interrupt signals (INTSREx (x=0~3)).
0	Disables the generation of error interrupts INTSREx (generate INTSRx).
1	Enable intSREx to be interrupted by errors (INTSRx is not generated when an error occurs).

In SSPI mode and simple I2C mode or when sending at UART, the EOCmn position must be placed in the "0" note 3.

Note 1 Limited to SCR00, SCR02, SCR10 registers.

2. Limited to SCR00 registers and SCR01 registers, the other fixed as "1".

3. Limited to SCR20, SCR21, SCR30, SCR31, other fixed as "01".

4. When the EOCmn bit is "0" and SSPImn is not used, it is possible to produce an error interrupt INTSREn.

Note that bit3, 6, and 11 must be set to "0" (SCR01, SCR03, SCR11 must also be set , bit5 of the SCR21 register is set to

"0"), and bit2 is set to "1" .

Remark m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)
q: UART number (q=0~3) r: IIC number (r=00, 01, 10, 11, 20, 21, 30, 31)

Figure 19-8 Format of serial communication operation set register mn (SCRmn) (2/3).

After reset: 0087H

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXE mn	RXE mn	Dap mn	CKP mn	0	EOC mn	PTC mn1	PTC mn0	You mn	0	SLCm n1 ^{note1}	SLC mn0	DLSm N3 ^{note3}	DLSm N2 ^{note3}	DISm n1 ^{note2}	DLS mn0

PTCmn1	PTCmn0	The setting of the parity bit in UART mode	
		Send	reception
0	0	Parity bits are not output.	There is no parity at the time of
0	1	Output parity ^{note 4} .	Parity is not judged.
1	0	Output parity.	Judgment parity.
1	1	Output odd check.	Judgment oddity check.

In SSPI mode and simple I²C mode, both PTCmn1 bits and PTCmn0 bits must be set to "0".

DIRmn	Selection of data transfer sequences in SSPI and UART modes
0	Perform MSB-first input/output.
1	Perform LSB-first input/output.

In simple I²C mode, the DIRmn must be positioned "0".

SLCmn1 ^{Note 1}	SLCmn0	The 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, 02, 10, 20).
1	1	Disable settings.

If an end-of-transmit interrupt is selected, an interrupt occurs after all stop bits have been transmitted.
At UART reception or in simple I²C mode, it must be set to 1 stop bit (SLCmn1, SLCmn0=0, 1))
In SSPI mode, it must be set to no stop bit (SLCmn1, SLCmn0=0, 0)
When the UART is transmitted, it must be set to 1 bit (SLCmn1, SLCmn0=0, 1) or 2 bits (SLCmn1). , SLCmn0=1, 0).

Note 1 Limited to SCR00, SCR02, SCR10 registers.

- Limited to SCR00 registers and SCR01 registers, the other fixed as "1".
- Limited to SCR20, SCR21, SCR30, SCR31, other fixed as "01".
- Regardless of the content of the data, always append a "0".

Note that bit3, 6, and 11 must be set to "0" (SCR01, SCR03, SCR11 must also be set Bit5 of the register is set to "0"), and bit2 is set to "1".

Remark m: unit number (m=0~2) n: channel number (n~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)

q: UART number (q=0~3) r: IIC number (r=00, 01, 10, 11, 20, 21, 30, 31)

Figure 19-8 Serial communication runs the format of the set 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	EOC mn	PTC mn1	PTC mn0	You mn	0	SLCm n1note1	SLC mn0	Dls mn3 Note 3	Dls mn2 Note 2	DLSm n1note2	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).	○	○	×
other				Prohibit settings.			

In the simple I²C mode, DLSmn3~ DLSmn0=0111B must be set.

Note 1 Limited to SCR00, SCR10, SCR20 registers only.

2. Limited to SCR00 registers and SCR01 registers, the other fixed as "1".

3. Limited to SCR20, SCR21, SCR30, SCR31, other fixed as "01".

Note that bit3, 6, and 11 must be set to "0" (SCR01, SCR03, SCR11 must also be set , bit5 of the SCR21 register is set to "0"), and bit2 is set to "1" 。

Remark m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)

q: UART number (q=0~3) r: IIC number (r=00, 01, 10, 11, 20, 21, 30, 31)

19.3.5 Serial data register mn(SDRmn) (SCI0 i.e. m=0).

The SDR mn register is the data register (16-bit) that channel n sends and receives.

SDR00, bit8 to 0 of SDR01 (9 bits lower) or SDR02, SDR03 Bit7 to 0 (low 8 bits) is used as a transmit and receive buffer register, bit15 to 9 (high 7 bits) is used as a crossover setting register for the operating clock (f_{MCK}).

If the CCSmn position of the serial mode register mn (SMRmn) is "0", the bit15 to 9 of the SDRmn register is used. The divider clock of the operating clock (7 bits high) is used as the transmission clock.

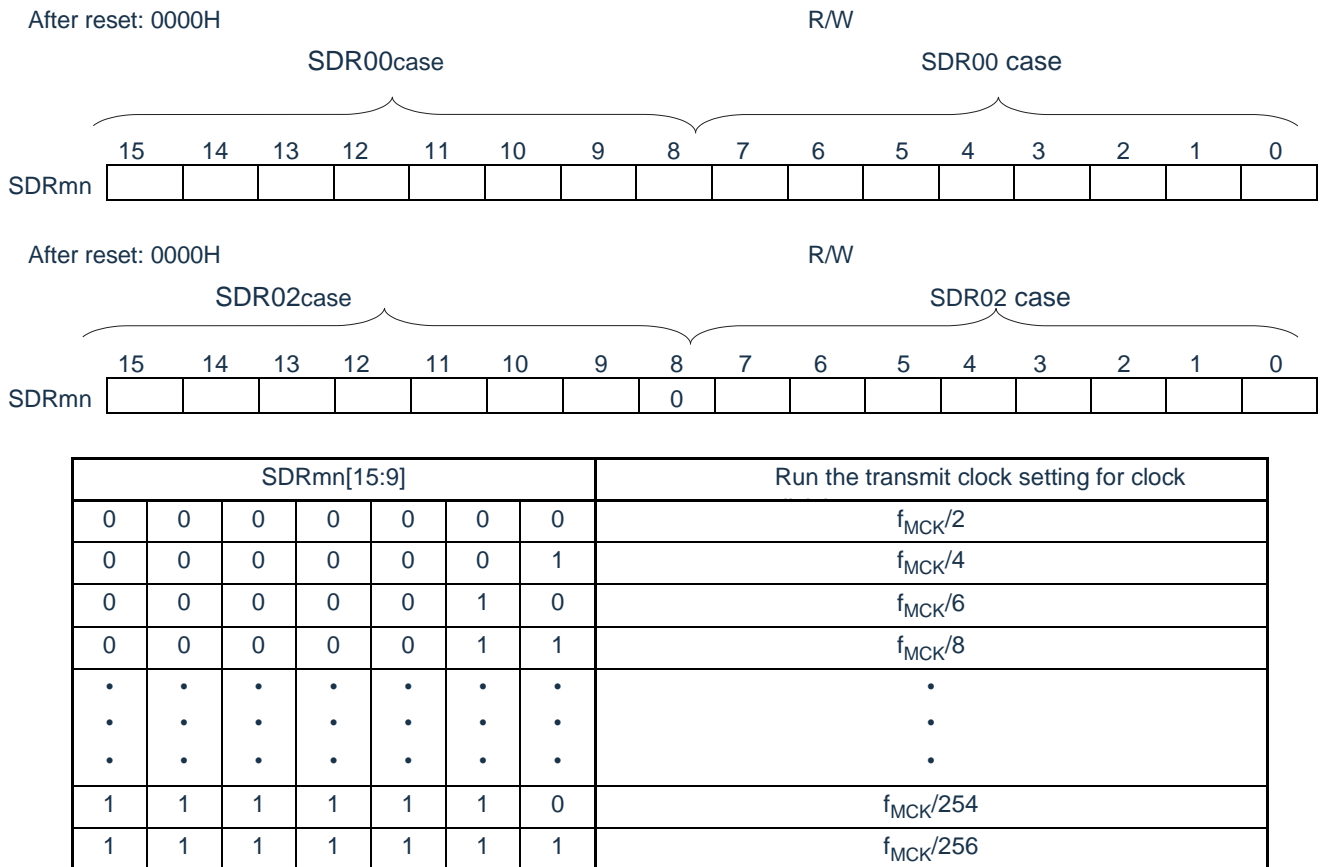
If the CCSmn position is "1", the bit15 to 9 (height 7) of SDR00 and SDR01 must be placed bit) to "0000000B". The input clock of the SCLKp pin, f_{SCLK} (slave transmission in SSPI mode), is the transmit clock.

The low 8 bits or 9 bits low of the SDRmn registers are used as transmit and receive buffer registers. When receiving data, save the parallel data converted by the shift register to 8 bits low or 9 bits low; When sending data, the transmit data transmitted to the shift register is set to 8 bits low or 9 bits low.

SDRmn registers can be read and written in 16-bit increments. However, the high 7 bits can only be read and written when the run stops (SEmn=0). In operation (SEmn=1) only the low 8 bits or 9 bits lower of the SDRmn register can be written, and the high of the SDRmn register is 7. The read value of the bit is always "0".

After generating the reset signal, the value of the SDRmn register changes to "0000H".

Figure 19-8 serial data register mn (SDRmn).



Note 1 The bit8 of the SDR02 and SDR03 registers must be set to "0".

2. When using UART, it is forbidden to set SDRmn [15:9] to "0000000B" and "0000001B".

3. When using Simple I2C, it is forbidden to set SDRmn [15:9] to "0000000B", and the setting value of SDRmn [15:9] must be greater than or equal "0000001B".

4. When the operation stops (SEmn=0), it is forbidden to rewrite SDRmn [7:0] through the 8-bit memory operation instruction (otherwise, SDRmn [15:9] is all cleared "0").

Note 1 For the function of the SDRmn register low 8 bit or low 9 bit, refer to "19.2 19.2The structure of a universal serial communication unit.

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

19.3.6 Serial data register mn (SDRmn) (SCI1/SCI2 i.e. m=1/2).

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_{MCK}). During operation (SEmn=1) bit15~9 is used as a transmit and receive buffer register.

If the CCSmn position of the serial mode register mn (SMRmn) is "0", the bit15 to 9 of the SDRmn register is used. The divider clock of the operating clock (7 bits high) is used as the transmission clock.

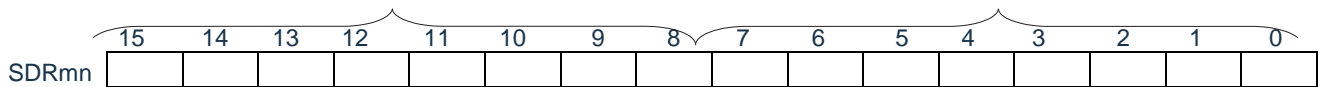
The SIRmn register is set by means of a 16-bit memory operation instruction.

After generating the reset signal, the value of the SDRmn register changes to "0000H".

Figure 19-9 serial data register mn (SDRmn).

After reset: 0000H

R/W



SDRmn[15:9]							Run the transmit clock setting for clock
0	0	0	0	0	0	0	f_{MCK}
0	0	0	0	0	0	1	$f_{MCK}/2$
0	0	0	0	0	1	0	$f_{MCK}/3$
0	0	0	0	0	1	1	$f_{MCK}/4$
•	•	•	•	•	•	•	•
1	1	1	1	1	1	0	$f_{MCK}/127$
1	1	1	1	1	1	1	$f_{MCK}/128$

Note 1 When the operation stops (SEmn=0), bit8~0 must be cleared to zero.

2. When using UART, it is forbidden to set SDRmn [15:9] to "0000000B" and "0000001B".

3. When using Simple I²C, setting SDRmn[15:9] to "0000000B" is prohibited and the setting value of SDRmn[15:9] must be greater than or equal "0000001B".

4. At run stop (SEmn=0), it is forbidden to rewrite SDRmn [7:0] via the 8-bit memory operation instruction (otherwise, SDRmn[15:9] is all cleared "0").

Note 1. The function of the SDRmn register during operation, please refer to "19.2The structure of a universal serial communication unit".

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

19.3.7 The serial flag clears the trigger register mn (SIRmn).

This is the trigger register used to clear each error flag for channel n.

If you set "1" for each of you (FECTmn, PECTmn, OVCTmn), the corresponding bit of the serial status register mn (SSRmn) (FEFmn, PEFmn, OV Fmn) clear "0". Because the SDIRmn register is a trigger register, if the corresponding bit of the SSRmn register is cleared, the SDIRmn register is also cleared immediately.

The SIRmn register is set by means of a 16-bit memory operation instruction.

I can set the lower 8 bits of the SIRmn register with the SIRmnL and through the 8-bit memory operation instruction.

After generating a reset signal, the value of the SIRmn register changes to "0000H".

Figure 19-10 serial flag clears the format of the 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	0	0	FECTmn ^{note1}	PEC Spinning	OVC Tmn

FECTmn	Clearing of the channel n-frame error flag is triggered																
Note 1																	
0	Do not clear.																
1	Clear the FEFmn bit of the SSRmn register to "0".																

PECTmn	Channel n parity error flag clear triggered																
0	Do not clear.																
1	Clear the PEFmn bit of the SSRmn register to "0".																

OVCTmn	The clearing of the channel n overflow error flag is triggered																
0	Do not clear.																
1	Clear the OV Fmn bit of the SSRmn register to "0".																

Note1: Limited to SIR01, SIR03, SIR11, SIR21 registers.

Note: that bits 15 to 3 (SIR00, SIR02, SIR10, SIR20) must be changed the register is bit15~2) set "0".

Note 1.m: Unit number (m=0~2) n: Channel number (n=0~3).

2. The read value of the SIRmn register is always "0000H".

Figure 19-11 Format of serial status register mn (SSRmn) (2/2).

After reset: 0000H

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	0	TSF mn	BFF mn	0	0	FEF mn ^{note1}	PEF mn	OVF mn

FEFmn ^{Note 1}	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 ² C was sent).
[Clear Condition]. • When the PECTmn bit of the SIRmn register is written "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 ² C is sent and when the ACK receives the timing slave does not return an ACK signal (ACK not detected).	

OVFmn	Channel n overflow error detection flag
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, SSR03, SSR11, SSR21 registers.

Note 1 If you write the SDRmn register with the BFFmn bit "1", the saved send or receive data is destroyed and an overflow error is detected (OVEmn=1).

Note m: Unit number (m=0~2) n: Channel number (n=0~3).

19.3.9 Serial channel start register m(SSm).

The SSm register is a trigger register that sets the allowed communication/start count of each channel.

If you write "1" to each of you (SSmn), set the corresponding bit (SEmn) of the serial channel allowed status register m(SEm) to "1" (run). Allow status). Because the SSmn bit is the trigger bit, the SSmn bit is cleared immediately if the SEmn bit is "1".

The SSm register is set via the 16-bit memory operation instruction.

I can set the low 8 bits of the SSm register with the SSmL and through the 8-bit memory operation instruction.

After generating the reset signal, the value of the SSm register changes to "0000H".

Figure 19-12 Format of serial channel start register m(SSm).

After reset: 0000H															
R/W															
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1 0
SS0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SS03 SS02 SS01 SS00

After reset: 0000H															
R/W															
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1 0
SS1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SS11 SS10

After reset: 0000H															
R/W															
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1 0
SS2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SS21 SS20

SSmn	The trigger for channel n run start														
0	No triggering.														
1	Place the SEmn position "1" and transfer to the Communication Standby ^{note} .														

Note If you place the SSmn position "1" during communication, the communication is stopped and enters the standby state. At this point, the values of the control registers and shift registers, the SCLKmn pin and SDOmn pins, the FEFmn flag, the PEFmn flag, and the OVFn flag remain in state.

Note 1 Bit15~4 of SS0 registers and bit15~2 of SS1 registers must be set to "0" , the bit15~2 of the SS 2 register is set to "0".

2. When receiving UART, it must be spaced at least 4 fMCK after the RXEmn position of the SCRmn register "1" Clock, then set SSmn to "1".

Note 1.m: Unit number (m=0~2) n: Channel number (n=0~3).

2. The read value of the SSm register is always "0000H".

19.3.10 Serial channel stop register m(STm).

The STm register is a trigger register that sets the allowable communication/stop count for each channel.

If you write "1" to each of you (STmn), the corresponding bit (SEmn) of the serial channel allowed status register m(SEm) is cleared to "0" (run Stop State). Because the STmn bit is the trigger bit, the STmn bit is immediately cleared if the SEmn bit is "0".

The STm register is set by means of a 16-bit memory operation instruction.

I can set the lower 8 bits of the STm register with STmL and through the 8-bit memory operation instruction.

After generating the reset signal, the value of the STm register changes to "0000H".

Figure 19-13 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	
ST0	0	0	0	0	0	0	0	0	0	0	0	0	0	ST03	ST02	ST01	ST00
After reset: 0000H																	
R/W																	
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
ST1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ST11	ST10
After reset: 0000H																	
R/W																	
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
ST2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ST21	ST20
STmn	Channel n runs to stop triggering																
0	No triggering.																
1	Clear the SEmn bit to "0" and stop the communication run ^{note} .																

Note the values of the control registers and shift registers, the SCLKmn pin and SDOmn pins, and the FEFmn, PEFmn, and OVFMn flags to hold state.

Note that the bit15~4 of st0 registers and bit15~2 of ST1 registers must be set to "0" , BIT15~2 of st 2 registers is set to "0".

Note 1.m: Unit number (m=0~2) n: Channel number (n=0~3).

2. The read value of the STm register is always "0000H".

19.3.11 Serial channel enable status register m (SEm).

SEm registers are used to confirm the allow or stop status of serial transmits and receives for each channel.

If you write "1" to each of you who allow register m (SSm) to start serially, place it at "1". If you write "1" to each of the serial channel stop register m(STm), it corresponds to bit "0".

For channel n that enable operation, the value of the CKOm_n bit (the serial clock output of channel n) of the serial output register m (SOM) described below cannot be rewritten by software, and the value reflected by the communication operation is output from the serial clock pin.

For a decommissioned channel n, the value of the CKOm_n bit of the SOM register can be set by software and output from the serial clock pin. Thus, arbitrary waveforms such as start conditions or stop conditions can be generated by software.

Read the SEm registers via 16-bit memory manipulation instructions.

It can read the lower 8 bits of the SEm register with SEmL and through the 8-bit memory operation instruction.

After generating a reset signal, the value of the SEm register changes to "0000H".

Figure 19-14 the 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	
SE0	0	0	0	0	0	0	0	0	0	0	0	0	0	SE03	SE02	SE01	SE00

After reset: 0000H															R	
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SE1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

After reset: 0000H																R	
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Hersef	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Sign	Indication of the allowed or stopped state of channel n running
0	Run stopped state
1	Run allowed status

Note m: Unit number (m=0~2) n: Channel number (n=0~3).

19.3.13 Serial output register m(SOm).

SOm registers are buffer registers for the serial output of each channel.

Outputs the value of the SOmn bit of this register from the serial data output pin of channel n.

Outputs the value of the CKOm_n bit of this register from the serial clock output pin of channel n.

The SOmn bit of this register can only be rewritten by software when serial output is disabled (SOEmn=0). When serial output (SOEmn=1) is allowed, the value of the SOmn bit of this register can only be changed by serial communication by software overriding.

The CKOm_n bit of this register can only be rewritten by software when the channel is stopped (SEmn=0). When allowing the channel to run (SEmn=1), the value of the CKOm_n bit of this register can only be changed by serial communication by software overrides.

To use the pins of a serial interface as a non-serial interface function such as a port function, the corresponding CKOm_n bit and SOmn position "1" must be placed.

The SOm registers are set via 16-bit memory operation instructions.

After generating a reset signal, the value of the SOm register changes to "0F0FH".

Figure 19-16 the format of serial output register m(SOm).

After reset: 0F0FH

R/W

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

SO0	0	0	0	0	CKO 03	CKO 02	CKO 01	CKO 00	0	0	0	0	SO 03	SO 02	SO 01	SO 00
-----	---	---	---	---	-----------	-----------	-----------	-----------	---	---	---	---	----------	----------	----------	----------

After reset: 0303HR/W

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

SO1	0	0	0	0	0	0	CKO 11	CKO 10	0	0	0	0	0	0	SO 11	SO 10
-----	---	---	---	---	---	---	-----------	-----------	---	---	---	---	---	---	----------	----------

After reset: 0303HR/W

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

SO2	0	0	0	0	0	0	CKO 21	CKO 20	0	0	0	0	0	0	SO 21	SO 20
-----	---	---	---	---	---	---	-----------	-----------	---	---	---	---	---	---	----------	----------

CKO mn	Serial clock output for channel n														
0	The output value of the serial clock is "0".														
1	The output value of the serial clock is "1".														

SO mn	Serial data output for channel n														
0	The output value of the serial data is "0".														
1	The output value of the serial data is "1".														

Note that the SO0 registers must be set to "0" for bit15~12 and bit7~4.

The BIT15~10 and bit7~2 of the SO1 register must be placed with "0".

The SO2 registers must be set to bit15 to 10 and bit7 to 2 to "0".

Note m: Unit number (m=0~2) n: Channel number (n=0~3).

19.3.14 Serial output level register m(SOLm).

The SOLm register is a register that sets the data output level inversion of each channel.

This register can only be set in UART mode. In SSPI mode and simple I²C mode, the corresponding position "0" must be placed. Only when serial output is allowed (SOEmn=1), the inverting setting of each channel n of this register is reflected to the pin output. When serial output is disabled (SOEmn=0), the value of the SOMn bit is output directly. It is forbidden to overwrite the SOLm register during operation (SEmn=1).

The SOLm register is set by means of a 16-bit memory operation instruction.

I can set the low 8 bits of the SDOLm register with SOLmL and through the 8-bit memory operation instruction.

After generating a reset signal, the value of the SOLm register changes to "0000H".

Figure 19-17 format of 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	
SOL0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
															SOL 02	0	
																SOL 00	

After reset: 0000H																
R/W																
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOL1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

After reset: 0000H																
R/W																
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Ground	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SOL 20																

Ground mn	Channel n transmits data level in UART mode for selection of inverted phases															
0	Output communication data directly.															
1	Outputs communication data in reverse.															

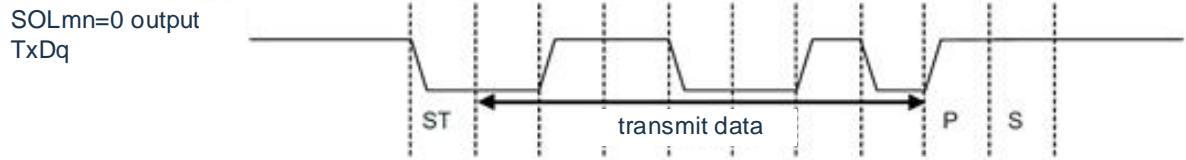
Note that the bit15 to 3 of the SOL0 register and bit1 and bit1 of the SOL1 register must be changed, bit15~1 of the SOL 2 register is set to "0".

Note m: Unit number (m=0~2) n: Channel number (n=0, 2).

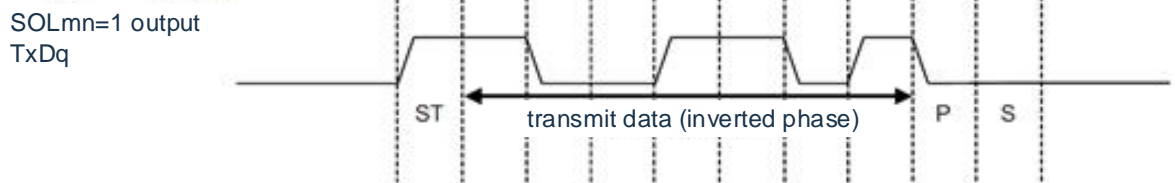
When performing a UART transmission, an example of the level reversal of the transmitted data is shown in Figure 19-18.

Figure 19-18 Example of level inversion of the transmitted data

(a) Non-inverting output (SOLmn=0)



(b) phase inverting output (SOLmn=0)



Note m: Unit number (m=0~2) n: Channel number (n=0, 2).

19.3.15 Input Switch 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 SS1E00 bit controls the SS00 pin input for channel 0 in slave mode of SSPI00 communication. During the input of a high level to the SS00 pin, no transmission and reception is made even if the serial clock is input; During the period when the SS00 pin is input low, if a serial clock is input, it is sent and received according to the settings of each mode.

The ISC register is set by the 8-bit memory operation instruction.

After generating a reset signal, the value of the ISC register changes to "00H".

Figure 19-19 Format of input switching control register (ISC).

After reset: 00H

R/W

Symbol

ISC	SYESE00	0	0	0	0	0	ISC1	ISC0
-----	---------	---	---	---	---	---	------	------

SSDIE00	SSPI00 communication in slave mode settings for channel 0's SS00 input
0	The SS00 pin input is invalid.
1	The SS00 pin input is valid.

ISC1	Timer4 is switched to the input of channel 3
0	Use the input signal from the TI03 pin as the input to the timer (usually operating).
1	Use the input signal from the RxD0 pin as the input to the timer (detect the wake-up signal and measure the low width of the interval segment and the pulse width of the sync segment).

ISC0	Input switching of external interrupt (INTP0).
0	Use the input signal from the INTP0 pin as input to an external interrupt (usually operating).
1	Use the input signal from the RxD0 pin as the input to an external interrupt (detect wake-up signal).

Note that bit6~0 must be set to "0".

19.3.16 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 I²C communication, the corresponding position must be "0" to invalidate the noise filter. For pins used for UART communication, the corresponding position "1" must be placed 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 operation instruction.

After generating a reset signal, the value of the NFEN0 register changes to "00H".

Figure 19-20 the format of noise filter enable register 0 (NFEN0).

After reset: 00H	R/W							
Symbol	7	6	5	4	3	2	1	0
NFEN0	0	SNFEN20	0	SNFEN20	0	SNFEN10	0	SNFEN00

SNFEN30	RxD 3-pin noise filter is used or not
0	Noise filter OFF
1	Noise filter ON
When used as the RxD3 pin, SNFEN3 0 must be placed at position "1". When used as a function other than the RxD3 pin, SNFEN30 must be placed at position "0".	

SNFEN20	The RxD2 pin is used or not for noise filters
0	Noise filter OFF
1	Noise filter ON
When used as the RxD2 pin, SNFEN20 must be positioned "1". When used as a function other than the RxD2 pin, the SNFEN20 must be placed at position "0".	

SNFEN10	The noise filter of the RxD1 pin is used or not
0	Noise filter OFF
1	Noise filter ON
When used as the RxD1 pin, SNFEN10 must be positioned "1". When used as a function other than the RxD1 pin, the SNFEN10 must be placed at "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 placed at position "1". When used as a function other than the RxD0 pin, the SNFEN00 must be placed at "0".	

Note that bit7, 5, 3, 1 must be set to "0".

19.3.17 Registers that control serial input/output pin port functions

When using a universal serial communication unit, control registers for port functions that are multiplexed with the object channel (port mode registers (PMxx), port registers (Pxx), and port mode control registers (PMCxx)) must be set. For details, please refer to "2.3.1 Port Mode Register (PMxx)", "2.3.2 Port Register (Pxx.) for details and "2.3.6 Port Mode Control Registers (PMCxx)".

The set port mode registers (PMxx), port registers (Pxx), and port mode control registers (PMCxx) vary by product. For details, please refer to "Register Settings when Using the Multiplexing Function in 2.5".

When using the multiplex port of the serial data output pin or serial clock output pin as the serial data output or serial clock output, the position of the port mode control register (PMCxx) and the position of the port mode register (PMxx) must be "0" and the position of the port register (Pxx) "1" must be used for each port. In addition, when used for N-channel open-drain output mode, the position of the port output mode register (POMxx) corresponding to each port must be "1".

(Example) P02 is used as a serial data output in case the port mode controls register 0 at PMC02 position "0". Place the PM02 position "0" of port mode register 0. Place the P02 position of port register 0 "1".

When using the multiplexing port of the serial data input pin or serial clock input pin as a serial data input or serial clock input, the position of the corresponding port mode register (PMxx) for each port must be "1" and the position of the port mode control register (PMCxx) "0". At this point, the bit of the port register (Pxx) can be "0" or "1".

In addition, when used as a TTL input buffer, the position of the port input mode register (PIMxx) corresponding to each port must be "1".

(Example) P03 is used as a serial data input in case the port mode controls the PMC03 position "0" of register 0. Place the PORT mode register 0 at PM03 position "1". Place the P03 position of port register 0 at "0" or "1".

19.4 Run stop mode

Each serial interface of a universal serial communication unit has a stop-run mode. Serial communication is not possible in run-stop mode, so power consumption is reduced. In addition, pins for the serial interface can be used as port functions in run-stop mode.

19.4.1 Case when the operation is stopped on a unit basis

The unit stop is set by peripheral allow register 0/2 (PER0/2).

Per0/2 registers are registers that are set to allow or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocks to unused hardware.

To stop universal serial communication unit 0, the bit2 (SCI0EN) of PER0 must be set to "0"; To stop Universal Serial Communication Unit 1, the bit3 (SCI1EN) of PER0 must be set to "0"; To stop Universal Serial Communication Unit 2, bit3 (SCI2EN) of PER2 must be placed at "0" .

Note 1 When the SCImEN bit is "0", the write operation of the control register of the universal serial communication unit m is ignored, and the read values are initial values. However, the following registers are excluded:

- Input Switch Control Register (ISC).
- Noise filter enable register 0 (NFEN0).
- Port Input Mode Register (PIM).
- Port Output Mode Register (POMx).
- Port Mode Register (PMx).
- Port register (Px).

19.4.2 Case of stop operation by channel

Stop operation by channel through each of the register settings below.

Figure 19-21 Setting of each register during channel based stop operation

(a) Serial channel stop register m(STm)... This is the register that sets the allowed communication/stop count for 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	0	0	STm3 ^{note}	STm2 ^{note}	STm1	STm0
																	0/1	0/1	0/1	0/1

1: Clear the SE_mn bit to "0" and stop the communication operation

※ Because the ST_mn bit is the trigger bit, if the SE_mn bit is "0", the ST_mn bit is cleared immediately.

(b) The serial channel enable status register m(SEm)... This register indicates the running or stopping state of data transmission and reception for each channel.

	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	0	0	SEm3 ^{note}	SEm2 ^{note}	SEm1	SEm0
																	0/1	0/1	0/1	0/1

0: Run stopped

※The SE_m register is a read-only status register that stops running through the ST_m register. For channels that have been stopped, the value of the CKO_mn bit of the SO_m register can be set by software.

(c) The serial output enable register m (SOEm)... This is the register that sets the output of serial communication that enable or stops 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	0	0	SOEm3 ^{note}	SOEm2 ^{note}	SOEm1	SOEm0
																	0/1	0/1	0/1	0/1

0: Stops the output by running through serial communication

※For channels that have stopped serial output, the value of the SO_mn bit of the SO_m register can be set by software.

(d) Serial output register m(SOm)... This is the buffer register for the serial output of each channel.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
SOm	0	0	0	0	0	CKO _m 3 ^{Note}	CKO _m 2 ^{Note}	CKO _m 1	CKO _m 0	0	0	0	0	0	0	0	SOm3 ^{note}	SOm2 ^{note}	SOm1	SOm0
						0/1	0/1	0/1	0/1								0/1	0/1	0/1	0/1

1: The output value of the serial clock is "1" 1: The output value of the serial data is "1"

※When the corresponding pin of each channel is used as a port function, the corresponding CKO_mn bit and SO_mn position "1" must be used.

Note is limited to Universal Serial Communication Unit 0.

Note 1.m: Unit number (m=0~2)n: Channel number (n=0~3).

2.  : Cannot be set (set initial value). 0/1: Set "0" or "1" according to the user's purpose.

19.5 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI20, SSPI20, SSPI20, SSPI21, SSPI30, SSPI31) communication

This is a clock synchronization communication function implemented by three lines of serial clock (SCLK) and serial data (SDI and SDO).

[Sending and receiving data].

- 7-bit or 8-bit data length (SCI0).
- Data length of 7 to 16 bits (SCI1/SCI2).
- Phase control of sending and receiving data
- MSB/LSB preferred

[Clock Control].

- Master or Slave selection
- Phase control of input/output clocks
- Set the transmission period generated by the prescaler and the internal counter of the channel.
- Maximum transfer rate ^{note}

Master communication: $\text{Max.f}_{\text{CLK}}/2$

Slave communication: $\text{Max.f}_{\text{MCK}}/6$

[Interrupt function].

- Transmit end interrupt, buffer empty interrupt

[Error Detection Flag].

- Overflow error

Note must be used within the range that satisfies the SCLK Cycle Time (t_{KCY}) characteristics. Please refer to the data sheet for details.

Channels 0 to 3 for SCI0, channels 0 to 1 for SCI1, and channels for SCI2 0~1 is supported for 3-wire serial I/O (SSPI00, SSPI01, SSPI01 SSPI10, SSPI11, SSPI20, SSPI21, SSPI30, SSPI31) channel.

3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI20, SSPI20, SSPI20, SSPI21, SSPI30, SSPI31) has the following 6 types of communication operation:

- Master send (see 19.5.1).
- Master receive (see 19.5.2).
- Master sending and receiving (cf. 19.5.3).
- Slave sending (cf19.5.4).
- Slave reception (cf19.5.5).
- Slave sending and receiving (cf19.5.6).

19.5.1 Master send

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	SSPI30	SSPI31
Object channels	Channel 0 of SCI0	Channel 1 of SCI0	Channel 2 of SCI0	Channel 3 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	SCLK30, SDO30	SCLK31, SDO31
interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21	INTSSPI30	INTSSPI31
Error detection flags	Selectable end-of-transmit interrupt (single-pass mode) or buffer-empty interrupt (continuous transfer mode).							
The length of the transferred data	SCI0: 7 or 8 bits SCI1/SCI2: 7~16bit							
Transfer Rate Note	Max. $f_{CLK}/2$ [Hz] Min. $f_{CLK}/(2^{15} \sim 128)$ [Hz] f_{CLK} : 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 SCRmn register. • CKPmn=0: Normal phase • CKPmn=1: Inverted							
Data direction	MSB priority or LSB priority							

Note It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

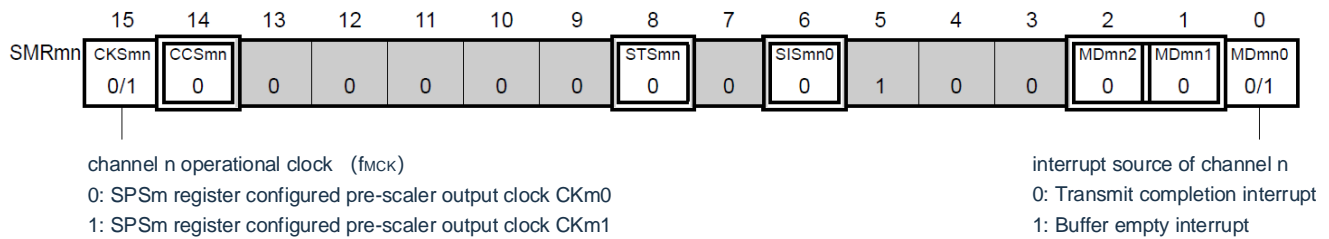
Note m: Unit number (m=0~2)n: Channel number (n=0~3)mn=00~ 03, 10~11, 20~21

(1) Register settings

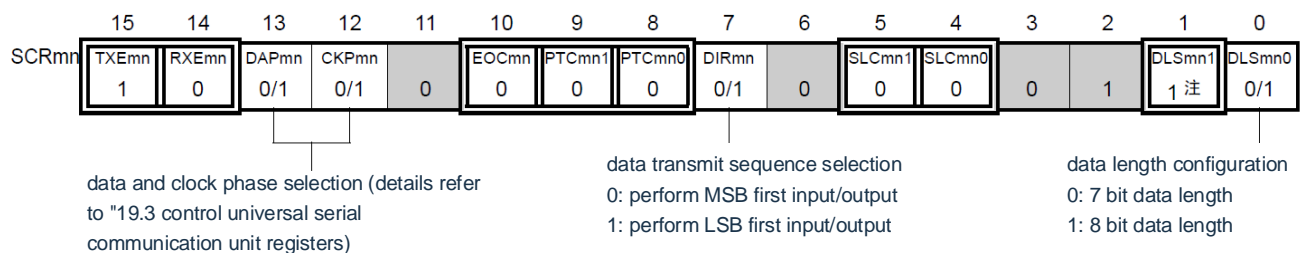
Fig19-22: 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21, SSPI30, SSPI31)

Example of register setting content when the master sends

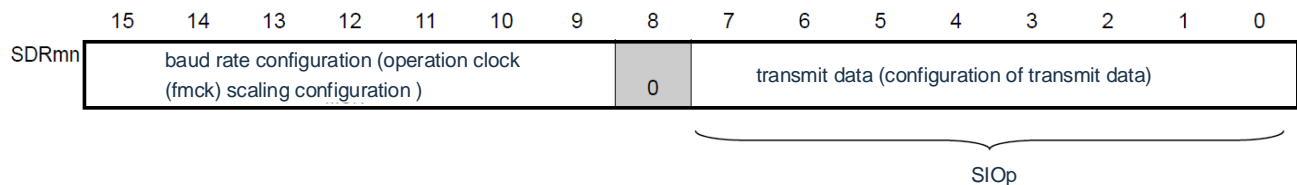
(a) serial mode register mn (SMRmn)



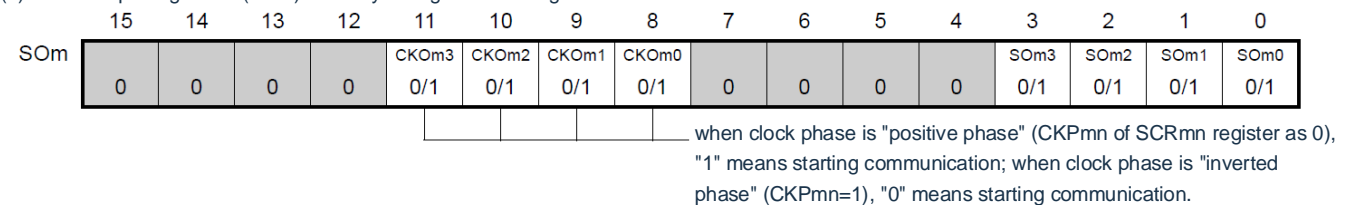
(b) serial communication operation configuration registermn mn(SCRmn)



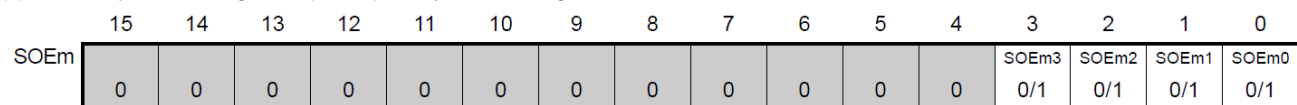
(c) serial data registermn mn(SDRmn)



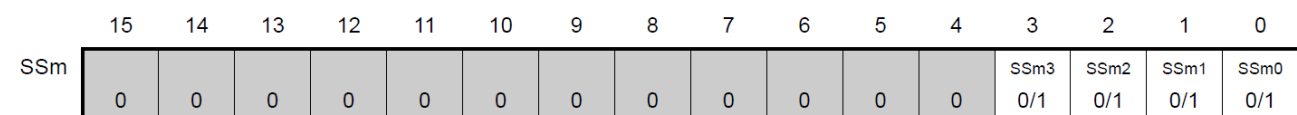
(d) serial output register m(SOm)Only configure bit of target channel



(e) serial output enable register (SOEm)....only set bit of target channel to 1.



(f) serial channel start registerm (SSm)....only set bit of target channel to 1.



Note This example is the setting method for SCI0. The data length of SCI1/SCI2 and the serial data register SDRmn are set differently from this example.

For data length settings, refer to Chapter 19.3.4Serial communication runs the set register mn (SCRmn).Chapter 17).

For the setting method of serial data register SDRmn, refer to "19.3.6Serial data register mn(SDRmn) (SCI1/SCI2 i. e. $m=1/2$).

Note 1.m: Unit number ($m=0\sim2$)n: Channel number ($n=0\sim3$)mn=00~ 03, 10~11, 20~21

2.  : Cannot be set (set initial value). 0/1: Set "0" or "1" according to the user's purpose.

(2) Procedure

Figure 19-23 The initial setup steps of the master transmission

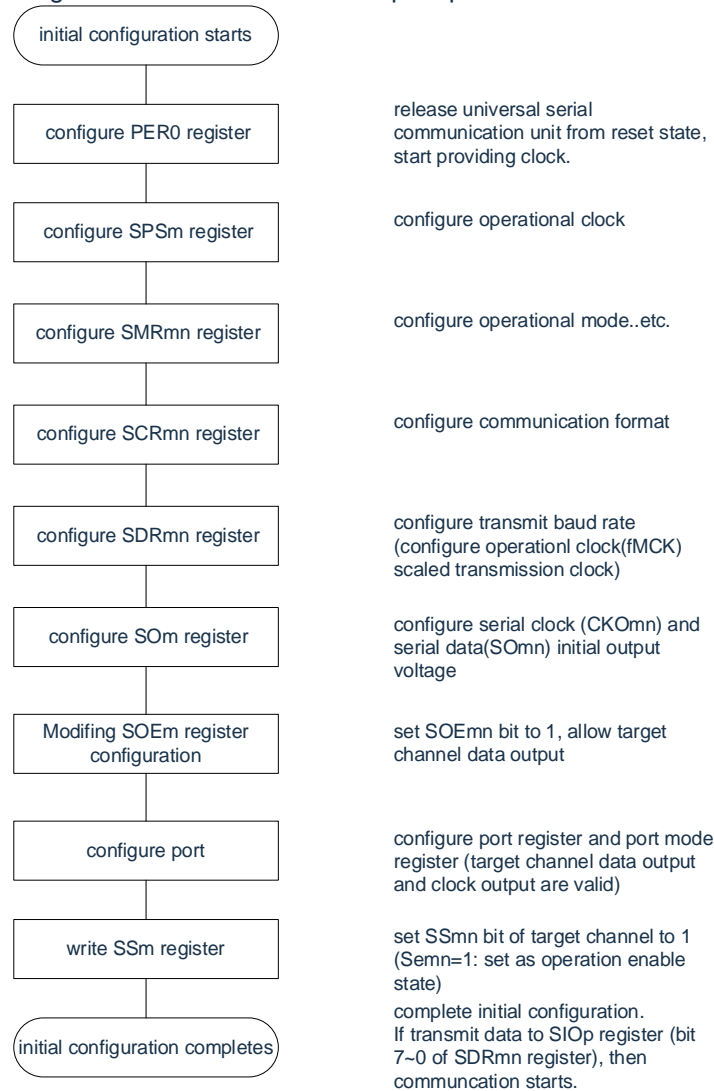


Figure 19-24 The abort step of the master transmission

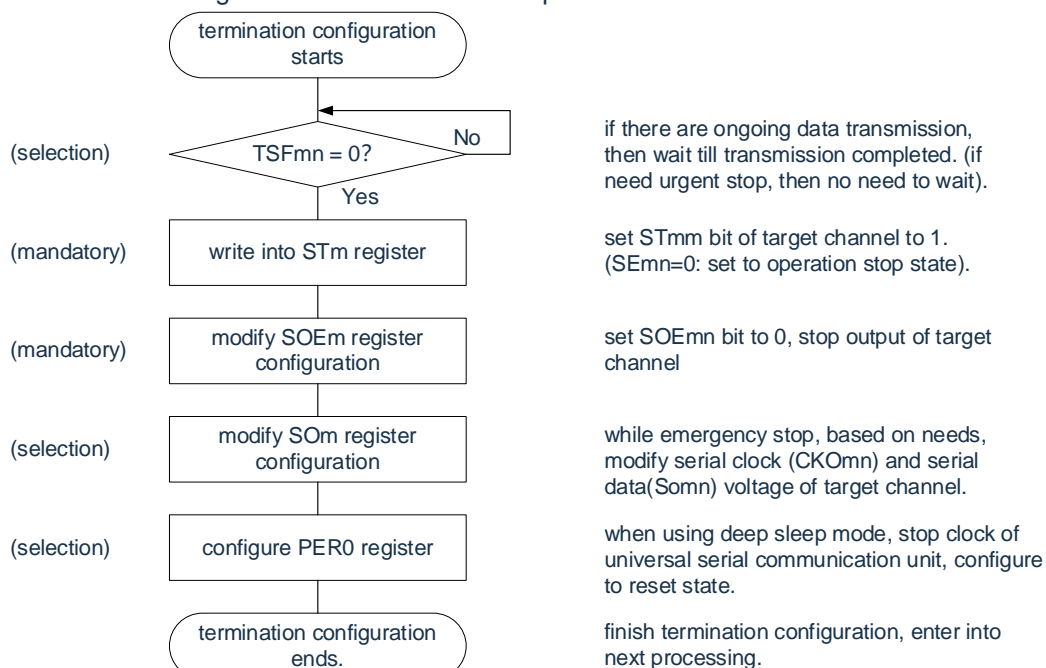
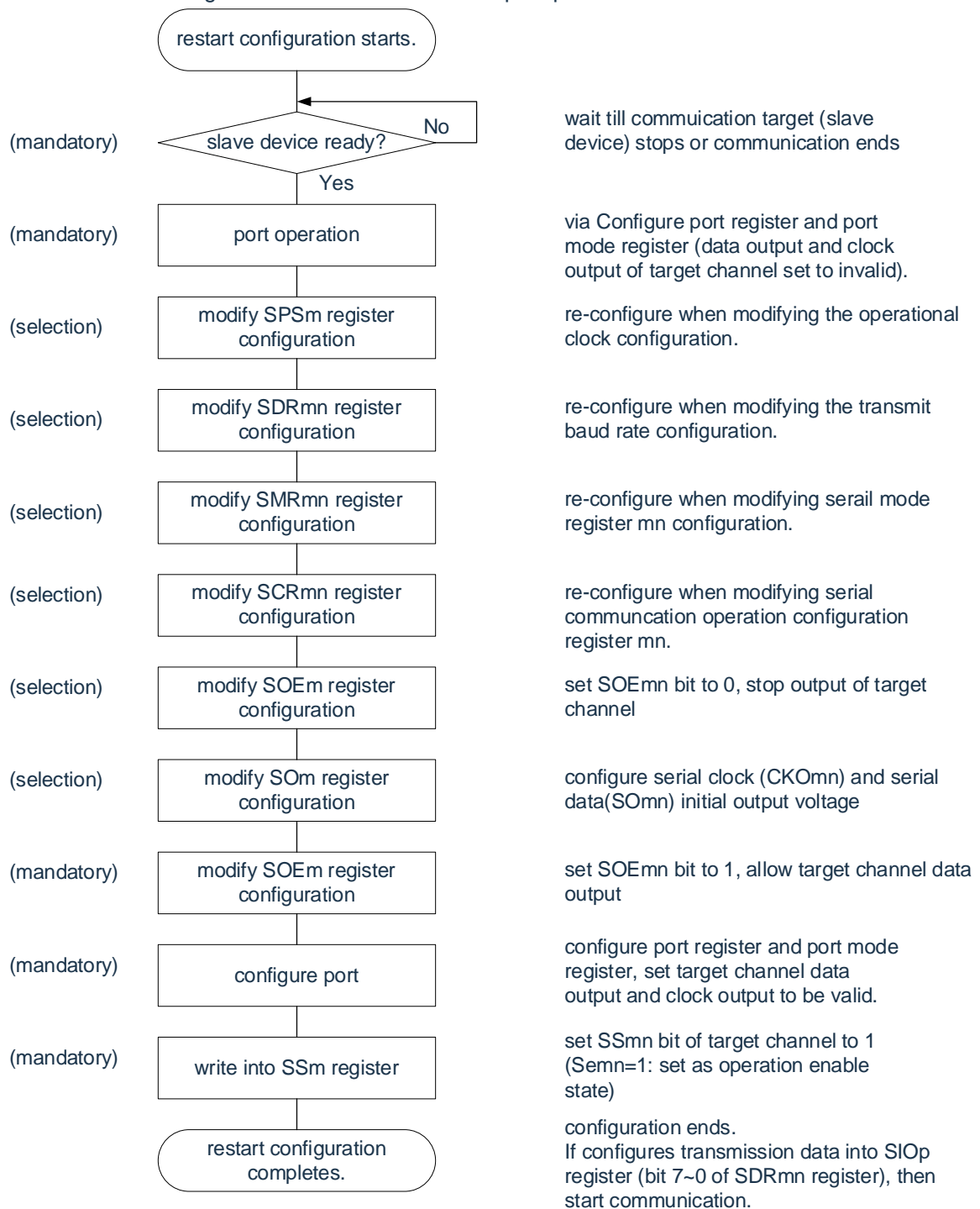


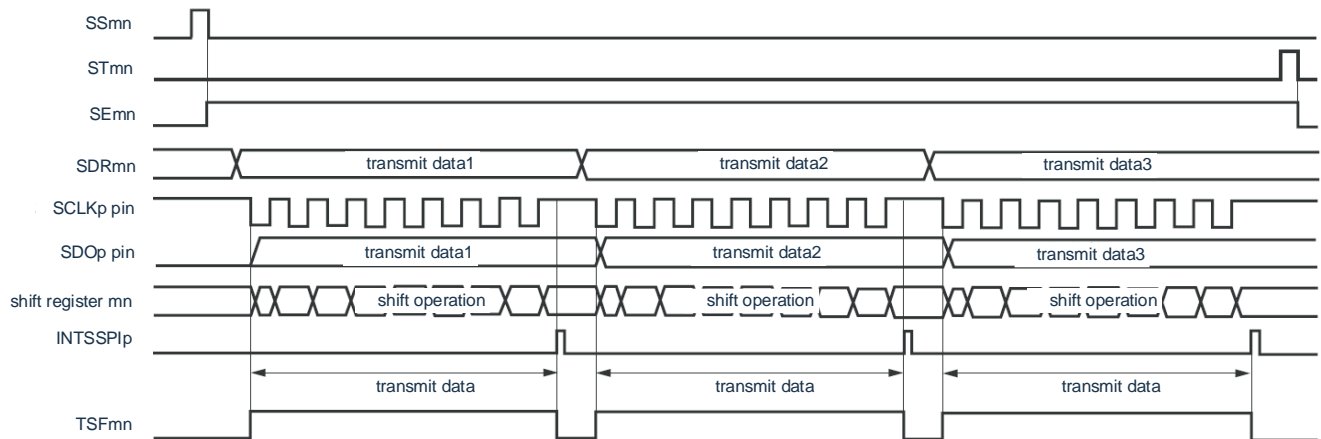
Figure 19-25 Restart the setup step of the master send



Note 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.

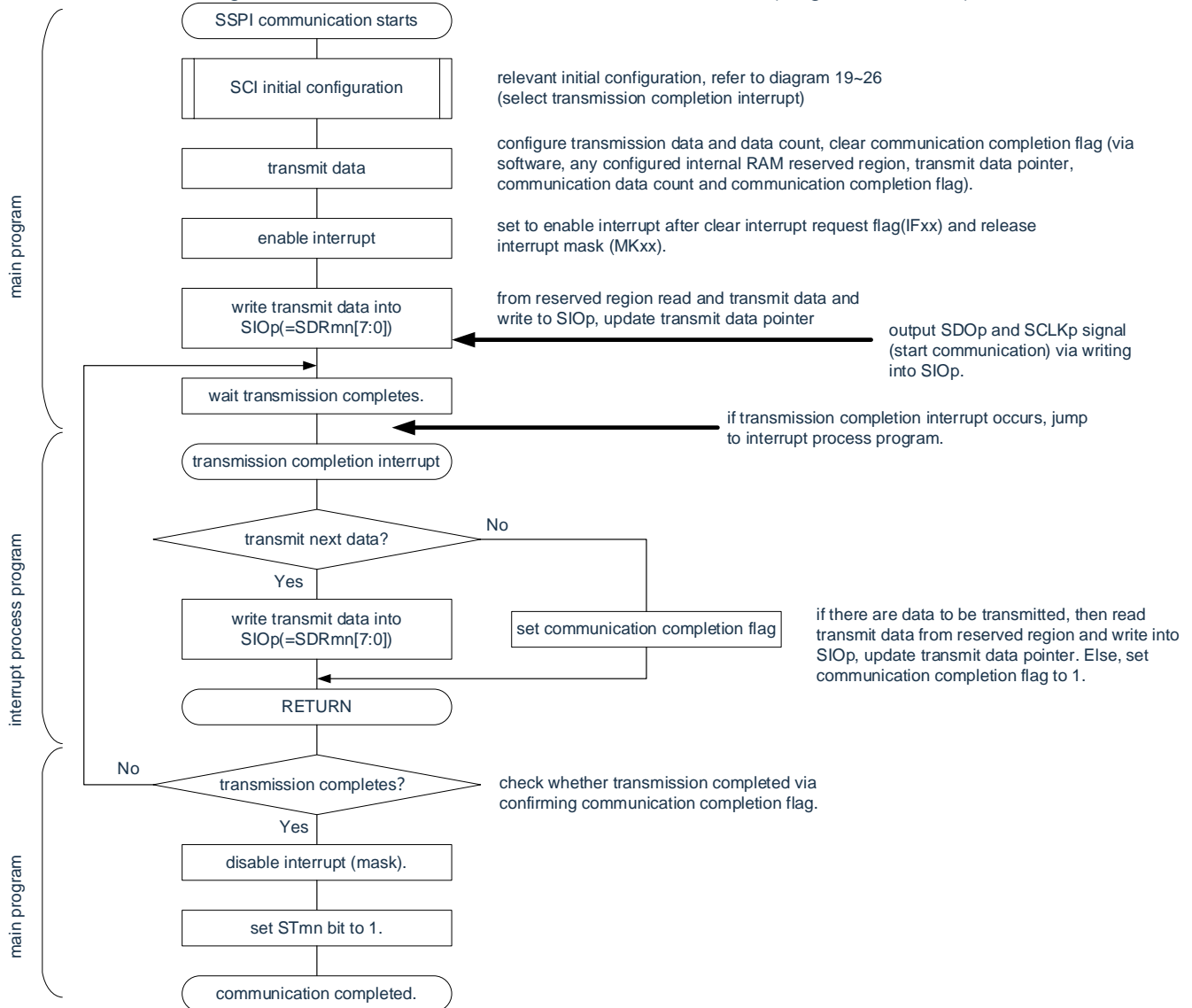
(3) Process flow (single send mode).

Figure 19-26 Timing diagram of the master transmit (single send mode) (type 1: DAPmn= 0, CKPmn = 0).



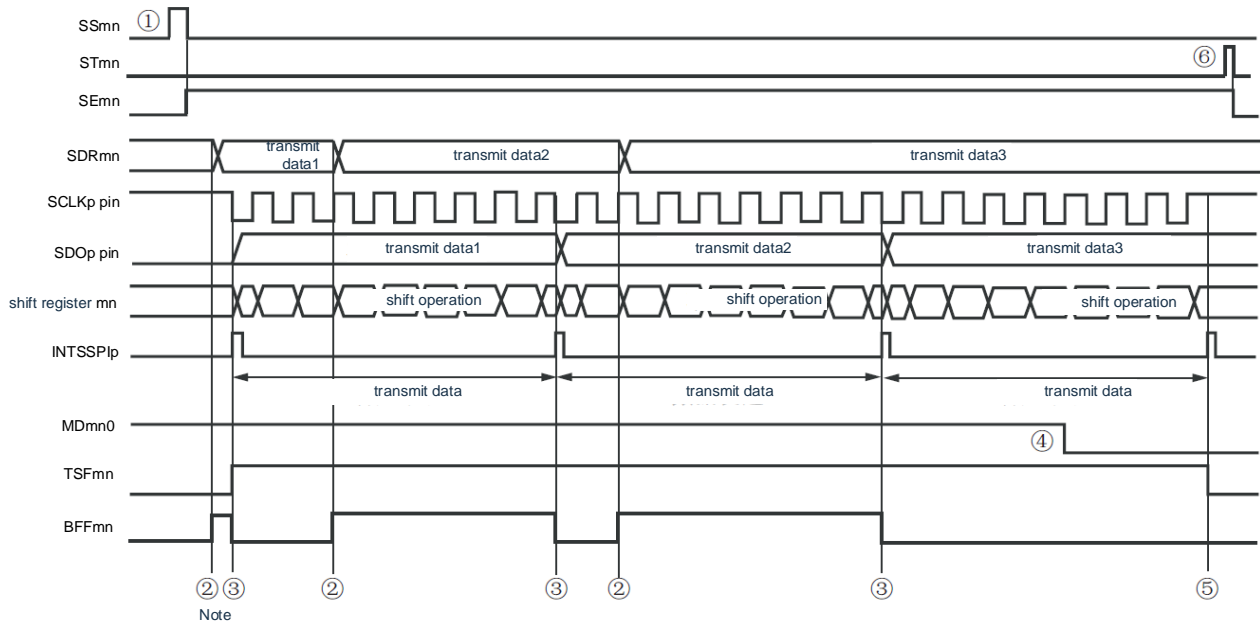
Remark m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)
mn=00~03, 10~11, 20~21

Figure 19-27 Flowchart of the master transmission (single send mode).



(4) Process flow (continuous send mode).

Figure 19-28 Timing diagram of the master transmit (continuous send mode) (type 1: DAPmn= 0, CKPmn = 0).

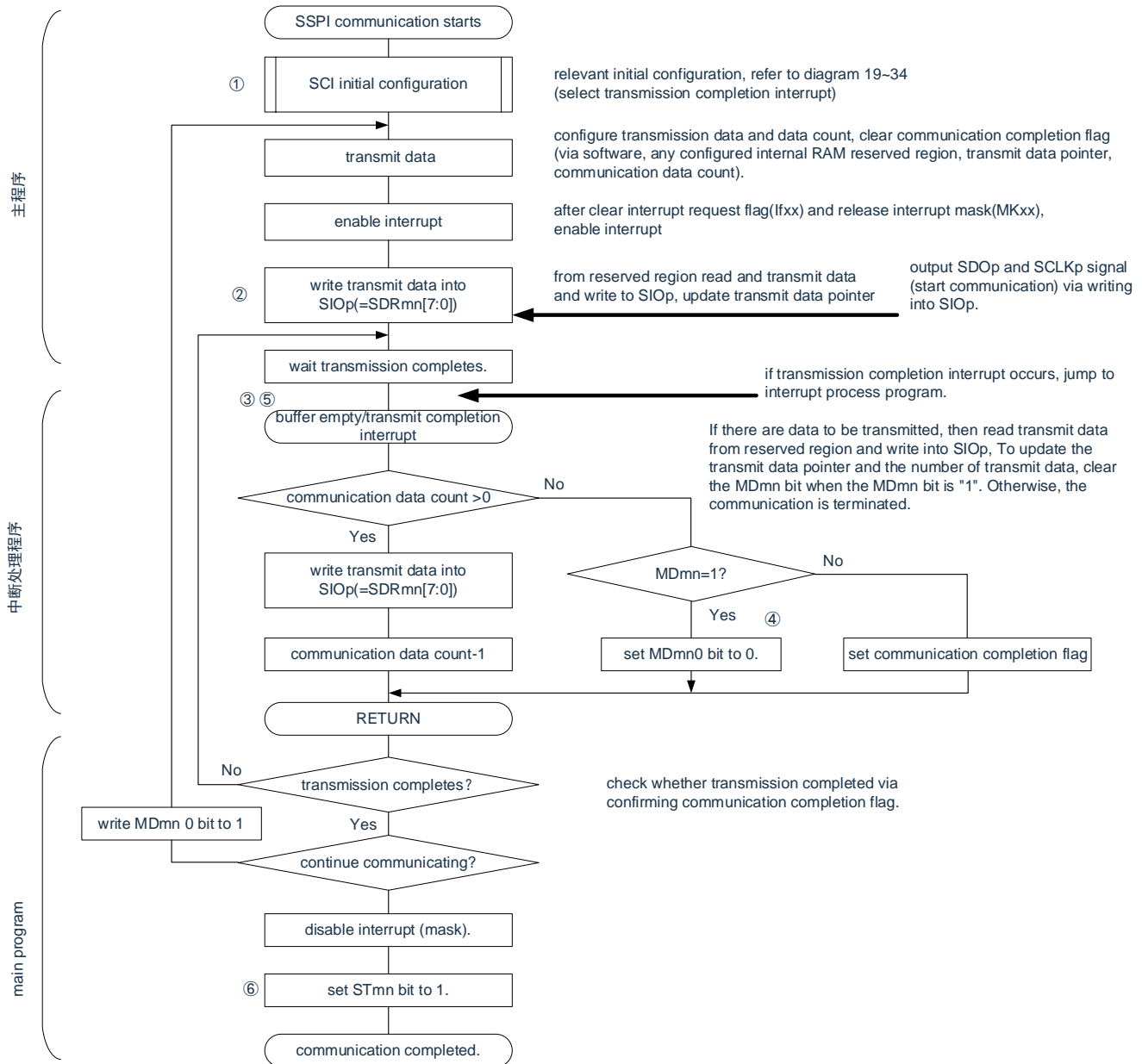


Note If the BFFmn bit of the serial status register mn (SSRmn) is "1" (when valid data is saved in the serial data register mn (SDRmn)) is given When the SDRmn register writes the transmit data, it rewrites the send data.

Note that the MDmn0 bit of the serial mode register mn (SMRmn) can be rewritten even in operation. However, in order to catch the end-of-transmission interruption of the last sent data, it must be rewritten before the last bit of transmission begins.

Remark m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)
mn=00~03, 10~11, 20~21

Figure 19-29 Flowchart of the master transmission (continuous transmission mode).



Note (1)~(6) in the figure corresponds to (1)~(6) in the "Timing Diagram of Figure 19-28 Master Transmission (Continuous Send Mode)".

19.5.2 Master receive

Master receiver 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 for SCI0	Channel 1 for SCI0	Channel 2 for SCI0	Channel 3 for SCI0	Channel 0 for SCI1	Channel 1 for SCI1
Pin Used	SCLK00, SDO00	SLK01, SDO01	SLK10, SDO10	SLK11, SDO11	SLK20, SDO20	SLK21, SDO21
interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21
Error detection flag	Interrupt at that end of the transfer may be selecte (single transfer mode) or buffer air-discontinuity (continuous transfer mode).					
length of transmit data	SCI0: 7 or 8 bits SCI1/SCI2: 7~16 bit					
Transfer Rate Note	Max.fCLK/2[Hz] Min.fCLK/(2×2 ¹⁵ ×128) [Hz] fCLK: system clock frequency					
data phase	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	Can be selected by the CKPmn bit of the SCRmn register. · CKPmn=0: prime CKPmn=1: inversion					
data orientation	MSB First or LSB First					

Note It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

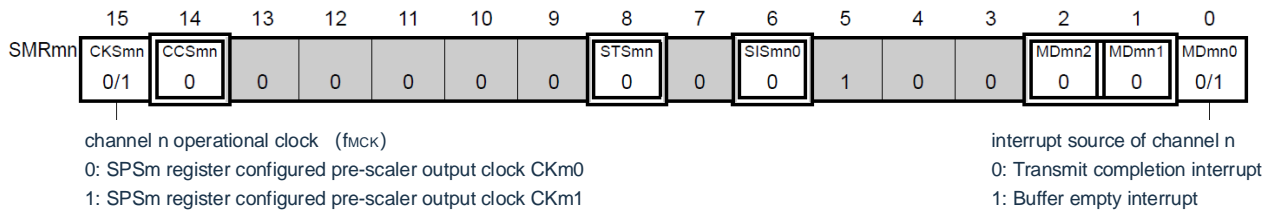
Remark m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)

mn=00~03, 10~11, 20~21

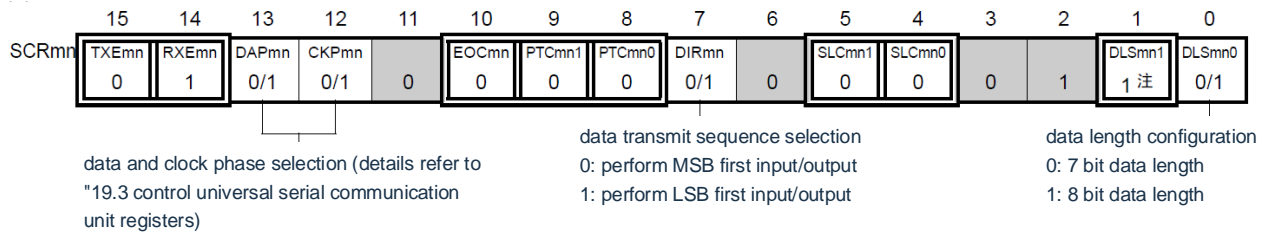
(1) Register settings

Fig19-30: 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21, SSPI30, SSPI31)
Example of register setting content when the master receives

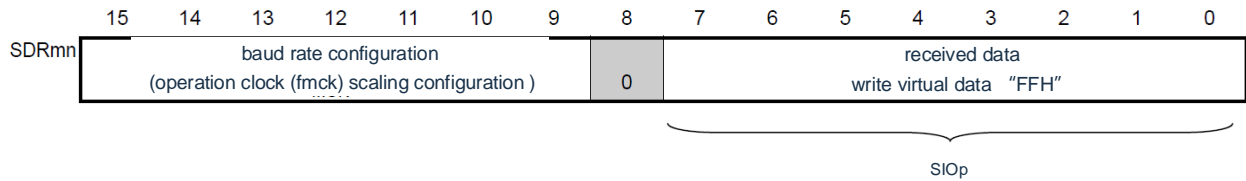
(a) serial mode register mn(SMRmn)



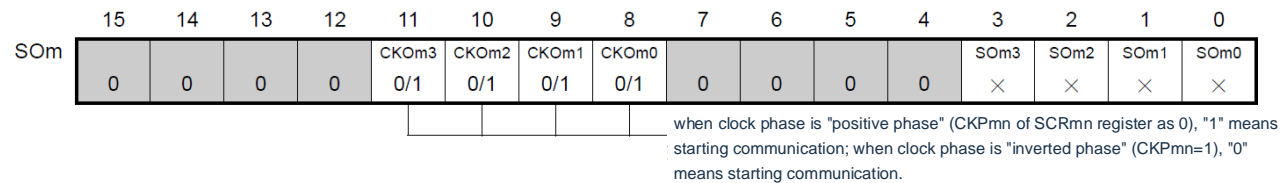
(b) serial communication operation configuration registermn mn(SCRmn)



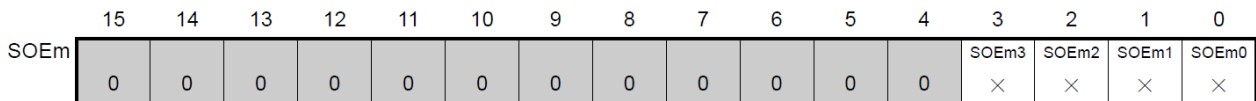
(c) serial data register mn(SDRmn) (low 8 bit: SIOp)



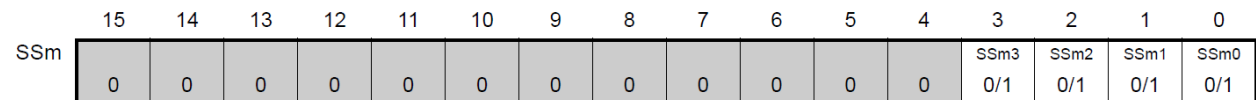
(d) serial output register (SOM)..... Only configure bit of target channel



(e) serial output register m(SOEm)Not used in this mode



(f) serial channel start register m (SSm) Only set bit of target channel to 1.



Note This example is the setting method for SCI0. The data length of SCI1/SCI2 and the serial data register SDRmn are set differently from this example.

For data length settings, refer to Chapter 19.3.4Serial communication runs the set register mn (SCRmn).Chapter 17).

For the setting method of serial data register SDRmn, refer to "19.3.6Serial data register mn(SDRmn) (SCI1/SCI2 i. e. m=1/2).

Remark1.m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)
mn=00~03, 10~11, 20~21

2. : Fixed setting in SSPI master receive mode. : Cannot be set (initial value is set).

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

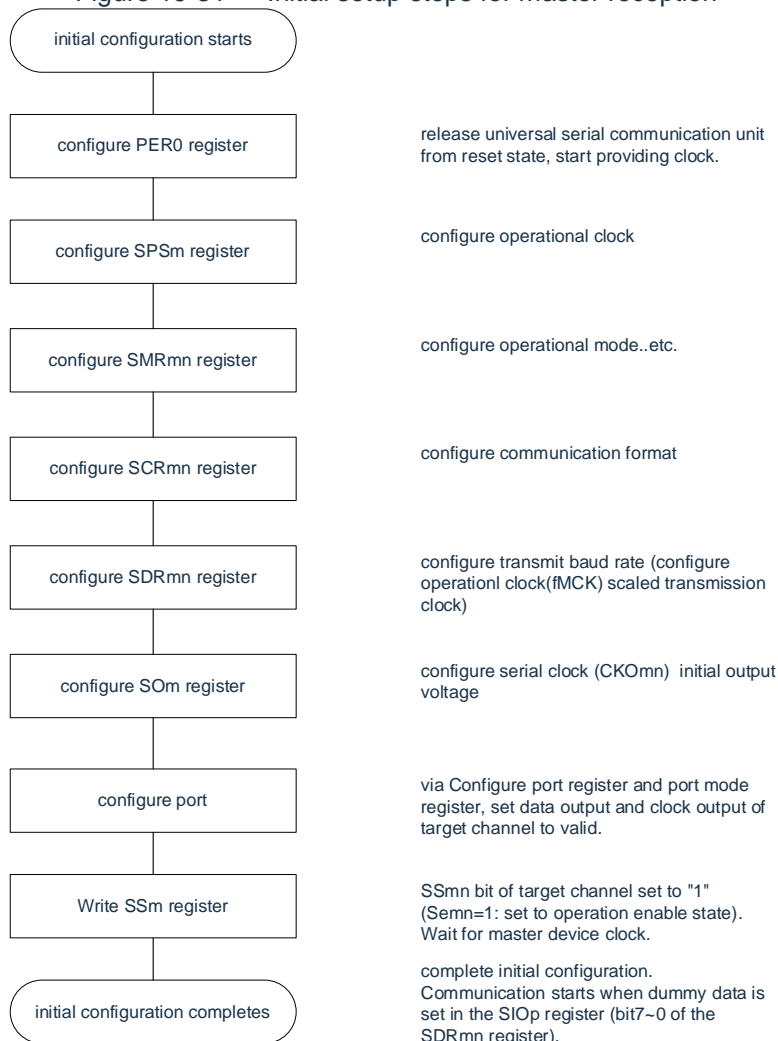
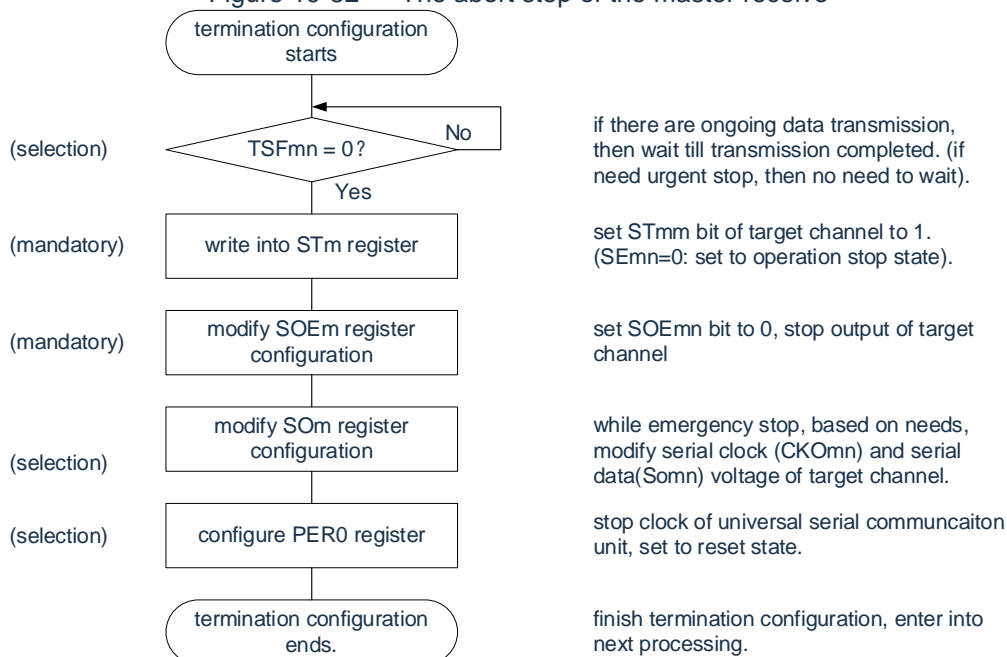
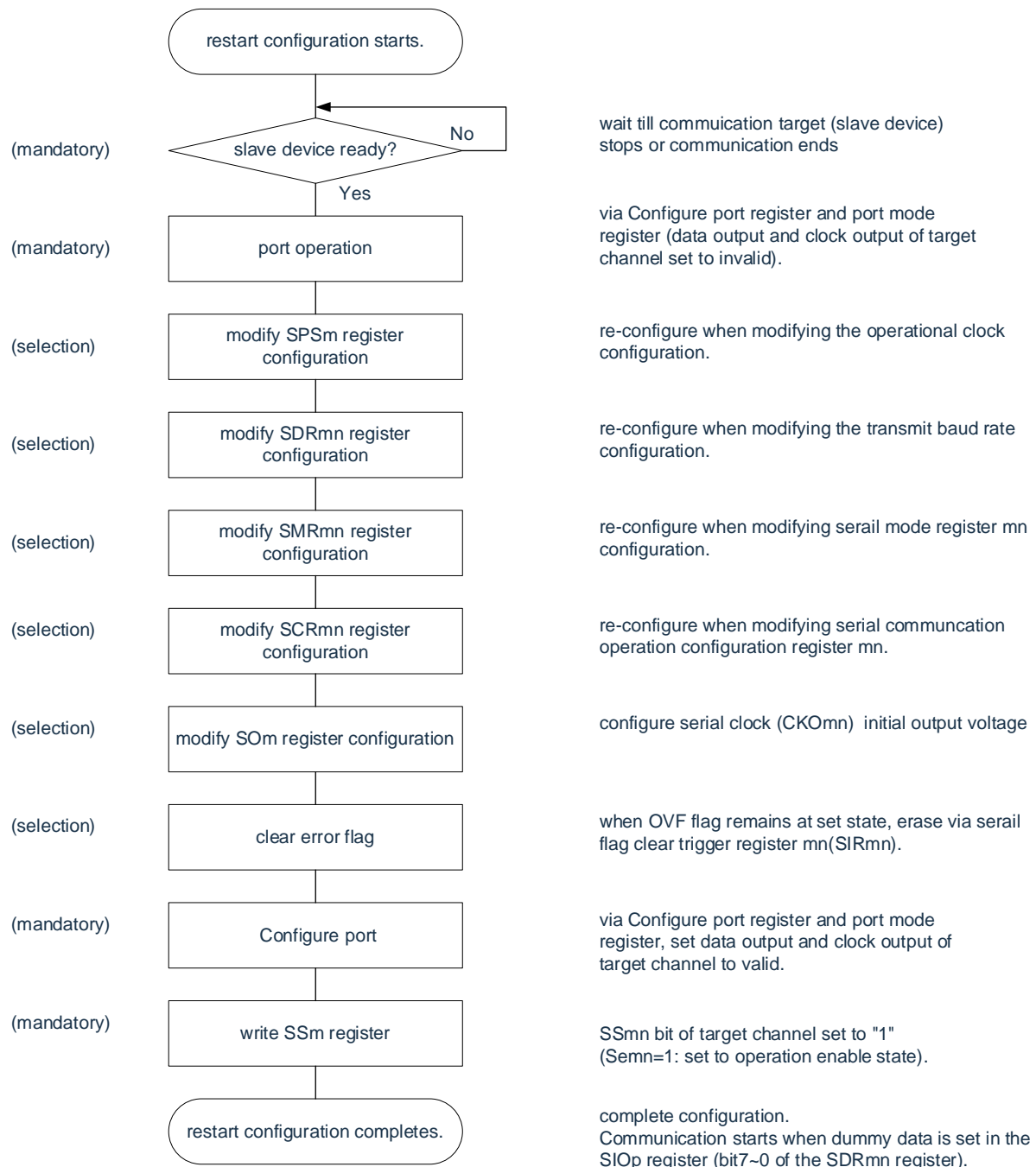
Figure 19-31 Initial setup steps for master reception

Figure 19-32 The abort step of the master receive


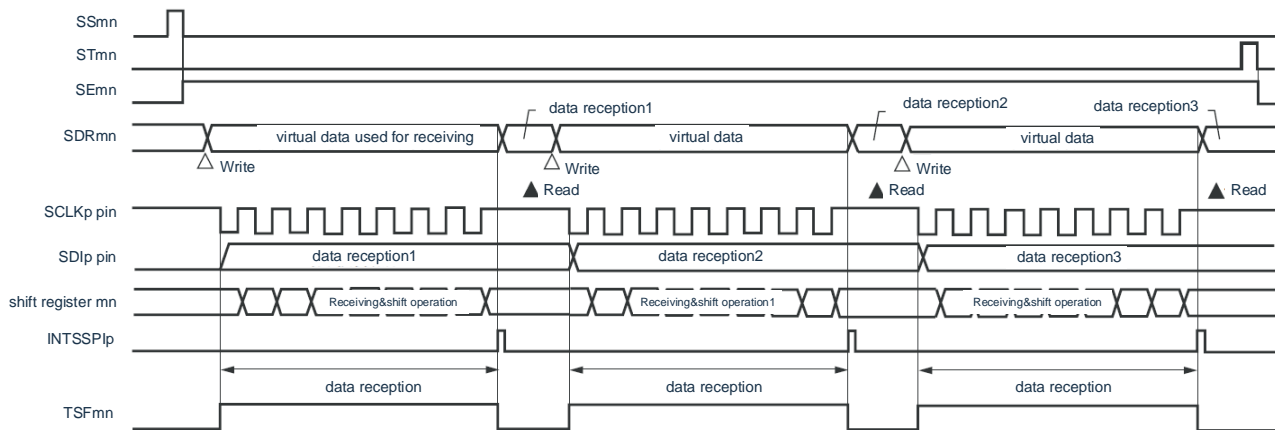
Figure 19-33 Restarts the setup step of the master receive



Note 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.

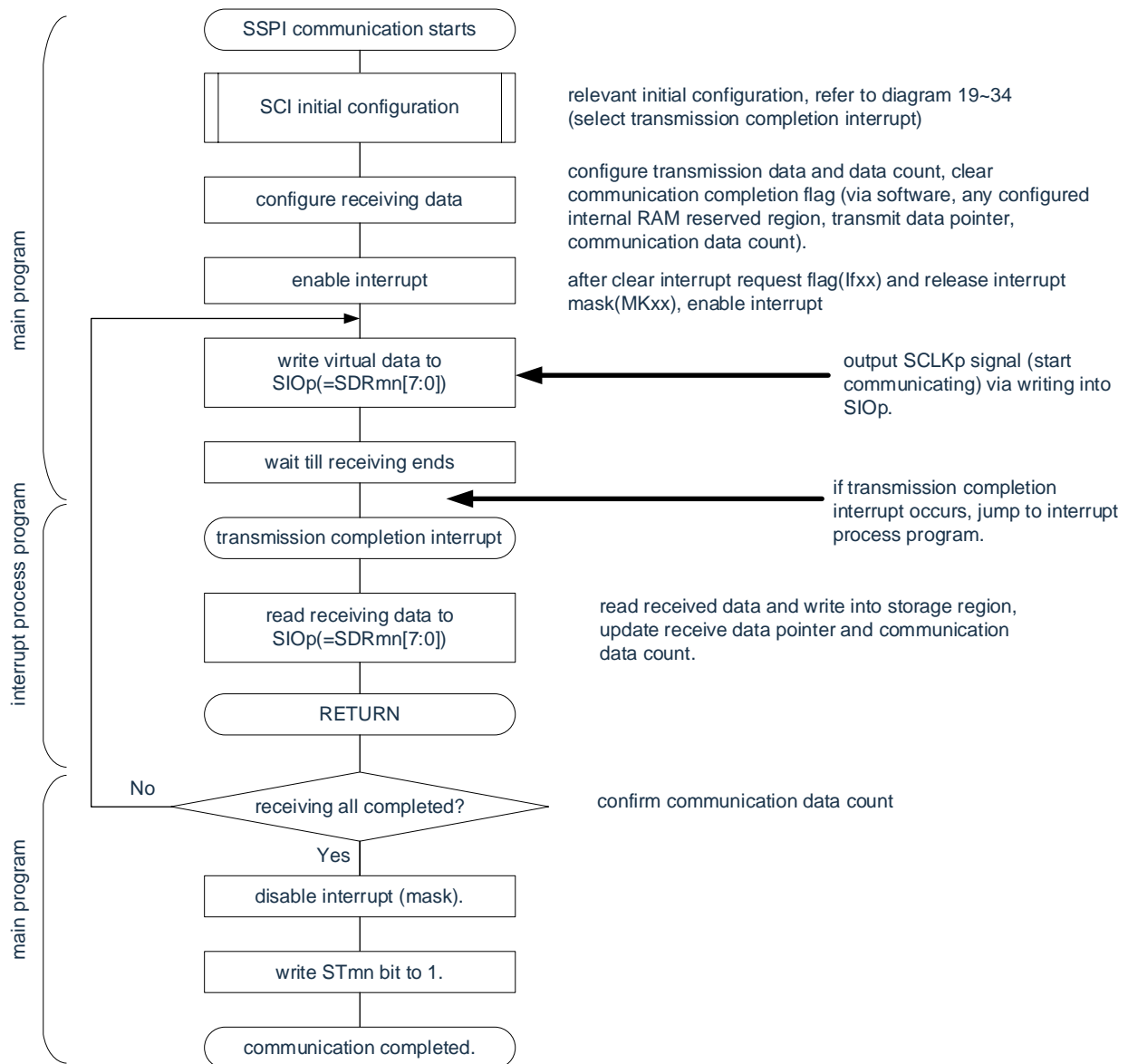
(3) Process flow (single receive mode).

Figure 19-34 Timing diagram of the master receive (single receive mode) (type 1: DAPmn = 0, CKPmn = 0).



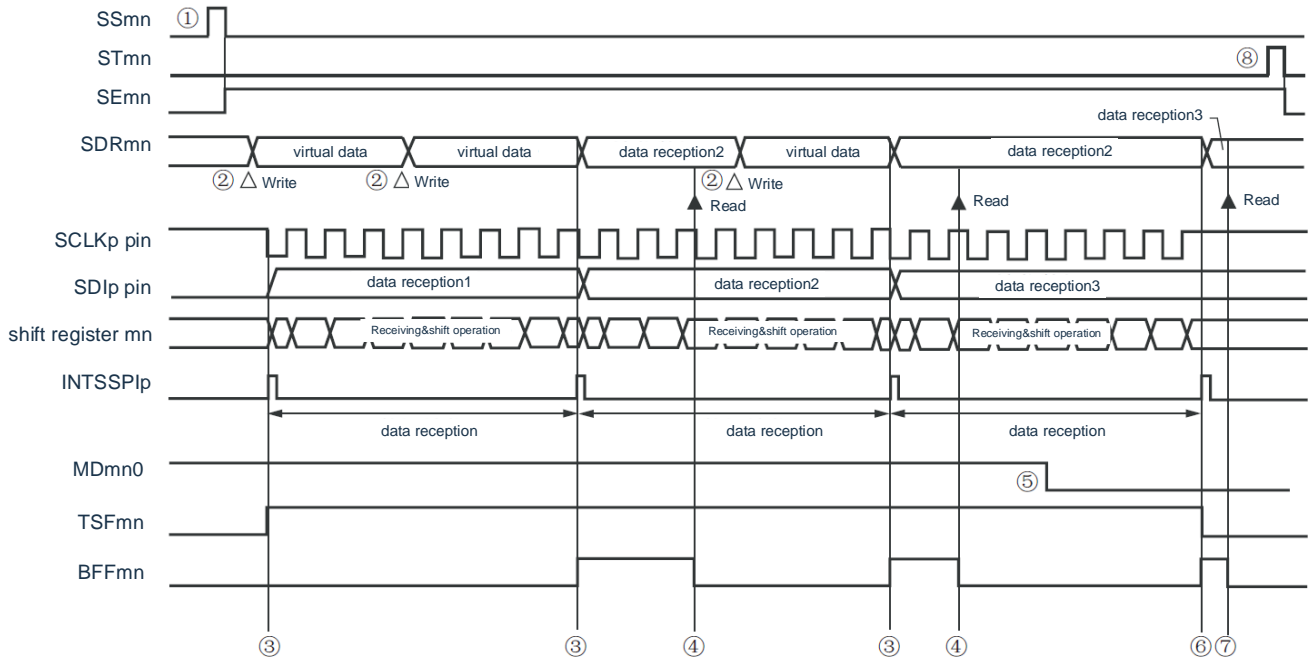
Remark m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)
mn=00~03, 10~11, 20~21

Figure 19-35 Flowchart of the master receive (single receive mode).



(4) Process flow (continuous receive mode).

Figure 19-36 the master receive (continuous receive mode) (type 1: DAPmn= 0, CKPmn = 0).

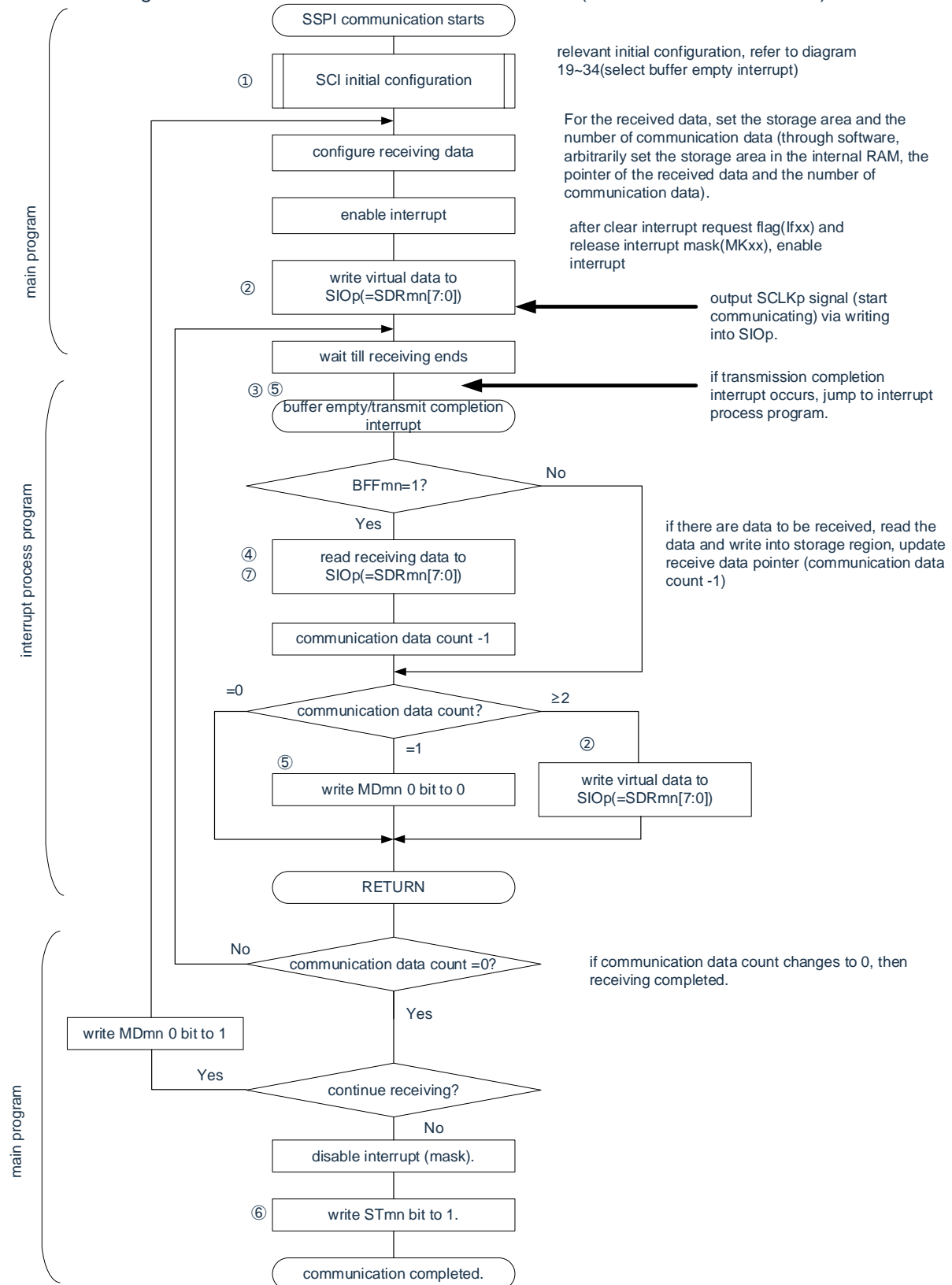


Note that the MDmn0 bit can be rewritten even during operation. However, in order to catch up with the end-of-transmission interruption of the last received data, it must be rewritten before the last bit can be received.

Note 1 (1) ~ (8) in the figure corresponds to (1) ~ (8) in Fig. 19-371937 Master Receive (Continuous Receive Mode)".

2.m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)

mn=00~03, 10~11, 20~21

Fig. 19-37 Flowchart of the master receiver (continuous receive mode).


Note (1) ~ (8) in the figure corresponds to (1) ~ (8) in the "Timing Diagram of Figure 19-36 Master Receive (Continuous Receive Mode)".

19.5.3 Sending and receiving of the master

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	SSPI30	SSPI31
Object channels	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1	SCI2 Channel 0	SCI2 Channel 1
The pins used	SCLK0 0.SDI0 0.SDO 0 0	SCLK01, SDI01, SDO01	SCLK10, SDI10, SDO10	SCLK1 1:Sdi1 1:SDO1 1	SCLK20, SDI20, SDO20	SCLK21, SDI21, SDO21	SCLK30, SDI30, SDO30	SCLK31, SDI31, SDO31
interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21	INTSSPI30	INTSSPI31
Error detection flags	You can choose between an end-of-transmit interrupt (single-pass mode) or a buffer-null interrupt (continuous transfer mode).							
The length of the transferred data	There is only the overflow error detection flag (OVFmn)							
Transfer Rate Note	SCI0: 7 or 8 bits SCI1/SCI2: 7 ~ 16bit							
Data phase	Max. $f_{CLK}/2$ [Hz] Min. $n.f_{CLK}/(2 \times 2^{15} \times 128)$ [Hz] f_{CLK} : System clock frequency							
Clock 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.							
Data direction	It can be selected by the CKPmn bit of the SCRm n register. • CKPmn=0: Normal phase • CKPmn=1: Inverted							
	MSB takes precedence or LSB takes precedence							

Note It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

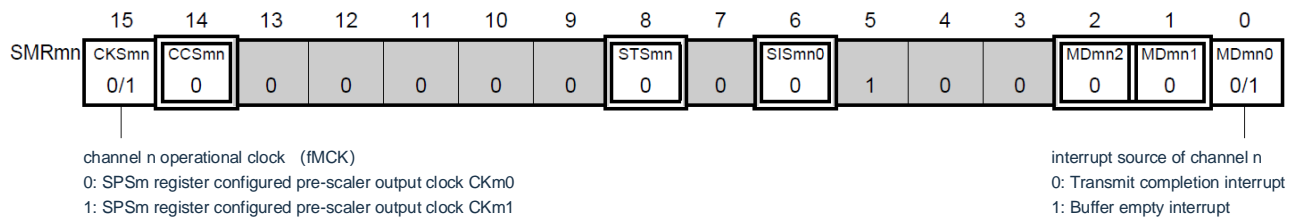
Remark m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)

mn=00~03, 10~11, 20~21

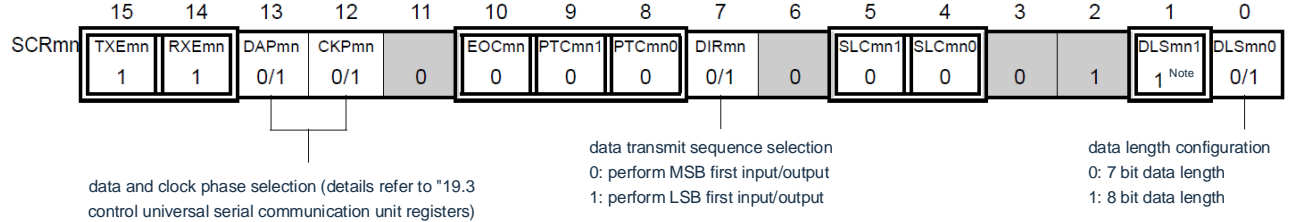
(1) Register settings

Fig19-38 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21, SSPI30, SSPI31)
Example of register settings for master sending and receiving

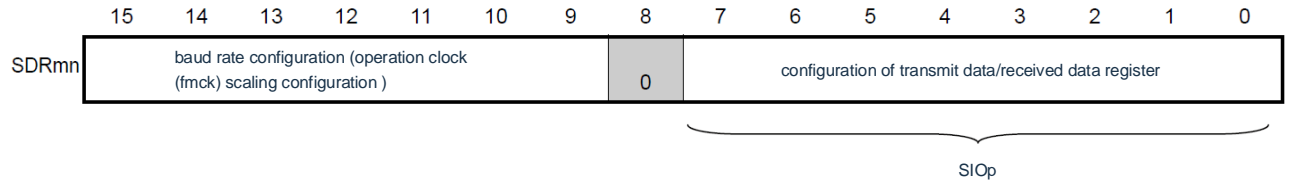
(a) serial mode register mn (SMRmn)



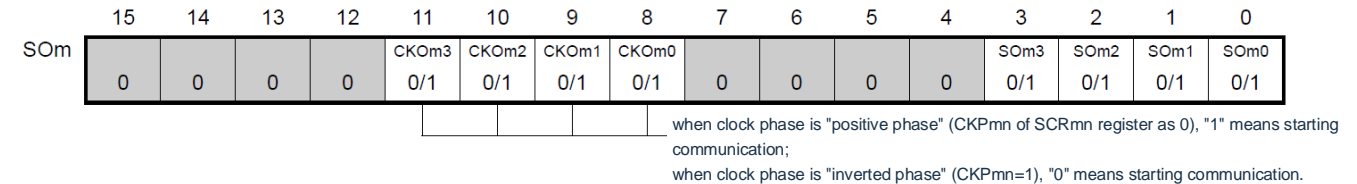
(b) serial communication operation configuration registermn mn(SCRmn)



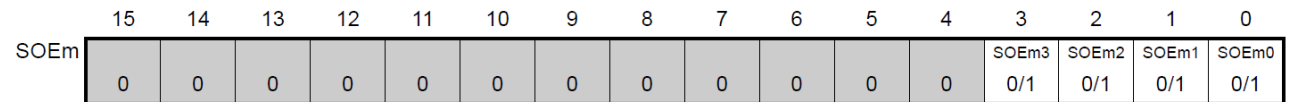
(c) serial data register mn (SDRmn) (low 8 bit: SIOp)



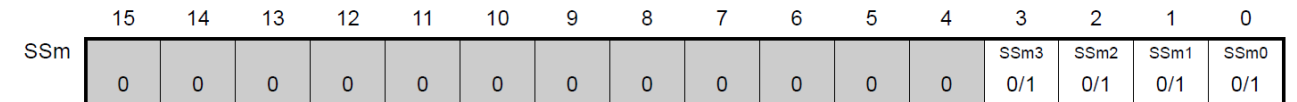
(d) serial output register m(SOm)Only configure bit of target channel



(e) serial output enable registerm (SOEm).....only set bit of target channel to 1.



(f) serial channel start register m (SSm) Only set bit of target channel to 1.




Note This example is the setting method for SCI0. The data length of SCI1/SCI2 and the serial data register SDRmn are set differently from this example.

For data length settings, refer to Chapter 19.3.4Serial communication runs the set register mn (SCRmn).Chapter 17).

For the setting method of serial data register SDRmn, refer to "19.3.6Serial data register mn(SDRmn) (SCI1/SCI2 i. e. m=1/2).

Remark1.m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)

mn=00~03, 10~11, 20~21

2.  Fixed setting (initial value is set).

 in SSPI master transmit and receive mode. : Cannot be set

0/1: Set "0" or "1" according to the user's purpose.

(2) Procedure

Figure 19-39 Initial setup steps for master sending and receiving

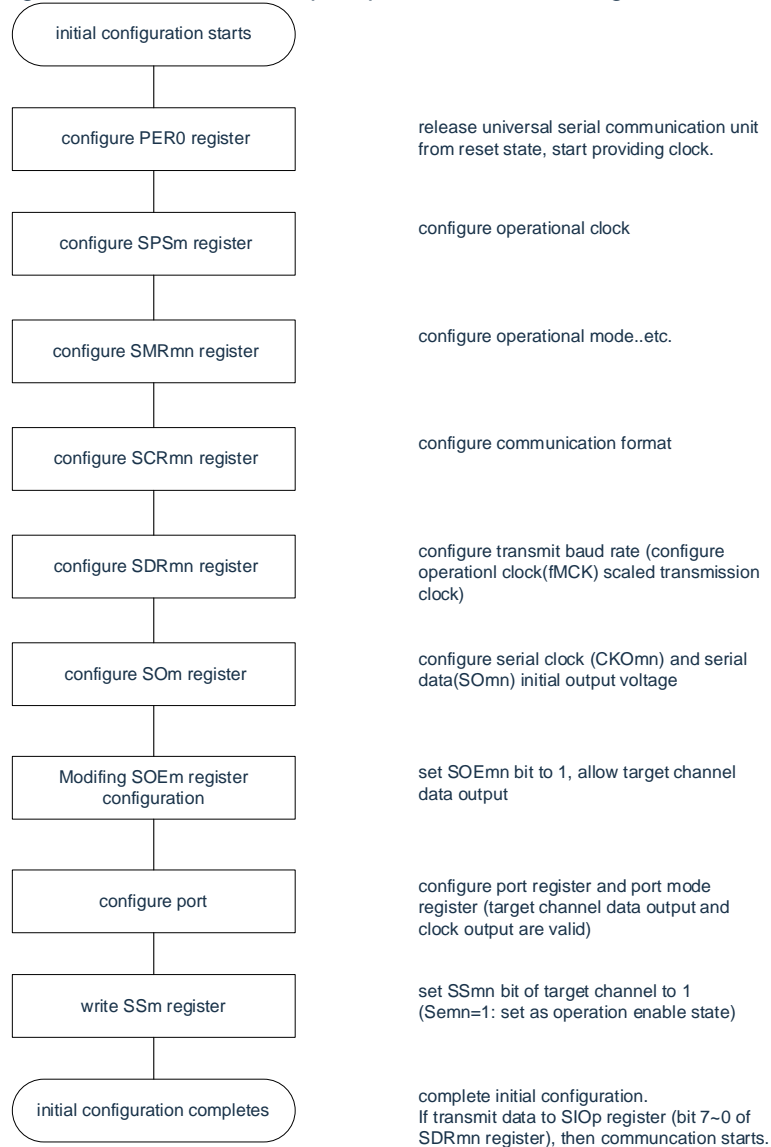


Figure 19-40 Abort steps of the master transmit and receive

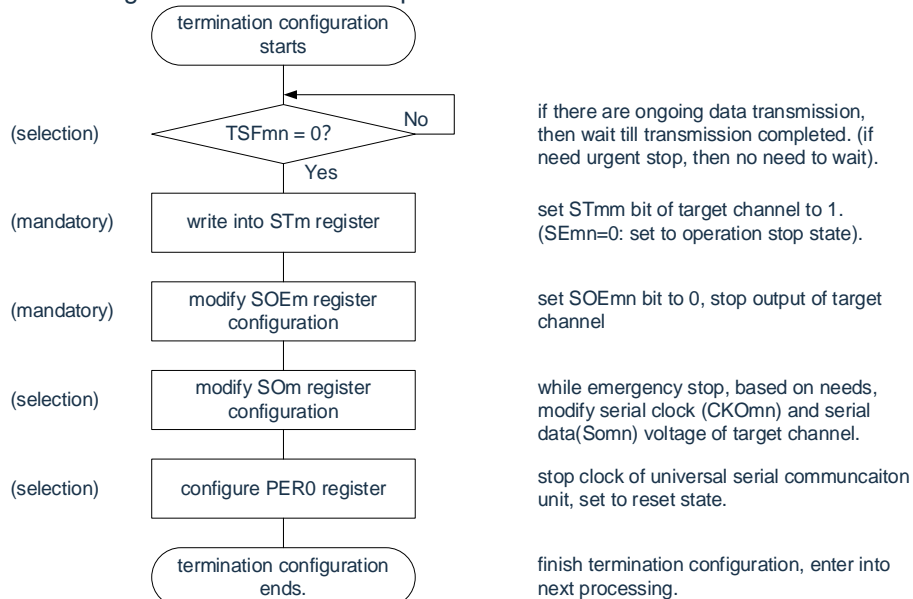
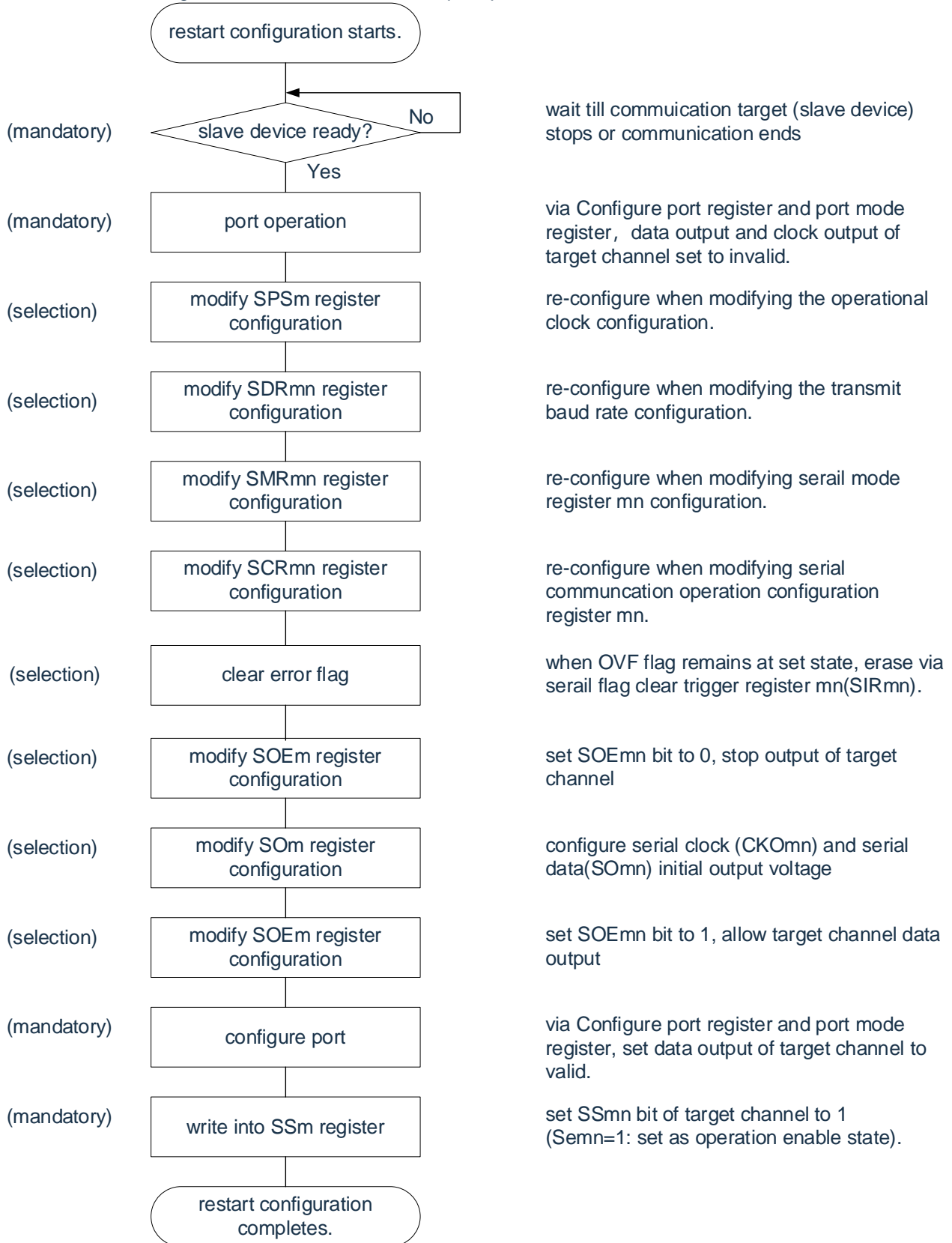
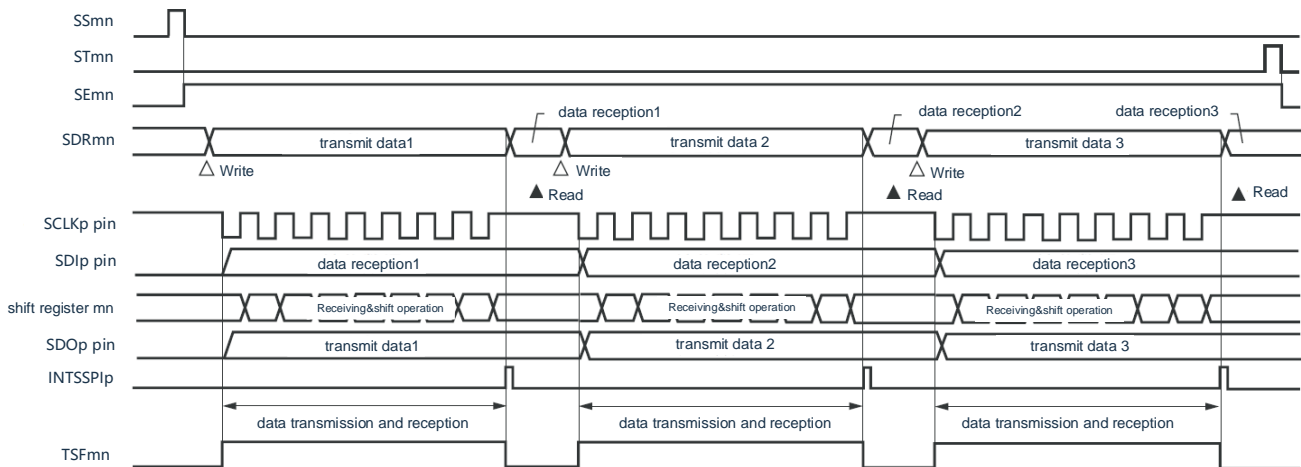


Figure 19-41 Restart the setup steps of the master send and receive



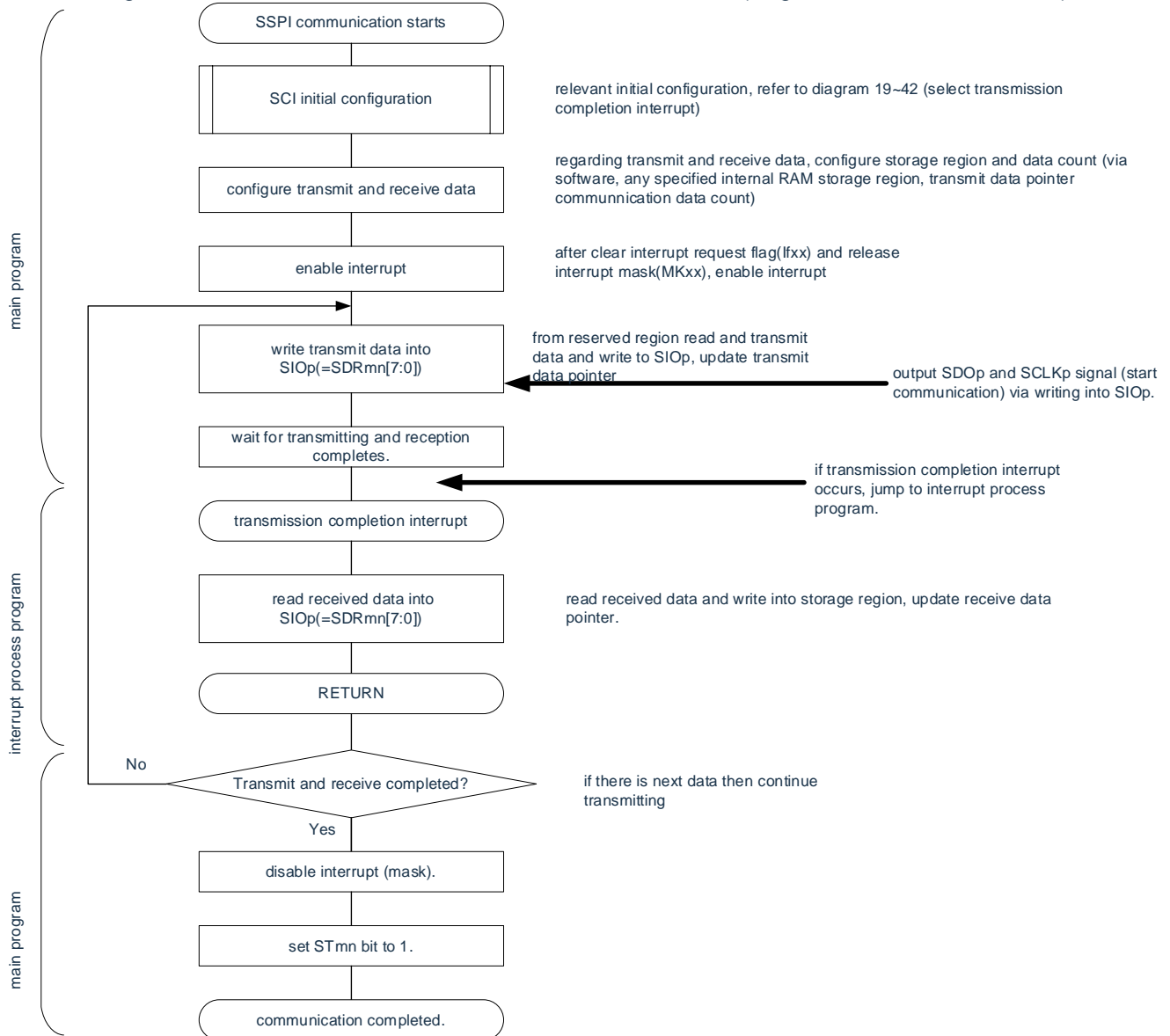
(3) Process flow (single send and receive mode).

Figure 19-42 Timing diagram of the master transmit and receive (single transmit and receive mode) (type 1: DAPmn=0, CKPmn=0).



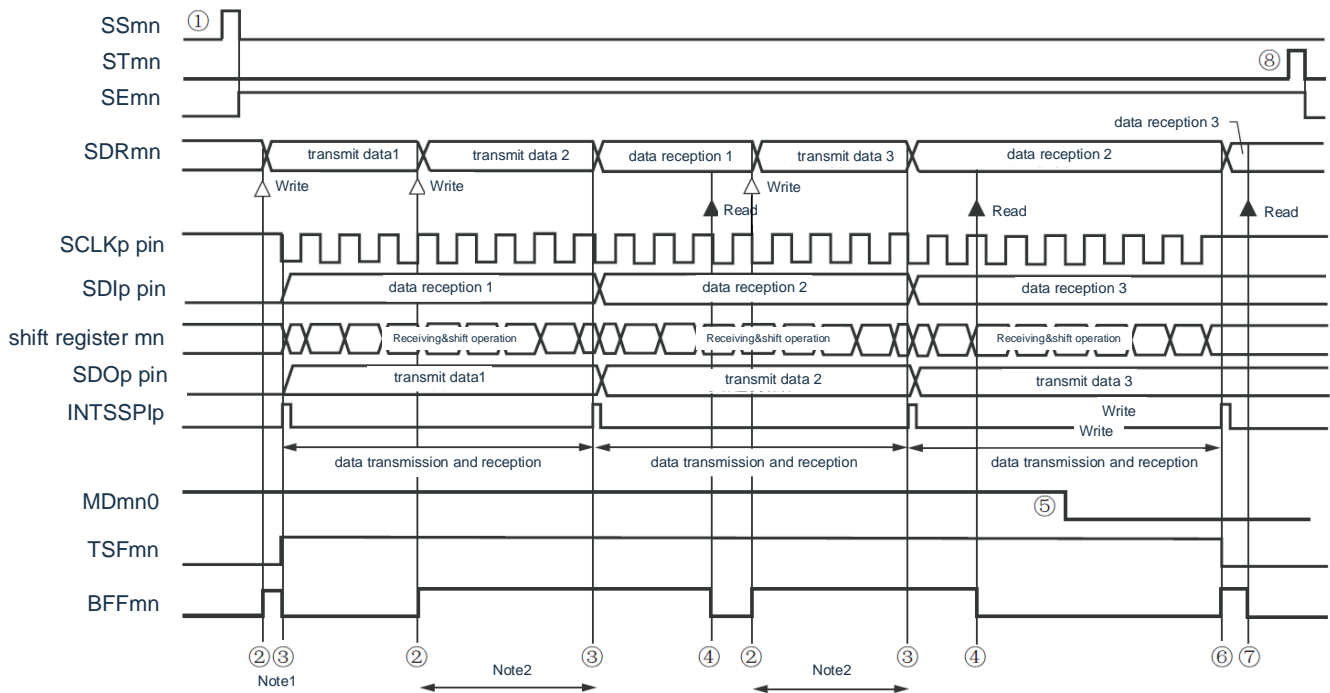
Remark m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)
mn=00~03, 10~11, 20~21

Figure 19-43 Flowchart of the master transmit and receive (single send and receive mode).



(4) Process flow (continuous send and receive mode).

Fig. 19-44: 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.

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.

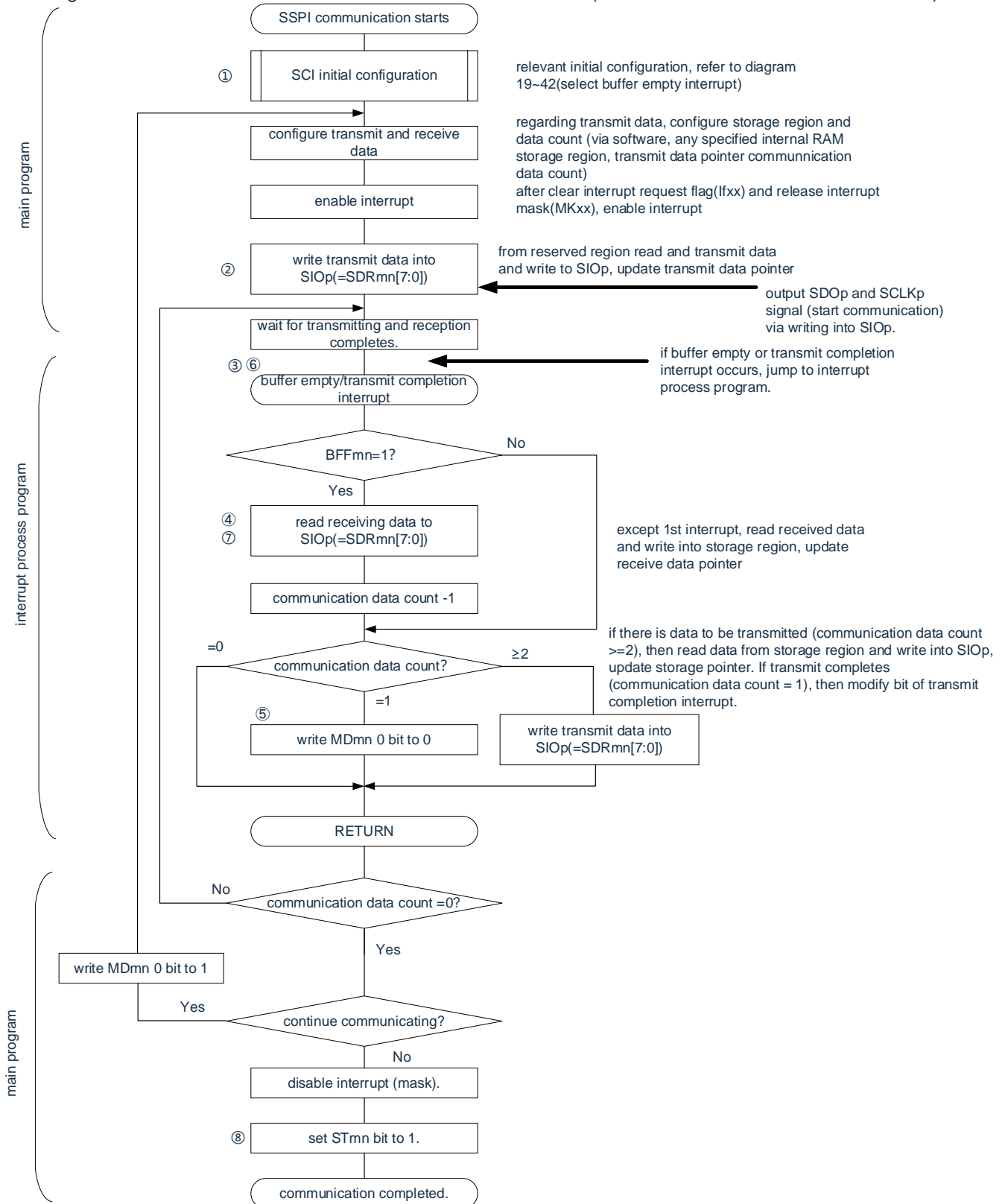
Note that the MDmn0 bit of the serial mode register mn (SMRmn) can be rewritten even in operation. However, in order to catch the end-of-transmission interruption of the last sent data, it must be rewritten before the last bit of transmission begins.

Note 1 (1) ~ (8) in the figure corresponds to "Figure 19-45 Master Transmit and Receive (Continuous Transmit and Receive Mode) flowchart" in (1) ~ (8).

2.m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)

mn=00~03, 10~11, 20~21

Figure 19-45 Flowchart of the master transmit and receive (continuous transmit and receive mode).



Note (1) ~ (8) in the figure corresponds to (1) ~ (8) in the "Timing Diagram of Fig. 19-44 Master Transmit and Receive (Continuous Transmit and Receive Mode)".

19.5.4 Slave sending

Slave transmission refers to the operation of the BAT32A2x9 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	SSPI30	SSPI31
Object channels	Channel 0 for SCI0	Channel 1 of SCI0	Channel 2 of SCI0	Channel 3 of SCI0	Channel 0 for SCI1	Channel 1 of SCI1	Channel 0 for SCI2	Channel 1 of SCI2
The pins used	SCLK00, SDO00	SCLK01, SDO01	SCLK10, SDO10	SCLK11, SDO11	SCLK20, SDO20	SCLK21, SDO21	SCLK30, SDO30	SCLK31, SDO31
interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21	INTSSPI30	INTSSPI31
Error detection flags	Selectable end-of-transmit interrupt (single-pass mode) or buffer-empty interrupt (continuous transfer mode).							
The length of the transferred data	SCI0: 7 or 8 bits SCI1/SCI2: 7 to 16 bits							
Transfer rate	Max. $f_{MCK}/6[Hz]$ ^{note 1, 2}							
Data phase	It can be selected by the DAPmn bit of the SCRmn register. • DAPmn=0: Starts data 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: Normal phase • CKPmn=1: Inverted							
Data direction	MSB priority or LSB priority							

Note 1 Because internally to SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, SCLK21, SCLK30, SCLK31 pin input external serial clock for sampling, so the maximum transfer rate is $f_{MCK}/6[Hz]$.

2. Must be used within the scope of peripheral functional characteristics (refer to data sheet) that meet this condition and meet the electrical characteristics.

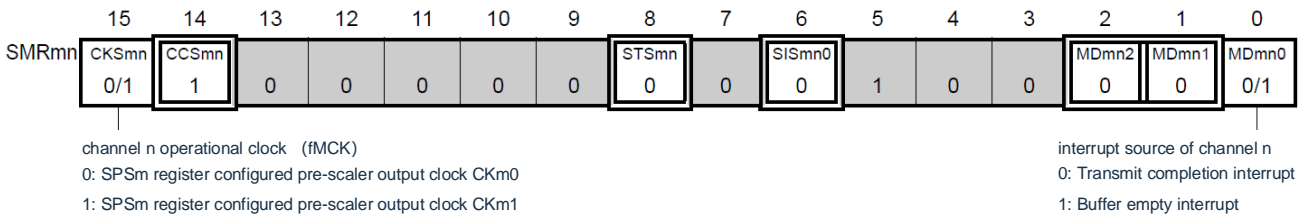
Note 1. f_{MCK} : The operating clock frequency of the object channel

2.m: unit number(m=0~2)n:channel number(n=0~3)mn=00~03, 10~11, 20~21

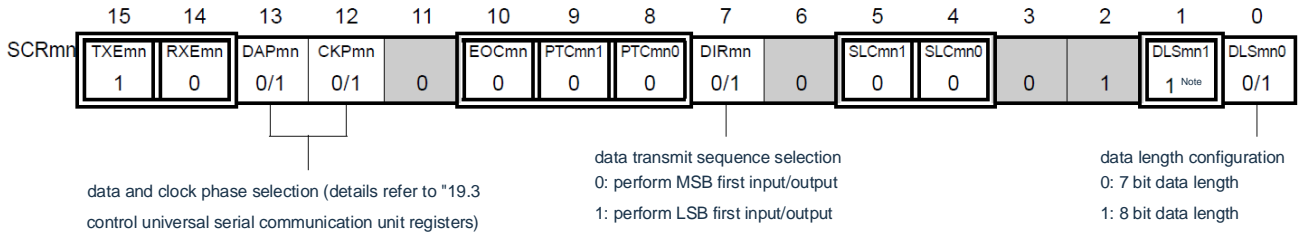
(1) Register settings

Fig19-46 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21, SSPI30, SSPI31)
Example of register settings at the time of Slave sending

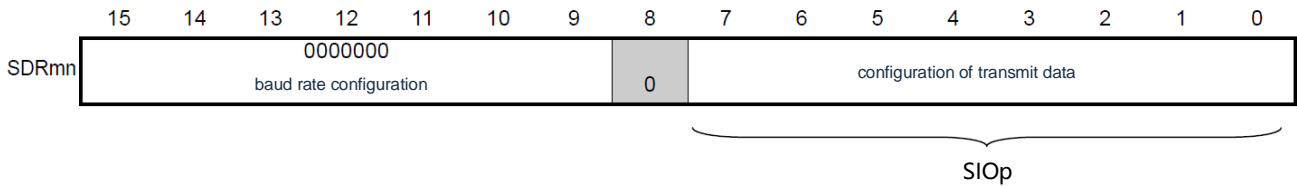
(a) serial mode register mn (SMRmn)



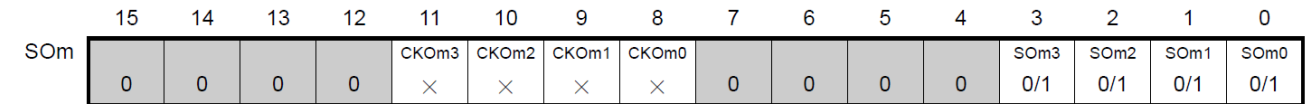
(b) serial communication operation configuration registermn mn(SCRmn)



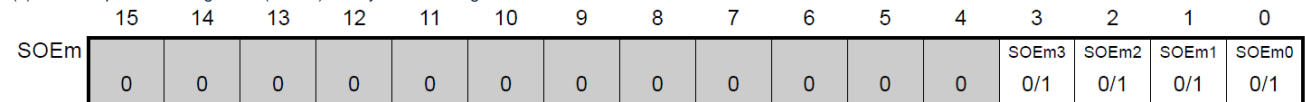
(c) serial data register mn (SDRmn) (low 8 bit: SIOp)



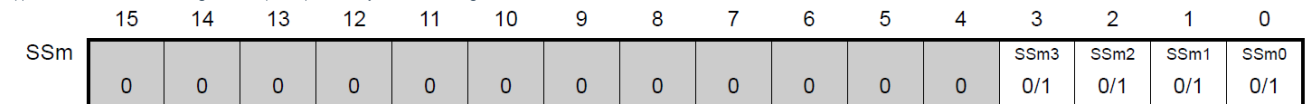
(d) serial output register m(SOm)Only configure bit of target channel



(e) serial output enable registerm (SOEm).....only set bit of target channel to 1.



(f) serial channel start register m (SSm) Only set bit of target channel to 1.



Note This example is the setting method for SCI0. The data length of SCI1/SCI2 and the serial data register SDRmn are set differently from this example.

For data length settings, refer to Chapter 19.3.4Serial communication runs the set register mn (SCRmn).Chapter 17).

For the setting method of serial data register SDRmn, refer to "19.3.6Serial data register mn(SDRmn) (SCI1/SCI2 i. e. m=1/2).

Remark1.m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)
mn=00~03, 10~11, 20~21

2. : Fixed setting in SSPI Slave send mode.

 : Cannot be set (initial value is set).

×

 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

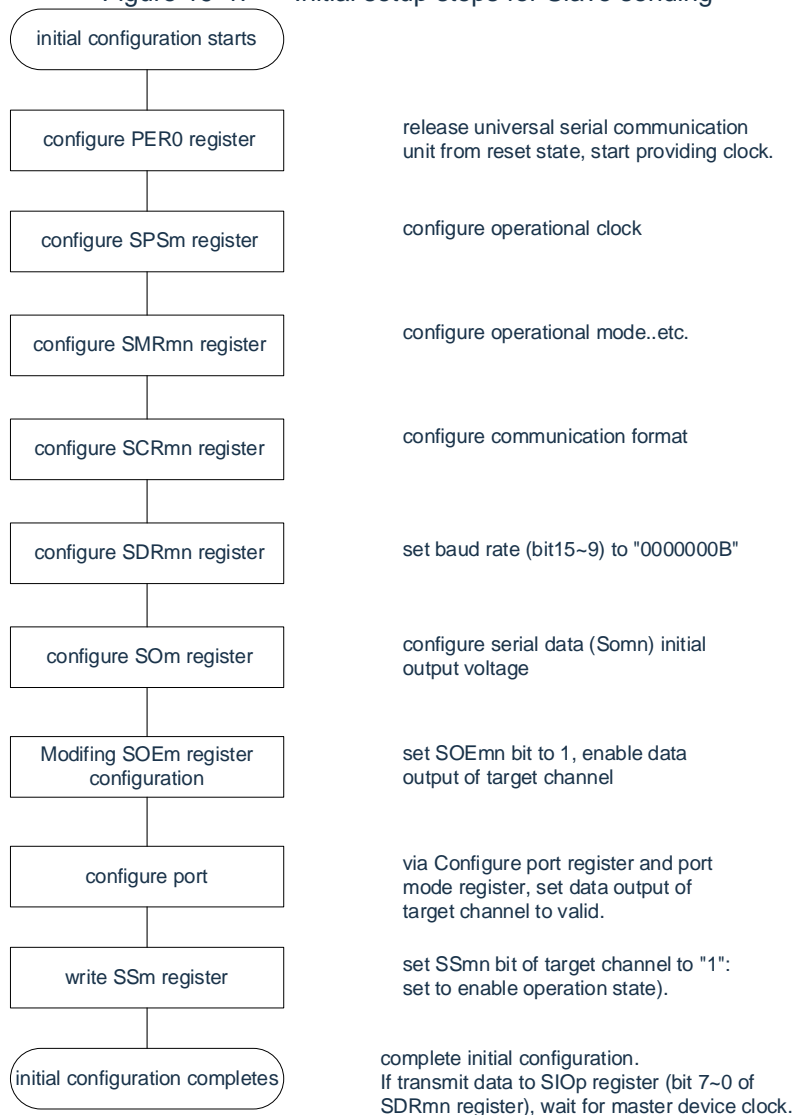
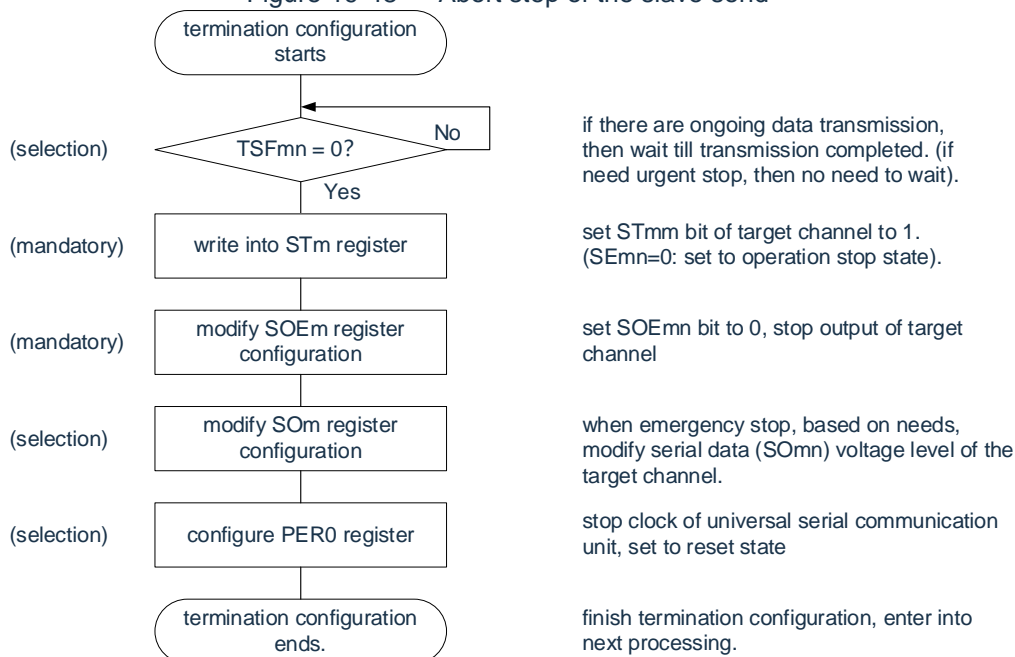
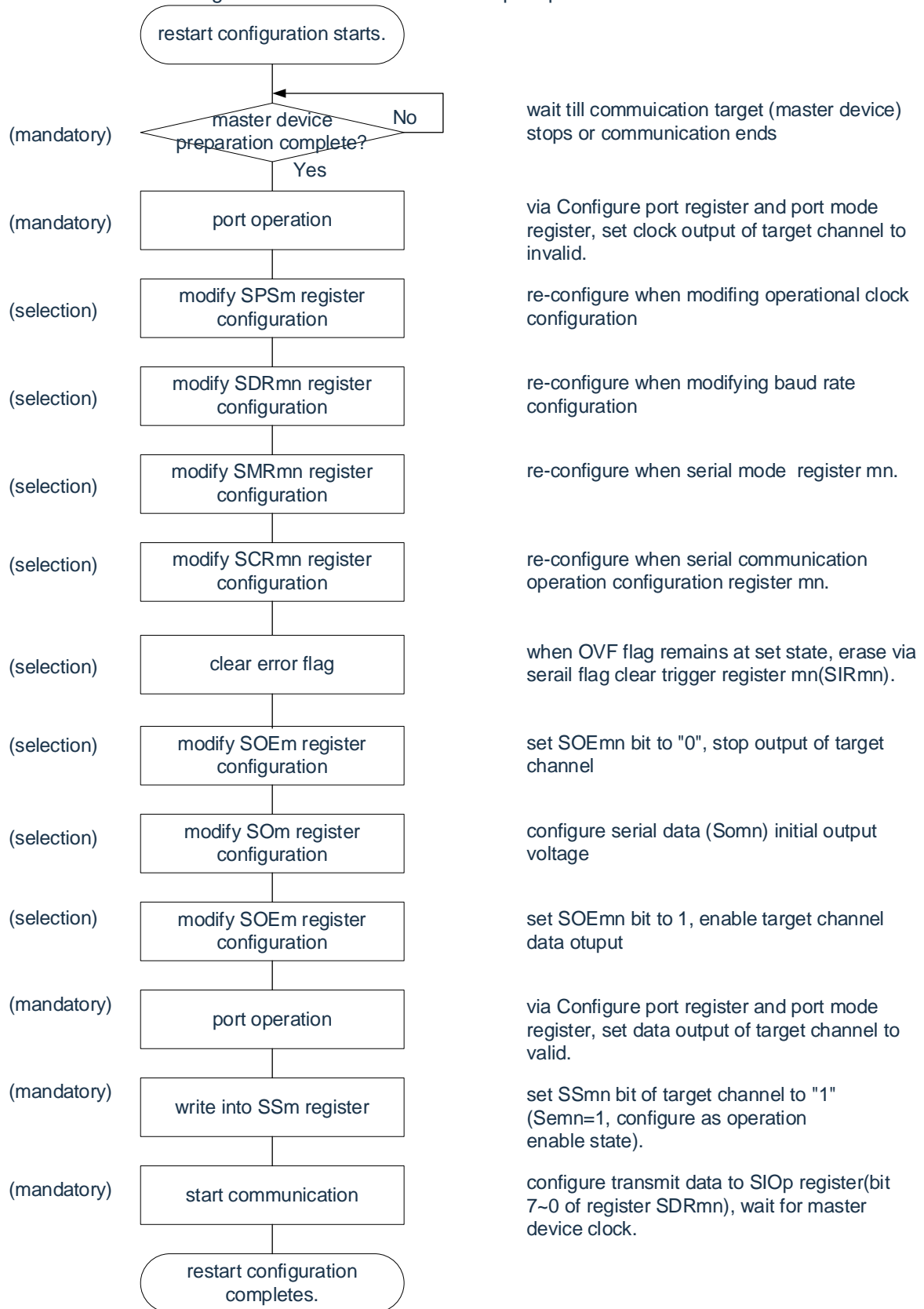
Figure 19-47 Initial setup steps for Slave sending

Figure 19-48 Abort step of the slave send


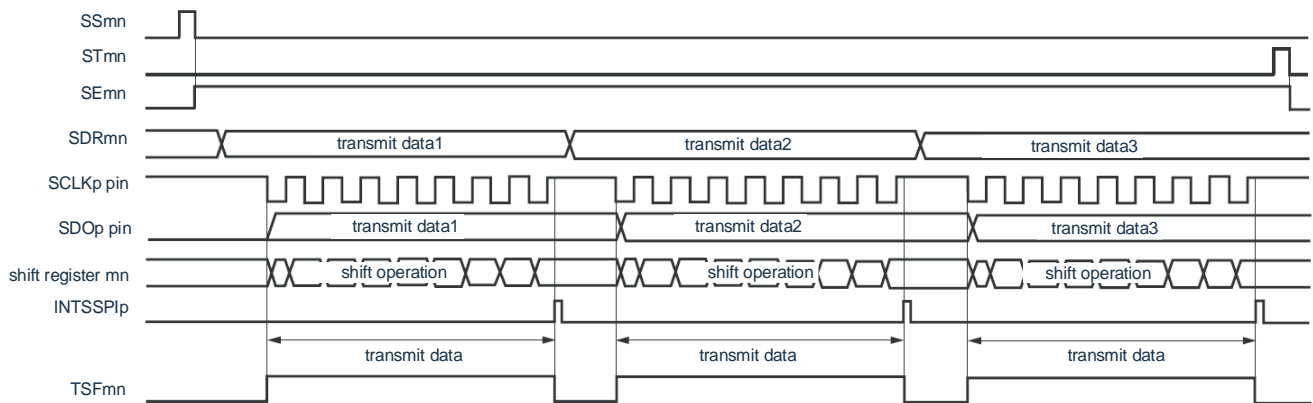
Figure 19-49 Restarts the setup step of the slave send



Note If you override PER0 in the abort 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.

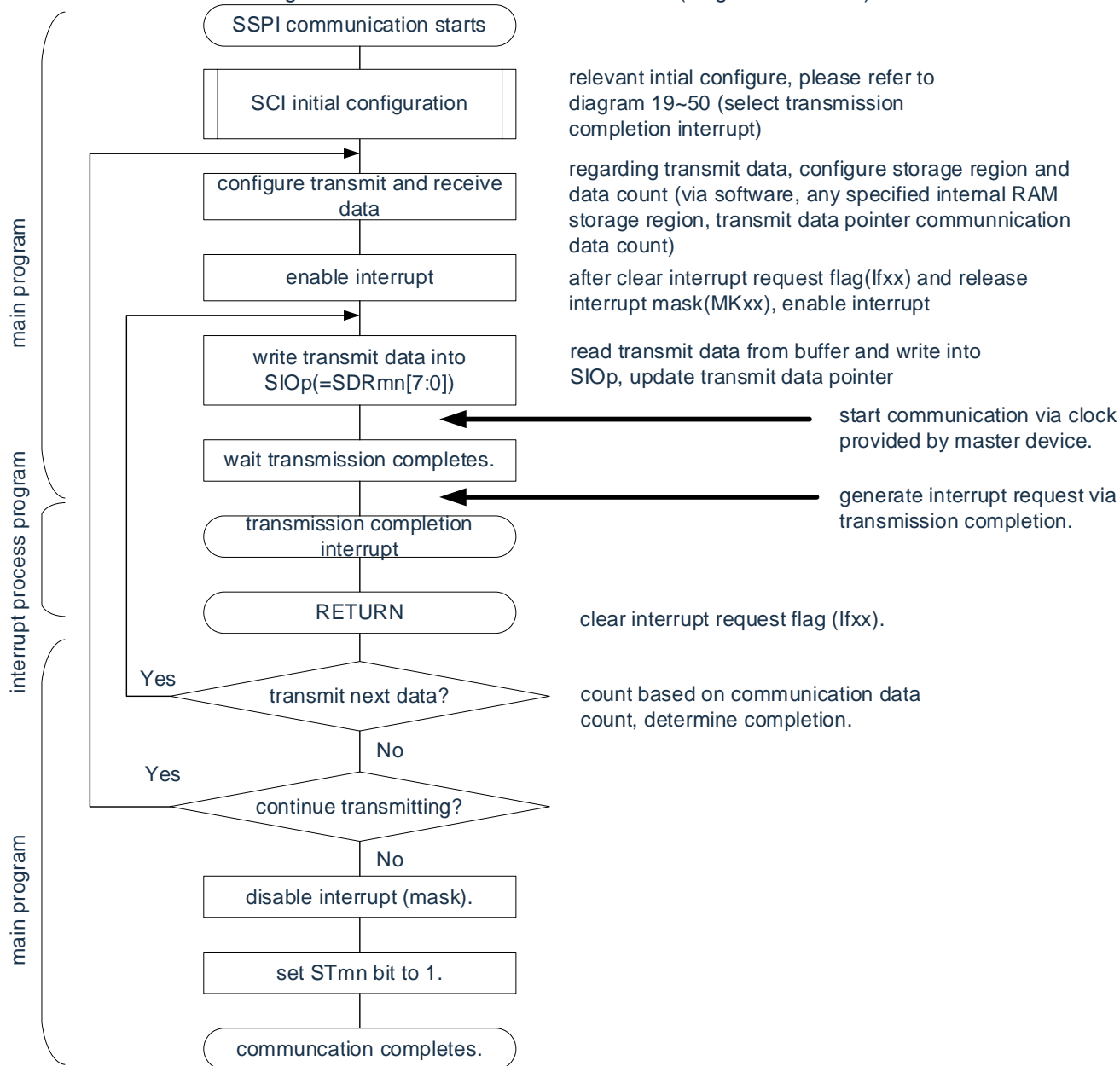
(3) Process flow (single send mode).

Figure 19-50 Timing diagram of a Slave send (single send mode) (type 1: DAPmn=0, CKPmn=0).



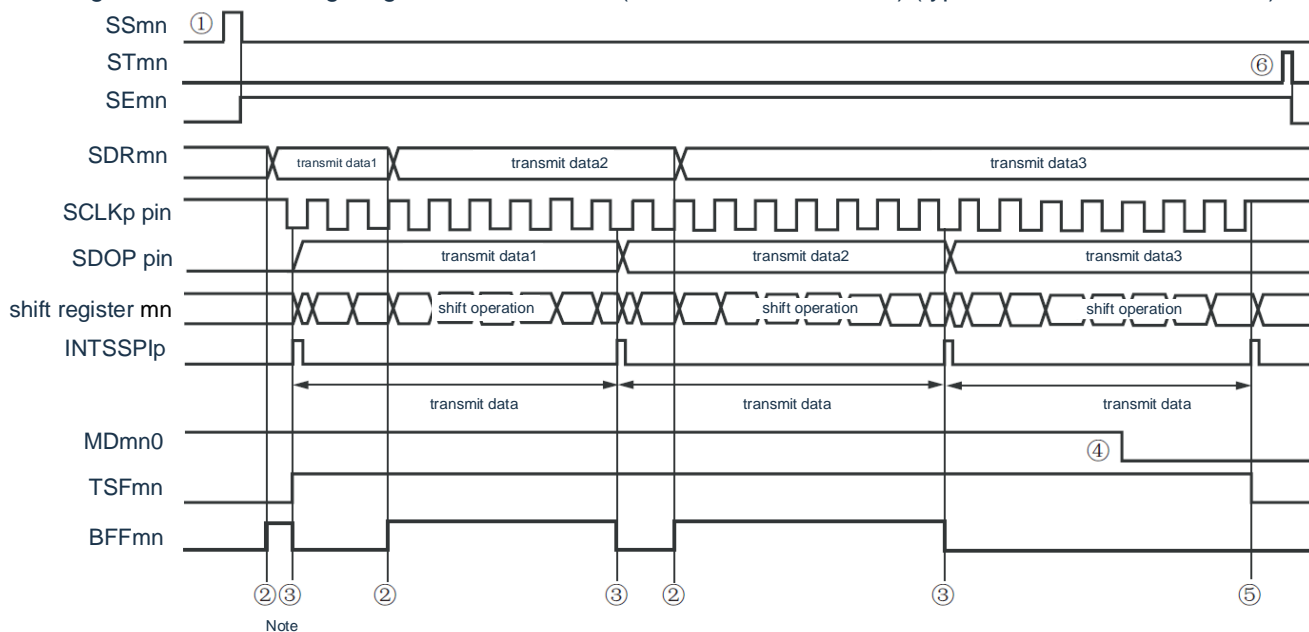
Remark m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)
mn=00~03, 10~11, 20~21

Figure 19-51 Flowchart of Slave send (single send mode).



(4) Process flow (continuous send mode).

Fig. 19-52 Timing diagram of Slave send (continuous send mode) (type 1: DAPmn=0, CKPmn=0).



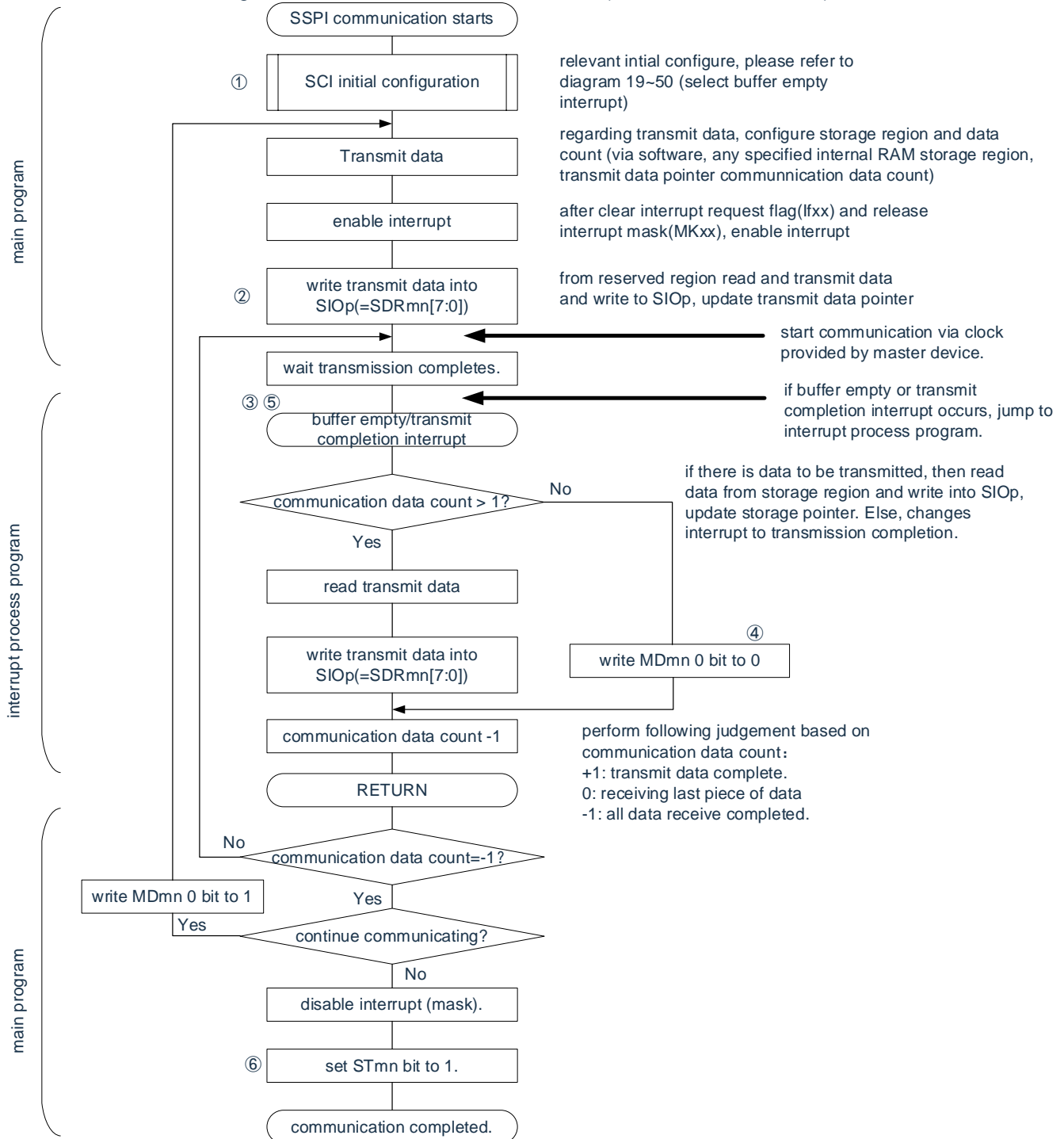
Note If the BFFmn bit of the serial status register mn(SSRmn) is "1" (when valid data is saved in the serial data register mn(SDRmn)) is given When the SDRmn register writes the transmit data, it rewrites the send data.

Note that the 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=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)
mn=00~03, 10~11, 20~21



Figure 19-53 Flowchart of Slave send (continuous send mode).



Note (1)~(6) in the note figure corresponds to (1)~(6) in the "Timing Diagram of Fig. 19-52 Slave Sending (Continuous Send Mode)".

19.5.5 Slave receive

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	SSPI30	SSPI31
Object channels	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1	SCI2 Channel 0	SCI2 Channel 1
The pins used	SCLK00, SDI00	SCLK01, SDI01	SCLK10, SDI10	SCLK11, SDI11	SCLK20, SDI20	SCLK21, SDI21	SCLK30, SDI30	SCLK31, SDI31
interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21	INTSSPI30	INTSSPI31
	Limited to end-of-transmit interrupts (buffer null interrupts are prohibited).							
Error detection flags	There are only overflow error detection flags (OVFmn).							
The length of the transferred data	SCI0: 7 or 8 bits SCI1/SCI2: 7 to 16 bits							
Transfer rate	Max. $f_{MCK}/6[Hz]$ ^{note1, 2}							
Data phase	It can be selected by the DAPmn bit of the SCRmn register. •DAPmn=0: 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: Normal phase • CKPmn=1: Inverted							
Data direction	MSB priority or LSB priority							

Note 1 Because internally to SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, SCLK21, SCLK30, SCLK31 pin input external serial clock for sampling, so the maximum transfer rate is $f_{MCK}/6[Hz]$.

- Must be used within the scope of peripheral functional characteristics (refer to data sheet) that meet this condition and meet the electrical characteristics.

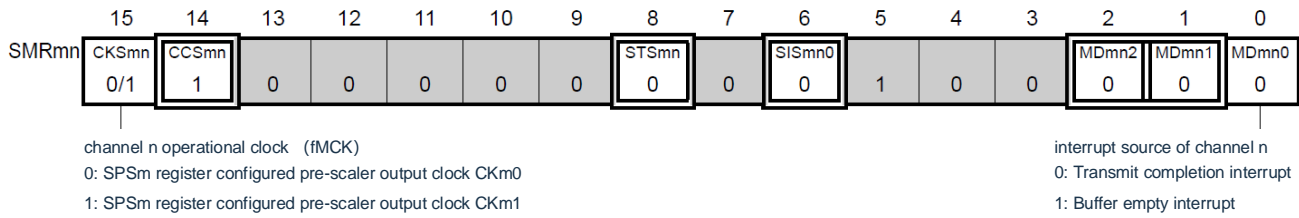
Note 1. f_{MCK} : The operating clock frequency of the object channel

2.m:unit number(m=0~2)n:channel number(n=0~3)mn=00~03, 10~11, 20~21

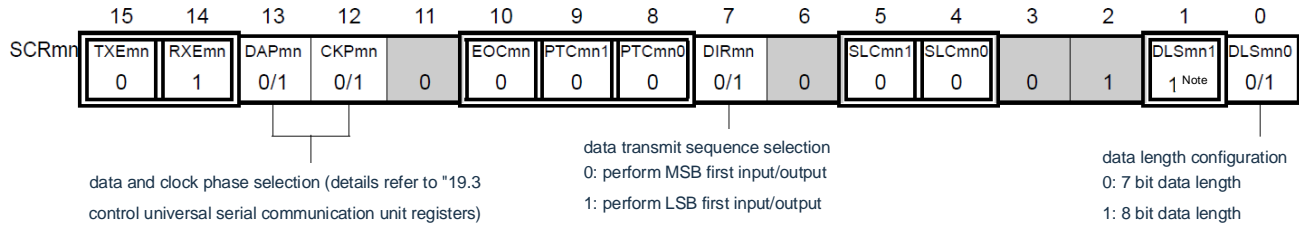
(1) Register settings

Fig19-54 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21, SSPI30, SSPI31)
Example of register setting content at slave receive time

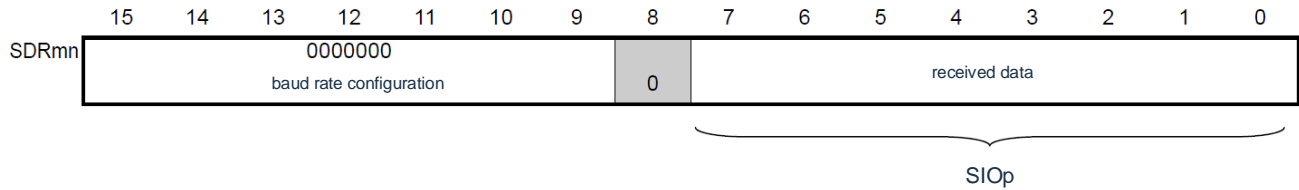
(a) serial mode register mn (SMRmn)



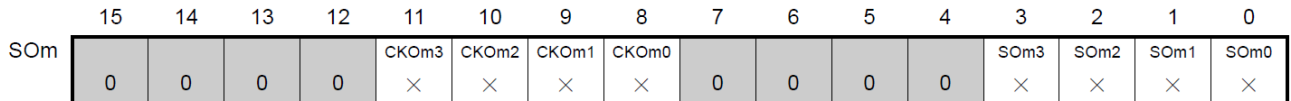
(b) serial communication operation configuration registermn mn(SCRmn)



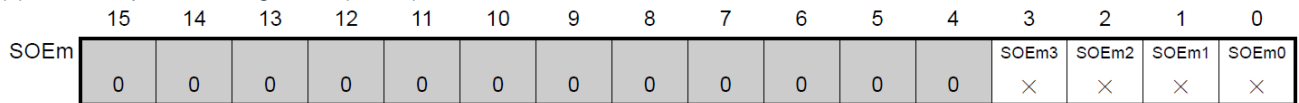
(c) serial data register mn (SDRmn) (low 8 bit: SIOp)



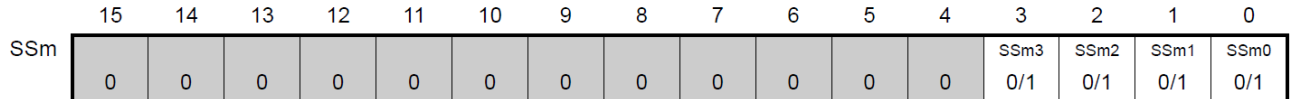
(d) serial output register m (Som) Not used in this mode.



(e) serial output enable register m (SOEm).... Not used in this mode.



(f) serial channel start register m (SSm) Only set bit of target channel to "1".



Note This example is the setting method for SCI0. The data length of SCI1/SCI2 and the serial data register SDRmn are set differently from this example.

For data length settings, refer to Chapter 19.3.4Serial communication runs the set register mn (SCRmn).Chapter 17).

For the setting method of serial data register SDRmn, refer to "19.3.6Serial data register mn(SDRmn) (SCI1/SCI2 i. e. m=1/2).

Remark1.m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)
mn=00~03, 10~11, 20~21

- : Fixed setting in Slave receive mode.
 : Cannot be set (initial value is set).
- 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) Operation steps

Fig. 19-55

Initial setup step of Slave reception

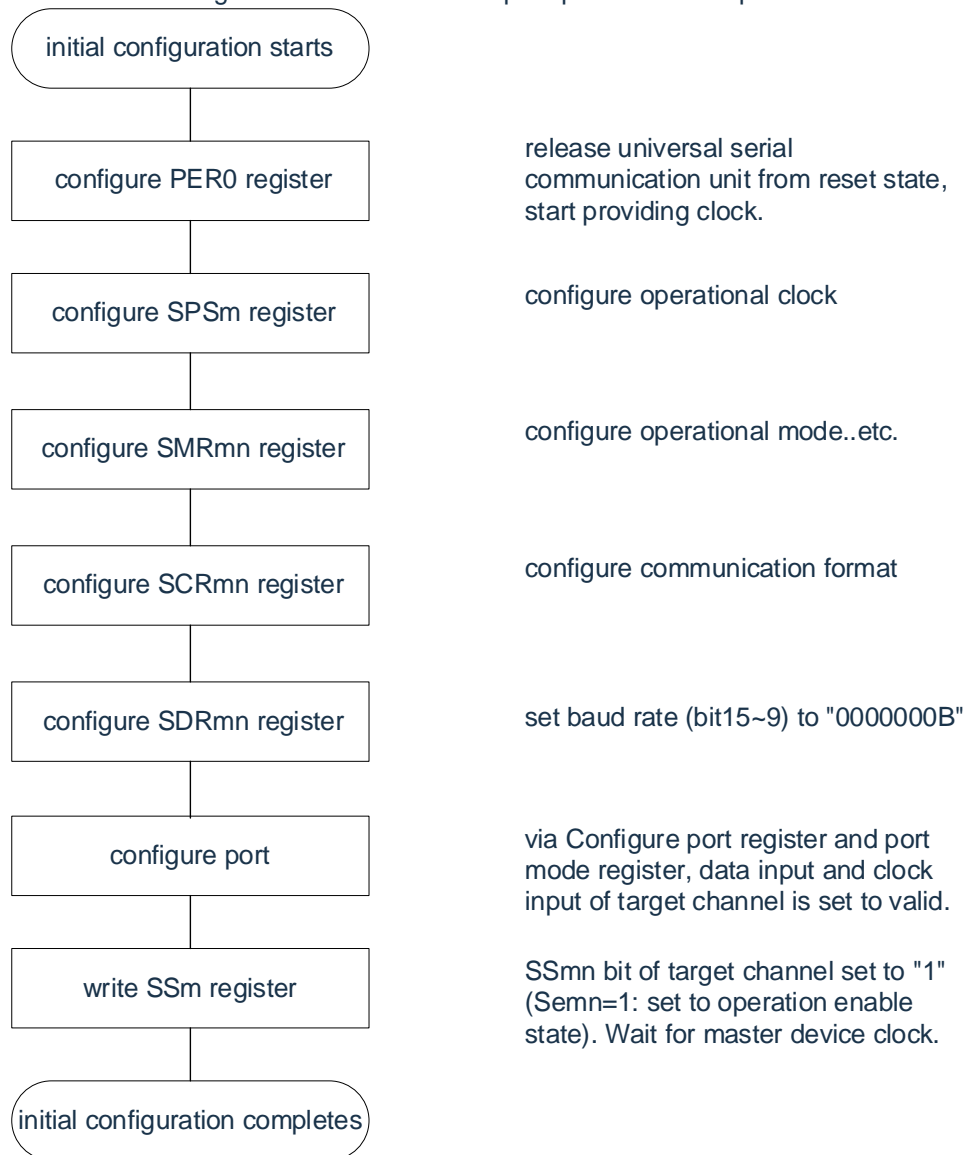


Figure 19-56

Abort step of Slave reception

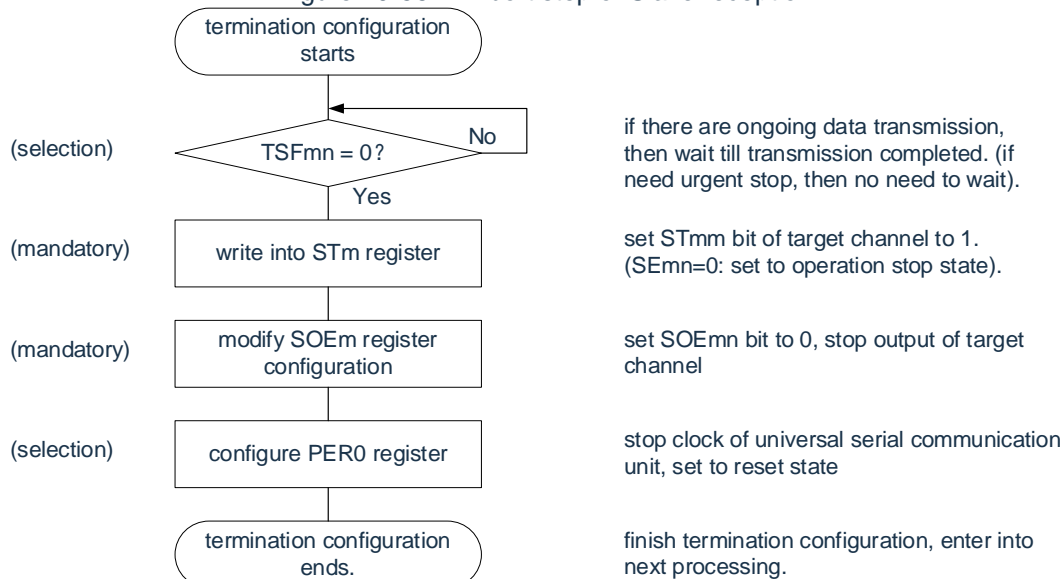
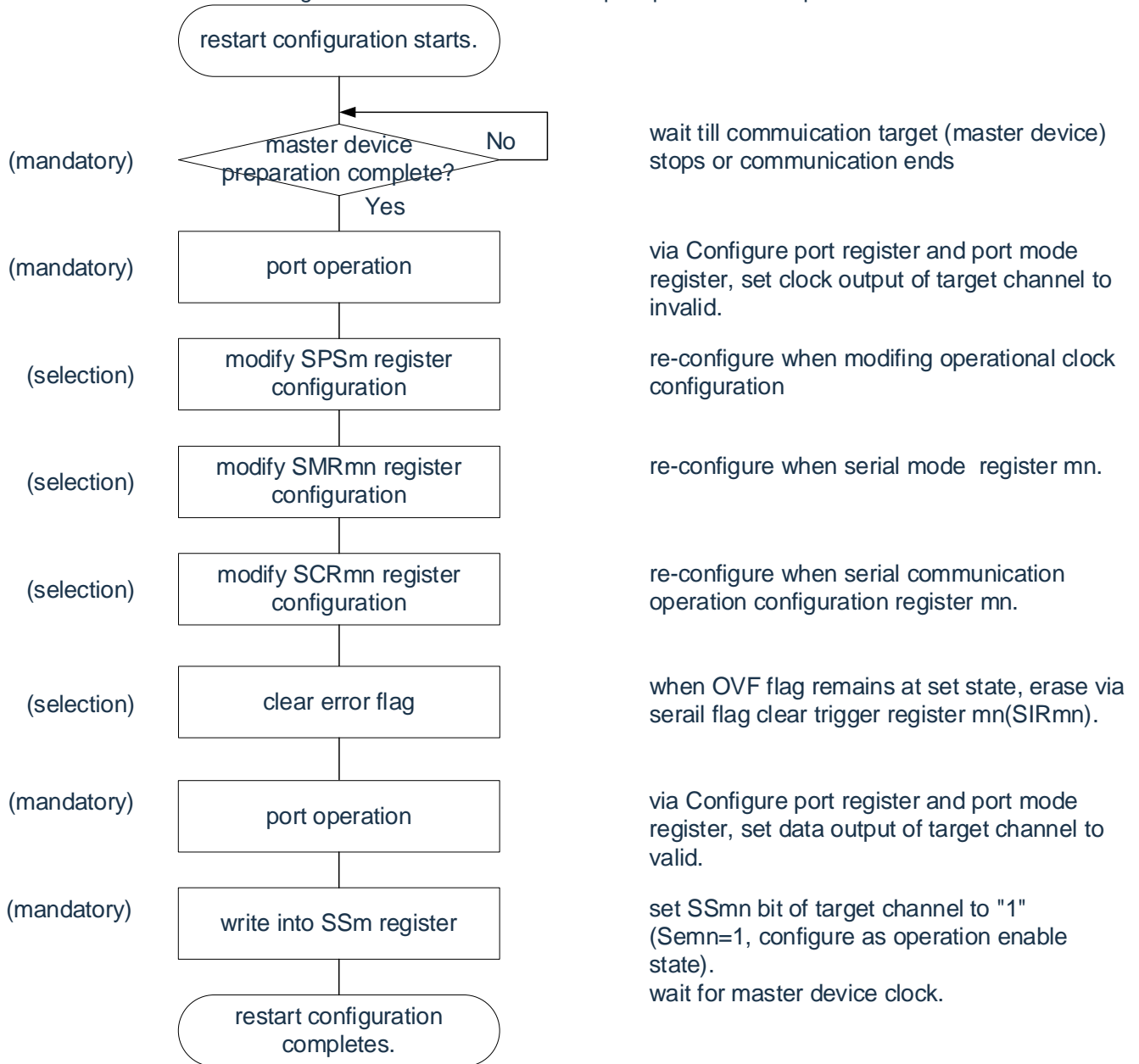


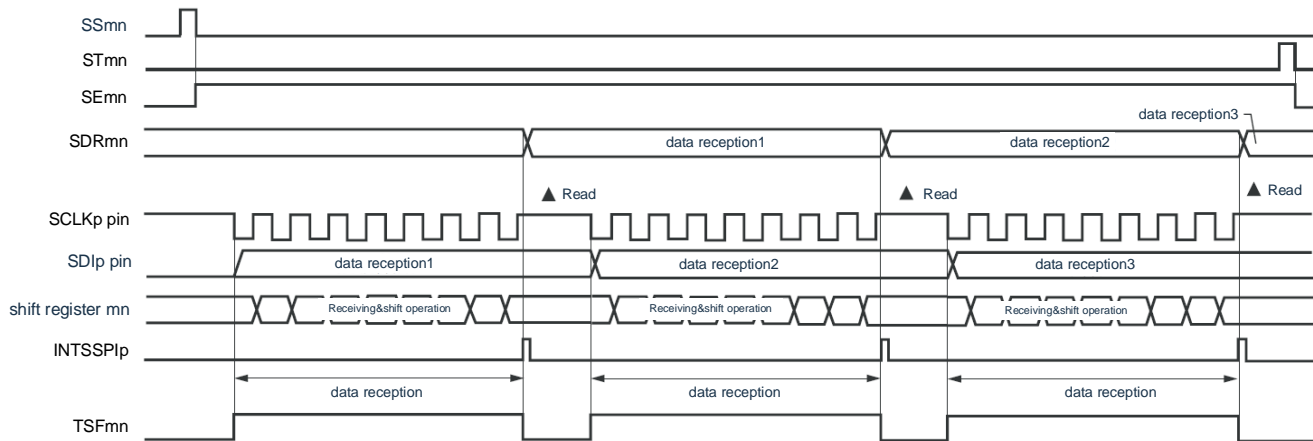
Fig. 19-57 Restart the setup step of Slave reception



Notelf you override PER0 in the abort 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.

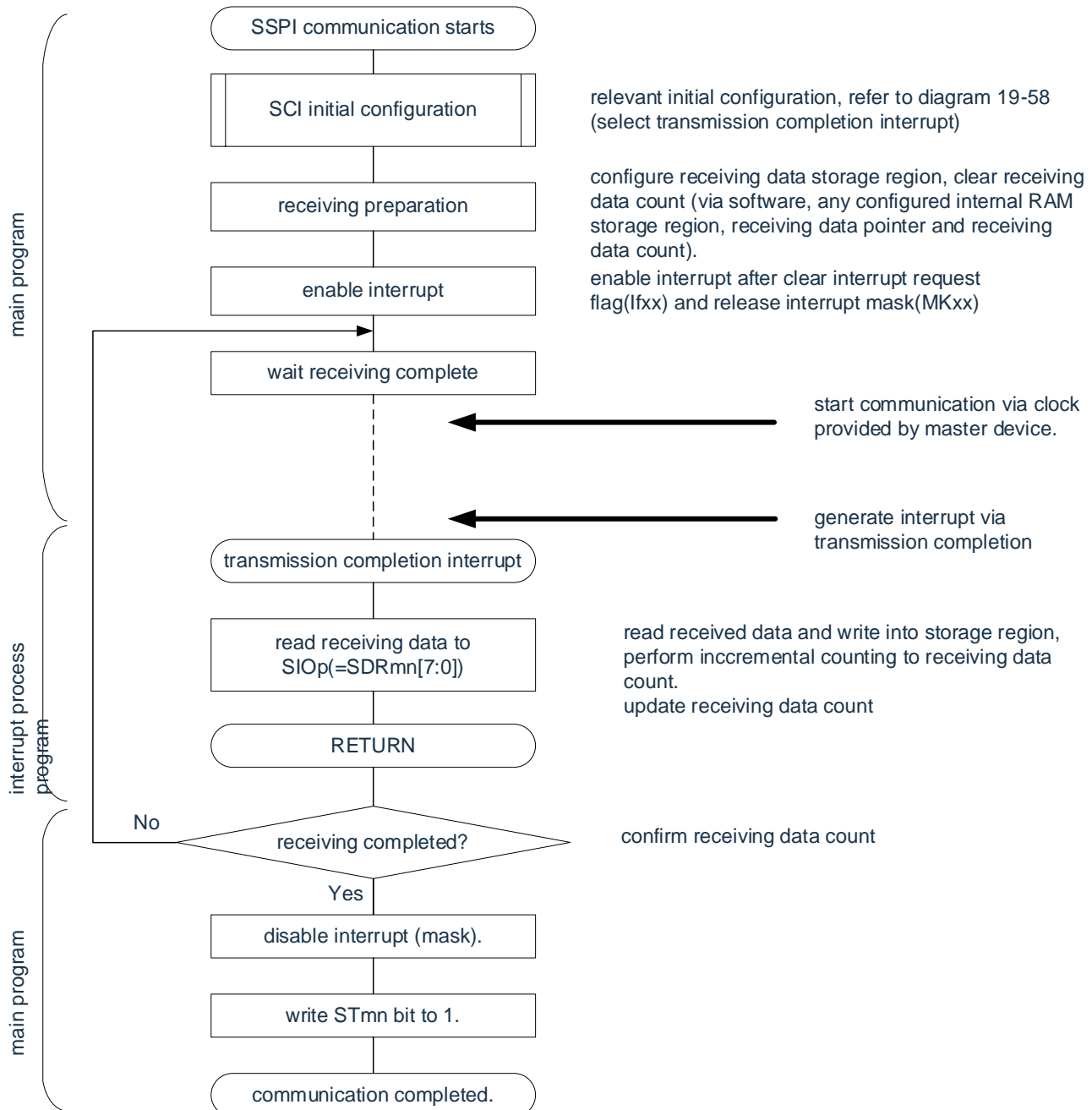
(3) Process flow (single receive mode).

Fig. 19-58 Timing diagram of Slave receive (single receive mode) (type 1: DAPmn = 0, CKPmn = 0).



Remark m: Cell number (m=0~2) n: Channel number (n=0~3) p: SSPI number (p=00,01,10,11,20,21,30,31)
mn=00~03,10~11, 20~21

Figure 19-59 Flowchart of Slave receive (single receive mode).



19.5.6 Slave sending and receiving

Slave transmit and receive 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	SSPI30	SSPI31
Object channels	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1	SCI2 Channel 0	SCI2 Channel 1
The pins used	SCLK00, SDI00, SDO00	SCLK01, SDI01, SDO01	SCLK10, SDI10, SDO10	SCLK11, SDI11, SDO11	SCLK20, SDI20, SDO20	SCLK21, SDI21, SDO21	SCLK30, SDI30, SDO30	SCLK31, SDI31, SDO31
interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21	INTSSPI30	INTSSPI31
Error detection flags	There are only overflow error detection flags (OVFmn).							
The length of the transferred data	SCI0: 7 or 8 bits SCI1/SCI2: 7 to 16 digits							
Transfer rate	Max. $f_{MCK}/6[Hz]$ ^{Note 1, 2}							
Data phase	It can be selected by the DAPmn bit of the SCRmn register. •DAPmn=0: Starts data input/output when the serial clock starts running. •DAPmn=1: Starts data input/output half a clock before the serial clock starts running.							
Clock phase	It can be selected by the CKPmn bit of the SCRmn register. • CKPmn=0: Normal phase • CKPmn=1: Inverted							
Data direction	MSB priority or LSB priority							

Note 1 Because internally to SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, SCLK21, SCLK30, SCLK31 pin input of the external serial clock for sampling after sampling, so the maximum transmission rate $f_{MCK}/6[Hz]$.

2. Must be used within the scope of peripheral functional characteristics (refer to data sheet) that meet this condition and meet the electrical characteristics.

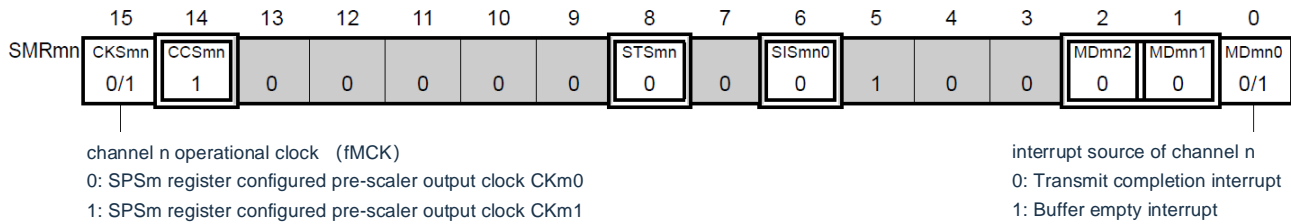
Note 1. f_{MCK} : The operating clock frequency of the object channel

2.m:unit number(m=0~2)n:channel number(n=0~3)mn=00~03, 10~11, 20~21

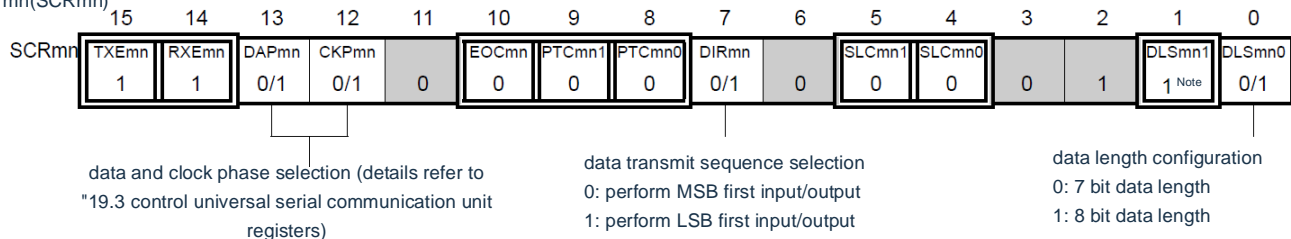
(1) Register settings

Fig19-60 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21, SSPI30, SSPI31)
Example of register settings for Slave transmit and receive

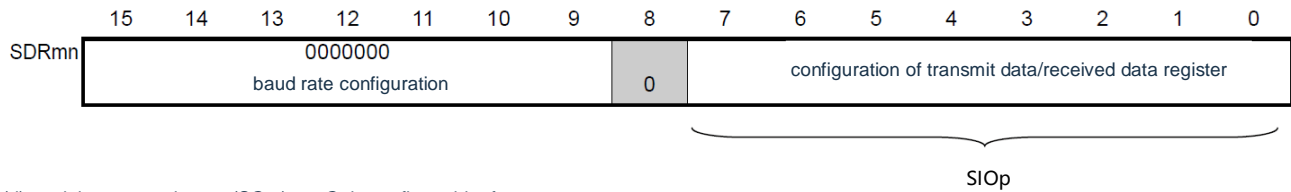
(a) serial mode register mn (SMRmn)



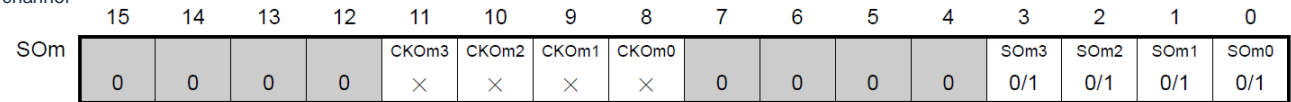
(b) serial communication operation configuration register mn (SCRmn)



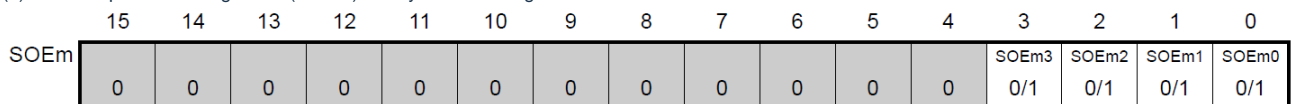
(c) serial data register mn (SDRmn) (low 8 bit: SIOp)



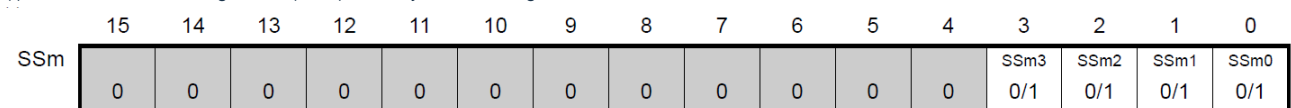
(d) serial output register m(SOm)Only configure bit of target channel



(e) serial output enable register m(SOEm).....only set bit of target channel to 1.



(f) serial channel start register m(SSm) Only set bit of target channel to 1.



Note This example is the setting method for SCI0. The data length of SCI1/SCI2 and the serial data register SDRmn are set differently from this example.

For data length settings, refer to Chapter 19.3.4Serial communication runs the set register mn (SCRmn).Chapter 17).

For the setting method of serial data register SDRmn, refer to "19.3.6Serial data register mn(SDRmn) (SCI1/SCI2 i. e. m=1/2).

Note that the SIOp register must be set to send data before the master device starts to output the clock.

Remark1.m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)

mn=00~03, 10~11, 20~21

2. : Fixed setting in SSPI Slave transmit and receive modes. : Cannot be set (initial value is set).
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 19-61 Initial setup steps for slave sending and receiving



Note that the SIOp register must be set to send data before the master device starts to output the clock.

Figure 19-62 Abort steps for slave sending and receiving

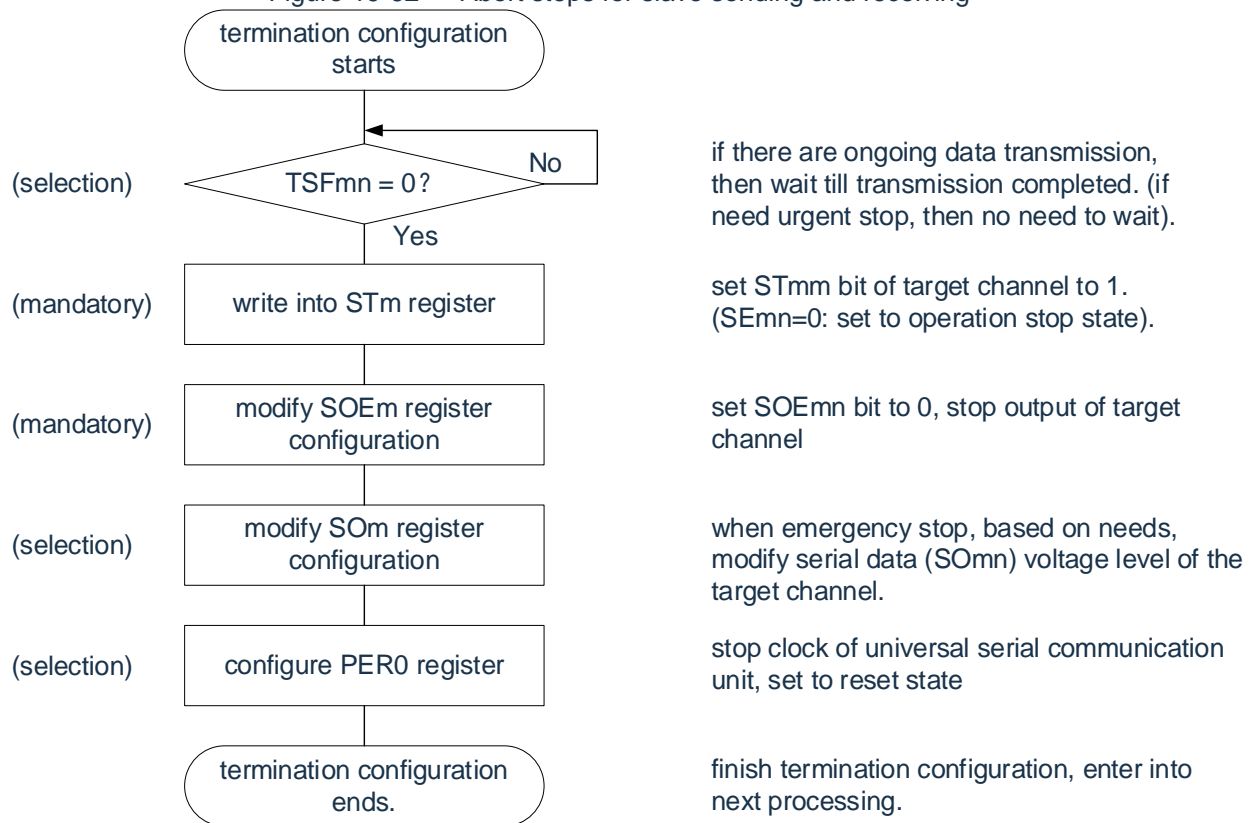
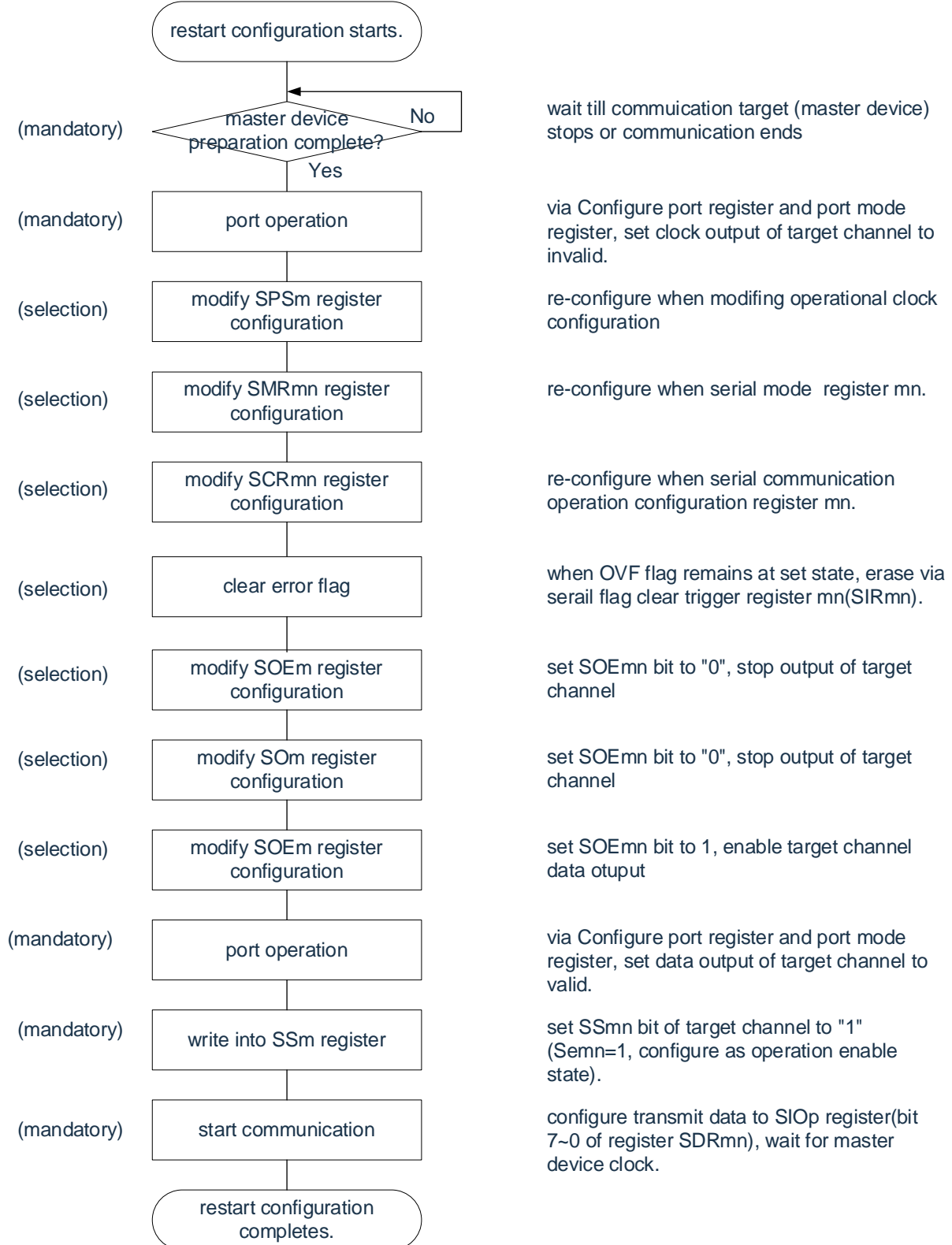


Figure 19-63 restarts the setup steps for Slave sending and receiving

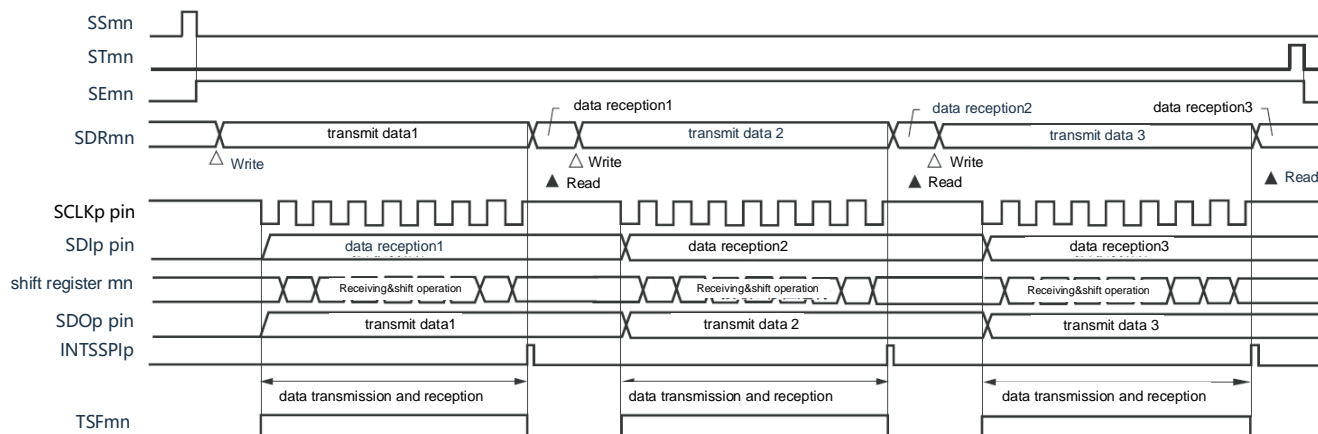


Note 1 Before the master device can start outputting the clock, data must be sent to the SIOp register settings.

2. If PER0 is rewritten in the abort setting to stop providing the clock, it is necessary to make the initial setting instead of restarting the setting when the communication object (the master device) stops or after the communication ends.

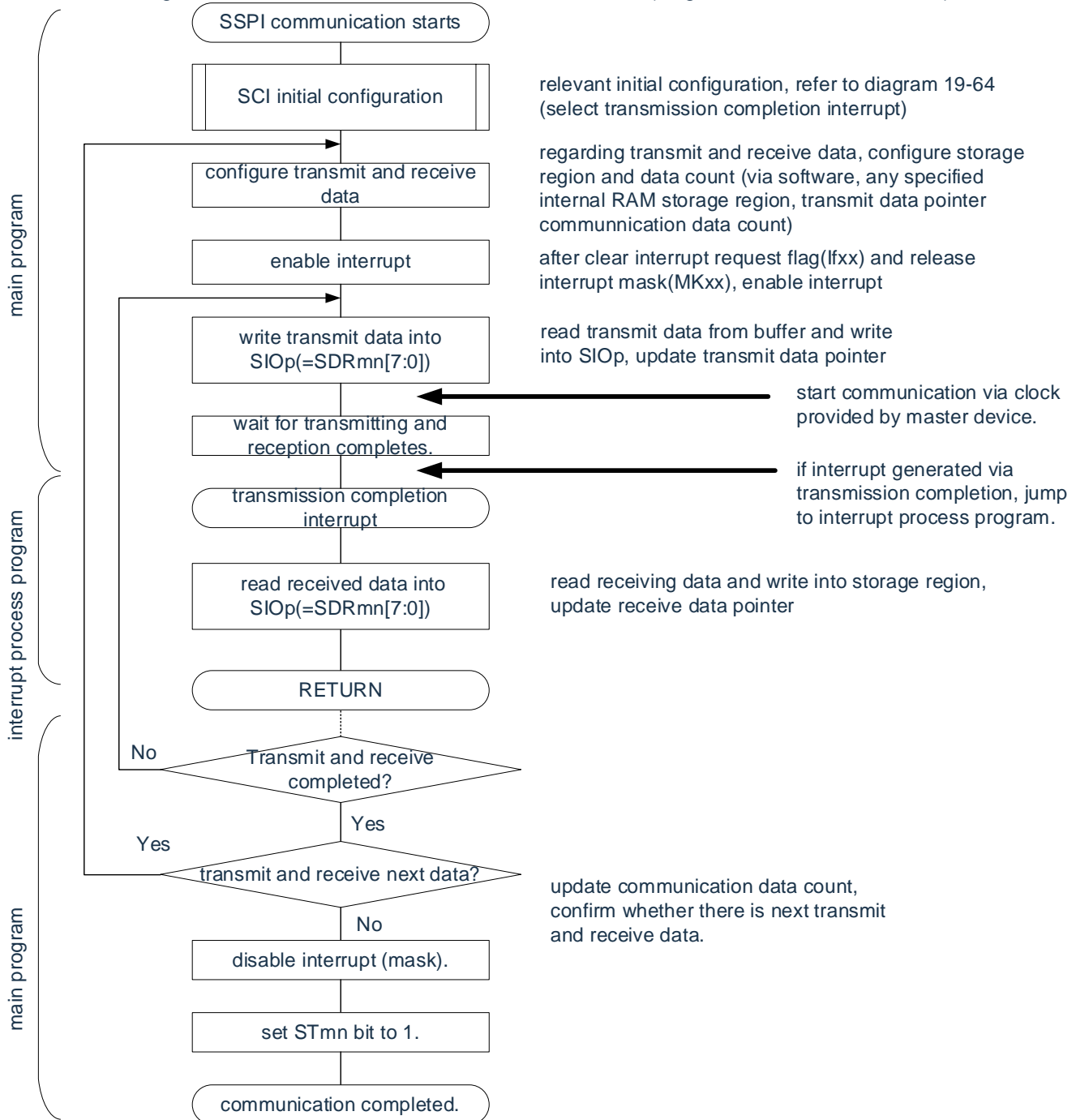
(3) Process flow (single send and receive mode).

Fig. 19-64: Timing diagram of Slave transmit and receive (single transmit and receive mode)
(type 1: DAPmn= 0, CKPmn = 0).



Remark m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)
mn=00~03, 10~11, 20~21

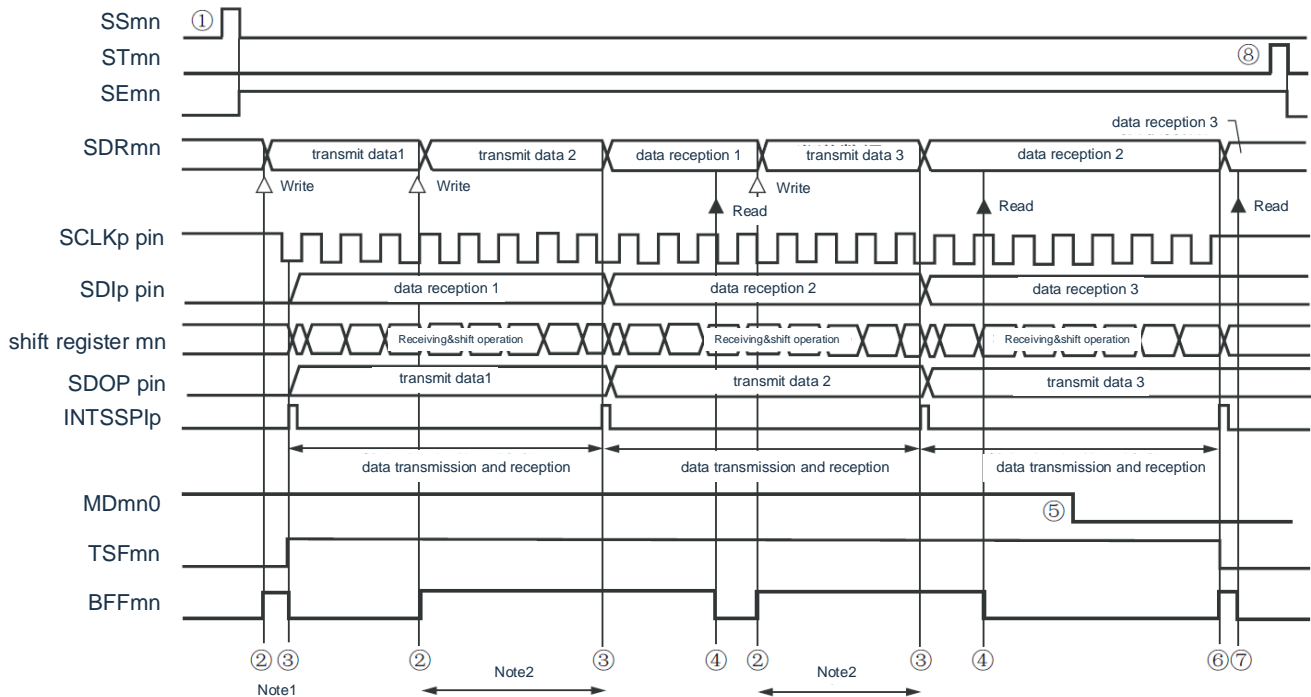
Figure 19-65 Flowchart of slave send and receive (single send and receive mode).



Note that the 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).

Fig. 19-66 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.

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.

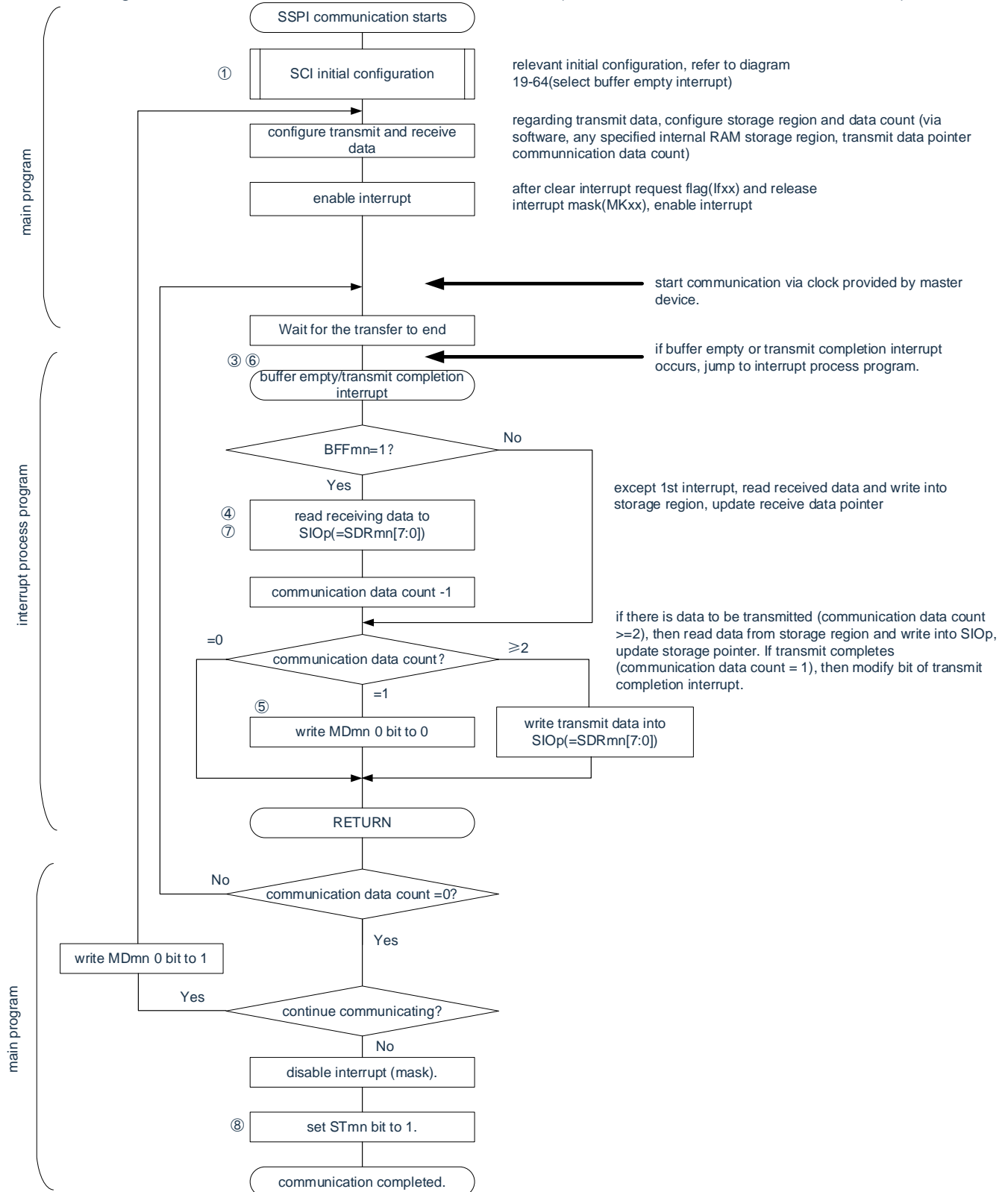
Note that the MDmn0 bit of the serial mode register mn (SMRmn) can be rewritten even in operation. However, in order to catch the end-of-transmission interruption of the last sent data, it must be rewritten before the last bit of transmission begins.

Note 1 (1) ~ (8) in the figure corresponds to (1) ~ (8) in "Fig. 19-67 Flowchart of Slave send and receive (continuous transmit and receive mode)".

2.m: unit number (m=0~2) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)

mn=00~03, 10~11, 20~21

Fig. 19-67: Flowchart of Slave transmit and receive (continuous transmit and receive mode).



Note that the SIOp register must be set to send data before the master device starts to output the clock.

Note (1)~(8) in the note figure corresponds to (1)~(8) in "Fig. 19-66 Sequential Send and Receive (Sequential Transmit and Receive Mode)".

19.5.7 Calculate the transmit clock frequency

3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI20, SSPI20, SSPI21, SSPI30, SSPI31) communication transmission clock frequency can be calculated using the following calculation equation.

(1) Master device

$$(\text{Transmit clock frequency}) = \{\text{Object Channel's operating clock (f}_{\text{MCK}}\text{) frequency}\} (\text{SDRmn}[15:9] + 1) \div 2 [\text{Hz}]$$

(2) Slaves

$$(\text{Transmit Clock Frequency}) = \{\text{Serial Clock (SCLK) Frequency Rate Provided by the Master Device}\}^{\text{Note}} [\text{Hz}].$$

Note The maximum allowable transmit clock frequency is $f_{\text{MCK}}/6$.

Note 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.

The operating clock(f_{MCK}) depends on bit15 of the serial clock selection register m (SPSm) and the serial mode register mn (SMRmn). (CKSmn).

Table 19-2 3-wire serial I/O operating clocks

SMRmn register	SPSm register								Running Clock (f_{MCK}) ^{Note}	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		$f_{CLK}=32\text{MHz}$ 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
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 Universal Serial Communication Unit (SCI) (serial channel stop register m (STm)=000FH) after making changes.

Note 1.X: Ignore

2.m:unit number(m=0~2)n:channel number(n=0~3)mn=00~03, 10~11, 20~21

19.5.8 In 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21, SSPI30, SSPI31) Processing steps when an error occurs during communication

In 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21, SSPI30, SSPI31) The processing steps when an error occurs during communication, such as Figure 19-68 shows.

Figure 19-68 processing steps when an overflow error occurs

software operation	Hardware Status	Comments
Read the serial data →	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.

Note m: Unit number (m=0~2)n: Channel number (n=0~3)mn=00~ 03, 10~11, 20~21

19.6 The operation of the clock synchronization serial communication of the slave selection input function

Channel 0 of SCI0 is a channel that supports clock synchronization serial communication with the Slave select input function.

[Sending and receiving data].

- 7-bit or 8-bit data length (SCI0).
Data length from 7 to 16 bits (SCI1/SCI2).
- Phase control of sending and receiving data
- MSB/LSB preferred
- Level setting for sending and receiving data

[Clock Control].

- Phase control of input/output clocks
- Set the transmission period generated by the prescaler and the internal counter of the channel.
- Maximum transfer rate ^{note} Slave communication: $\text{Max.f}_{\text{MCK}}/6$

[Interrupt function].

- Transmit end interrupt, buffer empty interrupt

[Error Detection Flag].

- Overflow error

Note must be used within the range that satisfies the SCLK Cycle Time (t_{KCY}) characteristics. Please refer to the data sheet for details.

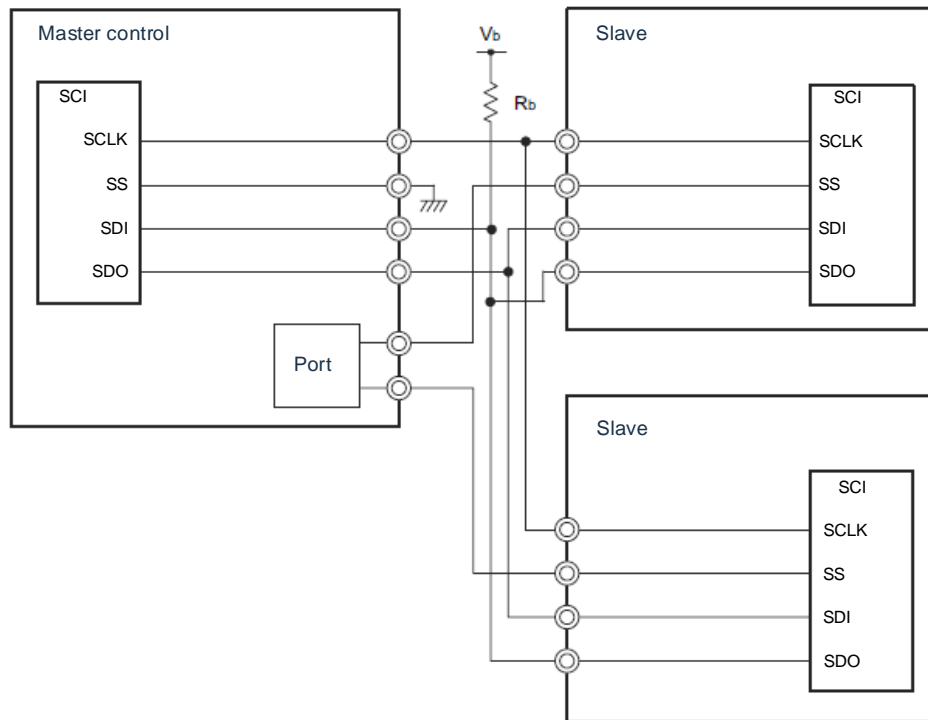
The slave selection input function has the following 3 kinds of communication operation:

- Slave send (see 19.6.1).
- Slave reception (cf.19.6.2).
- Slave sending and receiving (cf.19.6.3).

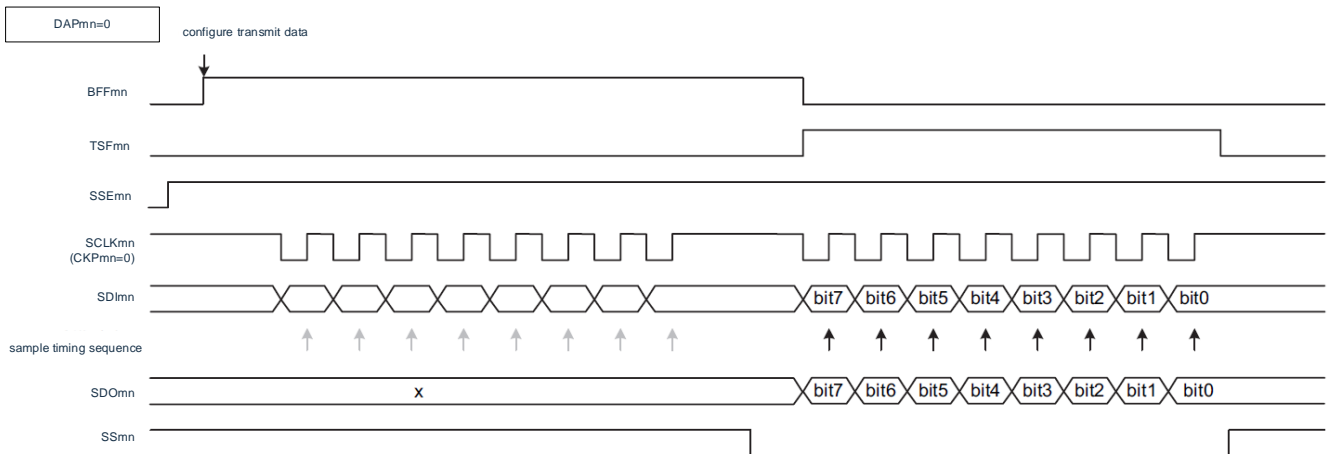
By using the slave selection input function, a master device can be connected to multiple slave devices for communication. The master device performs the output of the slave selection signal of the slave device (1) of the communication object, and each slave device determines whether it is selected as the communication object and controls the output of the SDO pin. When a slave device is selected as a communication object, the SDO pin can communicate data to the master device; When a slave is not selected as the communication object, the SDO pin becomes a high-level output, so in an environment where multiple slaves are connected, the SDO pin needs to be set to Nch-O.D and the node pulled up. In addition, even the serial clock entered into the master device is not transmitted and received.

Note the Slave selection signal must be output through the operation of the port.

Fig. 19-69 Structural example of the Slave selection input function

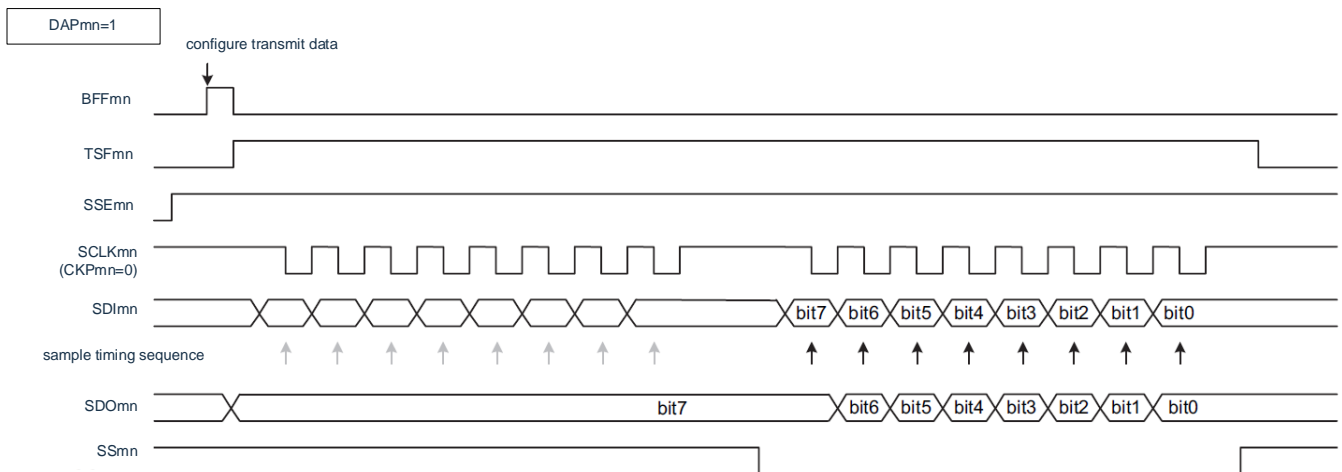


Note that the SDO00 pin is selected as the N-channel open-drain output mode.

Figure 19-70 Timing diagram of the Slave selection input function


During periods when SSmn is high, no transmission is made even on the falling edge of THECKmn (serial clock), and no sampling of received data synchronized with the rising edge is performed.

During the SSmn low level, the output data is synchronized (shifted) with the falling edge of the serial clock and the 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.

Note m: Unit number (m=0) n: Channel number (n=0).

19.6.1 Slave sending

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	SSPI00
Object channels	Channel 0 for SCIO
The pins used	SCLK00, SDO00, SS00
interrupt	INTSSPI00 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).
The length of the transferred data	7 or 8 bits
Transfer rate	$\text{Max. } f_{\text{MCK}}/6[\text{Hz}]^{\text{note1, 2}}$
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: Normal phase • CKPmn=1: Inverted
Data direction	MSB priority or LSB priority
Slave selection input function	You can select the run 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 $f_{\text{MCK}}/6$ [Hz].

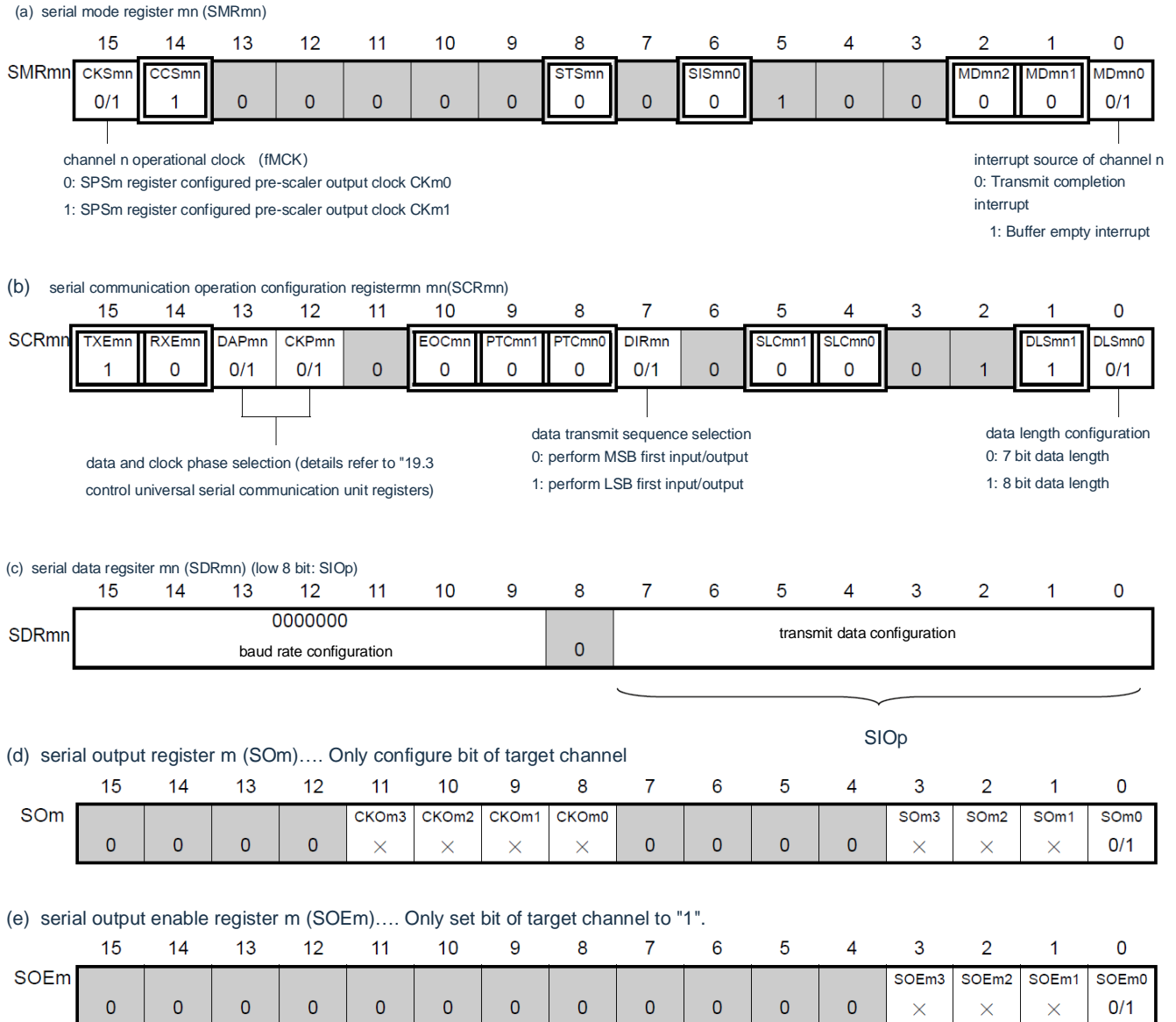
- Must be used within the scope of peripheral functional characteristics (refer to data sheet) that meet this condition and meet the electrical characteristics.

Note 1. f_{MCK} : The operating clock frequency of the object channel

2.m: unit number (m=0)n: channel number (n=0).

(1) Register settings

Figure 19-71 Slave Selection Input Function (SSPI00) Example of register setting content when slave sending (1/2).



Note 1.m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

- : Fixed setting in SSPI Slave send mode.
 : Cannot be set (initial value is set).
 : 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 19-72 Slave Selection Input Function (SSPI00) Example of Register Setting Content when Slave Sends (2/2).

(f) serial channel start register m (SSm) Only set bit of target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	SSm3 ×	SSm2 ×	SSm1 ×	SSm0 0/1

(g) input switch control register (ISC).... This is controlled by SS00 pin of SSPI00 slave channel (channel 0 of unit 0).

	7	6	5	4	3	2	1	0
ISC	SSIE00 0/1	0	0	0	0	0	ISC1 0/1	ISC0 0/1

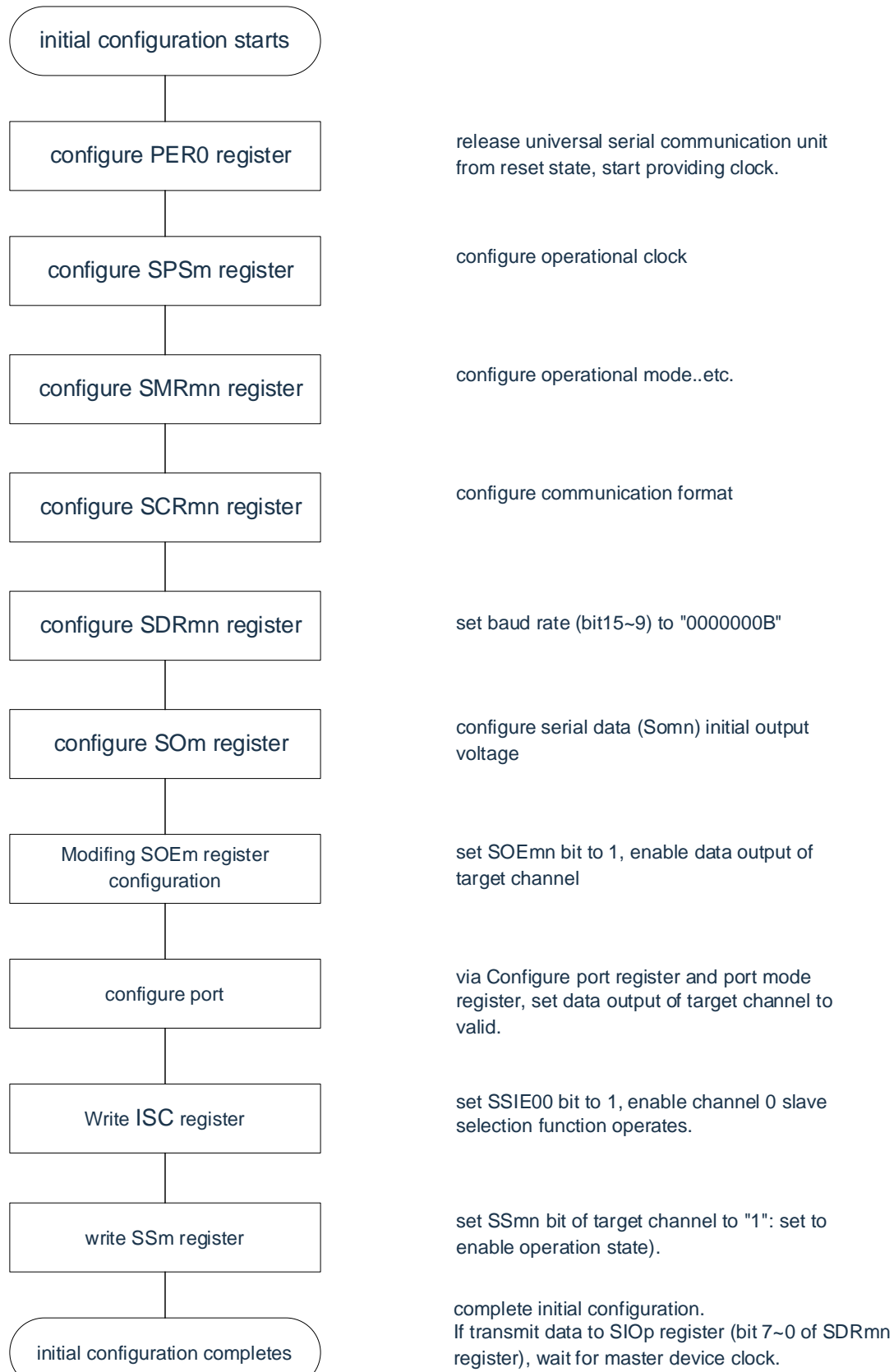
0: SS00 pin input invalid
1: SS00 pin input valid

Note 1.m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

2. ☐ : Fixed setting in SSPI Slave send mode. ☐ : Cannot be set (initial value is set).
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.

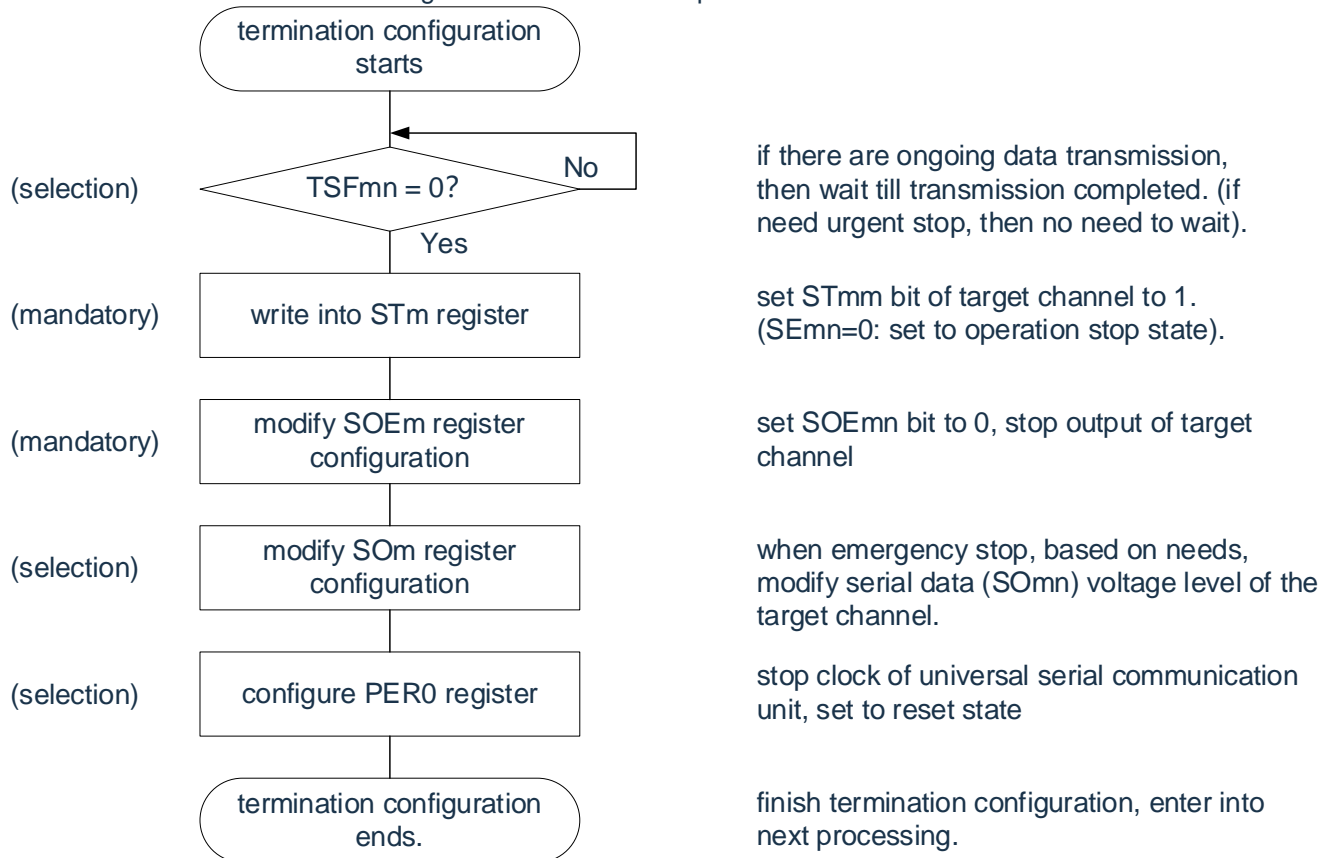
(2) Procedure

Figure 19-72 Initial setup steps for slave sending



Note m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

Figure 19-73 Abort step of the slave send

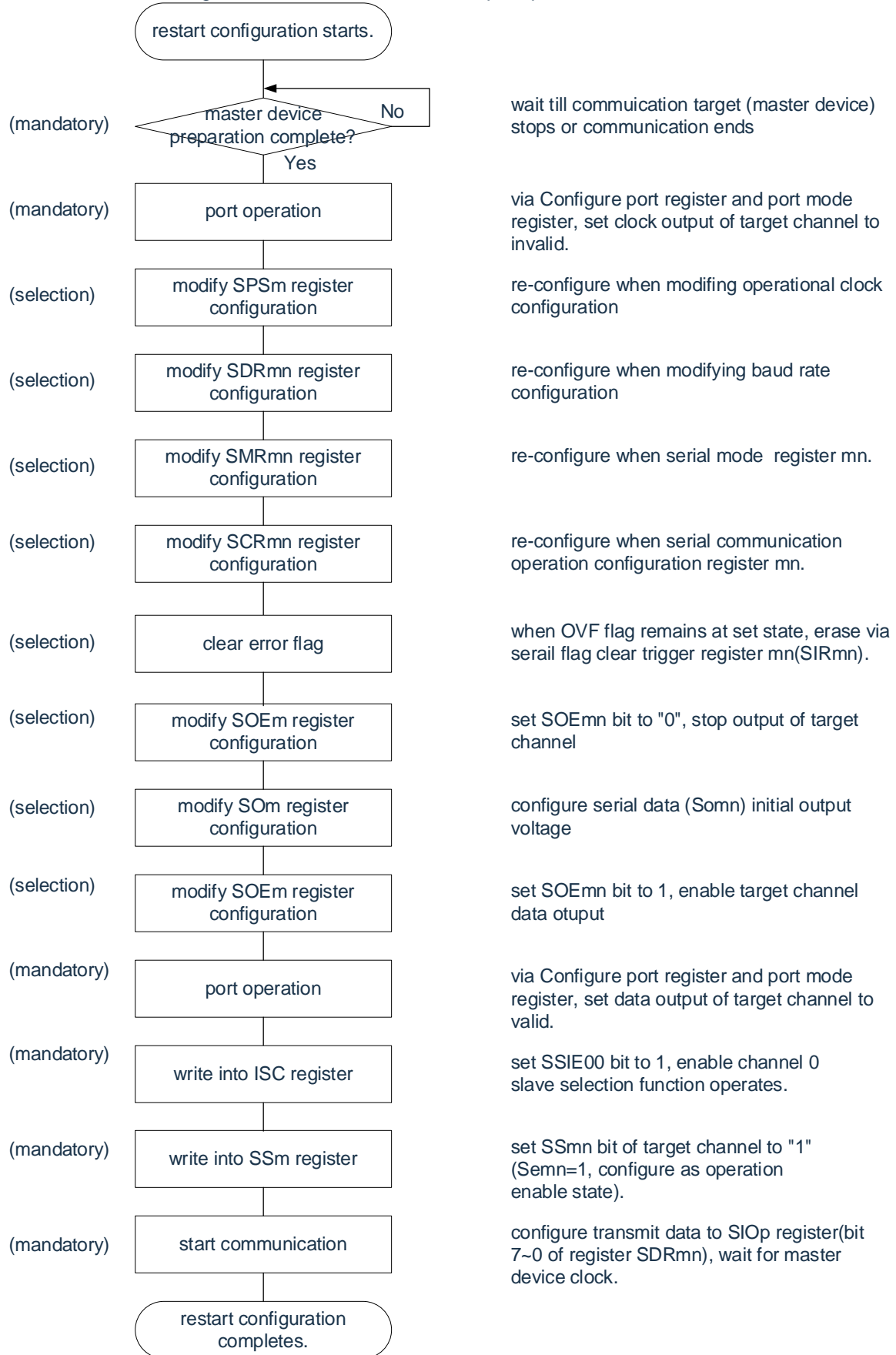


Note m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

Note 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.

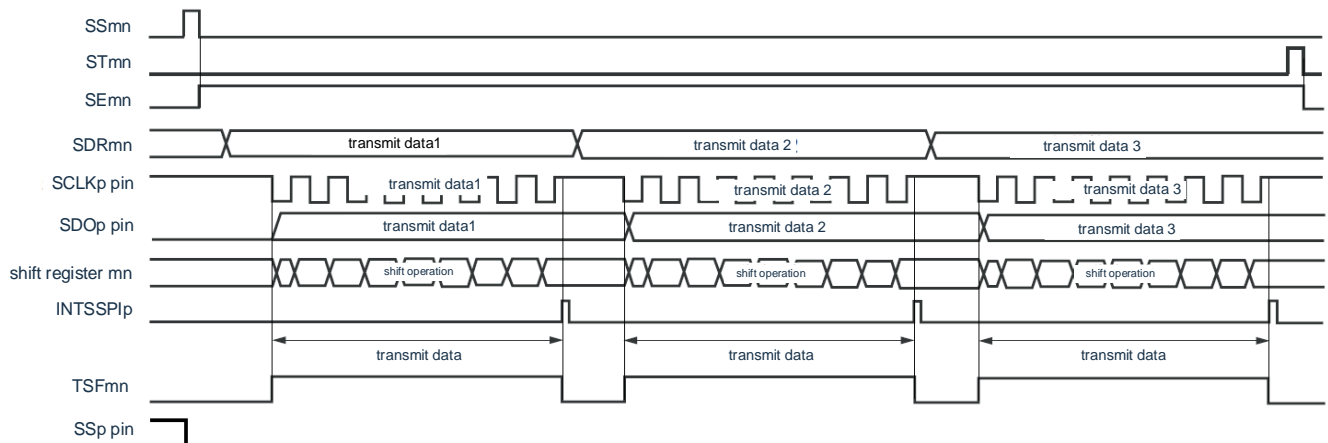
2.m: unit number (m=0)n: channel number (n=0)p: SSPI number (p=00).

Figure 19-74 Restarts the setup step of the Slave send



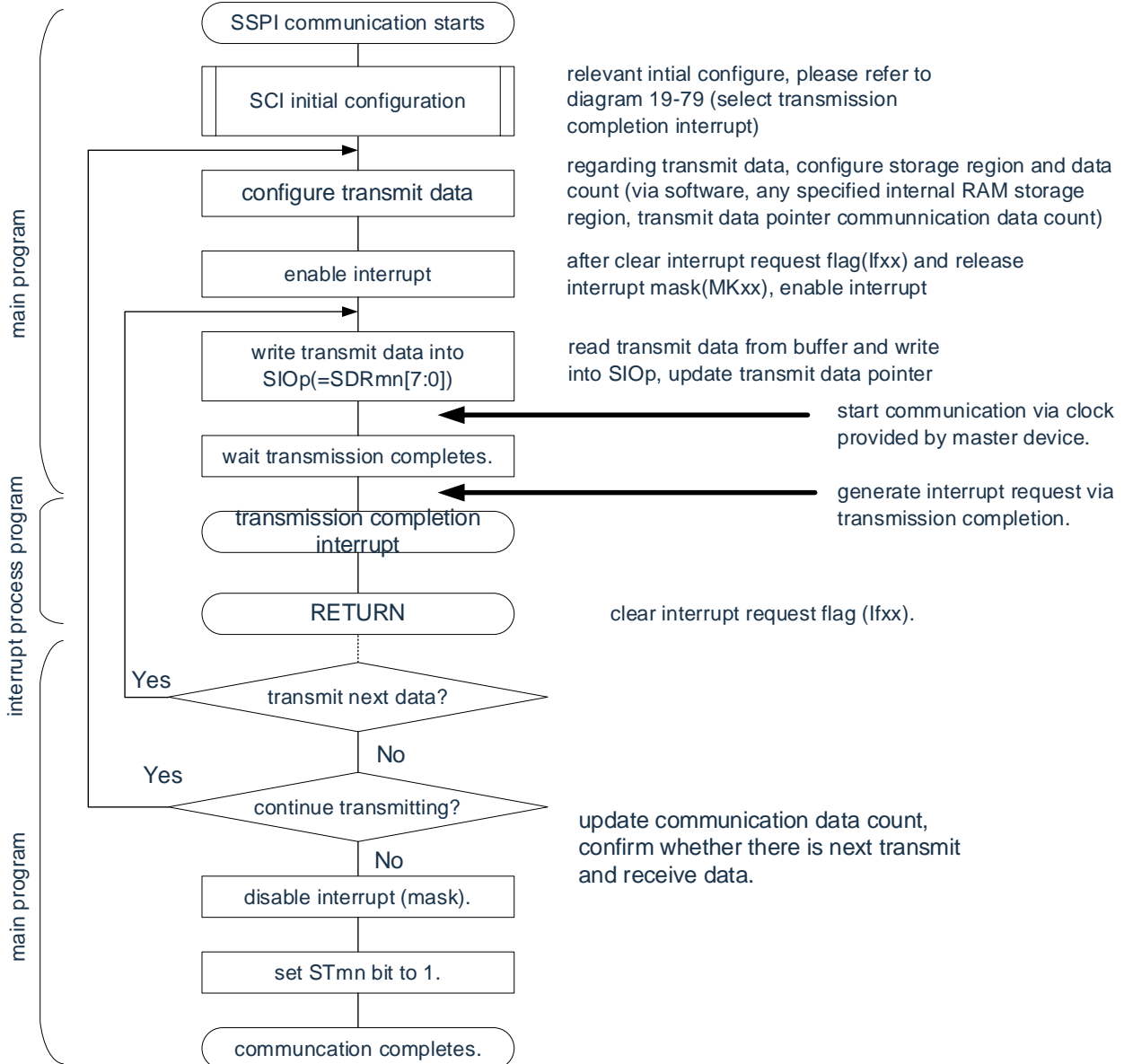
(3) Process flow (single send mode).

Fig. 19-75 Timing diagram of a Slave send (single send mode) (type 1: DAPmn=0, CKPmn=0).



Note m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

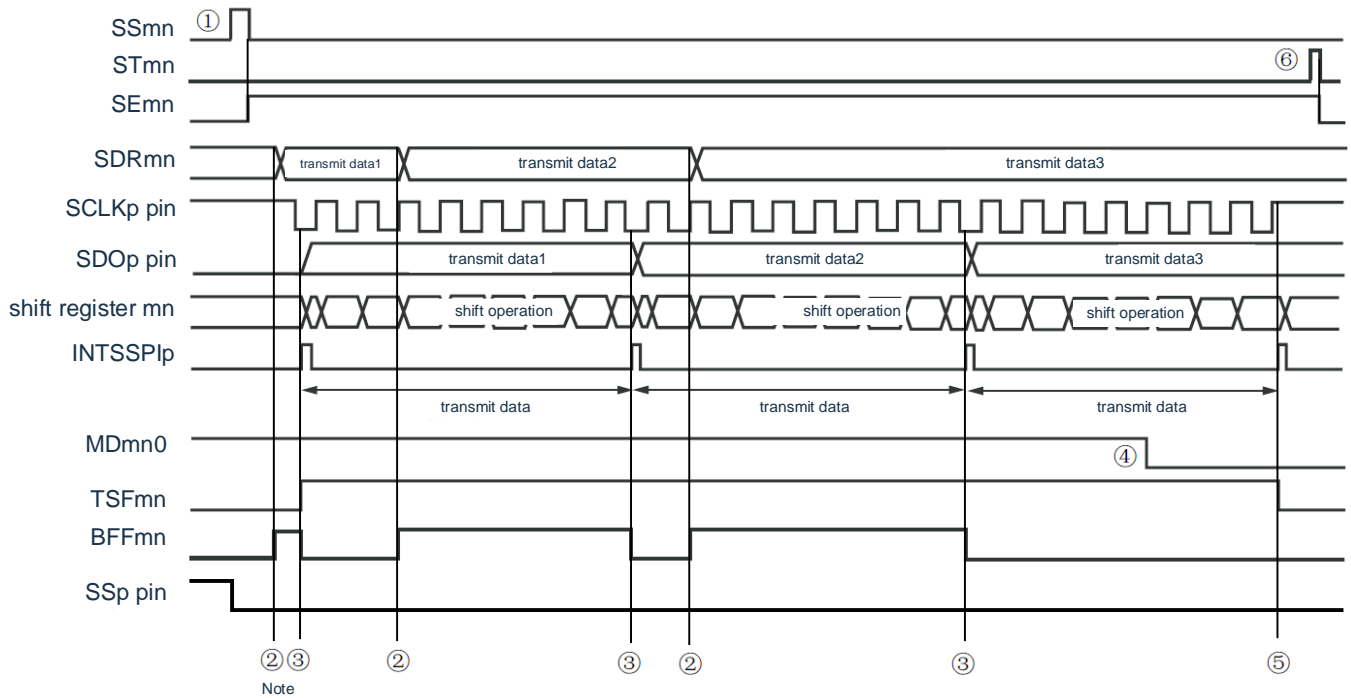
Figure 19-76 Flowchart of Slave send (single send mode).



Note m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

(4) Process flow (continuous send mode).

Figure 19-77 Timing diagram of Slave send (continuous send mode) (type 1: DAPmn= 0, CKPmn = 0).

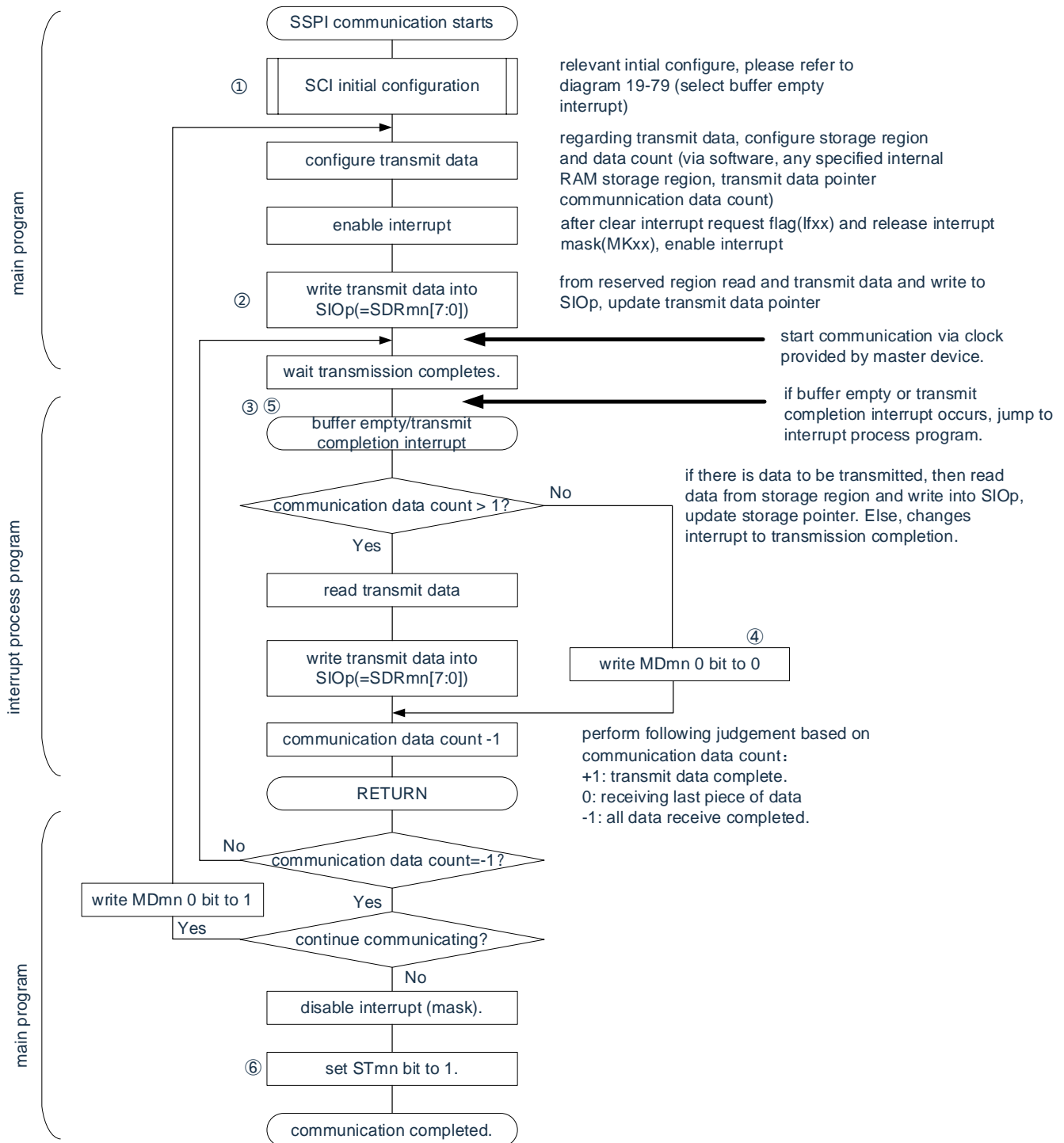


Note If the BFFmn bit of the serial status register mn (SSRmn) is "1" (when valid data is saved in the serial data register mn (SDRmn)) is given When the SDRmn register writes the transmit data, it rewrites the send data.

Note that the 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.

Note m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

Figure 19-78 Flowchart of Slave send (continuous send mode).



Note 1 (1) ~ (6) in the figure corresponds to (1) ~ (6) in Figure 19-77 Time Series Diagram of Slave Sending (Continuous Send Mode)".

2.m: unit number (m=0)n: channel number (n=0)p: SSPI number (p=00).

19.6.2 Slave receive

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	SSPI00
Object channels	Channel 0 for SCIO
The pins used	SCLK00, SDI00, SS00
interrupt	INTSSPI00 Limited to end-of-transmit interrupts (buffer null interrupts are prohibited).
Error detection flags	There are only overflow error detection flags (OVFmn).
The length of the transferred data	7 or 8 bits
Transfer rate	$\text{Max}f_{\text{MCK}}/6[\text{Hz}]^{\text{note1, 2}}$
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: Normal phase • CKPmn=1: Inverted
Data direction	MSB priority or LSB priority
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 $f_{\text{MCK}}/6$ [Hz].

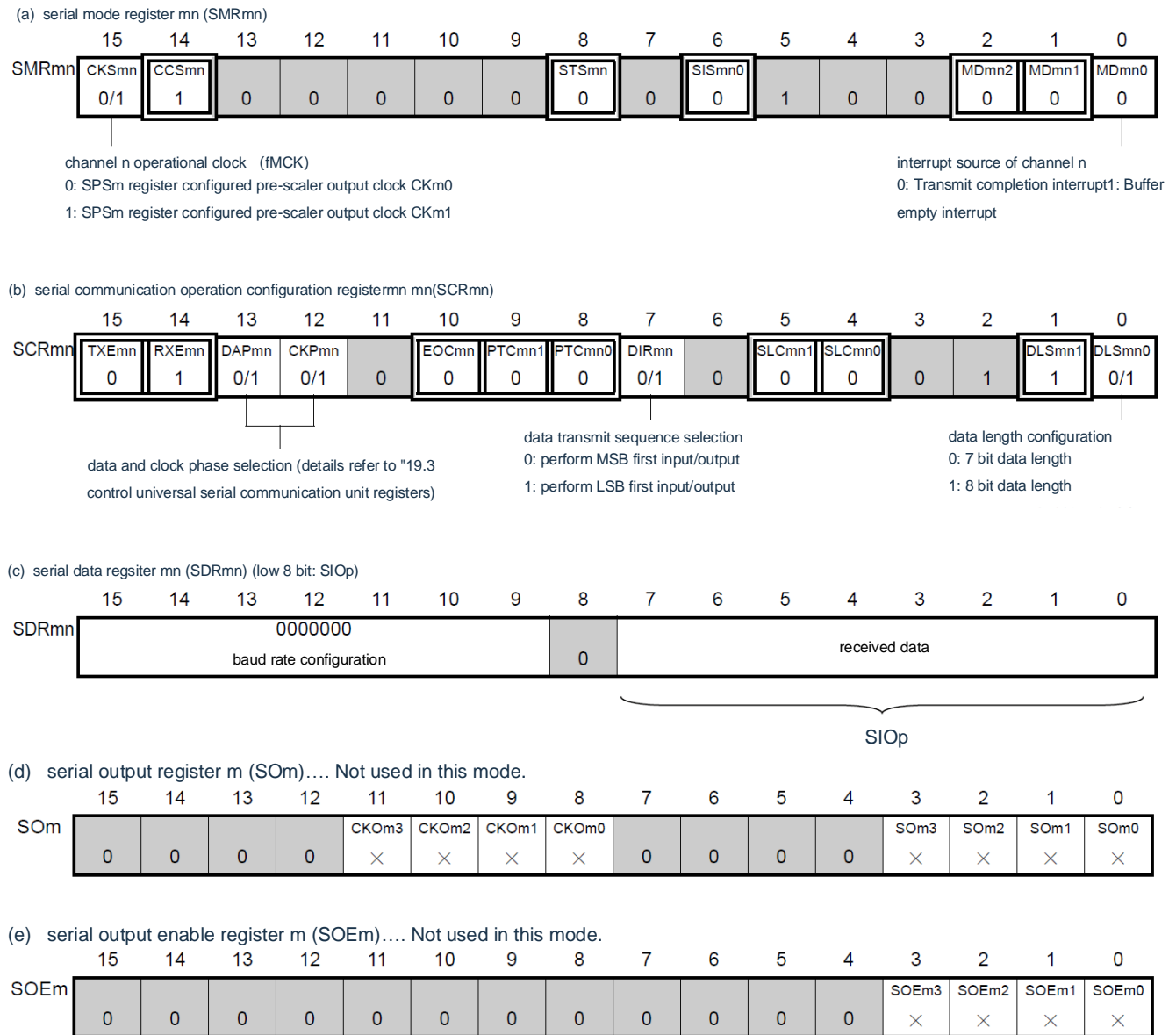
- Must be used within the scope of peripheral functional characteristics (refer to data sheet) that meet this condition and meet the electrical characteristics.

Note 1. f_{MCK} : The operating clock frequency of the object channel

2.m: unit number (m=0)n: channel number (n=0).

(1) Register settings

Figure 19-79 Slave Selection Input Function (SSPI00) Example of register setting content when slave receives (1/2).



Note 1.m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

- : Fixed setting in Slave receive mode.
 : Cannot be set (initial value is set).
 : 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).
 : Set "0" or "1" according to the user's purpose.

Figure 19-80 Slave Selection Input Function (SSPI00) Example of Register Setting Content when Slave Receives (2/2).

(f) serial channel start register m (SSm) Only set bit of target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	SSm3 ×	SSm2 ×	SSm1 ×	SSm0 0/1

(g) input switch control register (ISC).... This is controlled by SS00 pin of SSPI00 slave channel (channel 0 of unit 0).

	7	6	5	4	3	2	1	0
ISC	SSIE00 0/1	0	0	0	0	0	ISC1 0/1	ISC0 0/1

0: SS00 pin input invalid
1: SS00 pin input valid

Note 1.m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

- : Fixed setting in Slave receive mode.
 : Cannot be set (initial value is set).
- 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 19-81 Initial setup step of Slave reception

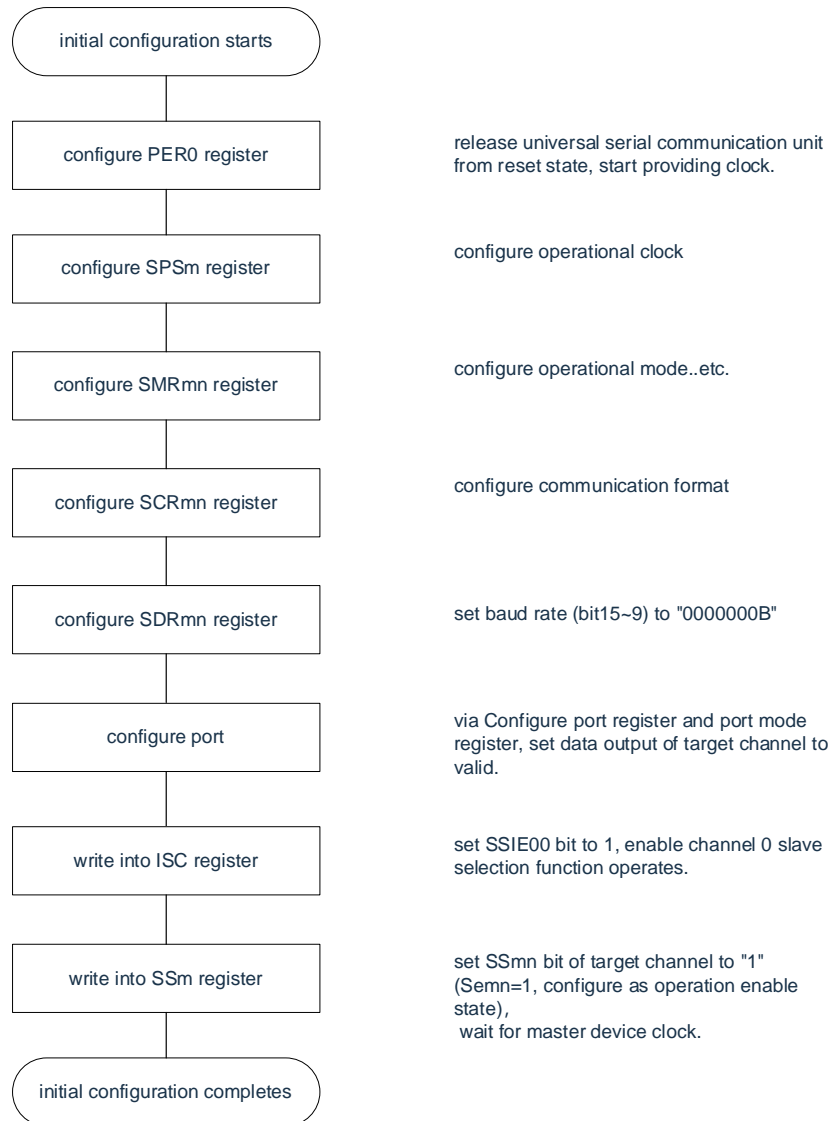
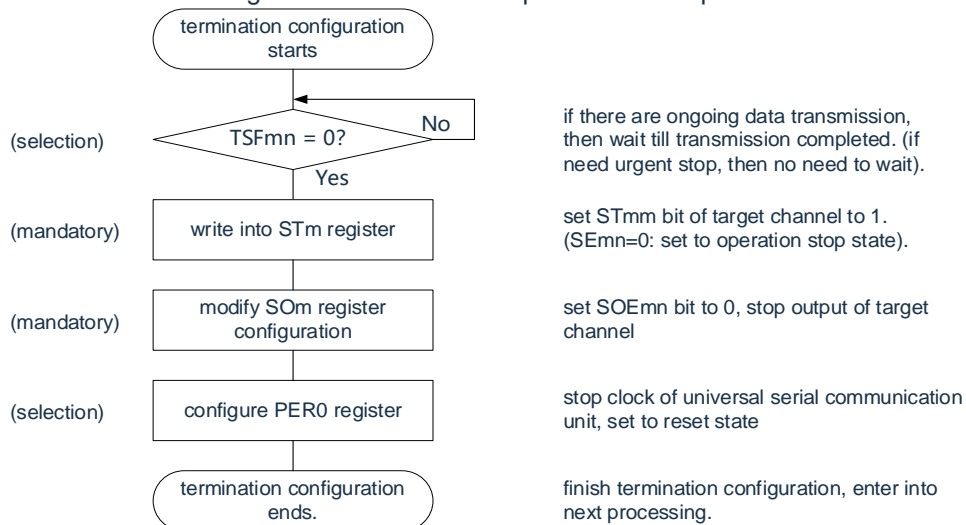
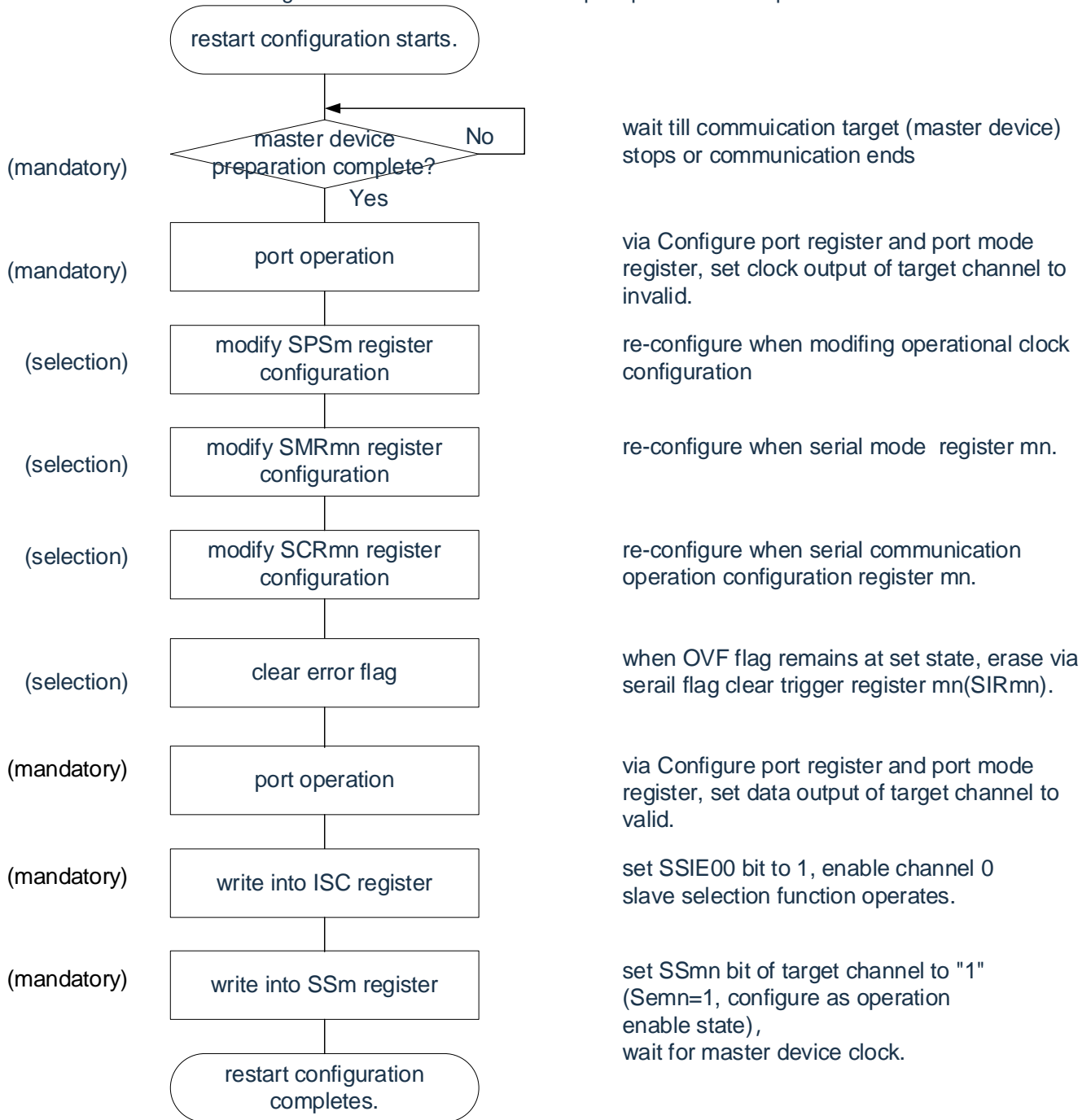


Figure 19-82 Abort step of Slave reception



Note m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

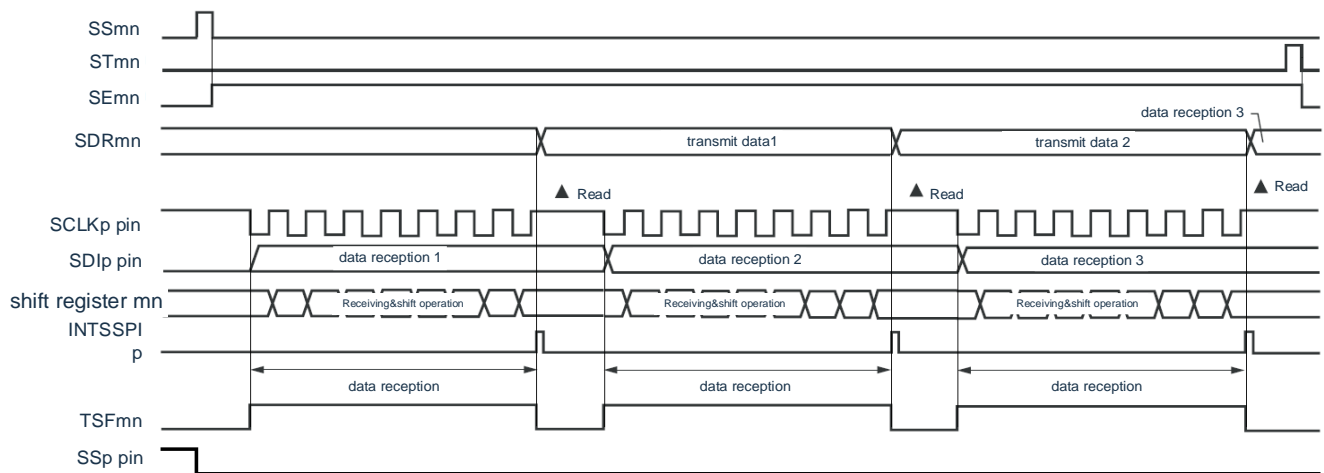
Figure 19-83 Restarts the setup step of Slave reception



Note m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

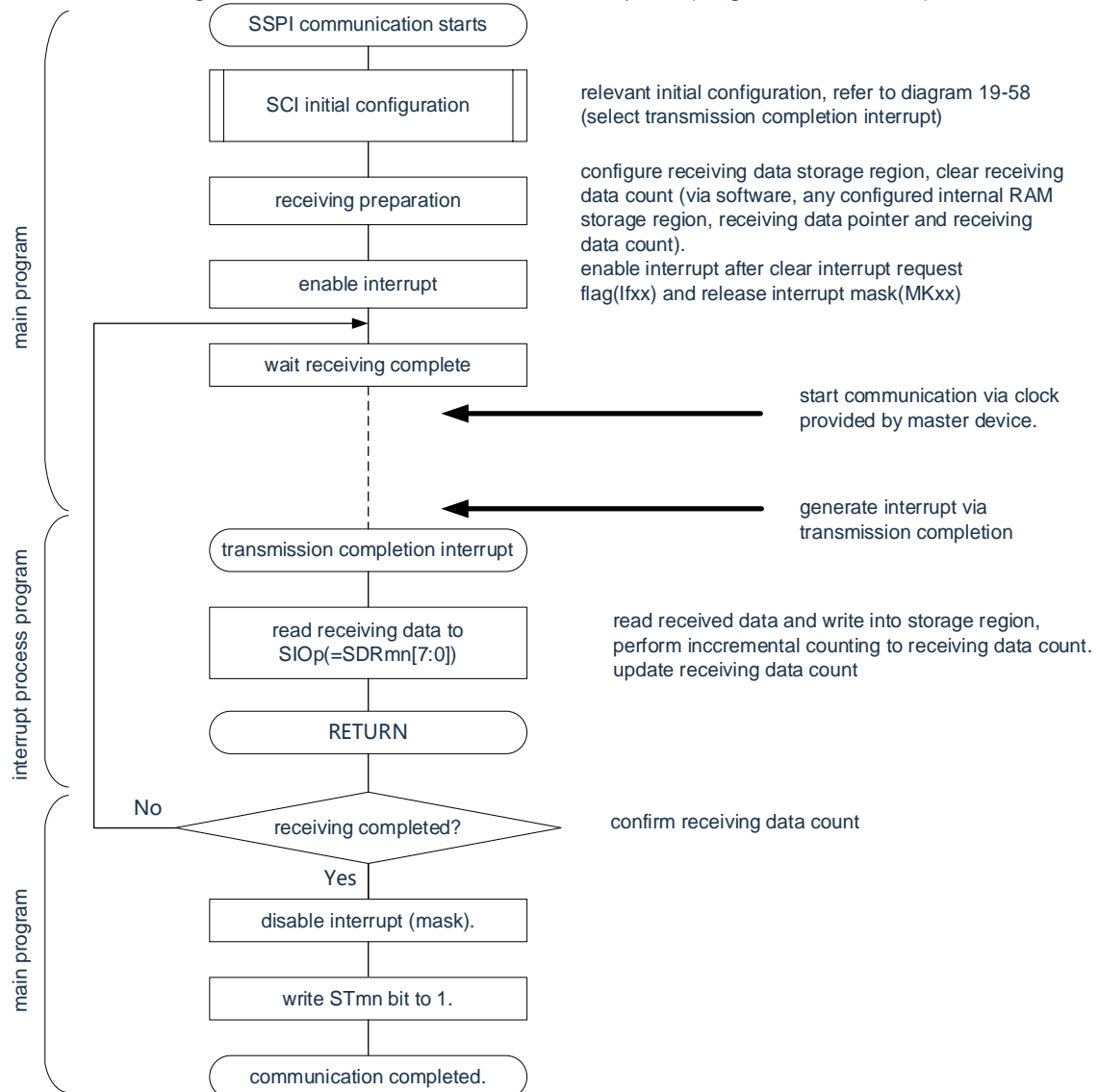
(3) Process flow (single receive mode).

Fig. 19-84 Timing diagram of Slave receive (single receive mode) (type 1: DAPmn=0, CKPmn=0).



Note m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

Figure 19-85 Flowchart of Slave reception (single receive mode).



19.6.3 Slave sending and receiving

Slave transmit and receive 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	SSPI00
Object channels	Channel 0 for SCIO
The pins used	SCLK00, SDI00, SDO00, SS00
interrupt	INTSSPI00 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).
The length of the transferred data	7 or 8 bits
Transfer rate	$\text{Max. } f_{\text{MCK}}/6[\text{Hz}]^{\text{note 1, 2}}$
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: Normal phase • CKPmn=1: Inverted
Data direction	MSB priority or LSB priority
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 $f_{\text{MCK}}/6$ [Hz].

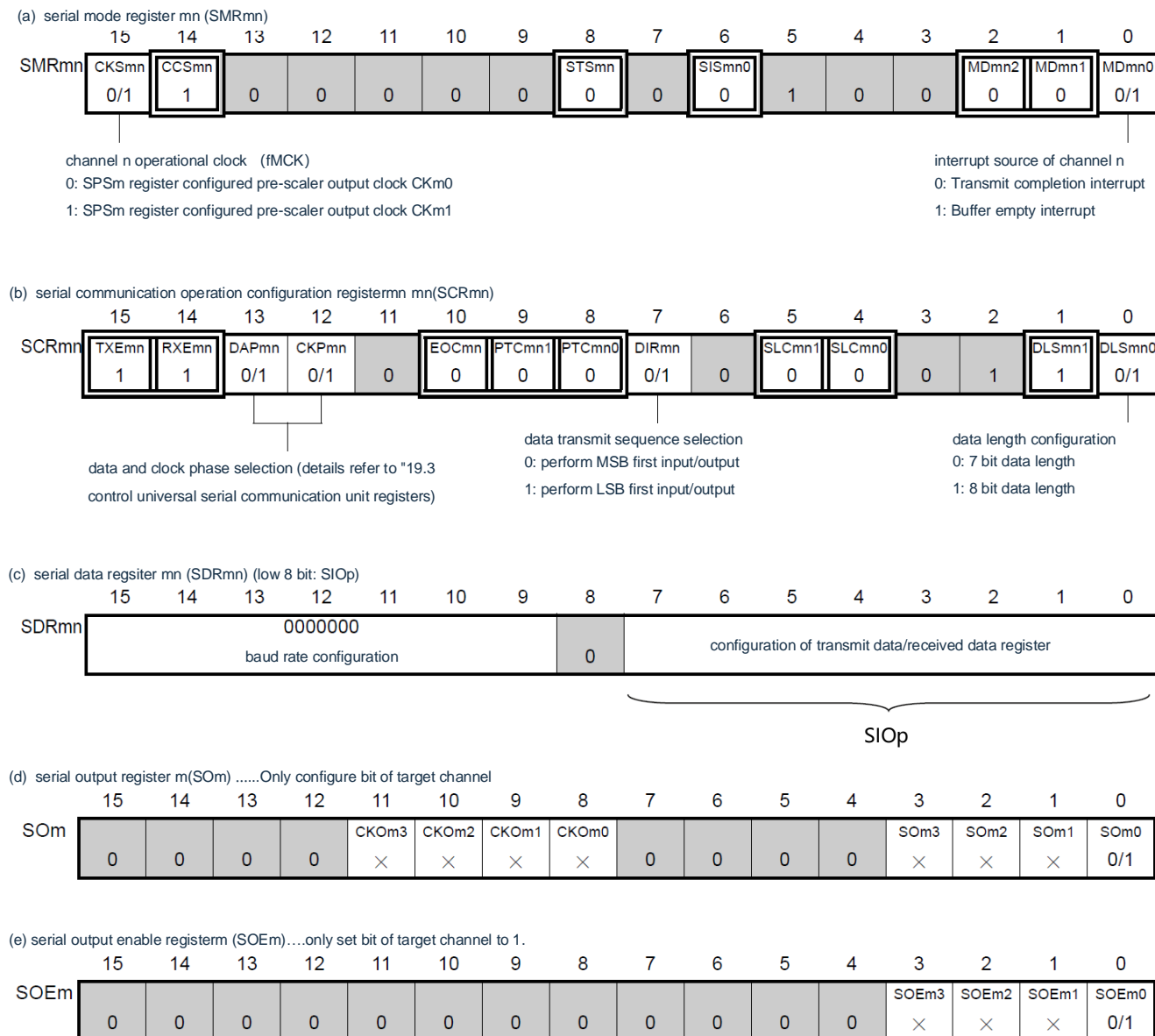
- Must be used within the scope of peripheral functional characteristics (refer to data sheet) that meet this condition and meet the electrical characteristics.

Note 1. f_{MCK} : The operating clock frequency of the object channel

2.m: unit number (m=0)n: channel number (n=0).

(1) Register settings

Figure 19-86 Slave Selection Input Function (SSPI00) Example of register setting content when slave send and receive (1/2).



Note that the SIOp register must be set to send data before the master device starts to output the clock.

Note 1.m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

- : Fixed setting in Slave receive mode.
 : Cannot be set (initial value is set).

 ×: 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 19-87 Slave Selection Input Function (SSPI00) Example of register settings during slave sending and receiving (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													SSm3	SSm2	SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	×	×	×	0/1

(g) input switch control register (ISC).... This is controlled by SS00 pin of SSPI00 slave channel (channel 0 of unit 0).

	7	6	5	4	3	2	1	0
ISC	SSIE00						ISC1	ISC0
	0/1	0	0	0	0	0	0/1	0/1

0: SS00 pin input invalid
1: SS00 pin input valid

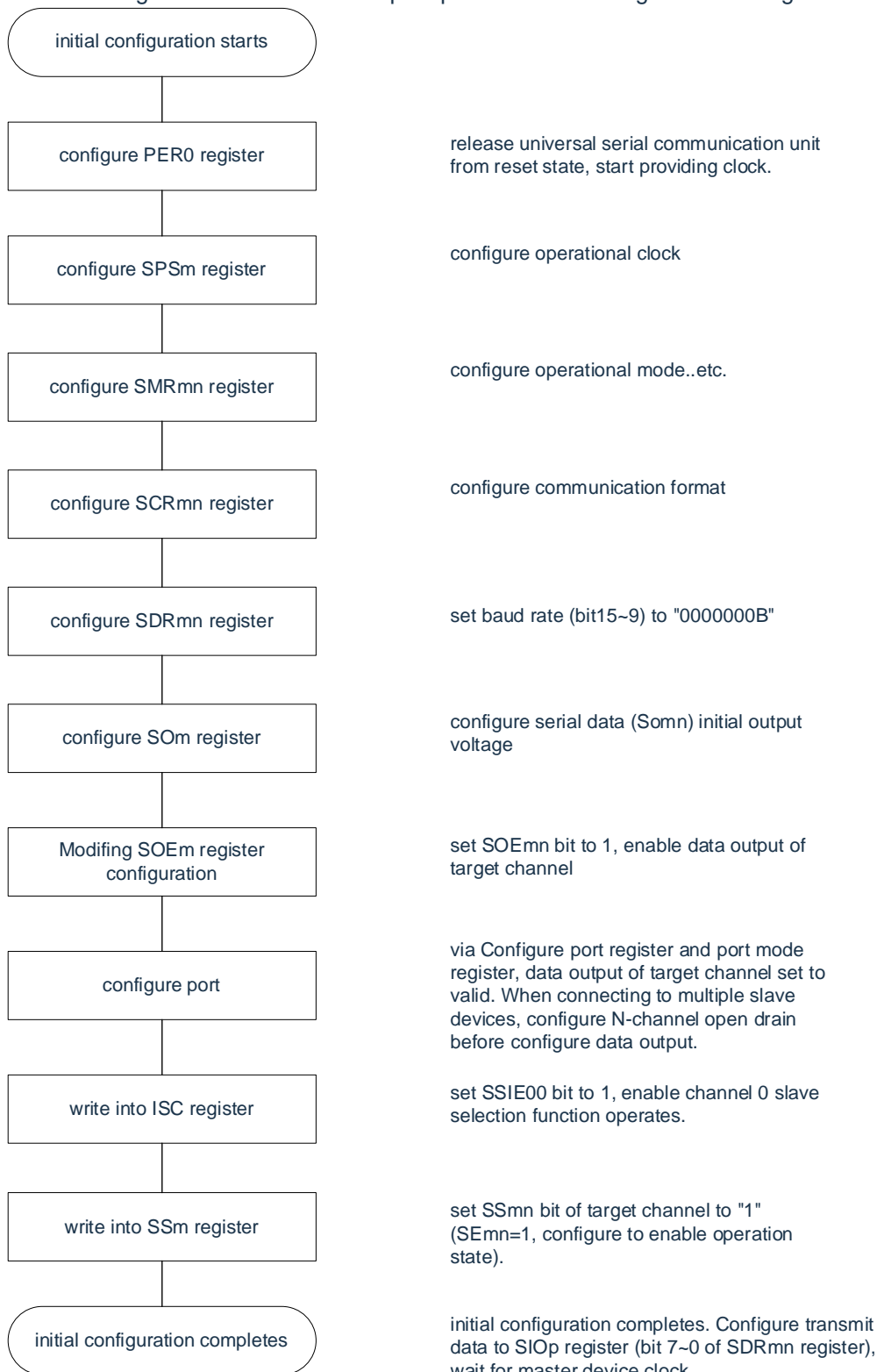
Note that the SIOp register must be set to send data before the master device starts to output the clock.

Note 1.m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

- : Fixed setting in Slave receive mode.
 : Cannot be set (initial value is set).
- 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

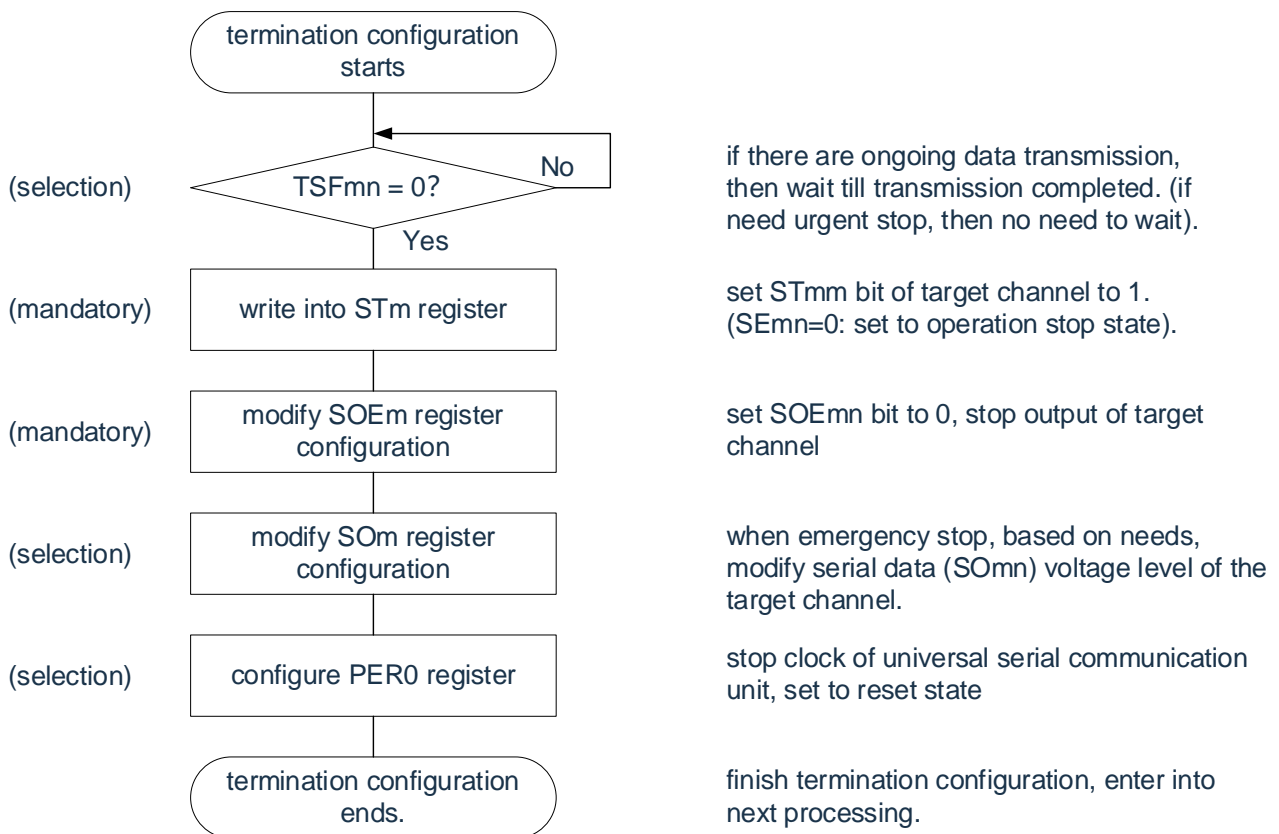
Figure 19-87 Initial setup steps for slave sending and receiving



Note that the SIOp register must be set to send data before the master device starts to output the clock. Note m:

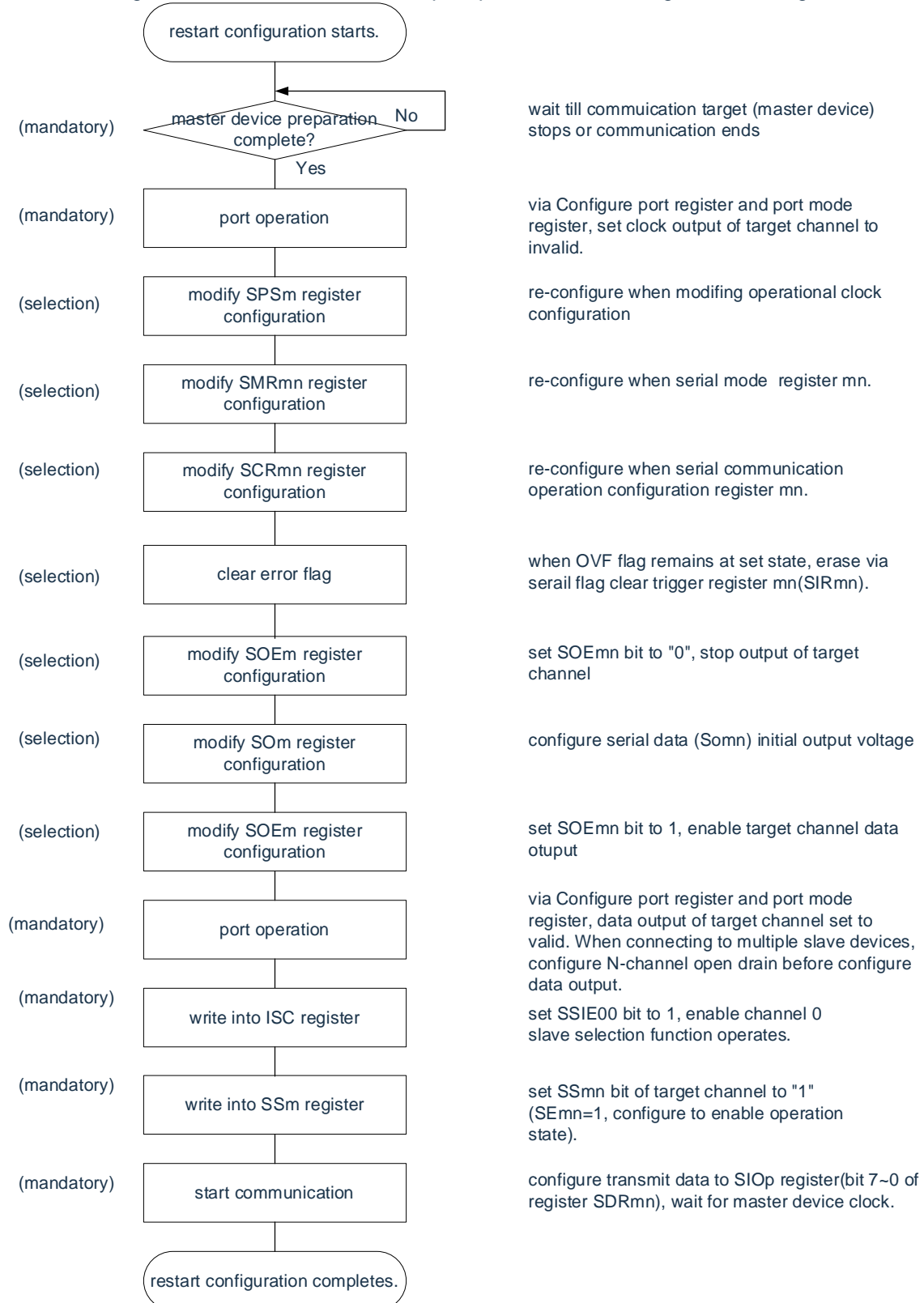
Unit number (m=0)n: Channel number (n=0)p:SSPI number (p=00)

Figure 19-88 Abort steps for slave sending and receiving



Note 1.m: Unit number (m=0) n: Channel number (n=0)p: SSPI number (p=00)

Figure 19-89 Restarts the setup steps for Slave sending and receiving

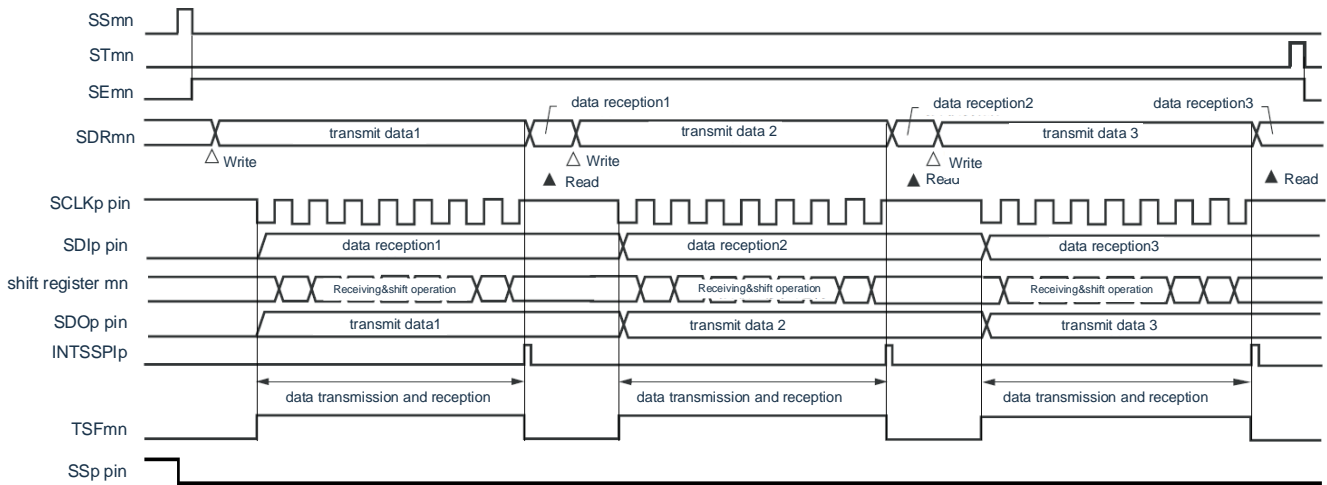


Note 1 Before the master device can start outputting the clock, data must be sent to the SIOp register settings.

2. If PER0 is rewritten in the abort setting to stop providing the clock, it is necessary to make the initial setting instead of restarting the setting when the communication object (the master device) stops or after the communication ends.

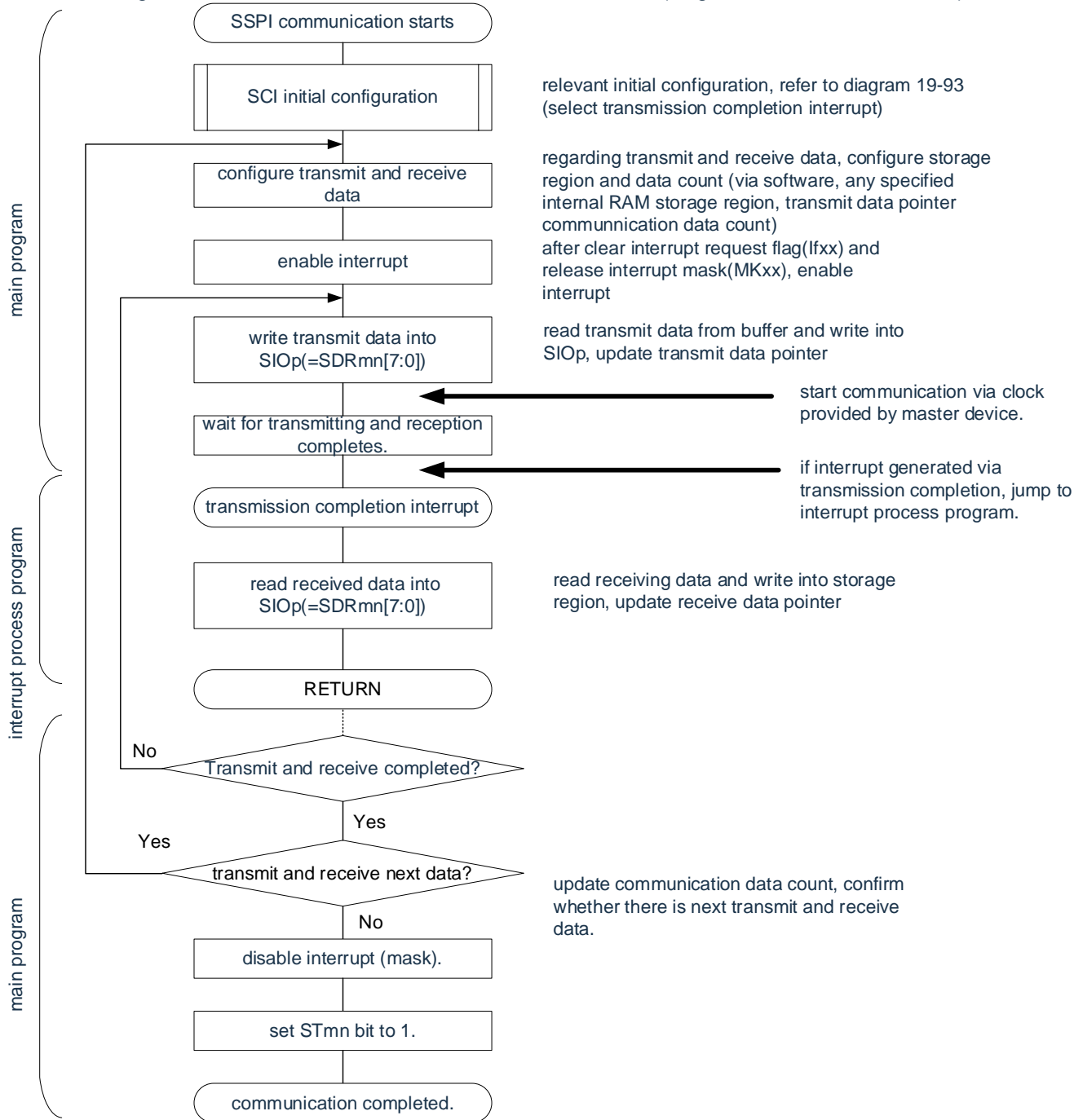
(3) Process flow (single send and receive mode).

Fig. 19-90 Timing diagram of Slave transmit and receive (single send and receive mode) (type 1: DAPmn=0, CKPmn=0).



Note m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

Fig. 19-91 Flowchart of Slave transmit and receive (single send and receive mode).

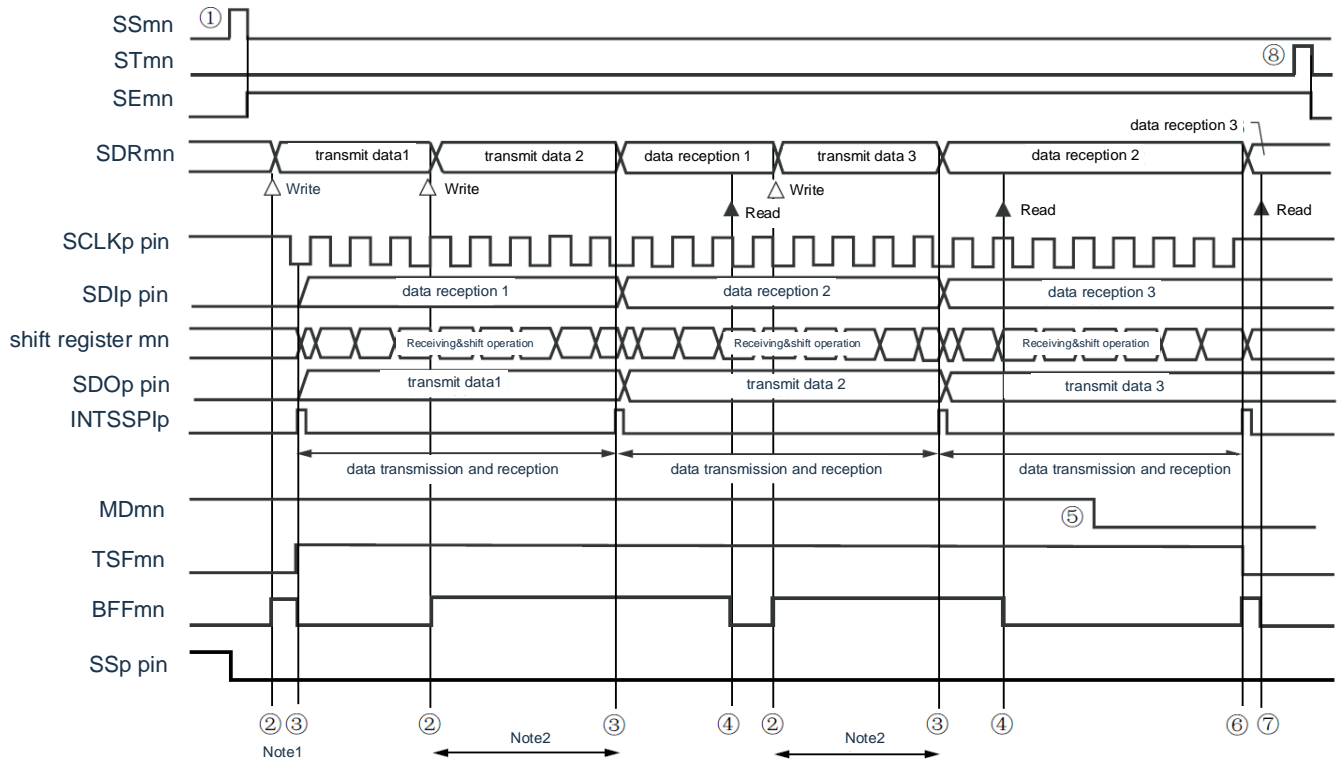


Note that the SIOp register must be set to send data before the master device starts to output the clock.

Note m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

(4) Process flow (continuous send and receive mode).

Fig. 19-92 Timing diagram of Slave send and receive (continuous transmit and receive mode) (type 1: DAPmn=0, CKPmn=0).



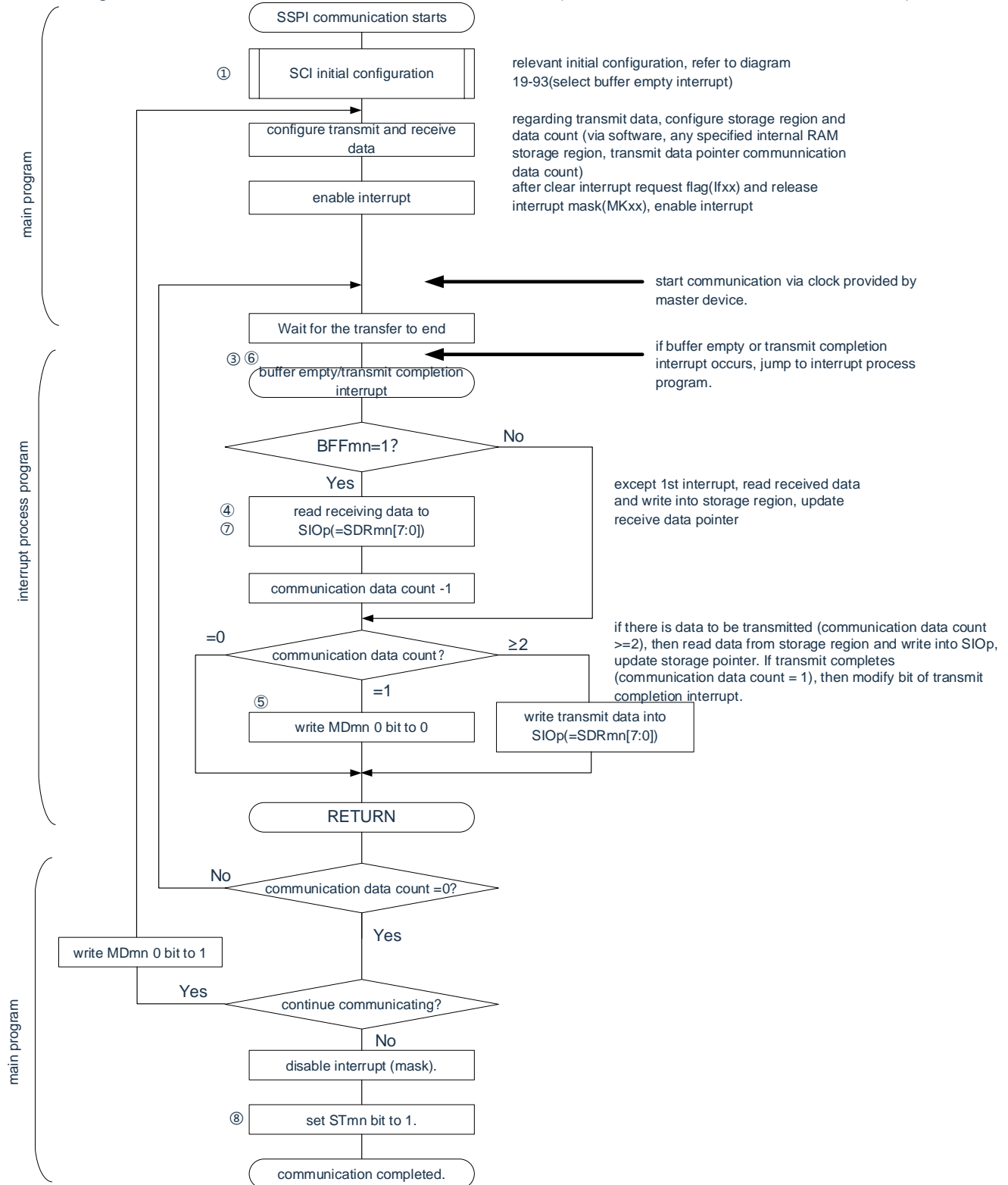
Note 1 If the BFFmn bit of serial status register mn (SSRmn) is "1" during the period (valid data is saved in serial data register mn (SDRmn) (when writing transmit data to the SDRmn register, rewrite the send data).

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.

Note that the MDmn0 bit of the serial mode register mn (SMRmn) can be rewritten even in operation. However, in order to catch the end-of-transmission interruption of the last sent data, it must be rewritten before the last bit of transmission begins.

Note 1 (1) ~ (8) in the figure corresponds to "Fig. 19-93 Flowchart of Slave send and receive (continuous transmit and receive mode)" in (1) ~ (8).

2.m: unit number (m=0)n: channel number (n=0)p: SSPI number (p=00).

Fig. 19-93 Flowchart of Slave send and receive (continuous transmit and receive mode).


Note that the SIOp register must be set to send data before the master device starts to output the clock.

Note 1 (1) to (8) in the figure corresponds to (1) to (8) in the "Fig. 19-92 Timing Diagram of Slave Send and Receive (Continuous Transmit and Receive Mode)".

2.m: unit number (m=0)n: channel number (n=0)p: SSPI number (p=00).

19.6.4 Calculate the transmit clock frequency

The transmit clock frequency of the slave select input function (SSPI00) communication can be calculated using the following calculation equation.

(1) Slaves

$$(\text{Transmit Clock Frequency}) = \{\text{Serial Clock (SCLK) Frequency Rate Provided by the Master Device}\}^{\text{Note}} [\text{Hz}].$$

Note The maximum allowable transmit clock frequency is $f_{\text{MCK}}/6$.

Note m: Unit number (m=0)n: Channel number (n=0)p: SSPI number (p=00)

Table 19-3 Slave Selection Input Function Running Clock Selection

SMRmn register	SPSm register								Running Clock (f_{MCK}) ^{Note}	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		$f_{\text{CLK}}=32\text{MHz}$ runtime
0	X	X	X	X	0	0	0	0	f_{CLK}	32MHz
	X	X	X	X	0	0	0	1	$f_{\text{CLK}}/2$	16MHz
	X	X	X	X	0	0	1	0	$f_{\text{CLK}}/2^2$	8MHz
	X	X	X	X	0	0	1	1	$f_{\text{CLK}}/2^3$	4MHz
	X	X	X	X	0	1	0	0	$f_{\text{CLK}}/2^4$	2MHz
	X	X	X	X	0	1	0	1	$f_{\text{CLK}}/2^5$	1kHz
	X	X	X	X	0	1	1	0	$f_{\text{CLK}}/2^6$	500kHz
	X	X	X	X	0	1	1	1	$f_{\text{CLK}}/2^7$	250kHz
	X	X	X	X	1	0	0	0	$f_{\text{CLK}}/2^8$	125kHz
	X	X	X	X	1	0	0	1	$f_{\text{CLK}}/2^9$	62.5kHz
	X	X	X	X	1	0	1	0	$f_{\text{CLK}}/2^{10}$	31.25kHz
	X	X	X	X	1	0	1	1	$f_{\text{CLK}}/2^{11}$	15.63kHz
	X	X	X	X	1	1	0	0	$f_{\text{CLK}}/2^{12}$	7.81kHz
	X	X	X	X	1	1	0	1	$f_{\text{CLK}}/2^{13}$	3.91kHz
	X	X	X	X	1	1	1	0	$f_{\text{CLK}}/2^{14}$	1.95kHz
	X	X	X	X	1	1	1	1	$f_{\text{CLK}}/2^{15}$	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 Universal Serial Communication Unit (SCI) (serial channel stop register m (STm)=000FH) after making changes.

Note 1.X: Ignore

2.m: unit number (m=0)n: channel number (n=0).

19.6.5 The processing step when an error occurs during clock synchronization 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 Figure 19-94.

Figure 19-94 processing steps when the overflow error occurs

software operation	Hardware Status	Comments
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. 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.

Note m: Unit number (m=0) n: Channel number (n=0).

19.7 Operation of UART (UART0~UART3) communication

This is a function of asynchronous communication over two lines: Serial Data Transmission (TxD) and Serial Data Receiving (RxD). Using these two communication lines, data is sent and received asynchronously (using the internal baud rate) data frame (consisting of start bits, data, parity bits, and stop bits) with other communicators. Full-duplex asynchronous UART communication can be achieved by using two channels dedicated for transmit (even channel) and receive dedicated (odd channel).

[Sending and receiving data].

- 7-bit, 8-bit, or 9-bit data length ^{note} (SCI0).
Data length from 7 to 16 bits (SCI1/SCI2).
- MSB/LSB preferred
- Level setting for sending and receiving data (select whether the level is reversed).
- Additional, parity function of parity bits
- Stop bit attachment, stop bit detection function

[Interrupt function].

- Transmit end interrupt, buffer empty interrupt
- Error interrupts caused by frame errors, parity errors, and overflow errors

[Error Detection Flag].

- Frame error, parity error, overflow error

Note Only UART0 supports 9 bits of data length.

UART0 uses channel 0 and channel 1 of SCI0.

UART1 uses CHANNEL 2 and CHANNEL 3 of SCI0.

UART2 uses CHANNEL 0 and CHANNEL 1 of SCI1.

UART3 uses CHANNEL 0 and CHANNEL 1 of SCI2.

Each channel arbitrarily selects one function to use, except for the selected function, other functions can not be run.

For example, SSPI00 and IIC01 cannot be used when using UART0 for channel 0 and channel 1 of unit 0. However, while using UART0, channels 2 and 3 of different channels can use SSPI10, UART1, or IIC10.

Note that when used as A UART, the sender (even channel) and receiver (odd channel) can only be used for UART.

UART has the following 4 kinds of communication operation:

- UART send (see 19.7.1).
- UART reception (see 19.7.2).

19.7.1 UART sends

UART sending 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	UART3
Object channels	Channel 0 for SCI0	Channel 2 of SCI0	Channel 0 for SCI1	Channel 0 for SCI2
The pins used	TxD0	TxD1	TxD2	TxD3
interrupt	INTST0	INTST1	INTST2	INTST3
	Selectable end-of-transmit interrupt (single-pass mode) or buffer-empty interrupt (continuous transfer mode).			
Error detection flags	not			
The length of the transferred data	SCI0: 7-digit, 8-digit or 9-bit ^{Note 1} SCI1/SCI2: 7 to 16 bits			
Transfer rate	Max. $f_{MCK}/6$ [bps] (SDRmn[15:9]≥2), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps] ^{note2}			
Data phase	Normal-phase output (default: high). Inverting output (default: low).			
Parity bits	You can choose from the following: • No parity bits. • Additional zero checks. • Additional parity. • Additional odd checksum.			
Stop bit	You can choose from the following: • Additional 1 bit. • Additional 2 digits.			
Data direction	MSB priority or LSB priority			

Note 1 Only UART0 supports 9 bits of data length.

- Must be used within the scope of peripheral functional characteristics (refer to data sheet) that meet this condition and meet the electrical characteristics.

Note 1. f_{MCK} : The operating clock frequency
of the object channel

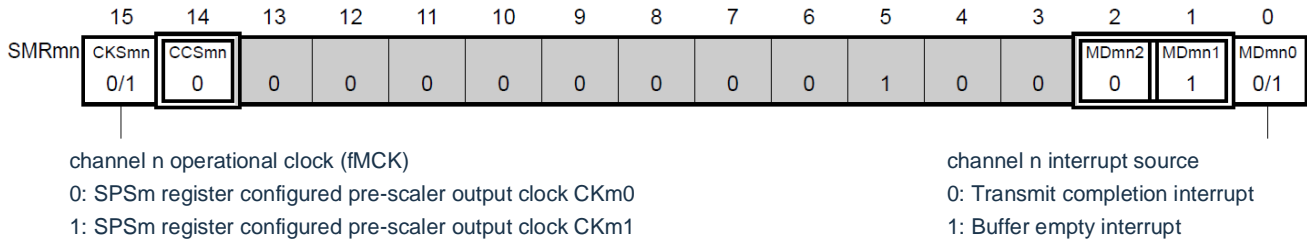
f_{CLK} : System clock frequency

2.m: unit number (m=0, 1, 2)n: channel number (n=0, 2)mn=00, 02, 10, 20

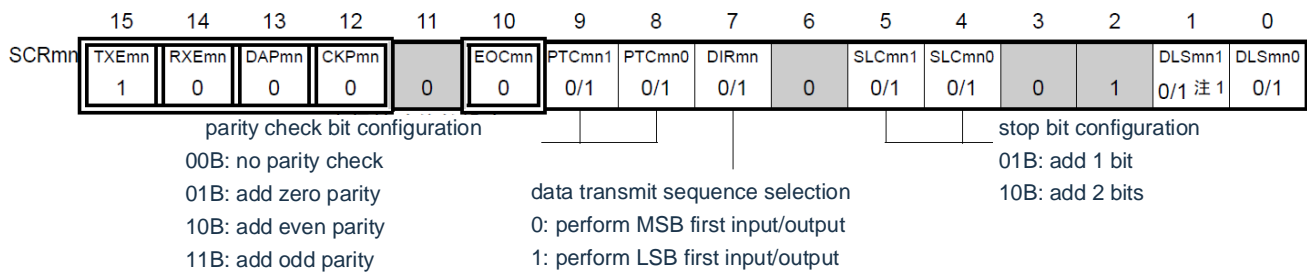
(1) Register settings

Fig. 19-95 register setting content when UART is transmitted by UART (UART0~UART3) (1/2).

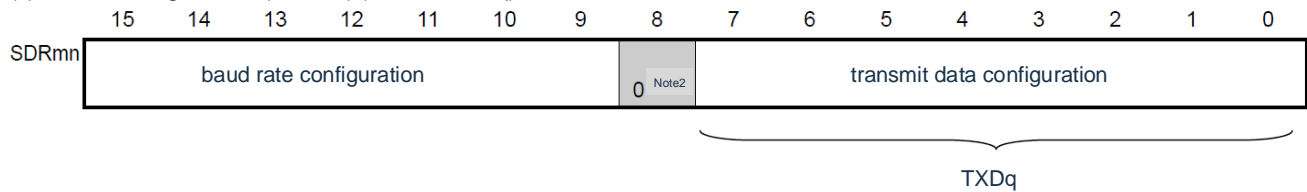
(a) serial mode register mn (SMRmn)



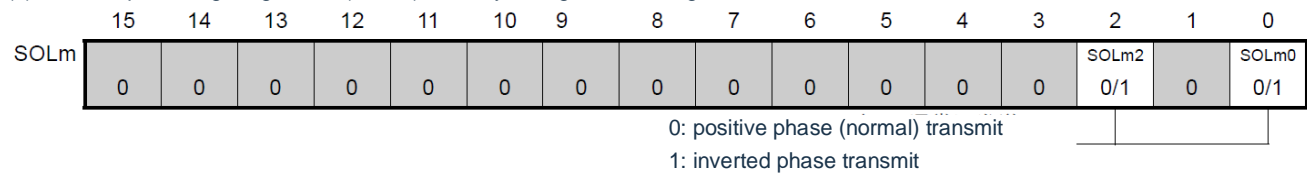
(b) serial communication operation configuration register mn (SCRmn)



(c) serial data register mn (SDRmn) (low 8 bit:TXDq)



(d) serial output voltage register m (SOLm) Only configure bit of target channel.



Note 1 Limited to SCR00 registers, other fixed as "1".

- When communicating with a 9-bit data length, bit0 to 8 of the SDRm0 register is the setting area for sending data. Only UART0 can communicate with a data length of 9 bits.

Note: This example is the setting method for SCI0. The data length of SCI1/SCI2 and the serial data register SDRmn are set differently from this example.

For data length settings, refer to Chapter 19.3.4Serial communication runs the set register mn (SCRmn).Chapter 17).

For the setting method of serial data register SDRmn, refer to "19.3.6Serial data register mn(SDRmn) (SCI1/SCI2 i. e. m=1/2).

Note 1.m: Unit number (m=0~2)n: Channel number (n=0, 2)q: UART number (q=0~3)mn=00, 02, 10, 2 0

2□ : Fixed setting in UART send mode. □ : Cannot be set (initial value is set).

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.

Fig19-96 register setting content when UART is transmitted by UART (UART0 to UART 3) (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					CKOm3	CKOm2	CKOm1	CKOm0					SOM3	SOM2	SOM1	SOM0
	0	0	0	0	×	×	×	×	0	0	0	0	×	0/1 ^{Note}	×	0/1 ^{Note}

0: serial data output value as "0"
1: serial data output value as "1"

(f) serial output enable register m (SOEm).... Only set bit of target channel to "1".

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm													SOEm3	SOEm2	SOEm1	SOEm0
	0	0	0	0	0	0	0	0	0	0	0	0	×	0/1	×	0/1

(g) serial channel start register m (SSm) Only set bit of target channel to "1".

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm													SSm3	SSm2	SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	×	0/1	×	0/1

Note That 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.

Note 1.m: Unit number (m=0~2)n: Channel number (n=0, 2)q: UART number (q=0~3)mn=00, 02, 10, 2 0

2□ : Fixed setting in UART send mode. □ : Cannot be set (initial value is set).

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 19-96 UART transmission

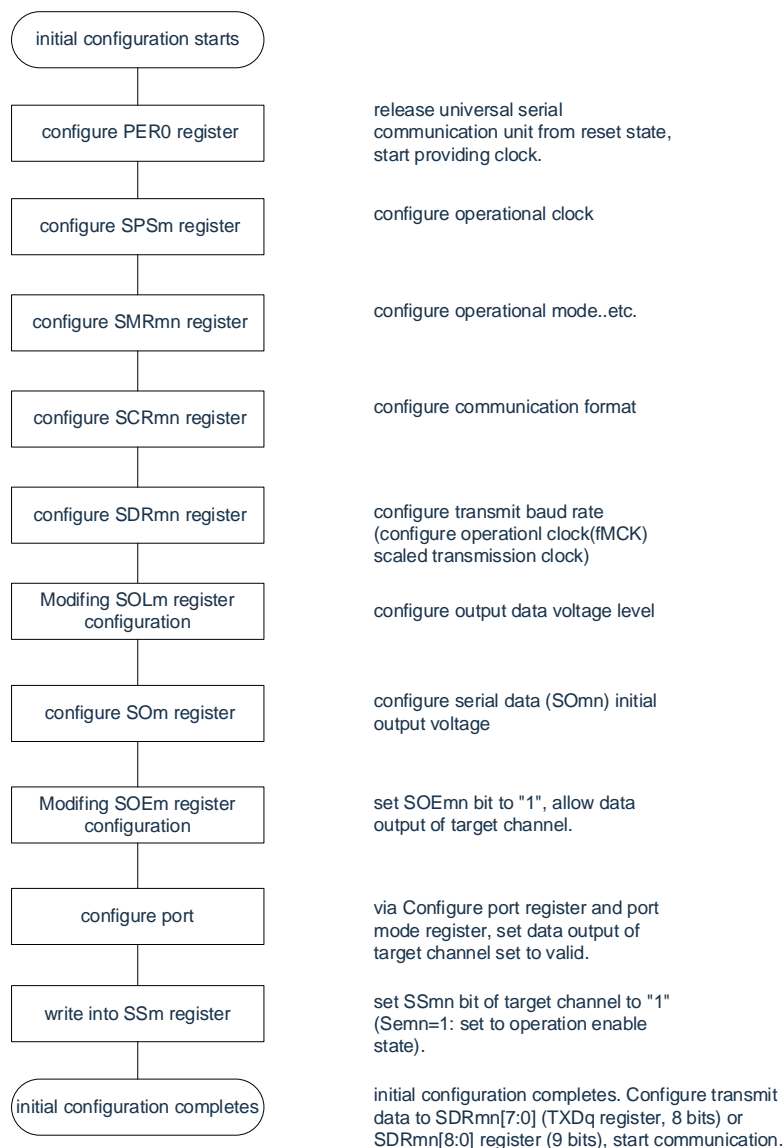


Figure 19-97 The abort steps sent by the UART

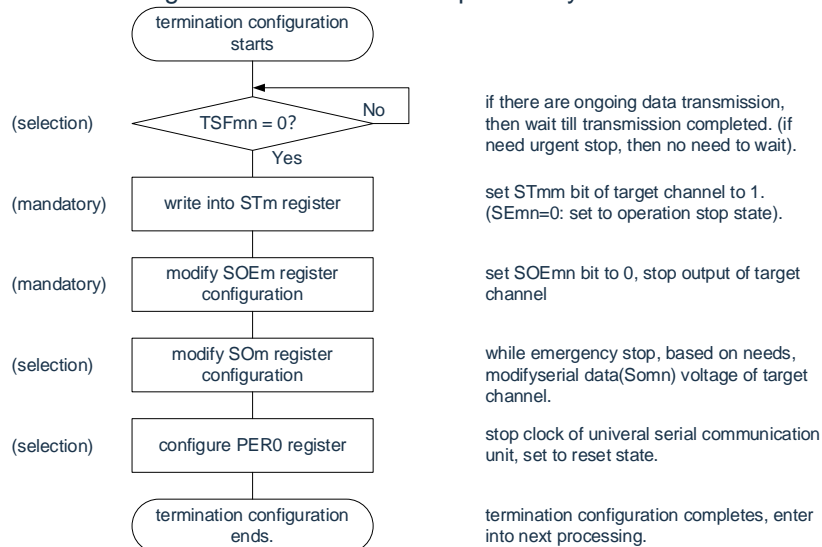
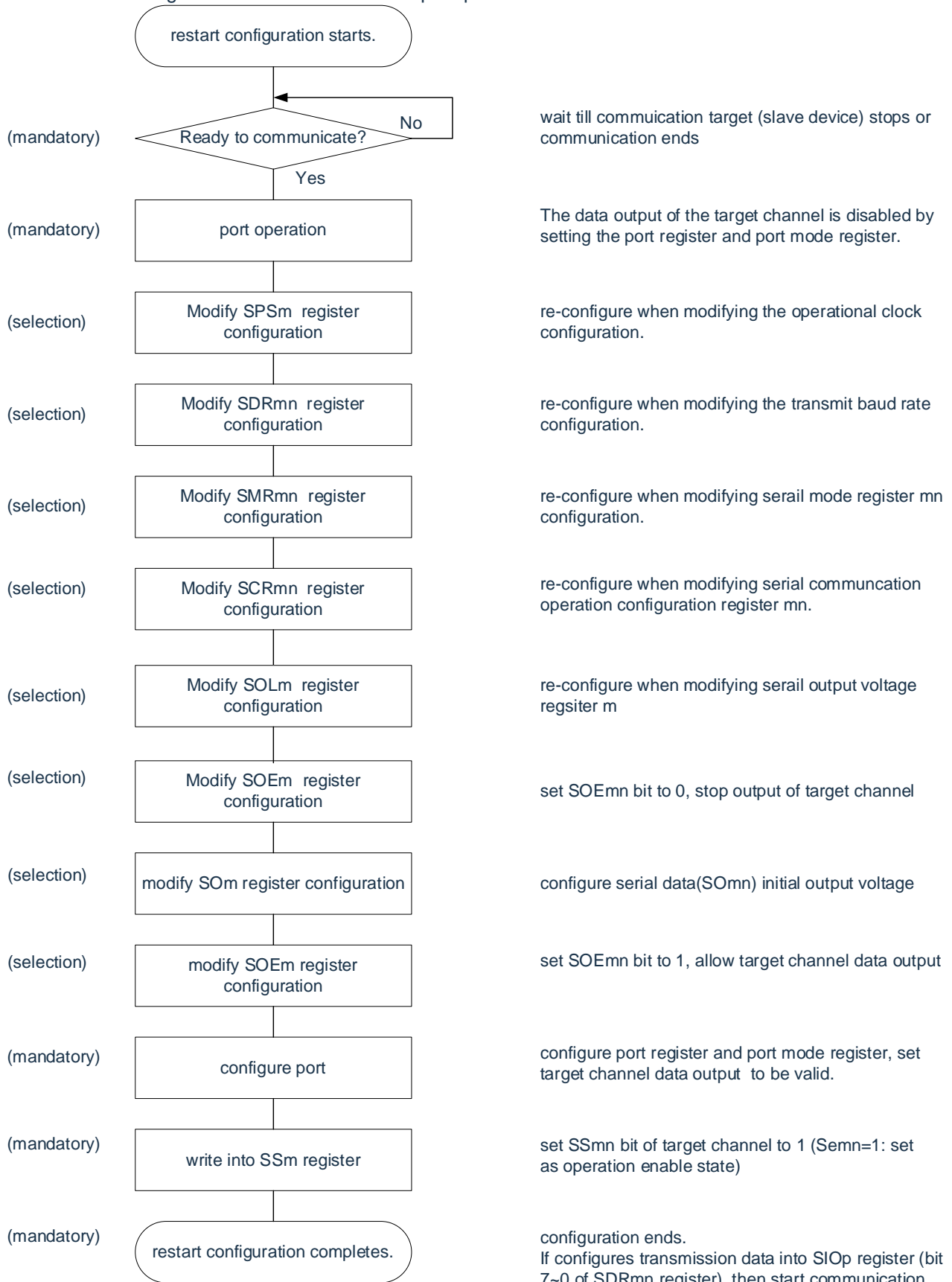


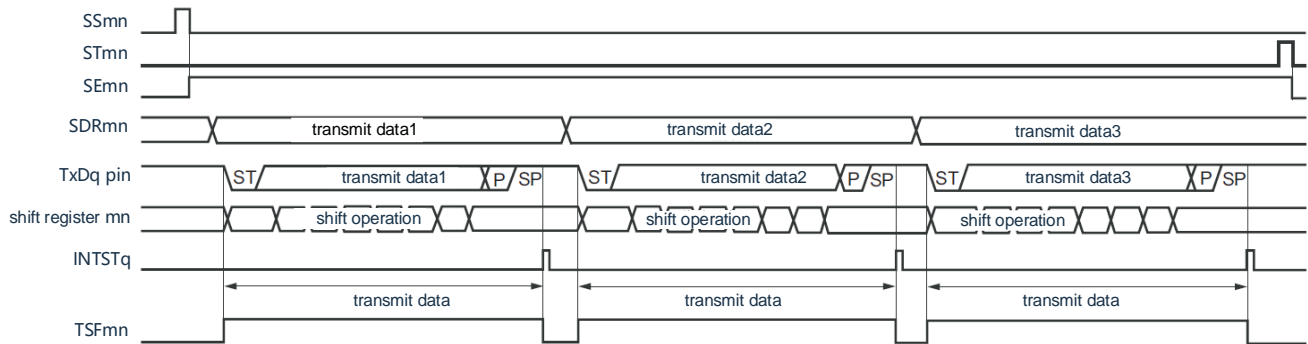
Figure 19-98 Reset the setup steps for the new start UART transmission



Note 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.

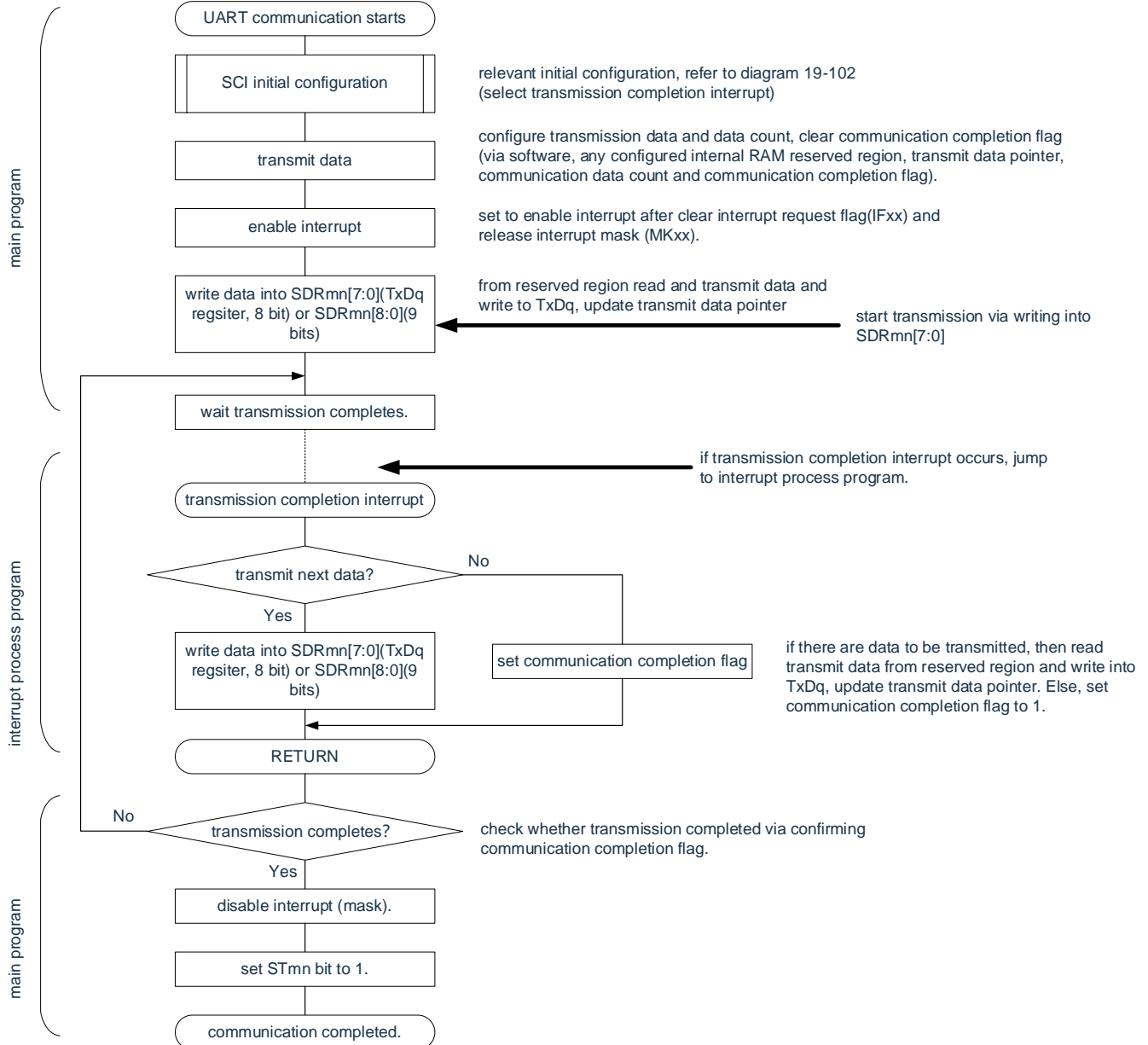
(3) Process flow (single send mode).

Figure 19-99 UART send (single send mode).



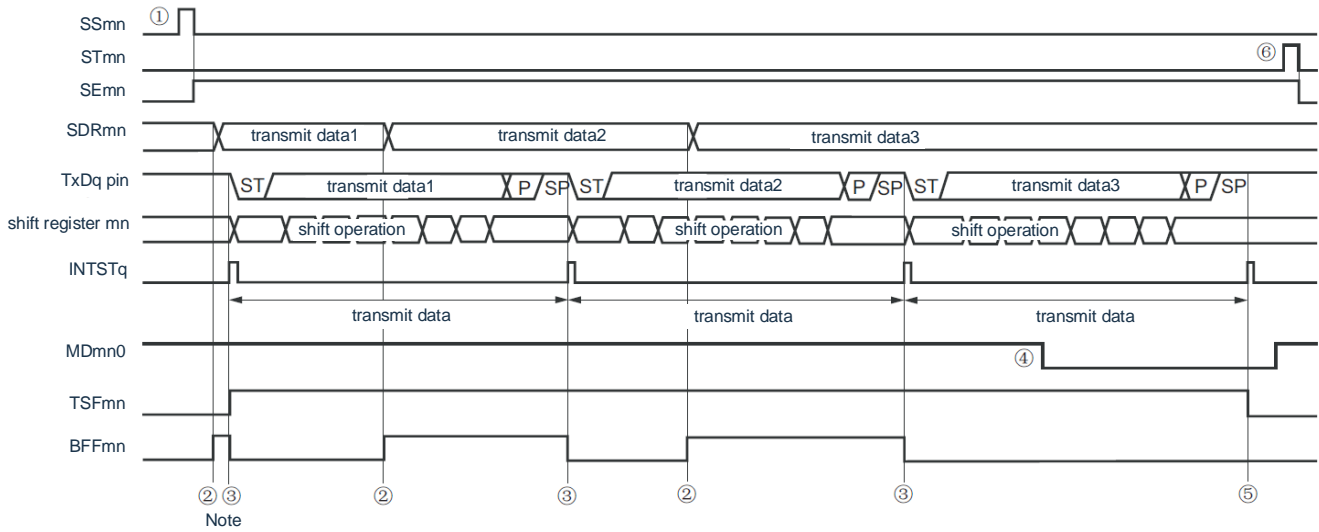
Remark m: unit number (m=0~2) n: channel number (n=0, 2) q: UART number (q=0~3) mn=00, 02, 10, 20

Figure 19-100 UART transmit (single-pass mode).



(4) Process flow (continuous send mode).

Figure 19-101 UART transmission (continuous send mode).

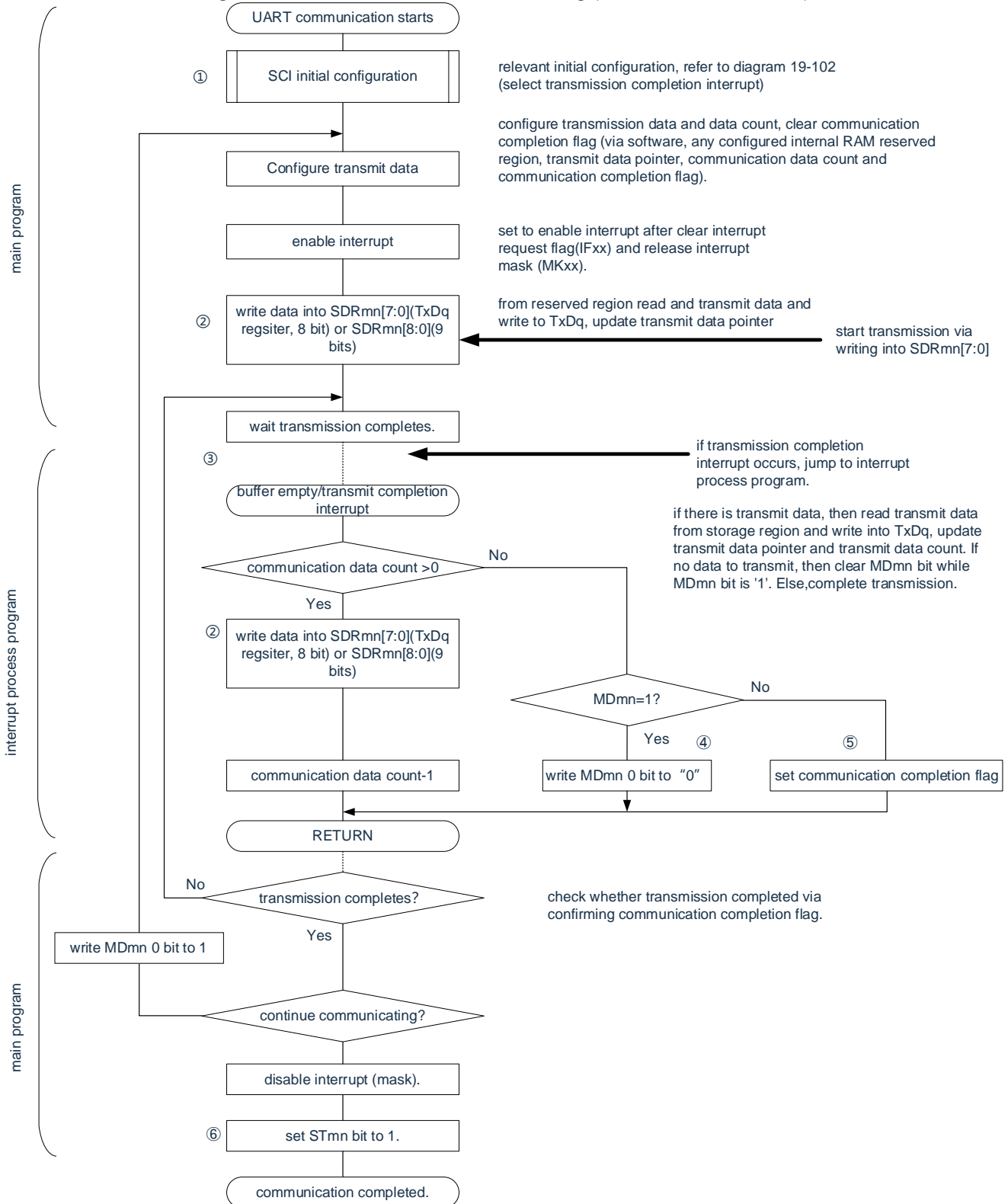


Note If the BFFmn bit of the serial status register mn(SSRmn) is "1" (when valid data is saved in the serial data register mn(SDRmn)) is given When the SDRmn register writes the transmit data, it rewrites the send data.

Note that the MDmn0 bit of the serial mode register mn (SMRmn) can be rewritten even in operation. However, in order to catch the end-of-transmission interruption of the last sent data, it must be rewritten before the last bit of transmission begins.

Remark m: unit number (m=0~2) n: channel number (n=0, 2) q: UART number (q=0~3) mn=00, 02, 10, 20

Figure 19-102 Flowchart of UART sending (continuous send mode).



Note (1) to (6) in the figure corresponds to (1) to (6) in Figure 19-101 UART Transmission (Continuous Send Mode)".

19.7.2 UART receives

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	UART3
Object channels	Channel 1 of SCI0	Channel 3 of SCI0	Channel 1 of SCI1	Channel 1 of SCI2
The pins used	RxD0	RxD1	RxD2	RxD3
interrupt	INTSR0	INTSR1	INTSR2	INTSR3
	Limited to end-of-transmit interrupts (buffer null interrupts are prohibited).			
The error is interrupted	INTSRE0	INTSRE1	INTSRE2	INTSRE3
Error detection flags	<ul style="list-style-type: none"> • Frame Error Detection Flag (FEFmn). • Parity Error Detection Flag (PEFmn). • Overflow Error Detection Flag (OVFmn). 			
The length of the transferred data	SCI0: 7-digit, 8-digit or 9-bit ^{Note 1} SCI1/SCI2: 7 to 16 bits			
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	Normal-phase output (default: high). Inverting output (default: low).			
Parity bits	You can choose from the following: <ul style="list-style-type: none"> • No parity bits (no parity). • Additional zero check (no parity). • Parity • Odd checksum 			
Stop bit	1 additional bit.			
Data direction	MSB priority or LSB priority			

Note 1 Only UART0 supports 9 bits of data length.

2. Must be used within the scope of peripheral functional characteristics (refer to data sheet) that meet this condition and meet the electrical characteristics.

Note 1. f_{MCK} : The operating clock frequency
of the object channel

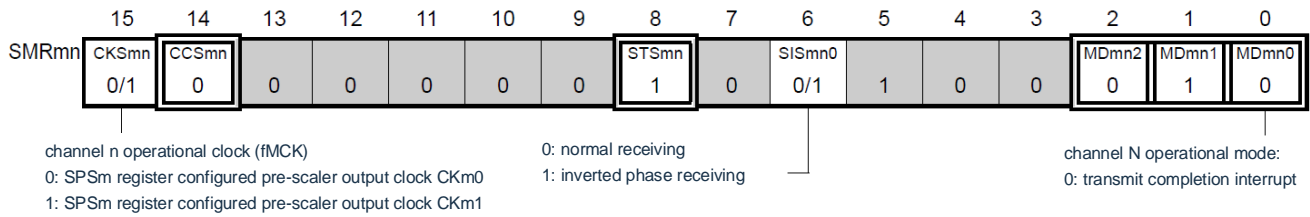
f_{CLK} : System clock frequency

2.m: unit number (m=0~2)n: channel number (n=1, 3)mn=01, 03, 11, 21

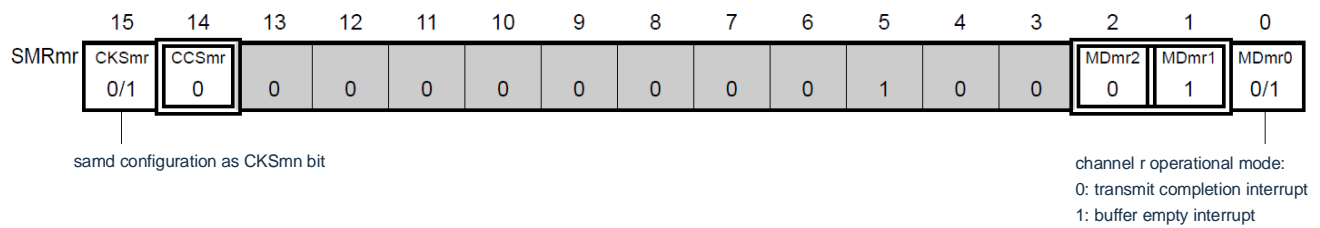
(1) Register settings

19-103 Example of register setting content when UART receives 103 UART (UART0~UART 3) (1/2).

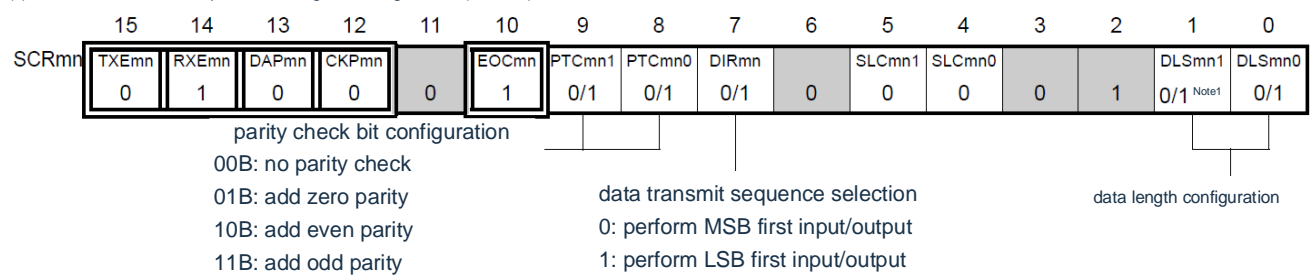
(a) serial mode register mn (SMRmn)



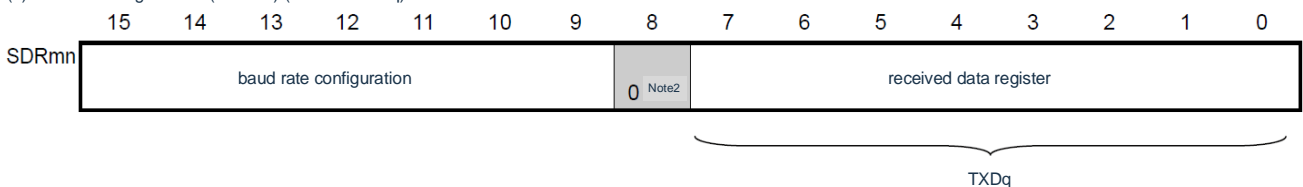
(b) serial mode register mr (SMRmr)



(c) serial communication operation configuration register mn (SCRmn)



(d) serial data register mn (SDRmn) (low 8 bit:TXDq)



Note 1 Limited to SCR01 registers, other fixed as "1".

- When communicating with a 9-bit data length, bit0~8 of the SDRm1 register is the setting area for sending data. Only UART0 can communicate with a data length of 9 bits.

Note: This example is the setting method for SCI0. The data length of SCI1/SCI2 and the serial data register SDRmn are set differently from this example.

For data length settings, refer to Chapter 19.3.4Serial communication runs the set register mn (SCRmn).Chapter 17).

For the setting method of serial data register SDRmn, refer to "19.3.6Serial data register mn(SDRmn) (SCI1/SCI2 i. e. m=1/2).

Note that when the UART receives, the SMRmr register for channel r paired with channel n must also be set.

Remarks 1. m: unit number (m=0~2)n: channel number (n=1, 3) mn=01, 03, 11, 21

r: channel number (r=n-1) q: UART number (q=0~3)

- Fixed setting in UART receive mode.

 : Cannot be set (initial value is set).
 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.

Fig. 19-104: Example of UART (UART0~UART 2) of UART (UART0~UART 2) (2/2).

(e) serial output register m (SOM).... Not used in this mode.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOM	0	0	0	0	CKOm3	CKOm2	CKOm1	CKOm0	0	0	0	0	SOM3	SOM2	SOM1	SOM0
	0	0	0	0	×	×	×	×	0	0	0	0	×	×	×	×

(f) serial output enable register m (SOEm).... Not used in this mode.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	SOEm3	SOEm2	SOEm1	SOEm0
	0	0	0	0	0	0	0	0	0	0	0	0	×	×	×	×

(g) serial channel start register m (SSm) Only set bit of target channel to "1".

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	SSm3	SSm2	SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	0/1	×	0/1	×

Note 1.m: Unit number (m=0~2).

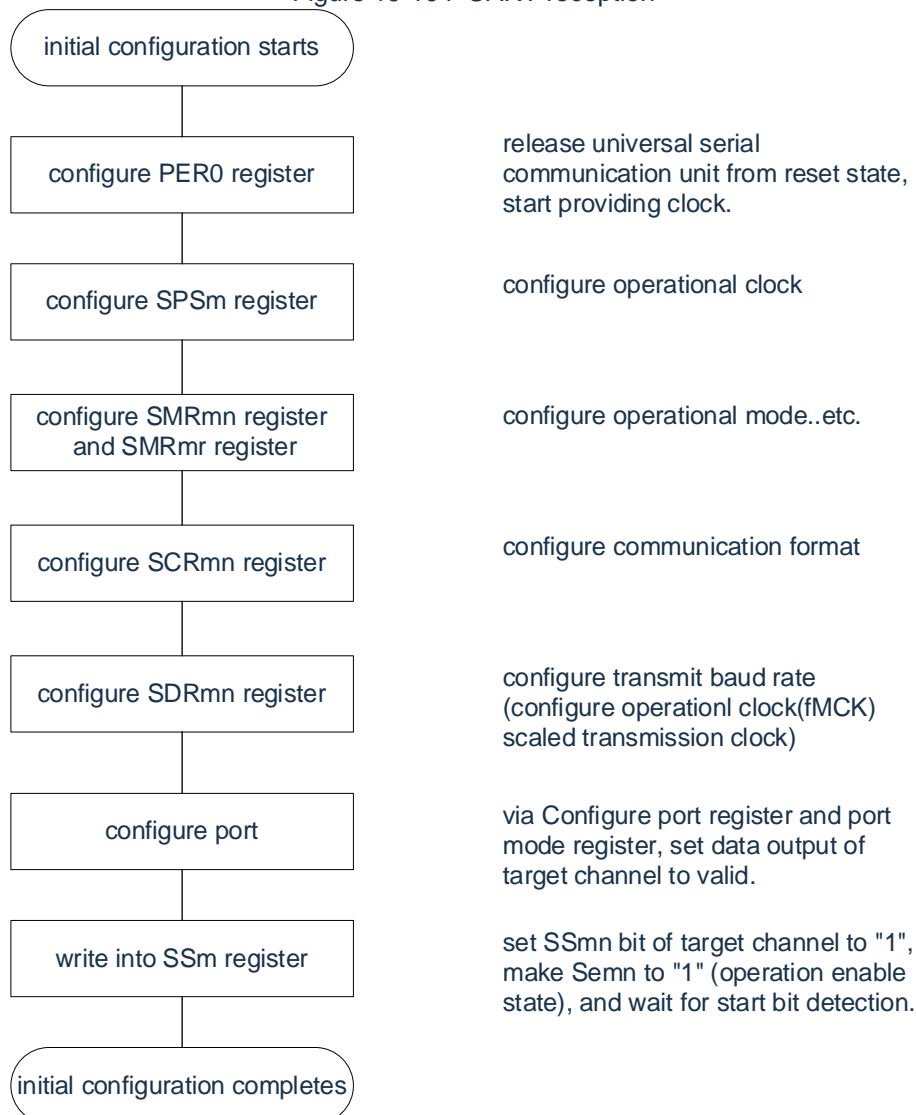
- : Fixed setting in UART receive mode.
 : Cannot be set (initial value is set).

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 19-104 UART reception



Note that at least 4 fMCK clocks must be spaced after the RXEmn position of the SCRmn register "1" and then the SSmn position "1".

Figure 19-105 Abort steps for UART reception

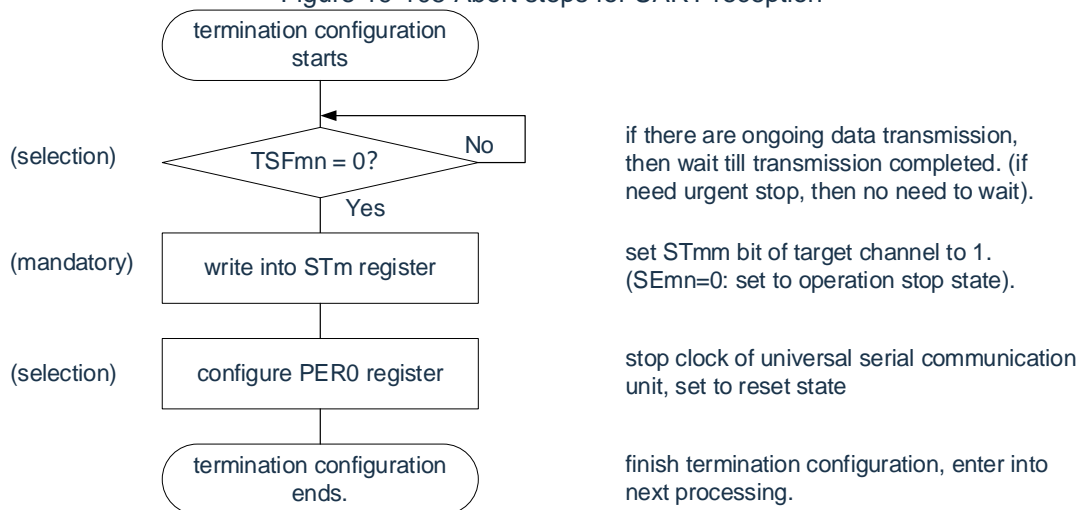
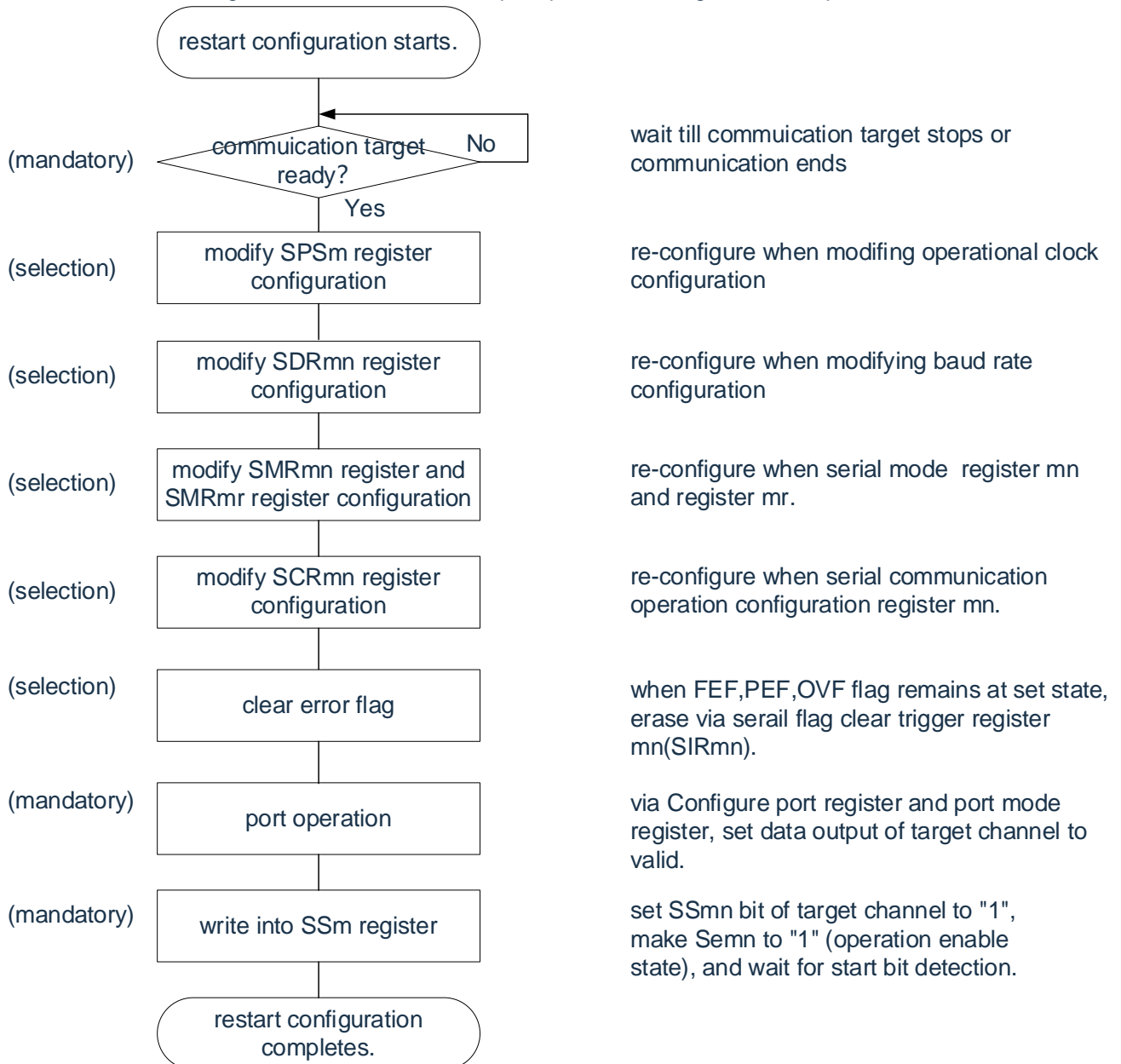


Figure 19-106 Reset the setup steps for restarting UART reception

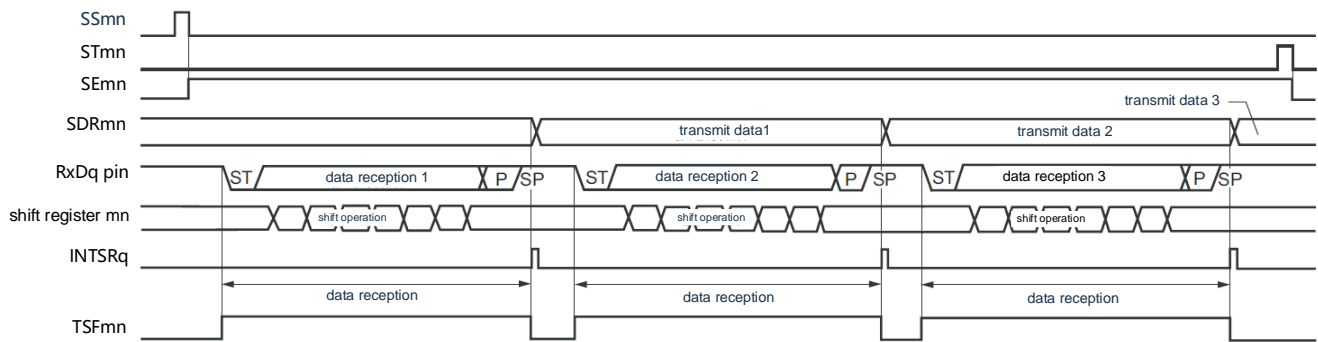


Note that at least 4 f_{MCK} clocks must be spaced after the RXEmn position of the SCRmn register "1" and then the SSmn position "1".

Note 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.

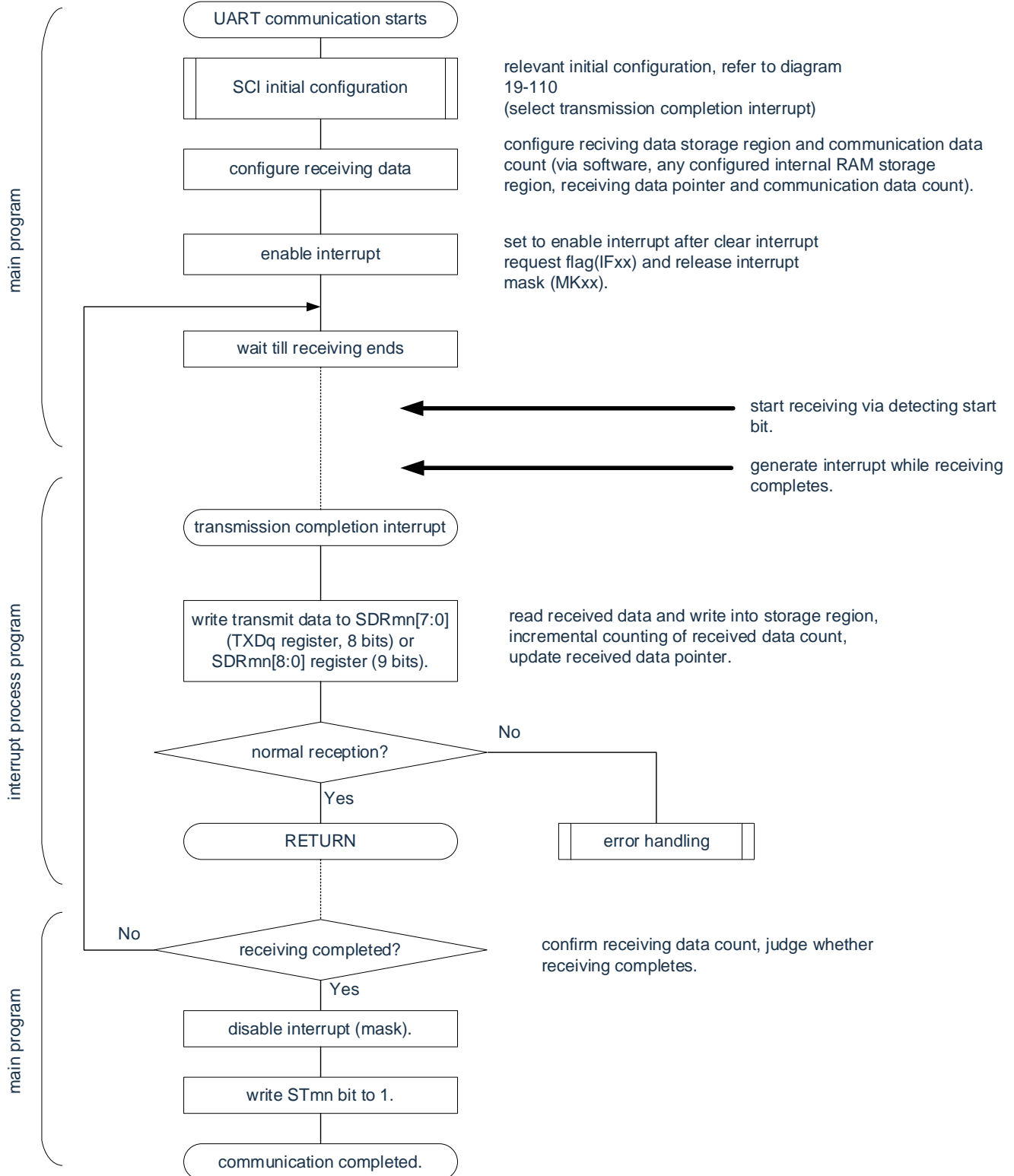
(3) Process flow

Figure 19-107 UART reception



Note m: Unit number (m=0~2)n: Channel number (n=1, 3)mn=01, 03, 11, 21
r: channel number (r=n-1) q: UART number (q=0~3)

Figure 19-108 Flowchart of UART reception



19.7.3 Calculation of baud rate

(1) Equation for calculating baud rate

The baud rate of UART (UART0~UART3) communication can be calculated using the following calculation equation:

$$(\text{baud rate}) = \{\text{Operating clock (f}_{\text{MCK}}) \text{ frequency of the object channel}\} (\text{SDRmn}[15:9] + 1) \div 2[\text{bps}]$$

Note It is forbidden to set the serial data register mn (SDRmn) to SDRmn [15:9] to "0000000B" and "0000001B".

Note 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=0~2)n: channel number (n=0~3). mn=00~03, 10~11, 20~21

The operating clock (fMCK) depends on bit15 of the serial clock selection register m (SPSm) and the serial mode register mn (SMRmn). (CKSmn bit).

Table 19-4 UART operating clocks

SMRmn register	SPSm register								Running Clock (f_{MCK}) ^{Note}	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		$f_{CLK}=32\text{MHz}$ 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
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 Universal Serial Communication Unit (SCI) (serial channel stop register m (STm)=000FH) after making changes.

Note 1.X: Ignore

2.m:unit number(m=0~2)n:channel number(n=0~3)mn=00~03, 10~11, 20~21

(2) Baud rate error when sending

The baud rate error during UART (UART0~UART3) communication transmission can be calculated using the following calculation equation, and the baud rate of the sender must be set within the allowable baud rate of the receiver.

$$(\text{baud rate error}) = (\text{calculated value of baud rate}) \div (\text{value of the target baud rate}) \times 100 - 100 [\%]$$

An example of the UART baud rate at $f_{\text{CLK}}=32\text{MHz}$ is shown below.

UART baud rate (Target baud rate)	$f_{\text{CLK}}=32\text{MHz}$			
	Running Clock	SDRmn[15:9]	The calculated	The error from the
300bps	$f_{\text{CLK}}/2^9$	103	300.48bps	+0.16%
600bps	$f_{\text{CLK}}/2^8$	103	600.96bps	+0.16%
1200bps	$f_{\text{CLK}}/2^7$	103	1201.92bps	+0.16%
2400bps	$f_{\text{CLK}}/2^6$	103	2403.85bps	+0.16%
4800bps	$f_{\text{CLK}}/2^5$	103	4807.69bps	+0.16%
9600bps	$f_{\text{CLK}}/2^4$	103	9615.38bps	+0.16%
19200bps	$f_{\text{CLK}}/2^3$	103	19230.8bps	+0.16%
31250bps	$f_{\text{CLK}}/2^3$	63	31250.0bps	±0.0%
38400bps	$f_{\text{CLK}}/2^2$	103	38461.5bps	+0.16%
76800bps	$f_{\text{CLK}}/2$	103	76923.1bps	+0.16%
153600bps	f_{CLK}	103	153846bps	+0.16%
312500bps	f_{CLK}	50	313725bps	±0.39%

Note m: Unit number (m=0~2)n: Channel number (n=0, 2)mn=00, 02, 10, 20

(3) The allowable range of baud rate at the time of reception

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 rate tolerance range of the receiver.

$$(\text{Maximum baud rate that can be received}) = \frac{2 \times K \times \text{Nfr}}{2 \times K \times \text{Nfr} - K + 2} \times \text{Brate}$$

$$(\text{Minimum baud rate that can be received}) = \frac{2 \times K \times (\text{Nfr} - 1)}{2 \times K \times \text{Nfr} - K - 2} \times \text{Brate}$$

Brate: The calculated value of the baud rate of the receiver (cf. "19.7.4 (1) Baud rate calculation").

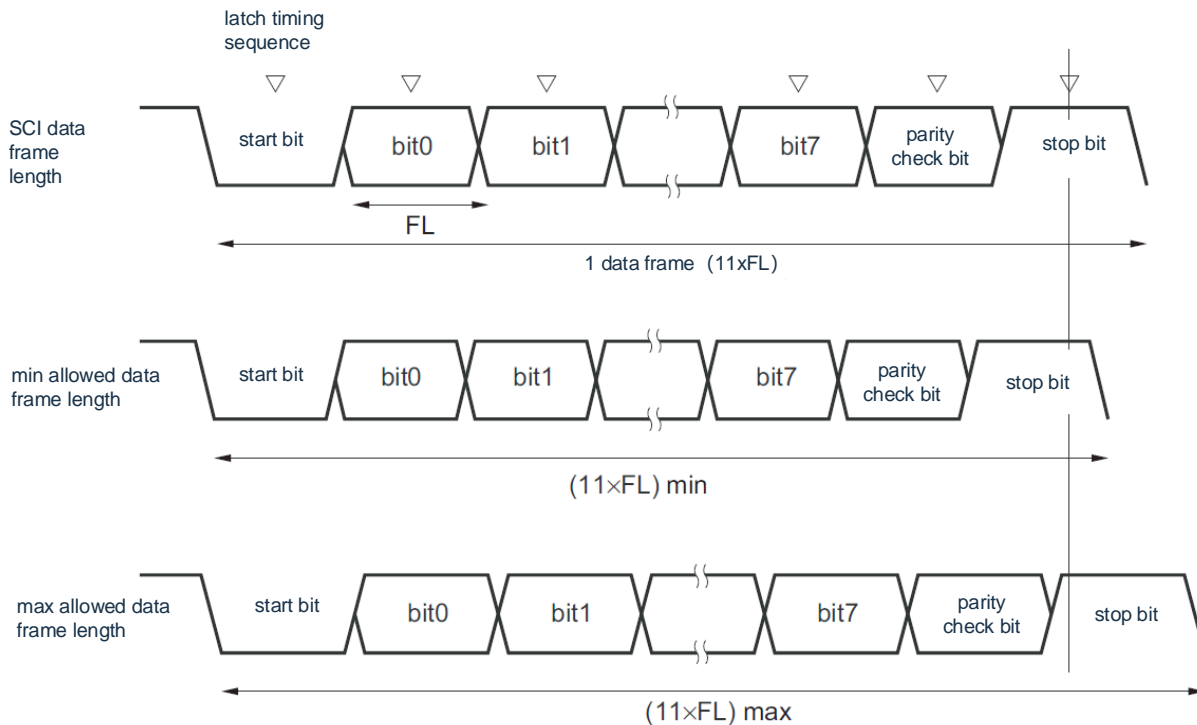
k: SDRmn [15:9]+1

Nfr: 1 frame length [bits] of data

= (start bit) + (data length) + (parity bit) + (stop bit).

Note m: Unit number (m=0~2) n: Channel number (n=1, 3) mn=01, 03, 11, 21

Fig. 19-109 The allowable range of baud rates at the time of reception (in the case of frame length = 11 bits of 1 data).



As shown in Fig. 19-109, 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.

19.7.4 The processing step when an error occurs during UART (UART0~UART3) communication

The processing steps for errors that occur during UART (UART0~UART 3) communication are shown in Figure 19-110 and Figure 19-111.

Figure 19-110: The 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". →	Clears 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 19-111: Processing steps when a frame error occurs

Software operation	Hardware status	remark
Read the serial data register mn (SDRmn). →	The BFFmn bit of the SSR mn register is "0" and channel n is in a receivable state.	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) 。 →	Clears 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.
Stop the serial channel register m(STm). STmn position "1". →	The serial channel allows the SEm n bit of the status register m(SEm) to be "0" and the channel n is the run stopped state.	
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.
Register the serial channel starting m (SSm). SSmn position "1". →	The serial channel allows the SEm n bit of the status register m(SEm) to be "1" and channel n to be operational.	

Note: m: Unit number (m=0~2)n: Channel number (n=0~3)mn=00~ 03, 10~11, 20~21

19.8 Operation of LIN communications

19.8.1 LIN sends

In UART transmission, UART0 supports LIN communication.

LIN sends channel 0 using unit 0.

UART	UART0	UART1	UART2	UART3
Support for LIN communication	Yes	No	No	No
Object channels	Channel 0 for SCI0	—	—	—
The pins used	TxD0	—	—	—
interrupt	INTST0	—	—	—
Error detection flags	not			
The length of the transferred data	8 bits			
Transfer Rate ^{Note}	Max. $f_{MCK}/6$ [bps] (SDR00[15:9]≥2), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps]			
Data phase	Normal-phase output (default: high). Inverting output (default: low).			
Parity bits	There are no parity bits.			
Stop bit	1 additional bit.			
Data direction	LSB takes precedence			

Note it must be used within the scope of peripheral functional characteristics (refer to data sheet) that meet this condition and meet the electrical characteristics, and 2.4/9.6/19.2kbps are often used in LIN communications.

Note f_{MCK} : Object channel's
operating clock frequency f_{CLK} :
System clock frequency

LIN, short for Local Interconnect Network, is a low-speed (1 to 20kbps) serial communication protocol for reducing the cost of automotive networks. LIN communication is a single master communication, and a master device can connect up to 15 slave devices.

LIN slaves are used for the control of switches, actuators, sensors, etc., which are connected to the master control device via LIN.

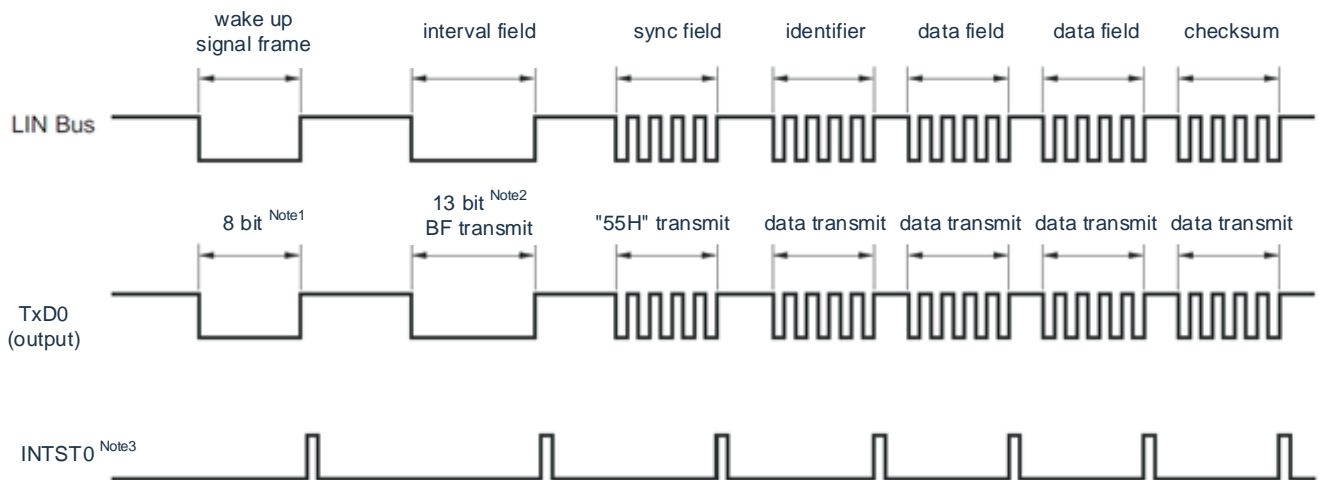
The LIN master is generally connected to a network such as CAN (Controller Area Network).

The LIN bus is a one-wire bus that connects nodes through ANDO9141-compliant transceivers.

According to the LIN protocol, the master device sends a frame with additional baud rate information, and the slave receives this frame and corrects the baud rate error with the master device. Therefore, if the baud rate error of the slave is not greater than 15% of the \pm , communication can be carried out.

A summary of the send operation of LIN is shown in Figure 19-112.

Figure 19-112: The send operation of a LIN



Note 1 In order to meet the requirements of the wake-up signal, the baud rate is set and corresponds by sending the data "80H".

2. The spacer segment is specified as a 13-bit wide low-level output, so it is assumed that the baud rate used by the main transmission is N[bps], and the baud rate used in the spacer segment is as follows:

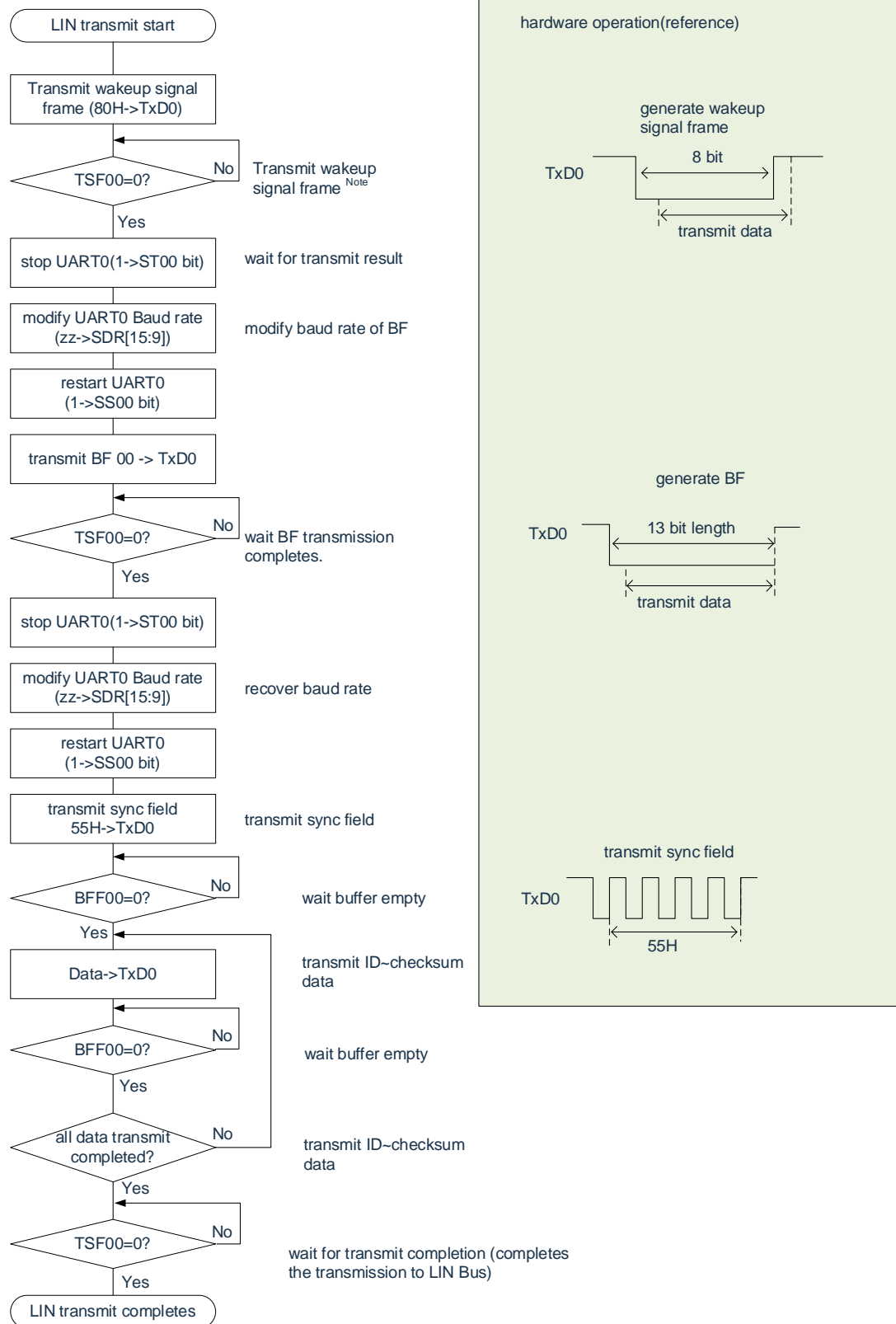
$$(\text{baud rate of the interval segment}). = 9/13 \times N$$

The data of "00H" is sent at this baud rate to generate interval segments.

3. Output INTST0 at the end of each data transmission, and also output INTST0 when BF is sent.

Note The software controls the interval between segments.

Figure 19-113 Flowchart sent by LIN



Note is limited to cases where booting from the LIN-bus sleep state is limited.

Note This is the process that begins with the initial setup of the end UART and enable Slave sending.

19.8.2 LIN receives

In UART reception, UART0 supports LIN communication.

LIN receives channel 1 using unit 0.

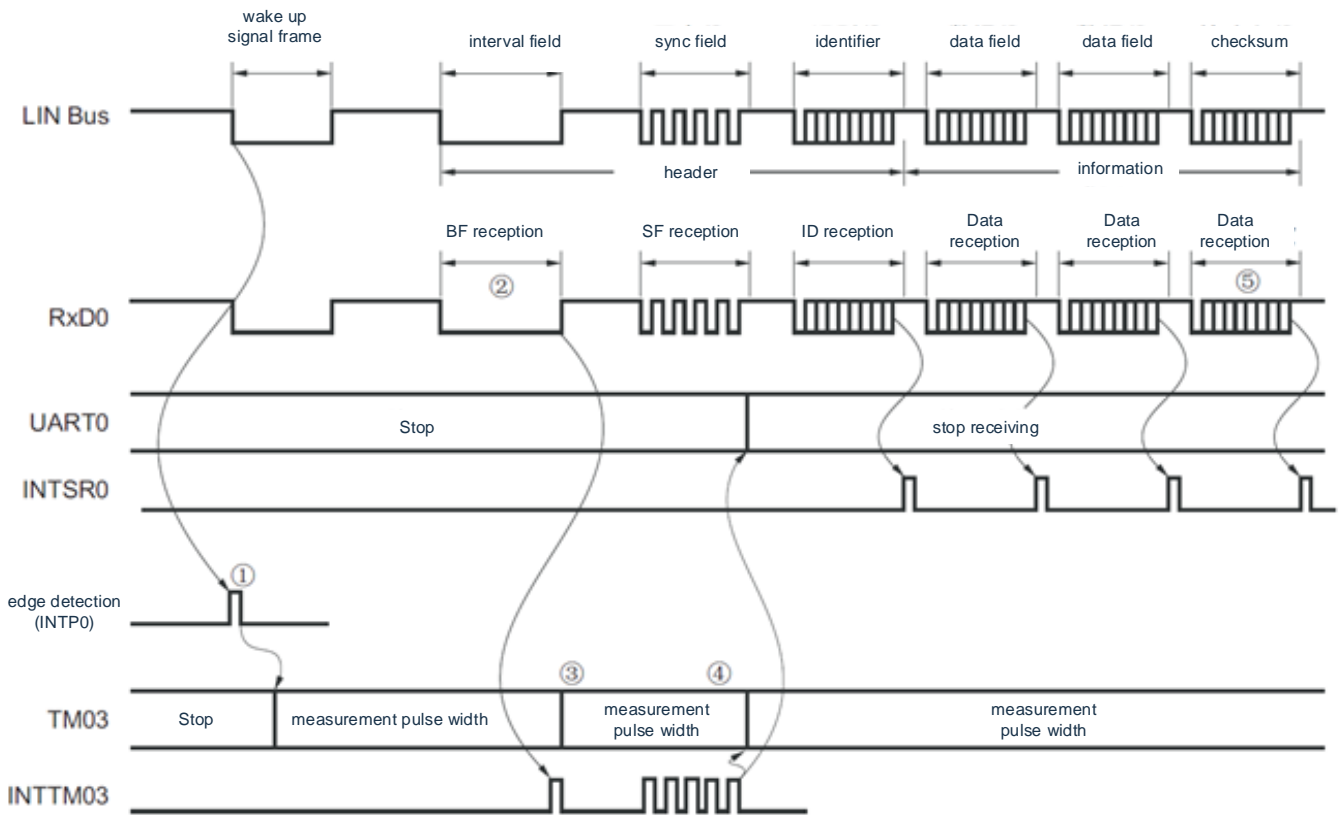
UART	UART0	UART1	UART2	UART3
Support for LIN communication	Yes	No	No	No
Object channels	Channel 1 of SCI0	—	—	—
The pins used	RxD0	—	—	—
interrupt	INTSR0	—	—	—
	Limited to end-of-transmit interrupts (buffer null interrupts are prohibited).			
The error is interrupted	INTSRE0	—	—	—
Error detection flags	•Frame Error Detection Flag (FEF01). •Overflow Error Detection Flag (OVF01).			
The length of the transferred data	8 bits			
Transfer Rate ^{Note}	Max. $f_{MCK}/6$ [bps] (SDR01[15:9]≥2), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps]			
Data phase	Normal-phase output (default: high). Inverting output (default: low).			
Parity bits	No parity bits (no parity).			
Stop bit	1 additional bit.			
Data direction	LSB takes precedence			

Note It must be used within the scope of peripheral functional characteristics that meet this condition and meet the electrical characteristics (see data sheet).

Note f_{MCK} : Object channel's operating clock frequency
 f_{CLK} : System clock frequency

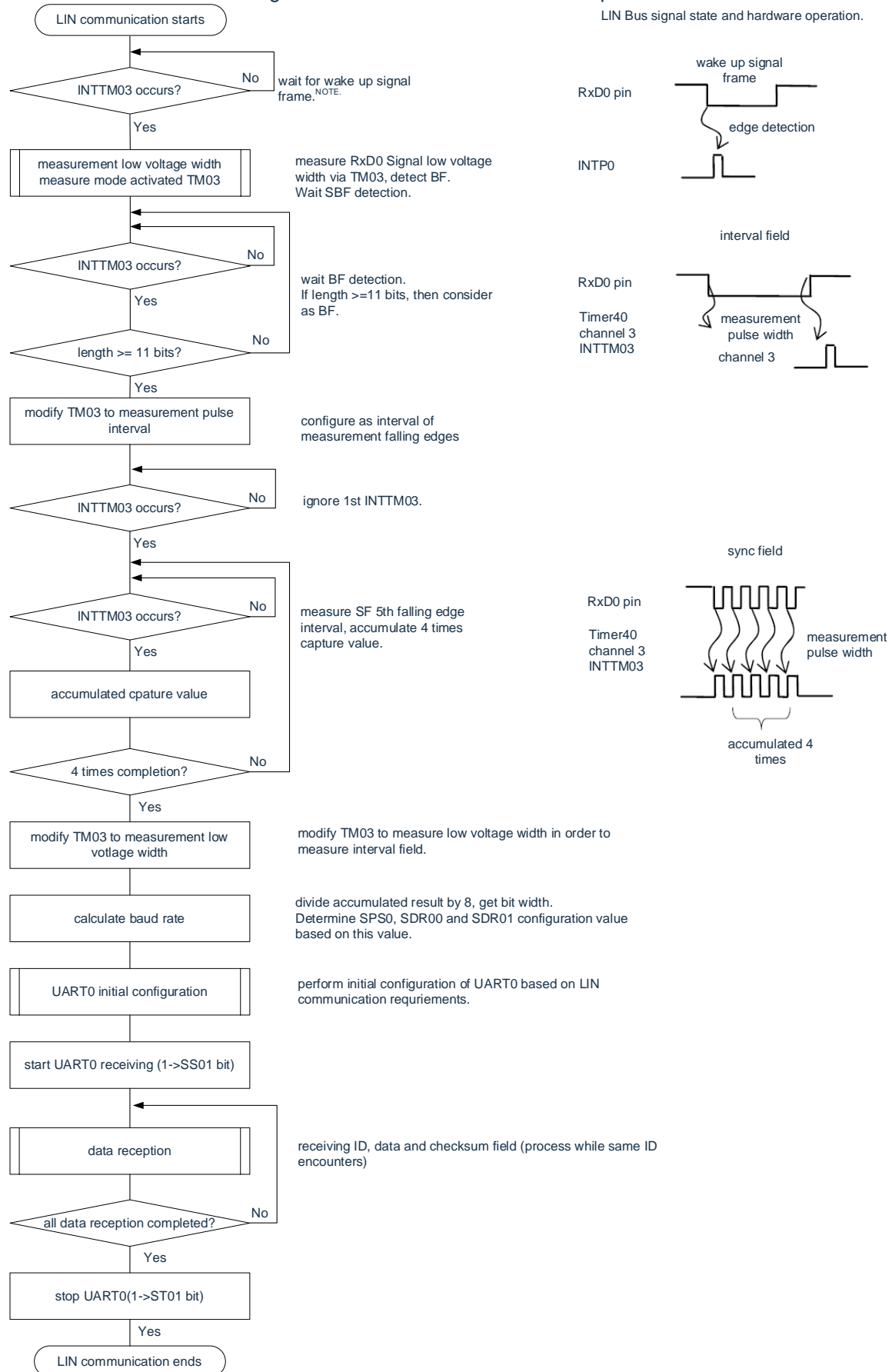
A summary of the receive operation of LIN is shown in Figure 19-114.

Figure 19-114 The receive operation of LIN



The flow of signal processing is as follows:

- (1) Detect the wake-up signal by detecting the interrupt edge of the detection pin (INTP0). When a wake-up signal is detected, to measure the low level width of BF, set TM03 to measure pulse width, and then enter the BF receive waiting state.
- (2) If the falling edge of BF is detected, the TM03 begins to measure the low level width and snaps on the rising edge of BF. Based on the captured value, it is a BF signal.
- (3) When the BF reception ends normally, TM03 must be set to measure the pulse interval and the interval at which the RxD0 signal drops in the 4 sync segments must be measured (cf. "5." .8.4 Operation as input pulse interval measurement").
- (4) Calculate the baud rate error according to the bit spacing of the synchronization segment (SF). The baud rate must then be adjusted (reset) after the UART0 run is paused.
- (5) Must pass the software area branch check section. UART0 must also be initialized by software after receiving the checksum segment and set again to the BF receive wait state.

Figure 19-115 Flowchart of LIN reception


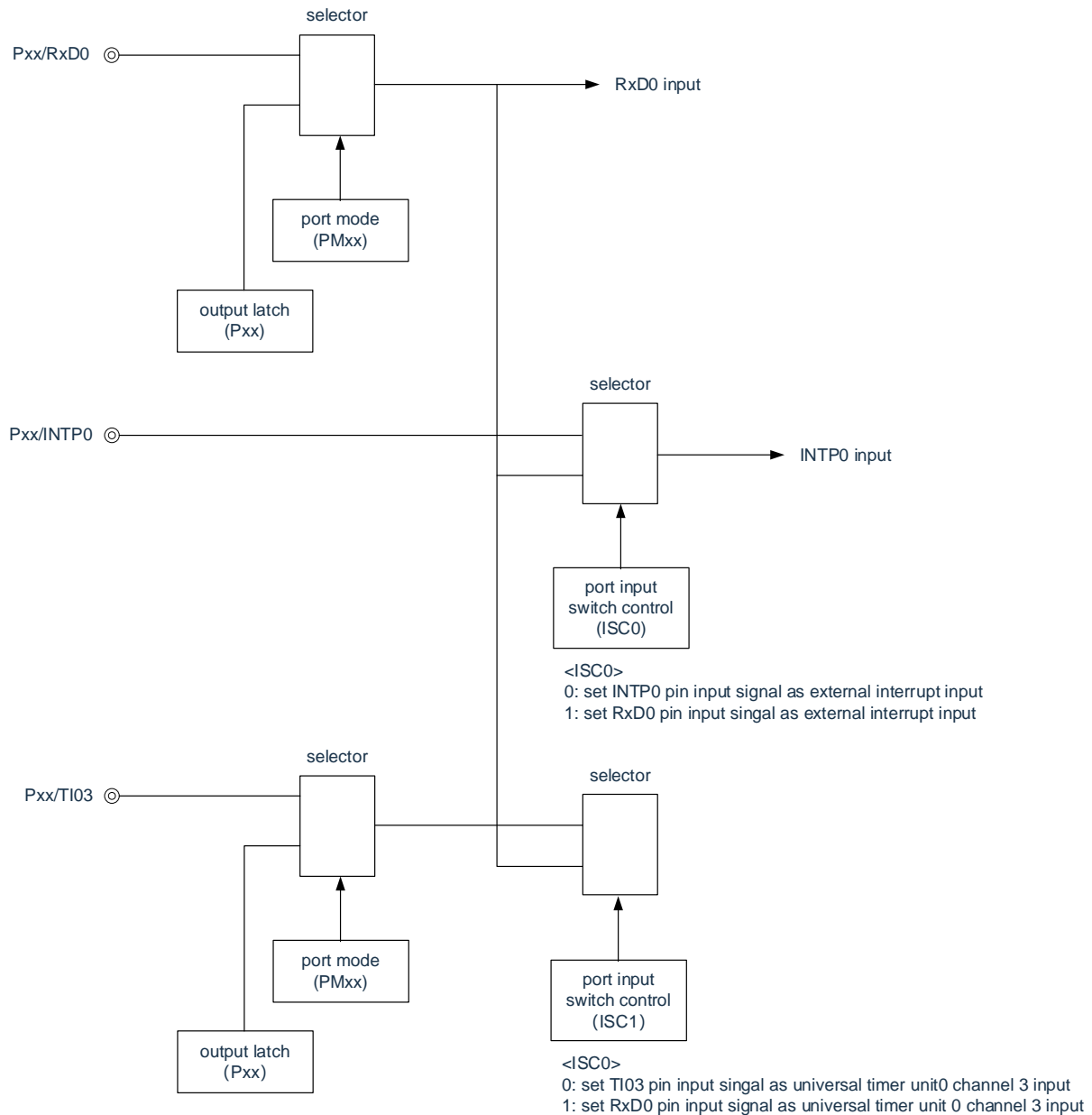
Note: Required only during sleep.

The port structure diagram for THE LIN receive operation is shown in Figure 19-116.

The wake-up signal sent by the LIN master is received through edge detection of the external interrupt (INTP0). It can measure the length of the synchronization segment sent by the LIN master and calculate the baud rate error through the external event capture operation of the universal timer unit.

Port input switching control (ISC0/ISC1) enables the input source of the port input (RxD0) used for reception to be fed into an external interrupt (INTP0) and timer array unit without external wiring.

Figure 19-116 Port structure diagram for LIN receive operation



Note ISC0, ISC1: Enter bit0 and bit1 of the Switch Control Register (ISC) (refer to Figure 19-19)

The peripheral functions for LIN communication operation are summarized as follows:

The peripheral functions that < use >

- External Interrupt (INTP0): Detection of wake-up signals

Uses: Detects the edge of the wake signal and the start of communication.

Channel 3 of the universal timer unit: detection of baud rate error, detection of interval segment (BF).

Purpose: Detects the length of the synchronization segment (SF) and detects baud rate error by dividing its length by the number of bits (the interval between the RxD0 input edges is measured by snapping mode).

Measure the low level width to determine if it is a spacer segment (BF).

- Channel 0 and Channel 1 (UART0) of Universal Serial Communication Unit 0 (SCI0).

19.9 Simple I²C (IIC00, IIC01, IIC10, IIC11, IIC20, Operation of IIC21, IIC30, IIC31) communication

This is the function of clock synchronization communication with multiple devices through two lines of serial clock (SCL) and serial data (SDA). Because this simple I²C is designed for single communication with devices such as EEPROM, flash memory, A/D converters, etc., it is only used as a master device.

For start and stop conditions, AC specifications must be adhered to and processed by software while operating the control registers.

[Sending and receiving data].

- Master send, master receive (limited to single master master function).
- ACK output function ^{note}, ACK detection function
- 8 bits data length (when sending an address, specify the address with a high 7 bits, and R/W control with the lowest bit).
- Generate start and stop conditions via software.

[Interrupt function].

- End of transfer interrupted

[Error Detection Flag].

- ACK error

※[Function not supported by Simple I²C].

- Slave sending, Slave receiving
- Multi-master function (quorum failure detection function).
- Wait for detection feature

Note When receiving the last data, if you write "0" to the SDOEmn bit (SDOEm register) to stop the output of the serial communication data, the ACK is not output. For details, please refer to "19.9.3 (2) Process".

Note m: Unit number (m=0~2)n: Channel number (n=0~3)mn=00~ 03, 10~11, 20~21

Channels 0 to 3 for SCI0, channels 0 to 1 for SCI1, and channels for SCI2 0~1 is to support simple I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21, IIC30, IIC31) channels.

Simple I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21, IIC30, IIC31) has the following 4 types of communication operation:

- Address segment sending (cf19.9.1).
- Data transmission (see 19.9.2).
- Data reception (cf19.9.3).
- Generation of stop conditions (cf19.9.4).

19.9.1 The address segment is sent

The address segment transmission is the first transmission operation when communicating in I²C for the purpose of specifically specifying the transmitting object (the slave). After the start condition is generated, the address (7 bits) and the transmission direction (1 bit) are sent as 1 frame.

Simple I ² C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21	IIC30	IIC31
Object channels	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1	SCI2 Channel 0	SCI2 Channel 1
The pins used	SCL00, SDA00 ^{note 1}	SCL01, SDA01 ^{note 1}	SCL10, SDA10 ^{note 1}	SCL11, SDA11 ^{note 1}	SCL20, SDA20 ^{note 1}	SCL21, SDA21 ^{note 1}	SCL30, SDA30 ^{note 1}	SCL31, SDA31 ^{note 1}
interrupt	INTIIC00	INTIIC01	INTIIC10	INTIIC11	INTIIC20	INTIIC21	INTIIC30	INTIIC31
	Limited to end-of-transmit interrupts (buffer null interrupts cannot be selected).							
Error detection flags	ACK Error Detection Flag (PEFmn).							
The length of the transferred data	8 bits (send the high 7 bits as the address and the lower 1 bit as the R/W control).							
Transfer rate ^{note 2}	Max.f _{MCK} /4[Hz] (SDRmn[15:9] ≥ 1)f _{MCK} : Object Channel's operating clock frequency However, it must be at I ² C The following conditions are met in each pattern of C: <ul style="list-style-type: none"> • Max.1MHz (Enhanced Quick Mode). • Max.400kHz (fast mode). • Max.100kHz (standard mode). 							
Data level	Normal-phase output (default: high).							
Parity bits	There are no parity bits.							
Stop bit	Additional 1 bit (for ACK reception).							
Data direction	MSB takes precedence							

Note 1 To communicate via Simple I²C, the N-channel open-drain output mode (POMxx=1) must be set via the port output mode register (POMxx). For details, please refer to "Chapter 2 Pin Functions"

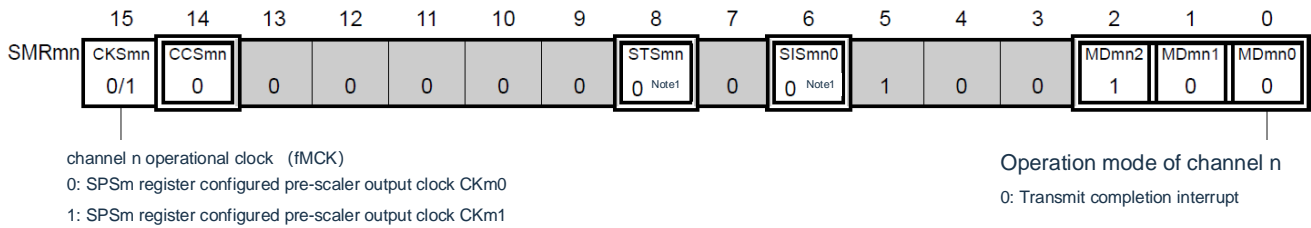
2. Must be used within the scope of peripheral functional characteristics (refer to data sheet) that meet this condition and meet the electrical characteristics.

Note m: Unit number (m=0~2)n: Channel number (n=0~3)mn=00~ 03, 10~11, 20~21

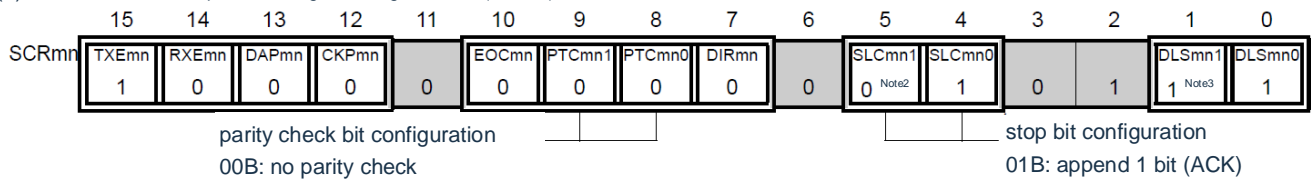
(1) Register settings

Figure 19-117 Simple I²C (IIC00, IIC01, IIC10, IIC10 IIC11, IIC20, IIC21, IIC30, IIC31) when the address segment is sent Example of register setting content

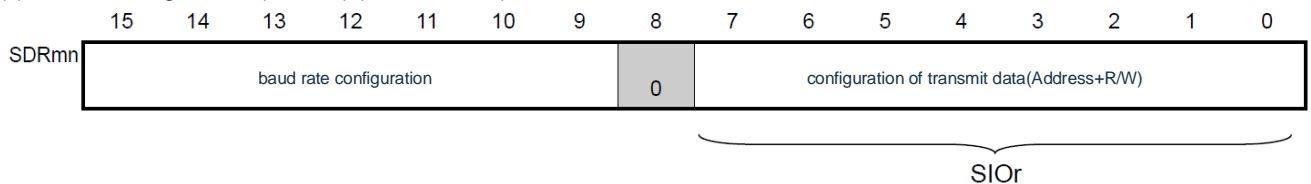
(a) serial mode register mn (SMRmn)



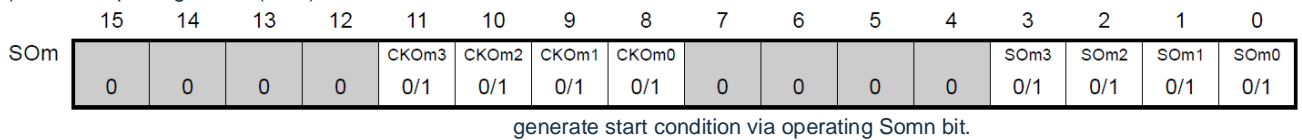
(b) serial communication operation configuration registermn (SCRmn)



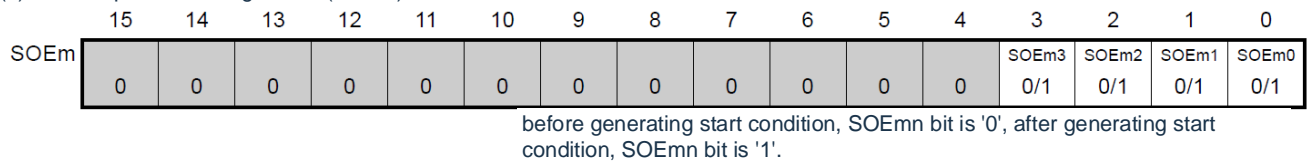
(c) serial data register mn (SDRmn) (low 8 bit: SIOr)



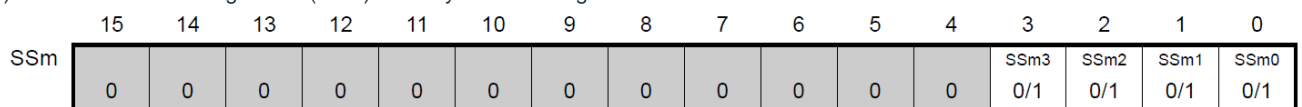
(d) serial output register m (Som)



(e)serial otuput enable register m (SOEm)



(f) serial channel start register m (SSm) Only set bit of target channel to 1.



Note 1 Limited to SMR00, SMR03, SMR11, SMR21.

2. Limited to SCR00, SCR02, SCR10, SCR20.

3. Limited to SCR00 registers and SCR01 registers, other fixed as "1".

Remark1.m:unit number(m=0~2)n:channel number(n=0~3)r:IIC number(r=00, 01, 10, 11, 20, 21, 30, 31)

mn=00~03, 10~11, 20~21

2. ☐ : Fixed setting in IIC mode. ☐ : Cannot be set (initial value is set).

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

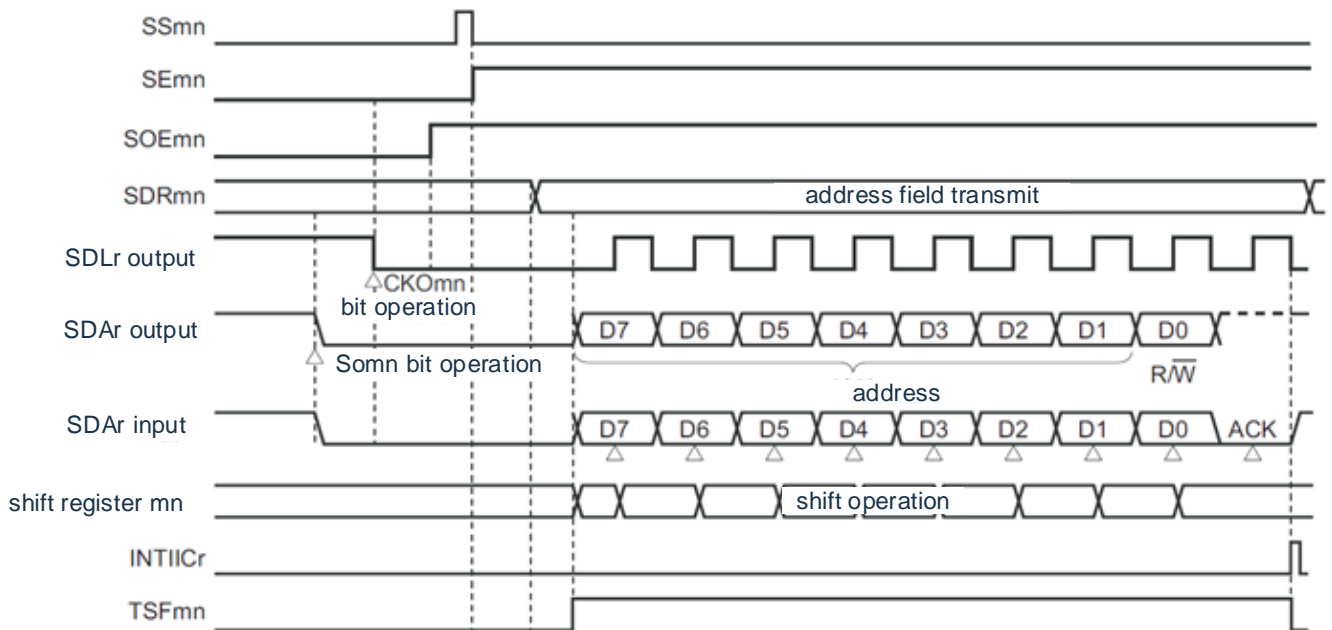
Figure 19-118 The initial setup steps for the transmission of the address segment



Note: At the end of the initial setup, the simple I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21, IIC30, IIC31) are disabled outputs and are in a stop state.

(3) Process flow

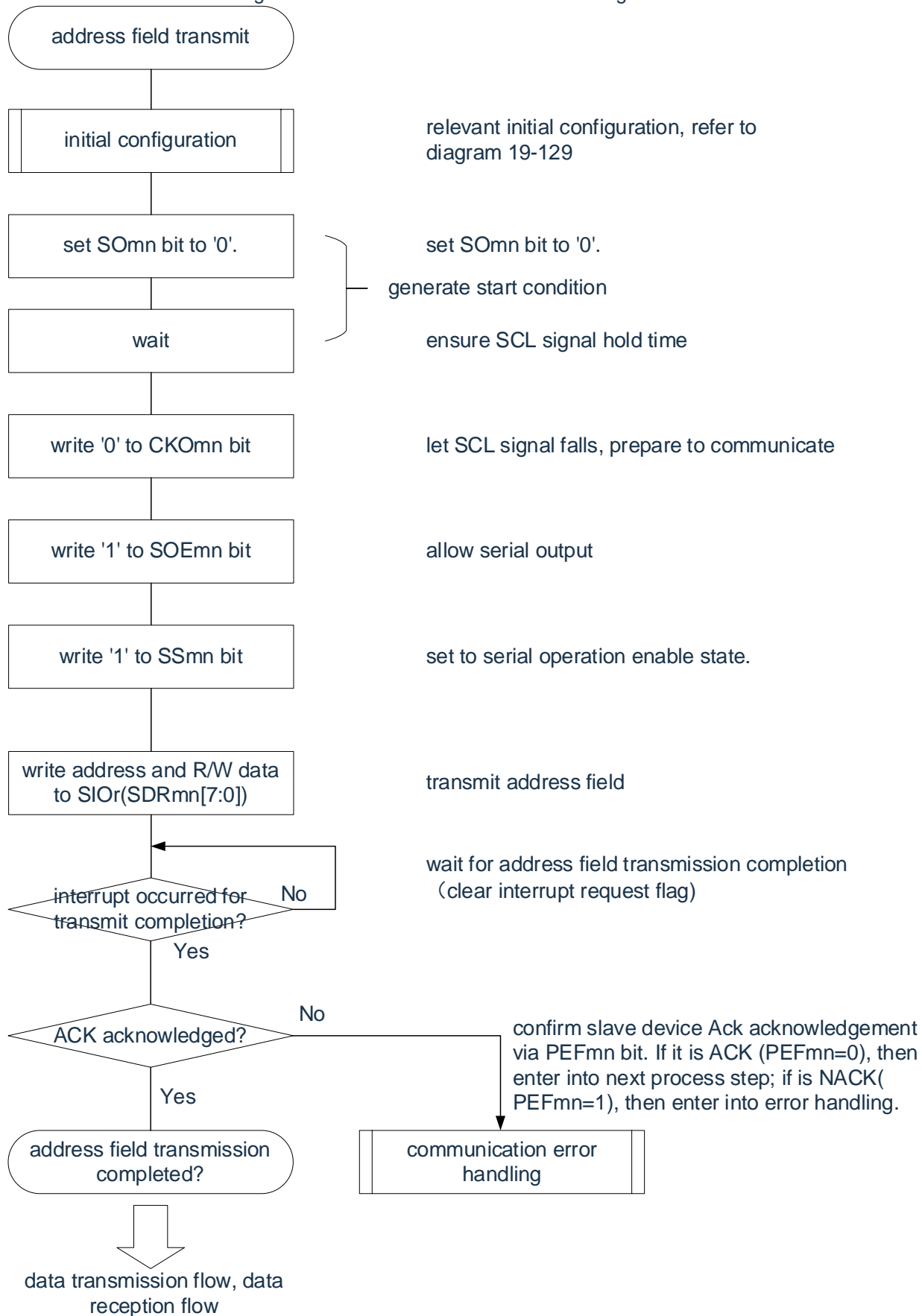
Figure 19-119 Timing diagram of the address segment sent



Remark m: unit number(m=0~2) n:channel number(n=0~3)r:IIC number(r=00, 01, 10, 11, 20, 21, 30, 31)

mn=00~03, 10~11, 20~21

Figure 19-120: Flowchart of the address segment sent



19.9.2 Data sending

Data sending is the operation of sending data to the transmitting object (slave) after sending an address segment. A stop condition is generated and the bus is released after all data is sent to the object slave.

Simple I ² C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21	IIC30	IIC31
Object channels	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1	SCI2 Channel 0	SCI2 Channel 1
The pins used	SCL00, SDA00 ^{note1}	SCL01, SDA01 ^{note1}	SCL10, SDA10 ^{note1}	SCL11, SDA11 ^{note1}	SCL20, SDA20 ^{note1}	SCL21, SDA21 ^{note1}	SCL30, SDA30 ^{note1}	SCL31, SDA31 ^{note1}
interrupt	INTIIC00	INTIIC01	INTIIC10	INTIIC11	INTIIC20	INTIIC21	INTIIC30	INTIIC31
	Limited to end-of-transmit interrupts (buffer null interrupts cannot be selected).							
Error detection flags	ACK Error Flag (PEFmn).							
The length of the transferred data	8 bits							
Transfer rate ^{note2}	Max.f _{MCK} /4[Hz] (SDRmn[15:9] ≥ 1) f _{MCK} : The operating clock frequency of the object channel However, the following conditions must be met in each mode of I ² C: • Max.1MHz (Enhanced Quick Mode). • Max.400kHz (fast mode). • Max.100kHz (standard mode).							
Data level	Normal-phase output (default: high).							
Parity bits	There are no parity bits.							
Stop bit	Additional 1 bit (for ACK reception).							
Data direction	MSB takes precedence							

Note 1 To communicate via Simple I²C, the N-channel open-drain output mode (POMxx=1) must be set via the port output mode register (POMxx). For details, please refer to "Registers for 2.3 Control Port Function" and "Register Settings for 2.5 Using the Multiplexing Function".

2. Must be used within the scope of peripheral functional characteristics (refer to data sheet) that meet this condition and meet the electrical characteristics.

Note m: Unit number (m=0~2)n: Channel number (n=0~3)mn=00~ 03, 10~11, 20~21

(1) Register settings

Figure 19-121 Simple I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21, IIC30, IIC31) when the data is sent
Example of register setting content

(a) serial mode register mn (SMRmn).....do not operate this register while data is transmitting or receiving.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKSmn	CCSmn						STSmn		SISmn0				MDmn2	MDmn1	MDmn0
0/1	0	0	0	0	0	0	0 ^{Note1}	0	0 ^{Note1}	1	0	0	1	0	0

(b) serial communication operation configuration register mn (SCRmn).....do not operate bits other than TXEmn and RXEmn of this register while data is transmitting or receiving.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXEmn	RXEmn	DAPmn	CKPmn		EOCmn	PTCmn1	PTCmn0	DIRmn		SLCmn1	SLCmn0			DLSmn1	DLSmn0
1	0	0	0	0	0	0	0	0	0	0 ^{Note2}	1	0	1	1 ^{Note3}	1

(c) serial data register mn (SDRmn) (low 8 bit: SIOr)only lower 8 bits valid while data is transmitting or receiving.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
baud rate configuration ^{Note4}								0	configuration of transmit data						
SIOr															

(d) serial output register m (Som)do not operate this register while data is transmitting or receiving.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				CKOm3	CKOm2	CKOm1	CKOm0					SOM3	SOM2	SOM1	SOM0
0	0	0	0	0/1 ^{Note5}	0/1 ^{Note5}	0/1 ^{Note5}	0/1 ^{Note5}	0	0	0	0	0/1 ^{Note5}	0/1 ^{Note5}	0/1 ^{Note5}	0/1 ^{Note5}

(e) serial output enable register m (SOEm)do not operate this register while data is transmitting or receiving.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												SOEm3	SOEm2	SOEm1	SOEm0
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

(f) serial channel start register m (SSm)do not operate this register while data is transmitting or receiving.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												SSm3	SSm2	SSm1	SSm0
0	0	0	0	0	0	0	0	0	0	0	0	0/1	0/1	0/1	0/1

Note 1 Limited to SMR01, SMR03, SMR11, SMR21 registers.

2. Limited to SCR00, SCR02, SCR10, SCR20 registers.

3. Limited to SCR00 registers and SCR01 registers, other fixed as "1".

4. Because it has already been set when sending the address segment, it does not need to be set.

5. During the operation of the communication, the value changes due to the communication data.

Remark1.m:unit number(m=0~2)n:channel number(n=0~3)r:IIC number(r=00, 01, 10, 11, 20, 21, 30, 31)

mn=00~03, 10~11, 20~21

2. : Fixed setting in IIC mode. : Cannot be set (initial value is set).

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 19-122 Timing diagram of data transmission

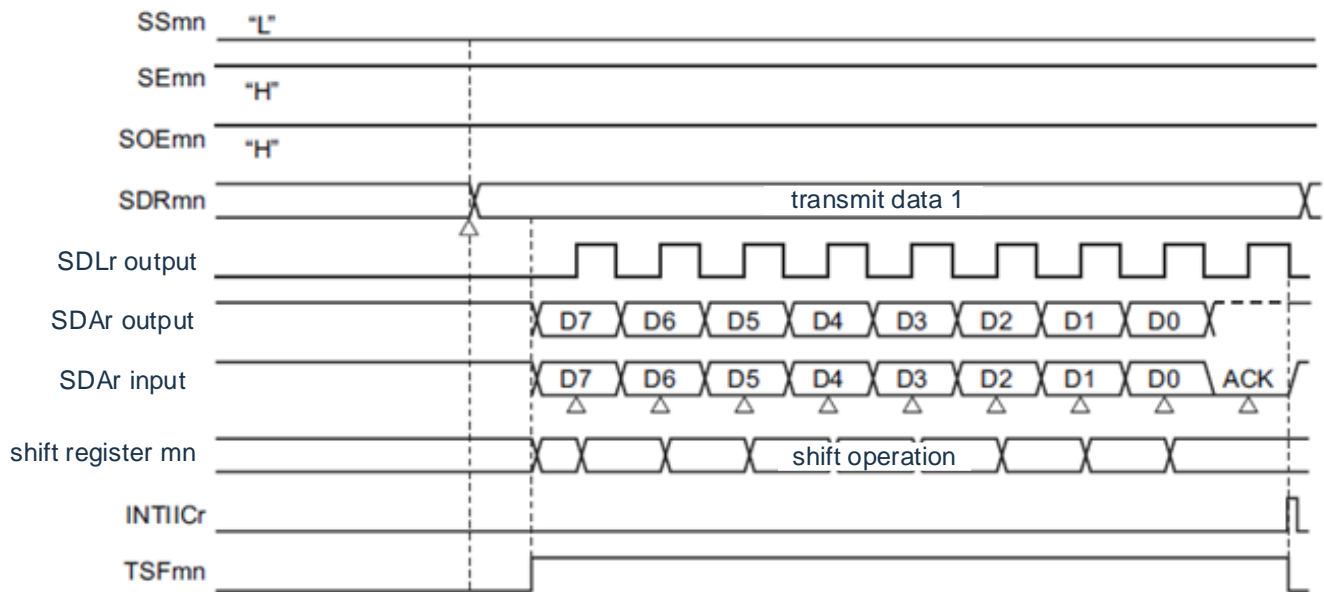
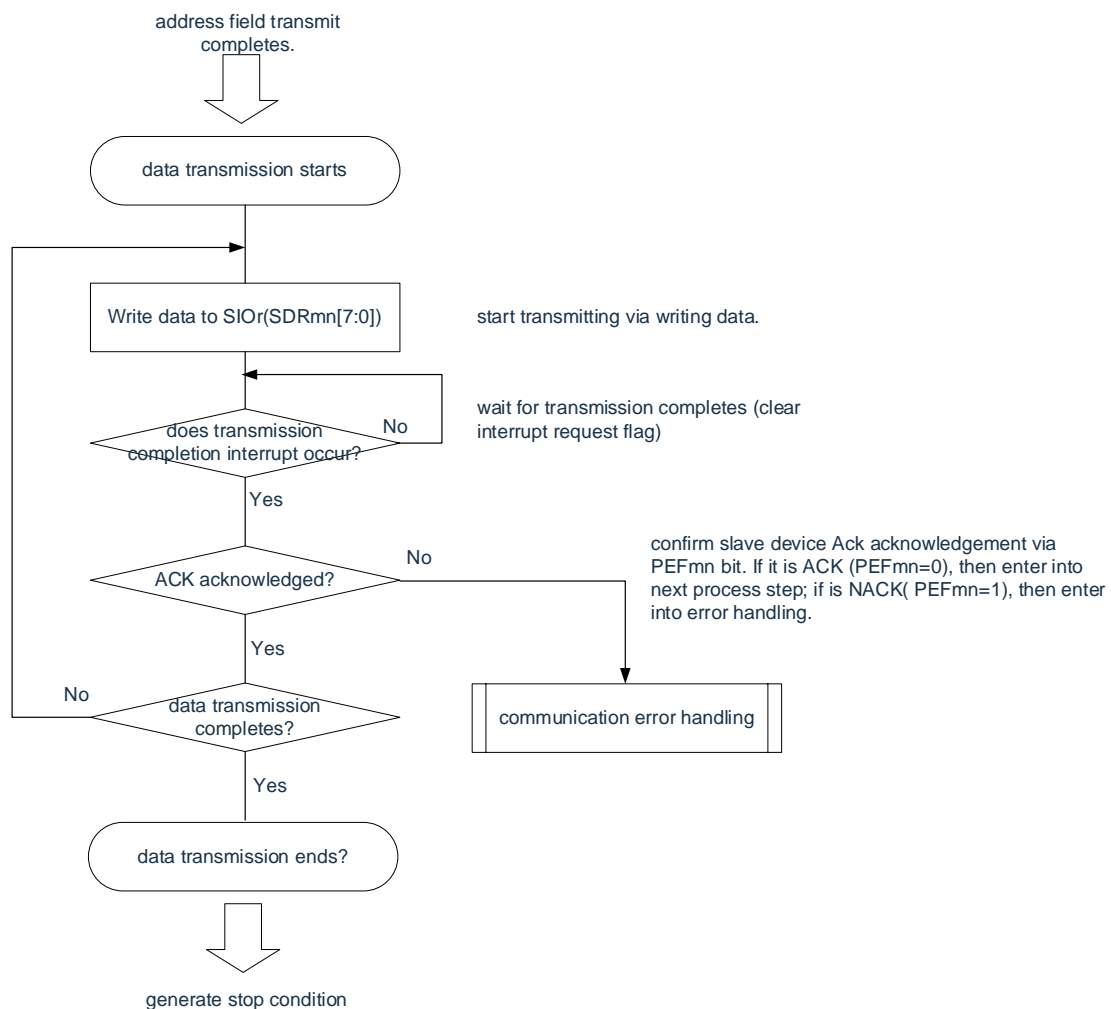


Figure 19-123 Flowchart of data transmission



19.9.3 Data reception

Data ingestion is the operation of receiving data from a transmitting object (slave) after sending an address segment. Generates a stop condition and releases the bus after the slave receives all the data.

Simple I ² C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21	IIC30	IIC31
Object channels	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1	SCI2 Channel 0	SCI2 Channel 1
The pins used	SCL00, SDA00 ^{note1}	SCL01, SDA01 ^{note1}	SCL10, SDA10 ^{note1}	SCL11, SDA11 ^{note1}	SCL20, SDA20 ^{note1}	SCL21, SDA21 ^{note1}	SCL30, SDA30 ^{note1}	SCL31, SDA31 ^{note1}
interrupt	INTIIC00	INTIIC01	INTIIC10	INTIIC11	INTIIC20	INTIIC21	INTIIC30	INTIIC31
	Limited to end-of-transmit interrupts (buffer null interrupts cannot be selected).							
Error detection flags	There are only overflow error detection flags (OVFmn).							
The length of the transferred data	8 bits							
Transfer rate ^{note 2}	Max. $f_{MCK}/4$ [Hz] ($SDRmn[15:9] \geq 1$) f_{MCK} : The operating clock frequency of the object channel However, the following conditions must be met in each mode of I ² C: • Max.1MHz (Enhanced Quick Mode). • Max.400kHz (fast mode). • Max.100kHz (standard mode).							
Data level	Normal-phase output (default: high).							
Parity bits	There are no parity bits.							
Stop bit	Additional 1 bit (ACK sending).							
Data direction	MSB takes precedence							

Note 1 To communicate via Simple I²C, the N-channel open-drain output mode (POMxx=1) must be set via the port output mode register (POMxx). For details, please refer to "Registers for 2.3 Control Port Function" and "Register Settings for 2.5 Using the Multiplexing Function".

2. Must be used within the scope of peripheral functional characteristics (refer to data sheet) that meet this condition and meet the electrical characteristics.

Note m: Unit number (m=0~2)n: Channel number (n=0~3)mn=00~ 03, 10~11, 20~21

(1) Register settings

Figure 19-124 Simple I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21, IIC30, IIC31) when the data is received
Example of register setting content

(a) serial mode register mn (SMRmn).....do not operate this register while data is transmitting or receiving.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn	CCSmn						STSmn		SISmn0				MDmn2	MDmn1	MDmn0
	0/1	0	0	0	0	0	0	0 <small>Note1</small>	0	0 <small>Note1</small>	1	0	0	1	0	0

(b) serial communication operation configuration register mn (SCRmn).....do not operate bits other than TXEmn and RXEmn of this register while data is transmitting or receiving.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn	RXEmn	DAPmn	CKPmn		EOCmn	PTCmn1	PTCmn0	DIRmn		SLCmn1	SLCmn0			DLSmn1	DLSmn0
	0	1	0	0	0	0	0	0	0	0	0 <small>Note2</small>	1	0	1	1 <small>Note3</small>	1

(c) serial data register mn (SDRmn) (low 8 bit: SIO_r)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration <small>Note4</small>								0	virtual transmit data configuration (FFH)						

SIO_r

(d) serial output register m (Som)do not operate this register while data is transmitting or receiving.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Som					CKOm3	CKOm2	CKOm1	CKOm0					SOm3	SOm2	SOm1	SOm0
	0	0	0	0	0/1 <small>Note5</small>	0/1 <small>Note5</small>	0/1 <small>Note5</small>	0/1 <small>Note5</small>	0	0	0	0	0/1 <small>Note5</small>	0/1 <small>Note5</small>	0/1 <small>Note5</small>	0/1 <small>Note5</small>

(e) serial output enable register m (SOEm)do not operate this register while data is transmitting or receiving.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm													SOEm3	SOEm2	SOEm1	SOEm0
	0	0	0	0	0	0	0	0	0	0	0	0	0/1	0/1	0/1	0/1

(f) serial channel start register m (SSm)do not operate this register while data is transmitting or receiving.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm													SSm3	SSm2	SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	0/1	0/1	0/1	0/1

Note 1 Limited to SMR01, SMR03, SMR11, SMR21 registers.

2. Limited to SCR00, SCR02, SCR10, SCR20 registers.

3. Limited to SCR00 registers and SCR01 registers, other fixed as "1".

4. Because it has already been set when sending the address segment, it does not need to be set.

5. During the operation of the communication, the value changes due to the communication data.

Remark1.m:unit number(m=0~2)n:channel number(n=0~3)r:IIC number(r=00, 01, 10, 11, 20, 21, 30, 31)

mn=00~03, 10~11, 20~21

2. ☐ : Fixed setting in IIC mode. ☐ : Cannot be set (initial value is set).

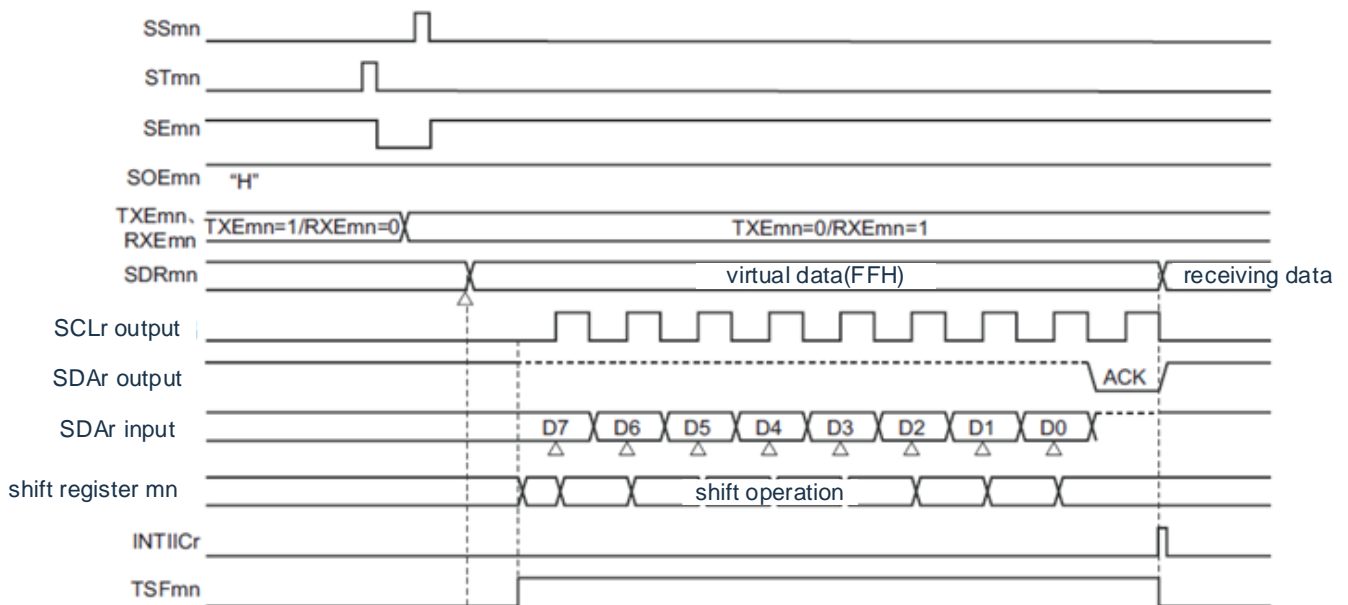
x: This is a bit that cannot be used in this mode (and the initial value is set if it is not used in other modes).

0/1: Set "0" or "1" according to the user's purpose.

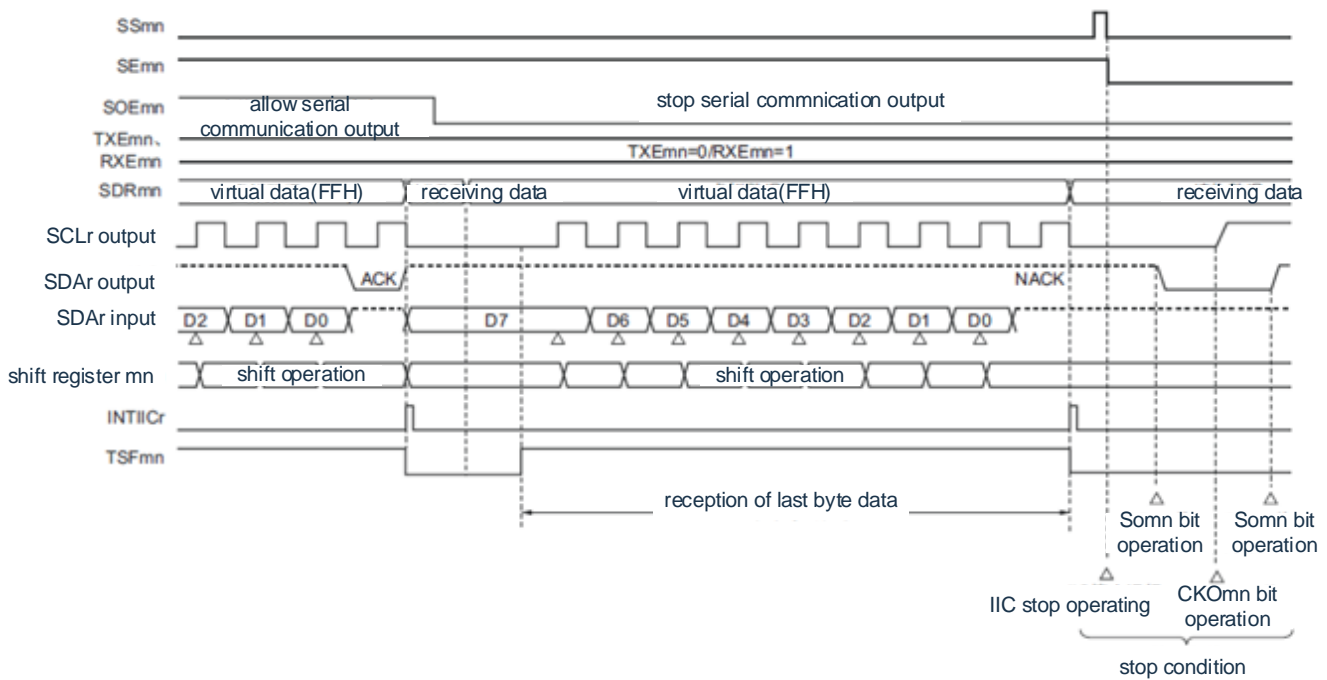
(2) Process flow

Figure 19-125 Timing diagram of data reception

(a) The case when you start receiving data



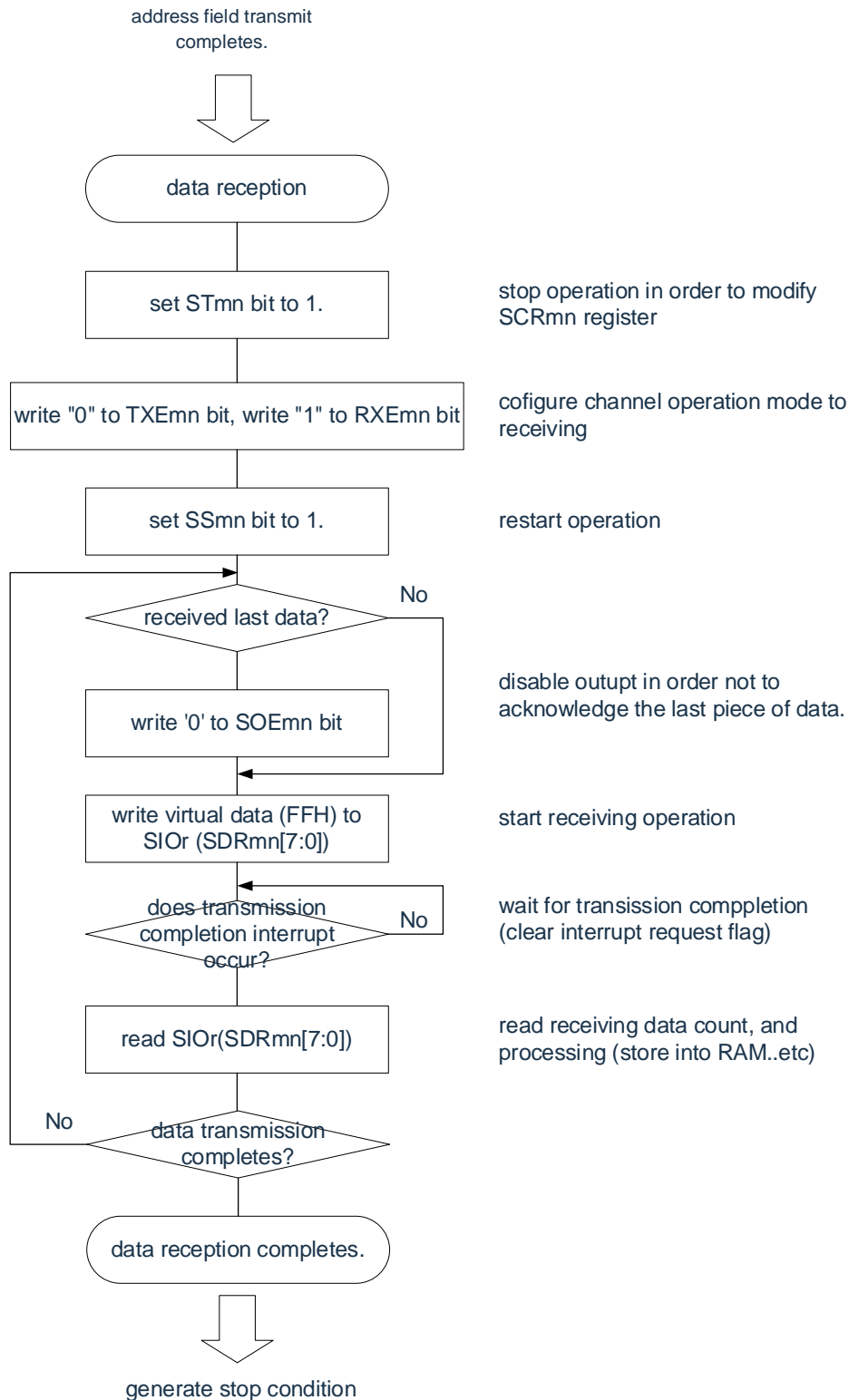
(b) The case of receiving the last data



Remark m: unit number(m=0~2)n:channel number(n=0~3)r:IIC number(r=00, 01, 10, 11, 20, 21, 30, 31)

mn=00~03, 10~11, 20~21

Fig. 19-126 Flowchart of data reception



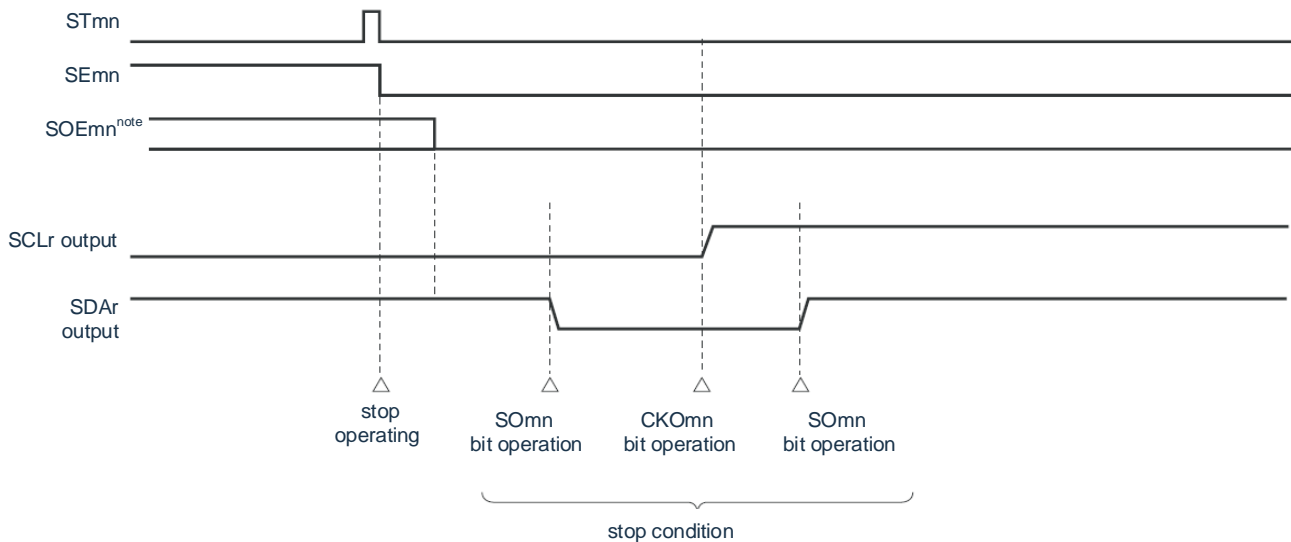
Note that the ACK (NACK) is not output when the last data is received. Thereafter, the operation is first stopped by placing the STmn position "1" of the serial channel stop register m(STm), and then the stop condition is generated to end the communication.

19.9.4 Stop conditions generation

After all data has been sent and received with the object slave, a stop condition is generated and the bus is released.

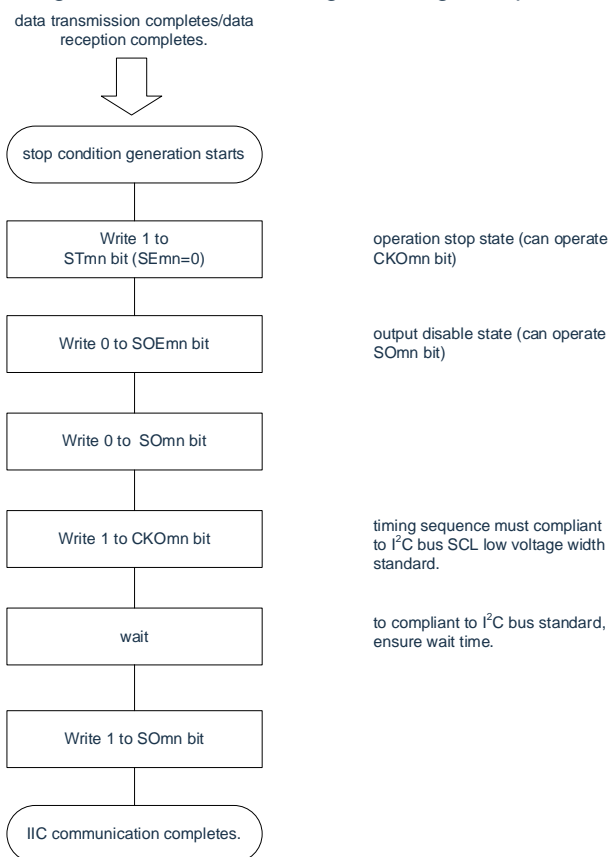
(1) Process flow

Figure 19-127: A timing diagram of the generation stop condition



Note That at the time of reception, the serial output enable register m (SOEm) of the SOEmn position "0" before receiving the last data.

Fig. 19-128 Flowchart of generating a stop condition



19.9.5 Calculation of the transfer rate

Simple I²C (IIC00, IIC01, IIC10, IIC11, IIC20, The transmission rate of IIC21, IIC30, IIC31) communication can be calculated using the following calculation equation.

$$(\text{Transfer Rate}) = \{\text{Runtime Clock } (f_{\text{MCK}}) \text{ Frequency}\} (\text{SDRmn} \div [15:9] + 1) \div 2$$

Note It is forbidden to set SDRmn[15:9] to "0000000B", and the setting value of SDRmn[15:9] must be greater than or equal to "0000001B". The SCL signal output of simple I²C has a duty cycle of 50%. In the I²C-bus specification, the low-level width of the SCL signal is greater than the high-level width. Therefore, if set to 400kbps in fast mode or 1Mbps in enhanced fast mode, the low level width of the SCL signal output is less than I²C The specification value of the bus. SDRmn [15:9] must be set to meet the I²C-bus specifications.

Note 1 Because the value of SDRmn[15:9] is the value of bit15~9 of the serial data register (SDRmn) (0000001B~1111111B).), so it is 1 to 127.

2.m:unit number(m=0~2)n:channel number(n=0~3)mn=00~03, 10~11, 20~21

The operating clock (f_{MCK}) depends on bit15 of the serial clock selection register m (SPSm) and the serial mode register mn (SMRmn). (CKSmn bit).

Table 19-5 Selection of simple I²C operating clocks

SMRmn register	SPSm register								Running Clock (f _{MCK}) Note	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		f _{CLK} =32MHz 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
1	0	0	0	0	X	X	X	X	f _{CLK}	32MHz
	0	0	0	1	X	X	X	X	f _{CLK} /2	16MHz
	0	0	1	0	X	X	X	X	f _{CLK} /2 ²	8MHz
	0	0	1	1	X	X	X	X	f _{CLK} /2 ³	4MHz
	0	1	0	0	X	X	X	X	f _{CLK} /2 ⁴	2MHz
	0	1	0	1	X	X	X	X	f _{CLK} /2 ⁵	1MHz
	0	1	1	0	X	X	X	X	f _{CLK} /2 ⁶	500kHz
	0	1	1	1	X	X	X	X	f _{CLK} /2 ⁷	250kHz
	1	0	0	0	X	X	X	X	f _{CLK} /2 ⁸	125kHz
	1	0	0	1	X	X	X	X	f _{CLK} /2 ⁹	62.5kHz
	1	0	1	0	X	X	X	X	f _{CLK} /2 ¹⁰	31.25kHz
	1	0	1	1	X	X	X	X	f _{CLK} /2 ¹¹	15.63kHz
Other than the above									Disable settings.	

Note 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 Universal Serial Communication Unit (SCI) (serial channel stop register m (STm)=000FH) after making changes.

Note 1.X: Ignore

2.m:unit number(m=0~2)n:channel number(n=0~3)mn=00~03, 10~11, 20~21

An example of the I²C transfer rate at f_{MCK}=f_{CLK}=32MHz is shown below.

I ² C transfer mode (Expected transfer rate)	f _{CLK} =32MHz			
	Running Clock (f _{MCK})	SDRmn[15:9]	Calculated transfer rate	Error with the expected transfer rate
100kHz	f _{CLK} /2	79	100kHz	0.0%
400kHz	f _{CLK}	41	380kHz	5.0% note
1MHz	f _{CLK}	18	0.84MHz	16.0% note

Note That because the duty cycle of the SCL signal is 50%, the error cannot be set to about "0"%.

19.9.6 In simple I2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21, IIC30, IIC31) Processing steps when an error occurs during communication

In simple I2C (IIC00, IIC01, IIC10, IIC11, IIC20, The processing steps for errors that occur during IIC21, IIC 3 0, IIC31) are Figure 19-129and Figure 19-130shown.

Figure 19-129 Processing steps when an overflow error occurs

Software operation	Hardware status	remark
Read the serial data register mn (SDRmn). →	The BFFmn bit of the SSR mn register is "0" and channel n is in a receivable state.	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 type of class that is judged incorrectly, and the read value is used to clear the error flag.
Clear the trigger register mn to the serial flag →	Clears 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 19-130 Processing steps when an ACK error occurs in simple I2C mode

Software operation	Hardware status	remark
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 →	Clears 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.
Stop the serial channel register m(STm). STmn position "1".	→ The serial channel allows the SEm n bit of the status register m(SEm) to be "0" and the channel n is the run stopped state.	Because ACK is not returned, the slave device is not prepared for reception. As a result, a stop condition is generated and the bus is released, and the communication is started again from the start condition, or a restart can also be generated Conditions and re-proceeds starting from the address sending.
Cease to be generated.		
Generate start conditions.		
Place the serial channel starting register m(SSm). SSmn position "1".	→ The serial channel allows the SEm n bit of the status register m(SEm) to be "1" and channel n to be operational.	

Remark m: unit number(m=0~2)n:channel number(n=0~3)r:IIC number(r=00, 01, 10, 11, 20, 21, 30, 31)

mn=00~03, 10~11, 20~21

Chapter 20 Serial interface IICA

20.1 The serial interface IICA functions

This product is equipped with two serial interfaces IICA0, IICA1, and has the following three modes.

20.1.1 Run stop mode

This is a mode used when serial transfer is not in progress and reduces power consumption.

20.1.2 I2 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 I2C-bus format, the master device can generate "start conditions" and "addresses" for the slave device on the serial data bus, Indication of Transfer Direction, Data, and Stop Condition. The slave automatically detects the received status and data through the hardware. This feature simplifies the I2 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 lifted 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 20-1.

Note n = 0,1

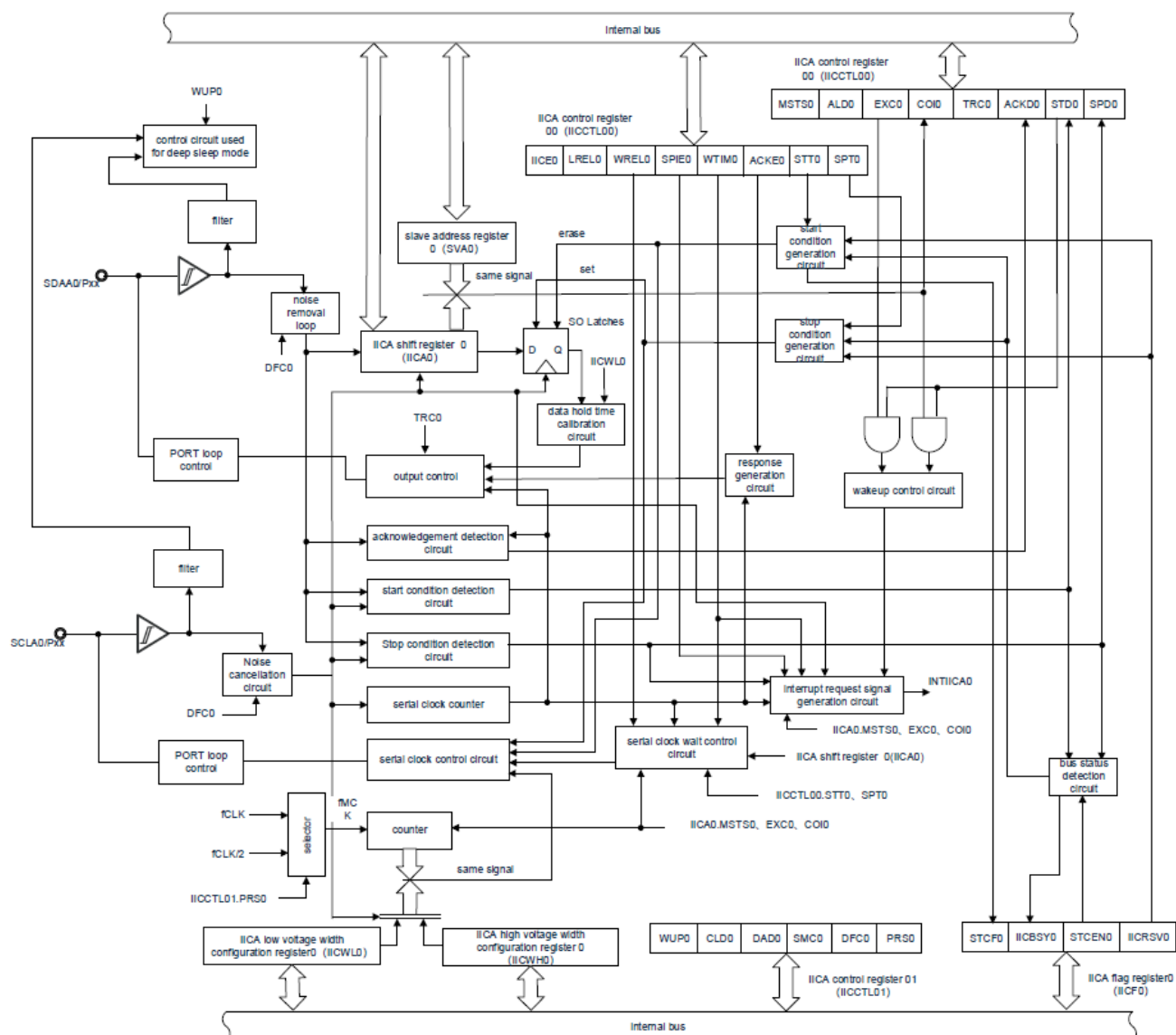
20.1.3 Wake-up mode

In deep sleep mode, when receiving an extension code or local station address of the autonomous control device, the deep sleep mode can be lifted 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 20-1201.

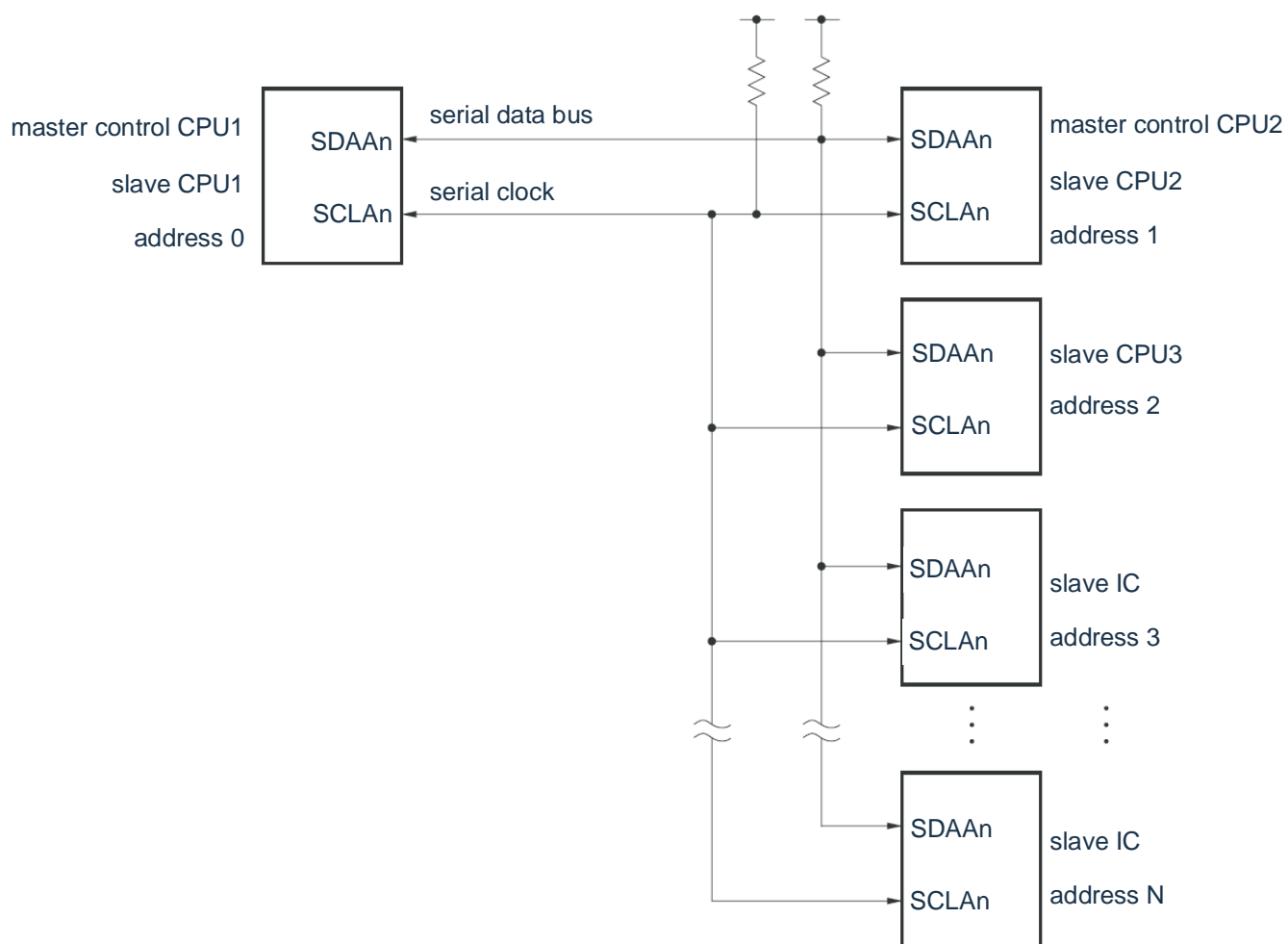
Note n = 0,1

Figure 20-1 diagram of the serial interface IICA



An example of the structure of a serial bus is shown in Figure 20-2.

Figure 20-2 example of a serial bus structure for I2C-bus



Note: n=0,1

20.2 Structure of the serial interface IICA

The serial interface IICA consists of the following hardware.

Table 20-1 Structure of the serial interface IICA

item	structure
register	IICA shift register n (IICAn) slave address register n (SVAn).
Control registers	Perimeter 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).

Note: 1, n=0,1

- 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:

Address of base	Offset address	Name of register	R/W	Value of reset
0x40041800	0x350	IICA0	R/W	00H
	0x351	IICS0	R	00H
	0x352	IICF0	R/W	00H
	0x230	IICCTL00	R/W	00H
	0x231	IICCTL01	R/W	00H
	0x232	IICWL0	R/W	FFH
	0x233	IICWH0	R/W	FFH
	0x234	SVA0	R/W	00H
0x40042C00	0x350	IICA1	R/W	00H
	0x351	IICS1	R	00H
	0x352	IICF1	R/W	00H
	0x230	IICCTL10	R/W	00H
	0x231	IICCTL11	R/W	00H
	0x232	IICWL1	R/W	FFH
	0x233	IICWH1	R/W	FFH
	0x234	SVA1	R/W	00H

20.2.1 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 lifted by writing the IICAn register and the data is transferred. The IICAn registers are set via the 8-bit memory operation instructions. After generating a reset signal, the value of this register changes to "00H".

Figure 20-3 Format of the IICAn shift register n (IICAn).

After reset: 00H, R/W								
Symbol	7	6	5	4	3	2	1	0
IICAn								

Note 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.

Note: n=0,1

20.2.2 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 operation instruction. However, when the STDn bit is "1" (start condition detected), it is forbidden to overwrite this register.

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

Figure 20-4 Format of the slave address register n (SVAn).

After reset: 00H, R/W

Symbol	7	6	5	4	3	2	1	0
Swan	A6	A5	A4	A3	A2	A1	A0	0 Note

Note: bit0 is fixed to "0".

20.2.3 SO latches

The SO latch holds the output level of the SDAAn pin.

20.2.4 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.

20.2.5 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.

20.2.6 Interrupt request signal generation circuit

This circuit control generates an interrupt request signal (INTIICAn). An I2C interrupt request is generated by the following two triggers.

- Drop of the 8th or 9th serial clock (set by the WTIMn bit).
- Interrupt request (set via SPIEn bit) due to detection of a stop condition.

Note: WTIMn bit: bit3 of IICA control register n0 (IICCTLn0).

SPIEn bit: bit4 of IICA control register n0 (IICCTLn0).

20.2.7 Serial clock control circuitry

In master mode, this circuit generates the output from the sample clock to the clock of the SCLAn pin.

20.2.8 Serial clock wait control circuit

This circuit controls the wait timing.

20.2.9 Ack generation circuit, stop condition detection circuit, start condition detection circuit, Ack detection circuit

These circuits generate and detect various states.

20.2.10 Data hold time correction circuit

This circuit generates a data hold time for the serial clock to drop.

20.2.11 Start conditional generation circuitry

If stTn is positioned "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 position "1" is changed.

20.2.12 Stop condition generation circuitry

If the SPTn position is "1", the circuit generates a stop condition.

20.2.13 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.

Remarks: 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=0.1

20.3 Controls registers of the serial interface IICA

The serial interface IICA is controlled by the following registers.

- Peripheral enable register 0 (PER0).
- IICA control register n0 (IICCTLn0).
- IICA flag register n (IICFn).
- IICA status register n (IICSn).
- IICA control register n1 (IICCTLn1).
- IICA low level width setting register n (IICWLn).
- IICA high level width setting register n (IICWHn).
- Port Mode Register (PMxx).
- Port Mode Control Register (PMCxx).
- Port Multiplexing Function Configuration Register (PxxCFG).

Note: n=0,1

20.3.1 Peripheral enable register 0/1 (PER0/1).

Per0/1 registers are registers that are set to enable or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use.

To use the serial interface IICA0, you must replace PER0 Bit5 (IICA0EN) is set to "1".

To use the serial interface IICA1, you must replace PER1 Bit2 (IICA1EN) is set to "1".

The PER0 register is set via the 8-bit memory operation instruction.

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

For more information about registers, see "4.3.6 Peripheral Allowable Registers 0, 1, 1 (PER0, PER1, PER1) ".

Note 1 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).

Note: n=0,1

20.3.2 IICA control register n0 (IICCTLn0).

This is a register that starts or stops I²C operation, sets the wait sequence, and sets other I²C operations.

The IICCTLn0 register is set by the 8-bit memory operation 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 generating a reset signal, the value of this register changes to "00H".

Note: n=0,1

Figure 20-5 Format of IICA control register n0 (IICCTLn0) (1/4).

After reset: 00HR/W

symbol	7	6	5	4	3	2	1	0
IICCTLn0	IICEn	LRELn	WRELn	SPIEn	WTIMn	ACKEn	STTn	SPTn

IICEn	I ² C runs allowed
0	Stop running. Reset ^{note 1} to IICA status register n (IICSn) and stop internal operation.
1	Allowed to run.
This position "1" must be placed in the state where the SCLAn and SDAAn lines are high.	
Clear condition (IICEn=0,1).	
Set condition (IICEn=1).	
<ul style="list-style-type: none"> • Clear by command. • When resetting 	
<ul style="list-style-type: none"> • Set by command. 	

LRELn ^{Notes 2 and 3}	Exit of communication
0	Usually runs
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•MSTSn•EXCn•COIn•TRCn•ACKDn•STDn
Becomes a standby state to exit the communication until the following communication participation conditions are met.	
<ul style="list-style-type: none"> •Boot as master device after detecting a stop condition. •Addresses match or extended codes are received after the start condition is detected. 	
Clear condition (LRELn=0,1).	
Set condition (LRELn=1).	
<ul style="list-style-type: none"> •Automatically clears after execution. • When resetting 	
<ul style="list-style-type: none"> • Set by command. 	

WRELn ^{notes}	Pending release
0	Do not dismiss the wait.
1	Lift the wait. Clears automatically after the wait is lifted.
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=0,1).	
Clear condition (WRELn=0,1).	
Set condition (WRELn=1).	
<ul style="list-style-type: none"> •Automatically clears after execution. • When resetting 	
<ul style="list-style-type: none"> • 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.

2. In the state where the IICEn bit is "0", the signal for this bit is invalid.

3. The read values for LRELn bits and WRELn bits are always "0".

Note: If the SCLAn line is high, the SDAAn line is low, and the digital filter is ON (DFCn=1 for the IICCTLn1 register), then enable I²C operation (IICEn=1) immediately detects the start condition. At this point, the LRELn position "1" must be placed continuously through the bit memory operation instruction after allowing I²C to run (IICEn=1).

Note: n=0,1

Figure 20-6 Format of IICA control register n0 (IICCTLn0) (2/4).

SPIEn ^{Note 1}	Allow or disallow interrupt requests resulting from stop condition detection
0	forbid
1	allow
When the WUPn bit of IICA control register n1 (IICCTLn1) is "1", even if the SPIEN position is "1" There is also no stop condition interrupt.	
Clear condition (SPIEn=0,1).	Set condition (SPIEn=1).
• Clear by command. • When resetting	• Set by command.

WTIMn ^{Note 1}	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=0,1).	Position condition (WTIMn=1).
• Clear by command. • When resetting	• Set by command.

ACKEn ^{Notes 1, 2}	Reply control
0	No Ack.
1	Allow Acks. Set the SDAAn line low during the 9th clock.
Clear condition (ACKEn=0,1).	Set condition (ACKEn=1).
• Clear by command. • When resetting	• 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.

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.

Note: n=0,1

Figure 20-6 Format of IICA control register n0 (IICCTLn0) (3/4).

STTn ^{note1, 2}	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 position is "1", a start condition (boot as the master device) is generated. When a third party is communicating:</p> <ul style="list-style-type: none"> • Allow communication in the case of the reservation function (IICRSVn=0,1). Used as a start condition reservation sign. If you place this position "1", a start condition is automatically generated just after the bus is released. • The case where the communication reservation function is prohibited (IICRSVn=1). Even if this position 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 lifted.
<p>Notes on position timing:</p> <ul style="list-style-type: none"> • Master Receive: Disables this position "1" during transmission. This position "1" can only be placed during the waiting period when ACKEn is at position "0" and notifying the slave that receiving it has completed. • Master Send: During the Ack, the start condition may not be generated properly. This position "1" must be placed during the wait period after the 9th clock is output. • Prohibit and Trigger of Stop Condition (SPTn) with "1" at the same time. • After placing stTn at the position "1", it is forbidden to use this bit "1" again until the purge condition is met. 	
Clear condition (STTn=0,1).	Position condition (STTn=1).
<ul style="list-style-type: none"> • Place stTn at the position "1" in a state where communication reservation is prohibited. • When arbitration fails • Master device generation start condition. • Cleared because the LRELn bit is "1" (Exit Communication). • When the IICEn bit is "0" (stop running). • When resetting 	<ul style="list-style-type: none"> • Set by command.

Note 1 In the state where the IICEn bit is "0", the signal for this bit is invalid.

2. The read value of the STTn bit is always "0".

Remarks: 1 If bit1 (STTn) is read after setting the data, this bit becomes "0".

IICRSVn : Bit0 of IICA flag register n (IICFn).

STCFn : Bit7 of IICA flag register n (IICFn).

3. n=0.1

Figure 20-6 Format of IICA control register n0 (IICCTLn0) (4/4).

SPTn Note	The trigger of the stop condition	
0	No stop conditions are generated.	
1	Generate a stop condition (end of transfer as master).	
Notes on position timing:		
<ul style="list-style-type: none">• Master Receive: Disables this position "1" during transmission. This position "1" can only be placed during the waiting period when ACKEn is at position "0" and notifying the slave that receiving it has completed.• Master Send: During the Ack, the stop condition may not be generated properly. This position "1" must be placed during the wait period after the 9th clock is output.• Prohibits placing a "1" simultaneously with the trigger of the start condition (STTn).• SPTn can only be placed in the case of the master device "1".• When the WTIMn bit is "0", it must be noted that if the SPTn position "1" is placed during the wait after 8 clocks are output, it is the 9th after the wait is lifted Stop conditions are generated during high levels of clocks. The WTIMn bit must be set from "0" to "1" during the wait period after the output of 8 clocks and during the wait period after the output of the 9th clock SPTn position "1".• After placing the SPTn position "1", it is forbidden to place this position "1" again until the purge condition is met.		
Purge condition (SPTn=0,1).		Position condition (SPTn=1).
<ul style="list-style-type: none">• When arbitration fails• Clear automatically when a stop condition is detected.• Cleared because the LRELn bit is "1" (Exit Communication).• When the IICEn bit is "0" (stop running).• When resetting		<ul style="list-style-type: none">• Set by command.

Note: The read value for the SPTn bit is always "0".

Note: 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 SDAA_n line to high impedance. Waiting for the TRCn bit of "1" (send state) must be released by writing the IICA shift register n.

Note: n=0,1

20.3.3 IICA status register n (IICS_n).

This is the register that represents the I²C state.

The IICS_n register can only be read by the 8-bit memory operation instruction during the STT_n bit "1" and waiting. After generating a reset signal, the value of this register changes to "00H".

Note: In the allowed address matching wake function (WUP_n=1) state in deep sleep mode, reading the IICS_n register is prohibited. In the state where the WUP_n bit is "1", it has nothing to do with the INTIICAn interrupt request if you change the WUP_n 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 IICS_n register after the interrupt is detected.

remark STT_n : Bit1 of IICA control register n0 (IICCTLn0).

WUP_n : Bit7 of IICA control register n1 (IICCTLn1).

Figure 20-6 Format of IICA status register n (IICS_n) (1/3).

After reset: 00H		R							
symbol	7	6	5	4	3	2	1	0	
IICS _n	MSTS _n	ALD _n	EXC _n	COI _n	TRC _n	ACKD _n	STD _n	SPD _n	
MSTS _n		Confirmation flag for the master status							
0		Slave state or communication standby							
1		Master communication status							
Clear condition (MSTS _n =0,1).						Set condition (MSTS _n =1).			
<ul style="list-style-type: none">•When a stop condition is detected• When the ALD_n bit is "1" (arbitration failed).• Cleared because the LREL_n bit is "1" (Exit Communication).• When the IICEn bit changes from "1" to "0" (stops running).• When resetting						<ul style="list-style-type: none">•When the build starts condition			

ALD _n		Detection of arbitration failures						
0		Indicates that no arbitration occurred or was won.						
1		Indicates that the arbitration failed. Clear the MSTS _n bit.						
Clear condition (ALD _n =0,1).						Set condition (ALD _n =1).		
<ul style="list-style-type: none">• Automatically clears the note after reading the IICS_n register.• When the IICEn bit changes from "1" to "0" (stops running).• When resetting						<ul style="list-style-type: none">• When arbitration fails		

Note: This bit is cleared even if the bit memory manipulation instruction is executed on a bit other than the IICS_n register. Therefore, when using aldn bits, the data for the ALD_n bits must be read before reading other bits.

Remarks: 1. LREL_n: Bit6 of IICA control register n0 (IICCTLn0).

IICEn: Bit7 of IICA control register n0 (IICCTLn0).

2. n=0.1

Figure 20-7 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=0,1).		Position condition (EXCn=1).
<ul style="list-style-type: none"> •When a start condition is detected •When a stop condition is detected •Cleared because the LRELn bit is "1" (Exit Communication). •When the IICEn bit changes from "1" to "0" (stops running). •When resetting 		<ul style="list-style-type: none"> • When the high 4 bits of the received address data are "0000" or "1111" (asserted on the rising edge of the 8th clock).

COIn	Detection of address matches	
0	The address is different.	
1	The address is the same.	
Clear condition (COIn=0,1).		Set condition (COIn=1).
<ul style="list-style-type: none"> •When a start condition is detected •When a stop condition is detected •Cleared because the LRELn bit is "1" (Exit Communication). •When the IICEn bit changes from "1" to "0" (stops running). •When resetting 		<ul style="list-style-type: none"> • When receiving address and local station address (slave address register n (SVAn)) are the same (asserted on the rising edge of the 8th clock).

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=0,1).		Position condition (TRCn=1).
<p>< master and slave ></p> <ul style="list-style-type: none"> •When a stop condition is detected •Cleared because the LRELn bit is "1" (Exit Communication). •When the IICEn bit changes from "1" to "0" (stops running). •Clear note due to WRELn bit "1" (unsheath wait). •When the ALDn bit changes from "0" to "1" (arbitration failed). •When resetting •Cases of non-participation in communication (MSTS_n, EXC_n, COIn=0,1). <p>< the master device ></p> <ul style="list-style-type: none"> •When the LSB (Transmit Direction Indicator bit) of byte 1 outputs "1" <p>< slave ></p> <ul style="list-style-type: none"> •When a start condition is detected •When the LSB (Transmit Direction Indicator bit) of byte 1 enters "0" 		<p>< the master device ></p> <ul style="list-style-type: none"> •When the build starts condition •LSB (transmit direction indication bit) when byte 1 (address is transmitted). <p>When output "0" (master sends).</p> <p>< slave ></p> <ul style="list-style-type: none"> •LSB (Transmit) when the master device is byte 1 (Address Transfer Direction indicator bit) when entering "1" (Slave send).

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.

Note: 1. LRELn: Bit6 of IICA control register n0 (IICCTLn0).

IICEn: Bit7 of IICA control register n0 (IICCTLn0).

2.n=0,1

Figure 20-7 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=0,1).	
<ul style="list-style-type: none"> • When a stop condition is detected • When the next byte of the 1st clock rises • Cleared because the LRELn bit is "1" (Exit Communication). • When the IICEn bit changes from "1" to "0" (stops running). • When resetting 	
Set condition (ACKDn=1).	
<ul style="list-style-type: none"> • Place the SDAAn line low on the 9th clock rising edge of the SCLAn line 	

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=0,1).	
<ul style="list-style-type: none"> • When a stop condition is detected • When the 1st clock rises after the next byte of the address is transmitted • Cleared because the LRELn bit is "1" (Exit Communication). • When the IICEn bit changes from "1" to "0" (stops running). • When resetting 	
Position condition (STDn=1).	
<ul style="list-style-type: none"> • When a start condition is detected 	

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=0,1).	
<ul style="list-style-type: none"> • After this position bit, the address transmits the byte after the start condition is detected when the clock rises • When the WUPn bit changes from "1" to "0" • When the IICEn bit changes from "1" to "0" (stops running). • When resetting 	
Set condition (SPDn=1).	
<ul style="list-style-type: none"> • When a stop condition is detected 	

Note: 1. LRELn: Bit6 of IICA control register n0 (IICCTLn0).

IICEn: Bit7 of IICA control register n0 (IICCTLn0).

2.n=0,1

20.3.4 IICA flag register n (IICFn).

This is the register that sets the I²C operating mode and indicates the status of the I2 C-bus.

The IICFn register is set by the 8-bit memory operation instruction. However, only the STTn clear flag (STCFn) and the I2 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 20-7 Format of the IICA flag register n (IICFn).

After reset: 00HR/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=0,1).	
Position condition (STCFn=1).	
<ul style="list-style-type: none"> • Clearance due to STTn bit being "1" • When the IICEn bit is "0" (stop running). • When resetting 	<ul style="list-style-type: none"> • 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

IICBSYn	I ² C-bus status flag
0	Bus release state (initial communication state at STCENn=1).
1	Bus communication state (initial communication state at STCENn=0,1).
Clear condition (IICBSYn=0,1).	
Position condition (IICBSYn=1).	
<ul style="list-style-type: none"> • When a stop condition is detected • When the IICEn bit is "0" (stop running). • When resetting 	<ul style="list-style-type: none"> • When a start condition is detected • The position of the IICEn bit when the STCENn bit is "0"

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=0,1).	
Position condition (STCENn=1).	
<ul style="list-style-type: none"> • Clear by command. • When a start condition is detected • When resetting 	<ul style="list-style-type: none"> • Set by command.

IICRSVn	The communication appointment function prohibits bits
0	Allow correspondence appointments.
1	Communication appointments are prohibited.
Clear condition (IICRSVn=0,1).	
Set condition (IICRSVn=1).	
<ul style="list-style-type: none"> • Clear by command. • When resetting 	<ul style="list-style-type: none"> • Set by command.

Note: Bit6 and bit7 are read-only bits.

Note: 1. The STCENn bit can only be written when it is stopped (IICEn=0,1).

If the STCENn bit is "1", the bus is considered to be free (IICBSYn=0,1) 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=0,1).

Note: 1. STTn: Bit1 of IICA control register n0 (IICCTLn0).

2. IICEn: bit7 of IICA control register n0 (IICCTLn0).

20.3.5 IICA control register n1 (IICCTLn1).

This is a register used to set the I²C operating mode and to detect the status of the SCLAn pin and SDAAn pins.

The IICCTLn1 register is set by the 8-bit memory operation instruction. However, only CLDn bits and DADn bits can be read.

In addition to the WUPn bit, bit7 must be disabled to run in I²C (IICA control register n0 (IICCTLn0). (IICEn)=0) when setting the IICCTLn1 register.

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

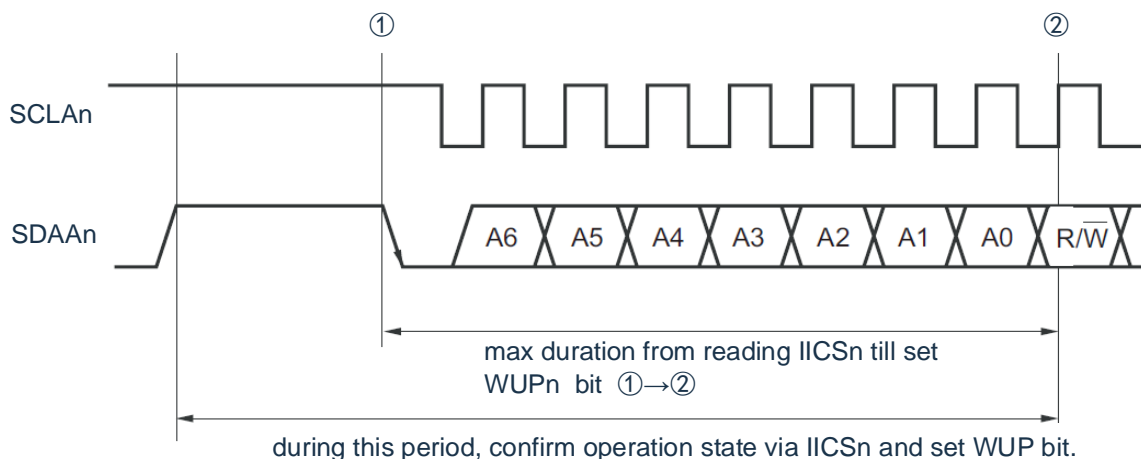
Figure 20-8 Format of IICA control register n1 (IICCTLn1) (1/2).

After reset: 00H R/W Note 1

symbol	<div>7</div>	6	<div>5</div>	<div>4</div>	<div>3</div>	<div>2</div>	1	<div>0</div>
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.						
<p>To transfer to deep sleep mode by placing the WUPn position "1", at least 3 f must pass after the WUPn position "1" MCK clock, followed by deep sleep instructions (refer to "Figure 14-28 will wupn position" 1 "When the process"). After the address matches or the extension code is received, the WUPn bit must be cleared to "0". Can participate in subsequent communication by clearing the WUPn bit "0" (you need to cancel the wait and write the send data after the WUPn bit is cleared "0").</p> <p>In the state where the WUPn bit is "1", the interrupt timing when the address matches or the extension code is received is the same as the interrupt timing when the WUPn bit is "0"</p> <p>(The delay difference that produces the sampling error based on the clock). In addition, when the WUPn bit is "1", even if the SPIEn position "1" is placed, it does not produce a stop condition interrupt.</p>								
Clear condition (WUPn=0,1).					Position condition (WUPn=1).			
• Cleared by directive (after the address matches or the extension code is received).					• Set by instruction (MSTSn=0,1, EXCn=0,1, COIn=0,1 and STDn=0,1 (do not participate in communications)) Note 2.			

Note: 1.bit4 and bit5 are read-only bits.

2. During the period shown below, it is necessary to confirm the status of the IICA status register n (IICSn) and set it in place.



Note: n=0,1

Figure 20-9 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=0,1).		Position condition (CLDn=1).
<ul style="list-style-type: none"> • When the SCLAn pin is low • When the IICEn bit is "0" (stop running). • When resetting 		<ul style="list-style-type: none"> • When the SCLAn pin is high

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=0,1).		Set condition (DADn=1).
<ul style="list-style-type: none"> • When the SDAAn pin is low • When the IICEn bit is "0" (stop running). • When resetting 		<ul style="list-style-type: none"> • When the SDAAn pin is high

SMCn	Switching of operating modes
0	Operates in standard mode (maximum 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 position is "1" or the clear "0", the transmission clock is unchanged.	

PRSn	Control of the running clock (f MCK).
0	Select f_{CLK} ($1MHz \leq f_{CLK} \leq 20MHz$).
1	Select $f_{CLK}/2$ ($20MHz < f_{CLK}$).

Note: 1. The maximum operating frequency of the IICA Operating Clock (f_{MCK}) is 20MHz (Max.). . The IICA control register n1 (IICCTLn1) must only be placed when the f_{CLK} exceeds 20MHz bit0 (PRSn) is set to "1".

2. In the case of setting the transmission clock, the minimum operating frequency of the f_{CLK} must be paid attention to. The minimum operating frequency of the f_{CLK} of the serial interface IICA depends on the operating mode.

Fast mode: $f_{CLK} = 3.5MHz$ (Min.).

Enhanced Fast Mode: $f_{CLK}=10MHz$ (Min.).

Standard mode: $f_{CLK}=1MHz$ (Min.).

Note: 1. IICEn: Bit7 of IICA control register n0 (IICCTLn0).

2.n=0,1

20.3.6 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 the 8-bit memory operation instruction.

Must be disabled in bit7 (IICEn) where I^2C is disabled to run (IICA control register n0 (IICCTLn0).)=0) when setting the IICWLn register. After the reset signal is generated, the value of this register changes to "FFH".

For how to set the IICWLn register, refer to "20.4.2The method of transmitting the clock is set by the IICWLn register and the IICWHn register".

The data retention time is 1/4 of the time set by IICWLn.

Figure 20-9 The format of the IICA low-level width setting register n (IICWLn).

After reset: FFH					R/W			
Symbol	7	6	5	4	3	2	1	0
IICWLn								

20.3.7 IICA high level width setting register n (IICWHn).

This register controls the SCLAn pin signal high level width and SDAAn pin signal of the serial interface IICA output. The IICWHn register is set by the 8-bit memory operation instruction.

Must be disabled in bit7 (IICEn) where I^2C is disabled to run (IICA control register n0 (IICCTLn0).)=0) when setting the IICWHn register. After the reset signal is generated, the value of this register changes to "FFH".

Figure 20-10 IICA high level width sets the format of register n (IICWHn).

After reset: FFH					R/W			
Symbol	7	6	5	4	3	2	1	0
IICWHn								

Remarks: 1 For the method of setting the clock transmitted by the main controller, please refer to 20.4.2(1); For how to set the slave IICWLn register and the IICWHn register, refer to 20.4.2(2).

2.n=0,1

20.3.8 Port mode register x (PMx).

This register sets the input/output of the port.

When using the Pxx/SCLAn pin as the clock input/output and the Pxx/SDAAn pin as the serial data input/output, The port mode register PMx and the port output latch Px must be placed at "0".

When the IICEn bit (bit7 of the IICA control register n0 (IICCTLn0)) is "0", P The xx/SCLAn pin and the Pxx/SDAAn pin are low-level outputs (fixed), so the IICEn must be placed "1" After switching to output mode.

Set the PM x register via the 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "FFH".

For details of the port mode registers, see "2.3.1 Port Mode Registers (PMxx)".

20.4 The functionality of I²C-bus mode

20.4.1 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

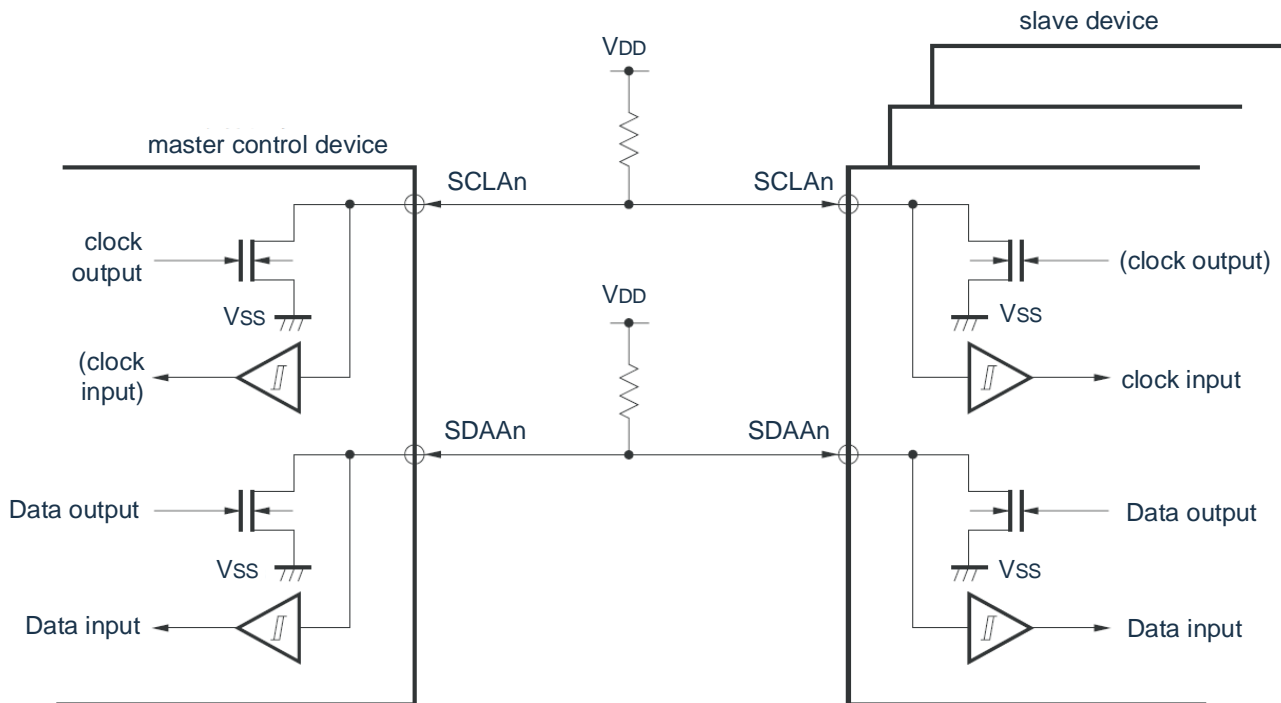
The outputs of both the master and slave devices are N-channel open-drain outputs, and the inputs are Schmidt inputs.

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 20-11 Pin Structure Diagram



Note: n=0,1

20.4.2 The method of transmitting the clock is set by the IICWLn register and the IICWHn register

- (1) The method by which the master transmits the clock

$$\text{Transmission clock} = \frac{f_{MCK}}{IICWL + IICWH + f_{MCK}(t_R + t_F)}$$

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

$$IICWLn = 0, 1.50 \mu s \times f_{MCK}$$

$$IICWHn = (0.50 \mu s - t_R - t_F) \times f_{MCK}$$

Note: 1. The maximum operating frequency of the IICA Operating Clock (f_{MCK}) is 20MHz (Max.). . The IICA control register n1 (IICCTLn1) must only be placed when the f_{CLK} exceeds 20MHz bit0 (PRSn) is set to "1".

2. In the case of setting the transmission clock, the minimum operating frequency of the f_{CLK} must be paid attention to. The minimum operating frequency of the f_{CLK} of the serial interface IICA depends on the operating mode.

Fast mode: $f_{CLK} = 3.5\text{MHz (Min.)}$.

Enhanced Fast Mode: $f_{CLK} = 10\text{MHz (Min.)}$.

Standard mode: $f_{CLK} = 1\text{MHz (Min.)}$.

Remarks: 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

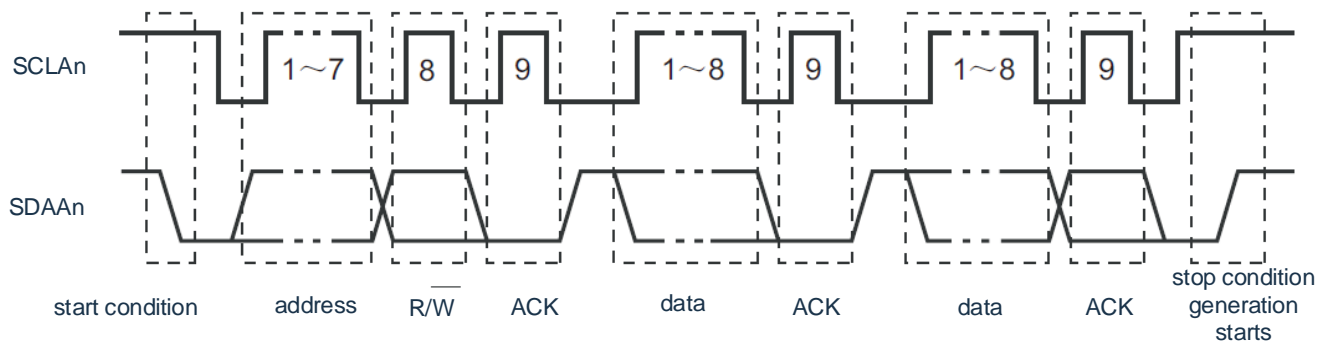
3. $n=0.1$

20.5 Definition and control method of the I²C-bus

The following describes the serial data communication format and the signals used for the I²C-bus.

The Start Condition, Address, Data generated on the serial data bus of the I²C-bus. The respective transmission timings for and "Stop Condition" are shown in the following figure.

Figure 20-12 the I²C-bus



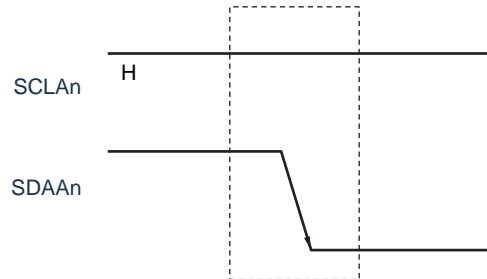
The master generates start conditions, slave addresses, and stop conditions.

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.

20.5.1 Start conditions

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 20-13 starting conditions



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".

Note: n=0,1

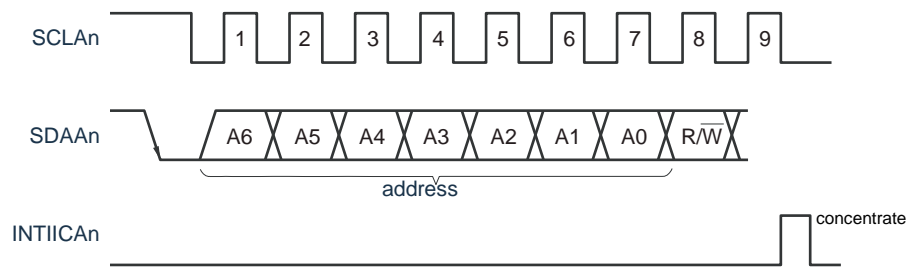
20.5.2 address

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 20-14 address



Note: If data other than the local station address or extension code is received while the slave is running, INTIICAn is not generated.

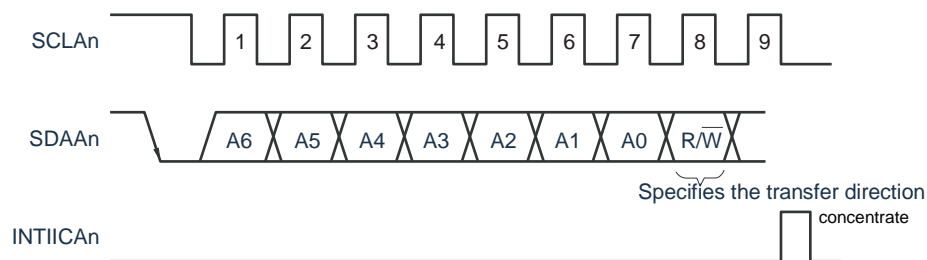
If the 8-bit data consisting of the designation of the transmission direction of the Transmission Direction in 20.5.3" is written to the IICA shift register n (IICAn), the output address. The received address is written to the IICAn register. The slave address is assigned 7 bits high in the IICAn register.

20.5.3 The designation of the transmission direction

The master device sends 1 bit of data in the specified transmission direction after the 7-bit address.

When this transfer direction is specified as a bit of "0", it means that the master device sends data to the slave device; When this transfer direction is specified as bit "1", it means that the master device receives data from the slave.

Figure 20-15 Designation of the transmission direction



Note: If data other than the local station address or extension code is received while the slave is running, INTIICAn is not generated. Note n = 0,1

20.5.4 Ack (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 abort 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.

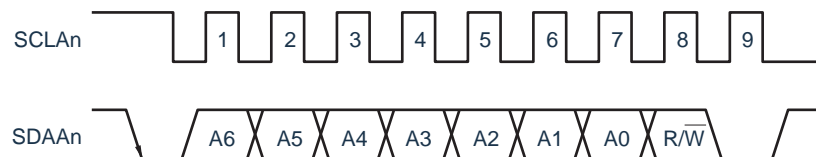
The receiver sets the SDAAn line low on the 9th clock to generate a reply (normal receive).

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=0,1), it is usually necessary to place the ACKEn position "1".

During the slave receive run (TRCn=0,1) 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=0,1), 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 20-16 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 the extension code is received, a reply is generated by placing the ACKEn position "1" in advance. The Ack generation method when receiving data varies depending on the waiting timing setting, as shown below.

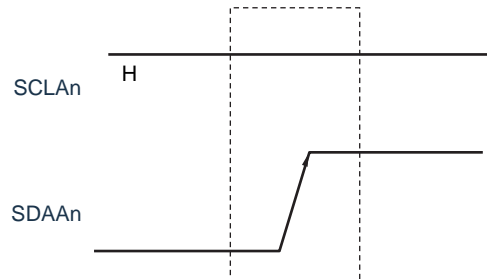
- When selecting a wait for 8 clocks (bit3 (WTIMn) =0 of the IICCTLn0 register): pass before releasing the wait ACKEn position "1" generates a reply synchronously with the 8th clock drop edge of the SCLAn pin.
- When selecting a wait for 9 clocks (bit3 (WTIMn) =1 of the IICCTLn0 register): pass beforehand ACKEn position "1" to generate a reply.

Note: n=0,1

20.5.5 Stop Conditions

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 20-17 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 (IICCSn) to "1" and generates INTIICAn when bit4 (SPIEn) of the IICCTLn0 register is "1".

Note: n=0,1

20.5.6 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 lifted, the next transfer can begin.

Figure 20-18 Wait (1/2).

(1) The master device waits for 9 clocks and the slave device waits for 8 clocks

(Master: Transmit, Slave: Receive, ACKEn=1).

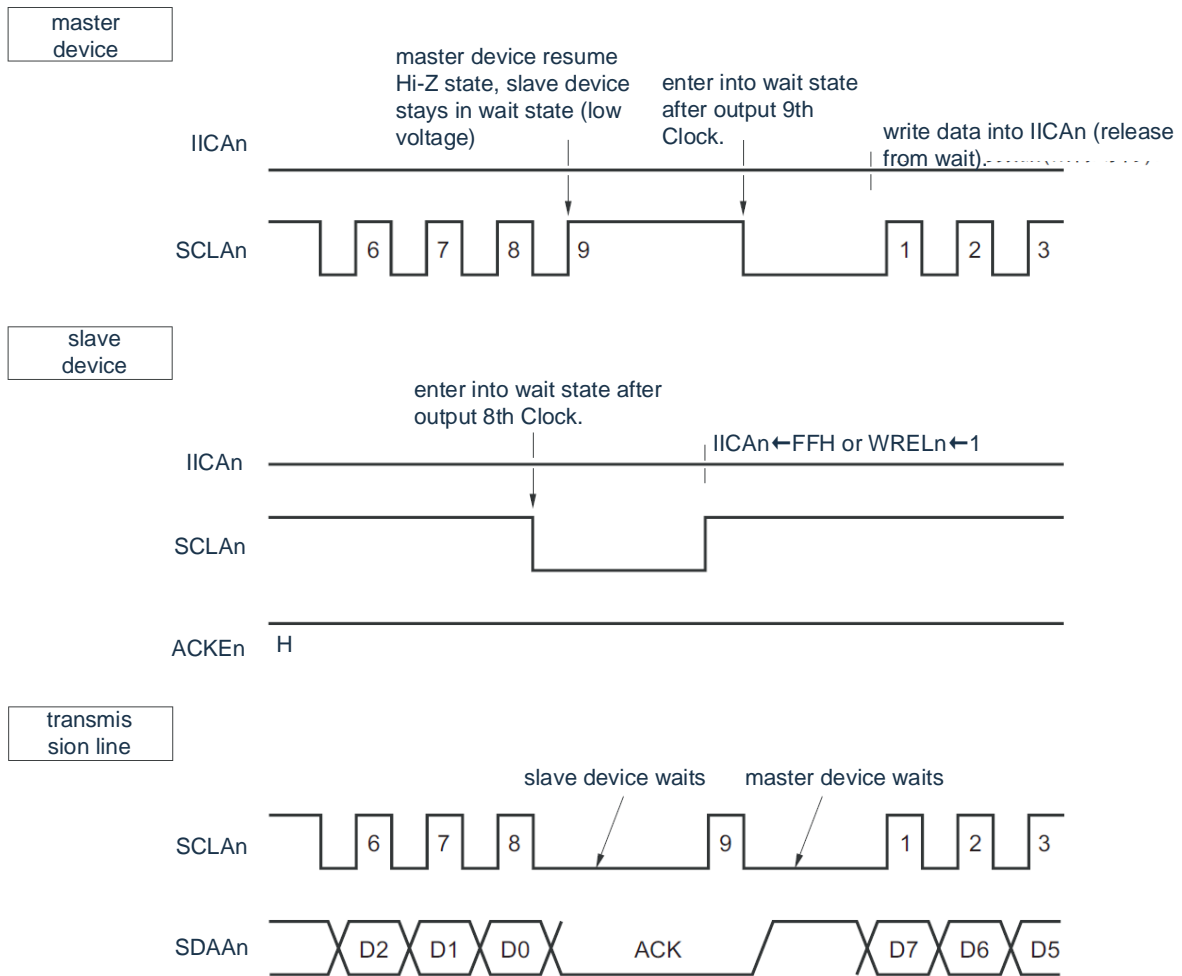
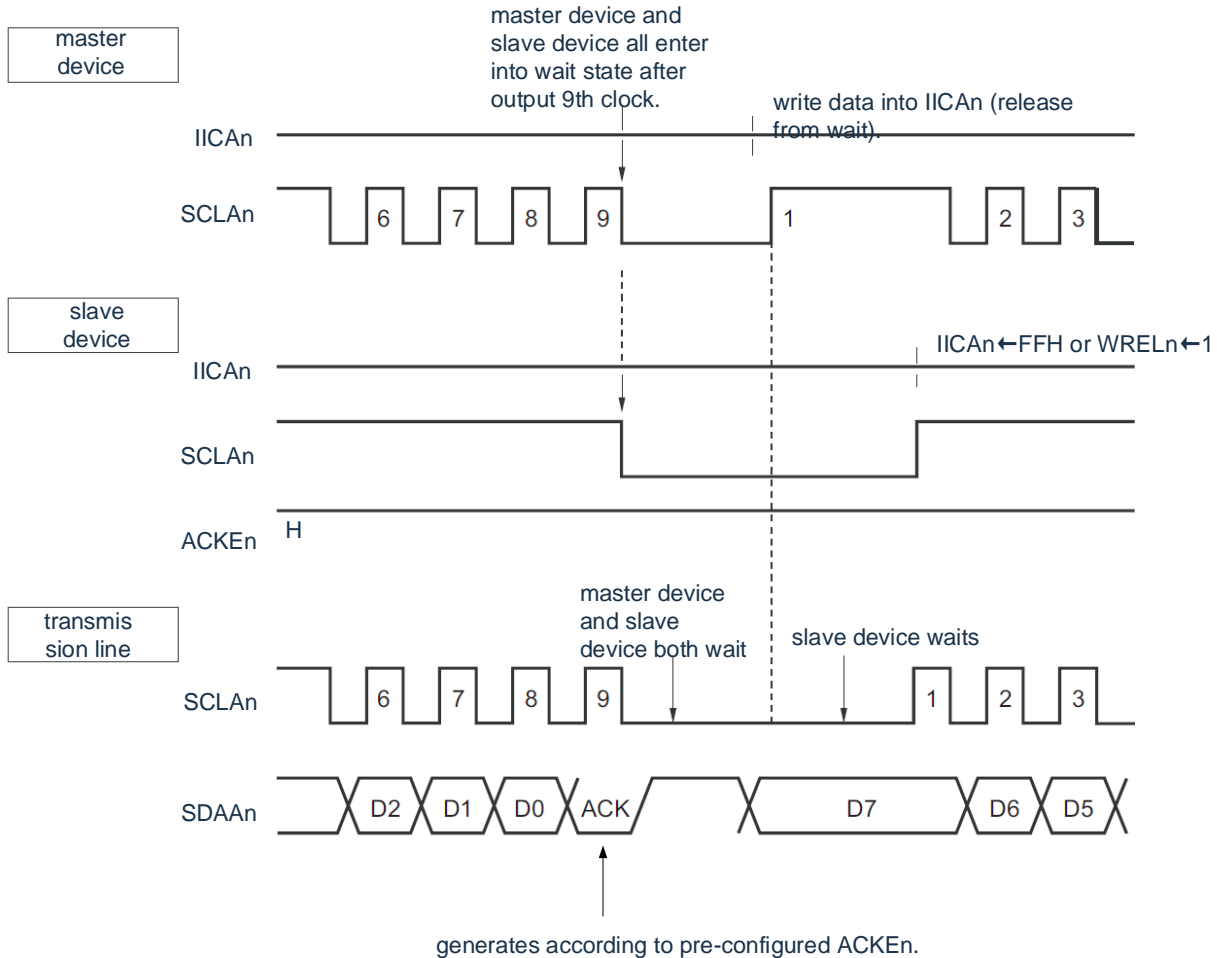


Figure 20-19 Wait (2/2).

(2) A situation where both the master and slave devices are waiting for 9 clocks

(Master: Transmit, Slave: Receive, ACKEn=1).



Note: ACKEn: Bit2 of IICA control register n0 (IICCTLn0).

WRELn: Bit5 of the IICA control register n0 (IICCTLn0).

A wait state is automatically generated by bit3 (WTIMn) by setting the IICA control register n0 (IICCTLn0). Typically, on the receiver side, if the bit5 (WRELn) of the IICCTLn0 register is "1" or give the IICA a shift register n (IICAn) write "FFH", the wait is lifted; On the sender, if data is written to the IICAn register, the wait is lifted. The master can also lift the wait by:

- Set bit1 (STTn) of the IICCTLn0 register to "1".
- Set bit0 (SPTn) of the IICCTLn0 register to "1".

Note: n=0,1

20.5.7 method of release from wait state

In general, I²C can be relieved of waiting by 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) ^{note}.
- Set the bit0 (SPTn) of the IICCTLn0 register (generate stop condition) ^{note}.

Note: Limited to master devices only.

If these waiting releases are performed, I²C dismisses the wait and resumes communication. To send data (including addresses) after de-waiting, data must be written to the IICAn registers.

To receive data after the wait is lifted or when sending data ends, the bit5 (WRELn) of the IICCTLn0 register must be placed at "1". To generate a restart condition after a wait is lifted, the bit1 (STTn) of the IICCTLn0 register must be placed at "1". To generate a stop condition after lifting the wait, the bit0 (SPTn) of the IICCTLn0 register must be set to "1". Only one release process can be performed for a wait.

For example, if you write data to the IICAn register after de-waiting by placing the WRELn position "1", the changing timing of the SDAAn line may conflict with the write timing of the IICAn register, resulting in the wrong value being output to SDAAn line. In addition to these processes, in the case of aborting communication, if the IICEn bit is cleared "0", the communication is stopped, so the wait can be lifted. In the case where the I²C-bus state is deadlocked due to noise, if the bit6 (LRELn) of the IICCTLn0 register is set "1" exits the communication and therefore can lift the wait.

Note: If you perform a pending release process when the WUPn bit is "1", the wait is not dismissed.

Note: n=0,1

20.5.8 Generation timing and waiting control of interrupt requests (INTIICAn).

By setting the IICA control register n0 (IICCTLn0) bit3 (WTIMn), in Table 20-2 The timing shown generates INTIICAn and is subject to wait control.

Table 20-2: Generation timing and waiting control of INTIICAns

WTIMn	Slave run			The master runs		
	address	Data reception	Data is sent	address	Data reception	Data is sent
0	9Notes 1 and 2	8note 2	8note 2	9	8	8
1	9Notes 1 and 2	9note 2	9note 2	9	9	9

Note: 1. Only when the received address and the set address of the slave address register n(SVAn) are the same, the slave generates an INDICATIONn signal on the falling edge of the 9th clock and enters a waiting state.

At this point, regardless of the bit2 (ACKEn) setting of the IICCTLn0 register, a reply is generated. The slave that receives the extension code generates INTIICAn on the descending edge of the 8th clock. If the addresses are different after restarting, INTIICAn is generated on the falling edge of the 9th clock, but does not enter the waiting state.

2. If the contents of the received address and the slave address register n(SVAn) are different and the extension code is not received, THE INTIICAn is not generated and does not enter the waiting state.

Note: 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) The sending and receiving of 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 Run: Determines the timing of interrupts and waits via the WTIMn bit.

(3) Data is sent

- Master/Slave Run: Determines the timing of interrupts and waits via the WTIMn bit.

Note: n=0,1

(4) Wait for the method of release

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) ^{note}.
- Set the bit0 (SPTn) of the IICCTLn0 register (generate stop condition) ^{note}.

Note: Limited to master devices only.

When you select a wait for 8 clocks (WTIMn=0,1), you need to decide whether to generate a reply before you lift the wait.

(5) Detection of stop conditions

If a stop condition is detected, INTIICAn is generated (limited to the case of SPIEn=1).

20.5.9 The detection method for address matching

In I2C-bus mode, the master device can select a specific slave by sending a slave address. Address matching can be automatically detected by hardware. When the slave address sent by the master device and the set address of the slave address register n(SVAn) are the same or only the extension code is received, an INDICATIONAn interrupt request is generated.

20.5.10 Detection of errors

In I2C-bus mode, because the status of the serial data bus (SDAAn) during the transmission process is taken to the IICA shift register n (IICAn) of the transmitting device, Therefore, it is possible to detect send errors by comparing the IICA data before and after the start of sending. At this point, if the two data are different, it is judged that a sending error has occurred.

Note: n=0,1

20.5.11 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

Note: EXCn: bit5 of the IICA status register n (IICSn).

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 20-3 Bit definitions of the main extension codes

The slave address	R/Wbit	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).

Remarks: 1 For extension codes other than those listed above, please refer to the I2C-bus datasheet issued by NXP.

2.n=0,1

20.5.12 arbitration

When multiple master devices generate start conditions at the same time (in the case of STTn position "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

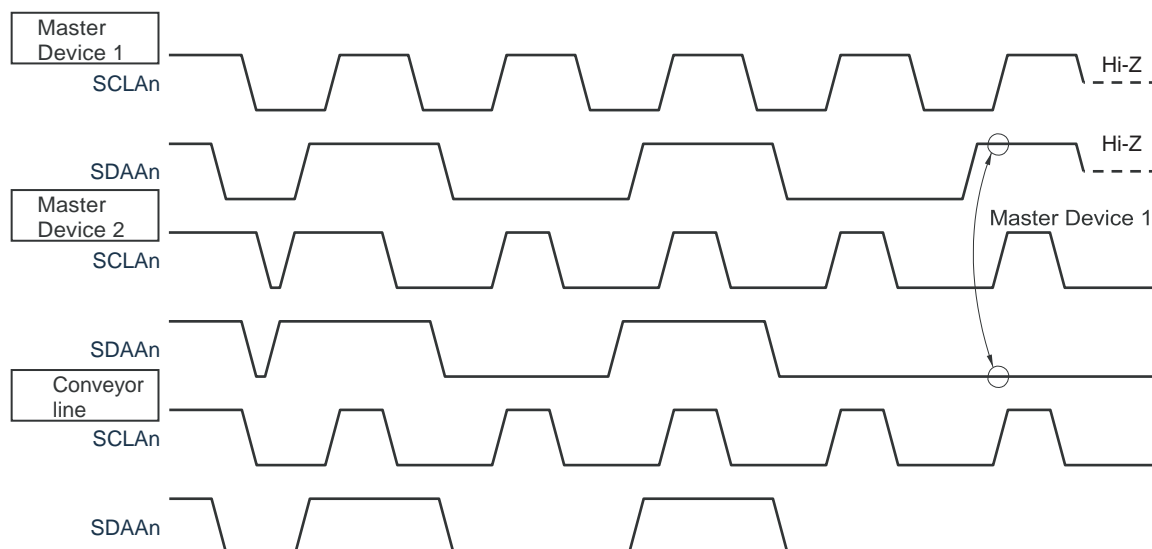
to detect the failure of the quorum.

For the timing of interrupt requests, please refer to "Generation Timing and Waiting Control for Interrupt Requests (INTIICAn) in 14.5.8".

Note: STDn: Bit1 of the IICA status register n (IICSn).

STTn: Bit1 of IICA control register n0 (IICCTLn0).

Figure 20-19 Arbitration timing example



Note: n=0,1

Table 20-4 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 ^{note 1}
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.	
A stop condition was detected during data transfer.	Note 2 when generating a stop condition (SPIEn=1).
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 ^{note 1}
You want to build a restart condition, but a stop condition is detected.	Note 2 when generating a stop condition (SPIEn=1).
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 ^{note 1}
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.

2. When there is a possibility of arbitration, the SPIEn position must be "1" when the master is running.

Note: 1. SPIEn: bit4 of the IICA control register n0 (IICCTLn0).

2.n=0,1

20.5.13 Wake-up function

This is a subordinate function of I²C, which is the function of generating an interrupt request signal (INTIICAn) when the local station address and extension code are received. The processing efficiency is improved by not generating unwanted INTIICAn signals under different addresses. If a start condition is detected, it enters wake-up standby. Because the master device (where a start condition has already been generated) may also become a slave due to 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 usual run.

The flow when the WUPn position "1" is shown in Figure 20-20. WUPn position "0" is matched by address matching is shown in Figure 20-21.

Figure 20-20 the process when the WUPn position "1" is placed

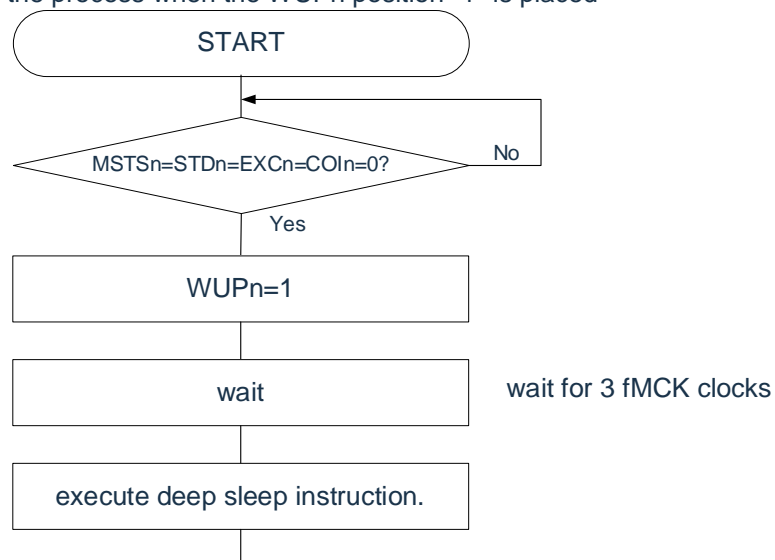
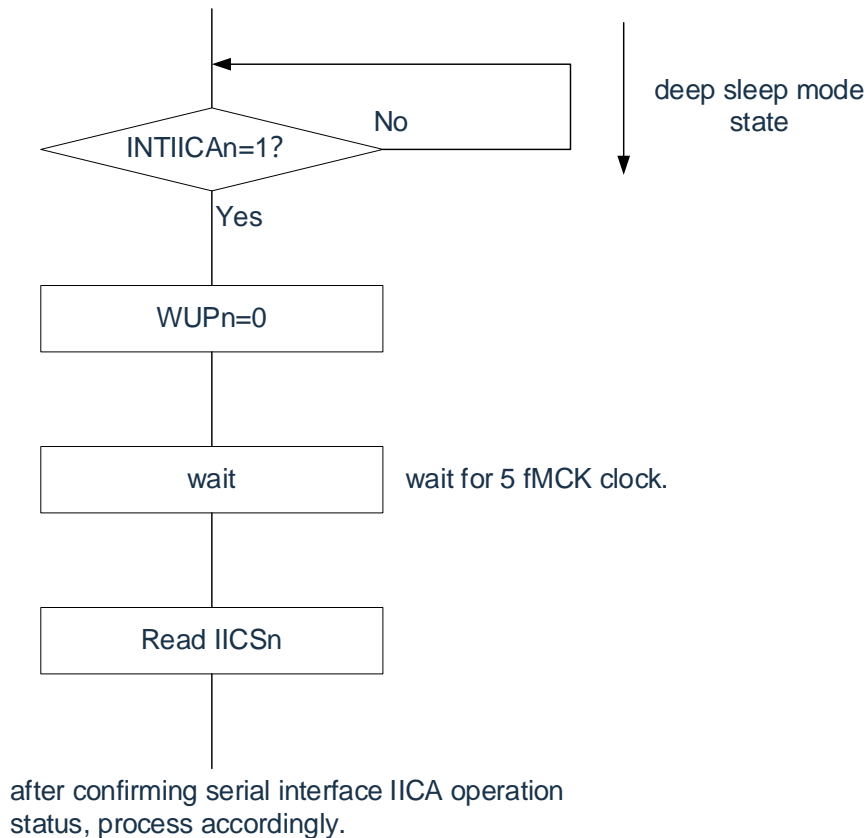


Figure 20-21 flow (including receiving the extension code) when the WUPn position "0" is matched by address matching



In addition to the interrupt request (INTIICAn) generated by the serial interface IICA, the deep sleep mode must be removed through the following procedure.

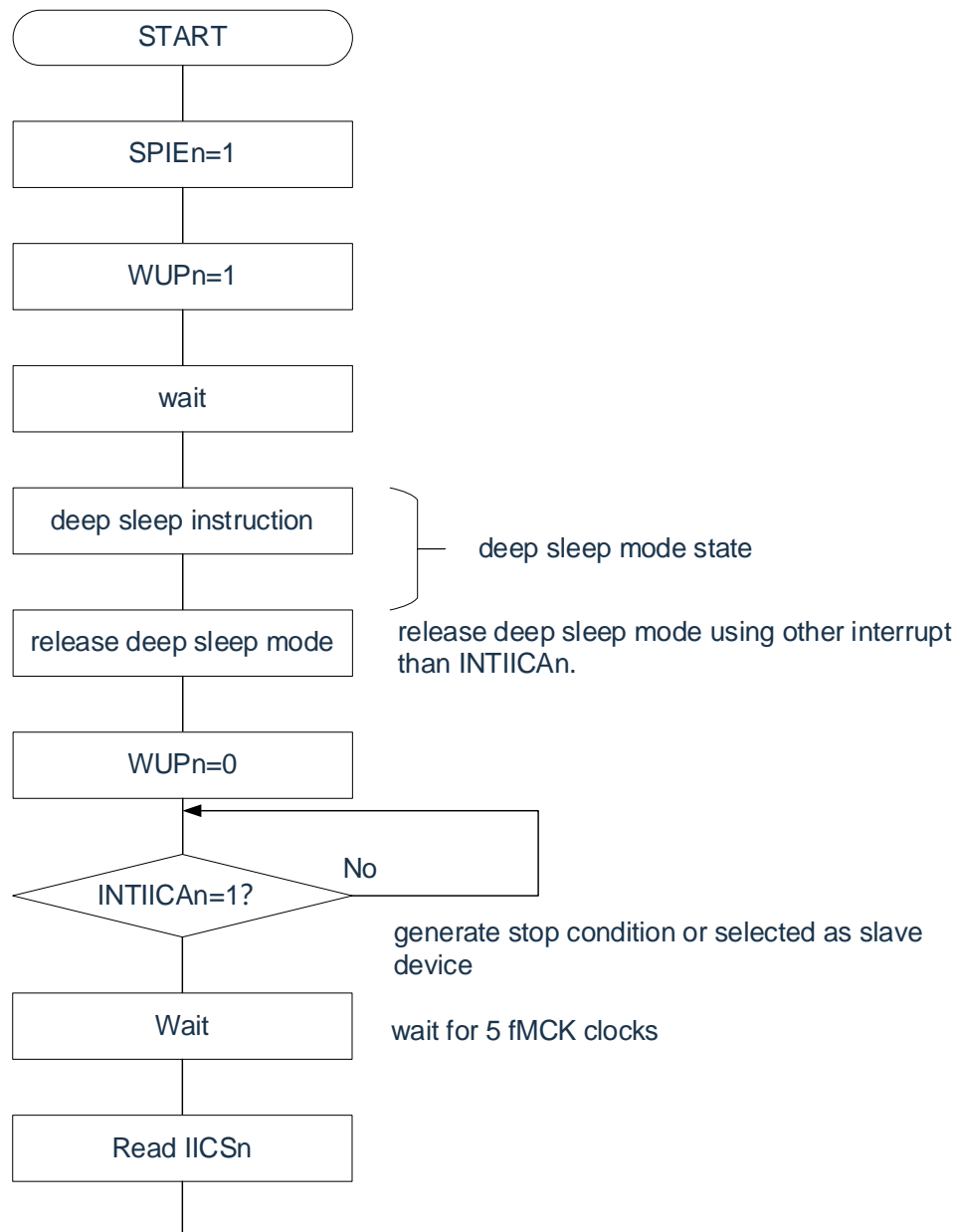
- The next IIC communication is the case of the operation of the master control device: Figure 20-22-22
- The next IIC communication is the case for the slave to run:

The situation of returning via ANTIICAn interrupt: The same process as Figure 20-21.

Cases returned by interrupts other than THETIICAn interrupts: The STATE in which the WUPn bit is "1" must be maintained to continue running before the INTIICAn interrupt is generated.

Note: n=0,1

Figure 20-22: deep sleep mode is lifted by interrupts other than INTIICAn as the master device



after confirming serial interface IICA operation status, process accordingly.

Note: n=0,1

20.5.14 Communication appointments

(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
- Bit6 that does not run as a slave after receiving the extension code (does not return a reply but puts the IICA control register n0 (IICCTLn0). (LRELn) is placed "1", 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 positioned "1", it is decided whether to run as a start condition or as a communication appointment depending on the bus state.

- When the bus is in a release state..... Build start conditions
- When the bus is not in the release state (standby state)... Correspondence appointments

After placing stTn at position "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 placing stTn position "1" until the MSTSn flag is confirmed:

$$(IICWLn \text{ setpoint} + IICWHn \text{ setpoint} + 4) / f_{MCK} + t_F \times 2$$

Remarks: 1. IICWLn: IICA low-level width setting register n

IICWHn: IICA high level width setting register n

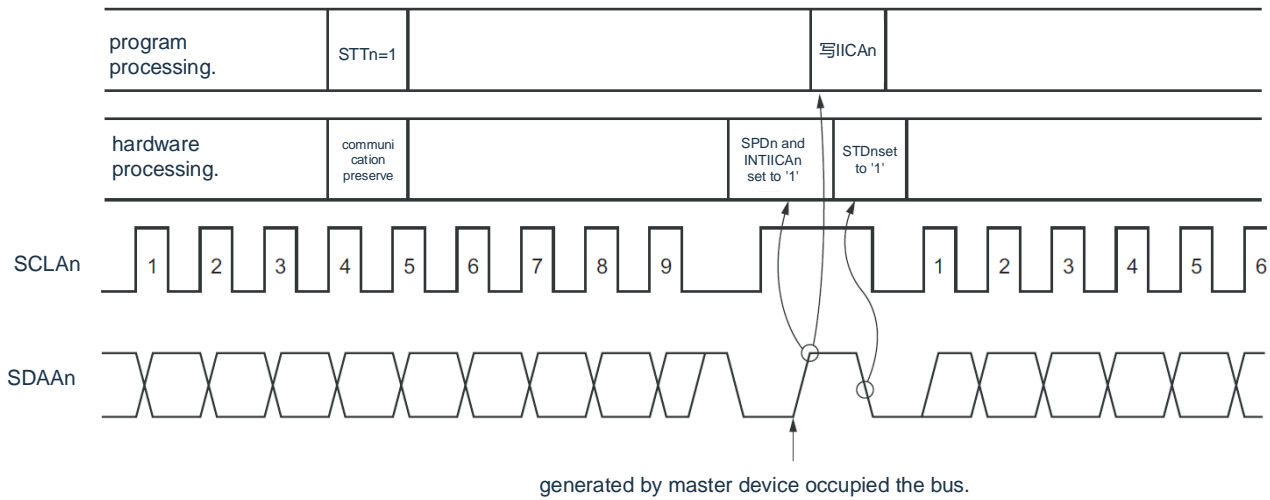
t_F: The descent time of the SDAAn signal and the SCLAn signal

f_{MCK}: IICA operating clock frequency

2. n=0.1

The timing of the communication appointment is shown in the following figure.

Figure 20-23 Timing of communication appointments



Note: IICAn : IICA shift register n

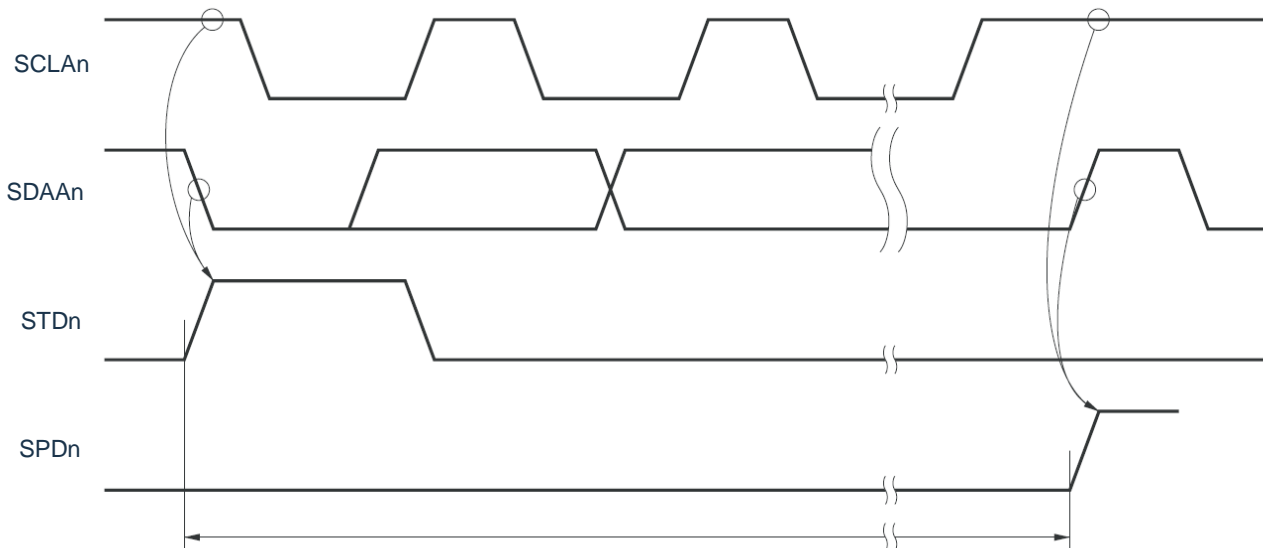
STTn : Bit1 of IICA control register n0 (IICCTLn0).

STDn : Bit1 of the IICA status register n (IICSn).

SPDn : Bit0 of IICA status register n (IICSn).

Accept communicationFigure 20-24. After bit1 (STDn) of the IICA status register n (IICSn) becomes "1" and before a stop condition is detected, it will be Bit1 (STTn) of the IICA control register n0 (IICCTLn0) is placed "1" Make a communication appointment.

Figure 20-24 Reception timing of communication appointments

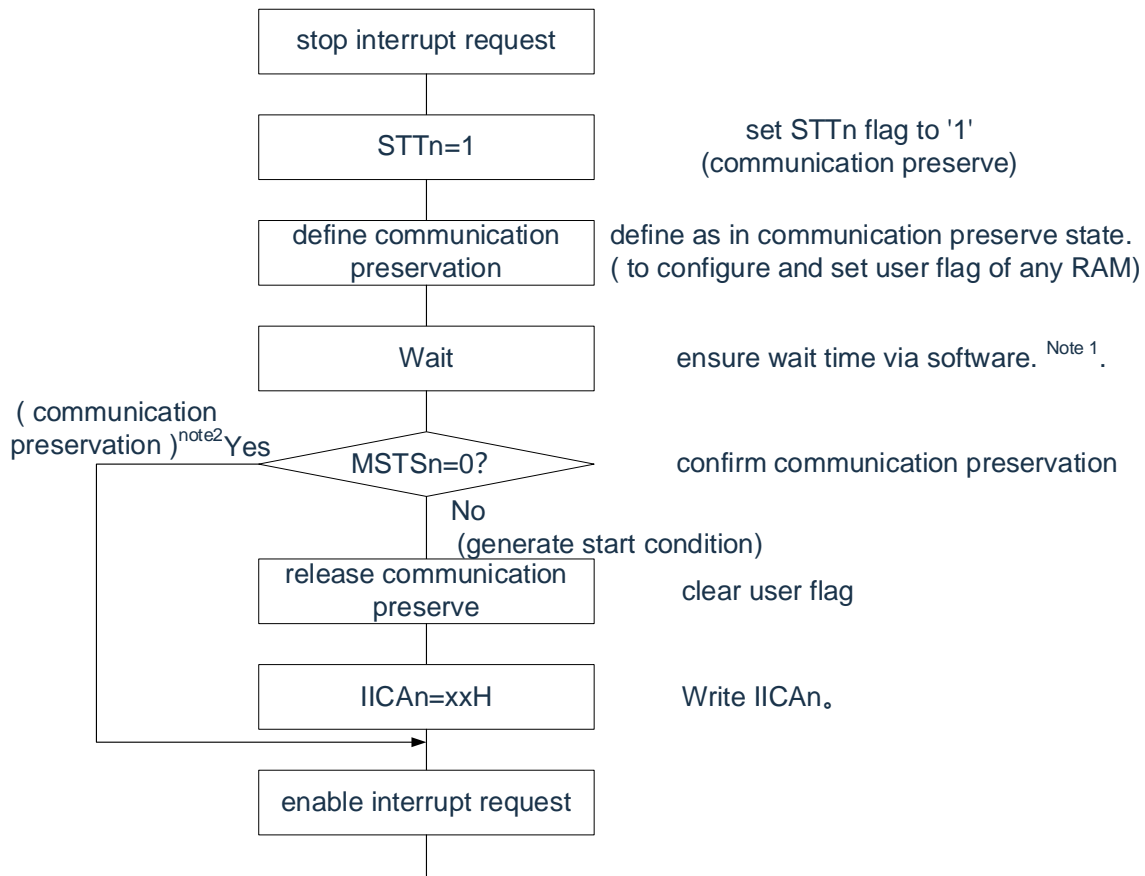


standby (during this, can preserve communication via setting STTn bit to '1')

The steps of the communication appointment are shown in Figure 20-25.

Note: n=0,1

Figure 20-25: Communication appointment steps



Note: 1. The waiting time is as follows: $(IICWL_n \text{ setpoint} + IICWH_n \text{ setpoint} + 4)/f_{MCK} + t_F \times 2$

2. Write the IICA shift register n (IICAn) by stopping the conditional interrupt request while the communication appointment is running.

Remarks: 1. STT_n : Bit1 of IICA control register n0 (IICCTLn0).

MSTS_n : Bit7 of the IICA status register n (IICS_n).

IICAn : IICA shift register n

IICWL_n : IICA low-level width setting register n

IICWH_n: IICA high level width setting register n

t_F : The descent time of the SDAAn signal and the SCLAn signal

f_{MC}: IICA operating clock frequency

2. n=0.1

(2) Case where the communication appointment function is prohibited (bit0 (IICRSV_n) of the IICA flag register n (IICFn) = 1)

During bus communication, if you are not participating in this communication, the bit1 (STT_n) of the IICA control register n0 (IICCTLn0) is placed

"1" rejects the request and does not generate a start condition. 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
- Does not run as a slave after receiving the extension code (bit6 (LREL_n) of the IICCTLn0 register is set to "1" without returning a reply, and the bus is released after exiting communication).

STCFn (bit7 of the IICFn register) can be used to confirm whether a start condition was generated or the request was rejected. Because it takes 5 fMCKs from the STTn bit "1" to the STCFn position "1" The time of the clock, so this time must be ensured by the software.

Note: n=0,1

20.5.15 Other considerations

- (1) The case where the STCENn bit is "0"

Just after I²C 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 I²C is allowed to run (IICEn=1), it is considered a release state (IICBSYn=0,1) 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) I²C communication with other devices is ongoing

When the SDAAn pin is low and the SCLAn pin is high, I²C macros are considered SDAAn citations if I²C 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 I²C communication with other devices. To avoid this, I²C 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 I²C to run.
- (3) Wait for the start condition to be detected.
- (4) IICCTLn0 is placed before returning to the reply (within 4 to 72 f_{MCK} clocks after the IICEn position "1"). The bit6 (LRELn) of the register is placed "1", forcing the detection to be invalid.

- (4) After setting the STTn bits and SPTn bits (bit1 and bit0 of the IICCTLn0 register), it is forbidden to clear "0" Reposition before.

- (5) If a communication appointment is made, the SPIEN bit (bit4 of the IICCTLn0 register) must be placed at "1" to generate an interrupt request when a stop condition is detected. After an interrupt request is generated, the transmission begins by writing communication data to the IICA shift register n (IICAn). If no interruption occurs when a stop condition is detected, it stops in the waiting state because no interrupt request is generated when communication begins. However, when the MSTSn bit (bit7 of the IICA status register n (IICSn)) is detected by software, There is no need to place the SPIEn position "1".

Note: n=0,1

20.5.16 Communication operation

Here, the following three running steps are represented by a flowchart.

(1) The 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 I2C bus, it is not possible to judge whether the bus is in the release state or in use during the stage of participating in the communication based on the specifications of the I2C bus. Here, if the data and clock are high for a certain amount of time (1 frame), the bus participates in communication as a release state. This process is roughly divided into "initial 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 run

An example of an I2C bus slave is shown below.

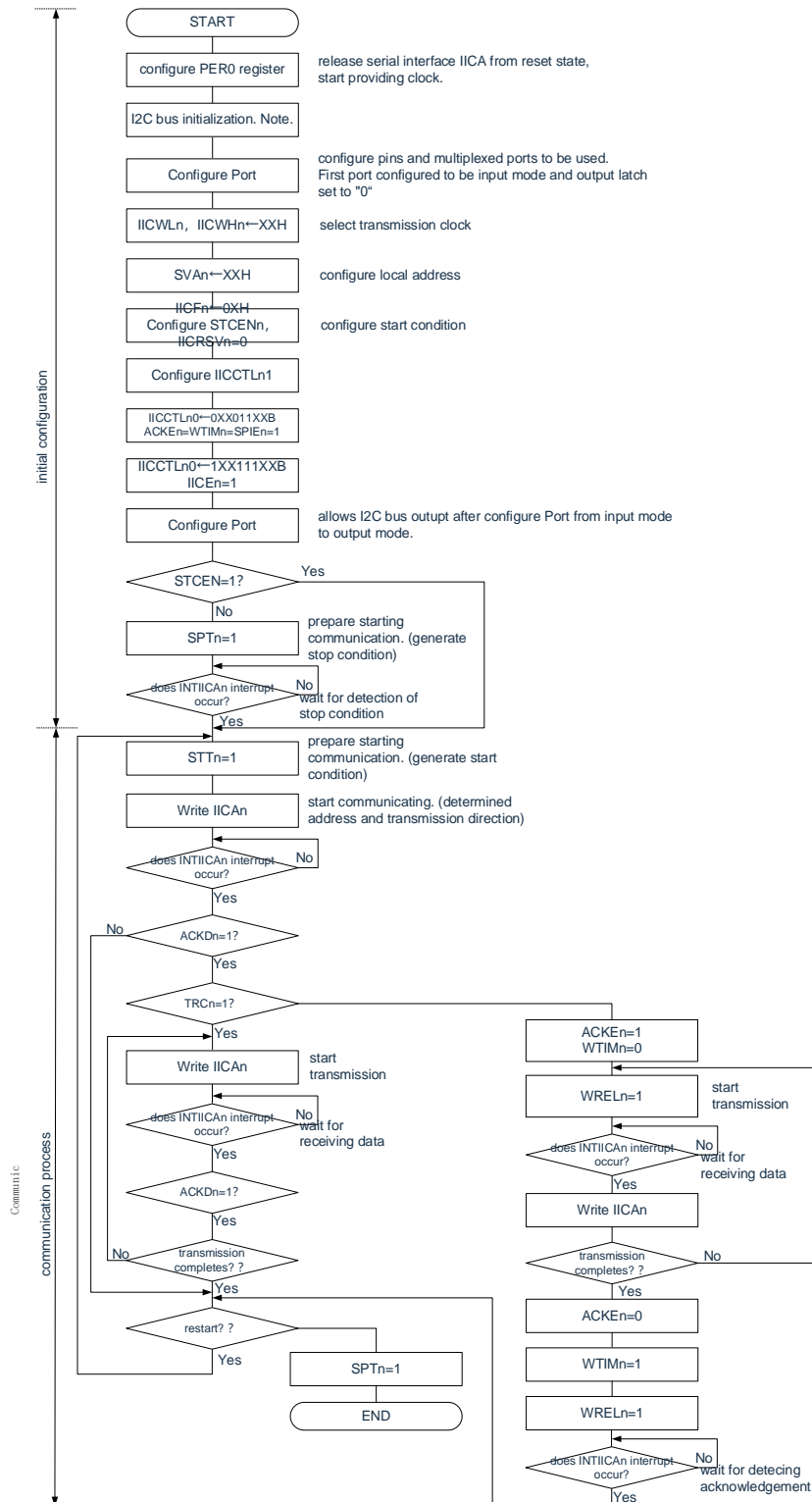
When used as a slave, it starts running with an interrupt. Perform the "Initial Setup" section at startup, then through "Communication Wait" and wait for the INTIICAn interrupt to occur. If an INTIICAn interrupt occurs, the communication status is determined and the flag is passed to the main processing department.

By checking each flag, the required "communication processing" is carried out.

Note n = 0,1

(1) The master operation of a single-master system

Figure 20-26 The master operation of the single master control system



Note: The I2C-bus must be released (SCLAn pins and SDAAn pins are high) depending on the specifications of the product in communication. For example, if the EEPROM is in a state that outputs a low level to the SDAAn pin, the SCLAn pin must be set to the output port and a clock pulse must be output from the output port before the SDAAn pin is fixed high.

Remarks: 1 The formats of the sending and receiving must

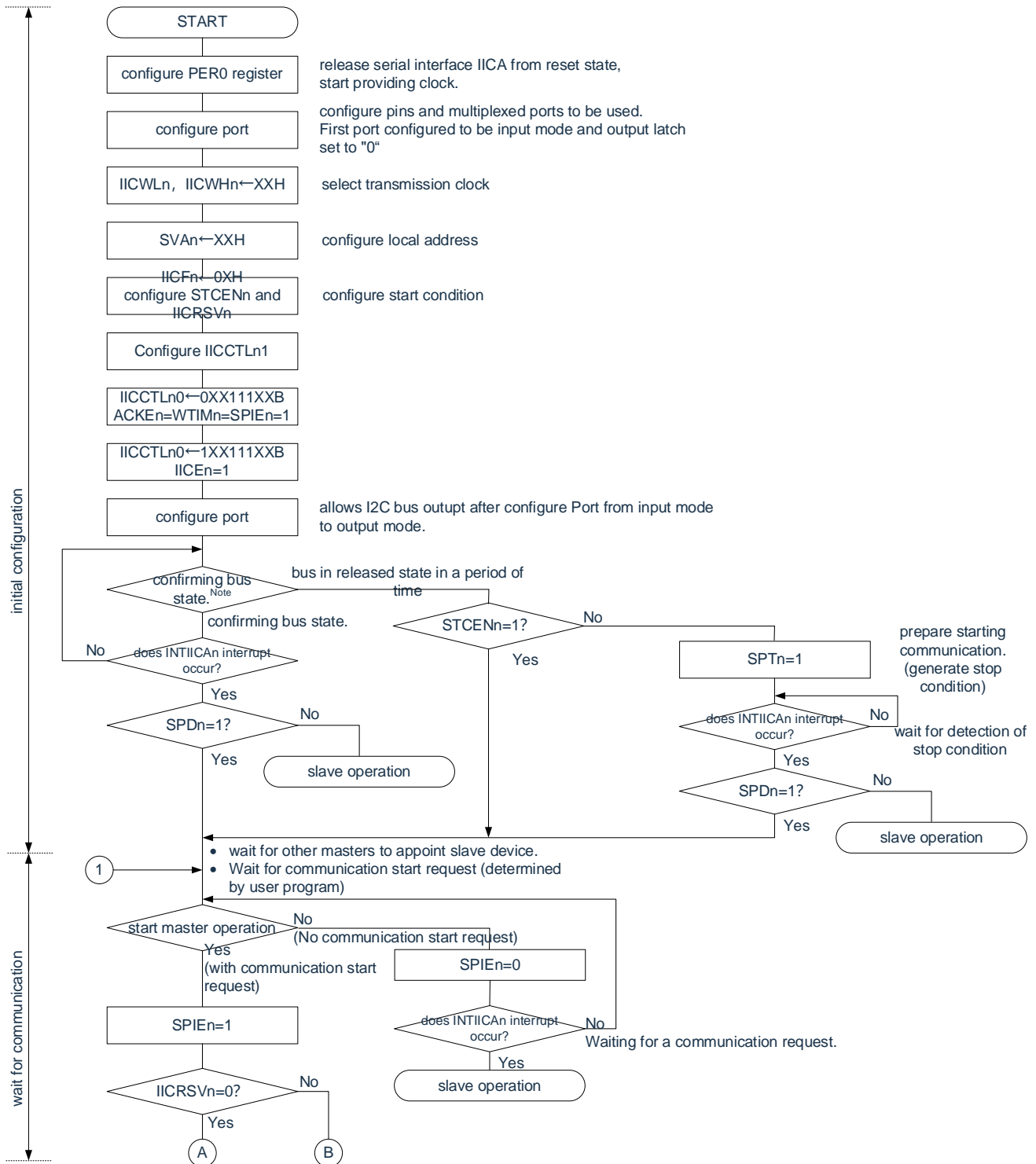
conform to the specifications of the products in the communication.

2.n=0,1

(2) Master operation of multi-master system

Figure 20-27

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 I2 C-bus (SCLAn pin and SDAAn) according to the specifications of the product in the communication Pin is high).

Note: n=0,1

Figure 20-28 Master operation of a multi-master system (2/3).

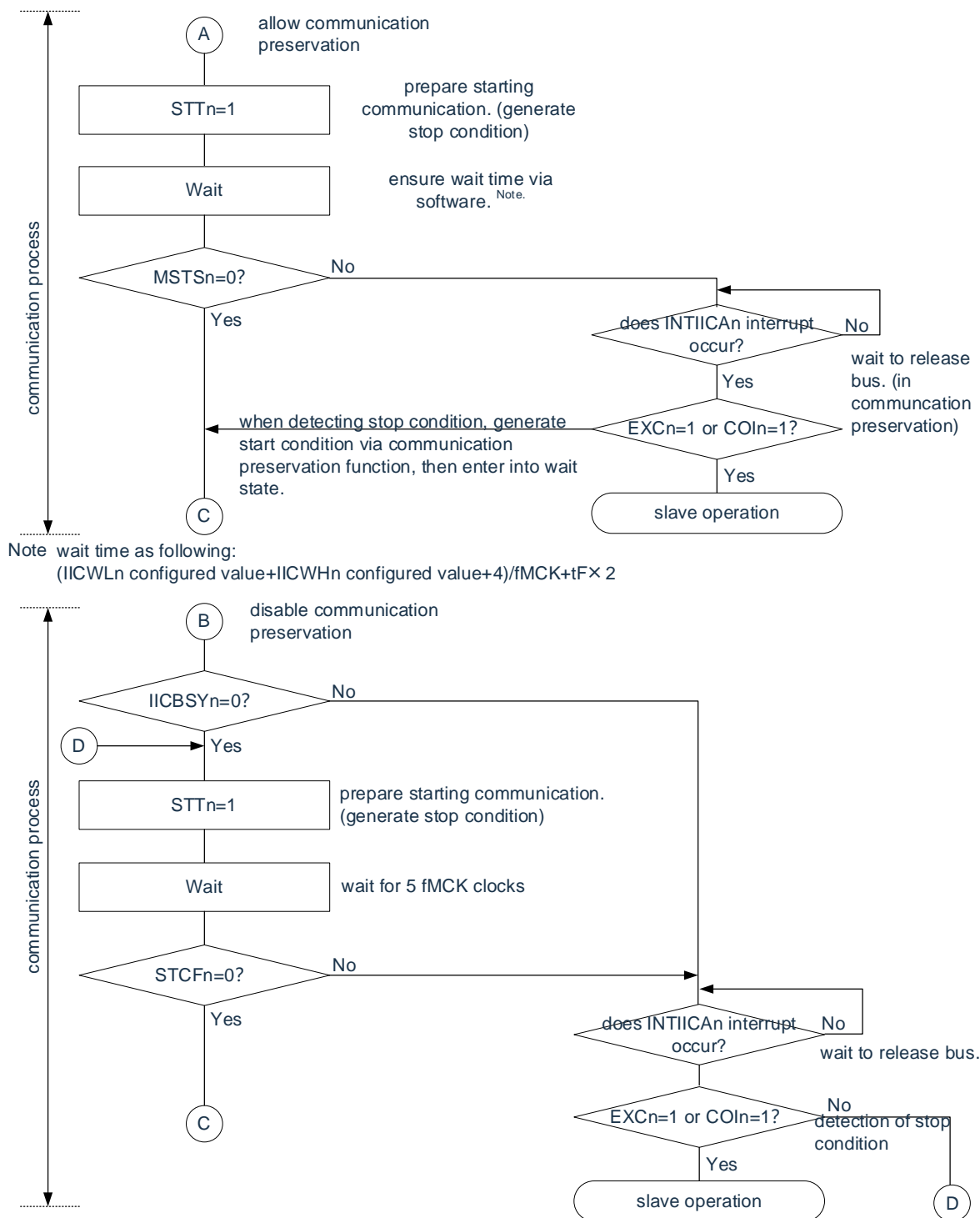
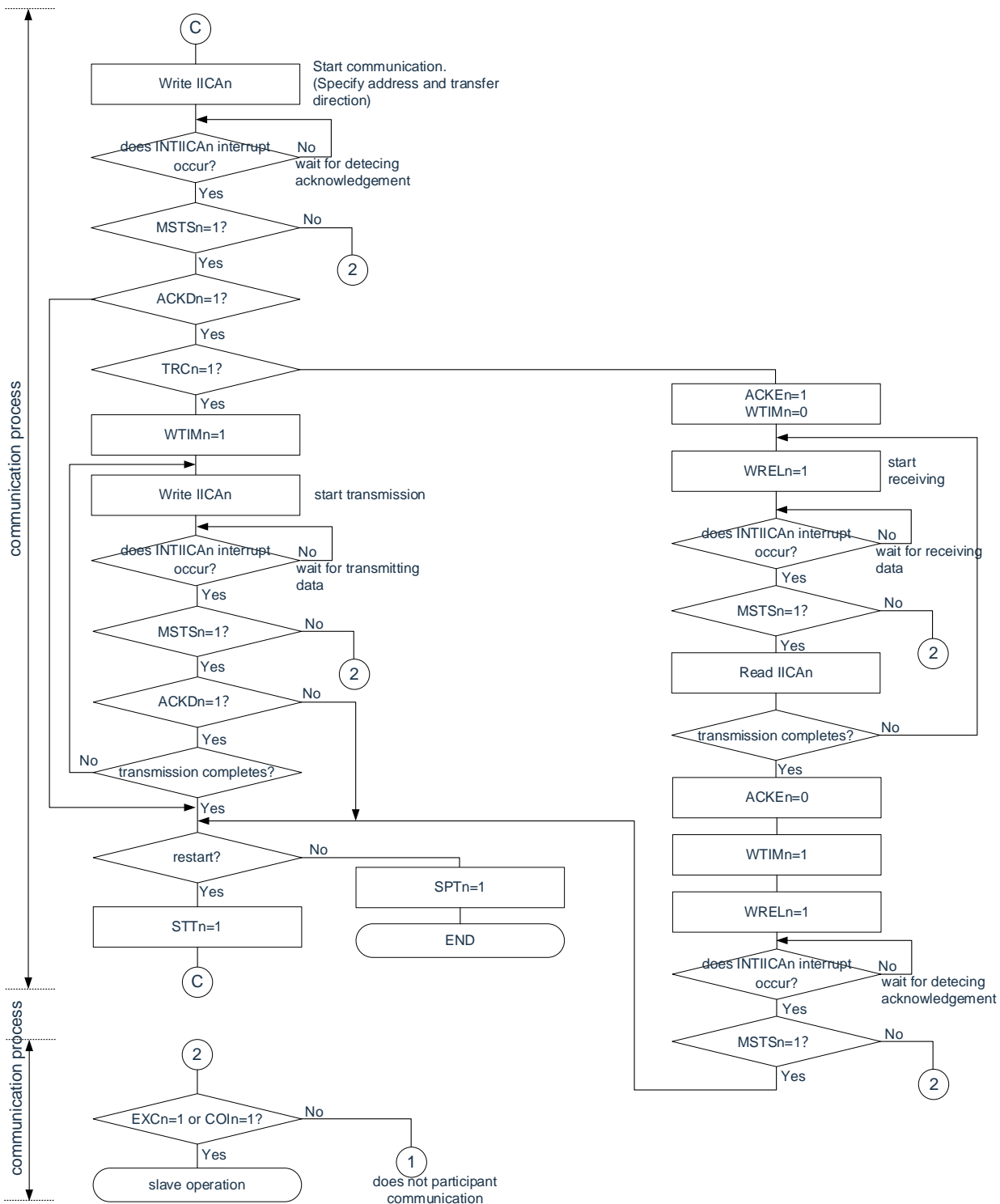


Figure 20-28: Master operation of a multi-master system (3/3).



Remarks: 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.

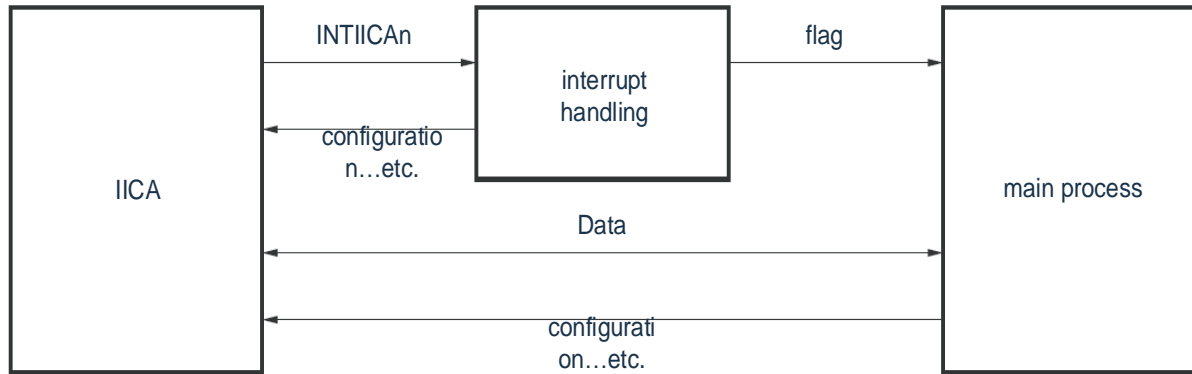
4. n=0.1

(3) Slave run

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 flags

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 sign

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 signs

This flag indicates the direction of communication, which is the same as the value of the TRCn bit.

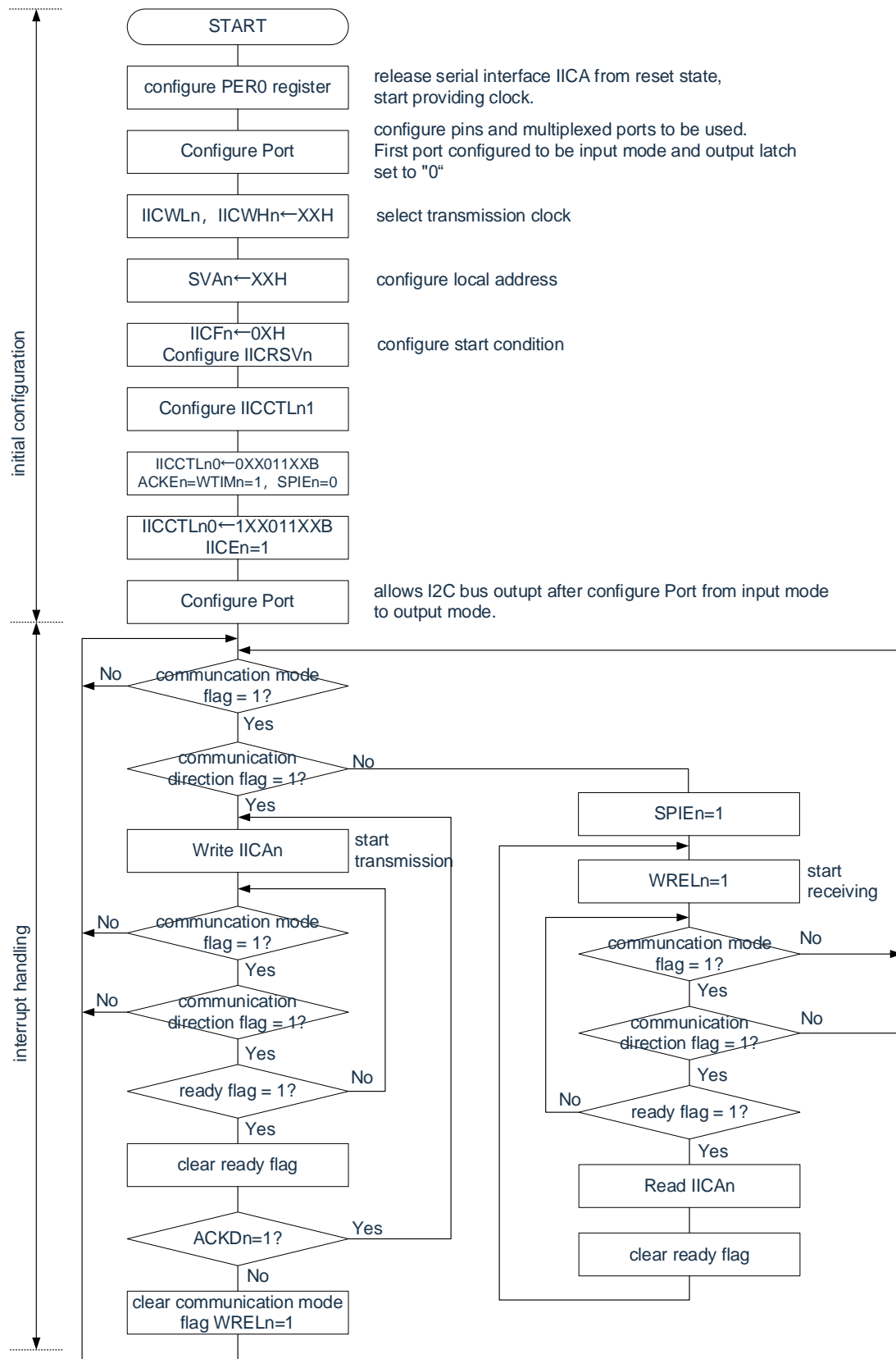
Note $n = 0, 1$

The run of the main processing unit of the slave run 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 20-28: Slave run step (1).



Remarks: 1 The format of transmission and reception must conform to the specifications of the product in the communication.

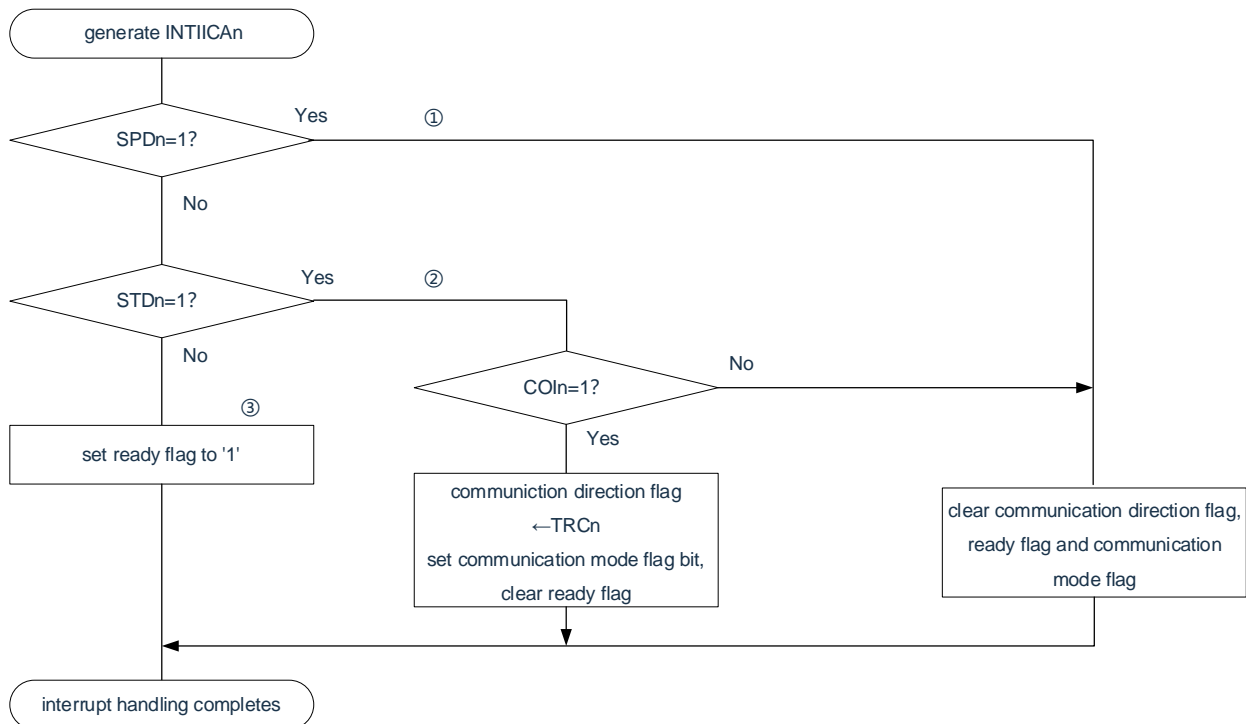
2. n=0.1

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 I2C bus remains waiting and returns from the interrupt as soon as the ready flag is set.

Note that (1) ~ (3) above corresponds to (1) ~ (3) of Figure 20-29 Slave run step (2). (2)".

Figure 20-29 Slave run step (2).



Note: n=0,1

20.5.17 Timing of the generation of I2C interrupt requests (INTIICAn).

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.

Remarks: 1. ST: Start condition

AD6~AD0 : Address

$\overline{R/W}$: The specified transmission direction

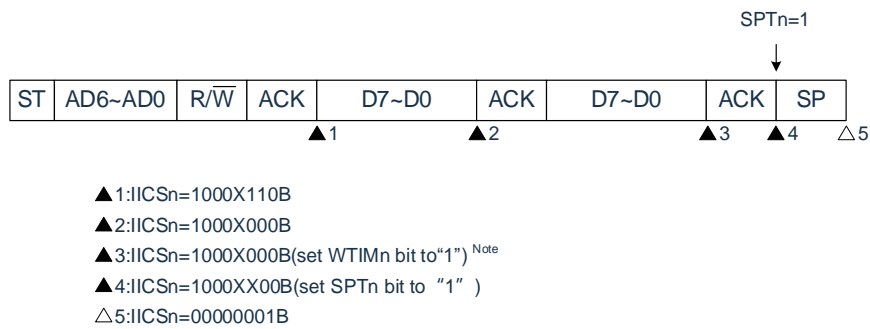
ACK : Acknowledge

D7~D0 : Data

SP : Stop Condition

2. n=0.1

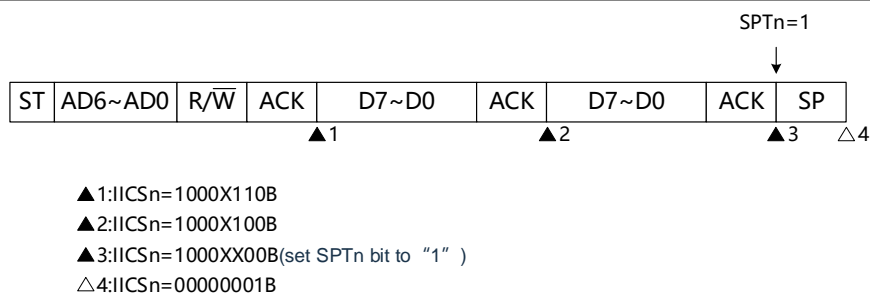
- (1) The master runs
- (a) Start~Address~Data~Data~Stop
- (i) In the case of WTIMn=0,1



Note: to generate stop condition, must set WTIMn 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) In the case of WTIMn=1

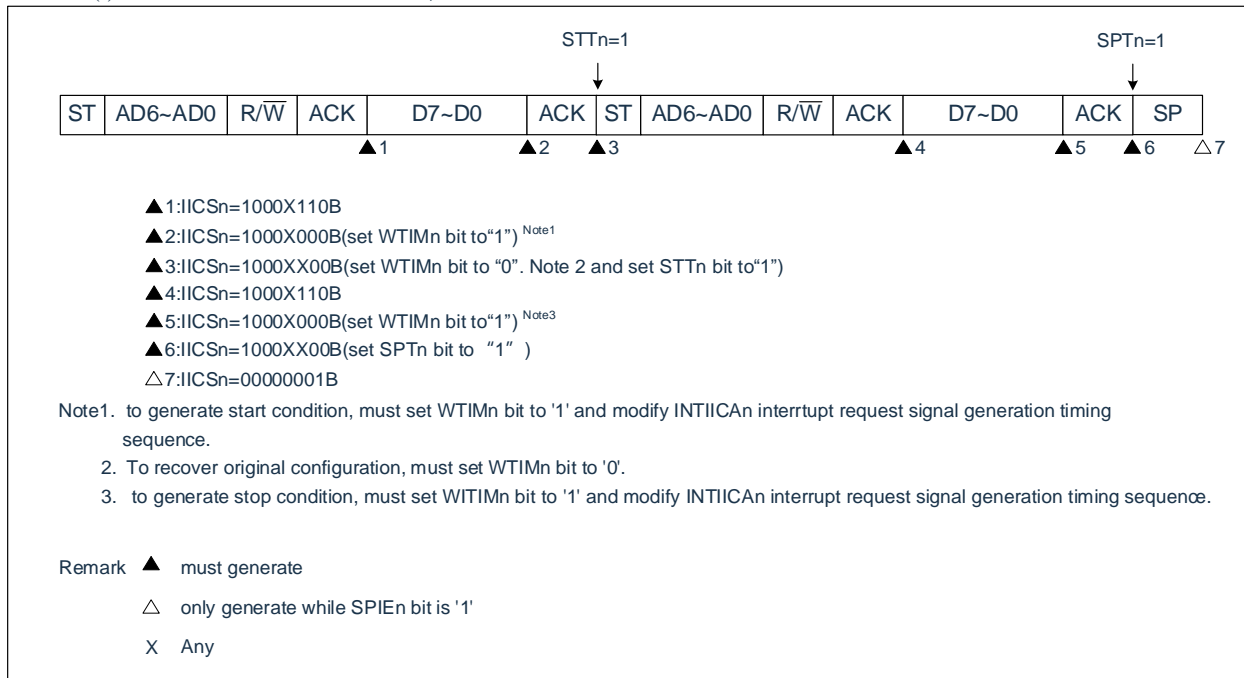


Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

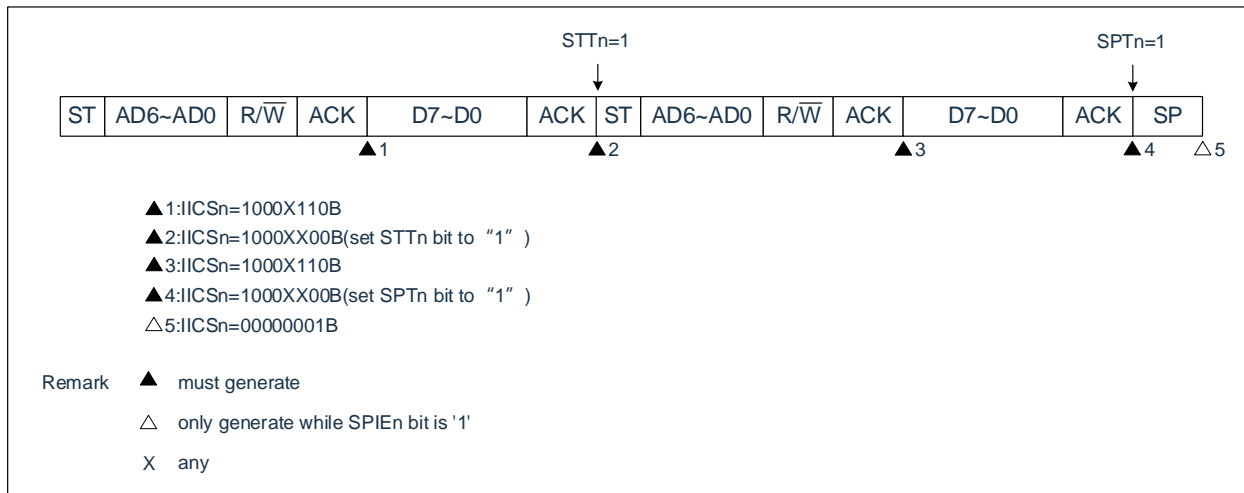
Note: n=0,1

(b) Start~Address~Data~Start~Address~Data~Stop (Start Over)

(i) In the case of WTIMn=0,1



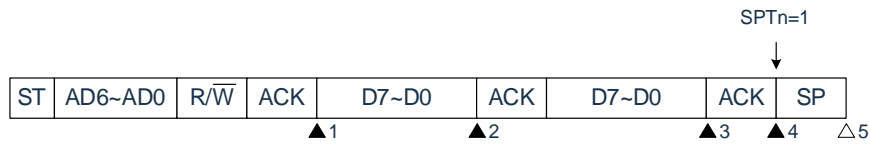
(ii) In the case of WTIMn=1



Note: n=0,1

(c) Start~Code~Data~Data~Stop

(i) In the case of WTIMn=0,1

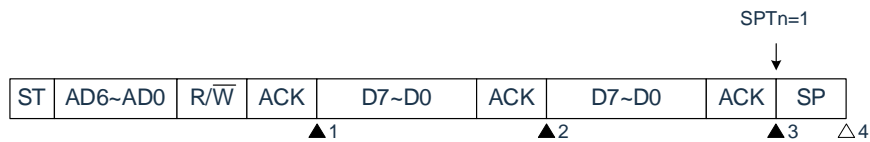


- ▲1:IICSn=1010X110B
- ▲2:IICSn=1010X000B
- ▲3:IICSn=1010X000B(set WTIMn bit to "1") ^{Note}
- ▲4:IICSn=1010XX00B(set SPTn bit to "1")
- △5:IICSn=00000001B

Note: 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) In the case of WTIMn=1



- ▲1:IICSn=1010X110B
- ▲2:IICSn=1010X100B
- ▲3:IICSn=1010XX00B(set SPTn bit to "1")
- △4:IICSn=00000001B

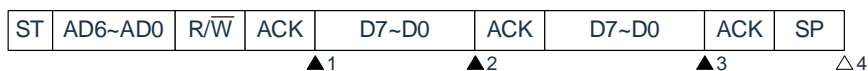
Remark ▲ must generate
 △ only generate while SPIEn bit is '1'
 X any

Note: n=0,1

(2) Slave operation (the case of receiving a slave address).

(a) Start~Address~Data~Data~Stop

(i) In the case of WTIMn=0,1



▲1:IICSn=0001X110B

▲2:IICSn=0001X000B

▲3:IICSn=0001X000B

△4:IICSn=00000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

(ii) In the case of WTIMn=1



▲1:IICSn=0001X110B

▲2:IICSn=0001X100B

▲3:IICSn=0001XX00B

△4:IICSn=00000001B

Remark ▲ must generate

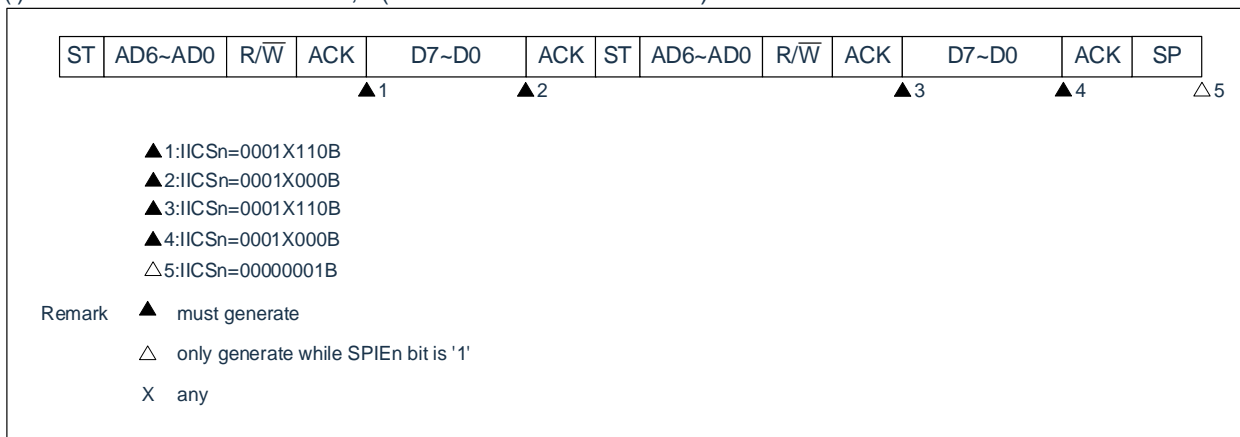
△ only generate while SPIEn bit is '1'

X any

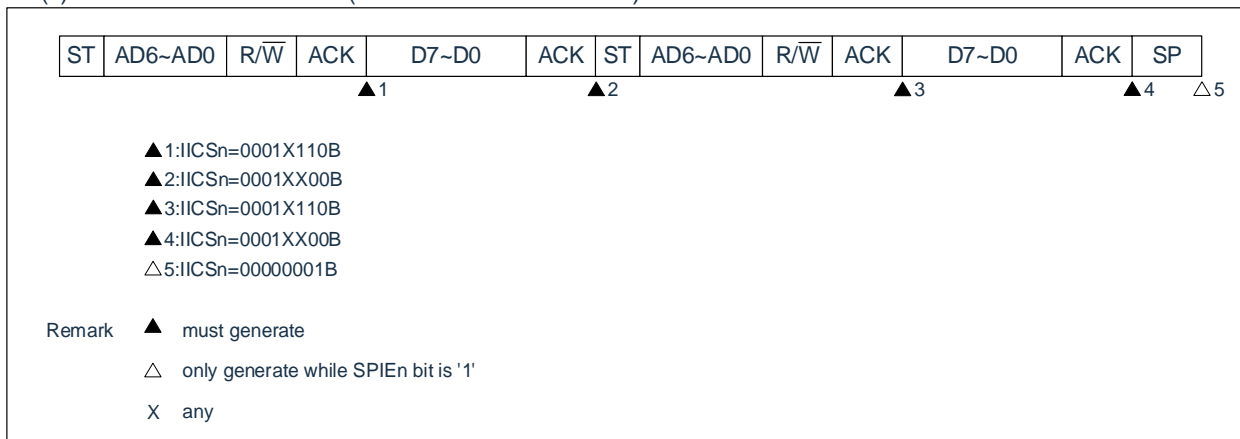
Note: n=0,1

(b) Start~Address~Data~Start~Address~Data~Stop

(i) The case where WTIMn = 0,1 (same for SVAn after restart).



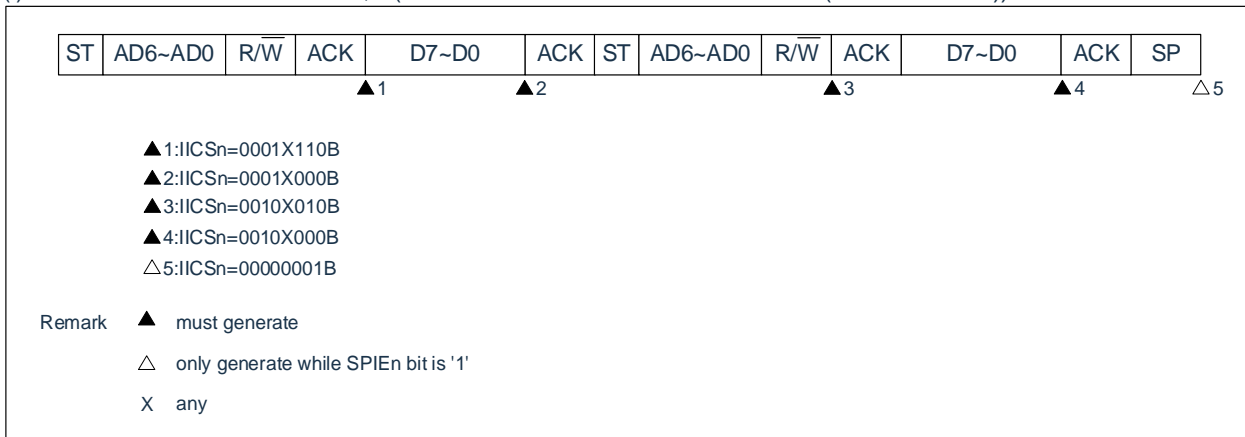
(ii) The case of WTIMn=1 (same SVAn after restart).



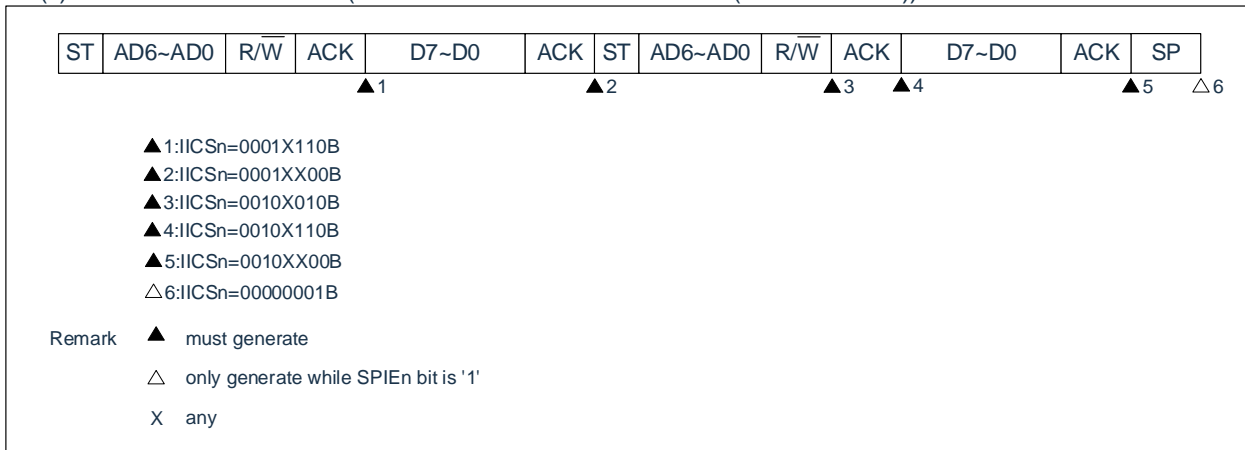
Note: n=0,1

(c) Start~Address~Data~Start~Code~Data~Stop

(i) The case of WTIMn= 0,1 (the address is different after restart (extension code)).



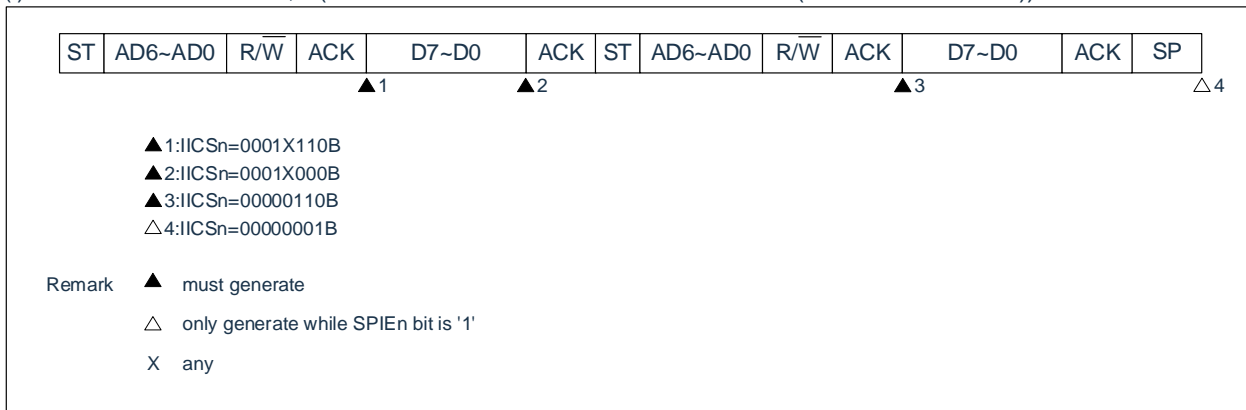
(ii) The case of WTIMn=1 (address is different after restart (extension code)).



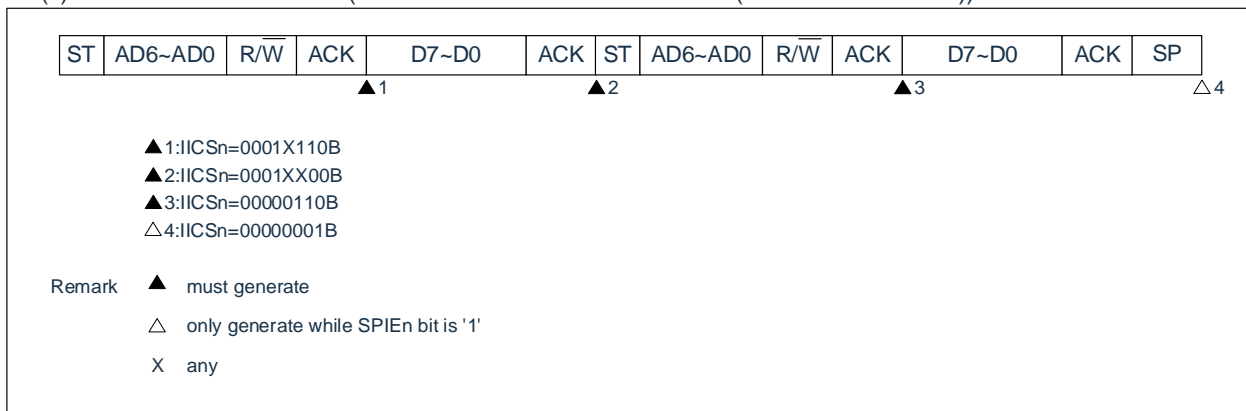
Note: n=0,1

(d) Start~Address~Data~Start~Address~Data~Stop

(i) The case of WTIMn=0,1 (the addresses are different after restart (non-extension code)).



(ii) The case of WTIMn=1 (different addresses after restart (non-extension code)).



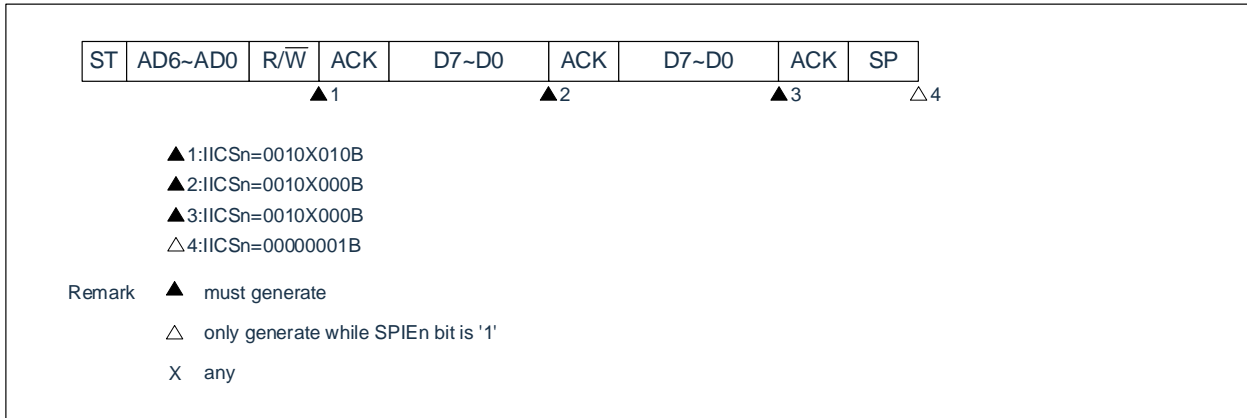
Note: n=0,1

(3) Slave run (in the case of receiving an extension code).

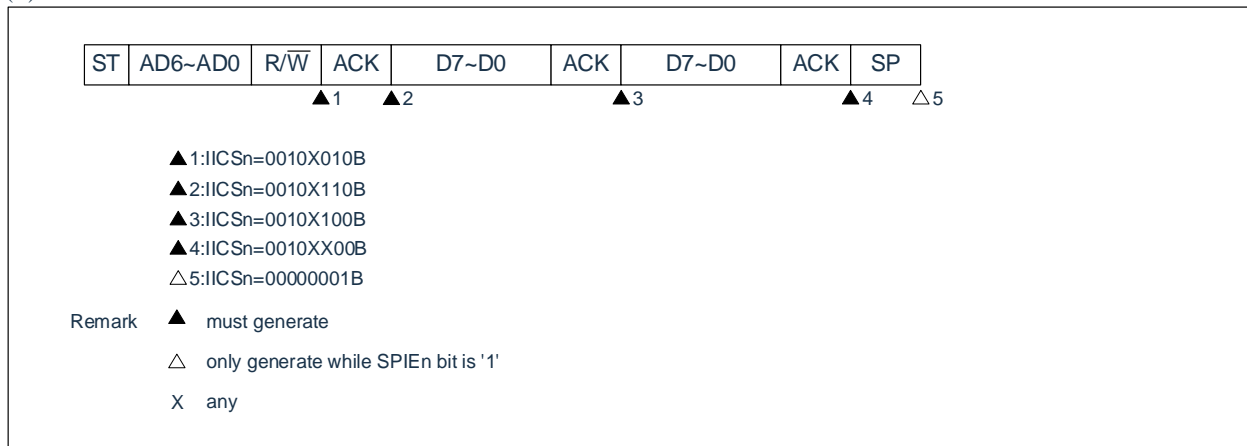
Always participate in the communication when receiving an extension code.

(a) Start~Code~Data~Data~Stop

(i) In the case of WTIMn=0,1



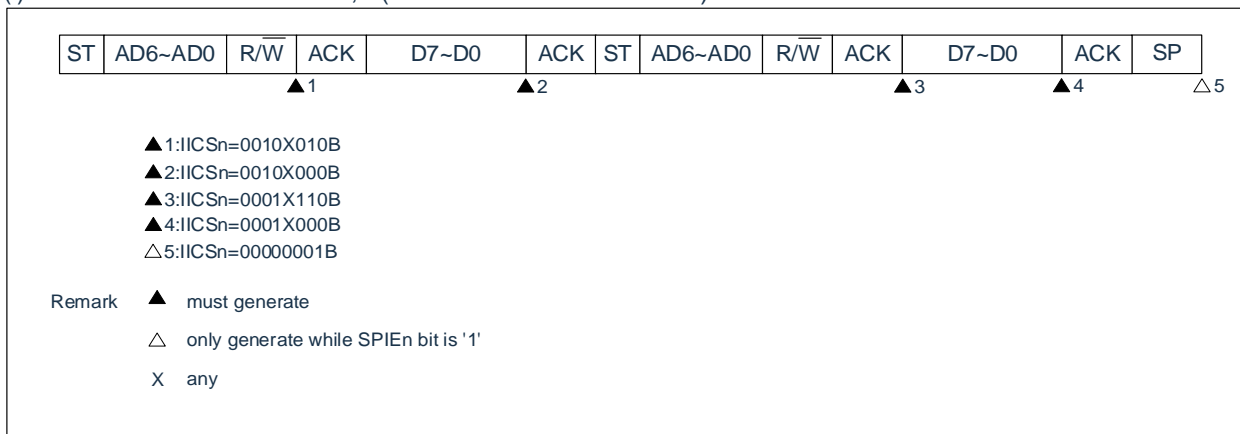
(ii) In the case of WTIMn=1



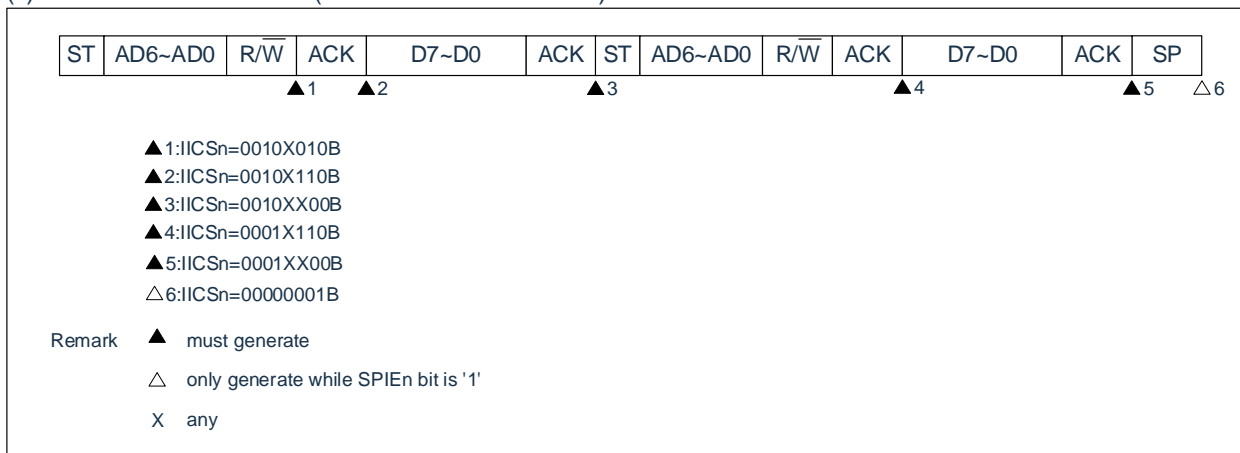
Note: n=0,1

(b) Start~Code~Data~Start~Address~Data~Stop

(i) The case where WTIMn = 0,1 (same for SVAn after restart).



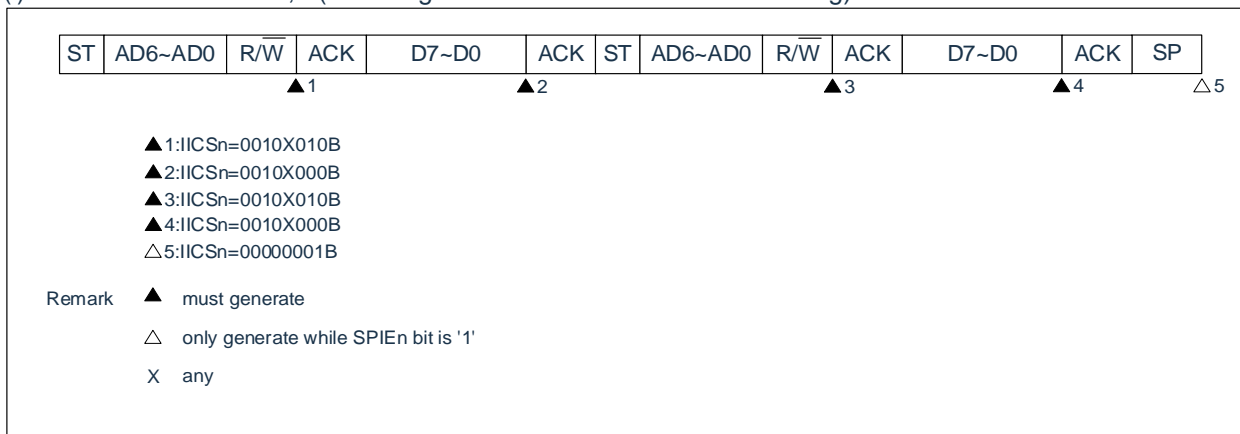
(ii) The case of WTIMn=1 (same SVAn after restart).



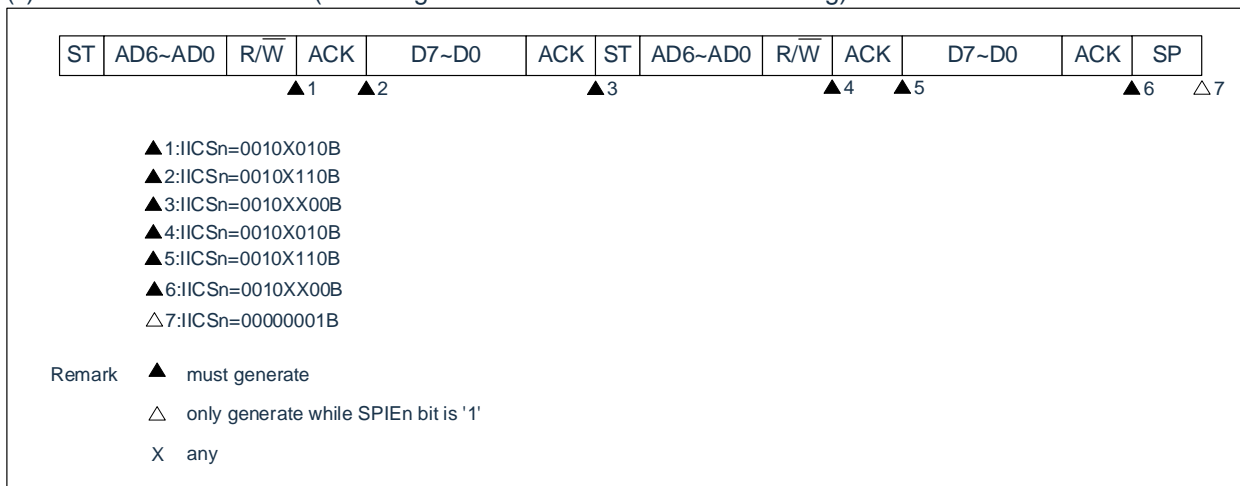
Note: n=0,1

(c) Start~Code~Data~Start~Code~Data~Stop

(i) The case of WTIMn=0,1 (receiving the extension code after restarting).



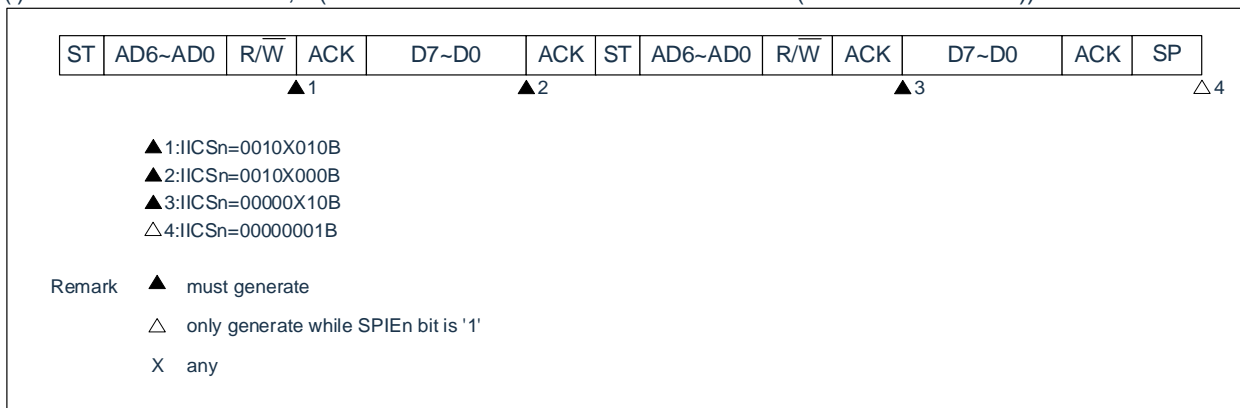
(ii) The case of WTIMn=1 (receiving the extension code after restarting).



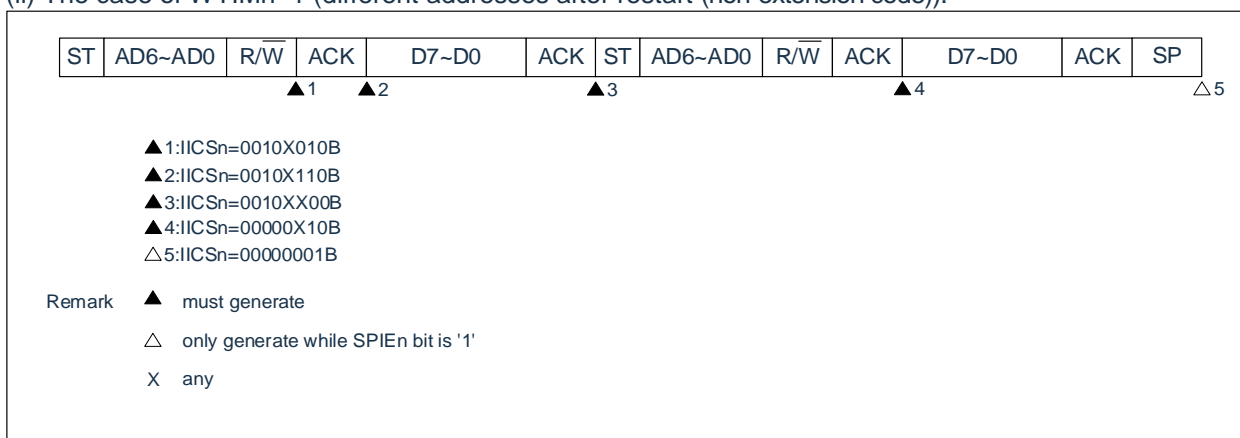
Note: n=0,1

(d) Start~Code~Data~Start~Address~Data~Stop

(i) The case of WTIMn=0,1 (the addresses are different after restart (non-extension code)).



(ii) The case of WTIMn=1 (different addresses after restart (non-extension code)).



Note: n=0,1

(4) Do not participate in the operation of the communication

(a) Start~Code~Data~Data~Stop



△1:IICSn=00000001B

Remark △ only generate while SPIEn bit is '1'

(5) The operation of the arbitration failure (running as a slave after the arbitration fails).

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) In the case of WTIMn=0,1



▲1:IICSn=0101X110B

▲2:IICSn=0001X000B

▲3:IICSn=0001X000B

△4:IICSn=00000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

Note: n=0,1

(ii) In the case of WTIMn=1



▲1:IICSn=0101X110B

▲2:IICSn=0001X100B

▲3:IICSn=0001XX00B

△4:IICSn=00000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

(b) A condition in which arbitration fails during the sending of an extension code

(i) In the case of WTIMn=0,1



▲1:IICSn=0110X010B

▲2:IICSn=0010X000B

▲3:IICSn=0010X000B

△4:IICSn=00000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

Note: n=0,1

(ii) In the case of WTIMn=1



▲1:IICSn=0110X010B

▲2:IICSn=0010X110B

▲3:IICSn=0010X100B

▲4:IICSn=0010XX00B

△5:IICSn=00000001B

Remark ▲ must generate

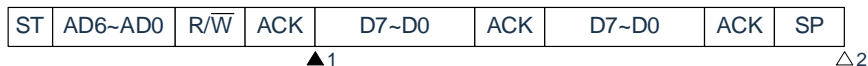
△ only generate while SPIEn bit is '1'

X any

(6) The operation of the arbitration failure (not participating in the communication after the arbitration failed).

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) The case where the arbitration fails during the sending of slave address data (WTIMn=1).



▲1:IICSn=01000110B

△2:IICSn=00000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

Note: n=0,1

(b) A condition in which arbitration fails during the sending of an extension code



▲1:IICSn=01000110B
set LRELn bit to '1' via software
△2:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'

(c) A condition in which the arbitration fails while transferring data

(i) In the case of WTIMn=0,1



▲1:IICSn=10001110B
▲2:IICSn=01000000B
△3:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

Note: n=0,1

(ii) In the case of WTIMn=1



▲1:IICSn=10001110B

▲2:IICSn=01000100B

△3:IICSn=00000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

(d) A situation where arbitration fails due to restart conditions when transferring data

(i) Non-extended codes (for example, SVAns are different).



▲1:IICSn=1000X110B

▲2:IICSn=01000110B

△3:IICSn=00000001B

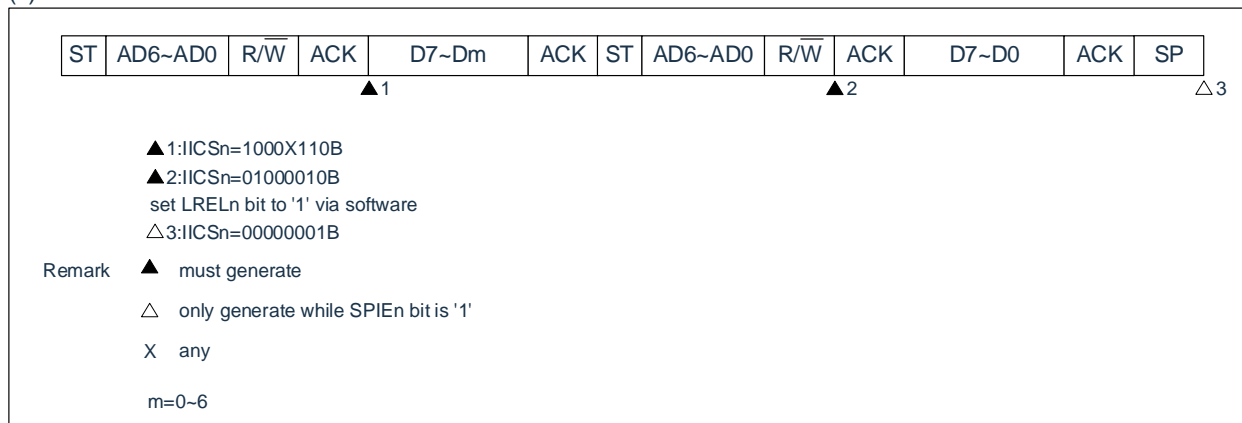
Remark ▲ must generate

△ only generate while SPIEn bit is '1'

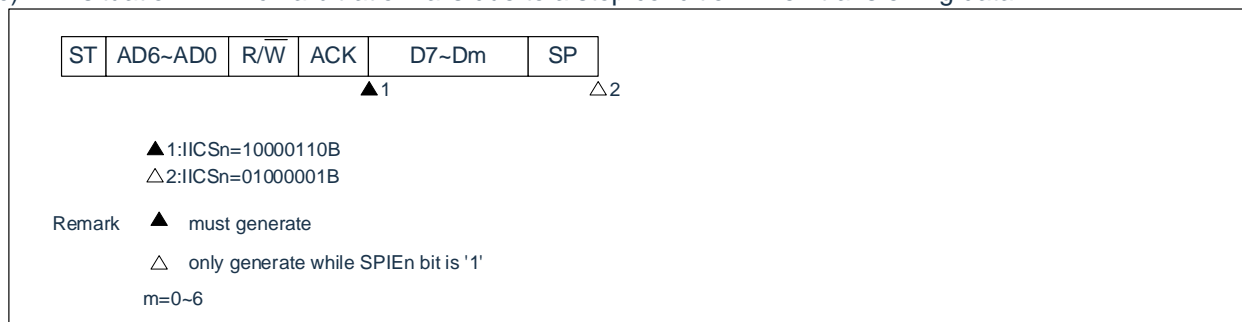
X any

Note: n=0,1

(ii) Extension code



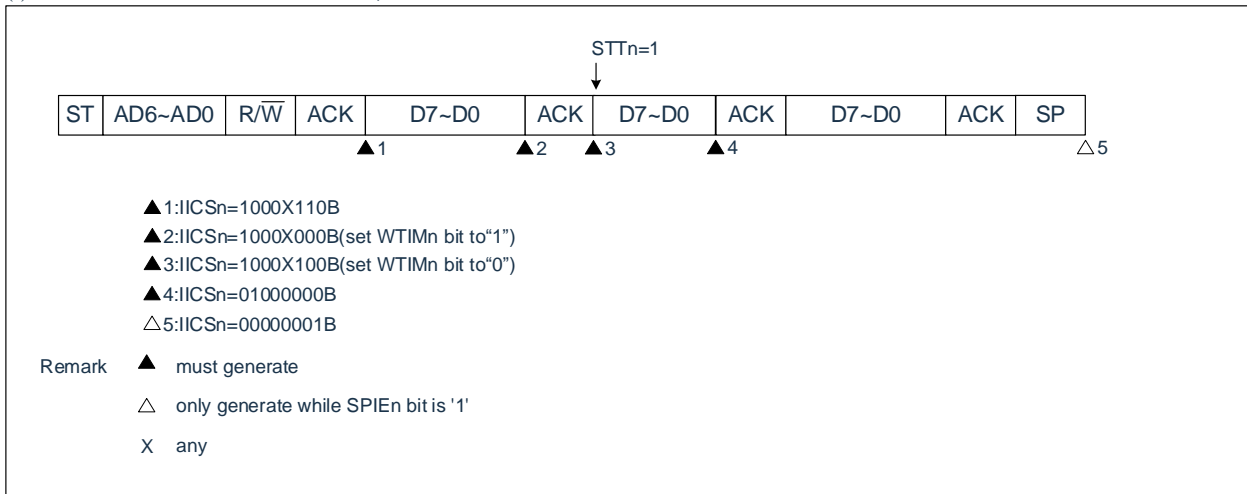
(e) A situation in which arbitration fails due to a stop condition when transferring data



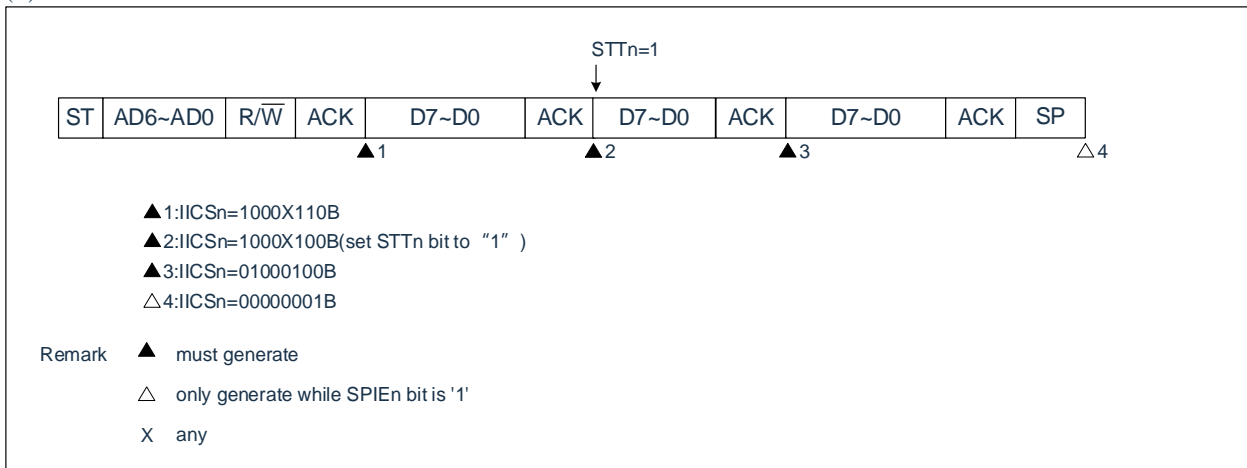
Note: n=0,1

(f) A situation where the arbitration fails because the data is low when you want to generate a restart condition

(i) In the case of WTIMn=0,1



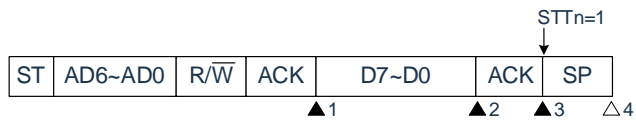
(ii) In the case of WTIMn=1



Note: n=0,1

(g) A case where the arbitration fails because of the stop condition when the restart condition is wanted to be generated

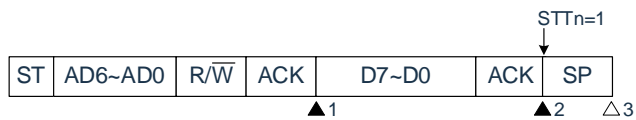
(i) In the case of $WTIMn=0,1$



- ▲1:IICSn=1000X110B
- ▲2:IICSn=1000X000B(set $WTIMn$ bit to "1")
- ▲3:IICSn=1000XX00B(set $STTn$ bit to "1")
- △4:IICSn=01000001B

Remark ▲ must generate
 △ only generate while $SPIEn$ bit is '1'
 X any

(ii) In the case of $WTIMn=1$



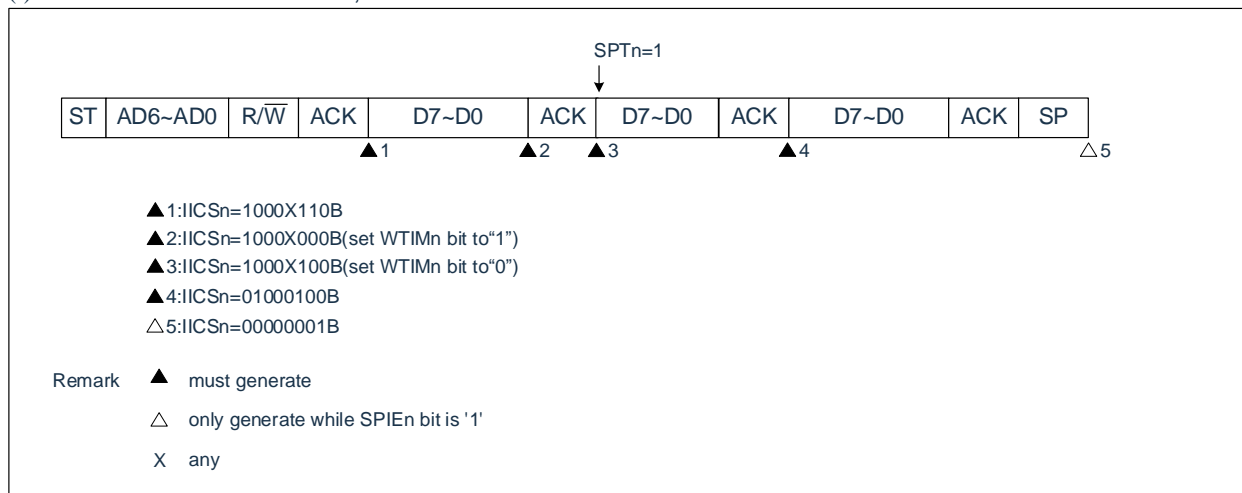
- ▲1:IICSn=1000X110B
- ▲2:IICSn=1000XX00B(set $STTn$ bit to "1")
- △3:IICSn=01000001B

Remark ▲ must generate
 △ only generate while $SPIEn$ bit is '1'
 X any

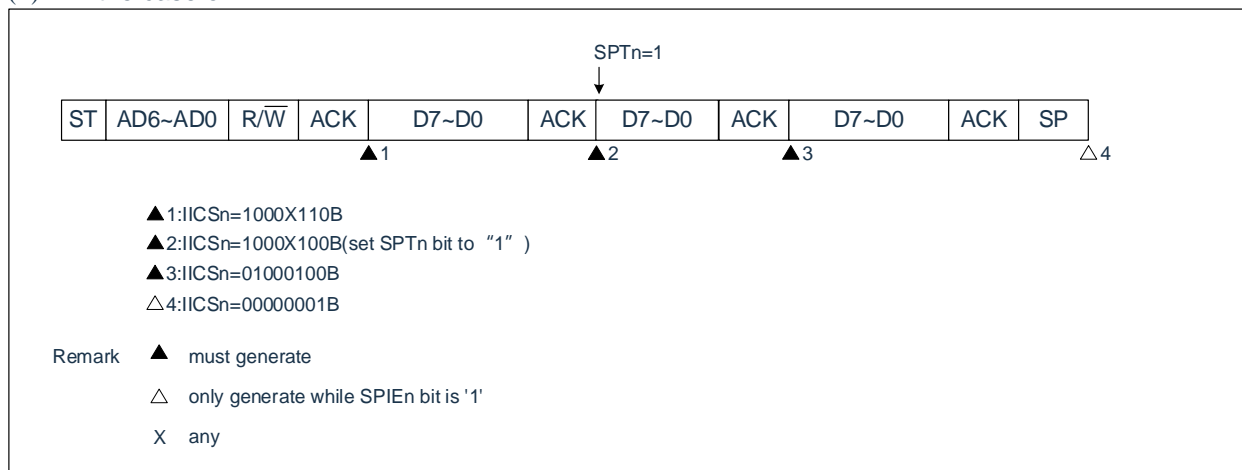
Note: $n=0,1$

(h) A situation where the arbitration fails because the data is low when you want to generate a stop condition

(i) In the case of WTIMn=0,1



(ii) In the case of WTIMn=1



Note: n=0,1

20.6 Timing diagram

In I2C-bus mode, the master device selects a slave device for a communication object from multiple slave devices by outputting an address to the serial bus. The master device sends a TRCn bit (bit3 of the IICA status register n (IICSn)) that indicates the direction of data transmission after the slave address. Begin serial communication with the slave. The timing diagram of data communication is shown in FIG 20-30 and FIG 20-31.

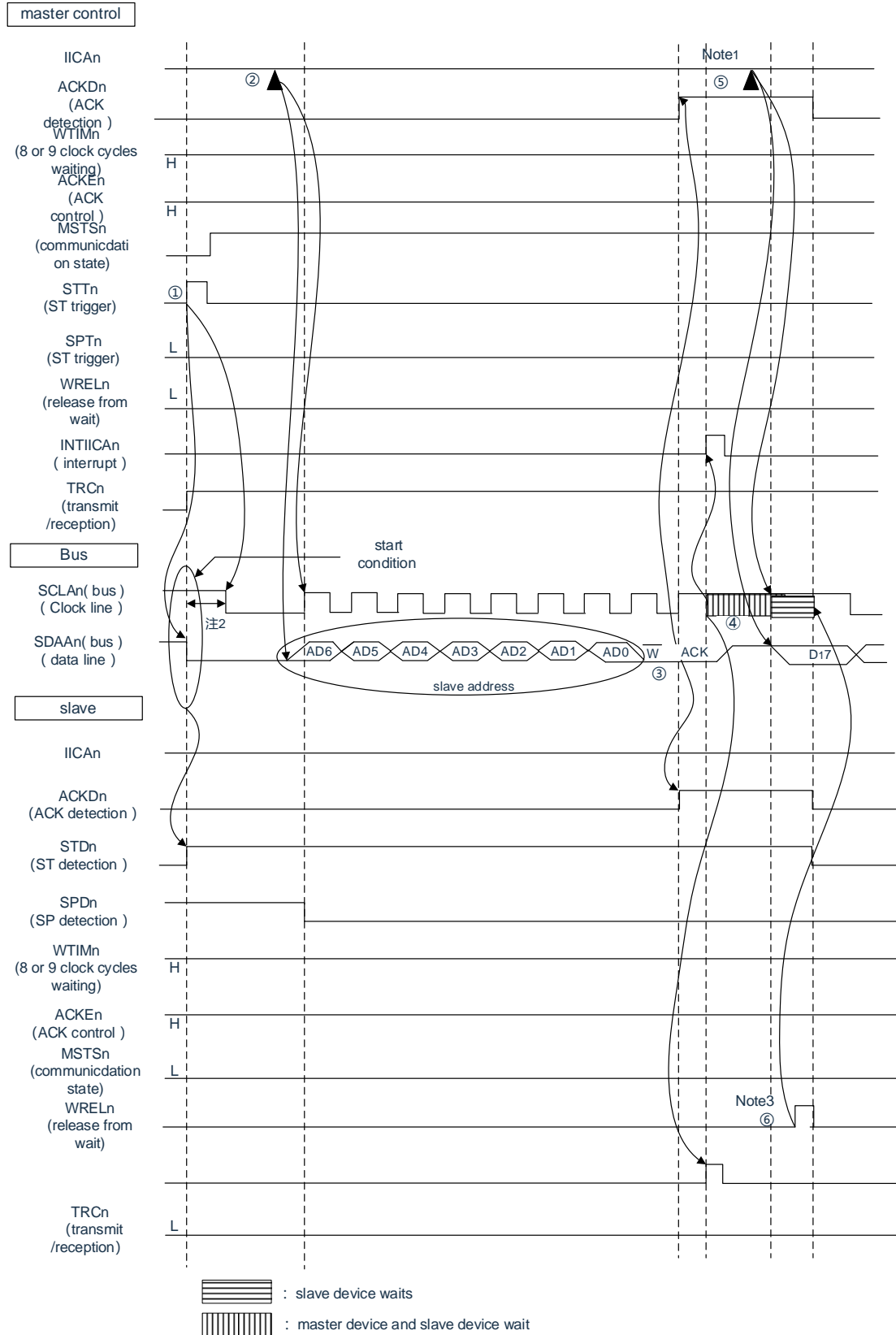
The shift of the IICA shift register n (IICAn) is carried out synchronously with the falling edge of the serial clock (SCLAn), and the transmit data is transmitted to the SO latch, in MSB Prioritize outputting data from the SDAAn pin.

Take the data from the SDAAn pin input to the IICAn on the rising edge of the SCLAn.

Note: n=0,1

FIG 20-30 example of a slave device → the 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 WRELn position bit.

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.

3. To lift the wait during the slave receive, the IICAn must be placed in the "FFH" or WRELn position.

FIG 20-30(1) start condition ~ address ~ data" of FIG-20-30 is 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=0,1), 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=0,1) 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 lifts 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.

Remarks: 1 FIG 20-30~(15)A series of operational steps for data communication via the I2C bus.

FIG 20-30of the "(1) start condition ~ address ~ data" illustrates steps (1) ~ (6).

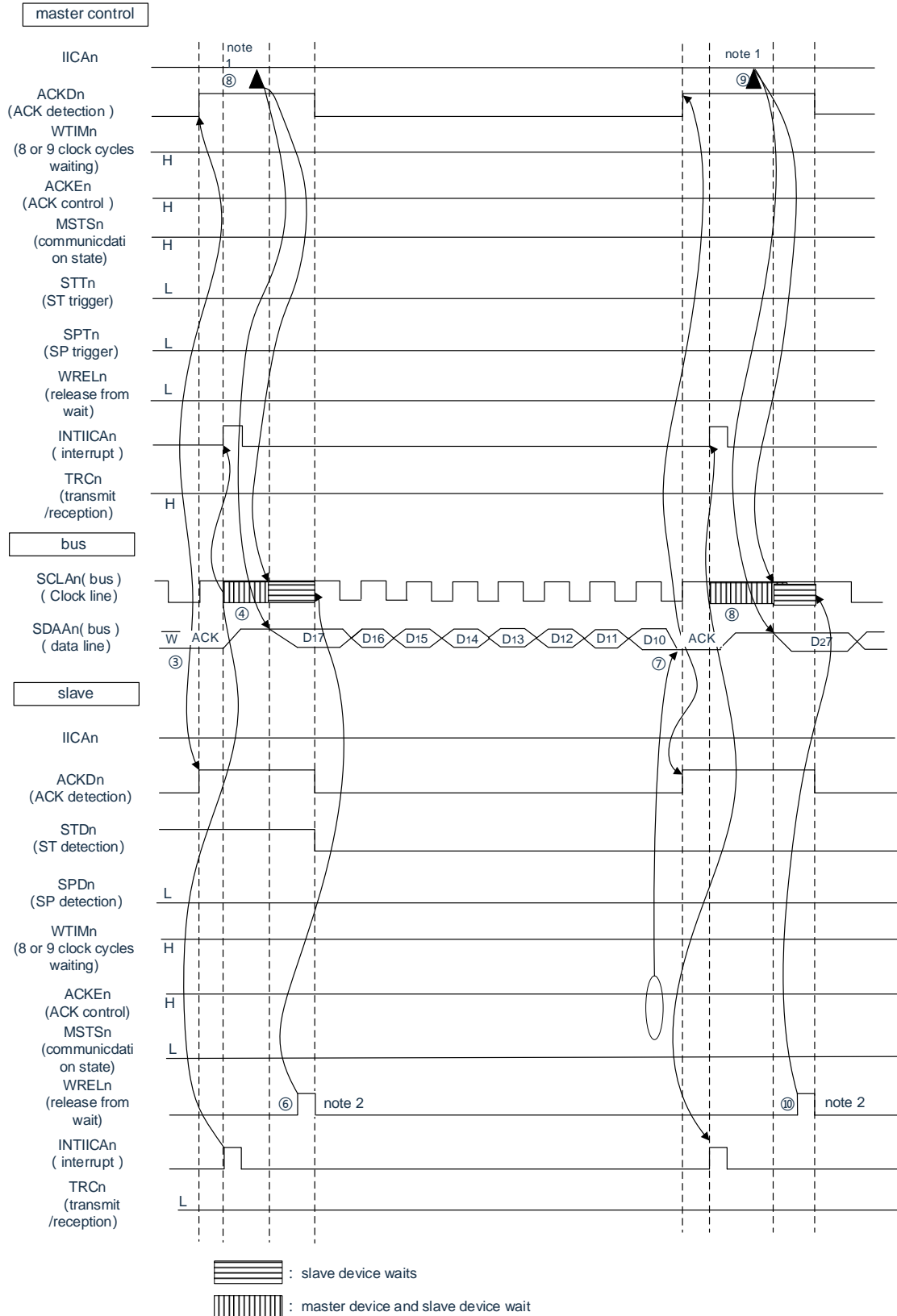
FIG 20-30 of the "(2) address ~ data ~ data" illustrates steps (3) ~ (10).

FIG 20-30of the "(3) data ~ data ~ stop condition" illustrates step (7) ~ . 15

2.n=0,1

FIG 20-31 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


Note: 1 To remove the wait during the master send, the IICAn must be written to the data instead of the WRELn position bit.

2. To lift the wait during slave reception, the IICAn must be placed in the "FFH" or WRELn position.

FIG 20-30 of "(2) address ~ data ~ data" of (3) ~ (10) is described as follows:

- (3) On the slave, if the receiving address and the local station address (the value of the SVAn) are the same ^{note}, the ACK is sent to the master through the hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).
- (4) The master generates an interrupt on the falling edge of the 9th clock (INTIICAn: address send end interrupt). Slaves with the same address enter a waiting state (SCLAn=0,1) and an interrupt (INTIICAn: address matching interrupt) ^{note}.
- (5) The master writes and sends data to the IICA shift register n (IICAn) to remove the wait of the main controller.
- (6) If the slave lifts 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= 0,1) 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.

Note 1 FIG 20-30A series of operational steps for data communication via the I2C bus.

FIG 20-30 of the "(1) start condition ~ address ~ data" illustrates steps (1) ~ (6).

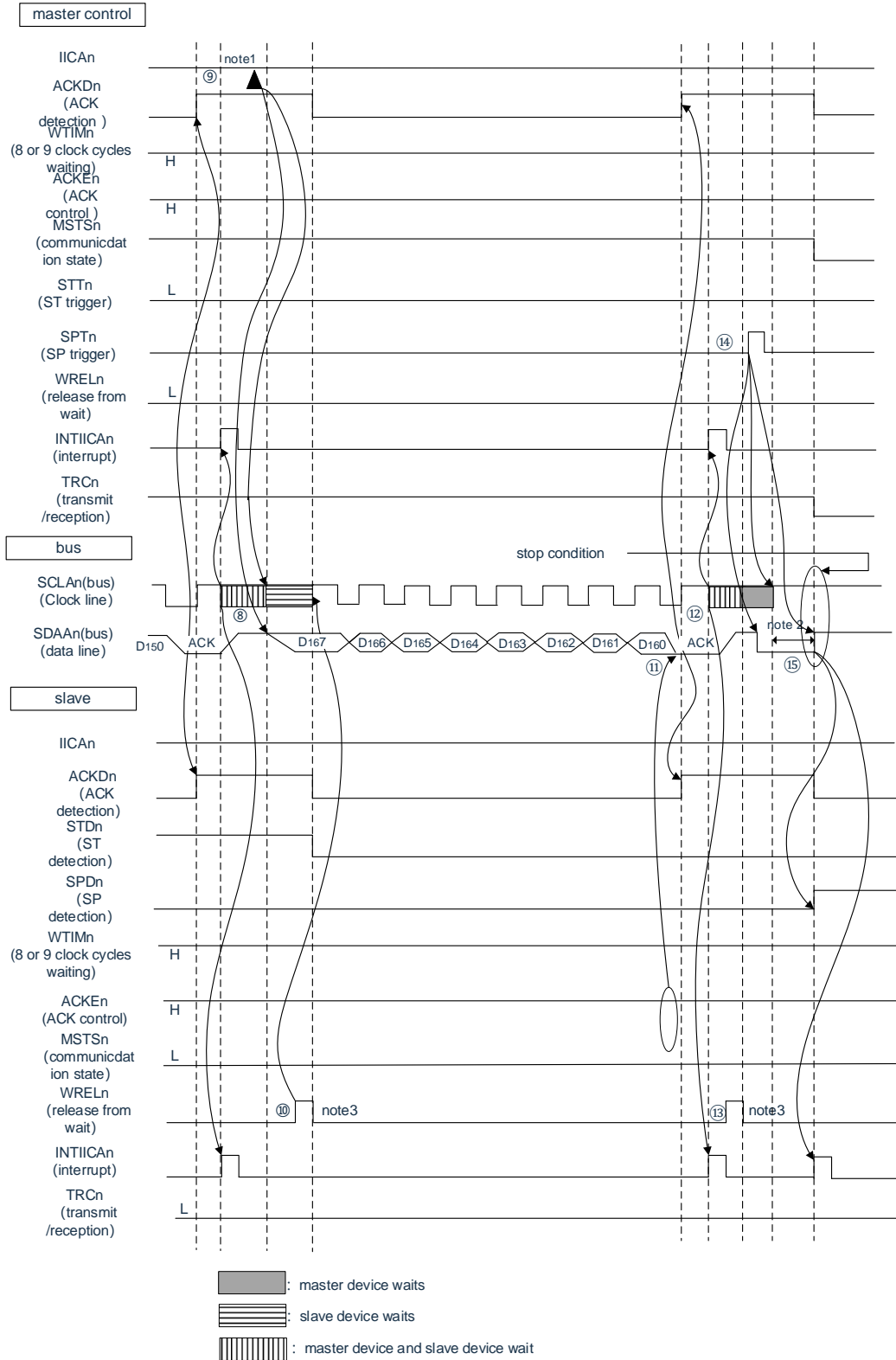
FIG 20-30 of the "(2) address ~ data ~ data" illustrates steps (3) ~ (10).

FIG 20-30 of the "(3) data ~ data ~ stop condition" illustrates step (7) ~ . 15

2.n=0,1

FIG 20-31 Communication example of a master device → slave device
(Master: Select 9 clocks to wait, Slave: Choose 9 clocks to wait) (3/4).

(2) Data ~ data ~ stop condition



Note 1 To remove the wait during the master send, the IICAn must be written to the data instead of the WRELn position bit.

2. After the stop condition is issued, the time from the SCLAn pin signal to generate the stop condition is at least 4.0μs when set to standard mode and at least 0.6μs when set to fast mode.

3. To lift the wait during the slave receive, the IICAn must be placed in the "FFH" or WRELn position.

FIG 20-30 of "(3) data ~ data ~ stop condition" of (7) ~ (15) description is as follows:

- 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=0,1) 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=0,1) 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=0,1) 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).

Note 1 FIG 20-30~(15) A series of operational steps for data communication via the I2C bus.

FIG 20-30of the "(1) start condition ~ address ~ data" illustrates steps (1) ~ (6).

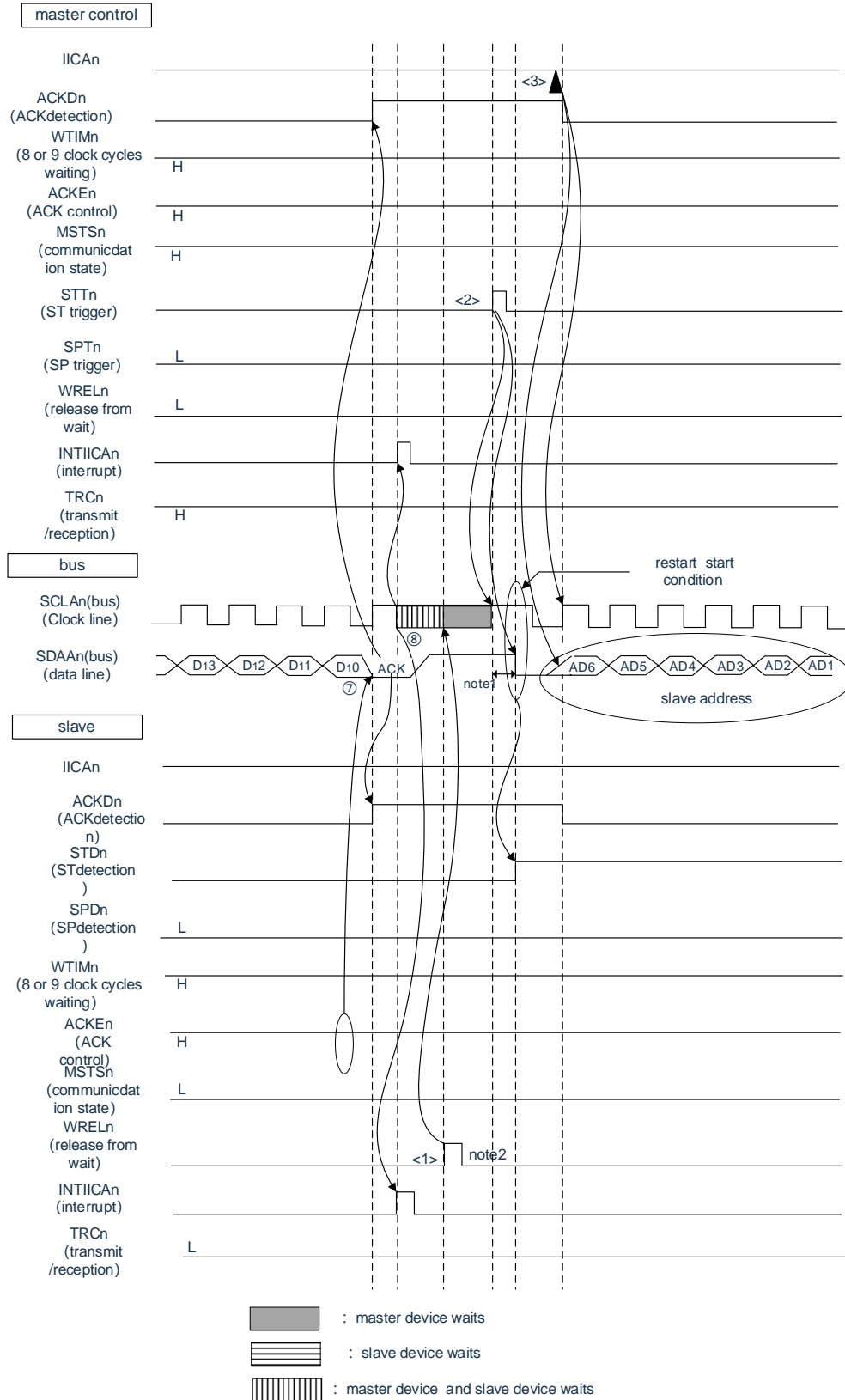
FIG 20-30 of the "(2) address ~ data ~ data" illustrates steps (3) ~ (10).

FIG 20-30of the "(3) data ~ data ~ stop condition" illustrates step (7) ~ . t_s

2.n=0,1

FIG 20-31 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.

2. To lift the wait during slave reception, the IICAn must be placed in the "FFH" or WRELn position.

FIG 20-30(4) data ~ restart condition ~ address" of Fig.20-30 are as follows. After performing steps (7) and (8), perform <1> to <3>, and return to the data sending step of step (3).

(7) After the data transfer is completed, because the ACKEn bit of the slave is "1", the ACK is sent to the master control through hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).

(8) Both the master and the slave enter a waiting state (SCLAn= 0,1) 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 lifted (WRELn=1).

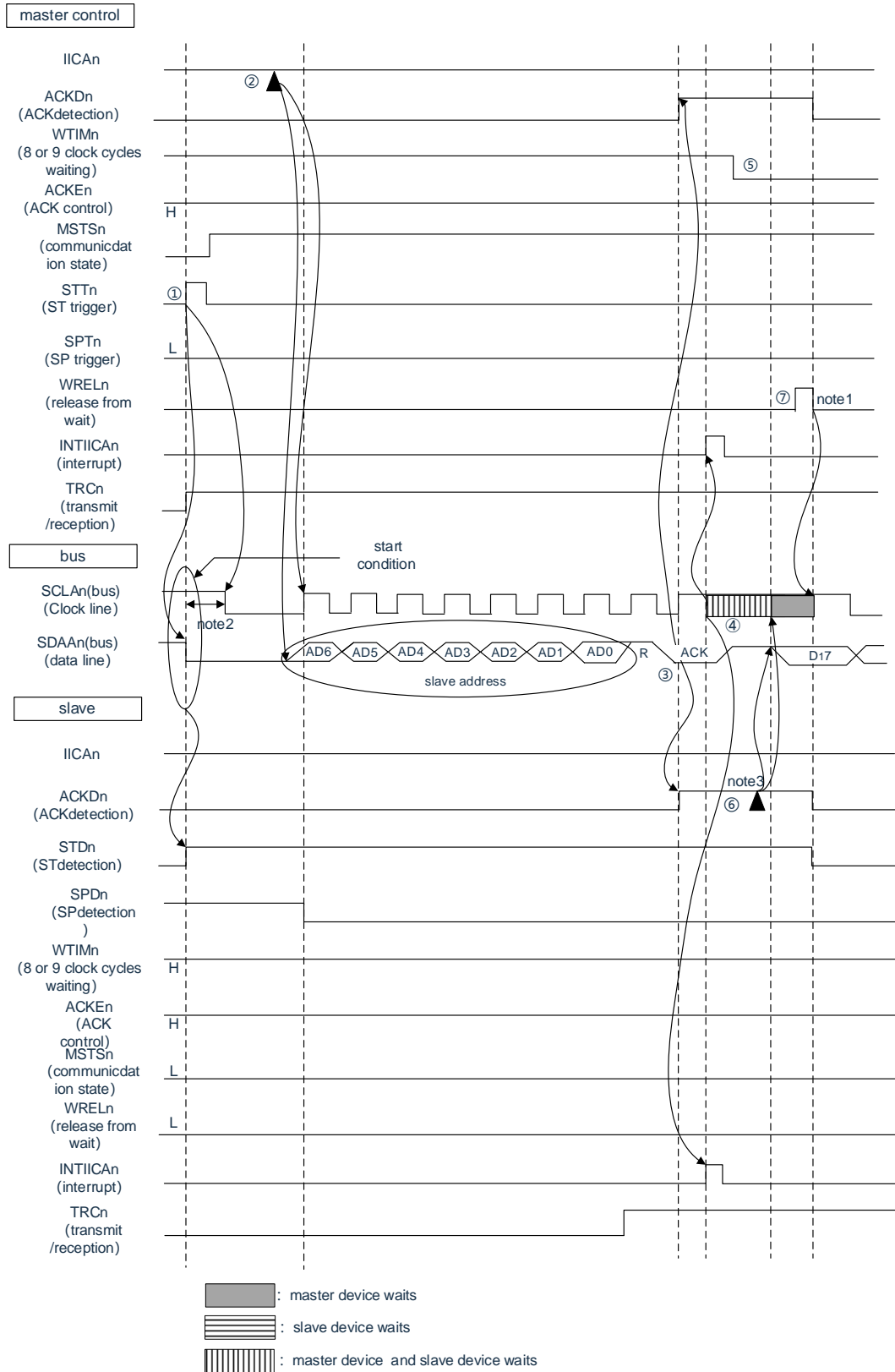
<2> if the master triggers the start condition again (STTn=1), the bus clock line rises (SCLAn=1) and the bus data line drops (SDAAn=0,1) after the preparation time for the new start condition), generate start conditions (change SDAAn from "1" to "0" by SCLAn=1). Then, if a start condition is detected, the bus clock line drops (SCLAn=0,1) 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.

Note: n=0,1

FIG 20-31 A communication example of a slave device → a master device

(Master: Select 8 clocks to wait, Slave: Choose 9 clocks to wait) (1/3).

(1) Start Condition ~ Address ~ Data


Note: 1 To remove the wait during the master's reception, the IICAn must be placed in the "FFH" or WRELn position.

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.

3. To undo the wait during the slave send, the data must be written to the IICAn instead of the WRELn position bit.

FIG 20-31 of the "(1) start condition ~ address ~ data" of (1) ~ (7) description is 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 (MSTS_n=1) and after the hold time elapses the bus clock line drops (SCLAn=0,1), ending the communication preparation.
- ② If the master writes address +R (receive) to the IICA shift register n (IICAn), the slave address is sent.
- ③ On the slave, if the receiving address and the local station address (the value of the SVAn) are the same note, an ACK is sent to the master through hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).
- ④ The master generates an interrupt on the falling edge of the 9th clock (INTIICAn: address send end interrupt). Slaves with the same address enter a waiting state (SCLAn=0,1) and an interrupt (INTIICAn: address matching interrupt) ^{note}.
- ⑤ The master changes the wait sequence to the 8th clock (WTIMn=0,1).
- 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.

Note 1 FIG 20-31(19) A series of operating steps for data communication via the I2C bus.

FIG 20-31 of the "(1) start condition ~ address ~ data" illustrates steps (1) ~ (7).

FIG 20-31 of the "(2) address ~ data ~ data" illustrates steps (3) ~ (12).

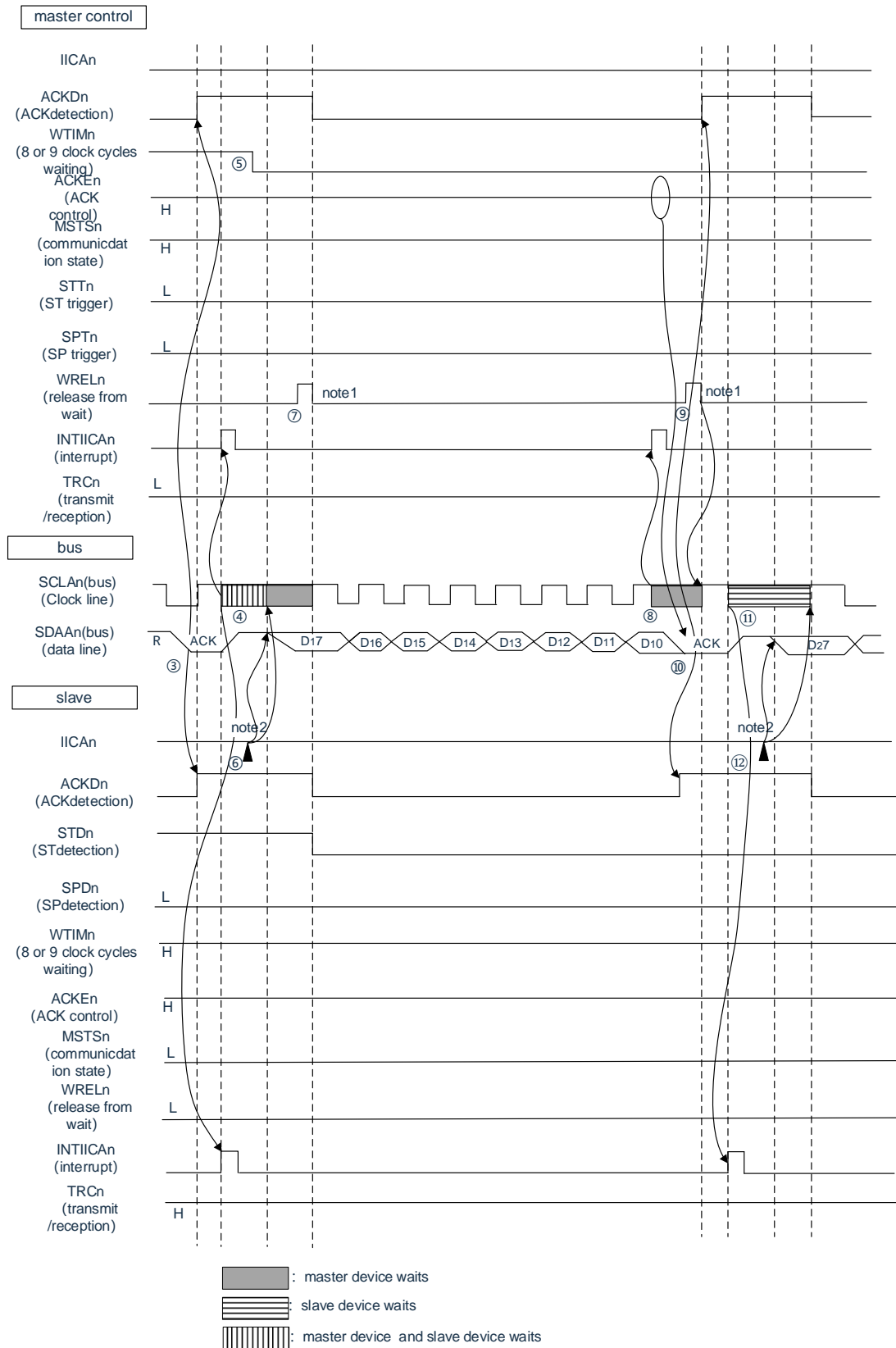
FIG 20-31 of the "(3) data ~ data ~ stop condition" illustrates steps (8) ~ (19).

2.n=0,1

Fig. 20-32 Example of communication between the slave device → the master device

(Master: Select 8 clocks to wait, Slave: Choose 9 clocks to wait) (2/3).

(2) Address ~ data ~ data



Note 1 To remove the wait during the master's reception, the IICAn must be placed in the "FFH" or WRELn position.

2. To remove the wait during the slave send, the IICAn must be written to the data instead of the WRELn position bit.

FIG 20-31 of the "(2) address ~ data ~ data" of (3) ~ (12) is described as follows:

- (3) On the slave, if the receiving address and the local station address (the value of the SVAn) are the same ^{note}, the ACK is sent to the master through the hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).
- (4) The master generates an interrupt on the falling edge of the 9th clock (INTIICAn: address send end interrupt). Slaves with the same address enter a waiting state (SCLAn=0,1) and an interrupt (INTIICAn: address matching interrupt) ^{note}.
- (5) The master changes the wait sequence to the 8th clock (WTIMn= 0,1).
- (6) The slave writes and sends data to the IICA shift register n (IICAn) to lift the slave's wait.
- (7) The master relieves the wait (WRELn=1) and begins data transfer from the slave.
- (8) The master enters a waiting state (SCLAn= 0,1) 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 = 0,1) 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 lifted 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.

Note 1 FIG 20-31(19) A series of operating steps for data communication via the I2C bus.

FIG 20-31 of the "(1) start condition ~ address ~ data" illustrates steps (1) ~ (7).

FIG 20-31 of the "(2) address ~ data ~ data" illustrates steps (3) ~ (12).

FIG 20-31 of the "(3) data ~ data ~ stop condition" illustrates steps (8) ~ (19).

2.n=0,1

(3) Data ~ data ~ stop condition



Note: 1 To lift the wait, the IICAn must be placed in the "FFH" position or the WRELn position must be placed.

2. After the release of the stop condition, the time from the SCLAn pin signal to generate the stop condition is at least 4.0 μ s when set to standard mode and at least 0.6 μ s when set to fast mode.
3. To undo the wait during the slave send, the data must be written to the IICAn instead of the WRELn position bit.
4. During the sending of the slave, if the wait is lifted by the assertion of the WRELn bit, the TRCn bit is cleared.

FIG 20-31 of "(3) data ~ data ~ stop condition" of (8) ~ (19) description is as follows:

- ⑧. The master enters a waiting state (SCLAn = 0,1) on the falling edge of the 8th clock and generates an interrupt (INTIICAn: Transmit End-of-Off). Because the ACKEn bit of the master is "0", the ACK is sent to the slave through the hardware.
- ⑨. The master reads the received data and dismisses the wait (WRELn=1).
- ⑩. The slave detects ACK (ACKDn=1) on the rising edge of the 9th clock.
- ⑪. The slave enters a waiting state on the falling edge of the 9th clock (SCLAn= 0,1) and generates an interrupt (INTIICAn: transmit end interrupt).
- ⑫. If the slave writes and transmits data to the IICA shift register n (IICAn), the slave's wait is lifted and the transfer of data from the slave to the master begins.
- ⑬. The master generates an interrupt (INTIICAn: transmit end interrupt) on the falling edge of the 8th clock and enters a waiting state (SCLAn=0,1). Because ACK control (ACKEn=1) occurs, the bus data line at this stage becomes low (SDAAn=0,1).
- ⑭. The master sets the NACK Acknowledge (ACKEn=0,1) and changes the wait sequence to the 9th clock (WTIMn=1). If the master relieves the wait (WRELn=1), the slave detects THEACK (ACKDn=0,1) on the rising edge of the 9th clock.
- ⑮. Both the master and slave enter a waiting state (SCLAn=0,1) on the falling edge of the 9th clock, and both produce an interrupt (INTIICAn: end-of-transmit interrupt).
- ⑯. If the master issues a stop condition (SPTn=1), the bus data cable (SDAAn=0,1) is cleared and the master's wait is lifted. After that, the master is on standby until the bus clock line is set in place (SCLAn=1).
- ⑰. The slave stops sending after confirming the NAK, in order to end the communication, the wait is lifted (WRELn=1). If the slave wait is lifted, the bus clock line is set in place (SCLAn=1).
- ⑱. If the master confirms that the bus clock line is set (SCLAn=1), the bus data line is set after the stop condition preparation time has elapsed
- ⑲. (SDAAn=1), and then issue a stop condition (SDAAn is changed from "0" to "1" by SCLAn=1). If a stop condition is generated, the slave detects the stop condition and generates an interrupt (INTIICAn: Stop Condition Interrupt).

Chapter 21 Serial interface SPI

The serial interface SPI function is a proprietary function of the BAT32A279.

21.1 The serial interface SPI functions

This product is equipped with two serial interfaces SPI0, SPI1, 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.

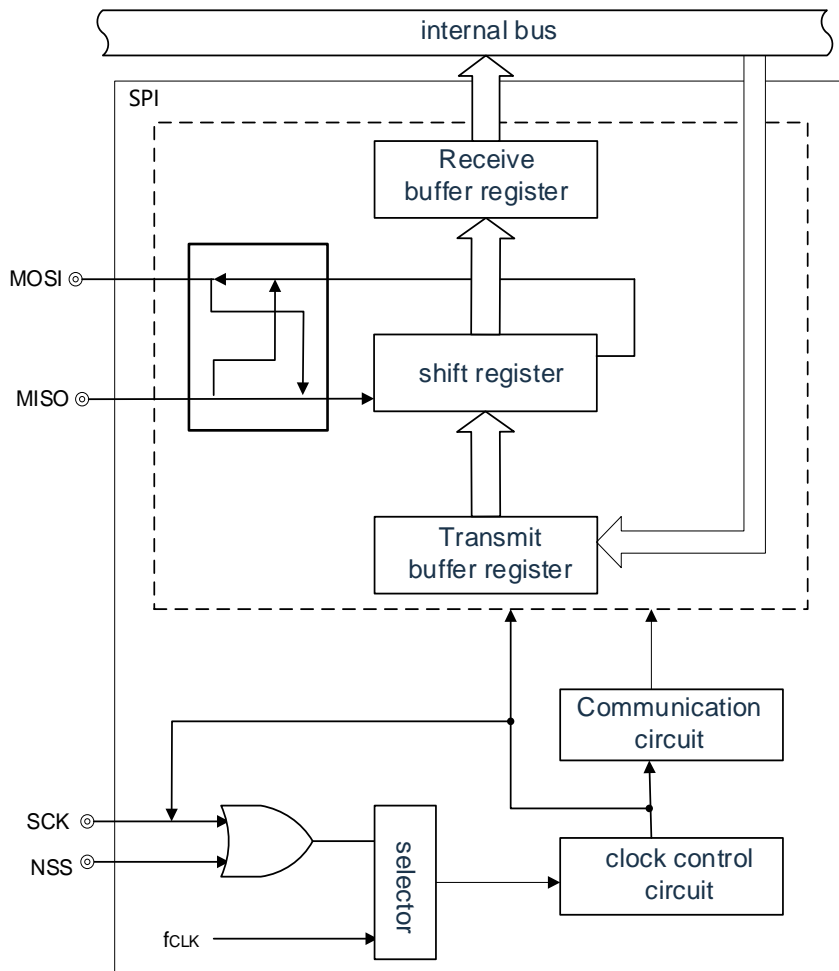
(2) 3-wire serial I/O mode

This mode transmits 8-bit or 16-bit data to multiple devices via three lines of serial clock (SCK n) and serial data bus (MISO n and MOSI n).

Note: $n=0,1$

21.2 Structure of the serial interface SPI

Figure 21-1 diagram of the serial interface SPI



21.3 Registers that control the serial interface SPI

The serial interface SPI is controlled by the following registers.

- Peripheral enable register 2 (PER2).
- Serial operating mode register (SPIMn).
- Serial clock selection register (n).
- Transmit buffer register (SDROn).
- Receive Buffer Register (SDRIn).
- Port Mode Register (PMxx).
- Port Mode Control Register (PMCxx).
- Port register (Pxx).

Note: n=0,1

21.3.1 Peripheral enable register 2 (PER2).

PER2 registers are registers that are set to allow or disable clocking to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use. To use the SPI function, SPInEN must be set to "1". For details, see "4.3.8 Perimeter Allow Registers 0, 1, 2, 3 (PER0, PER1, PER2, PER3)"

Note: n=0,1

21.3.2 SPI Operating Mode Register (SPIMn).

SPIM is used to select the mode of operation and control the allow or disallow of the operation.
 SPIM n can be set by 8-bit memory operation instructions.
 A reset signal is generated to clear the register to 00H.

Figure 21-2 the 2-mode control register (SPIMn).

Address: SPI0:0x40046C00 SPI1:0x40047000 After reset: 00H R/W ^{Note 1}

symbol	7	6	5	4	3	2	1	0
SPIMn	SPIEn	RMDT n	NSSEn	DIRn	INTMDn	DLSn	RECMDn	-

SPIEn	SPI runs allowed
0	Stop running.
1	Allowed to run.

TRMDn ^{Note3}	Transmit/Receive mode control
0	Receive mode
1	Send/Receive mode

NSSEn ^{Note4}	NSS pin uses selection
0	The NSS pin is not used
1	Use the NSS pin

DIRn	Data transfer order selection
0	Perform MSB-first input/output.
1	Perform LSB-first input/output.

INTMDn	Interrupt source selection
0	The end of transfer is interrupted
1	The send buffer is empty interrupt

DLSn	The setting of the data length
0	8 bits of data length
1	16-bit data length

RECMDn	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.

2. The MO or SO output is fixed low when the TRMD is 0.

3. Before setting the position to 1, fix the NSS pin input level to 0 or 1.

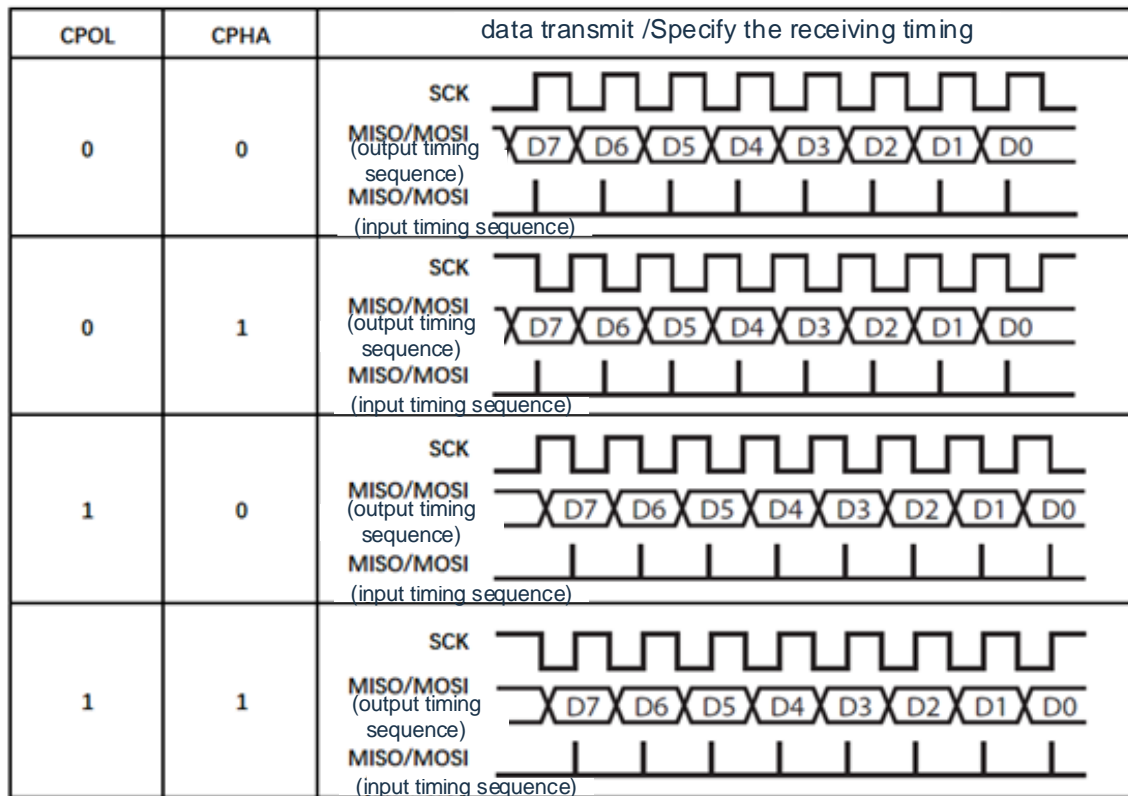
4.n=1 (channel 0 does not support the chip selection function, the bit must be set to 0).

21.3.3 SPI clock selection register (n).

This register specifies the timing of data send/receive and sets the serial clock.
can be set by 8-bit storage operation instructions.
A reset signal is generated to clear the register to 01H.

Figure 21-3 Format of the clock selection register (n).

Address: SPI0: 0x40046C04 SPI1: 0x40047004 After reset: 01H	R/W							
symbol	7	6	5	4	3	2	1	0
SPICn	0	0	0	CPOLn	CPHAn	CKS2n	CKS1n	CKS0n



CKS2n	CKS1n	CKS0n	SPI serial clock selection	mode
0	0	0	fCLK	Host mode
0	0	1	fCLK/2	
0	1	0	fCLK/2 ²	
0	1	1	fCLK/2 ³	
1	0	0	fCLK/2 ⁴	
1	0	1	fCLK/2 ⁵	
1	1	0	fCLK/2 ⁶	
1	1	1	An external clock input from SCK	Slave mode

Note 1. Write TOPICn is prohibited when SPIE n=1 (operation enabled).

2. The phase type of the data clock after reset is Type 1.

Note: n=0,1

21.3.4 SPI status register (SPISn).

The SPIS register is used to confirm the communication status of the SPI.

SPISn can be read by 8-bit storage operation instructions.

A reset signal is generated to clear the register to 00H.

Figure 21-4 status register (SPISn).

Address: SPI0: 0x40046C10 SPI1:0x40047010 After reset: 00H R							
symbol	7	6	5	4	3	2	1 0
LISTn	-	-	-	-	-	-	SDRIFn SPTFn

SDRIFn	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 SDRI is read, the bit is cleared to 0

SPTFnNote1	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.

2.n=0.1

21.3.5 Transmit buffer register (SDR0n).

The register is set to send data.

When setting bits 7 (SPIE n) and bit 6 (TRMDn) of the serial operating mode register (SPIMn) to 1 When sending/receiving starts by writing data to SDR0n.

Serial I/O shift registers convert data in SDR0n from parallel to serial data and output to the serial output pin.

SDR0n 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 21-5 Format of the transmit buffer register (SDR0n).

Address: SPI0:0x40046C08 SPI1:0x40047008 After Reset:															
														0000H R/W	
symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1 0
ROSS n	ROSS n														

21.3.6 Receive buffer register (SDR1n).

This register stores the received data.

If bit 6 (TRMDn) of the serial operating mode register (SPIMn) is set to 0, the reception begins by reading data from the SDRI.

During reception, data is read from the serial input pin into SDR1n.

SDRI n can be read with 8-bit or 16-bit memory manipulation instructions.

A reset signal is generated to clear the register to 0000H.

Figure 21-6 Format of the receive buffer register (SDR1n).

Address: SPI0: 0x40046C0C SPI1: 0x4004700C After reset:															
														0000H R	
symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1 0
SDR1n	SDR1n														

Note: n=0,1

21.3.7 The SPI pin functions the control register

When using SPI, the control registers (PMxx, P MCxx) must be set for port functions that are multiplexed with the SPI input and output pins. For details, please refer to "2.3.1 Port Mode Register (PMxx)".

When using the multiplex port of the SPI pin as the output of S CK/SO/MO, the position of the port mode register (PMxx, PMCxx) corresponding to each port must be "0". When using the multiplexing port of the SPI pin as an input to SCK/SI/MI, the bits "1", P MCxx of the port mode register (PMxx) corresponding to each port must be used Position "0". At this point, the bit of the port register (Pxx) can be "0" or "1". For details, please refer to "2.5 Register Settings When Using the Multiplexing Function".

21.4 Serial interface for operation of the 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 n) of SPISn is set to 1. When the communication of data is complete, set the communication completion interrupt request flag (SPIIFn) and clear SPTFn to 0. Then enable the next communication.

Precautions

1. When SPTFn=1 (during serial communication), access to control registers and data registers is prohibited.

2. Must be used within the range that satisfies the SCLK Cycle Time (tKCY) characteristics. Please refer to the data sheet for details.

Note: n=0,1

21.4.1 The sending and receiving of the master

If the serial operating mode register (SPIMn) has bit 6 (TRMDn) of 1, data can be sent or received. When a value is written to the transmit buffer register (SDR0n), send/receive starts.

(1) Procedure

Figure 21-7 Initial setup steps for master send/receive

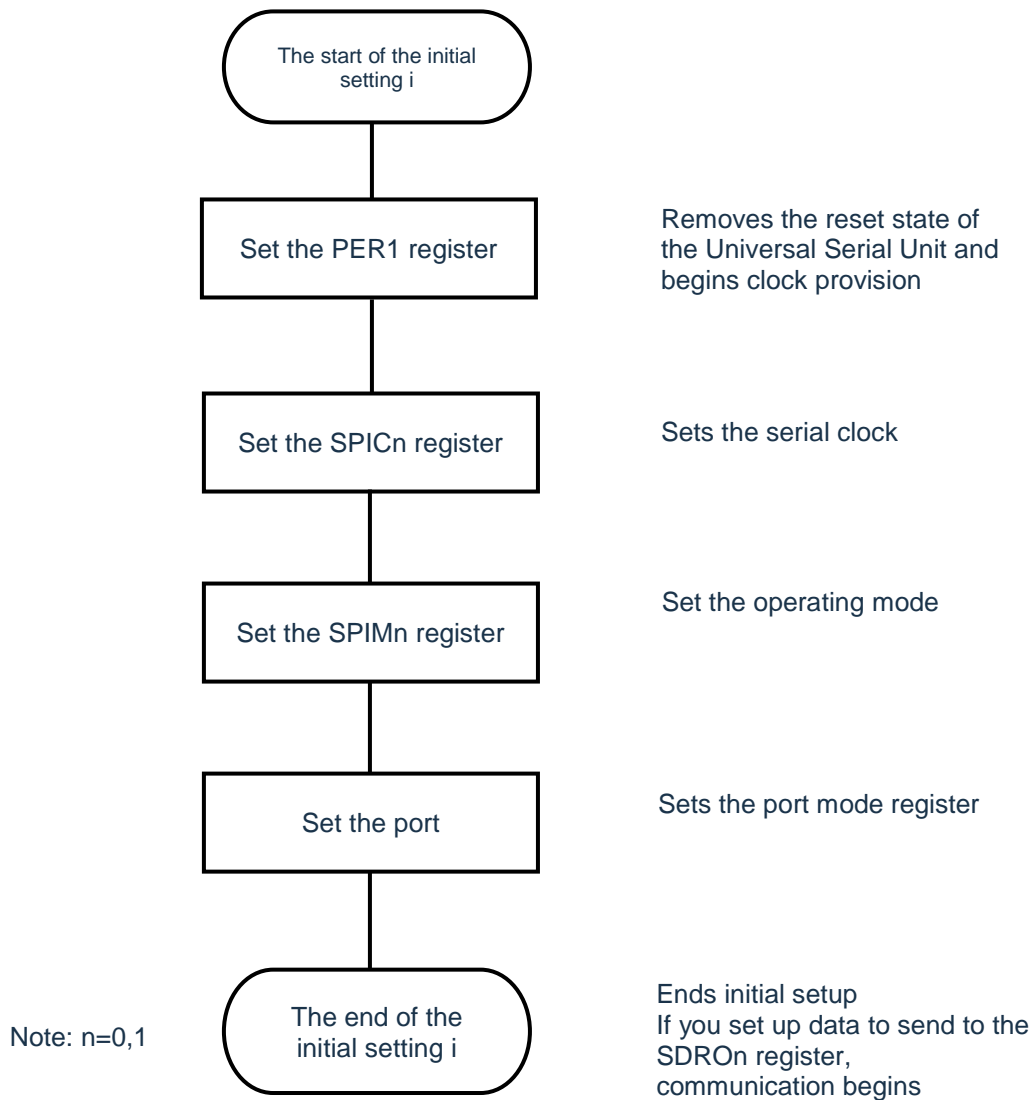
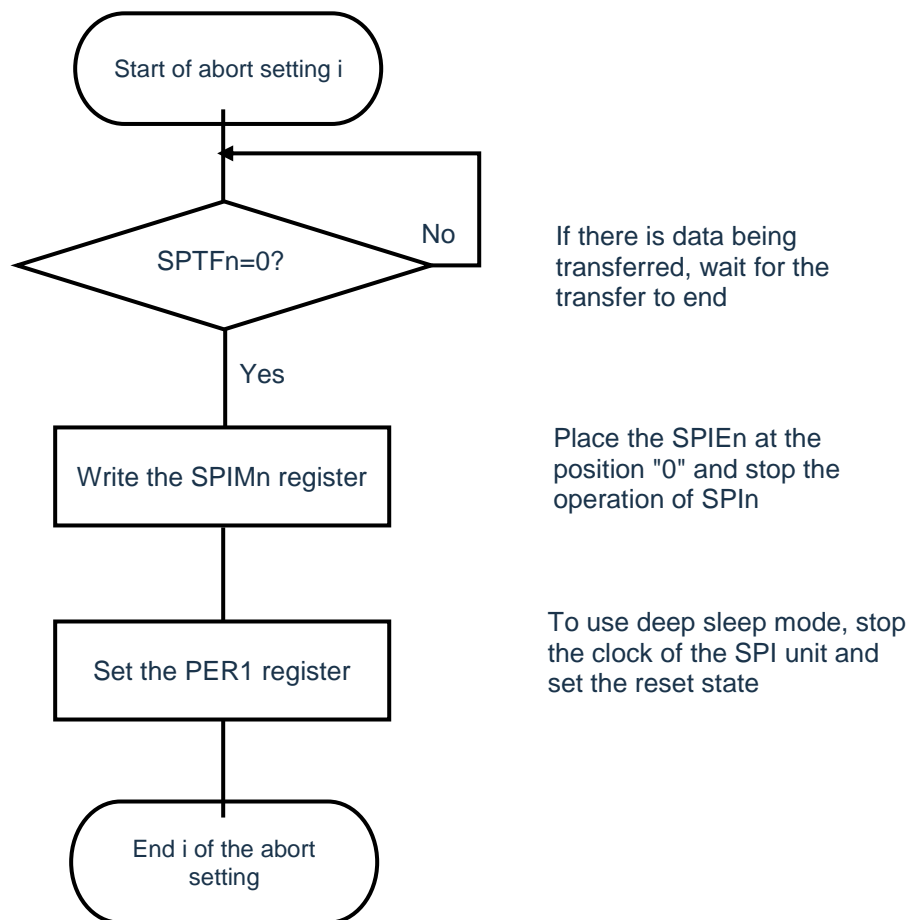


Figure 21-8 Abort step of the master transmit/receive



Note: n=0,1

(2) Processing process

Fig. 21-9 Timing diagram of transmit/receive timing (single transmit mode) (INTMD=0, CPHA=1, CPOL=1).

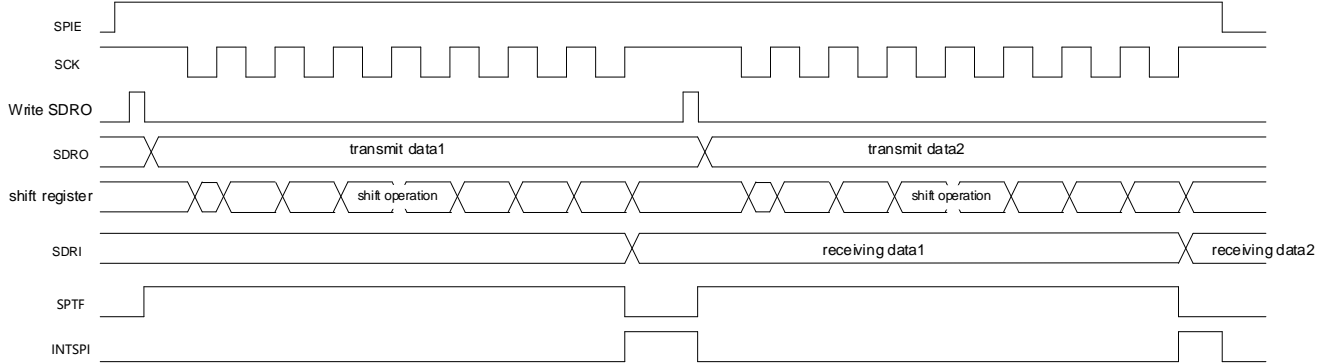
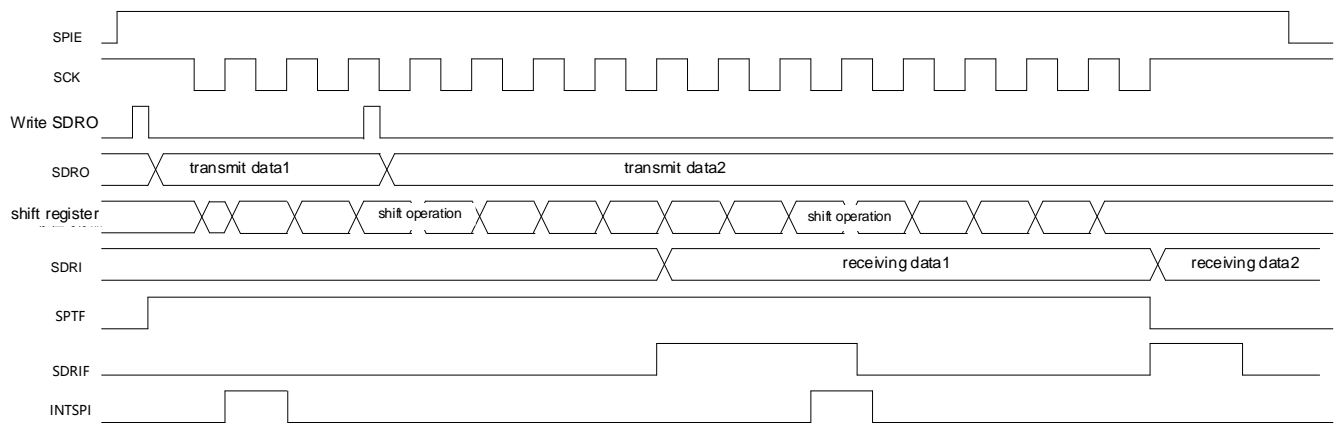


Fig. 21-10 transmit/receive timing (continuous transmit mode) (INTMD=1, CPHA=1, CPOL=1)

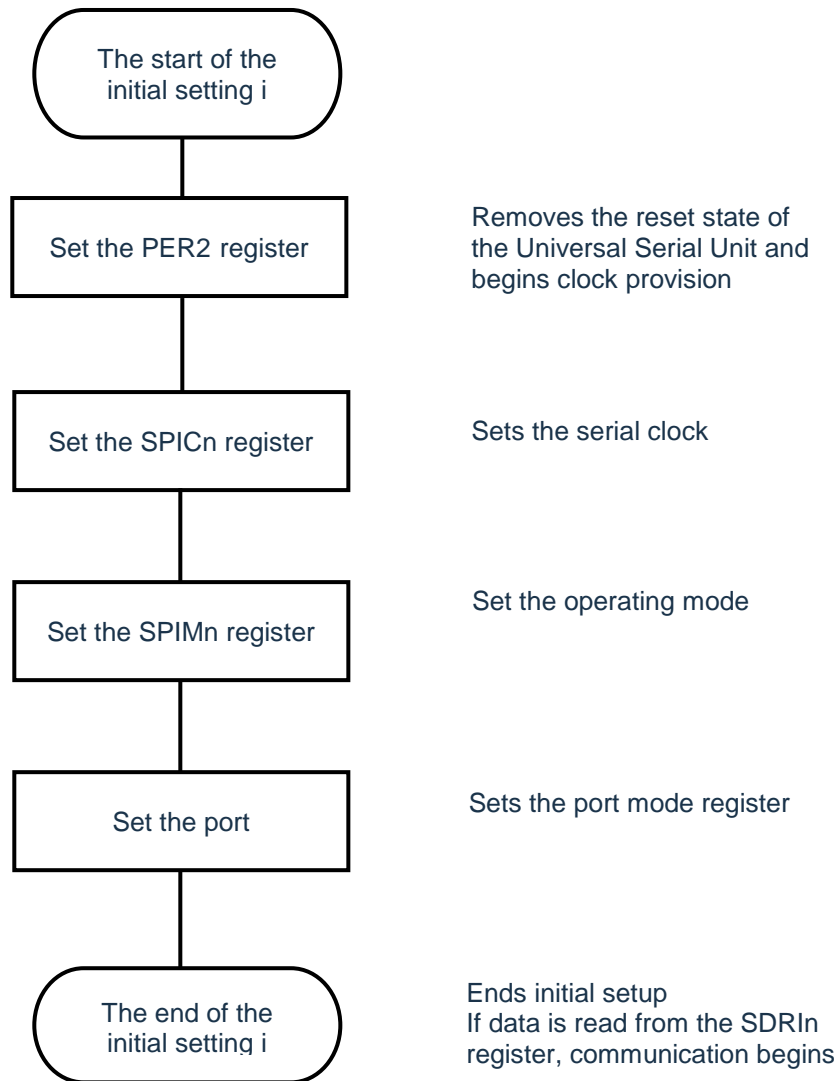


21.4.2 The reception of the master

If bit 6 (TRMDn) of the serial operating mode register (SPIMn) is 0, only data can be received. When data is read from the receive buffer register (SDRIn), the reception begins.

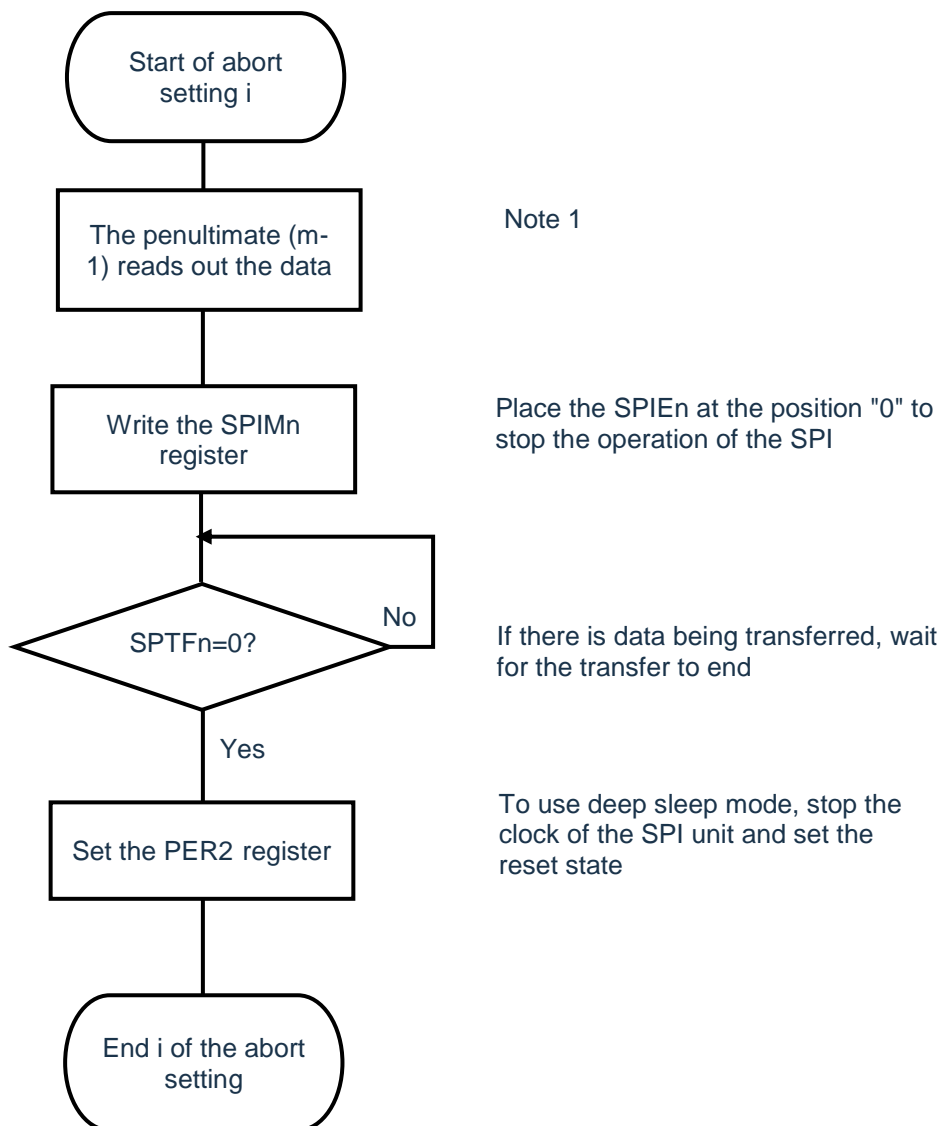
(1) Procedure

Figure 21-11 Initial setup steps for master reception



Note: n=0,1

Figure 21-12 The abort step of the master receive

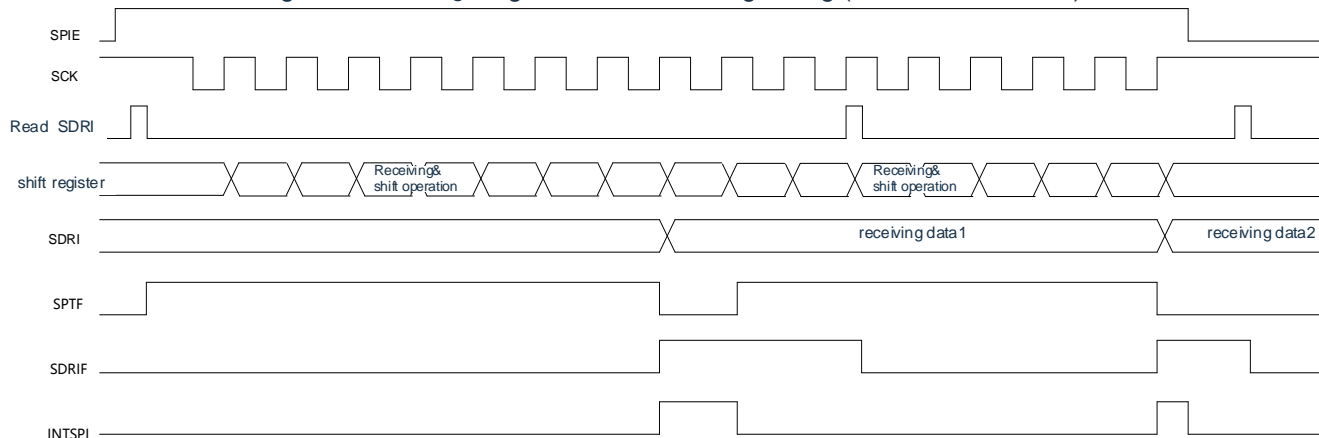


Note 1: In receive-only mode, the SPI transmission is triggered by reading the value of the SDRIn register. If the SPI is not aborted in time, there may be a redundant transmission after the last read of SDRIn. If you want to avoid the last redundant transmission, you can turn off SPIE N after waiting for an SCK cycle after the penultimate data readout. The transfer of the SPI will be aborted after the last data transfer is complete.

Note: n=0,1

(2) Processing process

Fig. 21-13: Timing diagram of the receiving timing (CPHA=1, CPOL=1).

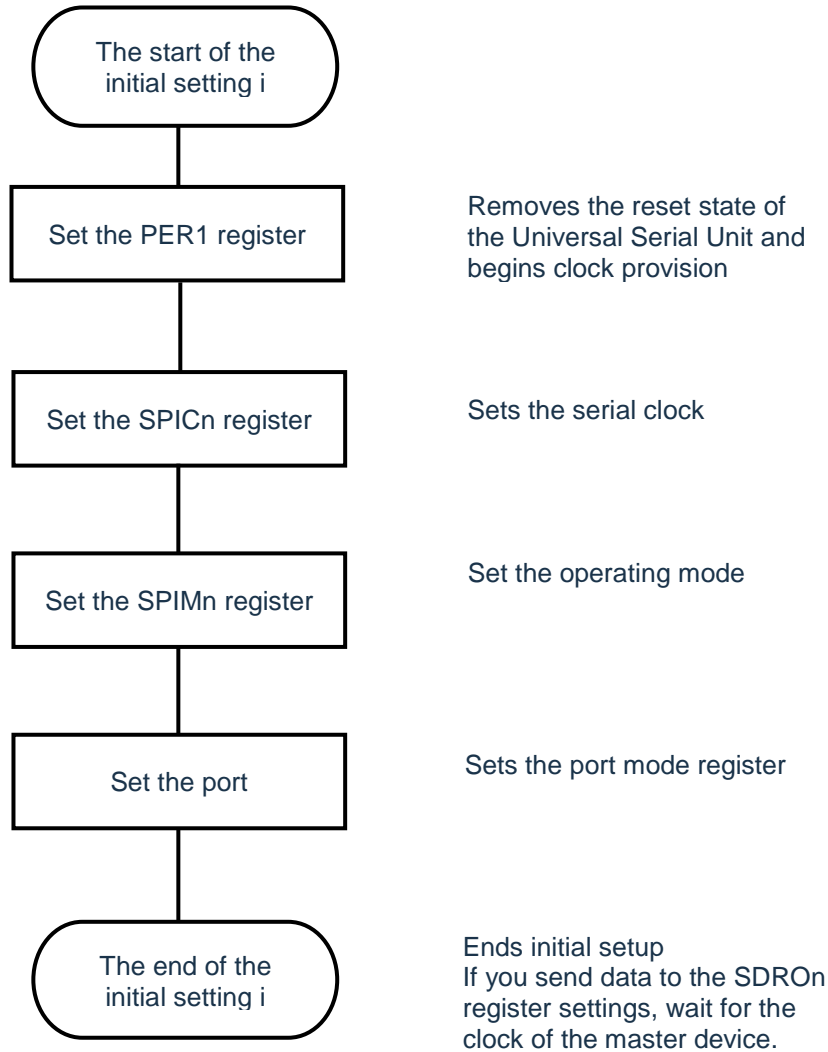


21.4.3 Slave sending and receiving

If bits CKS2-0 of the serial clock selection register (n) select slave mode, bit 6 (TRMDn) of the serial operation mode register (SPIMn) is 1, you enter slave send/receive mode. When a value is written to the transmit buffer register (SDRn), wait for the clock of the master device to start sending/receiving.

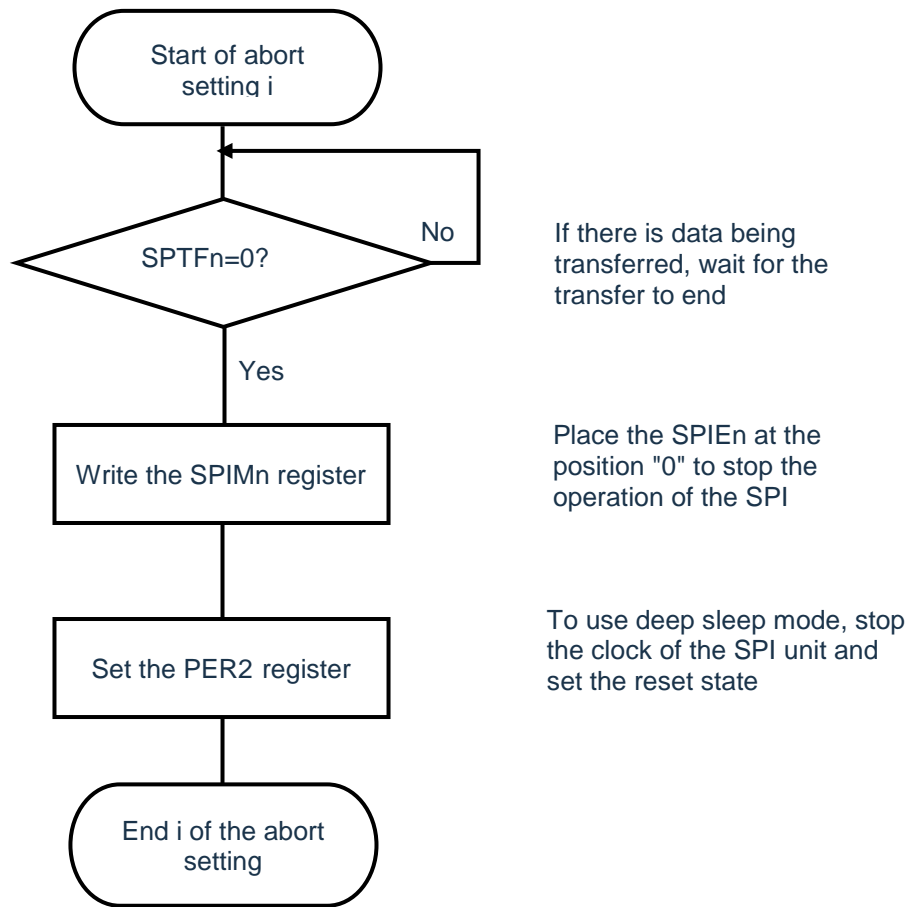
(1) Procedure

Figure 21-14 Initial setup steps for slave send/receive



Note: n=0,1

Figure 21-15 Abort step of Slave send/receive



Note: n=0,1

(2) Processing process

Figure 21-16 Timing diagram of send/receive timing (single-send mode) (INTMD= 0, CPHA= 1, CPOL = 1).

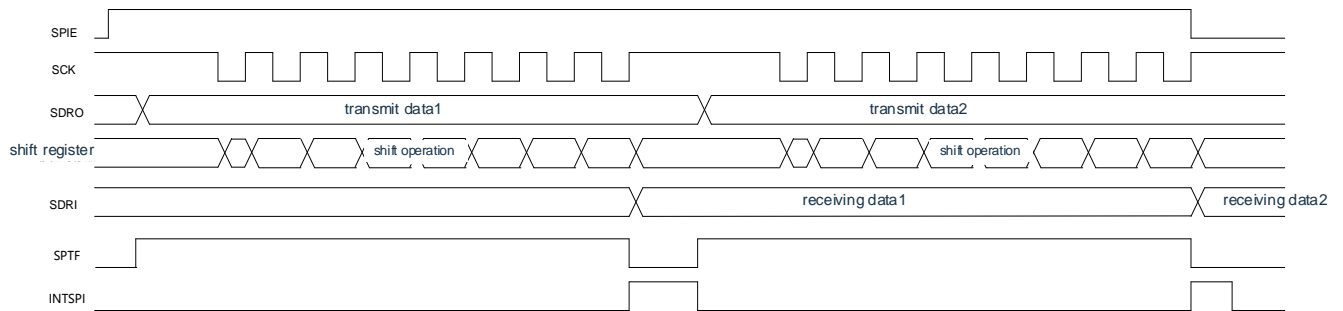
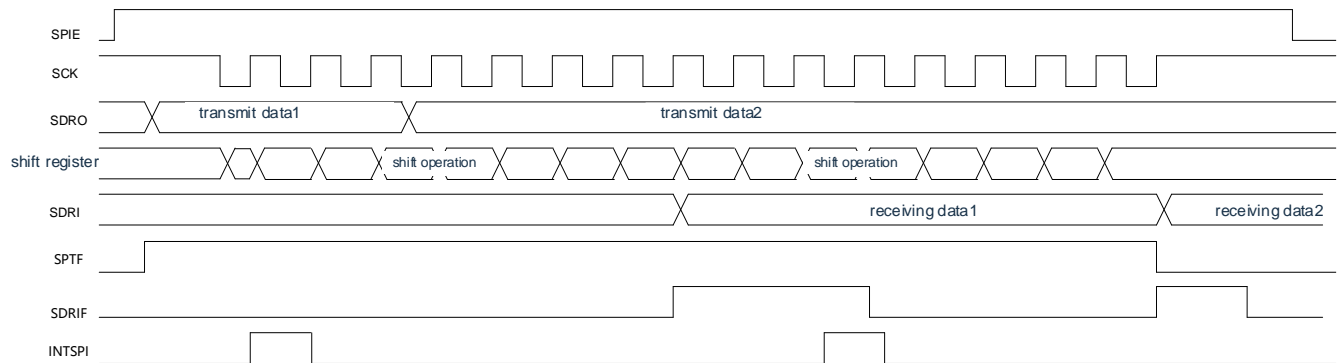


Fig. 21-17 Timing diagram of transmit/receive timing (continuous transmission mode) (INTMD=1, CPHA=1, CPOL=1).

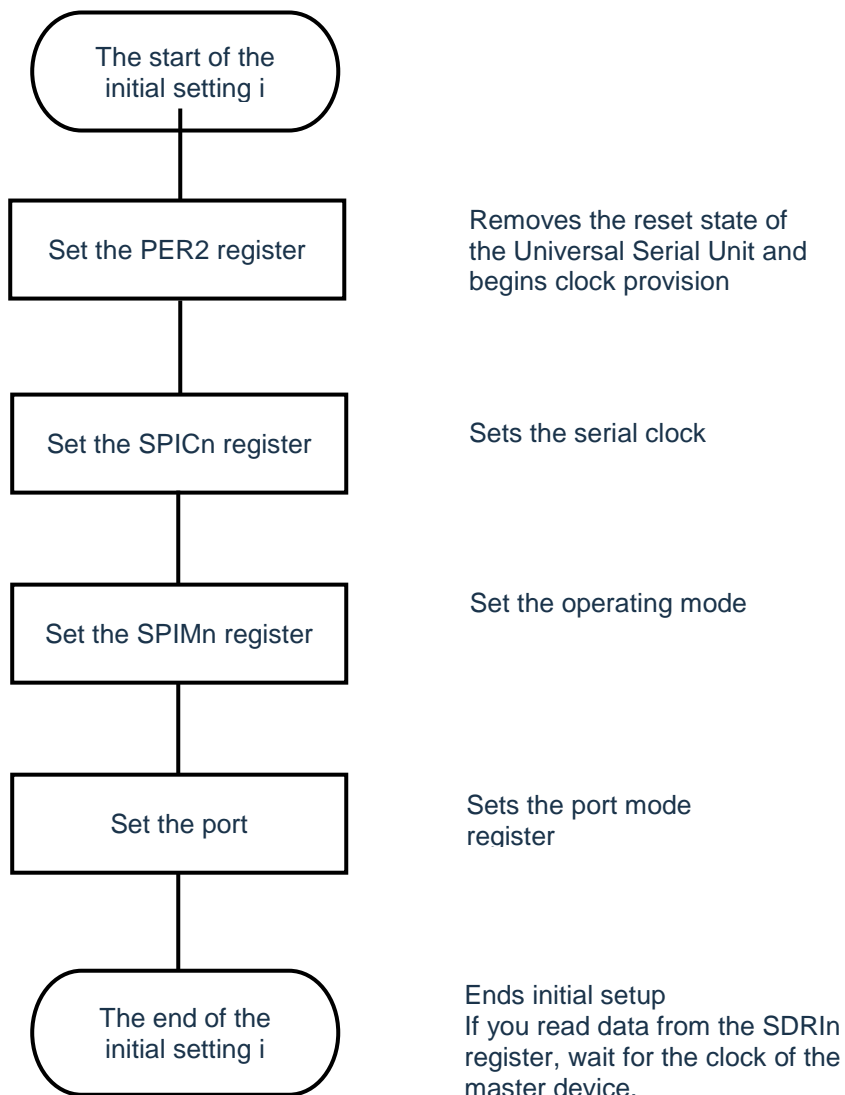


21.4.4 Slave reception

If bits C KS 2-0n of the serial clock selection register (n) select slave mode and bit 6 (TRMDn) of the serial operation mode register (SPIM n) is 0, slave receive mode is entered. When reading data from the receive buffer register (SDRIn), wait for the clock of the master device to start receiving.

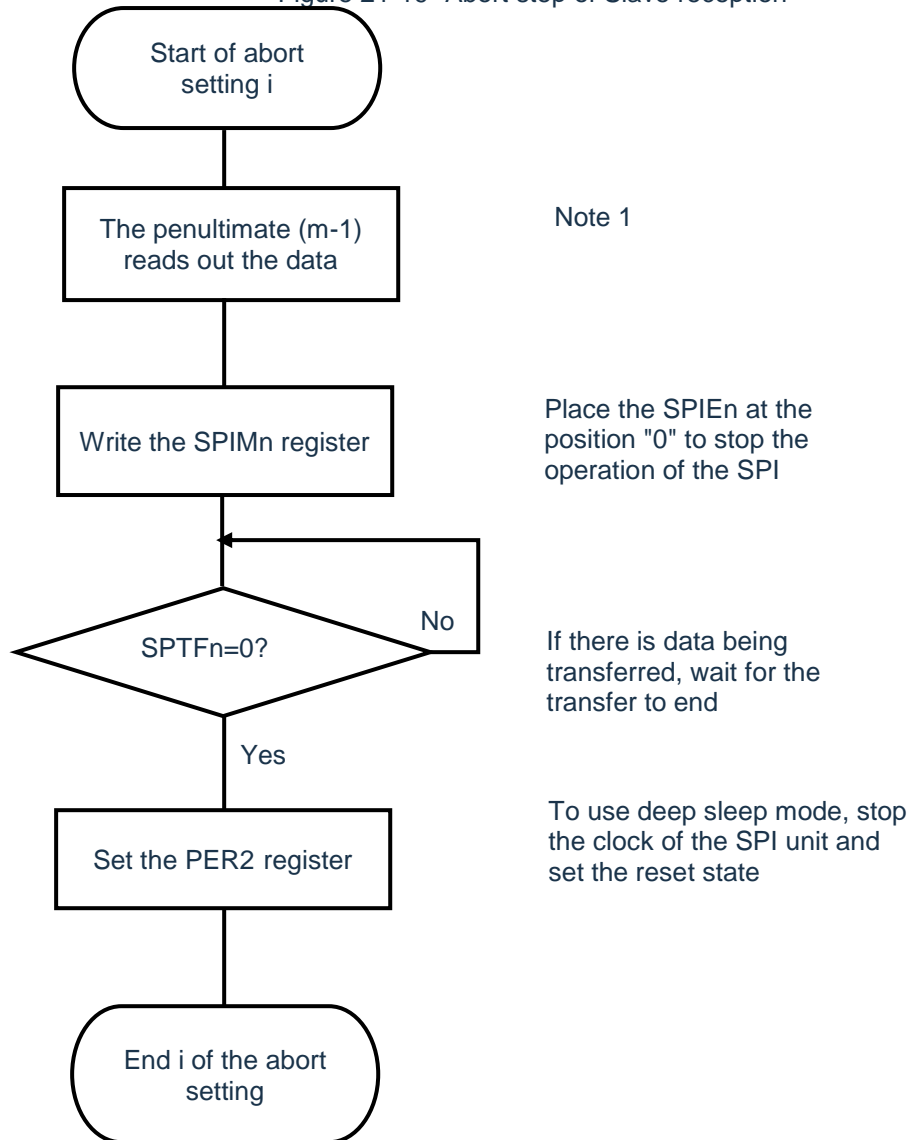
(1) Procedure

Figure 21-18 Initial setup steps for slave reception



Note: n=0,1

Figure 21-19 Abort step of Slave reception



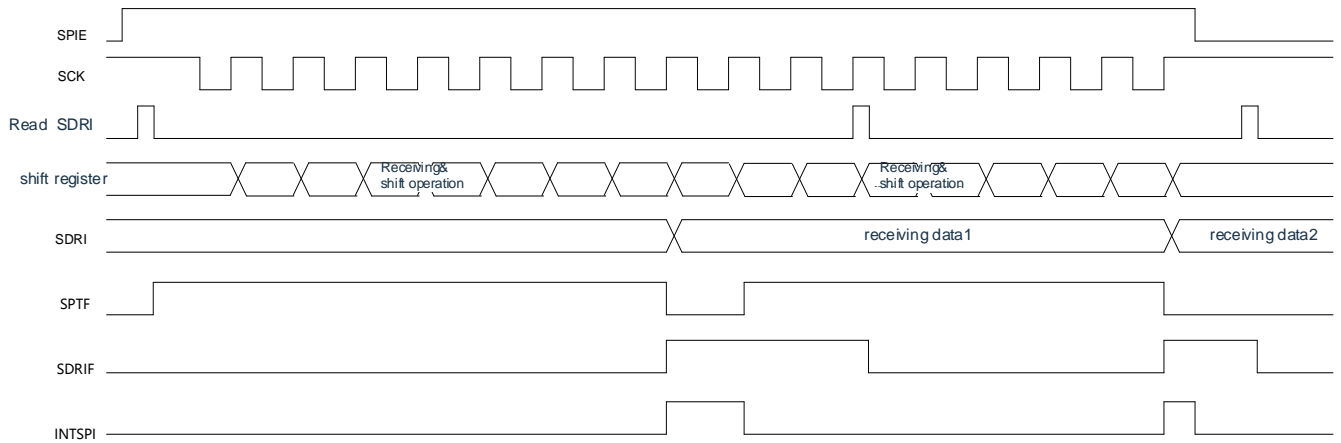
Note 1: In receive-only mode, the SPI transmission is triggered by reading the value of the SDRIn register. If the SPI is not aborted in time, there may be a redundant transmission after the last read of SDRIn.

If you want to avoid the last redundant transmission, you can turn off SPIE N after waiting for an SCK cycle after the penultimate data readout. The transfer of the SPI will be aborted after the

Note: n=0,1

(2) Processing process

Figure 21-20 diagram of the receiving timing (CPHA=1, CPOL=1).



Chapter 22 CAN control

The BAT32A239 is equipped with two CAN controllers, and the BAT32A279 product is equipped with three CAN controllers.

22.1 Summary description

The chip has on-chip CAN controller (Controller Area Network) capability and complies with the CAN protocol of the ISO 11898 standard

22.1.1 features

- Complies with ISO 11898 and is tested according to ISO/DIS 16845 (CAN compliance).
- Standard and extended frames are used for receiving and sending
- Communication speed: Maximum 1Mbps. (CAN input clock greater than or equal to 8MHz).
- 1 channel has 16 message caches
- Receive/Send History List function
- Automatic block transfer function
- Multi-cache receive block functionality
- Masking settings for four modes per channel

22.1.2 Feature overview

Table22-1 lists the functions of the CAN controller.

Table22-1: Function Overview

function	detail
agreement	ISO 11898 CAN Protocol (Standard and Extended Frame Send/Receive).
baud rate	Maximum: 1Mbps (CAN input clock 8MHz).
Data Storage	The information is stored in CAN RAM
The number of messages	<ul style="list-style-type: none"> - 16 message caches per channel - Each packet cache can be set up as either a receiving packet cache or a transmit packet cache
Message reception	<ul style="list-style-type: none"> - Unique ID can be set to each message cache - There are four Mask settings per channel - Each time a message is received and stored in the message cache, a receive interrupt is generated. - More than two receive packet caches can be used as receive FIFOs (Multi-cache receive block function) - Receive history list function
Message sending	<ul style="list-style-type: none"> - Unique ID can be set to each packet cache - Each cache can be associated with a send-complete interrupt - Packet buffers 0 through 7 are designated as transmit packet buffers and can be used for automatic block transfers. The message transmission interval is programmable (Automatic Block Transfer Function (hereinafter referred to as "ABT")). - Transfer history list function
Remote frame processing	Remote frame processing is cached through the transmission message
Timestamp function	<ul style="list-style-type: none"> - When using a 16-bit timer, you can set the timestamp function for message reception Timestamp capture triggering is optional (SOF or EOF can detect packet frames)
Diagnostic functions	<ul style="list-style-type: none"> - Readable error counter - A Valid Protocol Operation Flag used to verify bus connections - Accept mode only - Single-shot mode - Decoding of the CAN protocol error type - Self-test mode
Released from the bus shutdown state	<ul style="list-style-type: none"> - Software can set force release from bus shutdown (ignoring timing constraints) - Does not automatically release from the bus shutdown state (software settings must be re-enabled).
Power-saving mode	<ul style="list-style-type: none"> - CAN sleep mode (can be woken up by the CAN bus). - CAN stop mode (cannot be woken up by the CAN bus).

22.1.3 Configuration

The CAN controller consists of the following four modules

(1) interface

This module provides an internal bus interface to send and receive signals between the CAN module and the host

(2) Memory Control Module (MCM).

This function block controls access to the CAN protocol layer and CANRAM in the CAN module.

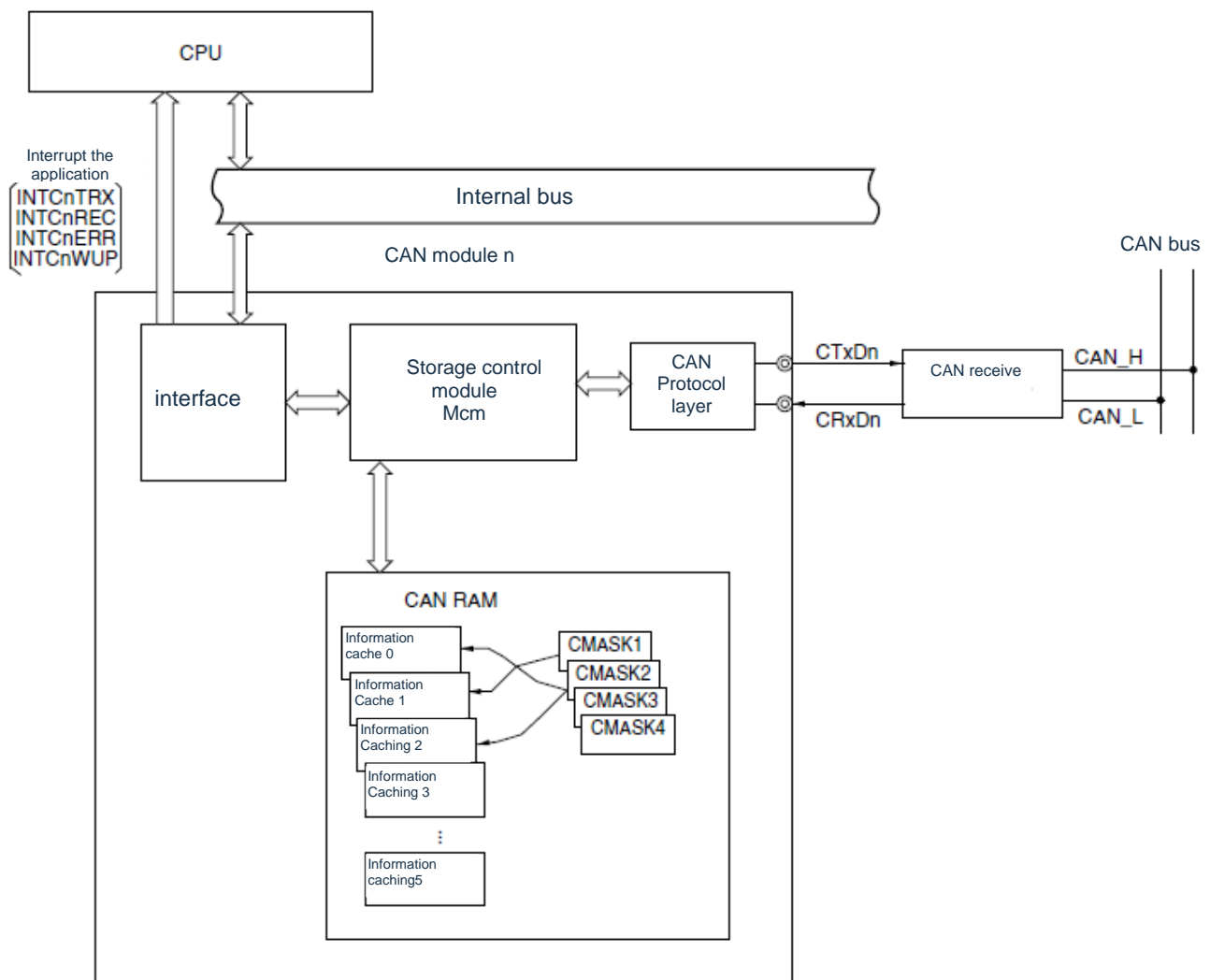
(3) CAN protocol layer

This function block involves the operation of the CAN protocol and its associated settings.

(4) CAN RAM

This is the CAN memory function block, which is used to store message ID, message data, etc.

Figure 22-1 diagram of 1 CAN module n



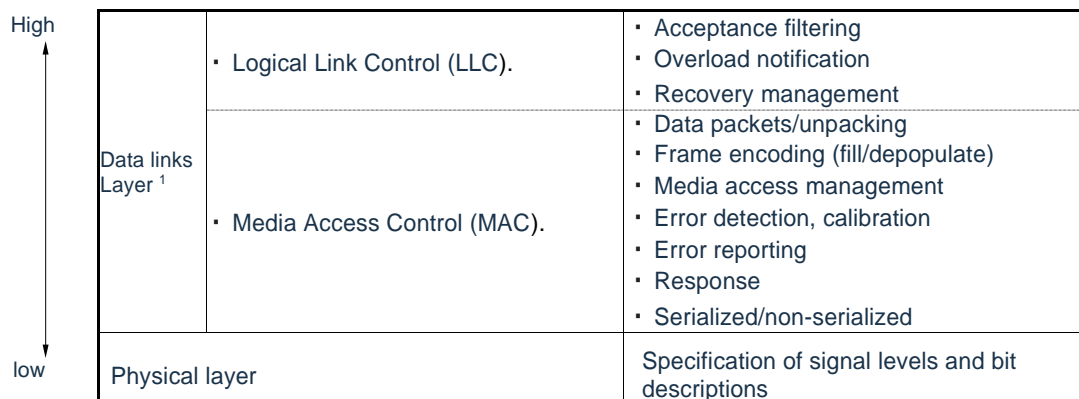
BAT32A239: n=0/1, BAT32A279: n=0/1/2

22.2 CAN protocol

CAN (Controller Area Network) is a high-speed multiplexed communication protocol for real-time communication in automotive applications (Class C). CAN is regulated by ISO 11898. For more information, see the ISO 11898 specification.

The CAN specification is usually divided into two layers: the physical layer and the data link layer, which in turn contains logical links and media access control. The composition of these layers is shown in the following figure:

Figure 22-2. The composition of layers



1: CAN controller specification

22.2.1 Frame format

(1) Standard frame format

- The standard frame format uses 11-bit identifiers, which means it can handle 2048 signals.

(2) Extend the format frame

- Extended format frames use 29-bit (11-bit + 18-bit) identifiers, which can handle 2048x218 signals.
- Sets the extended format frame when the "implicit level" is set for the SRR and IDE bits in the arbitration field (CMOS level equals "1").

22.2.2 Frame type

The four frame types in the following table are used in the CAN protocol.

Table22-2. Frame Type

The frame type	description
Data frames	The frame used to transmit the data
Remote frames	Go to the signal that requests the data frame
Error frame	A signal used to report an error
Overload frames	Lets you set a delay to wait for the next data frame or remote frame

(3) Bus value

Bus values are divided into explicit and implicit

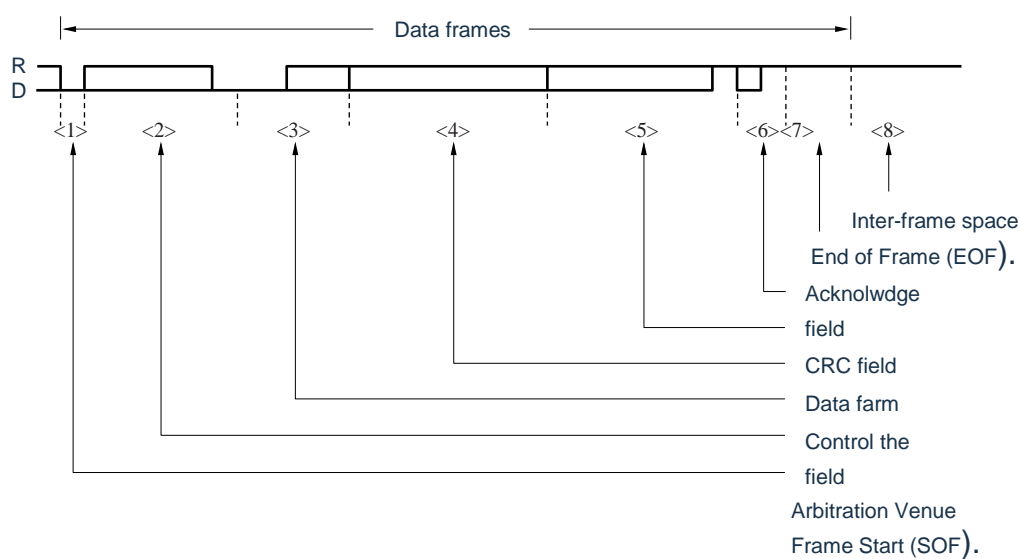
- The dominant value is logically 0
- The implicit value is logical 1
- When a bus is both dominant and implicit, the state of the bus is displayed as dominant

22.2.3 Data frames and remote frames

(4) Data frames

A data frame consists of 7 different bit fields

Figure 22-3. Data frames



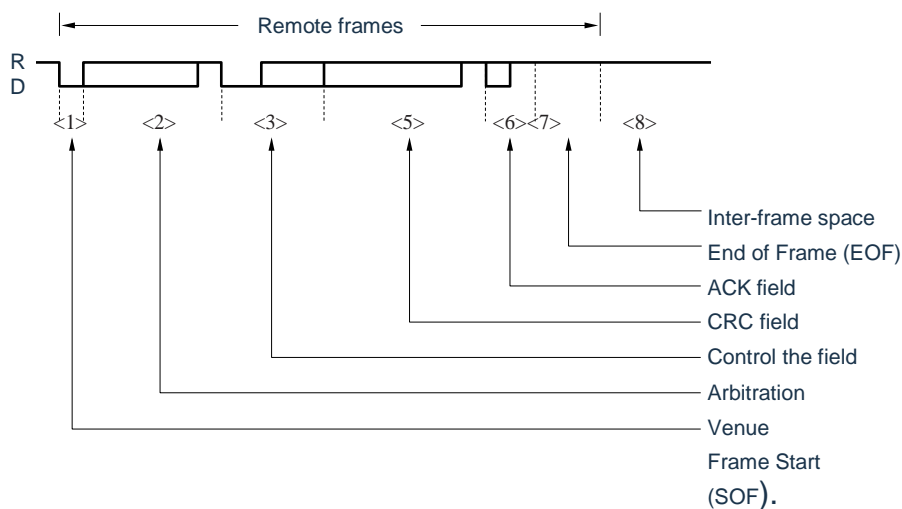
Note: D: Dominant =0

R: Recessive = 1

(5) Remote frames

The remote frame consists of 6 bits of field master

Figure 22-4. Remote frames



Note 1 Even if the data length of the control farm is not "0000B", the data farm will not send

2. D: Dominant = 0

R: Recessive

= 1

(6) Bit field description

< 1> Frame Start (SOF).

The frame start field is at the beginning of a data frame or remote frame

Figure 22-5. Frame Start (SOF).



Note D: Dominant = 0

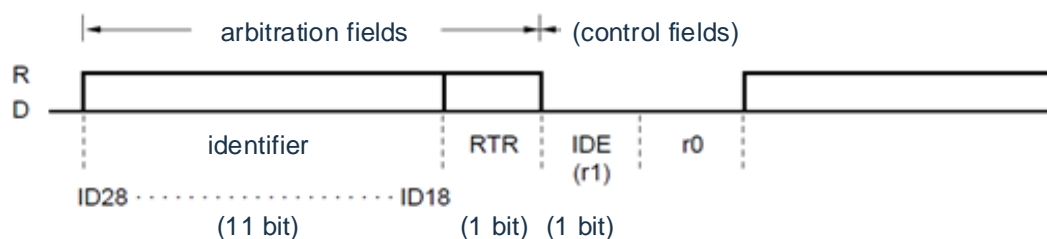
R: Recessive = 1

- If a dominant bit is detected in the bus idle state, a hard synchronization is performed (the current TQ is designated as the SYNC segment)
- If a dominant bit is sampled at a sample point after such a hard synchronization, the bit is assigned as SOF. If a recessive bit is detected, the protocol layer returns to the bus idle state and treats the preceding dominant pulse as interference-only. In this scenario, no error frame is generated.

< 2 > arbitration venue

The arbitral field is used to set priorities, data frames/remote frames, and frame formats

Figure 22-6. Arbitration field (in standard format).



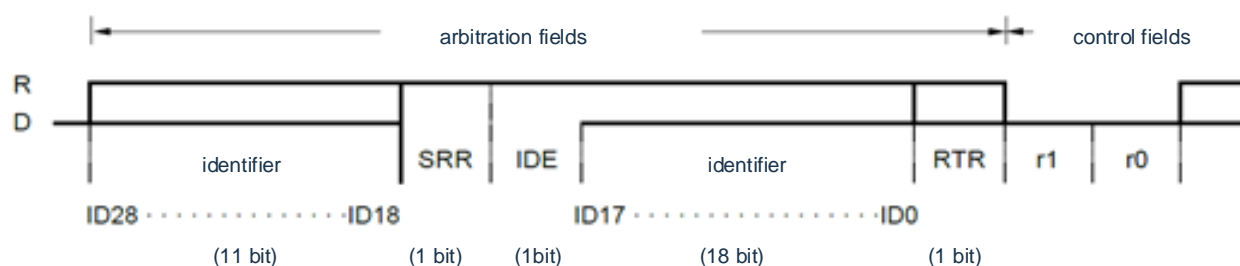
Note 1. ID18 through ID28 are identifiers.

2. The identifier is transmitted by the MSB first.

Note D: Dominant = 0

R: Recessive = 1

Figure 22-7. Arbitration field (in extended frame mode).



Note 1. ID18 through ID28 are identifiers.

2. The identifier is sent first for THE MSB.

Note D: Dominant = 0

R: Recessive = 1

Table22-3. RTR Frame configuration

The frame type	RTR bits
Data frames	0(D)
Remote frames	1(R)

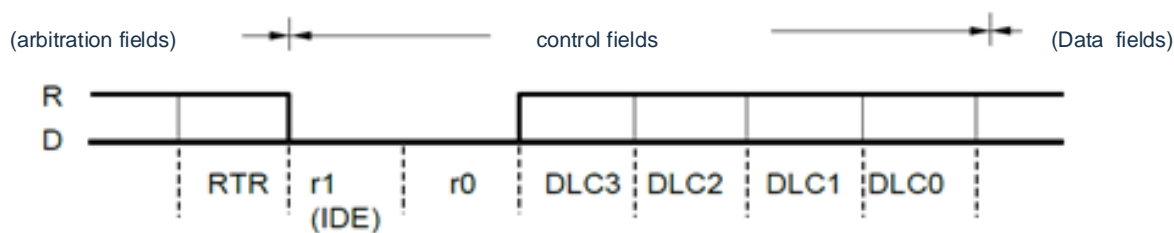
Table 22-4. Frame formatting (IDE bit) and identifier (ID) bit settings

Frame format	SRR bit	IDE bit	Number of digits
Standard frame format	None	0(D)	11 bits
Extend the frame format	1(R)	1(R)	29 bits

< 3 > control field

The control field is set "DLC" as the number of bytes of data in the data field (DLC=0 to 8).

Figure 22-8 Control the field



Note D: Dominant = 0

R: Recessive = 1

In standard format frames, the IDE bit of the control field is the same as the r1 bit

Table22-5. Data Length Configuration

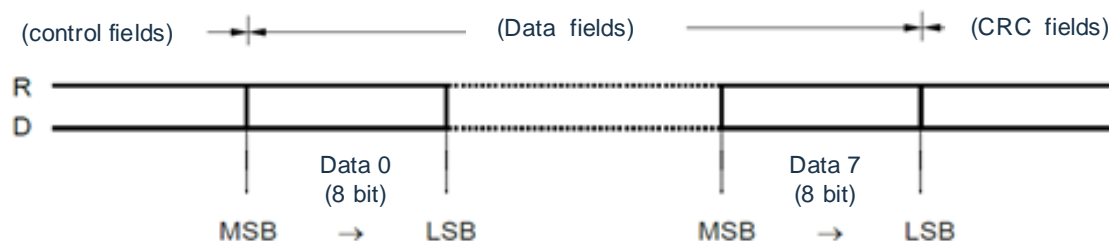
Data length code				Data byte calculation
DLC3	DLC2	DLC1	DLC0	
0	0	0	0	0 bytes
0	0	0	1	1 byte
0	0	1	0	2 bytes
0	0	1	1	3 bytes
0	1	0	0	4 bytes
0	1	0	1	5 bytes
0	1	1	0	6 bytes
0	1	1	1	7 bytes
1	0	0	0	8 bytes
Other values				8 bytes (regardless of the value of DLC0 to

Note: In a remote frame, there is no data farm even if the data length code is not (0000B).

< 4> data farm

The data farm contains the amount of data, in bytes, that controls the field settings. Up to 8 data units can be set.

Figure 22-9. Data farm



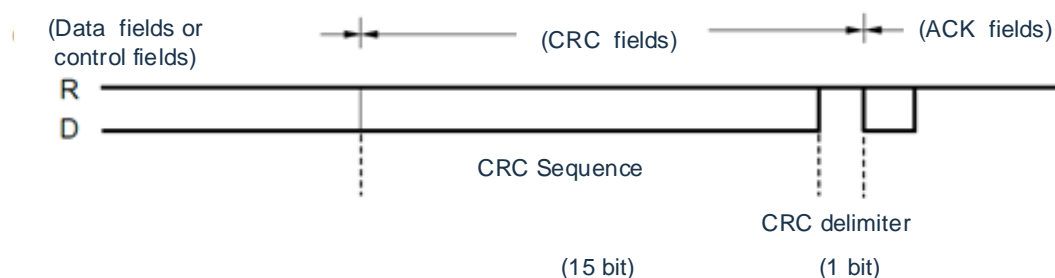
Note D: Dominant = 0

R: Recessive = 1

< 5> CRC field

A CRC field is a 16-bit field that detects errors in data transmission.

Figure 22-10. CRC field



Note D: Dominant = 0

R: Recessive = 1

- The polynomial $P(X)$ used to generate a 15-bit CRC sequence is represented as follows:
- $P(X) = X^{15} + X^{14} + X^{10} + X^8 + X^7 + X^4 + X^3 + 1$
- Send Node: Sends a sequence of CRC calculations for data from the start frame, arbitration field, control field, and data farm (before bit population).
- Receive Node: Compares the computed CRC sequence with the data bit in the CRC field using the data bits that exclude the padding bits in the received data. If the two CRC sequences do not match, the node issues an error frame.

The Ack field is mainly a response to normal reception

Figure 22-11. Ack field



- If no CRC error is detected, the receiving node sets the Ack to the dominant bit
- The sending node outputs two recessive bits

The end-of-frame field indicates the end of a data frame or a remote frame.

Figure 22-12. End of Frame (EOF).

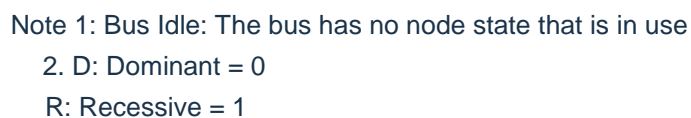


Inter-frame space is used to distinguish between two frames by inserting between data frames, remote frames, error frames, or overload frames.

- The bus state differs based on the state of the error

The interframe space contains a 3-bit gap field and a bus idle field.

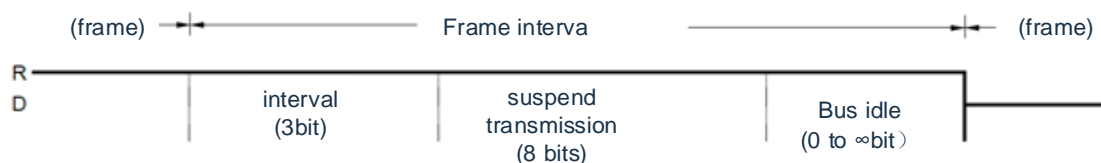
Figure 22-13: Inter-frame space (active error node).



(b) Passive error node

Interframe space contains a gap field, a suspend field, and a bus idle field.

Figure 22-14: Inter-frame space (passive error node).



Note 1: Bus idle: The bus is not in use by any nodes.

Pending Transfer: In a passive error condition, 8 hidden bits are sent on the node

2.D: Dominant = 0

R: Recessive = 1

In general, the gap field is 3 bits, if the sending node detects an explicit bit in the third bit of the gap field, the send operation will be performed anyway.

- Error status operation

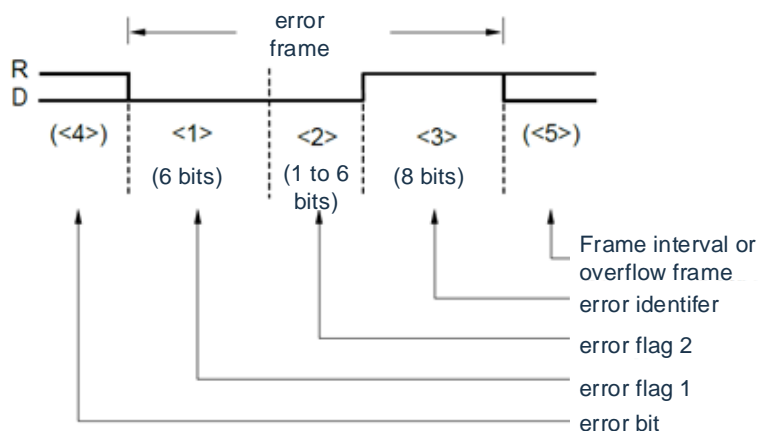
Table22-6: Operation in Error state

Error status	operate
Proactive error	3 bits gap can be transmitted immediately
Passive error	8 bits can be sent after the gap

22.2.4 Error frame

If an error is detected, an error frame is sent at the next node.

Figure 22-15: error frame



Note D: Dominant = 0

R: Recessive = 1

Table22-7: Error Frame Definition

No.	name	Number of digits	definition
<1>	Error flag 1	6	Active error node: Continuous output of 6 dominant bits Passive Error Node: Continuous output of 6 recessive bits When one node is outputting a passive error flag and another node outputs an explicit bit, the passive error flag is not cleared until six consecutive bits of the same polarity are detected.
<2>	Error flag 2	0~6	The node accepts error flag 1 detection bit fill error and then issues this error flag.
<3>	Error delimiter	8	Continuous output of 8 recessive bits. If an dominant bit is detected in the 8th bit, the next bit emits an overload frame
<4>	Error bit	–	Error detected An error flag is issued immediately after the error bit. . If it is a CRC error, the error bit is issued after the ACK delimiter
<5>	Inter-frame space/overflow frames	–	An inter-frame space or overload frame is emitted at this time

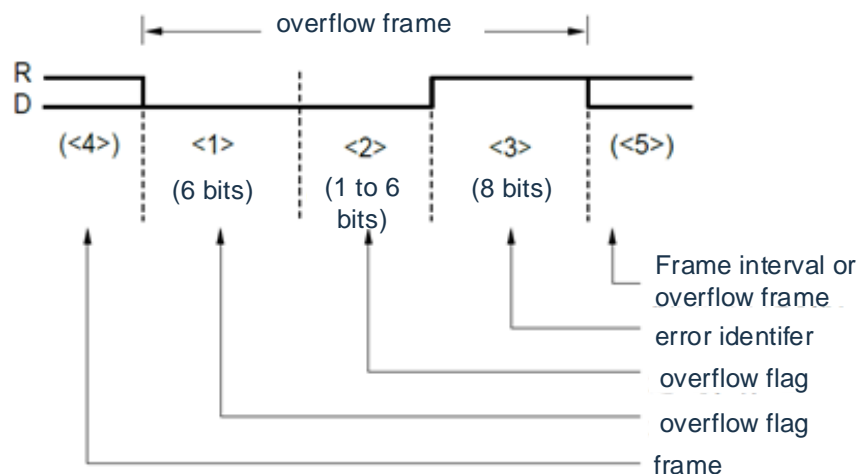
22.2.5 Overload frames

Overload frames are emitted in the following situations.

- When the receive node does not complete the receive operation ¹
- During the gap, if the first two bits are detected dominant bits
- If the dominant bit is at the last bit of the end frame (bit 7) or the last bit of the error delimiter/overflow delimiter (bit 8)

1: CAN is fast enough to process all received frames so that no overload frames are generated.

Figure 22-16: Overload frames



Note D: Dominant =0

R: Recessive = 1

Table22-8. Overload Frame field Definition

No	name	Number of digits	definition
<1>	Overflow flag	6	Continuous output of 6 dominant bits
<2>	Overflow flags for other nodes	0 to 6	An overflow flag is received in the inter-frame space and an overflow flag is emitted
<3>	Overflow delimiter	8	Continuous output of 8 recessive bits If the dominant bit is detected in the 8th bit, then the next bit emits an overflow frame
<4>	frame	–	Output after end frame, error delimiter, overload delimiter
<5>	Inter-frame space/overflow frames	–	This is when inter-frame space or overload frames are emitted

22.3 function

22.3.1 Bus prioritization

- (1) When a node starts transmitting:
 - When the bus is idle, the sending node sends the data first
- (2) When multiple nodes start transmitting:
 - The node that continuously outputs the longest dominant bit starting from the first bit of the arbitral field gets the bus priority (if the dominant and recessive bits are transmitted at the same time, the dominant bit is used as the bus value).
 - The transport node compares its output Arbitration field to the data level on the bus

Table22-9. Bus Priority Configuration

Level matching	Continuous transmission
Levels do not match	When a mismatch is detected, the transmission is stopped, and then the next bit begins to receive

- (3) The priority of data frames and remote frames
 - When a data frame and a remote frame are on the bus at the same time, the data frame has precedence because the RTR bit of the data frame (the last bit of the arbitral field) is the dominant bit.

Note If the extended format data frame and the standard format remote frame conflict on the bus (if ID18 through ID28 are the same), the standard format remote frame has priority.

22.3.2 Bit padding

Bit padding is when the level of the same polarity lasts 5 bits, and then a bit with opposite polarity is inserted to establish synchronization to prevent bursting errors.

Table22-10. Bit stuffing

Send	When sending a data frame or a remote frame, if 5 identical bits appear consecutively in the data between the start of the frame and the ACK field, a bit with opposite polarity is inserted before the next bit.
reception	When receiving a data frame or a remote frame, when the data between the start frame and the ACK field appears consecutively the same 5 bits, the next bit is deleted to continue receiving data.

22.3.3 Multi-master

Because the bus priority (the node that gets the transport capability) is determined by the identifier, either node can act as the master of the bus.

22.3.4 Multicast

Even if there is only one transport node, two or more nodes can receive the same data at the same time because the same identifier can be set on two or more nodes.

22.3.5 CAN sleep mode /CAN stop mode function

THE CAN sleep/CAN stop mode puts the CAN controller into wait mode for lower power consumption.

CAN sleep mode can wake up from bus operation, but CAN stop mode cannot be woken up by bus operation (CAN stop mode is controlled by the CPU).

22.3.6 Error control function

(4) The type of error

Table22-11. Error Type

type	Error description		Detection status	
	Detection method	Detection conditions	Send/Receive	Field/frame
Bit error	Compare output levels to bus levels	Mismatched level	Send/Receive nodes	Data bits on the bus between frame start and end frames, error frames, and overload frames
Fill errors	Detects received data at the padding bit	Six output levels are consecutively identical bits	Receive node	Frame starts to CRC timing
CRC error	Compare CRC from the received data and the received CRC timing	CRC mismatch	Receive node	CRC field
Malformation	Detects fields/frames in a mixed format	Detect mixed format errors	Receive node	CRC delimiter ACK field End frame Error frame Overload frames
ACK error	Detects the ACK gap in the sending node	Detect recessive bits in the ACK gap	Sending node	ACK clearance

(5) Error frame output timing

Table22-12. Error Frame output timing sequence

type	Output timing
Bit error, padding error, format error,	Starts outputting error frames when an error is detected
CRC error	Starts outputting error frames after the ACK delimiter

(6) Handling of errors

The sending node resends the data frame or remote frame after the wrong frame (the frame is not resended in single-shot mode).

(7) Error status

(a) The type of error status

The next three error states are defined by the CAN specification

- Proactive error
- Passive error
- Bus shutdown

The error type is determined by the values of the CAN Error Count Register (CnERC) of TEC0 to TEC7 (Transmit Error Count Bit) and REC0 to REC6 (Receive Error Count Bit), as shown in Table22-13.

The current error is determined by the information register (CnINFO) of the CAN module.

When each counter becomes equal or greater than the error warning level (96), the TECS0 or RECS0 of the CnINFO register is set to 1. In this case, the state of the bus must be tested, as there may be a serious fault at this point. When the value of the error counter is 128 or greater, indicating that it is in a passive error state, register CNINFO's TECS1 or RECS1 is set at 1.

- If the value of the transmit error counter is greater than or equal to 256 (in fact, the transmission error counter does not indicate a value greater than or equal to 256), the bus shutdown state arrives and the BOFF bit of the CnINFO register is set to 1.
- At startup, if only one node on the bus is activated (e.g. a special case, such as when the bus is only connected to the local site), the ACK will not be returned, even if the data is transferred. Therefore, duplicate transmission error frames and data. In the passive error state, the transmit error counter does not increment and the bus shutdown state does not appear.

Table22-13. Error Status Type

type	operate	The value of the error counter	Cn INFO register	Get the operational details of the error status
Proactive error	Send	0-95	TECS1, TECS0 =00	- Outputs an active error flag (6 persistent dominant bits).
	reception	0-95	RECS1, RECS0 =00	
	Send	96-127	TECS1, TECS0 =01	
	reception	96-127	RECS1, RECS0 =01	
Passive error	Send	128-255	TECS1, TECS0 =11	- Outputs a passive error flag (6 persistent recessive bits). - 8 recessive bits are sent in the transmission followed by a gap (suspended transmission).
	reception	128 or more	RECS1, RECS0 =11	
Bus shutdown	Send	256 or more (not specified). ¹	BOFF =1, TECS1, TECS0 =11	- Communication is no longer possible Receiving frames will not be stored, and the next <1>, <2>, and <3> are completed <1> TSOUT flip <2> REC increments/decrements <3> the VALID bit is set - If the CAN module enters initialization mode, a transition request to operating mode is made, when 11 persistent recessive bits are detected to 128 times, the error counter is reset to 0, and the active error status is logged

1: When the value of the BOFF bit is 1, the value of the Transmission Error Counter (TEC) is invalid if an error increases the value of the Error Transmission Counter by 8 when the value of the counter is in the range of 248 to 255, the counter does not increase any more and is in the bus off state.

(b) Error counters

When an error occurs, the error counter counts upwards; When successfully accepted and sent, the error counter counts down. When an error is detected, the value of the error counter is updated immediately.

Table22-14. Error Counter

state	Transmission error counter (TEC0 through TEC7).	Receive error counter (REC0 ~REC6).
The receiving node detects an error (in addition to the active error flag or overload flag).	Nothing has changed	+1 (when REPS bit=0).
The error flag for the error frame is emitted after the receiving node detects the dominant bit	Nothing has changed	+8 (when REPS bit=0).
The sending node issues an error flag [With exception, the error counter does not change in the following cases]. <1> ACK error is detected in the passive error state and the dominant bit is not detected when the passive error flag is emitted <2> the Arbitration padding error is detected, sending a recessive bit as padding, but the dominant bit is detected	+8	Nothing has changed
Bit error detected while active error flag or overload flag output (active error sending node)	+8	Nothing has changed
Bit error detected while active error flag or overload flag output (active error receiving node)	Nothing has changed	+8 (when REPS bit =0).
When a node starts with an active error flag or overload flag, it detects 14 consecutive dominant bits, and then detects 8 consecutive dominant bits. When a node detects 8 consecutive dominant bits after a passive error flag.	+8 (at transmission time).	+8 (when received, when REPS bit =0).
The transport node completed the transfer without errors (± 0 if the error counter = 0).	-1	Nothing has changed
The receiving node completed the receive without errors	Nothing has changed	- -1 (1 REC6 to REC0 127 when REPS bit = 0). - ± 0 (REC6 to REC0 = 0, When REPS bit =0). Values 119 to 127 are set (when REPS bit=1).

(c) Bit errors occur intermittently

Overload frame generation

Warning If an error occurs, the error flag output (active or passive) is controlled based on the content of the error counter that was transmitted before the error occurred and the content that receives the error pair. When the error flag is output, the value of the error counter is incremented.

(8) Bus shutdown resume

When the CAN module is in the bus off state, the CAN module permanently sets its output signal (CTxD) to the recessive bit

The CAN module recovers from the bus shutdown state in the following bus shutdown recovery sequence

<1> apply to enter CAN initialization mode

<2> apply to enter CAN mode of operation

(a) Recover from a normal recovery sequence

(b) Skip the recovery sequence to force a recovery operation

(a) Resume from the bus shutdown state in the normal recovery sequence

The first release of the CAN module enters the initialization request (refer to timing <1> in Figure 22-17). This application will be Acknowledged immediately and the OPMODE bit of cnCTRL will be cleared to 0. Analyzing the handling of this fault will result in a bus shutdown state, using software to redefine the CAN module and message cache, or stopping CAN operation by clearing the GOM bits.

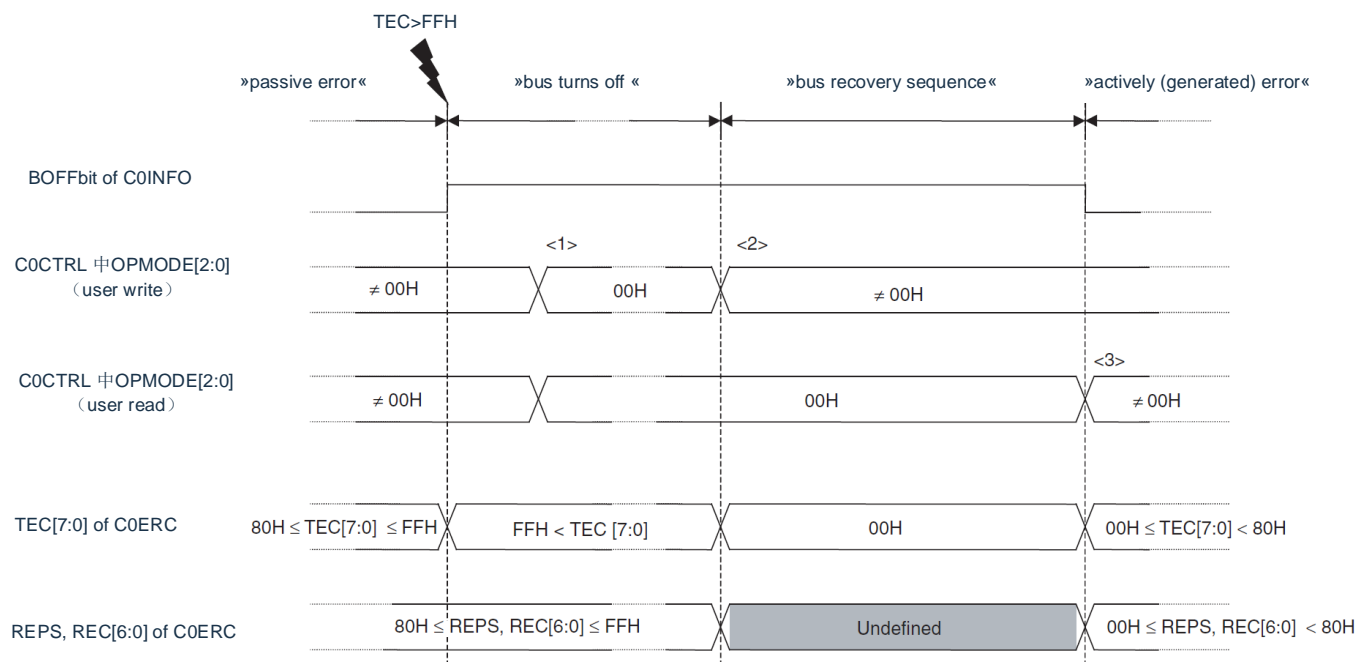
Next, the user requests from the initial state to the mode of operation (refer to timing <2> in Figure 22-17). This initiates an operation to recover the CAN module from the bus shutdown state. The CAN protocol ISO11898 defines the conditions under which a module recovers from a bus shutdown state and needs to detect 11 consecutive recessive bits 128 times. At this point, requests to change the mode of operation remain pending until the recovery conditions are met. When the recovery criteria are met. (Referring to Timing <3> in Figure 22-17), the CAN module can enter the operating mode it requests. It will remain in initialization mode until the CAN module enters this mode of operation. By reading the OPMODE bit of the CnCTRL register, you can confirm the completion of the operation mode to be requested.

During bus shutdown and during bus shutdown recovery sequences, the BOFF bit of the CnINFO register is set (set to 1). In the bus shutdown recovery sequence, the receive error counter (REC [6:0]) calculates the number of times 11 consecutive recessive bits were detected on the bus. Therefore, you can check the recovery status by reading REC_6:0.

warning

1. If the bus shutdown recovery sequence is interrupted by entering initialization mode and re-entering any operating mode, the bus shutdown recovery sequence will restart from scratch from this point and the waiting phase will again be 128 times 11 recessive bits.
 - a) In the bus shutdown recovery sequence, REC [6:0] counts (+1) each time 11 consecutive recessive bits are detected. The CAN module can enter CAN sleep mode or CAN stop mode even during bus shutdown. To start the bus shutdown recovery sequence, you must shift to initialization mode once.
2. However, when the CAN module is in CAN sleep mode or CAN stop mode, requests to transition to initialization mode are not accepted, so CAN sleep mode must be released first. In this case, once the CAN sleep mode is released, the bus shutdown resume sequence is started and there is no need to transition to initialization mode. If the CAN module detects an explicit edge on the CAN bus in sleep mode during bus shutdown, the sleep mode will leave and the bus shutdown recovery sequence will start (in the state where the CAN clock is provided, the PSMODE must be cleared by software after the dominant edge).

Figure 22-17. Resumes an operation from a bus shutdown state through a normal recovery sequence



(b) Skip the bus shutdown of the forced recovery operation for the recovery sequence

Regardless of the bus state, the CAN module can be forcibly released from the bus shutdown state by skipping the bus shutdown recovery sequence. This is the process

First, the CAN module requests to enter initialization mode. For actions and points to be aware of at this point, see (a) Resuming Operations from a Bus Shutdown State through a Normal Recovery Sequence

Next, the module requests to enter operation mode. At the same time, the CCERC bit of the CnCTRL register must be set to 1

Therefore, the bus shutdown recovery sequence defined by the CAN protocol ISO 11898 is skipped and the module immediately enters operating mode. In this case, the module is connected to the CAN bus after monitoring 11 consecutive recessive bits. For more information, see Fig.22-80

Warning This feature is not defined by the CAN protocol ISO 11898. When using this feature, thoroughly evaluate its impact on network systems

(9) Initialization of the CAN Module Error Counter Register (CnERC) in initialization mode

If you need to initialize the CAN Module Error Counter Register (CnERC) and the CAN Module Information Register (CnINFO) to debug or evaluate the program, it can be done in initialization mode. After initialization is complete, the CCERC bit is automatically cleared to 0

Warning 1 This feature is enabled only in initialization mode. Even if the CCERC bit is set to 1 in CAN mode of operation, the registers CnERC and CnINFO are not initialized

2. The CCERC bit can be set at the same time as the request to enter CAN operation mode

22.3.7 Baud rate control function

(10) Prescale

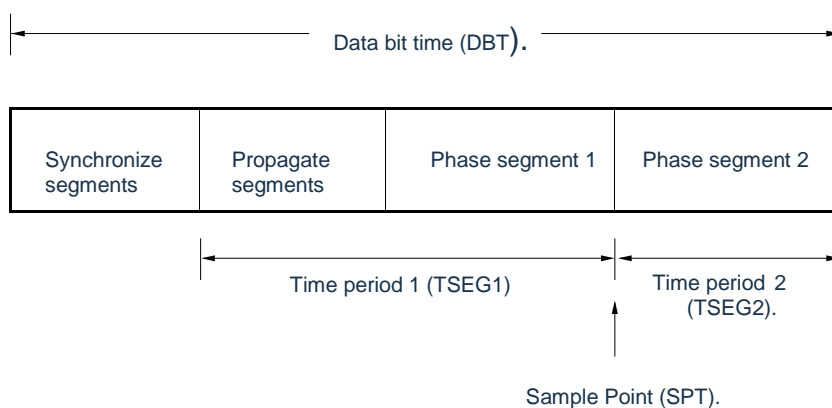
The CAN controller has a prescaled clock (fCAN) that is supplied to the CAN. This prescale generates a CAN Protocol Layer Base Clock (fTQ) derived from the CAN Module System Clock (fCANMOD), divided by frequencies from 1 to 256 (see 22.7.13CAN Bit Rate Prescaler Register (CnBRP)).

(11) Data bit time (8-25 time quantization)

A data bit time is defined as Figure 22-18

The CAN controller takes time period 1, period 2, and resynchronization jump width (SJW) as parameters for the data bit time, as Figure 22-18 Period 1 is equivalent to the sum of the propagation (prop) segment and phase segment 1 as defined by the CAN Protocol specification. Period 2 is equivalent to phase segment 2

Figure 22-18. Time period settings



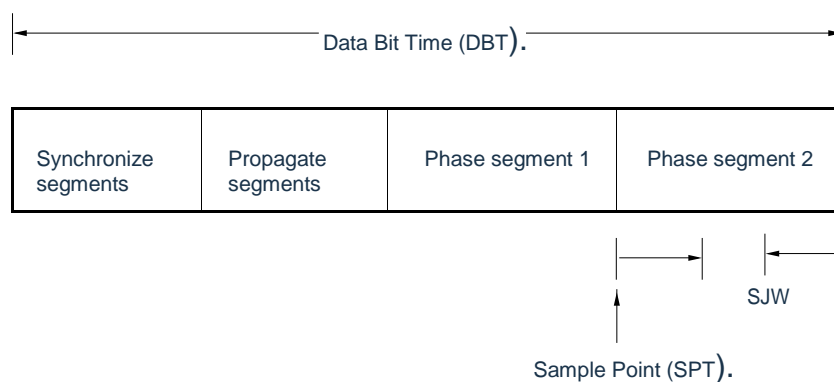
Segment name	The range that can be set	A note about setting up the SPECIFICATION for CAN
Time period 1 (TSEG1).	2TQ-16TQ	—
Time period 2 (TSEG2).	1TQ-8TQ	The IPT of the CAN controller is 0TQ. Therefore, in order to comply with the CAN Protocol specification, the length must be set here to be less than or equal to phase 1. This means that the length of the period 1 minus 1TQ is the upper limit of the period 2
Resynchronize Jump Width (SJW).	1TQ-4TQ	Period 1 length minus 1TQ or 4TQ, whichever is smaller

Remarks: IPT: Information processing time

TQ: Time share

Reference: The CAN standard ISO 11898 specification defines the segments that make up the data bit time, as shown Figure 22-19

Figure 22-19. Reference: Data bit time configuration as defined by the CAN specification



Segment name	Segment length	description
Synchronize segments	1	This segment begins at the edge where the level changes from recessive to dominant when hard synchronization is established
Propagate segments	Programmable from 1 to 8 or more	This segment absorbs the latency of the output buffer, CAN bus, and input buffer. Set the length of this segment so that ACK is returned before phase 1 starts Propagation period (output buffer delay) + 2x (CAN bus delay) + (input buffer delay).
Phase segment 1	Programmable 1 to 8	This segment compensates for errors in data bit times. The longer this segment, the wider the allowable range, but the slower the communication speed
Phase segment 2	Phase band 1 or IPT, whichever is the largest	
SJW	Programmable from 1TQ to segment 1 or 4TQ, whichever is the smallest	This width sets the upper limit of the phase segment that is expanded or contracted during resynchronization

Remarks: IPT: Information processing time

TQ: Time share

(12) Synchronize data bits

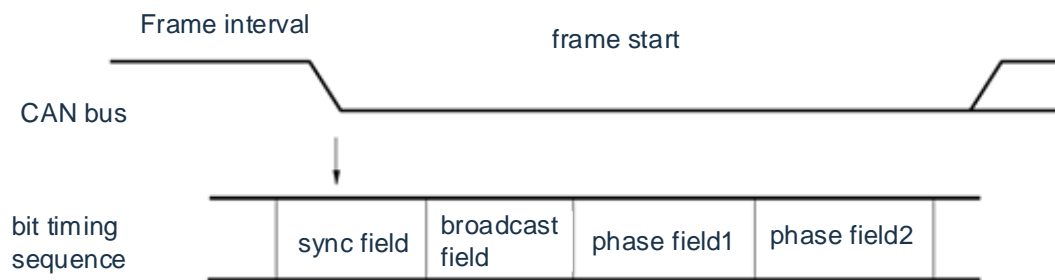
- The receive node establishes synchronization by changing the level on the bus because it has no asynchronous signal
- The transport node transmits data synchronously at the bit time of the transport node

(a) Hardware synchronization

This synchronization is established when the receiving node detects the start of a frame in the inter-frame space

- When a falling edge is detected on the bus, TQ represents a synchronous segment, and the next segment is a propagation segment. In this case, synchronization is established regardless of the SJW

Figure 22-20. Hard synchronization when the dominant bit is recognized during bus idle



(b) Resynchronize

If a level change is detected on the bus during the receive, synchronization is established again (only if the recessive level was previously sampled).

- The phase error of the edge is given by the relative position of the detected edge and the synchronization segment

< phase error signal >

0: If the edge is within the synchronization segment

Positive: If the edge is before the sample point (phase error).

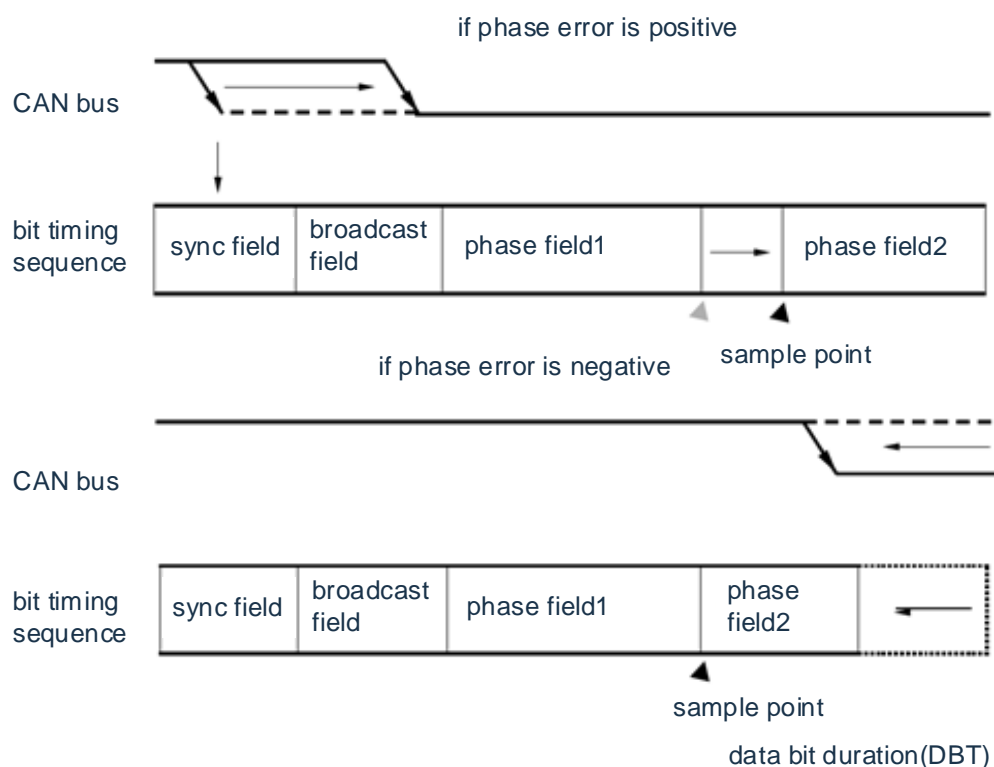
Negative: If the edge is behind the sample point (phase error).

If the phase error is positive: by the specified SJW, phase segment 1 is longer

If the phase error is negative: Phase segment 2 is shortened by the specified SJW

- Due to the "difference" in baud rate between the sending node and the receiving node, the sampling point of the receiving node data moves relatively

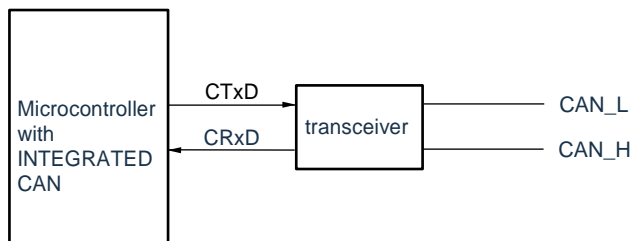
Figure 22-21: Resynchronize



22.4 The connection to the target system

The microcontroller that integrates CAN must be connected to the CAN bus using an external transceiver

Figure22-22. Connect to the CAN bus



22.5 Internal registers of the CAN controller

22.5.1 CAN controller configuration

Table22-15. CAN Control Register List(1/2)

item	Register name
Control registers	Peripheral enable register 0/2 (PER0/2).
	Serial communication pin selection register (PIOR3).
	Port registers 0, 4, 5, 6 (P0, P4, P5, P6).
	Port mode registers 0, 4, 5, 6 (PM0, PM4, PM5, PM6).
CAN global registers	CAN Global Module Control Register (CnGMCTRL).
	CAN Global Module Clock Selection Register (CnGMCS).
	CAN Global Automatic Block Transfer Control Register (CnGMABT).
	CAN Global Automatic Block Delay Setting Register (CnGMABTD).
CAN module registers	CAN module shield 1 register L (CnMASK1L).
	CAN module shield 1 register H (CnMASK1H).
	CAN module shield 2 register L (CnMASK2L).
	CAN module shield 2 register H (CnMASK2H).
	The CAN module shields 3 register L (CnMASK3L).
	THE CAN module shields 3 register H (CNMASK3H).
	THE CAN module shields 4 register L (CnMASK4L).
	THE CAN module shields 4 registers H (CnMASK4H).
	CAN Module Control Register (CnCTRL).
	CAN Module Last Error Code Register (CnLEC).
	CAN Module Information Register (CnINFO).
	CAN Module Error Count Register (CnERC).
	CAN Module Interrupt Enable Register (CnIE).
	CAN Module Interrupt Status Register (CnINTS).
	CAN Module Bit Rate Scaling Register (CnBRP).
	CAN Module Bit Rate Register (CnBTR).
	The last input pointer register (CnLIPT) of the CAN module
	The CAN module receives the History List Register (CnRGPT).
	The last output pointer register (CnLOPT) of the CAN module
	The CAN module sends the History List Register (CnTGPT).
	CAN Module Timestamp Register (CnTS).

Note: The CAN global register is > recognized by the CnGM < register function.

THE CAN module registers are > identified by the Cn< register function.

The message buffer registers are > recognized by the CnM < register function.

BAT32A239: n=0.1 BAT32A279: n=0.1.2

Table22-15. CAN Control register list(2/2)

item	Register name
Message cache registers	CAN message data byte 01 register m (CnMDB01m).
	CAN message data byte 0 register m (CnMDB0m).
	CAN message data byte 1 register m (CnMDB1m).
	CAN message data byte 23 register m (CnMDB23m).
	CAN message data byte 2 register m (CnMDB2m).
	CAN message data byte 3 register m (CnMDB3m).
	CAN message data byte 45 register m (CnMDB45m).
	CAN message data byte 4 register m (CnMDB4m).
	CAN message data byte 5 register m (CnMDB5m).
	CAN message data byte 67 register m (CnMDB67m).
	CAN message data byte 6 register m (CnMDB6m).
	CAN message data byte 7 register m (CnMDB7m).
	CAN message data length register m (CnMDLCm).
	CAN message configuration register m (CnMCONFm).
	CAN message ID register Lm (CnMIDLm).
	CAN message ID register Hm (CnMIDHm).
	CAN message control register m (CnMCTRLm).

Note 1: THE CAN global registers are > identified by the CnGM < register function.

Module registers are > identified by the Cn< register function.

The message buffer registers are > recognized by the CnM < register function.

2.m=0 to 15

BAT32A239: n=0.1 BAT32A279: n=0.1.2

22.5.2 Register access type

The access types of the control registers of the CAN controller are shown in the following table

where the base address is: CAN0 0x40045400; CAN1 0x40045800; CAN2 0x40046400

BAT32A239: n=0.1 BAT32A279: n=0.1.2

Table22-16. Register Access Type (1/9)

Offset address	Register name	Symbol	Read/write	Bit operating unit			The default value
				1	8	16	
0x000H	The CAN global module controls registers	CnGMCTRL	Read/write				0000H
0x006H	CAN global automatic block transfer control register	CnGMABT					0000H
0x008H	CAN global automatic block transfer delay setting	CnGMABTD					00H
0x002H	CAN global module clock selection register	CnGMCS					0FH
0x040H	CAN module mask 1 register	CnMASK1L					There is no definition
0x042H		CnMASK1H					There is no definition
0x044H	CAN module masks 2 registers	CnMASK2L					There is no definition
0x046H		CnMASK2H					There is no definition
0x048H	The CAN module masks 3 registers	CnMASK3L					There is no definition
0x04AH		CnMASK3H					There is no definition
0x04CH	The CAN module masks 4 registers	CnMASK4L					There is no definition
0x04EH		CnMASK4H					There is no definition
0x050H	THE CAN module controls the registers	CnCTRL					0000H
0x052H	THE LAST ERROR CODE REGISTER FOR THE CAN MODULE	CnLEC					00H
0x053H	CAN module information registers	CnINFO	read				00H
0x054H	CAN module error count register	CnERC	read				0000H
0x056H	CAN module interrupt enable register	Miss	Read/write				0000H
0x058H	CAN module interrupt status register	CnINTS					0000H
0x05AH	CAN module bit rate scaling register	CnBRP					FFH
0x05CH	CAN module bit rate register	CnBTR					370FH
0x05EH	The CAN module finally enters the pointer register	CnLIPT	read				There is no definition
0x060H	The CAN module receives the history list register	CnRGPT	Read/write				xx02H
0x062H	The CAN module finally outputs the pointer register	CnLOPT	read				There is no definition
0x064H	The CAN module sends a history list register	CnTGPT	Read/write				xx02H
0x066H	CAN module timestamp register	CnTS					0000H

Table22-16. Register Access Type (2/9)

Offset address	Register name	Symbol	Read/ write	Bit operating unit			The default value
				1	8	16	
0x100H	CAN message data byte 01 register 00	CnMDB0100	Read/ write	–	–		There is no definition
0x100H	CAN0 message data bytes 0 register 00	CnMDB000		–		–	There is no definition
0x101H	CAN0 message data byte 1 register 00	CnMDB100		–		–	There is no definition
0x102H	CAN message data byte 23 register 00	CnMDB2300		–	–		There is no definition
0x102H	CAN0 message data byte 2 register 00	CnMDB200		–		–	There is no definition
0x103H	CAN0 message data byte 3 register 00	CnMDB300		–		–	There is no definition
0x104H	CAN message data byte 45 register 00	CnMDB4500		–	–		There is no definition
0x104H	CAN message data byte 4 register 00	CnMDB400		–		–	There is no definition
0x105H	CAN message data byte 5 register 00	CnMDB500		–		–	There is no definition
0x106H	CAN message data byte 67 register 00	CnMDB6700		–	–		There is no definition
0x106H	CAN message data byte 6 register 00	CnMDB600		–		–	There is no definition
0x107H	CAN message data byte 7 register 00	CnMDB700		–		–	There is no definition
0x108H	CAN message data length register 00	CnMDLCn0		–		–	0000xxxxB
0x109H	CAN message configuration register 00	CnMCONF00		–		–	There is no definition
0x10AH	CAN message ID register 00	CnMIDL00		–	–		There is no definition
0x10CH		CnMIDH00		–	–		There is no definition
0x10EH	CAN message control register 00	CnMCTRL00		–	–		00x00000 000xx000B
0x110H	CAN message data byte 01 register 01	CnMDB0101		–	–		There is no definition
0x110H	CAN message data byte 0 register 01	CnMDB001		–		–	There is no definition
0x111H	CAN message data byte 1 register 01	CnMDB101		–		–	There is no definition
0x112H	CAN message data byte 23 register 01	CnMDB2301		–	–		There is no definition
0x112H	CAN message data byte 2 register 01	CnMDB201		–		–	There is no definition
0x113H	CAN message data byte 3 register 01	CnMDB301		–		–	There is no definition
0x114H	CAN message data byte 45 register 01	CnMDB4501		–	–		There is no definition
0x114H	CAN message data byte 4 register 01	CnMDB401		–		–	There is no definition
0x115H	CAN message data byte 5 register 01	CnMDB501		–		–	There is no definition
0x116H	CAN message data byte 67 register 01	CnMDB6701		–	–		There is no definition
0x116H	CAN message data byte 6 register 01	CnMDB601		–		–	There is no definition
0x117H	CAN message data byte 7 register 01	CnMDB701		–		–	There is no definition
0x118H	CAN message data length register 01	CnMDLCn1		–		–	0000xxxxB
0x119H	CAN message configuration register 01	CnMCONF01		–		–	There is no definition
0x11AH	CAN message ID register 01	CnMIDL01		–	–		There is no

							definition
0x11CH		CnMIDH01		–	–		There is no definition
0x11EH	CAN message control register 01	CnMCTRL01		–	–		00x00000 000xx000B

Table22-16. Register Access Type (3/9)

Offset address	Register name	Symbol	Read/ write	Bit operating unit			The default value
				1	8	16	
0x120H	CAN message data byte 01 register 02	CnMDB0102	Read/ write	–	–		There is no definition
0x120H	CAN message data byte 0 register 02	CnMDB002		–		–	There is no definition
0x121H	CAN message data byte 1 register 02	CnMDB102		–		–	There is no definition
0x122H	CAN message data byte 23 register 02	CnMDB2302		–	–		There is no definition
0x122H	CAN message data byte 2 register 02	CnMDB202		–		–	There is no definition
0x123H	CAN message data byte 3 register 02	CnMDB302		–		–	There is no definition
0x124H	CAN message data byte 45 register 02	CnMDB4502		–	–		There is no definition
0x124H	CAN message data byte 4 register 02	CnMDB402		–		–	There is no definition
0x125H	CAN message data byte 5 register 02	CnMDB502		–		–	There is no definition
0x126H	CAN message data byte 67 register 02	CnMDB6702		–	–		There is no definition
0x126H	CAN message data byte 6 register 02	CnMDB602		–		–	There is no definition
0x127H	CAN message data byte 7 register 02	CnMDB702		–		–	There is no definition
0x128H	CAN message data length register 02	CnMDLCn2		–		–	0000xxxxB
0x129H	CAN message configuration register 02	CnMCONF02		–		–	There is no definition
0x12AH	CAN message ID register 02	CnMIDL02		–	–		There is no definition
0x12CH		CnMIDH02		–	–		There is no definition
0x12EH	CAN message control register 02	CnMCTRL02		–	–		00x00000 000xx000B
0x130H	CAN message data byte 01 register 03	CnMDB0103		–	–		There is no definition
0x130H	CAN message data byte 0 register 03	CnMDB003		–		–	There is no definition
0x131H	CAN message data byte 1 register 03	CnMDB103		–		–	There is no definition
0x132H	CAN message data byte 23 register 03	CnMDB2303		–	–		There is no definition
0x132H	CAN message data byte 2 register 03	CnMDB203		–		–	There is no definition
0x133H	CAN message data byte 3 register 03	CnMDB303		–		–	There is no definition
0x134H	CAN message data byte 45 register 03	CnMDB4503		–	–		There is no definition
0x134H	CAN message data byte 4 register 03	CnMDB403		–		–	There is no definition
0x135H	CAN message data byte 5 register 03	CnMDB503		–		–	There is no definition
0x136H	CAN message data byte 67 register 03	CnMDB6703		–	–		There is no definition
0x136H	CAN message data byte 6 register 03	CnMDB603		–		–	There is no definition
0x137H	CAN message data byte 7 register 03	CnMDB703		–		–	There is no definition
0x138H	CAN message data length register 03	CnMDLCn3		–		–	0000xxxxB
0x139H	CAN message configuration register 03	CnMCONF03		–		–	There is no definition
0x13AH	CAN message ID register 03	CnMIDL03		–	–		There is no

							definition
0x13CH		CnMIDH03		–	–		There is no definition
0x13EH	CAN message control register 03	CnMCTRL03		–	–		00x00000 000xx000B

Table22-16. Register Access Type (4/9)

Offset address	Register name	Symbol	Read/ write	Bit operating unit			The default value
				1	8	16	
0x140H	CAN message data byte 01 register 04	CnMDB0104	Read/ write	–	–		There is no definition
0x140H	CAN message data byte 0 register 04	CnMDB004		–		–	There is no definition
0x141H	CAN message data byte 1 register 04	CnMDB104		–		–	There is no definition
0x142H	CAN message data byte 23 register 04	CnMDB2304		–	–		There is no definition
0x142H	CAN message data byte 2 register 04	CnMDB204		–		–	There is no definition
0x143H	CAN message data byte 3 register 04	CnMDB304		–		–	There is no definition
0x144H	CAN message data byte 45 register 04	CnMDB4504		–	–		There is no definition
0x144H	CAN message data byte 4 register 04	CnMDB404		–		–	There is no definition
0x145H	CAN message data byte 5 register 04	CnMDB504		–		–	There is no definition
0x146H	CAN message data byte 67 register 04	CnMDB6704		–	–		There is no definition
0x146H	CAN message data byte 6 register 04	CnMDB604		–		–	There is no definition
0x147H	CAN message data byte 7 register 04	CnMDB704		–		–	There is no definition
0x148H	CAN message data length register 04	CnMDLCn4		–		–	0000xxxxB
0x149H	CAN message configuration register 04	CnMCONF04		–		–	There is no definition
0x14AH	CAN message ID register 04	CnMIDL04		–	–		There is no definition
0x14CH		CnMIDH04		–	–		There is no definition
0x14EH	CAN message control register 04	CnMCTRL04		–	–		00x00000 000xx000B
0x150H	CAN message data byte 01 register 05	CnMDB0105		–	–		There is no definition
0x150H	CAN message data byte 0 register 05	CnMDB005		–		–	There is no definition
0x151H	CAN message data byte 1 register 05	CnMDB105		–		–	There is no definition
0x152H	CAN message data byte 23 register 05	CnMDB2305		–	–		There is no definition
0x152H	CAN message data byte 2 register 05	CnMDB205		–		–	There is no definition
0x153H	CAN message data byte 3 register 05	CnMDB305		–		–	There is no definition
0x154H	CAN message data byte 45 register 05	CnMDB4505		–	–		There is no definition
0x154H	CAN message data byte 4 register 05	CnMDB405		–		–	There is no definition
0x155H	CAN message data byte 5 register 05	CnMDB505		–		–	There is no definition
0x156H	CAN message data byte 67 register 05	CnMDB6705		–	–		There is no definition
0x156H	CAN message data byte 6 register 05	CnMDB605		–		–	There is no definition
0x157H	CAN message data byte 7 register 05	CnMDB705		–		–	There is no definition
0x158H	CAN message data length register 05	CnMDLCn5		–		–	0000xxxxB
0x159H	CAN message configuration register 05	CnMCONF05		–		–	There is no definition
0x15AH	CAN message ID register 05	CnMIDL05		–	–		There is no

							definition
0x15CH		CnMIDH05		–	–		There is no definition
0x15EH	CAN message control register 05	CnMCTRL05		–	–		00x00000 000xx000B

Table22-16. Register Access Type (5/9)

Offset address	Register name	Symbol	Read/ write	Bit operating unit			The default value
				1	8	16	
0x160H	CAN message data byte 01 register 06	CnMDB0106	Read/ write	–	–		There is no definition
0x160H	CAN message data byte 0 register 06	CnMDB006		–		–	There is no definition
0x161H	CAN message data byte 1 register 06	CnMDB106		–		–	There is no definition
0x162H	CAN message data byte 23 register 06	CnMDB2306		–	–		There is no definition
0x162H	CAN message data byte 2 register 06	CnMDB206		–		–	There is no definition
0x163H	CAN message data byte 3 register 06	CnMDB306		–		–	There is no definition
0x164H	CAN message data byte 45 register 06	CnMDB4506		–	–		There is no definition
0x164H	CAN message data byte 4 register 06	CnMDB406		–		–	There is no definition
0x165H	CAN message data byte 5 register 06	CnMDB506		–		–	There is no definition
0x166H	CAN message data byte 67 register 06	CnMDB6706		–	–		There is no definition
0x166H	CAN message data byte 6 register 06	CnMDB606		–		–	There is no definition
0x167H	CAN message data byte 7 register 06	CnMDB706		–		–	There is no definition
0x168H	CAN message data length register 06	CnMDLCn6		–		–	0000xxxxB
0x169H	CAN message configuration register 06	CnMCONF06		–		–	There is no definition
0x16AH	CAN message ID register 06	CnMIDL06		–	–		There is no definition
0x16CH		CnMIDH06		–	–		There is no definition
0x16EH	CAN message control register 06	CnMCTRL06		–	–		00x00000 000xx000B
0x170H	CAN message data byte 01 register 07	CnMDB0107		–	–		There is no definition
0x170H	CAN message data byte 0 register 07	CnMDB007		–		–	There is no definition
0x171H	CAN message data byte 1 register 07	CnMDB107		–		–	There is no definition
0x172H	CAN message data byte 23 register 07	CnMDB2307		–	–		There is no definition
0x172H	CAN message data byte 2 register 07	CnMDB207		–		–	There is no definition
0x173H	CAN message data byte 3 register 07	CnMDB307		–		–	There is no definition
0x174H	CAN message data byte 45 register 07	CnMDB4507		–	–		There is no definition
0x174H	CAN message data byte 4 register 07	CnMDB407		–		–	There is no definition
0x175H	CAN message data byte 5 register 07	CnMDB507		–		–	There is no definition
0x176H	CAN message data byte 67 register 07	CnMDB6707		–	–		There is no definition
0x176H	CAN message data byte 6 register 07	CnMDB607		–		–	There is no definition
0x177H	CAN message data byte 7 register 07	CnMDB707		–		–	There is no definition
0x178H	CAN message data length register 07	CnMDLCn7		–		–	0000xxxxB
0x179H	CAN message configuration register 07	CnMCONF07		–		–	There is no definition
0x17AH	CAN message ID register 07	CnMIDL07		–	–		There is no

							definition
0x17CH		CnMIDH07		–	–		There is no definition
0x17EH	CAN message control register 07	CnMCTRL07		–	–		00x00000 000xx000B

Table22-16. Register Access Type (6/9)

Offset address	Register name	Symbol	Read/ write	Bit operating unit			The default value
				1	8	16	
0x180H	CAN message data byte 01 register 08	CnMDB0108	Read/ write	–	–		There is no
0x180H	CAN message data byte 0 register 08	CnMDB008		–		–	There is no
0x181H	CAN message data byte 1 register 08	CnMDB108		–		–	There is no
0x182H	CAN message data byte 23 register 08	CnMDB2308		–	–		There is no
0x182H	CAN message data byte 2 register 08	CnMDB208		–		–	There is no
0x183H	CAN message data byte 3 register 08	CnMDB308		–		–	There is no
0x184H	CAN message data byte 45 register 08	CnMDB4508		–	–		There is no
0x184H	CAN message data byte 4 register 08	CnMDB408		–		–	There is no
0x185H	CAN message data byte 5 register 08	CnMDB508		–		–	There is no
0x186H	CAN message data byte 67 register 08	CnMDB6708		–	–		There is no
0x186H	CAN message data byte 6 register 08	CnMDB608		–		–	There is no
0x187H	CAN message data byte 7 register 08	CnMDB708		–		–	There is no
0x188H	CAN message data length register 08	CnMDLCn8		–		–	0000xxxxB
0x189H	CAN message configuration register 08	CnMCONF08		–		–	There is no
0x18AH	CAN message ID register 08	CnMIDL08		–	–		There is no
0x18CH		CnMIDH08		–	–		There is no
0x18EH	CAN message control register 08	CnMCTRL08		–	–		00x00000 000xx000B
0x190H	CAN message data byte 01 register 09	CnMDB0109		–	–		There is no
0x190H	CAN message data byte 0 register 09	CnMDB009		–		–	There is no
0x191H	CAN message data byte 1 register 09	CnMDB109		–		–	There is no
0x192H	CAN message data byte 23 register 09	CnMDB2309		–	–		There is no
0x192H	CAN message data byte 2 register 09	CnMDB209		–		–	There is no
0x193H	CAN message data byte 3 register 09	CnMDB309		–		–	There is no
0x194H	CAN message data byte 45 register 09	CnMDB4509		–	–		There is no
0x194H	CAN message data byte 4 register 09	CnMDB409		–		–	There is no
0x195H	CAN message data byte 5 register 09	CnMDB509		–		–	There is no
0x196H	CAN message data byte 67 register 09	CnMDB6709		–	–		There is no
0x196H	CAN message data byte 6 register 09	CnMDB609		–		–	There is no
0x197H	CAN message data byte 7 register 09	CnMDB709		–		–	There is no
0x198H	CAN message data length register 09	CnMDLCn9		–		–	0000xxxxB
0x199H	CAN message configuration register 09	CnMCONF09		–		–	There is no
0x19AH	CAN message ID register 09	CnMIDL09		–	–		There is no
0x19CH		CnMIDH09		–	–		There is no
0x19EH	CAN message control register 09	CnMCTRL09		–	–		00x00000 000xx000B

Table22-16. Register Access Type (7/9)

Offset address	Register name	Symbol	Read/write	Bit operating unit			The default value
				1	8	16	
0x1A0H	CAN message data byte 01 register 10	CnMDB0110	Read/write	–	–		There is no definition
0x1A0H	CAN message data byte 0 register 10	CnMDB010		–		–	There is no definition
0x1A1H	CAN message data byte 1 register 10	CnMDB110		–		–	There is no definition
0x1A2H	CAN message data byte 23 register 10	CnMDB2310		–	–		There is no definition
0x1A2H	CAN message data byte 2 register 10	CnMDB210		–		–	There is no definition
0x1A3H	CAN message data byte 3 register 10	CnMDB310		–		–	There is no definition
0x1A4H	CAN message data byte 45 register 10	CnMDB4510		–	–		There is no definition
0x1A4H	CAN message data byte 4 register 10	CnMDB410		–		–	There is no definition
0x1A5H	CAN message data byte 5 register 10	CnMDB510		–		–	There is no definition
0x1A6H	CAN message data byte 67 register 10	CnMDB6710		–	–		There is no definition
0x1A6H	CAN message data byte 6 register 10	CnMDB610		–		–	There is no definition
0x1A7H	CAN message data byte 7 register 10	CnMDB710		–		–	There is no definition
0x1A8H	CAN message data length register 10	CnMDLC10		–		–	0000xxxxB
0x1A9H	CAN message configuration register 10	CnMCONF10		–		–	There is no definition
0x1AAH	CAN message ID register 10	CnMIDL10		–	–		There is no definition
0x1ACH		CnMIDH10		–	–		There is no definition
0x1AEH	CAN message control register 10	CnMCTRL10		–	–		00x00000 000xx000B
0x1B0H	CAN message data byte 01 register 11	CnMDB0111		–	–		There is no definition
0x1B0H	CAN message data byte 0 register 11	CnMDB011		–		–	There is no definition
0x1B1H	CAN message data byte 1 register 11	CnMDB111		–		–	There is no definition
0x1B2H	CAN message data byte 23 register 11	CnMDB2311		–	–		There is no definition
0x1B2H	CAN message data byte 2 register 11	CnMDB211		–		–	There is no definition
0x1B3H	CAN message data byte 3 register 11	CnMDB311		–		–	There is no definition
0x1B4H	CAN message data byte 45 register 11	CnMDB4511		–	–		There is no definition
0x1B4H	CAN message data byte 4 register 11	CnMDB411		–		–	There is no definition
0x1B5H	CAN message data byte 5 register 11	CnMDB511		–		–	There is no definition
0x1B6H	CAN message data byte 67 register 11	CnMDB6711		–	–		There is no definition
0x1B6H	CAN message data byte 6 register 11	CnMDB611		–		–	There is no definition
0x1B7H	CAN message data byte 7 register 11	CnMDB711		–		–	There is no definition
0x1B8H	CAN message data length register 11	CnMDLC11		–		–	0000xxxxB
0x1B9H	CAN message configuration register 11	CnMCONF11		–		–	There is no definition
0x1BAH	CAN message ID register 11	CnMIDL11		–	–		There is no

							definition
0x1BCH		CnMIDH11		–	–		There is no definition
0x1BEH	CAN message control register 11	CnMCTRL11		–	–		00x00000 000xx000B

Table 22-16. Register Access Type (8/9)

Offset address	Register name	Symbol	Read/ write	Bit operating unit			The default value
				1	8	16	
0x1C0H	CAN message data byte 01 register 12	CnMDB0112	Read/ write	–	–		There is no definition
0x1C0H	CAN message data byte 0 register 12	CnMDB012		–		–	There is no definition
0x1C1H	CAN message data byte 1 register 12	CnMDB112		–		–	There is no definition
0x1C2H	CAN message data byte 23 register 12	CnMDB2312		–	–		There is no definition
0x1C2H	CAN message data byte 2 register 12	CnMDB212		–		–	There is no definition
0x1C3H	CAN message data byte 3 register 12	CnMDB312		–		–	There is no definition
0x1C4H	CAN message data byte 45 register 12	CnMDB4512		–	–		There is no definition
0x1C4H	CAN message data byte 4 register 12	CnMDB412		–		–	There is no definition
0x1C5H	CAN message data byte 5 register 12	CnMDB512		–		–	There is no definition
0x1C6H	CAN message data byte 67 register 12	CnMDB6712		–	–		There is no definition
0x1C6H	CAN message data byte 6 register 12	CnMDB612		–		–	There is no definition
0x1C7H	CAN message data byte 7 register 12	CnMDB712		–		–	There is no definition
0x1C8H	CAN message data length register 12	CnMDLC12		–		–	0000xxxxB
0x1C9H	CAN message configuration register 12	CnMCONF12		–		–	There is no definition
0x1CAH	CAN message ID register 12	CnMIDL12		–	–		There is no definition
0x1CCH		CnMIDH12		–	–		There is no definition
0x1CEH	CAN message control register 12	CnMCTRL12		–	–		00x00000 000xx000B
0x1D0H	CAN message data byte 01 register 13	CnMDB0113		–	–		There is no definition
0x1D0H	CAN message data byte 0 register 13	CnMDB013		–		–	There is no definition
0x1D1H	CAN message data byte 1 register 13	CnMDB113		–		–	There is no definition
0x1D2H	CAN message data byte 23 register 13	CnMDB2313		–	–		There is no definition
0x1D2H	CAN message data byte 2 register 13	CnMDB213		–		–	There is no definition
0x1D3H	CAN message data byte 3 register 13	CnMDB313		–		–	There is no definition
0x1D4H	CAN message data byte 45 register 13	CnMDB4513		–	–		There is no definition
0x1D4H	CAN message data byte 4 register 13	CnMDB413		–		–	There is no definition
0x1D5H	CAN message data byte 5 register 13	CnMDB513		–		–	There is no definition
0x1D6H	CAN message data byte 67 register 13	CnMDB6713		–	–		There is no definition
0x1D6H	CAN message data byte 6 register 13	CnMDB613		–		–	There is no definition
0x1D7H	CAN message data byte 7 register 13	CnMDB713		–		–	There is no definition
0x1D8H	CAN message data length register 13	CnMDLC13		–		–	0000xxxxB
0x1D9H	CAN message configuration register 13	CnMCONF13		–		–	There is no definition
0x1DAH	CAN message ID register 13	CnMIDL13		–	–		There is no

							definition
0x1DCH		CnMIDH13		–	–		There is no definition
0x1DEH	CAN message control register 13	CnMCTRL13		–	–		00x00000 000xx000B

Table 22-16. Register Access Type (9/9)

Offset address	Register name	Symbol	Read/ write	Bit operating unit			The default value
				1	8	16	
0x1E0H	CAN message data byte 01 register 14	CnMDB0114	Read/ write	–	–		There is no definition
0x1E0H	CAN message data byte 0 register 14	CnMDB014		–		–	There is no definition
0x1E1H	CAN message data byte 1 register 14	CnMDB114		–		–	There is no definition
0x1E2H	CAN message data byte 23 register 14	CnMDB2314		–	–		There is no definition
0x1E2H	CAN message data byte 2 register 14	CnMDB214		–		–	There is no definition
0x1E3H	CAN message data byte 3 register 14	CnMDB314		–		–	There is no definition
0x1E4H	CAN message data byte 45 register 14	CnMDB4514		–	–		There is no definition
0x1E4H	CAN message data byte 4 register 14	CnMDB414		–		–	There is no definition
0x1E5H	CAN message data byte 5 register 14	CnMDB514		–		–	There is no definition
0x1E6H	CAN message data byte 67 register 14	CnMDB6714		–	–		There is no definition
0x1E6H	CAN message data byte 6 register 14	CnMDB614		–		–	There is no definition
0x1E7H	CAN message data byte 7 register 14	CnMDB714		–		–	There is no definition
0x1E8H	CAN message data length register 14	CnMDLC14		–		–	0000xxxxB
0x1E9H	CAN message configuration register 14	CnMCONF14		–		–	There is no definition
0x1EAH	CAN message ID register 14	CnMIDL14		–	–		There is no definition
0x1ECH		CnMIDH14		–	–		There is no definition
0x1EEH	CAN message control register 14	CnMCTRL14		–	–		00x00000 000xx000B
0x1F0H	CAN message data byte 01 register 15	CnMDB0115		–	–		There is no definition
0x1F0H	CAN message data byte 0 register 15	CnMDB015		–		–	There is no definition
0x1F1H	CAN message data byte 1 register 15	CnMDB115		–		–	There is no definition
0x1F2H	CAN message data byte 23 register 15	CnMDB2315		–	–		There is no definition
0x1F2H	CAN message data byte 2 register 15	CnMDB215		–		–	There is no definition
0x1F3H	CAN message data byte 3 register 15	CnMDB315		–		–	There is no definition
0x1F4H	CAN message data byte 45 register 15	CnMDB4515		–	–		There is no definition
0x1F4H	CAN message data byte 4 register 15	CnMDB415		–		–	There is no definition
0x1F5H	CAN message data byte 5 register 15	CnMDB515		–		–	There is no definition
0x1F6H	CAN message data byte 67 register 15	CnMDB6715		–	–		There is no definition
0x1F6H	CAN message data byte 6 register 15	CnMDB615		–		–	There is no definition
0x1F7H	CAN message data byte 7 register 15	CnMDB715		–		–	There is no definition
0x1F8H	CAN message data length register 15	CnMDLC15		–		–	0000xxxxB
0x1F9H	CAN message configuration register 15	CnMCONF15		–		–	There is no definition
0x1FAH	CAN message ID register 15	CnMIDL15		–	–		There is no

							definition
0x1FCH		CnMIDH15		–	–		There is no definition
0x1FEH	CAN message control register 15	CnMCTRL15		–	–		00x00000 000xx000B

22.5.3 Register bit configuration

Table22-17. CAN Global Register Bit configuration

Offset address	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
0x000H	CnGMCTRL(W)	0	0	0	0	0	0	0	ClearGOM
0x001H		0	0	0	0	0	0	SET EFSD	SetGOM
0x000H	CnGMCTRL(R)	0	0	0	0	0	0	EFSD	Gather
0x001H		MBON	0	0	0	0	0	0	0
0x006H	CnGMABT(W)	0	0	0	0	0	0	0	ClearA BTTRG
0x007H		0	0	0	0	0	0	SetA BTCLR	SetAB TTRG
0x006H	CnGMABT(R)	0	0	0	0	0	0	ABTCLR	ABTTRG
0x007H		0	0	0	0	0	0	0	0
0x008H	CnGMABTD	0	0	0	0	ABTD3	ABTD2	ABTD1	ABTD0
0x002H	CnGMCS	0	0	0	0	CCP3	CCP2	CCP1	CCP0

BAT32A239: n=0.1 BAT32A279: n=0.1.2

Note The actual register address calculation refers to the following formula:

Register Address = Global Register Region Offset (CH Dependent) + Offset Address Listed in the table
above

Remarks: (R) When read
(W) When writing

Table 22-18. CAN Module Registers Bit configuration (1/2)

Offset address	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
0x040H	CnMASK1L	CM1ID[7:0]							
0x041H		CM1ID[15:8]							
0x042H	CnMASK1H	CM1ID[23:16]							
0x043H		0	0	0	CM1ID[28:24]				
0x044H	CnMASK2L	CM2ID[7:0]							
0x045H		CM2ID[15:8]							
0x046H	CnMASK2H	CM2ID[23:16]							
0x047H		0	0	0	CM2ID[28:24]				
0x048H	CnMASK3L	CM3ID[7:0]							
0x049H		CM3ID[15:8]							
0x04AH	CnMASK3H	CM3ID[23:16]							
0x04BH		0	0	0	CM3ID[28:24]				
0x04CH	CnMASK4L	CM4ID[7:0]							
0x04DH		CM4ID[15:8]							
0x04EH	CnMASK4H	CM4ID[23:16]							
0x04FH		0	0	0	CM4ID[28:24]				
0x050H	CnCTRL(W)	Clear CCERC	Clear AL	Clear VALID	ClearPS MODE1	ClearPS MODE0	ClearOP MODE2	ClearOP MODE1	ClearOP MODE0
0x051H		SetC CERC	SetAL	0	SetPSMO DE1	SetPSMO DE0	SetOPM ODE2	SetOPM ODE1	SetOPMO DE0
0x050H	CnCTRL(R)	CCERC	To the	VALID	PSMODE 1	PSMODE 0	OPMODE 2	OPMODE 1	OPMODE 0
0x051H		0	0	0	0	0	0	RSTAT	TSTAT
0x052H	CnLEC(W)	0	0	0	0	0	0	0	0
0x052H	CnLEC(R)	0	0	0	0	0	LEC2	LEC1	LEC0
0x053H	CnINFO	0	0	0	BOFF	TECS1	TECS0	RECS1	RECS0
0x054H	CnERC	TEC[7:0]							
0x055H		REPS	REC[7:0]						
0x056H	CnIE(W)	0	0	Clear CIE5	Clear CIE4	Clear CIE3	Clear CIE2	Clear CIE1	Clear CIE0
0x057H		0	0	SetCIE5	SetCIE4	SetCIE3	SetCIE2	SetCIE1	SetCIE0
0x056H	CnIE(R)	0	0	CIE5	CIE4	CIE3	CIE2	CIE1	CIE0
0x057H		0	0	0	0	0	0	0	0
0x058H	CnINTS(W)	0	0	Clear CINTS5	Clear CINTS4	Clear CINTS3	Clear CINTS2	Clear CINTS1	Clear CINTS0
0x059H		0	0	0	0	0	0	0	0

BAT32A239: n=0.1 BAT32A279: n=0.1.2

Note The actual register address calculation refers to the following formula:

Register Address = Global Register Region Offset (CH Dependent) + Offset Address Listed in the table
above

Remarks: (R) when read

(W) Write time

Table 22-18: CAN Module Registers Bit configuration (2/2)

Offset address	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
0x058H	CnINTS(R)	0	0	CINTS5	CINTS4	CINTS3	CINTS2	CINTS1	CINTS0
0x059H		0	0	0	0	0	0	0	0
0x05AH	CnBRP	TQPRS[7:0]							
0x05CH	CnBTR	0	0	0	0	TSEG1[3:0]			
0x05DH		0	0	SJW[1:0]		0	TSEG2[2:0]		
0x05EH	CnLIPT	LIPT[7:0]							
0x060H	CnRGPT(W)	0	0	0	0	0	0	0	Clear ROVF
0x061H		0	0	0	0	0	0	0	0
0x060H	CnRGPT(R)	0	0	0	0	0	0	RHPM	ROVF
0x061H		RGPT[7:0]							
0x062H	CnLOPT	LOPT[7:0]							
0x064H	CnTGPT(W)	0	0	0	0	0	0	0	Clear TOVF
0x065H		0	0	0	0	0	0	0	0
0x064H	CnTGPT(R)	0	0	0	0	0	0	THPM	TOVF
0x065H		TGPT[7:0]							
0x066H	CnTS(W)	0	0	0	0	0	ClearT SLOCK	Clear TSSEL	Clear TSEN
0x067H		0	0	0	0	0	SetT SLOCK	SetT SSEL	TSE N Set
0x066H	CnTS(R)	0	0	0	0	0	TSLOCK	TSSEL	TSEN
0x067H		0	0	0	0	0	0	0	0

BAT32A239: n=0.1 BAT32A279: n=0.1.2

Note The actual register address calculation refers to the following formula:

Register Address = Global Register Region Offset (CH Dependent) + Offset Address Listed in the table above

Remarks: (R) when read

(W) Write time

Table 22-19. Bit configuration of the message buffer registers

Offset address	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
0x1x0H	CnMDB01m	Message data (byte 0).							
0x1x1H		Message data (byte 1).							
0x1x0H	CnMDB0m	Message data (byte 0).							
0x1x1H	CnMDB1m	Message data (byte 1).							
0x1x2H	CnMDB23m	Message data (bytes 2).							
0x1x3H		Message data (byte 3).							
0x1x2H	CnMDB2m	Message data (bytes 2).							
0x1x3H	CnMDB3m	Message data (byte 3).							
0x1x4H	CnMDB45m	Message data (bytes 4).							
0x1x5H		Message data (byte 5).							
0x1x4H	CnMDB4m	Message data (bytes 4).							
0x1x5H	CnMDB5m	Message data (byte 5).							
0x1x6H	CnMDB67m	Message data (byte 6).							
0x1x7H		Message data (byte 7).							
0x1x6H	CnMDB6m	Message data (byte 6).							
0x1x7H	CnMDB7m	Message data (byte 7).							
0x1x8H	CnMDLCm	0	0	0	0	MDLC3	MDLC2	MDLC1	MDLC0
0x1x9H	CnMCONFm	OWS	Rtr	MT2	MT1	MT0	0	0	MA0
0x1xAH	CnMIDLm	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
0x1xBH		ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
0x1xCH	CnMIDHm	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
0x1xDH		IDE	0	0	ID28	ID27	ID26	ID25	ID24
0x1xEH	CnMCTRLm(W)	0	0	0	ClearMOW	ClearIE	ClearDN	ClearTRQ	ClearRDY
0x1xFH		0	0	0	0	SetIE	0	SetTRQ	SetRDY
0x1xEH	CnMCTRLm(R)	0	0	0	MOW	IE	DN	TRQ	RDY
0x1xFH		0	0	MUC	0	0	0	0	0

BAT32A239: n=0.1 BAT32A279: n=0.1.2

Note The actual register address calculation refers to the following formula:

Register Address = Global Register Region Offset (CH Dependent) + Offset Address Listed in the table above

- Note 1. (R) Read time
(W) Write time
2. m = 0 to 15
x = 0 to F

22.6 Bit setting/clear function

CAN control registers include registers whose bits can be set or cleared through the CPU and CAN interfaces.

If the following registers are written directly, an operation error occurs. Do not write any value directly through bitwise operations, read/modify/write, or directly write to the target value.

- CAN Global Control Register (CnGMCTRL).
- CAN Global Automatic Block Transfer Control Register (CnGMABT).
- CAN Module Control Register (CnCTRL).
- CAN Module Interrupt Enable Register (CnIE).
- CAN Module Interrupt Interrupt Status Register (CnINTS).
- CAN module receives the History List Register (CnRGPT).
- CAN module sends the History List Register (CnTGPT).
- CAN Module Timestamp Register (CnTS).
- CAN Message Control Register (CnMCTRLm).

Remark: m=0 to 15 BAT32A239: n=0.1 BAT32A279: n=0,1,2

All 16 bits in the above registers can be read by common methods. Use the procedure described in Figure22-23to set or clear the lower 8 bits in these registers.

The setting or clearance of the lower 8 bits in the above register is performed in conjunction with the higher 8 bits (see 16 bits of data after the write operation in Figure22-24). Figure22-23shows the value of the set or clear bit in relation to the set/clear/no change operation in the corresponding register.

Figure22-23. Bit setup/clearing operation example

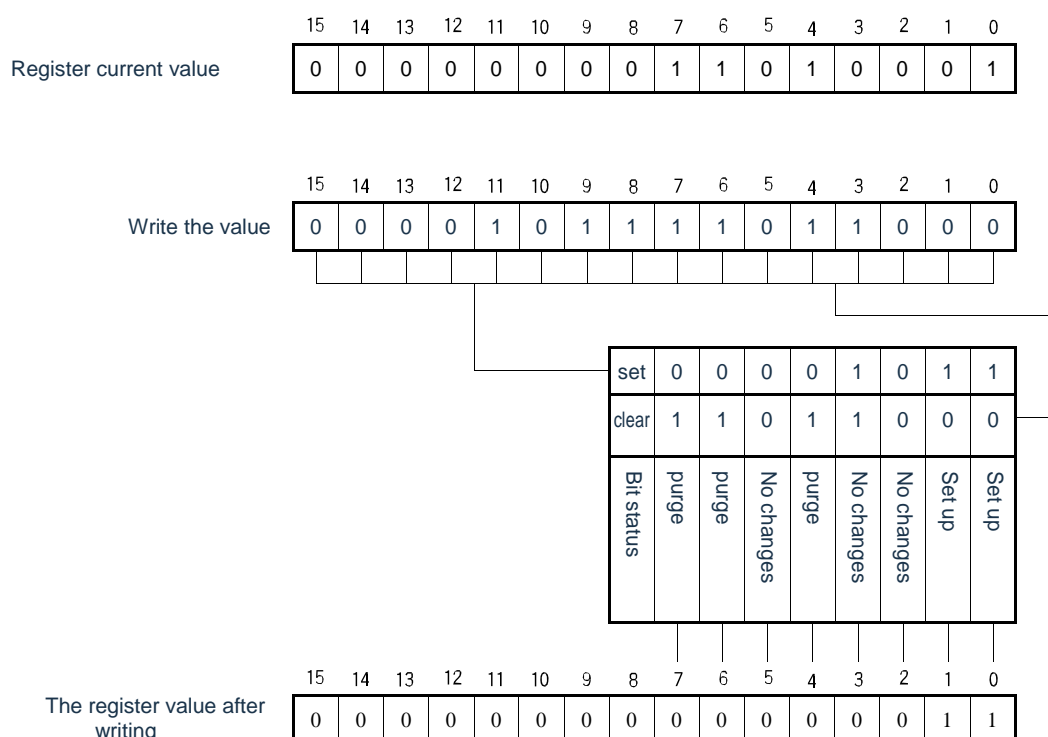


Figure22-24. Write 16-bit data in operations

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Set7	Set6	Set5	Set4	Set3	Set2	Set1	Set0	Clear7	Clear6	Clear5	Clear4	Clear3	Clear2	Clear1	Clear0

Setn	Clearn	n status after setting/clearing
0	0	No changes
0	1	0
1	0	1
1	1	No changes

Note n = 0 to 7

22.7 Control registers

Note m=0 to 15

22.7.1 Peripheral clock selection register (PER0/PER2).

The PER0/2 register is a register that is set to allow or disable clocking to each peripheral hardware.

Reduce power consumption and noise by stopping clocking hardware that is not in use.

To use the CAN function, theCANn EN must be placed at "1".

For details, see "4.3.8 Perimeter Allow Registers 0, 1, 2, 3 (PER0, PER1, PER2, PER3)"

BAT32A239: n=0.1 BAT32A279: n=0.1.2

Note: Before starting to operate each peripheral hardware device, set the PCKSEL registers.

22.7.2 CAN Global Module Control Register (CnGMCTRL).

The CnGMCTRL register is used to control the operation of the CAN module.

Figure22-25. CAN Global Module Control Register Format (CnGMCTRL) (1/2).

(a) read

	15	14	13	12	11	10	9	8
C0GMCTRL	MBON	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	EFSD	GOM

(a) write

	15	14	13	12	11	10	9	8
C0GMCTRL	0	0	0	0	0	0	Set EFSD	Set GOM
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	Clear GOM

(a) read

MBON	Message cache registers, transmit/receive bits of access to history list registers
0	Disables reads and writes to message cache registers and transmit/receive history list registers
1	Enables reads and writes to packet cache registers and transmit/receive history list registers

Note 1 When the MBON bit is cleared (to 0), the software accesses the packet buffer (CnMDB0m,CnMDB1m,CnMDB01m,CnMDB2m, CnMDB3m,CnMDB23m,CnMDB4m,CnMDB5m,CnMDB45m,CnMDB6m,CnMDB7m,CnMDB67m,CnMDLCm,CnMCONFm,CnMIDLm,CnMIDHm,andCnMCTRLm Or send history or receive history registers (CnLOPT, CnTGPT, CnLIPT, and CnRGPT) are disabled.

2. This bit is read-only, when MBON is 0 write to it 1, the value of MBON will not change, and access to the message cache register or register-related transmit history or receive history remains prohibited.

Note: When the CAN module enters CAN sleep mode/CAN stop mode or GOM bit clear (to 0), the MBON bit will be cleared (to 0).

When the CAN Sleep Mode/CAN Stop Mode is released or the GOM bit is set, the MBON bit is set to 1.

Figure 22-26. CAN Global Module Control Register Format (CnGMCTRL) (2/2).

EFSD	Bit enables force shutdown
0	It is forbidden to turn off by writing GOM=0
1	Enables to be turned off by writing GOM=0

Note To request a force shutdown, you must clear the GOM bit to 0 in subsequent operations and write immediately after the EFSD bit is set to 1. If an access to another register (including reading the CnGMCTRL register) is performed without immediately clearing the GOM bit after the EFSD bit is set to 1, the EFSD bit is forced to clear to 0 and the force close request is invalid.

When performing DMA, requests for forced shutdown may be ignored. Be sure to read the EFSD bits and confirm that force shutdown is enabled before issuing a forced shutdown request. If force shutdown cannot be enabled because DMA is being performed, it is recommended that you temporarily stop DMA.

Gather	Global operating mode bits
0	CAN module operation is disabled
1	The CAN module is started

Note: The GOM bit is cleared to 0 only after initialization mode or immediately after EFSD is set to 1.

(b) write

SET EFSD	EFSD bit settings
0	The EFSD bits have not changed
1	EFSD is set to 1

SetGOM	ClearGOM	GOM bit settings
0	1	The GOM bit is cleared to 0
1	0	The GOM bit is set to 1
Other values		The GOM bit is not changed

Note Set the GOM bit and the EFSD bit separately.

22.7.3 CAN Global Module Clock Selection Register (CnGMCS).

CnGMCS is used to select the system clock of the CAN module.

Figure 22-26: CAN Global Module Clock Selection Register Format (CnGMCS).

	7	6	5	4	3	2	1	0
CnGMCS	0	0	0	0	CCP3	CCP2	CCP1	CCP0

CCP3	CCP2	CCP1	CCP1	CAN Module System Clock (f_{CANMOD}).
0	0	0	0	$f_{CAN}/1$
0	0	0	1	$f_{CAN}/2$
0	0	1	0	$f_{CAN}/3$
0	0	1	1	$f_{CAN}/4$
0	1	0	0	$f_{CAN}/5$
0	1	0	1	$f_{CAN}/6$
0	1	1	0	$f_{CAN}/7$
0	1	1	1	$f_{CAN}/8$
1	0	0	0	$f_{CAN}/9$
1	0	0	1	$f_{CAN}/10$
1	0	1	0	$f_{CAN}/11$
1	0	1	1	$f_{CAN}/12$
1	1	0	0	$f_{CAN}/13$
1	1	0	1	$f_{CAN}/14$
1	1	1	0	$f_{CAN}/15$
1	1	1	1	$f_{CAN}/16$ (default).

Note f_{CAN} : The clock provided to CAN (f_{MAIN}).

22.7.4 CAN Global Automatic Block Transfer Control Register (CnGMABT).

The CnGMABT register is used to control automatic block transfer (ABT) operation

Figure 22-27 Format of the CAN Global Automatic Block Transfer Control Register (CnGMABT) (1/2).

(a) read

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

(a) write

	15	14	13	12	11	10	9	8
CnGMABT	0	0	0	0	0	0	Set ABTCLR	Set ABTTTRG
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	ABTCLR	ClearA BTTRG

Note Before changing the normal operating mode of ABT to initialization mode, make sure that the CnGMABT register is set to the default value (0000H) and confirm that the CnGMABT register must have been initialized to the default value (0000H).) .

(a) read

ABTCLR	Automatic block transfer clear status bits
0	Clearing the autotransport engine is complete
1	The autotransfer engine is being purged

Remarks

1. When ABTTTRG is cleared to 0, set the ABTCLR bit to 1.
Operation is not guaranteed when ABTTTRG is 1 and ABTCLR is 1.
2. When the autoblock transfer engine is cleared by setting the ABTCLR bit to 1, the ABTCLR bit will automatically clear to 0 once the requested purge processing is complete.

Figure 22-28. Format of the CAN Global Automatic Block Transfer Control Register (CnGMABT) (2/2).

ABTTRG	Automatic block transfer status bits
0	Automatic block transfer was stopped
1	Automatic block transfer is being performed

Note Do not set the ABTTRG bit (ABTTRG=1) in initialization mode. If the ABTTRG bit is set in initialization mode, operation cannot be guaranteed after the CAN module enters normal operation mode using ABT. In cnCTRL. When TSTAT is 1, do not set the ABTTRG bit (1). Before setting up ABTTRG, confirm TSTAT=0 in advance.

(b) write

SetABTCLR	The automatic block transfer engine clears the request bits
0	The automatic block transfer engine is idle or in operation
1	Request to clear Automatic Block Transfer. When the automatic block transfer engine is cleared, automatic block transfer starts from the packet cache 0 after setting ABTTRG to 1

SetABTTRG	ClearABTTRG	Automatic block transfer start bit
0	1	Request to stop automatic block transfer
1	0	Request to start automatic block transfer
Other values		The ABTTRG bit does not change

Note: When receiving packets from other nodes or sending packets other than ABT packets (packet buffers 8 to 15), transmission may not start immediately even if the ABTTRG bit is set to 1.

Even if the ABTTRG bit is cleared to 0, the transfer is not aborted until the currently transmitted ABT packet transmission is complete (successful or not). After that, the transfer is aborted.

22.7.5 CAN Global Automatic Block Delay Setting Register (CnGMABTD).

The CnGMABTD register is used to set the time interval at which packet buffer data is allocated to ABT and transmitted in ABT normal operating mode.

Figure 22-28 CAN Global Automatic Block Propagation Delay Setting Register Format (CnGMABTD).

	7	6	5	4	3	2	1	0
CnGMABTD	0	0	0	0	ABTD3	ABTD2	ABTD1	ABTD0

ABTD3	ABTD2	ABTD1	ABTD0	Data frame interval (unit: Data Bit Time (DBT)) for automatic block transfer
0	0	0	0	0 DBT (default).
0	0	0	1	2^5 DBT
0	0	1	0	2^6 DBT
0	0	1	1	2^7 DBT
0	1	0	0	2^8 DBT
0	1	0	1	2^9 DBT
0	1	1	0	2^{10} DBT
0	1	1	1	2^{11} DBT
1	0	0	0	2^{12} DBT
Other values				forbid

Note 1 When ABTTRG is 1, do not change the contents of cngMABTD.

- The time at which ABT packets are actually transmitted to the CAN bus varies depending on the status of the transmission from other workstations or the way in which ABT packets (packet buffers 8 to 15) are requested to be transmitted.

22.7.6 CAN module mask registers (CnMASKaL, CnMASKaH) (a=1, 2, 3, or 4).

The CnMASKaL and CnMASKaH registers expand the number of message caches that enter the same packet cache by comparing the masked portion of the packet ID and invalidating the ID of the masked portion.

Figure 22-29 Format of the CAN module mask register (CnMASKaL, CnMASKaH) (a=1, 2, 3, or 4) (1/2).

- CAN module shield 1 register (CnMASK1L, CnMASK1H).

	15	14	13	12	11	10	9	8
C0MASK1L	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
	7	6	5	4	3	2	1	0
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0

	15	14	13	12	11	10	9	8
C0MASK1H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
	7	6	5	4	3	2	1	0
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16

-CAN module shield 2 registers (CnMASK2L, CnMASK2H).

	15	14	13	12	11	10	9	8
C0MASK2L	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
	7	6	5	4	3	2	1	0
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0

	15	14	13	12	11	10	9	8
C0MASK2H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
	7	6	5	4	3	2	1	0
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16

Fig.22-29. Format of the CAN module mask register(CnMASKaL,CnMASKaH)(a=1,2,3,or4)(2/2)

- CAN module shield 3 registers (CnMASK3L, CnMASK3H).

	15	14	13	12	11	10	9	8
C0MASK3L	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
	7	6	5	4	3	2	1	0
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0

	15	14	13	12	11	10	9	8
C0MASK3H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
	7	6	5	4	3	2	1	0
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16

-CAN module shield 4 registers (CnMASK4L, CnMASK4H).

	15	14	13	12	11	10	9	8
C0MASK4L	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
	7	6	5	4	3	2	1	0
	CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0

	15	14	13	12	11	10	9	8
C0MASK4H	0	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
	7	6	5	4	3	2	1	0
	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16

CMID0 to CMID28	Sets the masking mode for the ID bit
0	The ID bits of the packet buffer set by CMID0 through CMID28 are compared to the ID bits of the received packet frames.
1	The ID bits of the packet buffer set by CMID0 through CMID28 are not compared to the ID bits of the received packet frames (they are masked)

Note: Masking is always determined by the ID length of 29 bits. If the mask is assigned to a message with a standard ID, cmID 0 to CMID 17 bits are ignored; Therefore, only CMID18 to CMID28 of the received ID is blocked. The same mask can be used for both standard and extended IDs.

22.7.7 CAN Module Control Register (CnCTRL).

The CnCTRL registers are used to control the operating mode of the CAN module.

Figure 22-30 CAN Module Control Registers (CnCTRL) (1/4).

(a) read

	15	14	13	12	11	10	9	8
C0CTRL	0	0	0	0	0	0	RSTAT	TSTAT
	7	6	5	4	3	2	1	0
	CCERC	AL	VALID	PSMODE1	PSMODE0	OPMODE2	OPMODE1	OPMODE0

(b) write

	15	14	13	12	11	10	9	8
C0CTRL	Set CCERC	Set AL	0	Set PSMODE1	Set PSMODE0	Set OPMODE2	Set OPMODE1	Set OPMODE0
	7	6	5	4	3	2	1	0
	Clear CCERC	Clear AL	Clear VALID	ClearP SMODE1	ClearP SMODE0	ClearO PMODE2	ClearO PMODE1	ClearO PMODE0

(a) read

RSTAT	Receive status bits
0	The receive is stopped
1	Receive is in progress

Note - The RSTAT bit is set to 1 under the following conditions (timing).

- The SOF bit of the received frame is detected
- Arbitration loss occurs in the transmission frame
- The RSTAT bit is cleared to 0 under the following conditions (timing).
 - The second bit in interframe space detects recessive bits
 - The transition to the initial mode, the first bit in the inter-frame space

Figure 22-30. Format of the CAN Module Control Register (CnCTRL) (2/4).

TSTAT	Transfer status bits
0	The transfer is stopped
1	The transfer is in progress

Note - The RSTAT bit is set to 1 under the following conditions (timing).

- The SOF bit of the sending frame is detected
- The RSTAT bit is cleared to 0 under the following conditions (timing).
 - During the transition to the bus shutdown state
 - Arbitration loss occurs in the transmission frame
 - The second bit in the interframe space detects the recessive bit
 - The transition to the initial mode occurs at the first bit of the inter-frame space

CCERC	Error counter clear bits
0	In initialization mode the CnERC and CnINFO registers are not cleared
1	The CnERC and CnINFO registers are cleared in initialization mode

Note 1. The CCERC bits are used to clear the CnERC and CnINFO registers for

reinitialization or force recovery from the bus shutdown state. This bit can only be set to 1 in initialization mode.

2. After clearing the CnERC and CnINFO registers, the CCERC bit is also automatically cleared to 0.

3. When requesting a change from initialization mode to operating mode, the CCERC bit is set to 1.

4. When the CCERC bit is set to 1 immediately after entering INIT mode from self-test mode, the received data may be corrupted.

To the	Sets the bit of the action when the Arbitration is lost
0	When a Arbitration loss occurs in single-time mode, no retransmission is performed
1	Perform a retransmission when an arbitral loss occurs in single-shot mode

Note The AL bit is only valid in single-pass mode.

VALID	Valid receive packet frame detection bits
0	Since the last time the VALID bit was cleared to 0, no valid message frames have been received
1	After clearing the VALID bit to 0, a valid message frame is received

Note 1 Detecting a valid receive packet frame does not depend on storage in the

receive packet buffer (data frame) or the transmit packet buffer (remote frame).

2. Clear the VALID bit before changing the initialization mode to operating mode

3. If only two CAN nodes are connected to the CAN bus, one transmits a message frame in normal operating mode and the other is in receive-only mode, the VALID bit will not be set to 1 until the transmitting node enters a passive error state, because no reply is generated in receive-only mode.

4. In order to clear the VALID bit, first set the ClearVALID bit to 1, then confirm that the VALID bit is cleared, and if it is still not cleared, perform the cleanup process again.

Figure 22-30 Format of CAN Module Control Register (C0CTRL) (3/4)

PSMODE1	PSMODE0	Power-saving mode
0	0	No power saving mode is selected
0	1	CAN sleep mode
1	0	Set Prohibited
1	1	CAN stop mode

Note 1 Transitions to or wake up from CAN stop mode are required through CAN sleep mode. Requests to go in and out of the CAN stop mode directly are ignored.

- After exiting the power saving mode, the MBON flag of the CnGMCTRL must be checked before the message buffer is accessed.
- The request for CAN sleep mode remains pending until the software cancels or enters the appropriate bus condition (bus idle). The software can check the actual status by reading the PSMODE

OPMODE2	OPMODE1	OPMODE0	Mode of operation
0	0	0	There is no operating mode selected (CAN mode is in initialization mode).
0	0	1	Normal mode of operation
0	1	0	There is a normal operating mode of automatic block transfer
0	1	1	Receive mode only
1	0	0	Single-shot mode
1	0	1	Self-test mode
Other values			Prohibit settings

Note: It may take some time to move to initialization mode or power saving mode. Be sure to verify that the schema change was successful by reading before continuing.

Note OpMODE [2:0] bits in CAN sleep and CAN stop modes are read-only.

(b) Write

SetCCERC	Set the CCERC bit
1	Set the CCERC bit to 1
0	CCERC remains

SetAL	ClearAL	Set the AL bit
0	1	AL is cleared 0
1	0	AL is set to 1
Other values		The AL remains unchanged

Figure 22-30. Format of can module control registers (CnCTRL) (4/4).

ClearVALID	Sets the VALID bit	
0	VALID remains	
1	VALID is cleared 0.	

SetPSMODE0	ClearPSMODE0	Set the PSMODE0 bit
0	1	PSMODE0 is cleared 0
1	0	PSMODE0 is set to 1
Other values		PSMODE0 does not change

SetPSMODE1	ClearPSMODE1	Set the PSMODE1 bit
0	1	PSMODE1 clear 0
1	0	PSMODE1 is set to 1
Other values		PSMODE1 does not change

SetOPMODE0	ClearOPMODE0	Set the OPMODE 0 bit
0	1	OPMODE0 clear 0
1	0	OPMODE0 1
Other values		OPMODE0 does not change

SetOPMODE1	ClearOPMODE1	Set the OPMODE1 bit
0	1	OPMODE1 clear 0
1	0	OPMODE1 is set to 1
Other values		OPMODE1 does not change

SetOPMODE2	ClearOPMODE2	Set the OPMODE2 bit
0	1	OPMODE2 clear 0
1	0	OPMODE2 set 1
Other values		OPMODE2 does not change

22.7.8 CAN Module Last Error Code Register (CnLEC).

The CnLEC register provides error messages for the CAN protocol.

Figure22-31. Format of the LAST Error Code Register for can modules (CnLEC).

	7	6	5	4	3	2	1	0
C0LEC	0	0	0	0	0	LEC2	LEC1	LEC0

Note 1 When CAN switches from operating mode to initialization mode, the contents of the CnLEC register are not cleared

2. If you try to write a value other than 00H to the CnLEC register through the software, the access will be ignored.

LEC2	LEC1	LECn	Last CAN protocol error message
0	0	0	No errors
0	0	1	Fill errors
0	1	0	Malformation
0	1	1	ACK error
1	0	0	Bit error (the CAN module attempts to transmit recessive bits as part of the transmitted message (except for the Arbitration field), but the value on the CAN bus is the dominant bit)
1	0	1	Bit error (the CAN module attempted to transmit the dominant bit as part of the transmitted message, ACK bit, error frame, or overloaded frame, but the value on the CAN bus is the recessive bit).
1	1	0	CRC error
1	1	1	There is no definition

22.7.9 CAN Module Information Register (CnINFO).

The CnINFO register indicates the status of the CAN module

Figure22-32. Format of the CAN Module Information Register (CnINFO).

	7	6	5	4	3	2	1	0
C0INFO	0	0	0	BOFF	TECS1	TECS0	RECS1	RECS0

BOFF	Bus shutdown status bit
0	Non-bus shutdown state (transmission error counter less than 255) (transmission counter value less than 256).
1	Bus shutdown status (transmission error counter greater than 255) (transmission counter value equal to or greater than 256).

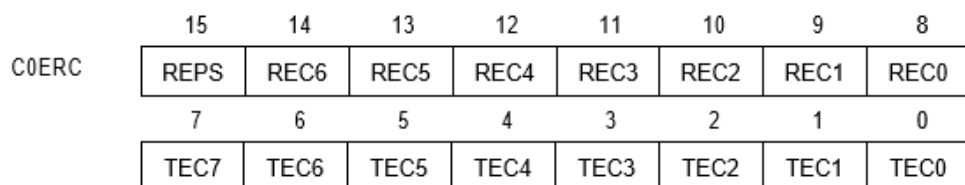
TECS1	TECS0	Send error counter status bits
0	0	The value of the Send Error counter is less than the warning level (<96).
0	1	Send the value of the error counter in the warning range (96 to 127).
1	0	There is no definition
1	1	Sends the value of the error counter within the passive error or bus shutdown state

RECS1	RECS0	Receive error counter status bits
0	0	The value of the received error counter is less than the warning level (<96)
0	1	The value of the receive error counter is in the warning range (96 to 127).
1	0	There is no definition
1	1	The value of the receive error counter is in the passive error range (128).

22.7.10 CAN Module Error Counter Register (CnERC).

The CnERC register records the count value of the transmit/receive error counter

Figure 22-33. CAN Module Error Counter Register Format (CnERC).



REPS	Receives passive error status bits
0	The Receive Error counter is not a passive error (<128).
1	The receive error counter is within the passive error range (128).

REC6-RECN	Receive error counter bits
0-127	The number of errors received. These bits reflect the status of the receive error counter. The number of errors is defined by the CAN protocol

Note: The REC [6:0] of the receive error counter is not valid in the receive passive error state (RECS[1:0]=11B).

TEC7-TECN	Send error counter bits
0-255	The number of transmission errors. These bits reflect the status of the transfer error counter. The number of errors is defined by the CAN protocol.

Note: The TEC [7:0] that sends the error counter is not valid in the bus shutdown state (BOFF=1).

22.7.11 CAN Module Interrupt Enable Register (CnIE).

CnIE registers are used to enable or disable interrupts to the CAN module.

Figure 22-34. CAN Module Interrupt Enable Register Format (CnIE) (1/2).

(a) read

	15	14	13	12	11	10	9	8
C0IE	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	0	0	CIE5	CIE4	CIE3	CIE2	CIE1	CIE0

(b) write

	15	14	13	12	11	10	9	8
C0IE	0	0	Set CIE5	Set CIE4	Set CIE3	Set CIE2	Set CIE1	Set CIE0
	7	6	5	4	3	2	1	0
	0	0	Clear CIE5	Clear CIE4	Clear CIE3	Clear CIE2	Clear CIE1	Clear CIE0

(a) read

CIE5-CIE0	CAN module interrupt enable bit
0	The interrupt output corresponding to the cnINTS [5:0] bit of the interrupt status register is disabled
1	The interrupt output corresponding to the cnINTS [5:0] bit of the interrupt status register is enabled

(b) write

SetCIE5	ClearCIE5	Set the CIE5 bit
0	1	CIE5 clear 0
1	0	CIE5 set 1
Other values		CIE5 has not changed

SetCIE4	ClearCIE4	Set the CIE4 bit
0	1	CIE4 bit clear 0
1	0	CIE4 Position 1
Other values		CIE4 has not changed

Figure 22-35. CAN Module Interrupt Enable Register Format (CnIE) (2/2).

SetCIE3	ClearCIE3	Set the CIE3 bit
0	1	CIE3 clear 0
1	0	CIE3 set 1
Other values		CIE3 has not changed

SetCIE2	ClearCIE2	Set the CIE2 bit
0	1	CIE2 clear 0
1	0	CIE2 set 1
Other values		CIE2 has not changed

SetCIE1	ClearCIE1	Set the CIE1 bit
0	1	CIE1 clear 0
1	0	CIE1 to 1
Other values		CIE1 has not changed

SetCIE0	ClearCIE0	Set the CIE0 bit
0	1	CIE0 clear 0
1	0	CIE0 to 1
Other values		CIE0 has not changed

22.7.12 CAN Module Interrupt Status Register (CnINTS).

The CnINTS register indicates the CAN module interrupt status

Figure 22-35. CAN Module Interrupt Status Register Format (CnINTS).

(a) read

	15	14	13	12	11	10	9	8
C0INTS	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	0	0	CINTS5	CINTS4	CINTS3	CINTS2	CINTS1	CINTS0

(b) write

	15	14	13	12	11	10	9	8
C0INTS	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	0	0	Clear CINTS5	Clear CINTS4	Clear CINTS3	Clear CINTS2	Clear CINTS1	Clear CINTS0

(a) read

CINTS5-CINTS0	CAN interrupt status bit
0	No related interrupt source events are pending
1	A related interrupt source event is pending

Interrupt StatusBit	Related interrupt source events
CINTS5	Wake up ¹ from CAN sleep mode
CINTS4	Arbitration lost interrupt
CINTS3	CAN protocol error interrupted
CINTS2	The CAN error status is interrupted
CINTS1	A valid packet frame receives a completion interrupt to the packet buffer m
CINTS0	Interrupt of packet frame transmission from message buffer m normally completed

1 The CINTS5 bit is set only when the CAN module wakes up from CAN sleep mode via CAN bus operation. When the software releases the CAN sleep mode, the CINTS5 bit is not set.

(b) write

Clear CINTS5-CINTS0	CINTS0 to CINTS5 bits
0	CINTS0 to CINTS5 has not changed
1	CINTS0 to CINTS5 is cleared 0

Note: When each state needs to be acknowledged during interrupt processing, clear the status bits of this register with the software, as these bits are not automatically cleared.

22.7.13 CAN Module Bit Rate Scaling Register (CnBRP).

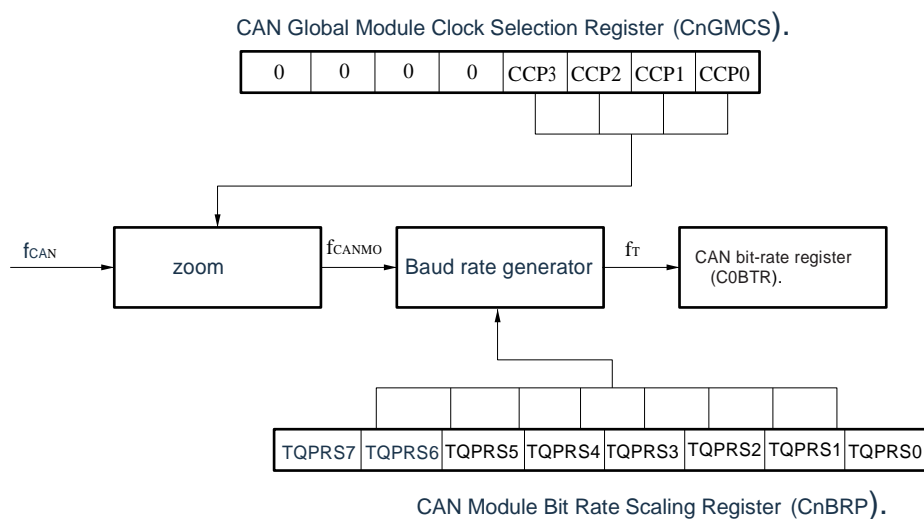
The CnBRP register is used to select the CAN protocol layer base clock (fTQ). The baud rate is set to the CnBTR register

Figure 22-36. CAN Module Bit Rate Scaling Register Format (CnBRP).

	7	6	5	4	3	2	1	0
C0BRP	TQPRS7	TQPRS6	TQPRS5	TQPRS4	TQPRS3	TQPRS2	TQPRS1	TQPRS0

TQPRS7-TQPRS0	CAN protocol layer basic system clock (fTQ).
0	fCANMOD/1
1	fCANMOD/2
:	:
n	fCANMOD/(n+1)
:	:
255	fCANMOD/256 (default).

Figure 22-37. CAN global clock



Note: The CnBRP register can only be written in initialization mode

Note: fCAN: Provides a clock (fMAIN) to CAN

fCANMOD: CAN module system clock

fTQ: CAN protocol layer base system clock

22.7.14 CAN Module Bit Rate Register (CnBTR).

The CnBTR register is used to control the data bit time of the baud rate.

Figure 22-38. CAN Module Bit Rate Register Format (CnBTR) (1/2).

	15	14	13	12	11	10	9	8
C0BTR	0	0	SJW1	SJW0	0	TSEG22	TSEG21	TSEG20
	7	6	5	4	3	2	1	0
	0	0	0	0	TSEG13	TSEG12	TSEG11	TSEG10

SJW1	SJW0	Synchronize the length of the jump width
0	0	1TQ
0	1	2TQ
1	0	3TQ
1	1	4TQ (default).

TSEG22	TSEG21	TSEG20	The length of the time period 2
0	0	0	1TQ
0	0	1	2TQ
0	1	0	3TQ
0	1	1	4TQ
1	0	0	5TQ
1	0	1	6TQ
1	1	0	7TQ
1	1	1	8TQ (default).

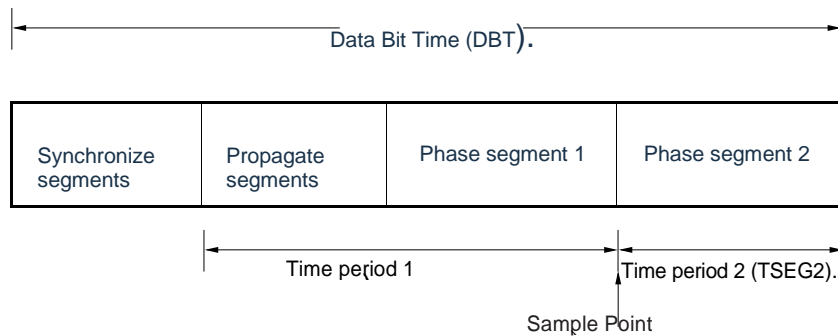
Fig.22-38. CAN module bit rate register format(CnBTR)(2/2)

TSEG13	TSEG12	TSEG11	TSEG10	The length of the time period 1
0	0	0	0	Prohibit settings
0	0	0	1	2TQ ¹
0	0	1	0	3TQ ¹
0	0	1	1	4TQ
0	1	0	0	5TQ
0	1	0	1	6TQ
0	1	1	0	7TQ
0	1	1	1	8TQ
1	0	0	0	9TQ
1	0	0	1	10TQ
1	0	1	0	11TQ
1	0	1	1	12TQ
1	1	0	0	13TQ
1	1	0	1	14TQ
1	1	1	0	15TQ
1	1	1	1	16TQ (default).

1 These settings must be performed when the CnBRP register is 00H

Note TQ=1/f_{TQ} (f_{TQ}: CAN protocol layer base system clock).

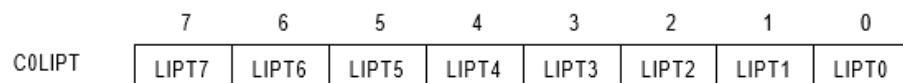
Figure 22-39. Data bit time



22.7.15 CAN module last entered the pointer register (CnLIPT).

The CnLIPT register indicates the number of message buffers that last stored a data frame or remote frame.

Figure 22-40. CAN module last entered the pointer register format (CnLIPT).



LIPT7-LIPT0	Last Input Pointer Register (CnLIPT).
0 to 15	When the CnLIPT register is read, the contents of the element indexed by the last input pointer (LIPT) of the received history list are read. These indicate the number of message buffers where the data frame or remote frame was last stored

Note: If the data frame or remote frame is never stored in the message buffer, the read value of the CnLIPT register is undefined. If the RHPM bit of the CnRGPT register is set to 1 after the CAN module changes from initialization mode to operating mode, the read value of the CnLIPT register is undefined.

22.7.16 CAN module receive History List Register (CnRGPT).

The CnRGPT register is used to read the received history list.

Figure 22-41. CAN module receives the History List Register Format (CnRGPT) (1/2).

(a) read

	15	14	13	12	11	10	9	8
CnRGPT	RGPT7	RGPT6	RGPT5	RGPT4	RGPT3	RGPT2	RGPT1	RGPT0
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	RHPM	ROVF

(b) write

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

(a) read

RGPT7-RGPT0	Receives a list of history to get a pointer
0 to 15	When the CnRGPT register is read, the contents of the index element obtained by receiving the history list (RGPT) are read. These indicate the number of message buffers that store data frames or remote frames.

RHPM ₁	Receives history list pointer matches
0	The receive history list has at least one packet buffer number that has not yet been read
1	The receive history list does not have a packet buffer number that has not yet been read

1 When the RHPM is 1, the reading values from RGPT0 to RGPT7 are invalid

ROVF ¹	Receives history list overflow bits
0	All packet buffer numbers that have not yet been read are retained. The packet cache numbers received and stored in all new data frames or remote frames have been recorded to the receive history list (the receive history list has an empty element)
1	At least 23 entries have been stored since the host processor last served THE RHL (i.e. read the CnRGPT). The first 22 entries are stored sequentially, and the last entry can be overwritten whenever a newly received message is stored, because when the ROVF bit is set, all buffer numbers are stored in the LIPT-1 position. Therefore, the order received cannot now be fully restored.

1 If ROVF is set, RHPM is no longer cleared when message storage is set, but if the software reads all entries of cnRGPT, RHPM is still set

Figure 22-41. CAN module receives the History List Register Format (CnRGPT) (2/2).

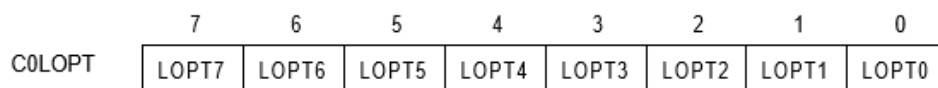
(b) write

ClearROVF	Set the ROVF bit
0	ROVF does not change
1	ROVF clear 0

22.7.17 CAN module last output pointer register (CnLOPT).

The CnLOPT register indicates the number of message buffers to which a data frame or remote frame was last transmitted.

Figure 22-42. CAN module last outputs pointer register format (CnLOPT).



LOPT7-LOPT0	Send History List Last Output Pointer (LOPT).
0 to 15	When the CnLOPT register is read, the contents of the element indexed by the last out pointer (LOPT) of the received history list are read. These indicate the number of message buffers to which the data frame or remote frame was last transmitted

Note: If a data frame or remote frame is never transmitted from the packet buffer, the value read from the CnLOPT register is undefined. If the THPM bit is set to 1 after the CAN module changes from initialization mode to operating mode, the read value of the CnLOPT register is undefined.

22.7.18 CAN module send History List Register (CnTGPT).

The CnTGPT register is used to read out the list of transmission histories.

Figure 22-43. CAN module sends history list register format (CnTGPT) (1/2).

(a) read

	15	14	13	12	11	10	9	8
C0TGPT	TGPT7	TGPT6	TGPT5	TGPT4	TGPT3	TGPT2	TGPT1	TGPT0
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	THPM	TOVF

(b) write

	15	14	13	12	11	10	9	8
C0TGPT	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	Clear TOVF

(a) read

TGPT7-TGPT0	Sends a history list read pointer
0 to 15	When the CnTGPT register is read, the contents of the read pointer (TGPT) index element of the send history list are read. These indicate the number of message buffers to which the data frame or remote frame was last transmitted

THPM ¹	Transmit History PointerMatch
0	The send history list has at least one packet buffer number that has not yet been read
1	Sends a history list to no packet buffer number that has not yet been read

1 When the THPM is 1, the reading values of TGPT0 to TGPT7 are invalid

TOVF	Send history list overflow bit
0	All packet buffer numbers that have not yet been read are retained. All numbers of the message buffer to which the new data frame or remote frame is transmitted are recorded to the transmission history list (the transmission history list has an empty element).
1	At least 7 entries (i.e., read CnTGPT) have been stored since the host processor last served THL. The first 6 entries are stored sequentially, and the last entry can be overwritten whenever a new packet is transmitted, because when the TOVF bit is set, all buffer numbers are stored in the LOPT-1 position. Therefore, the transfer sequence cannot now be fully recovered.

1. If TOVF is set, THPM is no longer cleared during message transmission, but if the software reads all entries of CTGPT, THPM is still set.

Note: In normal operating mode with ABT, the transmission history of packet buffer 0 to 7 is not put into the list

Figure22-43. CAN module sends history list register format (CnTGPT) (2/2).

(b) write

ClearTOVF	Set the TOVF bit
0	TOVF has not changed
1	TOVF clear 0

22.7.19 CAN Module Timestamp Register (CnTS).

CnTS registers are used to control the timestamp function

Figure22-44. CAN Module Timestamp Register Format (CnTS) (1/2).

(a) read

	15	14	13	12	11	10	9	8
C0TS	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	0	0	0	0	0	TSLOCK	TSSEL	TSEN

(b) write

	15	14	13	12	11	10	9	8
C0TS	0	0	0	0	0	Set TSLOCK	Set TSSEL	Set TSEN
	7	6	5	4	3	2	1	0
	0	0	0	0	0	ClearT SLOCK	Clear TSSEL	Clear TSEN

Note: When the CAN module is in normal operating mode with ABT, the lock function of the timestamp function must not be used.

(a) read

TSLOCK	Timestamp lock function enables bits
0	The timestamp lock feature stops Each time the selected timestamp capture event occurs, the TSOUT signal is toggled
1	The timestamp lock feature enables Each time the selected timestamp capture event occurs, the TSOUT signal is toggled. However, when the data frame is correctly received packet buffer 0, the TSOUT output signal is locked ¹

¹ TSEN bit is automatically cleared to 0

Fig.22-44. CAN module timestamp register format(CnTS)(2/2)

TSSEL	The timestamp capture event selects the bit
0	Timestamp capture events in SOF
1	The timestamp captures the last bit of the event in the EOF

TSEN	TSOUT signal operation setting bit
0	TSOUT signal flipping operation is prohibited
1	TSOUT signal flip operation enables

Note: The signal TSOUT is output from the CAN macro to the timer resource, depending on the implementation.
See Chapter 6, Universal Timer Units

(b) write

SetTSLOCK	ClearTSLOCK	TSLOCK setting bit
0	1	TSLOCK clear 0
1	0	TSLOCK set 1
Other values		TSLOCK does not change

SetTSSEL	ClearTSSEL	TSSEL setting bit
0	1	TSSEL clear 0
1	0	TSSEL set 1
Other values		TSSEL does not change

SetTSEN	ClearTSEN	TSEN setting bit
0	1	TSEN clear 0
1	0	TSEN set 1
Other values		TSEN does not change

22.7.20 CAN message data byte register (CnMDBxm) (x=0 to 7), (CnMDBzm) (z=01, 23, 45, 67).

CnMDBxm, CnMDBzm registers are used to store data for sending/receiving messages. The CnMDBxm register can be accessed in 8-bit units. The CnMDBzm registers provide access to the CnMDBxm registers of the 16-bit cell

Figure 22-45. CAN message data byte register format (CnMDBxm) (x=0 to 7), (CnMDBzm) (z=0,1,2,3,4,5,6,7) (1/2)

Reset value: No R/W is defined

- CnMDBxm register

	7	6	5	4	3	2	1	0
CnMDB0m	MDATA07	MDATA06	MDATA05	MDATA04	MDATA03	MDATA02	MDATA01	MDATA00
	7	6	5	4	3	2	1	0
CnMDB1m	MDATA17	MDATA16	MDATA15	MDATA14	MDATA13	MDATA12	MDATA11	MDATA10
	7	6	5	4	3	2	1	0
CnMDB2m	MDATA27	MDATA26	MDATA25	MDATA24	MDATA23	MDATA22	MDATA21	MDATA20
	7	6	5	4	3	2	1	0
CnMDB3m	MDATA37	MDATA36	MDATA35	MDATA34	MDATA33	MDATA32	MDATA31	MDATA30
	7	6	5	4	3	2	1	0
CnMDB4m	MDATA47	MDATA46	MDATA45	MDATA44	MDATA43	MDATA42	MDATA41	MDATA40
	7	6	5	4	3	2	1	0
CnMDB5m	MDATA57	MDATA56	MDATA55	MDATA54	MDATA53	MDATA52	MDATA51	MDATA50
	7	6	5	4	3	2	1	0
CnMDB6m	MDATA67	MDATA66	MDATA65	MDATA64	MDATA63	MDATA62	MDATA61	MDATA60
	7	6	5	4	3	2	1	0
CnMDB7m	MDATA77	MDATA76	MDATA75	MDATA74	MDATA73	MDATA72	MDATA71	MDATA70

Note m = 0 to 15

Figure 22-46. CAN message data byte register format (CnMDBxm) (x = 0 to 7),
(CnMDBzm) (z = 01, 23, 45, 67) (2/2)

-CnMDBzm register

	15	14	13	12	11	10	9	8
CnMDB01m	MDATA 0115	MDATA 0114	MDATA 0113	MDATA 0112	MDATA 0111	MDATA 0110	MDATA 019	MDATA 018
	7	6	5	4	3	2	1	0
	MDATA 017	MDATA 016	MDATA 015	MDATA 014	MDATA 013	MDATA 012	MDATA 011	MDATA 010
	15	14	13	12	11	10	9	8
CnMDB23m	MDATA 2315	MDATA 2314	MDATA 2313	MDATA 2312	MDATA 2311	MDATA 2310	MDATA 239	MDATA 238
	7	6	5	4	3	2	1	0
	MDATA 237	MDATA 236	MDATA 235	MDATA 234	MDATA 233	MDATA 232	MDATA 231	MDATA 230
	15	14	13	12	11	10	9	8
CnMDB45m	MDATA 4515	MDATA 4514	MDATA 4513	MDATA 4512	MDATA 4511	MDATA 4510	MDATA 459	MDATA 458
	7	6	5	4	3	2	1	0
	MDATA 457	MDATA 456	MDATA 455	MDATA 454	MDATA 453	MDATA 452	MDATA 451	MDATA 450
	15	14	13	12	11	10	9	8
CnMDB67m	MDATA 6715	MDATA 6714	MDATA 6713	MDATA 6712	MDATA 6711	MDATA 6710	MDATA 679	MDATA 678
	7	6	5	4	3	2	1	0
	MDATA 677	MDATA 676	MDATA 675	MDATA 674	MDATA 673	MDATA 672	MDATA 671	MDATA 670

Note: m= 0 to 15

22.7.21 CAN message data length register m (CnMDLCm).

The CnMDLCm register is used to set the number of bytes of the data segment of the message buffer

Figure 22-46 CAN message data length register m format (CnMDLCm).

Reset value: 0000xxxxB R/W

	7	6	5	4	3	2	1	0
CnMDLCm	0	0	0	0	MDLC3	MDLC2	MDLC1	MDLC0

MDLC3	MDLC2	MDLC1	MDLC0	The length of the data sent/received packet
0	0	0	0	0 bytes
0	0	0	1	1 byte
0	0	1	0	2 bytes
0	0	1	1	3 bytes
0	1	0	0	4 bytes
0	1	0	1	5 bytes
0	1	1	0	6 bytes
0	1	1	1	7 bytes
1	0	0	0	8 bytes
1	0	0	1	Set Prohibited (If these bits are set during transmission, 8 bytes of data are transferred, regardless of the DLC value set when the data frame was transmitted.) However, the DLC that is actually transmitted to the CAN bus is the DLC value set to this register) ¹
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

¹The data and DLC values actually transmitted to the CAN bus are as follows.

The type of frame sent	The length of the data sent	DLC transmission
Data frames	The number of bytes specified by the DLC (in any case, 8 bytes if	MDLC[3:0]
Remote frames	0 bytes	

Note: 1. Be sure to set bit 7 to 40000B

2. The received data is stored in cnMDBxm according to the number of bytes corresponding to the DLC of the received frame (however, the upper limit is 8). CnMDBxm is undefined without data storage.

Note m = 0 to 15

22.7.22 CAN Message Configuration Register (CnMCONFm).

The CnMCONFm register is used to specify the type of message buffer and set the mask.

Figure 22-47. CAN Message Configuration Register Format (CnMCONFm) (1/2).

Reset value: No R/W is defined

	7	6	5	4	3	2	1	0
CnMCONFm	OVS	Rtr	MT2	MT1	MT0	0	0	MA0

OVS	Override the control bit
0	The packet buffer for received data frame 1 is not overwritten by the newly received data frame. Newly received data frames are discarded.
1	The packet buffer of received data frame 1 is overwritten by the newly received data frame.

1 The packet buffer for the received data frame is the receive packet buffer whose DN bit has been set to 1

Note Regardless of the settings of the OVS bit and DN bits, remote frames are received and stored. Remote frames (ID matching, RTR=0, TRQ=0) that meet other criteria are always received and stored in the corresponding message buffer (generate interrupts, set DN flags, MDLC [3:0] bit updates, and recorded in the receive history list).

Rtr	Remote frame request bit ¹
0	Send a data frame
1	Sends a remote frame

The 1 RTR bit specifies the type of packet frame transferred from a packet buffer defined as a transmit packet buffer. Even if a valid remote frame has been received, the RTR of the transmitted packet buffer for the received frame is still 0. Even if a remote frame with a matching ID is received from the CAN bus, the RTR bit of the transmit packet buffer has been set to 1 so that the remote frame can be transmitted, and the remote frame may not be received or stored (an interrupt has been generated, the DN flag has been set, the MDLC (3:0) bit has been updated), and recorded to the receive history list).

MT2	MT1	MT0	The message cache type sets the bit
0	0	0	Send packet caching
0	0	1	Receive packet caching (no masking settings).
0	1	0	Receive message cache (1 masking setting).
0	1	1	Receive message cache (2 masking settings).
1	0	0	Receive packet caching (3 masking settings).
1	0	1	Receive message caching (4 masking settings).
Other values			Prohibit settings

Note m=0 to 15

Fig.22-47. CAN Frame configuration register format(CnMCONFm)(2/2)

MA0	Message cache allocation bits
0	Message caching is not used
1	Use message caching

Note: Be sure to write bits 1 and 2 to 0

Note m = 0 to 15

22.7.23 CAN message ID register m (CnMIDLm and CnMIDHm).

CnMIDLm and CnMIDHm registers are used to set identifiers (IDs)

Figure 22-48. CAN message ID registers in m format (CnMIDLm and CnMIDHm).

Reset value: No

R/W

	15	14	13	12	11	10	9	8
CnMIDLm	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
	7	6	5	4	3	2	1	0
	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	15	14	13	12	11	10	9	8
CnMIDHm	IDE	0	0	ID28	ID27	ID26	ID25	ID24
	7	6	5	4	3	2	1	0
	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16

IDE	The format mode specifies the bit
0	Standard format mode (ID18 to ID28: 11 bits) ¹
1	Extended format mode (ID0 to ID28:29 bits).

1 ID0 to ID17 bits are not used

ID0 to ID28	Message ID
ID18 through ID28	11-bit standard ID value (when IDE=0).
ID0 through ID28	29-bit extension ID value (when IDE=1).

Note 1 Be sure to write bits 13 and 14 of the CnMIDHm register to 0

- Make sure to align the ID values into this register according to the positioning position given. Note that for standard IDs, the ID value must be moved to the ID11 to ID28 positions.

Note m=0 to 15

22.7.24 CAN message control register m (CnMCTRLm).

The CnMCTRLm register is used to control the operation of the packet buffer.

Figure 22-49. CAN message control register m format (CnMCTRLm) (1/3).

Reset Value:000xx000B R/W

(a) read

(b)

	15	14	13	12	11	10	9	8
CnMCTRLm	0	0	MUC	0	0	0	0	0
	7	6	5	4	3	2	1	0
	0	0	0	MOW	IE	DN	TRQ	RDY

(c) write

	15	14	13	12	11	10	9	8
CnMCTRLm	0	0	0	0	Set IE	0	Set TRQ	Set RDY
	7	6	5	4	3	2	1	0
	0	0	0	Clear MOW	Clear IE	Clear DN	Clear TRQ	Clear RDY

(a) read

MUC ^{Note}	Message cache data update bits
0	The CAN module has not updated the message buffer (receive and store)
1	The CAN module is updating the message buffer (receive and store).

Note The MUC bit is undefined until the first receive and store is made

MOW	Message caching overrides the status bit
0	Newly received data frames do not overwrite the packet buffer
1	The newly received data frame overwrites the packet buffer

Note Even if a remote frame is received and stored in the transmit message cache, the DN is equal to 1 and the MOW bit is not set to 1. .

IE	Packet caching interrupts the request enable bit
0	Receive Packet Buffer: Valid Packet Receive Completion Interrupt disabled. Transmit packet buffer: Normal packet transfer completion interrupt is disabled.
1	Receive Packet Buffer: Valid packet receive completion interrupt enabled. Transmit packet buffer: Normal packet transfer completion interrupt is enabled.

Note m = 0 to 15

Fig.22-49. CAN Frame control register m format (CnMCTRLm)(2/3)

DN	Message cache data update bits
0	Data frames or remote frames are not stored in the message buffer
1	Data frames or remote frames are stored in a message buffer

TRQ	Packet caching sends request bits
0	There are no pending or transmitting packet frame transfer requests in the packet buffer.
1	The message buffer holds the transmission of the packet frame pending or transmitting the packet frame. .

Note: Do not set the TRQ bit and the RDY bit (1) at the same time. Set the RDY bit (1) before setting the TRQ bit

RDY	Message cache ready bit
0	Message buffers can be written by software. The CAN module cannot write to the packet buffer
1	Write packet buffers through the software (except write access to RDY, TRQ, DN, and MOW bits) is ignored. The CAN module can write to the packet buffer. .

Note: 1. Do not clear the RDY bit (0) during message transmission.

Clear the RDY bit (0) as per the transfer abort to redefine the packet buffer.

- When the RDY bit is not cleared (although it has been cleared), clear again
- Before writing to the packet buffer register, ensure that the RDY bit is cleared. This acknowledgment is performed by reading the RDY bits. However, it is not necessary to confirm setting the TRQ bit, clearing the DN bit, setting the RDY bit, or clearing the MOW bit of the CnMCTRLm register. .

(b) write

ClearMOW	Set the MOW bit
0	The MOW bits have not changed.
1	The MOW bit is cleared to 0

SetIE	ClearIE	Set the IE bit
0	1	IE is cleared 0
1	0	IE is placed 1
Other values		The IE bit does not change

Note: Always set the IE bit and the RDY bit separately

ClearDN	Set the DN bit
0	The DN does not change
1	The DN is cleared 0

Note: Do not set the DN bit to 1 with software. Be sure to write bit 10 to 0.

Note m=0 to 15

Fig.22-49. CAN frame control register m format (CnMCTRLm)(3/3)

SetTRQ	ClearTRQ	TRQ set bit
0	1	TRQ clear 0
1	0	TRQ set to 1
Other values		There is no change in the TRQ

Note: When receiving packets from other nodes or transmitting dissipative packets, even if the TRQ bit is set to 1, the transmission may not start immediately.

Even if the TRQ bit is cleared to 0, the transfer is not aborted. If a message is being transmitted, the transmission continues until the transmission is complete (successful or unsuccessful).

SetRDY	ClearRDY	RDY setting bit
0	1	RDY clear 0
1	0	RDY set to 1
Other values		RDY does not change

Note Set the IE bit and the RDY bit separately.

Note m= 0 to 15

22.7.25 Serial communication pin select register 1 (PIOR3).

The PIOR3 register is used to switch the input source to the timer array unit and the CAN0 communication pin. This register can be read or written in 8-bit form.

The PIOR3 registers can be used to select the CTxD0 and CRxD0 pins on two different ports.

Figure 22-50. Serial communication pin select register 1 format (PIOR3).

Address: 0x4004087C		reset value: 00H		R/W				
	7	6	5	4	3	2	1	0
PIOR3	0	0	0	0	WORST33	WORST32	WORST31	WORST30

PIOR33	CAN0 communication pin selection	
	CTxD0	CRxD0
0	P02	P03
1	P51	P50

22.7.26 Port mode registers 0/4/5/6 (PM0/4/5/6).

The PMx register is used to set port x as input or output.

When using the P02/CTxD0 or P51/CTxD0 or P64/CTxD1 or P46/CTxD2 pins for serial data output, clear the PM02 or PM51 or PM64 or PM46 bits to "0" and set the output latch of the P02 or P51 or P64 or P46 to "1".

When using the P03/CRxD0 or P50/CRxD0 or P65/CRxD1 or P47/CRxD2 pins for serial data input, the PM03 or PM50 will be used or PM65 or PM47 bits set to "1". At this point, the output latch of the P03 or P50 or P65 or P47 may be "0" or "1"

The PMx register can be set using the 8-bit memory manipulation instructions. Setting these registers to FFH produces a reset signal.

22.8 CAN controller initialization

22.8.1 CAN module initialization

Before enabling CAN module operation, the system clock of the CAN module needs to be determined by setting the CCP [3:0] bit of the CnGMCS register by software. After the CAN module is working, do not change the setting of the CAN module system clock.

Enable the CAN module by setting the GOM bit of the CnGMCTRL register.

For procedures on initializing the CAN module, see 22.16The operation of the CAN controller.

22.8.2 Initialization of the packet cache

When the CAN module is enabled, the packet buffer contains undefined values. Minimal initialization of all message buffers (even if the application is not in use) is required before switching the CAN module from initialization mode to one of the operating modes.

- Clear the RDY, TRQ, and DN bits (0) of cnMCTRLm
- Clear the MA0 bit (0) of the CnMCONFm

Note m= 0 to 15

22.8.3 Redefine the message cache

Redefining a packet buffer means that when a packet is received or transmitted, the ID and control information of the packet buffer is changed without affecting other send/receive operations.

(1) Redefine the message cache in initialization mode

Put the CAN module into initialization mode once, and then change the ID and control information of the packet buffer in initialization mode. After changing the ID and control information, set the CAN module to operating mode.

(2) Redefine the message cache in the receive

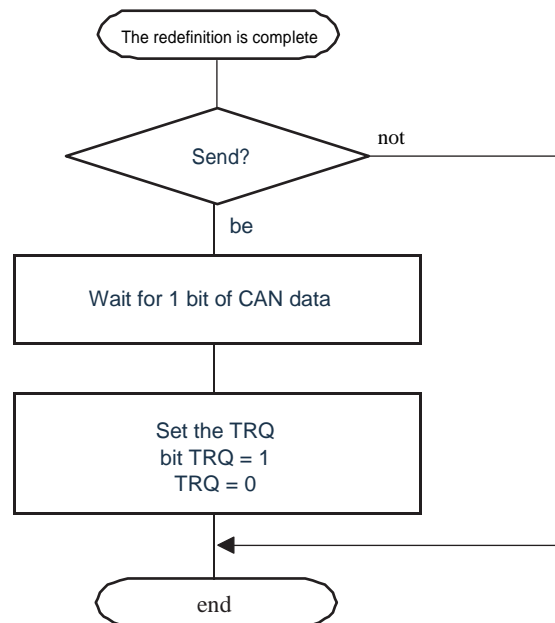
The redefined operation **Figure 22-65**.

(3) Redefine the message cache in the send

To rewrite the contents of the transmission packet buffer that has been set for a transfer request, perform transfer abort processing (see 22.10.4**abort processing outside of Automatic Block Transfer (ABT) in Normal Operating Mode and Transfer Abort Processing outside of Automatic Block Transfer (ABT) 22.10.4**).) .

Confirm that the transfer was aborted or completed, and then redefine the packet buffer. After you redefine the transfer message buffer, use the procedure described below to set up the transfer request. However, when a transfer request is set to a redefined packet buffer without aborting an in-progress transfer, a 1-bit wait time is not required.

Fig.22-51. After redefining the send request to the send message cache settings(TRQ)



note1. When a message is received, it will be set to each received packet buffer ID and masking perform receive filtering. If not complied Figure 22-65 in the process, the contents of the packet buffer after redefinition may contradict the received results (receiving filtered results). If this happens, check that it was first received and stored in the message buffer after redefining ID and IDE. Whether it is stored after the packet buffer is redefined ID and IDE. If it is not stored after redefining ID and IDE, redefine the packet buffer.

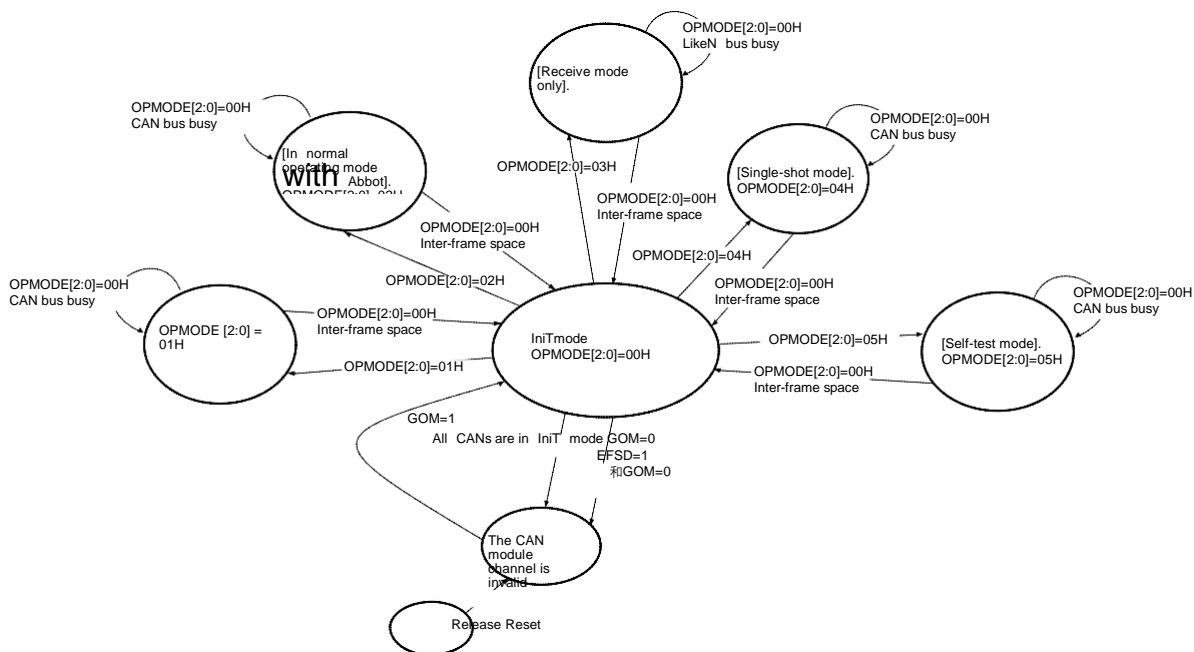
2. When sending a message, the transmission priority is checked against the ID, IDE, and RTR bits set to each sending message buffer. Select the transmission packet buffer with the highest priority for the transfer. If the process in FIG. Fig.22-51, then the message with id without the highest priority may be transmitted after redefinition.

22.8.4 Transition from initialization mode to operating mode

The CAN module can be switched to the following operating modes

- Normal operating mode
- Normal mode of operation with ABT
- Receive mode only
- Single-shot mode
- Self-test mode

Figure 22-52. Transition to operating mode



The transition from initialization mode to operating mode is controlled by the bit string OPMODE [2:0] in the CnCTRL register.

Changing from one mode of operation to another requires a transition from the two to an initialization mode. Do not directly change one mode of operation to another, otherwise the operation is not guaranteed.

When the CAN bus is not in the inter-frame space (i.e. the frame receive or transmit is in progress) and the CAN module is in the inter-frame space (the value of OPMODE [2:0] is changed to 00H), the application transition from operating mode to initialization mode is pending. After making a request to change the mode to initialization mode, read the OPMODE [2:0] bit until its value changes to 000B to confirm that the module has entered initialization mode (see Fig.22-62).

22.8.5 Resets the CAN Module Error Counter cnERC

If you need to reset the CAN module error counter CnERC and the CAN module information register CnINFO, when reinitializing or forced recovery from the bus shutdown state, set the CCPERC bit of the CnCTRL register to 1 in initialization mode. When this bit is set to 1, the CAN module error counter CnERC and the CAN module information register CnINFO will be cleared to their default values.

22.9 Message reception

22.9.1 Message reception

In all operating modes, the full packet buffer area is analyzed to find the appropriate buffer to store the newly received packets. All message buffers that meet the following criteria are included in the evaluation (RX search process):

- Packet cache used
(The MA0 bit of CnMCONFm is 1).)
- Is set up as a cache for receiving packets
(The MT[2:0] bit of CnMCONFm is set to 001B, 010B, 011B, 100B, or 101B.).)
- Ready to receive
(The RDY bit of CnMCTRLm is set to 1).

When there are two or more packet caches to receive a message, the message is stored in a higher priority cache. For example, when an unmasked receive packet buffer and a receive packet buffer linked to mask 1 have the same ID, the received packet is not stored in the packet buffer linked to mask 1, even if the packet buffer has not yet received the packet and the message is already in the unmasked receive packet buffer. In other words, when set to store messages in two or more packet buffers with different priorities, the message buffer with the highest priority always stores messages; Messages are not stored in lower priority packet buffers. This also applies when the message buffer with the highest priority cannot receive and store messages (i.e. when DN=1 indicates that a message has been received, but because of OWS=0, overrides are disabled). In this case, the message is not actually received and stored in the candidate buffer with the highest priority, but also not in the lower priority packet buffer.

Priority	Storage in case the same ID is set	
1 (high).	There is no blocked packet caching	DN=0
		DN=1 and OWS=1
2	Link to packet cache for block 1	DN=0
		DN=1 and OWS=1
3	Link to the packet cache for block 2	DN=0
		DN=1 and OWS=1
4	Links to the packet cache for block 3	DN=0
		DN=1 and OWS=1
5 (low).	Link to packet cache for block 4	DN=0
		DN=1 and OWS=1

Note m= 0 to 15

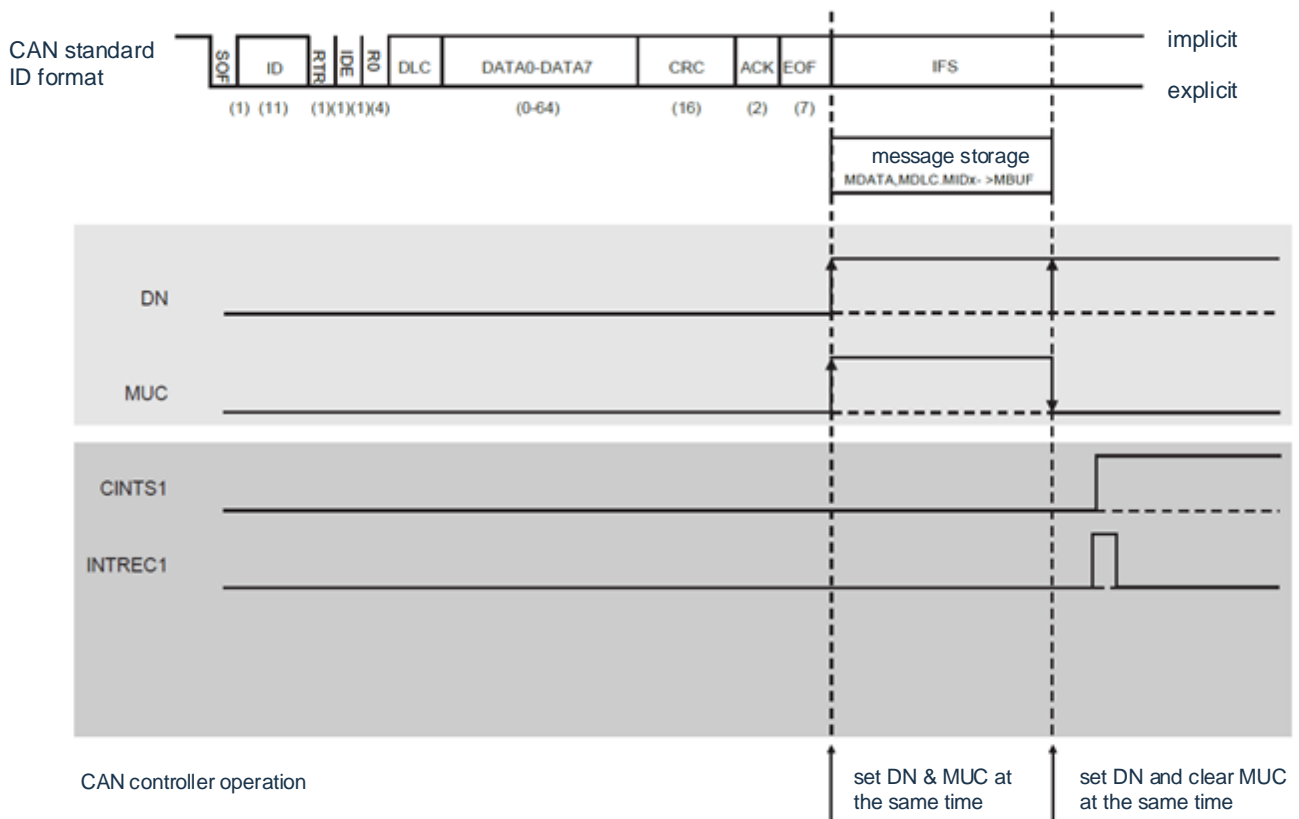
22.9.2 Receive data reads

To maintain data consistency when Figure 22-74-74 Fig.22-76.

During packet reception, the CAN module sets the DN of the CnMCTRLm register twice: at the beginning of storing the data into the packet buffer, and again at the end of this stored procedure. In this stored procedure, the MUC bit of the message buffer CnMCTRLm register is set. (See **Figure 22-53**).

The received history list is also updated before the stored procedure. In addition, during the stored procedure (MUC_1), the RDY bit of the CnMCTRLm register of the message buffer is locked to avoid coincidental data WR through the CPU. Note that when the CPU accesses the packet buffer, the stored process may be disturbed (delayed).

Figure 22-53. The DN and MUC bits set the period (to the standard ID format).



Note m= 0 to 15

22.9.3 Receive history list function

The Receive History List (RHL) function in the Receive History List records the number of receive message buffers that are received and stored for each data frame or remote frame. RHL consists of a storage element equivalent to up to 23 messages, the last packet-in pointer (LIPT) has the corresponding CnLIPT register and the Receive History List Fetch Pointer (RGPT) has the corresponding CnRGPT register.

Immediately after the CAN module transitions from initialization mode to one of the modes of operation, the RHL is immediately left undefined.

The CnLIPT register contains the contents of the RHL element indicated by the value of the LIPT pointer minus 1. Therefore, by reading the CnLIPT register, it is possible to check the number of packet buffers that receive and store the data frame or remote frame first. The LIPT pointer is used as a write pointer to indicate which part of the RHL a packet buffer number is logged. Whenever a data frame or remote frame is received and stored, the corresponding message buffer number is logged to the RHL element indicated by the LIPT pointer. Each time you record to THEHL, the LIPT pointer is automatically incremented. This way, the number of packet buffers that receive and store frames is recorded in chronological order.

The RGPT pointer is used as a read pointer to read the message buffer number of the record from THEHL. This pointer indicates the first RHL element that the CPU has not yet read. By reading the CnRGPT registers on a per-software basis, you can read the number of packet buffers that have received and stored data frames or remote frames. Each time the message buffer number is read from the CnRGPT register, the RGPT pointer is automatically incremented.

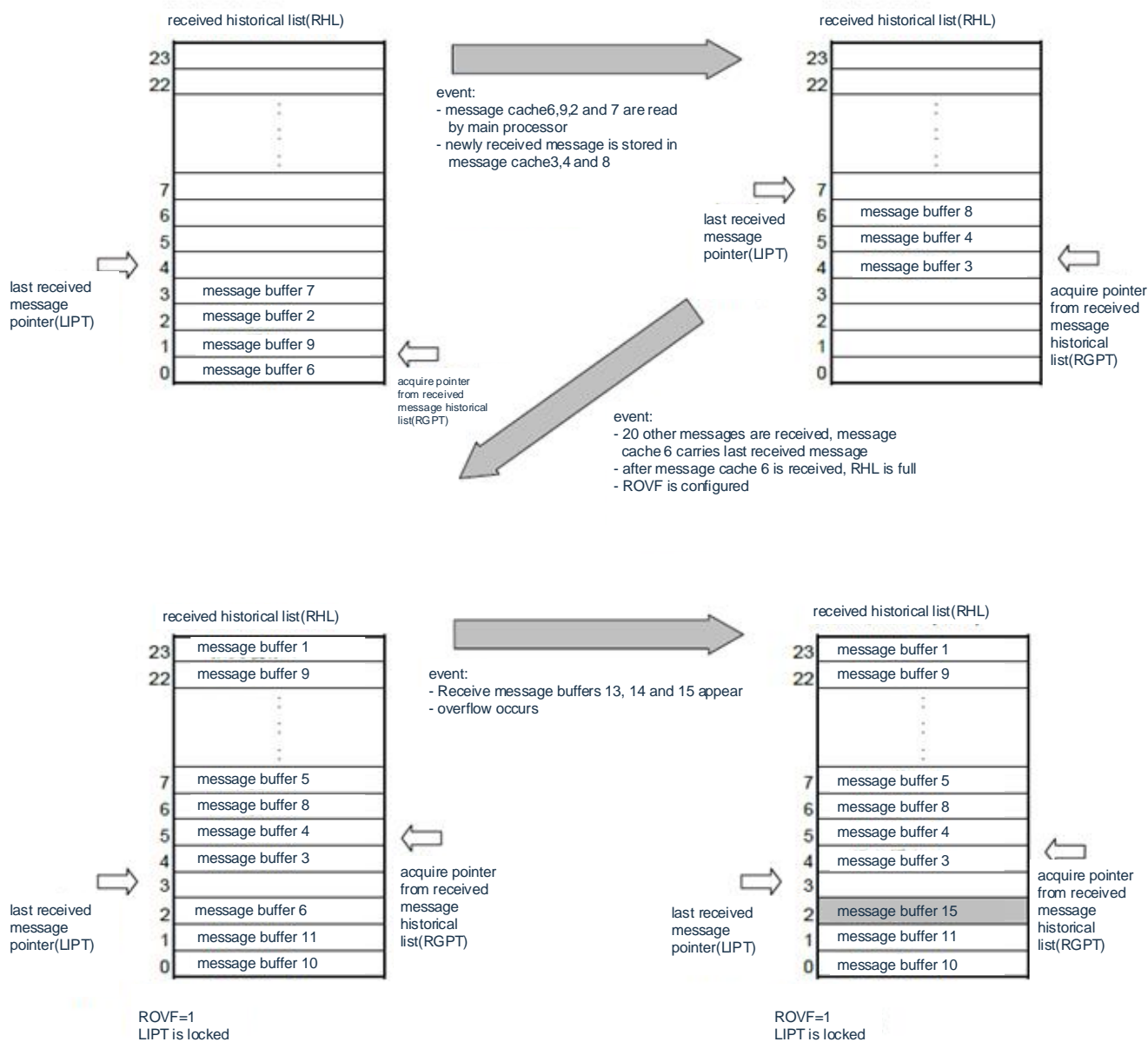
If the value of the RGPT pointer matches the value of the LIPT pointer, the RHPM bit of the CnRGPT register (receive history list pointer matches) is set to 1. This indicates that the unread packet buffer number in THEHL is not retained. If a new message buffer number is logged, the LIPT pointer is incremented, and since its value no longer matches the value of the RGPT pointer, the RHPM bit is cleared. In other words, the number of unread packet buffers exists in THER.

If the LIPT pointer increments and matches the value of the RGPT pointer minus 1, the ROVF bit of the CnRGPT register (Receive History List Overflow) is set to 1. This indicates that the RHL is filled with the number of packet buffers that have not yet been read. When further packets are received and stored, the message buffer number of the last record is overwritten by the number of packet buffers received and stored for the new message. In this case, after setting the ROVF bit (1), the packet buffer numbers recorded in the RHL are not in exact chronological order. However, the message itself is not lost and can be located in the message buffer memory with the help of DN bits through CPU search.

Note: If the history list is in an overflow condition (ROVF is set), it is still possible to read the contents of the history list until the history list is empty (indicated by the RHPM flag). However, the history list remains in an overflow state until the ROVF is cleared by software. If the ROVF is not cleared, the RHPM flag is also not updated (cleared) on the message of the newly received frame. This can lead to the situation where THE RHPM represents an empty history list, although a reception has occurred and the history list is in an overflow state (ROVF and RHPM are set).

As long as the RHL contains 23 or fewer entries, the order of occurrence is maintained. If more receives occur while the host processor is not reading THEHL, the full receive sequence cannot be resumed.

Figure 22-54. Receive a list of histories



ROVF=1 indicates that LIPT equals RGPT-1, and the packet buffer number is stored in the element indicated by LIPT-1.

22.9.4 Blocking function

For any packet buffer used for reception, you can choose to allocate a quarter of the receive mask (or no mask).

By using the masking function, packet ID comparisons can be reduced by masking bits, allowing multiple different IDs to be received into a buffer.

When the mask function is in effect, the identifier bit in the received message whose mask is defined as "1" is not compared to the corresponding identifier bit in the message buffer.

However, this comparison is performed in bits defined as 0 by the mask.

For example, all messages with standard format IDs (where the BITs 25 to ID27 are "0", ID22 and ID24 bits are "1") will be stored in the packet buffer 14. The procedure for this example is as follows.

<1> Identifier is stored in the message cache

ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18
x	0	0	0	1	x	1	x	x	x	x

x = Leave it alone

<2> identifier is configured into packet cache 14 (example).

(Use the CAN 0 message ID registers L14 and H14 (CnMIDL14 and CnMIDH14)).

ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18
x	0	0	0	1	x	1	x	x	x	x
ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
x	x	x	x	x	x	x	x	x	x	x
ID6	ID5	ID4	ID3	ID2	ID1	ID0				
x	x	x	x	x	x	x				

IDs from ID25 to ID27 are cleared as "0", ID24 and ID22 are set to "1", registered (initialized) to packet buffer 14.

Note Message buffer 14 is set to the standard format identifier linked to mask 1 (MT[2:0] of the CnMCONF14 register is set to 010B) .

<3> CAN Module 1 mask1 (example).

(Using CAN module mask 1 registers L and H (CnMASK1L and CnMASK1H)).

CMID28	CMID27	CMID26	CMID25	CMID24	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18
1	0	0	0	0	1	0	1	1	1	1
CMID17	CMID16	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8	CMID7
1	1	1	1	1	1	1	1	1	1	1
CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0				
1	1	1	1	1	1	1				

1: No comparison (masking).

0: Compare

THE CMID24 to CMID27 and CMID22 bits are cleared as "0", and the CMID0 to CMID21, CMID23, and CMID28 bits are set to "1".

22.9.5 Multi-buffer receive block capability

The Multiple Buffer Receive Block (MBRB) feature is used to store data blocks sequentially in two or more packet buffers without CPU interaction by setting the same ID to two or more packet buffers with the same packet buffer type.

For example, suppose the same message buffer type is set to 5 message buffers (message buffers 10 to 14), and they are given the same ID. If the first message ID received is the same as the buffer ID, the message is stored in the packet buffer 10. At this point, the DN bit of the packet buffer 10 will be set, and the packet buffer will be disabled when subsequent packets are received.

If the next message with a matching ID is received, the message is received and stored in the message buffer 11. Each time a message with a matching ID is received, it will be stored sequentially (in ascending order) in the message buffers 12, 13 and 14. Even if a block of data consisting of multiple messages is received, the packets are stored and received after scanning, without overwriting the previously received matching ID data.

It is possible to check whether the data block is received and stored by setting the IE bit of the CnMCTRLm register of each packet buffer, for example, if the data block consists of k packets, the k message buffer will be initialized to receive the data block. The IE bit in the packet buffer 0 to (k-2) is cleared to 0 (disable interrupt), and the IE bit in packet buffer k-1 is set to 1 (interrupt is enabled). In this case, a receive completion interrupt occurs when a message is received and stored in the message buffer k-1, indicating that the MBRB is full. Alternatively, by clearing the IE bits of the packet buffer 0 to (k-3) and setting the IE bits of the packet buffer k-2, you can issue a warning that the MBRB is about to overflow.

The basic conditions for storing the receive data in each packet buffer of the MBRB are the same as the conditions for storing the data in a single packet buffer.

Note 1. MBRB can be configured for each of the same type of message buffers. Thus, even if there is a gap in the packet buffer of another MBRB with a matching ID but a different packet buffer type, the received packet is not stored in that packet buffer, but is discarded.

2. MBRB does not have a ring cache structure. Therefore, after a message is stored in the packet buffer with the highest number in the MBRB configuration, the newly received packet will not be stored in the packet buffer with the lowest number in the packet buffer.
3. MBRB operates according to receiving and storage conditions; There are no settings specific to MBRB, such as function enabler bits. MbRB is automatically configured by setting the same packet buffer type and ID to two or more packet buffers. .
4. With MBRB, "Match ID" means "Match ID After Mask". Even if the ID set to each packet buffer is different, if the MASK register masked ID matches, it is considered a matching ID, and the buffer with this ID is considered the storage target for the packet. .
5. For the priority of MBRBs, refer to 22.9.1 Message reception.

Note m= 0 to 15

22.9.6 Remote frame reception

In all operating modes, when a remote frame is received, a packet buffer that can store the remote frame is searched from all packet buffers that meet the following criteria.

- Used for packet caching
(Bit MA0 of the CnMCONFm register is set to 1).
- Is set up as a cache for transmission messages
(The MT[2:0] bit of the CnMCONFm register is set to 000B.)
- Prepare to receive
(The RDY bit of the CnMCTRLm register is set to 1.).
- Set to transmit messages
(RTR bit clear 0 for cnMCONFm registers).
- The transfer request is not set
(The TRQ bit of the CnMCTRLm register is cleared to 0).

After accepting a remote frame, if the ID of the received remote frame matches the ID of the message buffer that meets the above criteria, do the following:

- The MDLC [3:0] bit of the CnMDLCLm register stores the received DLC value
- The Data Intervals from CnMDATA0m to CnMDATA7m are not updated (saving pre-receive data).
- The DN bit of the CnMCTRLm register is set at 1.
- The CNINTS register's CINTS1 bit is set to 1 (if the IE bit in the received and stored message buffer registers cnMCTRLm is set to 1).
- Output Receive Completion Interrupt (INTCREC) (if the IE bit in the message buffer register for receiving and storing frames is set to 1, and the CIE1 bit in the CnIE register is set to 1).
- The message buffer number is logged to the receive history list.

Note When a packet buffer is searched for to receive and store remote frames, the overwrite control of the OWS bit of the message buffer CnMCONFm register and the DN bit of the CnMCTRLm register is not affected. The settings for OWS are ignored and the DN is set in any case. If multiple transmit packet buffers have the same ID and the ID of the received remote frame matches that ID, the remote frame is stored in the buffer with the lowest number of packet buffers.

Note m=0 to 15

22.10 Message sending

22.10.1 Message sending

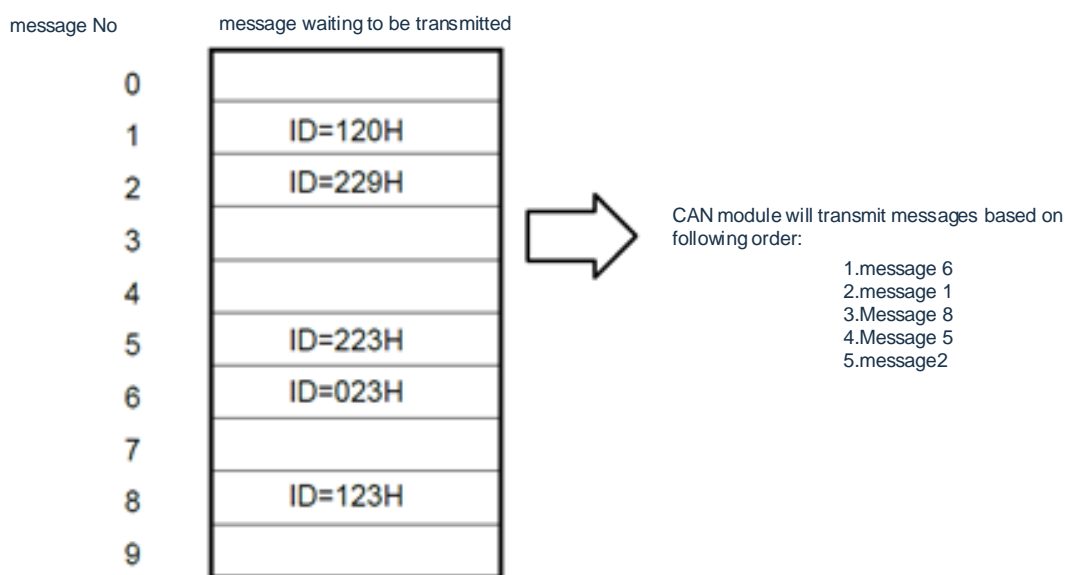
In all operating modes, if the TRQ bit is set to 1 in the packet buffer that meets the following criteria, the search for the packet buffer to transmit the message begins.

- It is used for packet caching
(MA0 position 1 of the CnMCONFm register).
- Set to send message caching
(The MT[2:0] bit of the CnMCONFm register is set to 000B.).
- Ready to send
(RDY position of the CnMCTRLm register 1.).

The CAN system is a multi-master communication system. In such a system, the priority of the message transmission is determined according to the message identifier (ID), in order to achieve software transmission processing when there are multiple messages waiting to be transmitted, the CAN module uses hardware to detect the message ID with the highest priority and automatically identify the message, which eliminates the need for software priority control.

The send priority is controlled by an identifier (ID).

Figure 22-55. Example of message handling



After sending a packet search, sends the highest priority send message with the highest priority of the send message buffer that has the pending send request (the message buffer with the TRQ bit set to 1).

If a new send request is set, the send packet buffer with the new send request is compared to the send packet buffer with the pending send request. If a new send request has a higher priority, the request is transmitted unless a lower priority message has already begun. However, if the transmission of lower priority messages has already begun, a new send request will be sent later. To address this priority reversal effect, the software can perform a transmit abort request on a low-priority message. The highest priority is determined according to the following rules.

Priority	condition	description
1 (high).	The value of the first 11 bits of the ID [ID18 to ID28]:	The message frame of the lowest value represented by the first 11 bits of the ID is sent first. If the value of the 11-bit standard ID is equal to or less than the first 11 bits of the 29-bit extension ID, the priority of the 11-bit standard ID is higher than that of the packet frame with the 29-bit extension ID.
2	The frame type	Data frames with an 11-bit standard ID (RTR bit cleared to 0) have a higher priority than remote frames with standard IDs and packet frames with extended IDs.
3	The ID type	Packet frames with a standard ID (ID bit cleared to 0) have a higher priority than packet frames with extended IDs
4	The value of the ID low 18 bits [ID0 to ID17]:	If multiple packet frames that send pending extension ID have the same value and the same frame type (equal to the RTR bit value) in the first 11 bits of the ID, the message frame with the lowest median value of their extended ID low 18 bits is transmitted first
5 (low).	Message cache number	If two or more packet buffers request to send a message frame with the same ID, the message from the message buffer with the lowest message buffer number is transmitted first.

Note 1 If the auto block transfer request bit ABTTRG is set to 1 in normal operating mode with ABT, the TRQ bit of only one packet cache in the ABT packet cache group is set to 1. If the ABT pattern is triggered by the ABTTRG bit, one of the TRQ bits in the ABT region is set to 1 (buffers 0 to 7). In addition to this TRQ bit, the application can request other TX packet buffer transfers that are not part of the ABT region (set the TRQ to 1). In this case, the Interval Arbitration Process (TX-search) evaluates all TX packet buffers with the TRQ bit set to 1 and selects the packet buffer with the highest priority identifier for the next transmission. If 2 or more identifiers have the highest priority (i.e. the same identifier), the message at the lowest packet buffer number is transmitted first.

After the message frame is successfully transmitted, the following actions are performed:

- The TRQ flag for the corresponding transmit packet buffer is automatically cleared to 0.
 - The transmit completion status bit CINTS0 of the CnINTS register is set to 1 (if the interrupt enable bit (IE) of the corresponding transmit packet buffer is set to 1).
 - Interrupt request signal INCNTRX output (if the CIE0 bit of the CnIE register is set to 1 and the interrupt enable bit (IE) of the corresponding transmit packet buffer is set to 1).
2. When changing the contents of a send buffer, you must clear the RDY flag for this buffer before updating the contents of the buffer. As with internal transfer operations, the RDY flag may be temporarily locked, and the status of the RDY must be checked by the software after the change.

m=0到15

22.10.2 Send history list function

The Transmission History List (THL) feature records the number of the packet buffer that sends data or remote frames in the transmission history list. THL contains storage elements equivalent to up to seven messages, the last outgoing message pointer (LOPT) with the corresponding CnLOPT register, and the Transmit History List Fetch Pointer (TGPT) of the CnTGPT register.

After the CAN module transitions from initialization mode to one of the operating modes, THL will immediately be left undefined.

The CnLOPT register contains the contents of the THL element indicated by the value of the LOPT pointer minus 1. Therefore, by reading the CnLOPT register, it is possible to check the number of packet buffers for the first transmitted data frame or remote frame. The LOPT pointer is used as a write pointer to indicate which part of the THL of the packet buffer number is logged. Whenever a data frame or remote frame is transmitted, the corresponding message buffer number is recorded to the THL element indicated by the LOPT pointer. Each time the THL is recorded, the LOPT pointer is automatically incremented. In this way, the packet buffer numbers of the frames received and stored are recorded in chronological order.

The TGPT pointer is used as a read pointer to read the message buffer number of the record from the THL. This pointer indicates the first THL element that the CPU has not yet read. By reading the CnTGPT registers by software, the number of packet buffers that have been transmitted can be read. Each time the message buffer number is read from the CnTGPT register, the TGPT pointer is automatically incremented.

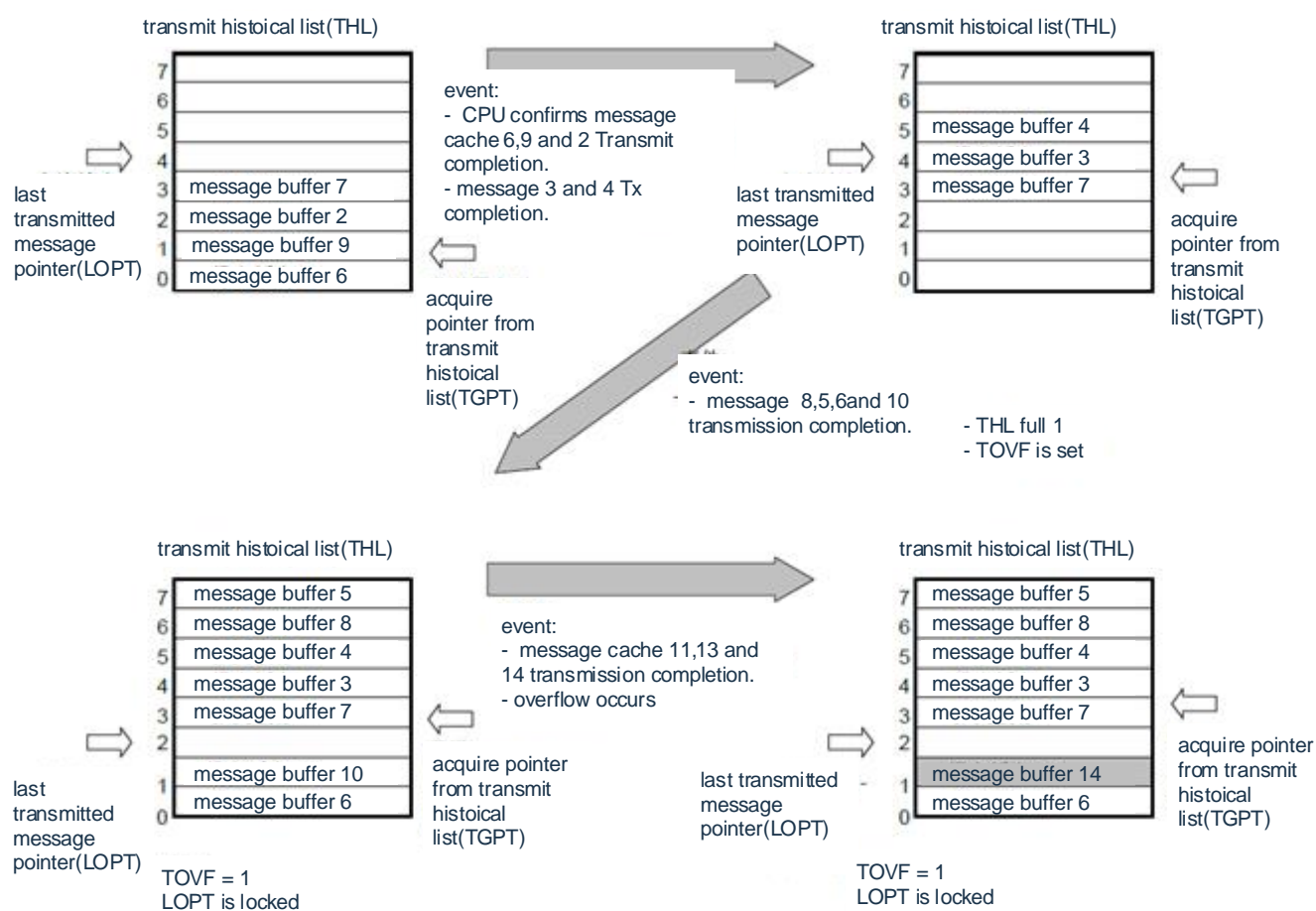
If the value of the TGPT pointer matches the value of the LOPT pointer, the THPM bit of the CnTGPT register (transfer history list pointer matches) is set to 1. This indicates that no unread packet buffer numbers are reserved in the TRL. If a new message buffer number is logged, the LOPT pointer will be incremented, and since its value no longer matches the value of the TGPT pointer, the THPM bit will be cleared. In other words, the number of unread packet buffers exists in the TRL.

If the LOPT pointer increments and matches the value of the TGPT pointer minus 1, the TOVF bit of the CnTGPT register (transmission history list overflow) is set to 1. This indicates that the TRL is filled with packet buffer numbers that have not yet been read. If a new message is received and stored, the last recorded message buffer number is overwritten by the number of message buffers for subsequent transmissions. After the TOVF bit is set to 1, the packet buffer numbers recorded in THL do not fully reflect the chronological order. However, other transport packets can be found by CPU searches that apply to all transmit packet buffers, unless the CPU has not already overwritten the transfer object in one of these buffers before. A total of up to six transfers occur to complete the scan without THL overflow.

Note: If the history list is in an overflow condition (TOVF is set), the contents of the history list can still be read until the history list is empty (indicated by the THPM flag). However, the history list remains in an overflow state until TOVF is cleared by the software. If TOVF is not cleared, the THPM flag is not updated (cleared) after a new message is successfully transmitted. This can lead to the situation where THPM indicates an empty history list, although the transfer was successful and the history list is in an overflow state (TOVF and THPM are set).

Note m= 0 to 15

Figure 22-56. Send a list of histories



TOVF=1 indicates that LOPT is equal to TGPT-1, and the message buffer number is stored in the element indicated by L OPT-1.

22.10.3 Automatic Block Transfer (ABT).

The Automatic Block Transfer (ABT) feature is used to successfully transmit two or more data frames in succession without CPU interaction. The maximum number of transmit packet buffers allocated to the ABT function is 8 (packet buffer numbers 0 to 7).

By setting the OPMODE [2:0] bit of the CnCTRL register to 010B, the "Normal Operating Mode with Automatic Block Transfer" (referred to here as ABT mode) can be selected.

To issue an ABT transfer request, first go through a software-defined packet buffer. Set bit MA0 (1) in all packet buffers used for ABT and define all buffers as transmission packet buffers by setting mt[2:0] bit to 000B. Make sure that the ABT has the ID set for each packet buffer, even if the same ID is used for all packet buffers. To use two or more IDs, set the ID of each packet buffer using the CnMIDLm and CnMIDHm registers. Before issuing a transfer request for the ABT function, set CnMDLCm and CnMDATA0m to CnMDATA7m.

After the initialization of the packet buffer of the ABT is complete, you need to set the RDY bit to 1. In ABT mode, the TRQ bit is not operated by software.

After preparing the data for the ABT message buffer, set the ABTTRG bit to 1. Then start the automatic block transfer. When starting ABT, the TRQ bit in the first packet buffer (packet buffer 0) is automatically set to 1. After the data transfer of packet buffer 0 is completed, the TRQ bit of the next packet buffer is automatically set, and the packet buffer 1 is automatically set. In this way, the transfers are performed sequentially.

The program can insert a delay time within the interval of the transfer request (TRQ) that is automatically set when performing a continuous transfer. The delay time to insert is defined by the CnGMABTD register. The unit of delay time is DBT (Data Bit Time). DbT depends on the setting of the CnBRP and CnBTR registers.

In the transfer object within the ABT region, the priority of the transport ID is not evaluated. Data from packet buffers 0 to 7 are transmitted sequentially. When the data frame transfer from packet buffer 7 is complete, the ABTTRG bit is automatically cleared to 0 and the ABT operation is complete.

If the RDY bits of the ABT packet buffer are cleared during ABT, no data frames are transmitted from that buffer, ABT stops, and the ABTTRG bits are cleared. After that, the RDY and ABTTRG bits can be set to 1 by software to resume transmission from the packet buffer stopped by ABT. In order not to resume transmission from the packet buffer that stopped the ABT, the internal ABT engine can be reset by setting the ABTCLR bit to 1 when the ABTRG bit is stopped and the ABTRG bit is cleared to 0. In this case, if the ABTCLR bit is cleared to 0 and then the ABTTRG bit is set to 1, the transfer is initiated from the packet buffer 0. .

Interrupts can be used to check whether data frames are transmitted from all packet buffers of the ABT. To do this, it is necessary to clear that the IE bit of the CnMCTRLm register of each packet buffer except the last one is 0.

If a transmit packet buffer other than the ABT function (packet buffer 8 to 15) is allocated to the transmit packet buffer, the next packet to be transmitted is prioritized by the transmit ID of the transmit ABT packet buffer that is currently in a suspended state and is the transmit ID of the packet buffer other than the buffer used by the ABT function.

Data frames transmitted from the ABT packet buffer are not recorded in the Transmission History List (THL).

- Note
- 1 Set the ABTCLR bit to 1 and the ABTTRG bit to 0 to resume ABT operation at buffer number 0. If the ABTCLR bit is set to 1 and the ABTTRG bit is set to 1, subsequent operations are not guaranteed. .
 2. If the automatic block transfer engine is cleared by setting the ABTCLR bit to 1, the ABTCLR bit is automatically cleared after the processing of the purge request is complete.
 3. Do not set the ABTTRG bit in initialization mode. If the ABTTRG bit is set in initialization mode, correct operation is not guaranteed after the mode changes from initialization mode to ABT mode.
 4. Do not set the TRQ bit of the ABT packet buffer to 1 by the software in normal operation mode of ABT. Otherwise, the operation is not guaranteed.
 5. The CnGMABTD register is used to set the insertion delay time between the completion of the previous ABT packet and the TRQ bit set for the next ABT packet, and transmits continuously in ABT mode when the transmission request is set in ABT message number order. The time at which the message is actually transmitted to the CAN bus varies depending on the state of transmission from other sites and the setting state of the packet sending request other than the ABT message (packet buffer 8 to 15).
 6. If a transmission request is made for a message other than an ABT message, and a delay time is not inserted in the time interval for the ABT transmission request (CnGMABTD = 00H) is automatically set, the message other than the ABT message may not depend on the priority of the ABT message.
 7. When ABTTRG =1, do not clear the RDY bit to 0.
 8. If a packet is received from another node while the normal mode of operation with ABT is active, the TX packet in the ABT region may be transmitted with a delay of one frame, although the CnGMABTD register is set to 00H.

Note m=0 to 15

22.10.4 Transfer abort processing

- (1) Transfers outside of Automatic Block Transfer (ABT) are aborted in normal operating mode
The user can abort the transmission request by clearing the TRQ bit of the CnMCTRLm register to 0, and if the abort is successful, the TRQ bit will be cleared immediately. The TSTAT bit of the CnCTRL register indicating the status of the CAN bus transmission and the CnTGPT register can be used to check whether the transfer was successfully aborted (for more information, see Processing in Figure 22-70).
- (2) ABT transmissions are accompanied by transfer abort processing outside of automatic block transfers in normal operating mode
The user can clear the ABTTRG bit of the CnGMABT register to 0 to abort the transfer request. After checking the ABTTRG bit of the CnGMABT register = 0, clear the TRQ bit of the CnMCTRLm register to 0. If the abort is successful, the TRQ bit is cleared immediately. The TSTAT bits of the CnCTRL registers indicating the transfer status on the CAN bus and the CnTGPT register can be checked for successful abortion (for details, see Processing in Figure 22-72).

- (3) ABT transmissions are aborted in normal operating mode with automatic block transfers

To abort the started ABT, clear the ABTTRG bit of the CnGMABT register to 0. In this case, if the ABT message is currently being transmitted and until the transfer is complete (successful or not), the ABTTRG bit will remain at 1 and cleared to 0 immediately after the transfer is complete. This will abort ABT.

If the last transfer (before ABT) is successful, the normal operating mode of the ABT is preserved, with the internal ABT pointer pointing to the next message buffer to be transmitted.

If the transfer is wrong, the position of the internal ABT pointer depends on the state of the TRQ bit in the last transmitted packet buffer. If the TRQ bit is set to 1 when the ABTTRG bit is requested to be cleared, the internal ABT pointer points to the last transmitted packet buffer (for details, see the process in Figure 22-71). If the TRQ bit is cleared to 0 when the ABTTRG bit is cleared, the internal ABT pointer will increment (+1) and point to the next message buffer in the ABT region (for details, see the process in Figure 22-72).

Note: Make sure to abort ABT by clearing ABTRG to 0, this may not be possible if the transfer is aborted by clearing the RDY bits.

When the normal mode of operation of ABT is restored after ABT abort and the ABTTRG bit is set to 1, the next ABT packet buffer to be transmitted can be determined from the following table.

ABT packet caching status TRQ	Abort after a successful transfer	Aborted after error transmission
Setting (1).	The next message cache for the ABT interval ¹	ABT interval identical message caching
Clear (0).	The next message cache for the ABT interval ¹	The next message cache for the ABT interval ¹

- 1: The above recovery operation can only be performed if there is a packet buffer in the ABT region that is ready to use ABT. For example, an abort request issued while the ABT of packet buffer 7 is being processed is considered to have been completed by the ABT, not aborted, and if the transmission of packet buffer 7 has completed successfully, the ABT TRG has been cleared to 0. If the RDY bit in the next packet buffer in the ABT region is cleared to 0, the internal ABT pointer is retained, but even if the ABT TRG is set to 1 and the ABT ends immediately, no recovery operation is performed. .

Note m=0 to 15

22.10.5 Remote frame transfer

Remote frames can only be transmitted from the transmit packet buffer. The RTR bit setting via the CnMCONFM register is either a data frame or a remote frame transmission. Set the RTR bit to (1) to set the remote frame transmission.

Note m= 0 to 15

22.11 Power-saving mode

22.11.1 CAN sleep mode

CAN sleep mode can be used to set the CAN controller to standby mode to reduce power consumption. The CAN module can enter CAN sleep mode from all operating modes. The release of the CAN sleep mode returns the CAN module to the operating mode before entering CAN sleep mode

In CAN sleep mode, the CAN module does not transmit messages even if the transfer request is issued or suspended.

(1) Enter CAN sleep mode

The CPU issues a CAN sleep mode conversion request by writing 01B to the PSMODE [1:0] bit of the CnCTRL register.

This conversion request is Acknowledged only under the following conditions.

- The CAN module is already operating in the following modes
 - Normal operating mode
 - Normal mode of operation with ABT
 - Only accept modes
 - Single-shot mode
 - Self-test mode
 - Stop mode in all of the above operating modes
- The CAN bus is idle (bit 4 in interframe space is invisible) ^{Note}
- There are no pending transfer requests

Note: If the CAN bus is fixed as dominant, requests transitioning to CAN sleep mode will be in a pending state.

In addition, the transition from CAN stop mode to CAN sleep mode is also independent of the CAN bus state.

Note: If a sleep mode request is pending and a message is received in the mailbox at the same time, the sleep mode request is not canceled, but is executed as soon as the message store is complete. If the CPU will perform an RX interrupt, this may result in FCAN in sleep mode. Therefore, if the interrupt is in sleep mode, the interrupt must use the MBON flag to check access to the packet buffer as well as to the receive history list register

If any of the above conditions are not met, the CAN module will run as follows:

- If the request enters CAN sleep mode from initialization mode, the CAN sleep mode conversion request is ignored and the CAN module remains in initialization mode.
- If the CAN bus state is not bus idle (i.e. the CAN bus state is transmit or receive), the CAN sleep mode cannot be immediately converted to CAN sleep mode when requested in one of the operating modes. In this case, the CAN sleep mode transition request remains suspended until the CAN bus state becomes bus idle (bit 4 of the interframe space is the recessive bit). During the time from the can sleep mode request to the successful conversion, the PSMODE [1:0] bit remains 00B. When the module enters CAN sleep mode, the PSMODE [1:0] bit is set to 01B. .
- If a transition request to initialization mode and a request to transition to CAN sleep mode are issued at the same time while the CAN module is in one of the operating modes, the request for initialization mode transition is enabled. The CAN module enters initialization mode at a predetermined time. At this point, the CAN sleep mode request does not remain pending, but is ignored.
- Even if both initialization mode and sleep mode are not requested at the same time (that is, the first request is not granted when the second request is issued), the initialization request has priority over the sleep mode request. When you request initialization mode, the sleep mode request is canceled. When a pending request for the initial mode appears, the subsequent requests for the sleep mode are deleted when published.

(2) The state of the CAN sleep mode

The CAN module is in the following state after entering CAN sleep mode

- The internal operating clock has stopped with minimal power consumption
- The CAN acceptance pin (CRxD) of the detected falling edge function remains active to wake up the CAN module from the CAN bus.
- To wake up the CAN module from the CPU, data can be written to the PSMODE [1:0] of the CAN Module Control Register (CnCTRL), but no other CAN module registers or bits can be written.
- In addition to CnLIPT, CnRGPT, CnLOPT and CnTGPT, the other registers of the CAN module can be read.
- The CAN packet cache register cannot be written or read
- The MBON bit of the CAN0 Global Control Register (CnGMCTRL) is cleared.
- Requests for transition to initialization mode were not Acknowledged and were ignored.

(3) CAN sleep mode release

THE CAN sleep mode is released by the following events:

- When the CPU writes 00B to the PSMODE [1:0] bit of the CnCTRL register
- The CAN receive pin (CRxD) detects the falling edge (i.e. the CAN bus level goes from recessive to dominant).

Note 1 Even if the falling edge appears in the SOF of the received message, it will not be received and stored. If the CPU turns off the CAN clock while in CAN sleep mode. Even then, the CAN sleep mode will not be released and the PSMODE [1:0] will continue to be 01B unless the clock for CAN is provided again. In addition to this, received messages will not be received afterwards.

2. If the falling edge on the CAN receive pin (CRxD) is detected in the state where the CAN clock is provided, the PSMODE0 bit needs to be cleared by software (for details, see **processing in Figure 22-79**).

After the sleep mode is released, the CAN module returns to the operating mode when the CAN sleep mode was requested, and the PSMODE [1:0] bit of the CnCTRL register is reset to 00B. If the CAN sleep mode is released by a change in the STATE of the CAN bus, the CINTS5 bit of the CnINTS register is set to 1, regardless of the CIE bit of the CnIE register. After the CAN module is released from CAN sleep mode, it enters the CAN bus again by automatically detecting 11 consecutive recessive level bits on the CAN bus. The user application must wait until MBON=1 before accessing the packet buffer. When the CAN module is in CAN sleep mode, a transition request to initialization mode is issued, and the request is ignored. Before entering initialization mode, the CPU releases sleep mode through software

Note: Please note if the CAN sleep mode is released by the CAN bus event; Therefore, if a CAN bus event occurs, a wake-up interrupt can occur even after the sleep mode is requested.

Note m= 0 to 15

22.11.2 CAN stop mode

THE CAN STOP MODE CAN BE USED TO SET THE CAN CONTROLLER TO STANDBY MODE TO REDUCE POWER CONSUMPTION. The CAN module can only enter CAN stop mode from CAN sleep mode. The release of can stop mode requires the CAN module to be in CAN sleep mode.

CAN stop mode can only be released by writing 01B to the PSMODE [1:0] bit of the CnCTRL register, not by a change in the CAN bus state (entering CAN sleep mode). Even if a transmission request is issued or suspended, no message is transmitted.

(1) Enter CAN stop mode

A CAN stop mode conversion request is made by writing 11B to the PSMODE [1:0] bit of the CnCTRL register. CAN stop mode requests are Acknowledged only when the CAN module is in CAN sleep mode. Applications are ignored in other modes.

Note: To set the CAN module to CAN stop mode, the module must be in CAN sleep mode. To confirm that the module is in sleep mode, check that PSMODE [1:0] is 01B and request that CAN stop mode. If a bus change occurs at the CAN receive pin (CRxD) while performing this procedure, the CAN sleep mode is automatically released. In this case, the CAN stop mode conversion request cannot be Acknowledged (however, in the state where the CAN clock is provided, the PSMODE0 bit needs to be cleared by software after a bus change occurs on the CAN receive pin (CRxD)).

(2) CAN stop mode status

After entering CAN stop mode, the CAN module is in one of the following states.

- The internal operating clock stops with minimal power consumption
- To wake up the CAN module, data can be written to the PSMODE [1:0] bit of the CAN Module Control Register (CnCTRL), not to other registers of the CAN
- In addition to the CnLIPT, CnRGPT, CnLOPT, and CnTGPT registers, all other CAN0 registers can be read.
- The CAN0 packet cache register cannot be written or read
- The MBON bit of the CAN0 Global Control Register (CnGMCTRL) is cleared
- Requests to go to initialization mode are not Acknowledged and are ignored

(3) Releases CAN stop mode

THE CAN STOP MODE CAN ONLY BE RELEASED BY WRITING 01B TO THE PSMODE [1:0] BIT OF THE CnCTRL register. After releasing the CAN stop mode, the CAN module enters CAN sleep mode.

When the CAN module is in CAN stop mode, the request is converted to initialization mode, and the request is ignored; Before entering initialization mode. Before entering initialization mode, the CPU must release stop mode and subsequently release CAN sleep mode. It is not possible to enter other operating modes directly from CAN stop mode, and if you do not enter CAN sleep mode, the request will be ignored.

Note m= 0 to 15

22.11.3 Example of power saving mode

In some application systems, it may be necessary to put the CPU into power-saving mode to reduce power consumption. By using a power-down mode specific to the CAN module and a CPU-specific power-down mode, the CAN bus can wake up the CPU from the power-down state.

Below is an example of using power saving mode

First, put the CAN module into CAN sleep mode (PSMODE=01B). Next, put the CPU into power saving mode. If the CAN receive pin (CRxD) in this state detects edge transition from recessive to dominant, the CINTS5 bit in the CAN module is set to 1. If the CIE5 bit of the CnCTRL register is set to 1, a wake-up interrupt (INTCnWUP) is generated. The CAN module is automatically released from CAN sleep mode (PSMODE=00B) and returns to normal operating mode (however, in the state where a CAN clock is provided, the PSMODE0 bit needs to be cleared by software after the CAN receive pin (CRxD) detects a bus change.) The CPU responds to INTCnWUP and can release its own power saving mode and return to normal operation mode. .

To further reduce the CPU's power consumption, the internal clock (including the clock of the CAN module) may stop. In this case, after the CAN module is placed in CAN sleep mode, the operating clock provided to the CAN module will stop. The CPU then enters a power-down mode where the clock provided to the CPU stops. If the CAN receive pin (CRxD) in this state detects edge transitions from recessive to dominant, the CAN module can set the CINTS5 bit to 1 and generate a wake-up interrupt (INTCnWUP), even if no clock is provided. However, the other features no longer work because the clock supply for the CAN module has stopped and the module remains in CAN sleep mode. The CPU responds to INTCnWUP, releases its power-down mode, restores power to the internal clock (including supplying the clock to the CAN module) after the oscillation settling time has elapsed, and begins executing instructions. As soon as the clock power is restored, the CAN module is released from CAN sleep mode and returns to normal operating mode (PSMODE=00B).

22.12 Interrupt function

The CAN module provides 6 different interrupt sources.

The occurrence of these interrupt sources is stored in the interrupt status register. Four separate interrupt request signals are generated from six interrupt sources. When generating an interrupt request signal that corresponds to two or more interrupt sources, you can use the interrupt status register to identify the interrupt source. After an interrupt source occurs, the software must clear the corresponding interrupt status bit to 0.

Table22-20. CAN Module interrupt source list

No.	Interrupt status bit		Interrupt enable bit		Interrupt the application signal	Description of the interrupt source
	name	register	name	register		
1	CINTS0 ^{Note}	CnINTS	CIE0 ^{Note}	Miss	INTCnTRX	The packet frame is successfully transmitted from the packet cache
2	CINTS1 ^{Note}	CnINTS	CIE1 ^{Note}	Miss	INTCnREC	The message cache receives a valid message frame
3	CINTS2	CnINTS	CIE2	Miss	INTCnERR	CAN Module Status Error Interrupt (Supplement 1).
4	CINTS3	CnINTS	CIE3	Miss		CAN module protocol error interrupt (supplement 2).
5	CINTS4	CnINTS	CIE4	Miss		CAN module Arbitration loss interrupt
6	CINTS5	CnINTS	CIE5	Miss	INTCnWUP	The CAN module wakes up interrupts from sleep mode (Supplement 3).

Note The IE bit (packet buffer interrupt enabled bit) in the CnMCTRLm register of the corresponding packet buffer must be set to 1 so that the packet buffer participates in interrupt generation. .

Supplement 1 This interrupt is generated when the Send/Receive Error counter is at the warning level or is in a passive error or bus shutdown state.

- This interrupt is generated when a content error, form error, ACK error, bit error, or CRC error occurs.
- This interrupt is generated when the CAN module wakes up from CAN sleep mode because a falling edge is detected at the CAN receive pin (the CAN bus transitions from recessive to dominant).

Note m=0 to 15

22.13 Diagnostic functions and special operating modes

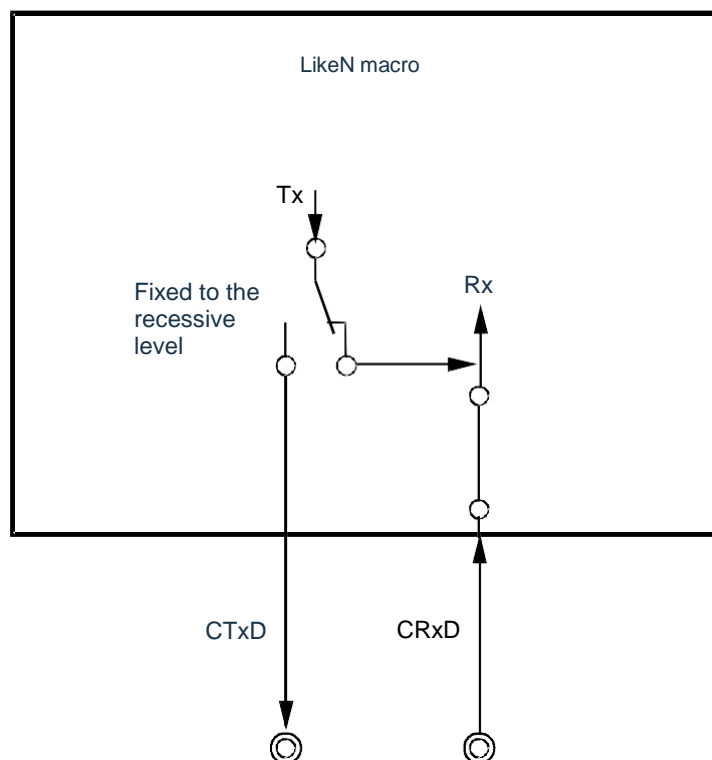
The CAN module provides receive-only mode, single-shot mode and self-test mode to support the operation of the CAN bus diagnostic function or specific CAN communication methods.

22.13.1 Receive mode only

Receive-only mode is used to monitor incoming messages without any interference with the CAN bus and can be used for the CAN bus analysis node.

For example, this mode can be used for automatic baud rate detection. The baud rate in the CAN module will change until "valid receive" is detected, so the module matches the baud rate in it ("valid receive" means that a packet frame is received in the CAN protocol layer without errors and there is a node with the appropriate ACK connected to the CAN bus). Valid receive does not require message frames to be stored in the receive message buffer (data frame) or the transmit message buffer (remote frame). Indicate a valid receive event by setting the VALID bit of cnCTRL register (1).

Figure 22-57. CAN mode only receives terminal connections in receive mode



In receive-only mode, message frames cannot be transmitted from the CAN module to the CAN bus. Packet caching is defined as sending a transition request for packet caching to be pending.

In receive-only mode, the CAN transmit pin (CTxD) in the CAN module is fixed to the recessive level. Therefore, when a message frame is received, no active error flag can be sent from the CAN module to the CAN bus even if a CAN bus error is detected. Since the transfer cannot be sent from the CAN module, the transfer error counter TEC is never updated. Therefore, the CAN module in receive-only mode does not enter the bus shutdown state.

In addition, after the packet frame is validly received, the ACK does not return to the CAN bus in this mode. Internally, the local node recognizes that it has transmitted a reply (ACK). Unable to transfer overloaded frames to the CAN bus.

Note: If only two CAN nodes are connected to the CAN bus and one of the nodes is running in receive-only mode, there is no ACK on the CAN bus. Due to the lack of an ACK, the transport node transmits the active error flag and sends the message frame repeatedly. The transport node becomes a passive error after sending the message frame 16 times (assuming that the error counter starts with 0 and no other errors occur). After the 17th packet frame is transmitted, the transmission node generates a passive error flag. The receive node in receive-only mode now detects the first valid message frame, and the VALID bit is set to 1 for the first time.

22.13.2 Single-shot mode

In single-pass mode, the automatic retransmission defined in the CAN protocol is turned off (under the CAN protocol, packet frame transmissions that are aborted due to loss of arbitration or the occurrence of an error must be repeated (without software control)). All other behaviors in single-shot mode are the same as in normal operating mode. Single-shot mode cannot be used with regular mode with ABT.

Single-pass mode disables retransmissions that abort packet frame transmissions based on the AL bit setting of the CnCTRL register. When the AL bit is cleared to 0, the Arbitration is lost and the retransmission is disabled when an error occurs. If the AL bit is set to 1, retransmission is disabled when an error occurs, but retransmission is enabled when Arbitration is lost. Therefore, the following event is defined as a TRQ bit in the packet buffer of the send packet buffer cleared to 0.

- The message frame was successfully sent
- Arbitration loss occurs for sending packet frames
- An error occurred while sending a message frame

By examining the CNNTS4 and CNTS3 bits of the CnINTS registers separately, it is possible to distinguish between event arbitration losses and error occurrences, and the type of error can be identified by reading the LEC [2:0] bit of the CnLEC register.

After the packet frame is successfully transmitted, the transmission of the CnINTS register completes with the interrupt bit CINTS0 set to 1. If the CIE0 bit of the CnIE register is set to 1 at this point, an interrupt request signal is output.

Single-pass mode can be used when simulating time-triggered communication methods such as TTCAN level 1.

Note: The AL bit is only valid in single-shot mode. The retransmission operation does not affect arbitrate losses in other operating modes.

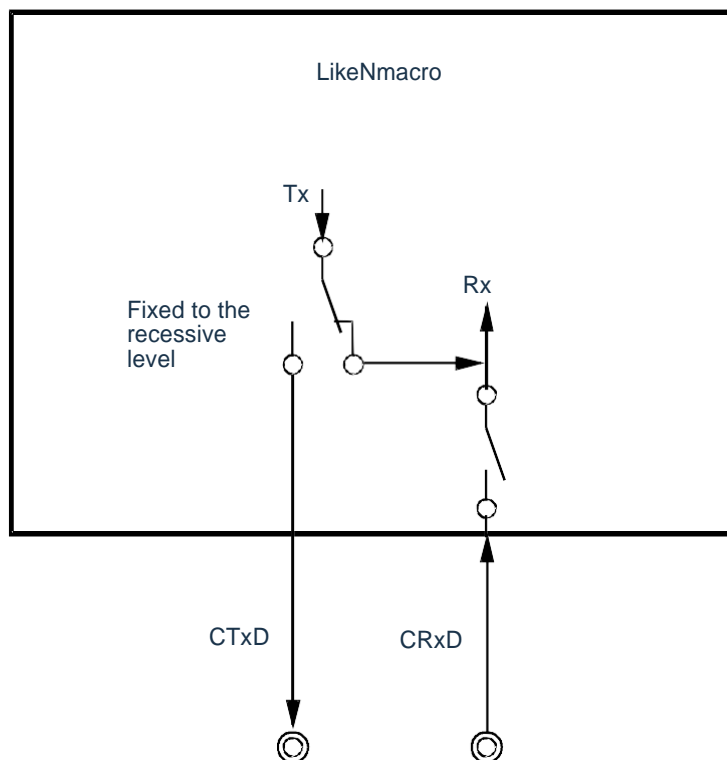
22.13.3 Self-test mode

In self-test mode, packet frame transmission and reception can be tested without connecting the CAN node to the CAN bus without affecting the CAN bus.

In self-test mode, the CAN module is completely disconnected from the CAN bus, but the transmission and reception loop internally. The CAN transmit pin (CTxD) is fixed to the recessive level.

If the upper descending edge of the CAN receive pin (CRxD) is detected after the CAN module enters CAN sleep mode from self-test mode, the module is released from CAN sleep mode in the same way as other modes of operation. (However, to release the CAN sleep mode, the PSMODE0 bit needs to be cleared by software after the falling edge on the CAN receive pin (CRxD) is detected in the state where the CAN clock is provided.) To keep the module in CAN sleep mode, the CAN receive pin (CRxD) needs to be used as the port pin.

Figure 22-58. Can module terminal connection in self-test mode



22.13.4 Receive/send operations in operation mode

Table 22-21 shows a summary of receive/send operations for each mode of operation.

Table 22-21. Send/receive profiles in operation mode

Mode of operation	Send data/remote frames	Send a reply	Send error/overload frames	Transfer retry	Automatic Block Transfer (ABT).	Sets the VALID bit	Store data to the message cache
Initialization mode	not	not	not	not	not	not	not
Normal operating mode	be	be	be	be	not	be	be
Normal mode of operation with ABT	be	be	be	be	be	be	be
Receive mode only	not	not	not	not	not	be	be
Single-shot mode	be	be	be	No ^{Note1}	not	be	be
Self-test mode e	It's Note2	It's Note2	It's Note2	It's Note2	not	It's Note2	It's Note2

Notes1. When the Arbitration is lost, the AL bit of the CnCTRL register can control the retransmission.

2. Each signal is not generated externally, but into the CAN module.

22.14 Timestamp function

CAN is an asynchronous serial protocol. All nodes connected to the CAN bus have local autonomous clocks. Therefore, the clocks of the nodes have no relationship (that is, the clocks are asynchronous and may have different frequencies).

However, in some applications, a public timebase needs to be established over the network (= global timebase). In order to establish a global time base, the timestamp function is required. The basic mechanism of the timestamp function is to capture the timer value triggered by the signal on the CAN bus.

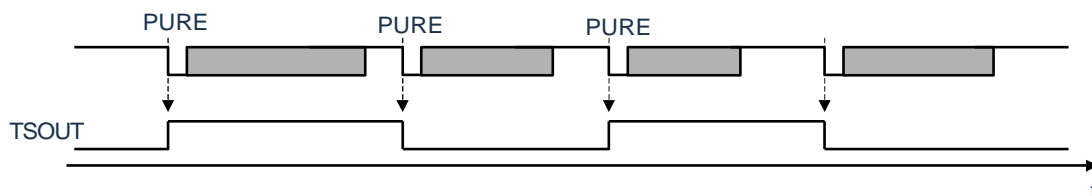
22.14.1 Timestamp function

The CAN controller supports capturing timer values triggered by specific frames. In addition to the CAN controller, an on-chip 16-bit capture timer unit in a microcontroller system is used. The 16-bit capture timer unit captures the timer value based on the trigger signal (TSOUT) and is used to capture the timer value output when receiving a data frame from the CAN controller. The CPU can retrieve the time of occurrence of the capture event by reading the captured value, that is, the timestamp of the message received from the CAN bus. The TSOUT signal can be selected from the following two event sources and specified by the TSSEL bit of the CnTS register.

- SOF event (frame start) (TSSEL=0).
- EOF event (last bit of the end frame) (TSSEL=1).

TSOUT signal enable requires setting the TSEN bit of the CnTS register to 1.

Figure 22-59. Timing diagram of the capture signal TSOUT



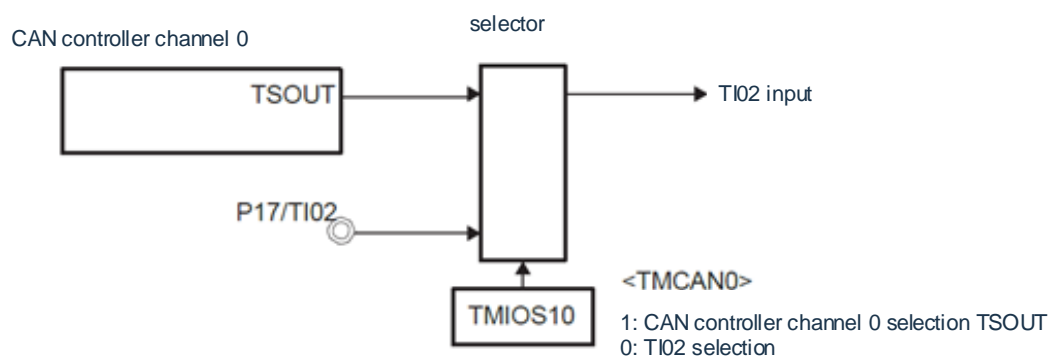
The TSOUT signal switches its level when a selected event occurs during data frame reception (in the sequence diagram above, the SOF is used as the trigger event source). To use the TSOUT signal capture timer value, the runtime must detect the capture signal on the rising and falling edges.

This timestamp function is controlled by the TSLOCK bit of the CnTS register. When TSLOCK is cleared to 0, the TSOUT bit switches when the selected event occurs. If the TSLOCK bit is set to 1, TSOUT will switch when the selected event occurs, but the switch will stop because the TSEN bit is automatically cleared to 0 when the message is stored in the message buffer 0. This will suppress subsequent switching of TSOUT in order to save the timestamp value of the last switch (= last capture) as the timestamp value of the received data frame in the packet buffer 0.

Note: The timestamp function using the TSLOCK bit stops switching the TSOUT bit by receiving a data frame in packet buffer 0. Therefore, message buffer 0 must be set to receive packet buffer. Because the receive packet buffer cannot receive remote frames, it is not possible to stop switching the TSOUT bit by receiving remote frames. When a data frame is received in a packet buffer other than the message buffer (packet buffer 0), the switching of the TSOUT bit does not stop. For these reasons, when the CAN module is in normal mode of operation with ABT, it is impossible to receive data frames in message buffer 0 because message buffer 0 must be set to send message buffer. Therefore, in this mode of operation, the TSLOCK bit cannot be used to stop the function of toggling the TSOUT bit.

By switching the input source (using TMOS1), the capture trigger signal (TSOUT of CAN controller channel 0) can be input to channel 2 of timer array unit 0 without having to externally connect the TSOUT of CAN controller channel 0 and TI02.

Figure 22-60. the switch of the input sources



- Note 1 TIOS10: Bit 0 of time input selection register TIOS1 (see Chapter 6 General Purpose Timer Units).
2. Available pins vary by product. For more information, see Chapter 2 Pin Functions

22.15 Baud rate setting

22.15.1 Baud rate setting

Make sure to set within the limit value to ensure proper operation of the CAN controller as shown below.

- (a) $5TQ$ SPT(sample point) $17TQSPT=TSEG1+1TQ$
- (b) $8TQ$ DBT (Data Bit Time) $25TQ$
 $DBT=TSEG1+TSEG2+1TQ=TSEG2+SPT$
- (c) $1TQ$ SJW (Synchronous Jump Width) $4TQSJW$ DBT–
 SPT
- (d) $4TQ$ TSEG1 $16TQ[3(TSEG1[3:0]$ 15 setting value].
- (e) $1TQ$ TSEG2 $8TQ[0(setting value for TSEG2[2:0]$ 7].

Note $TQ=1/f_{TQ}$ (f_{TQ} : CAN protocol layer base system clock).
 TSEG1 [3:0]: Bits 0 to 3 of the CAN bit rate register (CnBTR).
 TSEG2 [2:0]: Bits 8 to 10 of the CAN bit rate register (CnBTR).

Table22-22shows the combination of bit rates that meet the above criteria.

Table22-22. Configurable bit rate combination(1/3)

Valid bit rate settings					CnBTR register setting value		Sample points (%) in %)
DBT length	Synchronize segments	Propagate segments	Phase segment 1	Phase segment 2	TSEG1[3:0]	TSEG2[2:0]	
25	1	8	8	8	1111	111	68.0
24	1	7	8	8	1110	111	66.7
24	1	9	7	7	1111	110	70.8
23	1	6	8	8	1101	111	65.2
23	1	8	7	7	1110	110	69.6
23	1	10	6	6	1111	101	73.9
22	1	5	8	8	1100	111	63.6
22	1	7	7	7	1101	110	68.2
22	1	9	6	6	1110	101	72.7
22	1	11	5	5	1111	100	77.3
21	1	4	8	8	1011	111	61.9
21	1	6	7	7	1100	110	66.7
21	1	8	6	6	1101	101	71.4
21	1	10	5	5	1110	100	76.2
21	1	12	4	4	1111	011	81.0
20	1	3	8	8	1010	111	60.0
20	1	5	7	7	1011	110	65.0
20	1	7	6	6	1100	101	70.0
20	1	9	5	5	1101	100	75.0
20	1	11	4	4	1110	011	80.0
20	1	13	3	3	1111	010	85.0
19	1	2	8	8	1001	111	57.9
19	1	4	7	7	1010	110	63.2
19	1	6	6	6	1011	101	68.4
19	1	8	5	5	1100	100	73.7
19	1	10	4	4	1101	011	78.9
19	1	12	3	3	1110	010	84.2
19	1	14	2	2	1111	001	89.5
18	1	1	8	8	1000	111	55.6
18	1	3	7	7	1001	110	61.1
18	1	5	6	6	1010	101	66.7
18	1	7	5	5	1011	100	72.2
18	1	9	4	4	1100	011	77.8
18	1	11	3	3	1101	010	83.3
18	1	13	2	2	1110	001	88.9
18	1	15	1	1	1111	000	94.4

Table22-22. Configurable bit rate combination (2/3)

Valid bit rate settings					CnBTR register setting value		Sample points (%) in %)
DBT length	Synchr nize	Propaga te	Phase segment 1	Phase segment 2	TSEG1[3:0]	TSEG2[2:0]	
17	1	2	7	7	1000	110	58.8
17	1	4	6	6	1001	101	64.7
17	1	6	5	5	1010	100	70.6
17	1	8	4	4	1011	011	76.5
17	1	10	3	3	1100	010	82.4
17	1	12	2	2	1101	001	88.2
17	1	14	1	1	1110	000	94.1
16	1	1	7	7	0111	110	56.3
16	1	3	6	6	1000	101	62.5
16	1	5	5	5	1001	100	68.8
16	1	7	4	4	1010	011	75.0
16	1	9	3	3	1011	010	81.3
16	1	11	2	2	1100	001	87.5
16	1	13	1	1	1101	000	93.8
15	1	2	6	6	0111	101	60.0
15	1	4	5	5	1000	100	66.7
15	1	6	4	4	1001	011	73.3
15	1	8	3	3	1010	010	80.0
15	1	10	2	2	1011	001	86.7
15	1	12	1	1	1100	000	93.3
14	1	1	6	6	0110	101	57.1
14	1	3	5	5	0111	100	64.3
14	1	5	4	4	1000	011	71.4
14	1	7	3	3	1001	010	78.6
14	1	9	2	2	1010	001	85.7
14	1	11	1	1	1011	000	92.9
13	1	2	5	5	0110	100	61.5
13	1	4	4	4	0111	011	69.2
13	1	6	3	3	1000	010	76.9
13	1	8	2	2	1001	001	84.6
13	1	10	1	1	1010	000	92.3
12	1	1	5	5	0101	100	58.3
12	1	3	4	4	0110	011	66.7
12	1	5	3	3	0111	010	75.0
12	1	7	2	2	1000	001	83.3
12	1	9	1	1	1001	000	91.7

Table22-22. Configurable bit rate combination (3/3)

Valid bit rate settings					CnBTR register setting		Sample points (%) in %)
DBT length	Synchronize segments	Propagate segments	Phase segment 1	Phase segment 2	TSEG1[3:0]	TSEG2[2:0]	
11	1	2	4	4	0101	011	63.6
11	1	4	3	3	0110	010	72.7
11	1	6	2	2	0111	001	81.8
11	1	8	1	1	1000	000	90.9
10	1	1	4	4	0100	011	60.0
10	1	3	3	3	0101	010	70.0
10	1	5	2	2	0110	001	80.0
10	1	7	1	1	0111	000	90.0
9	1	2	3	3	0100	010	66.7
9	1	4	2	2	0101	001	77.8
9	1	6	1	1	0110	000	88.9
8	1	1	3	3	0011	010	62.5
8	1	3	2	2	0100	001	75.0
8	1	5	1	1	0101	000	87.5
7 ^{Note}	1	2	2	2	0011	001	71.4
7 ^{Note}	1	4	1	1	0100	000	85.7
6 ^{Note}	1	1	2	2	0010	001	66.7
6 ^{Note}	1	3	1	1	0011	000	83.3
5 ^{Note}	1	2	1	1	0010	000	80.0
4 ^{Note}	1	1	1	1	0001	000	75.0

Note Setting DBT with a value of 7 or less is only valid if the value of the CnBRP register is not 00H.

Note: The values in Table22-22do not guarantee the operation of the network system. The efficiency of the network system is detected by oscillation error and delay in the CAN bus and CAN transceiver.

22.15.2 A representative example of baud rate settings

Table 22-23 and Table 22-24 show a representative example of the baud rate setting.

Table 22-23. Representative example of baud rate setting ($f_{CANMOD}=8MHz$) (1/2).

Set the baud rate value in kbps	The ratio of the division of cnBRP	CnBRP Register setting value	The effective bit rate setting (in kbps).					The CnBTR register setting value		Sample point (in %)
			DBT length	Synchroniz e segments	Propagate segments	Phase segment 1	Phase segment 2	TSEG1[3:0]	TSEG2[2:0]	
1000	1	00000000	8	1	1	3	3	0011	010	62.5
1000	1	00000000	8	1	3	2	2	0100	001	75.0
1000	1	00000000	8	1	5	1	1	0101	000	87.5
500	1	00000000	16	1	1	7	7	0111	110	56.3
500	1	00000000	16	1	3	6	6	1000	101	62.5
500	1	00000000	16	1	5	5	5	1001	100	68.8
500	1	00000000	16	1	7	4	4	1010	011	75.0
500	1	00000000	16	1	9	3	3	1011	010	81.3
500	1	00000000	16	1	11	2	2	1100	001	87.5
500	1	00000000	16	1	13	1	1	1101	000	93.8
500	2	00000001	8	1	1	3	3	0011	010	62.5
500	2	00000001	8	1	3	2	2	0100	001	75.0
500	2	00000001	8	1	5	1	1	0101	000	87.5
250	2	00000001	16	1	1	7	7	0111	110	56.3
250	2	00000001	16	1	3	6	6	1000	101	62.5
250	2	00000001	16	1	5	5	5	1001	100	68.8
250	2	00000001	16	1	7	4	4	1010	011	75.0
250	2	00000001	16	1	9	3	3	1011	010	81.3
250	2	00000001	16	1	11	2	2	1100	001	87.5
250	2	00000001	16	1	13	1	1	1101	000	93.8
250	4	00000011	8	1	3	2	2	0100	001	75.0
250	4	00000011	8	1	5	1	1	0101	000	87.5
125	4	00000011	16	1	1	7	7	0111	110	56.3
125	4	00000011	16	1	3	6	6	1000	101	62.5
125	4	00000011	16	1	5	5	5	1001	100	68.8
125	4	00000011	16	1	7	4	4	1010	011	75.0
125	4	00000011	16	1	9	3	3	1011	010	81.3
125	4	00000011	16	1	11	2	2	1100	001	87.5
125	4	00000011	16	1	13	1	1	1101	000	93.8
125	8	00000111	8	1	3	2	2	0100	001	75.0
125	8	00000111	8	1	5	1	1	0101	000	87.5

Note: Table 22-23 do not guarantee the operation of the network system. The efficiency of the network system is detected by the crystal oscillation error and the delay of the CAN bus and THE CAN transceiver.

Table22-23. Representative example of baud rate setting ($f_{CANMOD}=8MHz$) (2/2).

Set the baud rate value in kbps	The ratio of the division of cnBRP	CnBRP Register setting value	The effective bit rate setting (in kbps).					The CnBTR register setting value		Sample point (in %)
			DBT length	Synchronize segments	Propagate segments	Phase segment 1	Phase segment 2	TSEG1[3:0]	TSEG2[2:0]	
100	4	00000011	20	1	7	6	6	1100	101	70.0
100	4	00000011	20	1	9	5	5	1101	100	75.0
100	5	00000100	16	1	7	4	4	1010	011	75.0
100	5	00000100	16	1	9	3	3	1011	010	81.3
100	8	00000111	10	1	3	3	3	0101	010	70.0
100	8	00000111	10	1	5	2	2	0110	001	80.0
100	10	00001001	8	1	3	2	2	0100	001	75.0
100	10	00001001	8	1	5	1	1	0101	000	87.5
83.3	4	00000011	24	1	7	8	8	1110	111	66.7
83.3	4	00000011	24	1	9	7	7	1111	110	70.8
83.3	6	00000101	16	1	5	5	5	1001	100	68.8
83.3	6	00000101	16	1	7	4	4	1010	011	75.0
83.3	6	00000101	16	1	9	3	3	1011	010	81.3
83.3	6	00000101	16	1	11	2	2	1100	001	87.5
83.3	8	00000111	12	1	5	3	3	0111	010	75.0
83.3	8	00000111	12	1	7	2	2	1000	001	83.3
83.3	12	00001011	8	1	3	2	2	0100	001	75.0
83.3	12	00001011	8	1	5	1	1	0101	000	87.5
33.3	10	00001001	24	1	7	8	8	1110	111	66.7
33.3	10	00001001	24	1	9	7	7	1111	110	70.8
33.3	12	00001011	20	1	7	6	6	1100	101	70.0
33.3	12	00001011	20	1	9	5	5	1101	100	75.0
33.3	15	00001110	16	1	7	4	4	1010	011	75.0
33.3	15	00001110	16	1	9	3	3	1011	010	81.3
33.3	16	00001111	15	1	6	4	4	1001	011	73.3
33.3	16	00001111	15	1	8	3	3	1010	010	80.0
33.3	20	00010011	12	1	5	3	3	0111	010	75.0
33.3	20	00010011	12	1	7	2	2	1000	001	83.3
33.3	24	00010111	10	1	3	3	3	0101	010	70.0
33.3	24	00010111	10	1	5	2	2	0110	001	80.0
33.3	30	00011101	8	1	3	2	2	0100	001	75.0
33.3	30	00011101	8	1	5	1	1	0101	000	87.5

Note: Table 22-23do not guarantee the operation of the network system. The efficiency of the network system is detected by oscillation error and delay in the CAN bus and CAN transceiver.

Table 22-24. Representative example of baud rate setting ($f_{CANMOD}=16MHz$) (1/2).

Set the baud rate value in kbps	The ratio of the division of cnBRP	CnBRP Register setting value	The effective bit rate setting (in kbps).					The CnBTR register setting value		Sample point (in %)
			DBT length	Synchroniz e segments	Propagate segments	Phase segment 1	Phase segment 2	TSEG1 [3:0]	TSEG2 [2:0]	
1000	1	00000000	16	1	1	7	7	0111	110	56.3
1000	1	00000000	16	1	3	6	6	1000	101	62.5
1000	1	00000000	16	1	5	5	5	1001	100	68.8
1000	1	00000000	16	1	7	4	4	1010	011	75.0
1000	1	00000000	16	1	9	3	3	1011	010	81.3
1000	1	00000000	16	1	11	2	2	1100	001	87.5
1000	1	00000000	16	1	13	1	1	1101	000	93.8
1000	2	00000001	8	1	3	2	2	0100	001	75.0
1000	2	00000001	8	1	5	1	1	0101	000	87.5
500	2	00000001	16	1	1	7	7	0111	110	56.3
500	2	00000001	16	1	3	6	6	1000	101	62.5
500	2	00000001	16	1	5	5	5	1001	100	68.8
500	2	00000001	16	1	7	4	4	1010	011	75.0
500	2	00000001	16	1	9	3	3	1011	010	81.3
500	2	00000001	16	1	11	2	2	1100	001	87.5
500	2	00000001	16	1	13	1	1	1101	000	93.8
500	4	00000011	8	1	3	2	2	0100	001	75.0
500	4	00000011	8	1	5	1	1	0101	000	87.5
250	4	00000011	16	1	3	6	6	1000	101	62.5
250	4	00000011	16	1	5	5	5	1001	100	68.8
250	4	00000011	16	1	7	4	4	1010	011	75.0
250	4	00000011	16	1	9	3	3	1011	010	81.3
250	4	00000011	16	1	11	2	2	1100	001	87.5
250	8	00000111	8	1	3	2	2	0100	001	75.0
250	8	00000111	8	1	5	1	1	0101	000	87.5
125	8	00000111	16	1	3	6	6	1000	101	62.5
125	8	00000111	16	1	7	4	4	1010	011	75.0
125	8	00000111	16	1	9	3	3	1011	010	81.3
125	8	00000111	16	1	11	2	2	1100	001	87.5
125	16	00001111	8	1	3	2	2	0100	001	75.0
125	16	00001111	8	1	5	1	1	0101	000	87.5

Note: The values in Table 22-24 do not guarantee the operation of the network system. The efficiency of the network system is detected by oscillation error and delay in the CAN bus and CAN transceiver.

Table 22-24. Representative example of baud rate setting ($f_{CANMOD}=16MHz$) (2/2).

Set the baud rate value in kbps	The ratio of the division of cnBRP	CnBRP Register setting value	The effective bit rate setting (in kbps).					The CnBTR register setting value		Sample point (in %)
			DBT length	Synchroniz e segments	Propagate segments	Phase segment 1	Phase segment 2	TSEG1 [3:0]	TSEG2 [2:0]	
100	8	00000111	20	1	9	5	5	1101	100	75.0
100	8	00000111	20	1	11	4	4	1110	011	80.0
100	10	00001001	16	1	7	4	4	1010	011	75.0
100	10	00001001	16	1	9	3	3	1011	010	81.3
100	16	00001111	10	1	3	3	3	0101	010	70.0
100	16	00001111	10	1	5	2	2	0110	001	80.0
100	20	00010011	8	1	3	2	2	0100	001	75.0
83.3	8	00000111	24	1	7	8	8	1110	111	66.7
83.3	8	00000111	24	1	9	7	7	1111	110	70.8
83.3	12	00001011	16	1	7	4	4	1010	011	75.0
83.3	12	00001011	16	1	9	3	3	1011	010	81.3
83.3	12	00001011	16	1	11	2	2	1100	001	87.5
83.3	16	00001111	12	1	5	3	3	0111	010	75.0
83.3	16	00001111	12	1	7	2	2	1000	001	83.3
83.3	24	00010111	8	1	3	2	2	0100	001	75.0
83.3	24	00010111	8	1	5	1	1	0101	000	87.5
33.3	30	00011101	24	1	7	8	8	1110	111	66.7
33.3	30	00011101	24	1	9	7	7	1111	110	70.8
33.3	24	00010111	20	1	9	5	5	1101	100	75.0
33.3	24	00010111	20	1	11	4	4	1110	011	80.0
33.3	30	00011101	16	1	7	4	4	1010	011	75.0
33.3	30	00011101	16	1	9	3	3	1011	010	81.3
33.3	32	00011111	15	1	8	3	3	1010	010	80.0
33.3	32	00011111	15	1	10	2	2	1011	001	86.7
33.3	37	00100100	13	1	6	3	3	1000	010	76.9
33.3	37	00100100	13	1	8	2	2	1001	001	84.6
33.3	40	00100111	12	1	5	3	3	0111	010	75.0
33.3	40	00100111	12	1	7	2	2	1000	001	83.3
33.3	48	00101111	10	1	3	3	3	0101	010	70.0
33.3	48	00101111	10	1	5	2	2	0110	001	80.0
33.3	60	00111011	8	1	3	2	2	0100	001	75.0
33.3	60	00111011	8	1	5	1	1	0101	000	87.5

Note: The values in Table 22-24 do not guarantee the operation of the network system. The efficiency of the network system is detected by oscillation error and delay in the CAN bus and CAN transceiver.

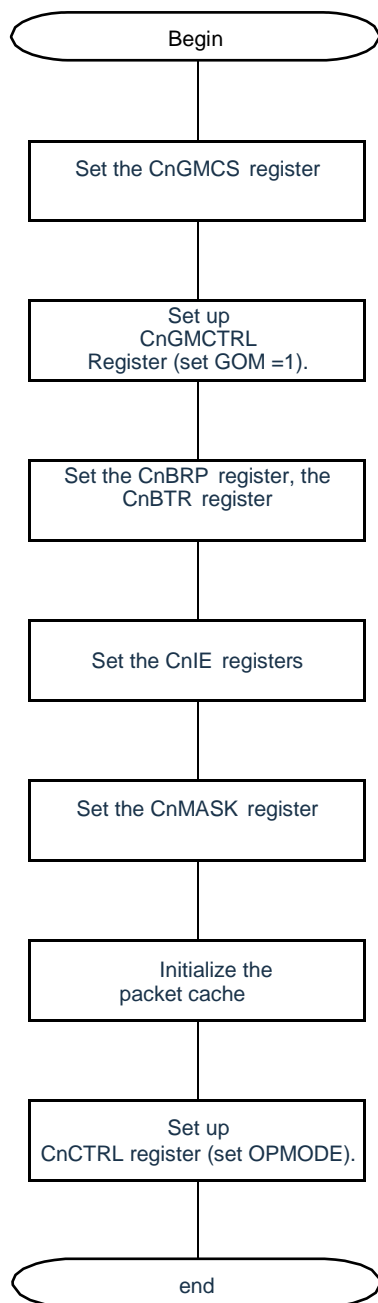
22.16 The operation of the CAN controller

The operating procedures in this chapter are the operational processing procedures for the CAN controller.

Please refer to the process of development in this chapter.

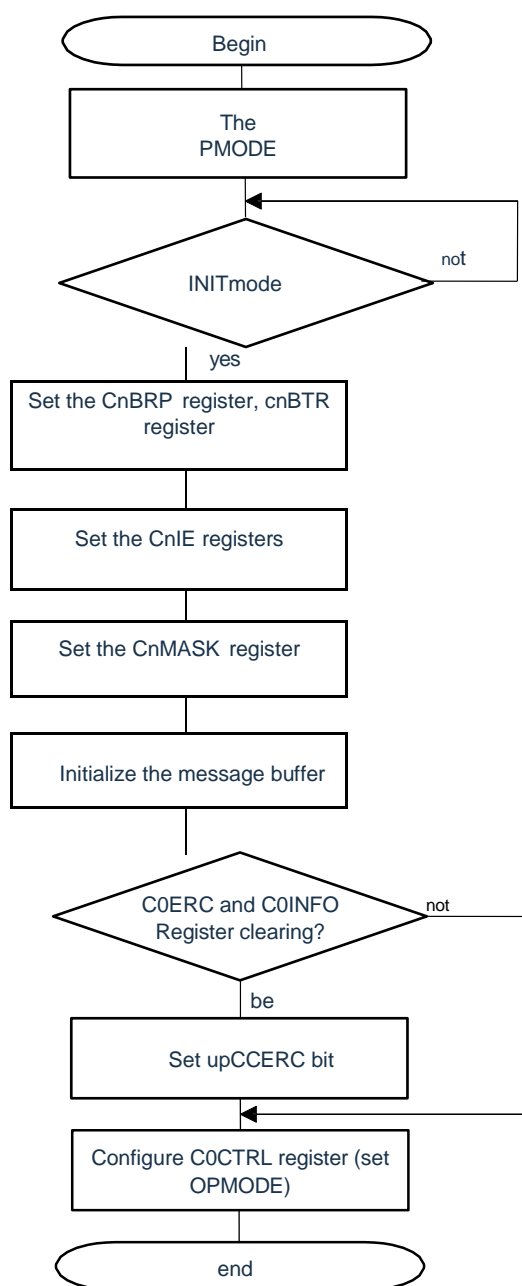
Note m=0 to 15

Fig.22-61.initialize



remark OPMODE: Normal operating mode, normal operating mode accompaniment Abbot, receive-only mode, single-shot mode, self-test mode

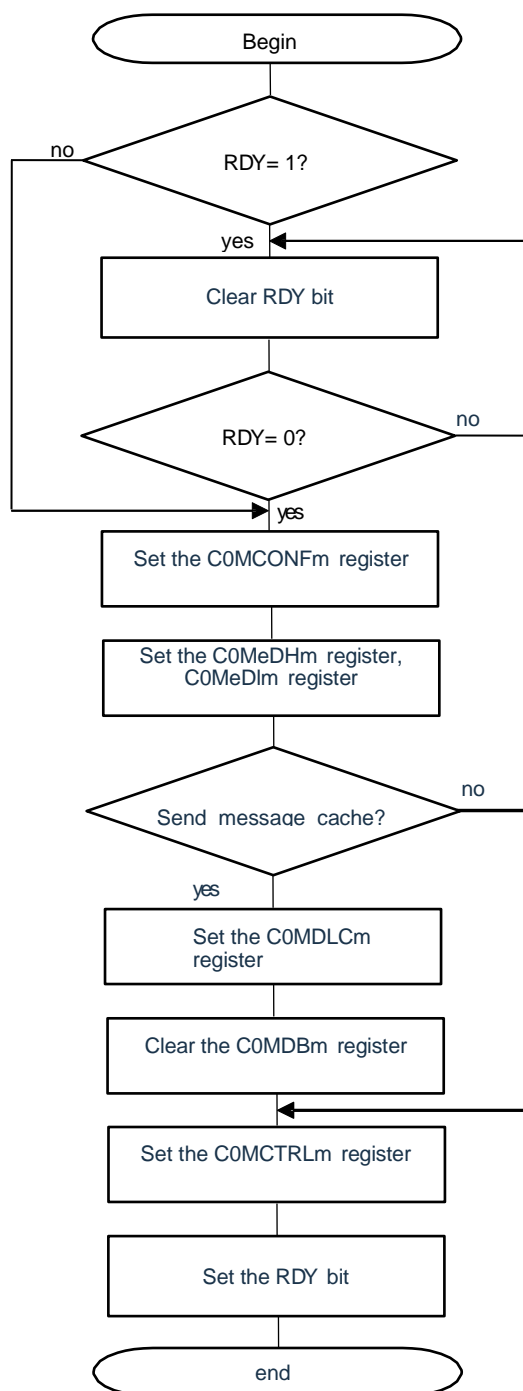
Fig.22-62.Reinitialize



Note: After the CAN module is set to initialization mode, do not immediately set it to another operating mode. If you need to immediately set the module to a different operating mode, visit CnCTRL and CnGMCTRL registers other than registers (for example, setting a message buffer).

Remarks OPMODE: normal operation mode, normal operation mode with ABT, receive only mode, single-shot mode, self-test mode.

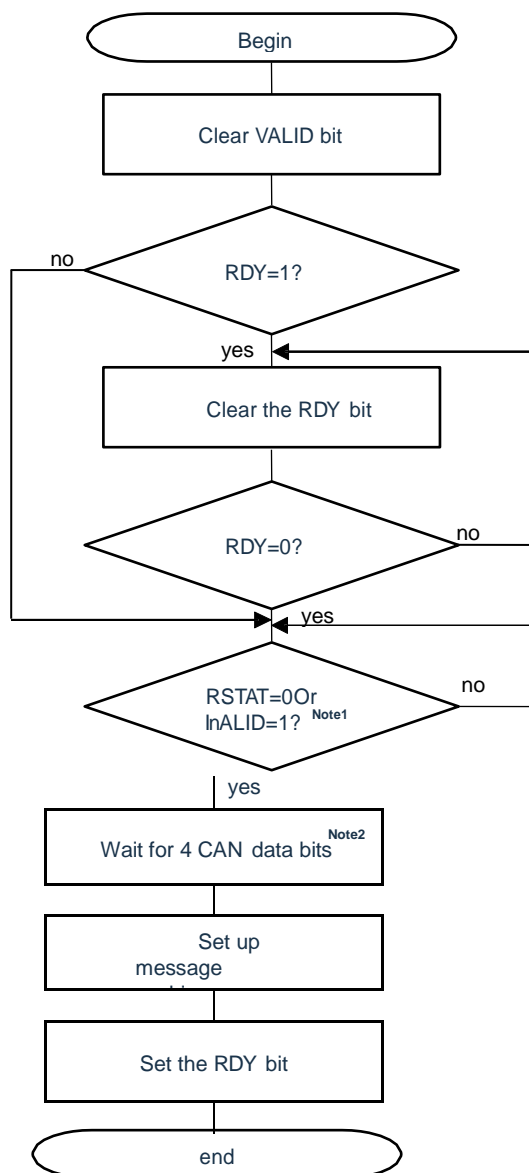
Figure 22-63. Packet cache initialization



- Note: 1. The RDY bit must be cleared before initializing the packet cache
2. Refer to the settings below for unused packet caching
- Clear the RDY, TRQ and DN bits of the CnMCTRLm register to 0
 - Clear the MA0 bit of the CnMCONFm register to 0

Fig.22-64 shows the receive packet cache processing (CnMCONFm register MT[2:0] bits = 001B to 101B).

Fig.22-64. Packet cache redefinition



Note 1. Acknowledgement of receipt of the message is due to the fact that the message has been fully receivedRDYBits are set.

2. Avoid redefining the packet buffer during a stored packet receive operation by waiting for an additional 4 CAN data bits.

Figure 22-65 shows the processing of the transmit packet buffer during transmission (CnMCONFm register MT [2:0] bits = 000B).

Figure 22-65. Packet cache redefinition during sending

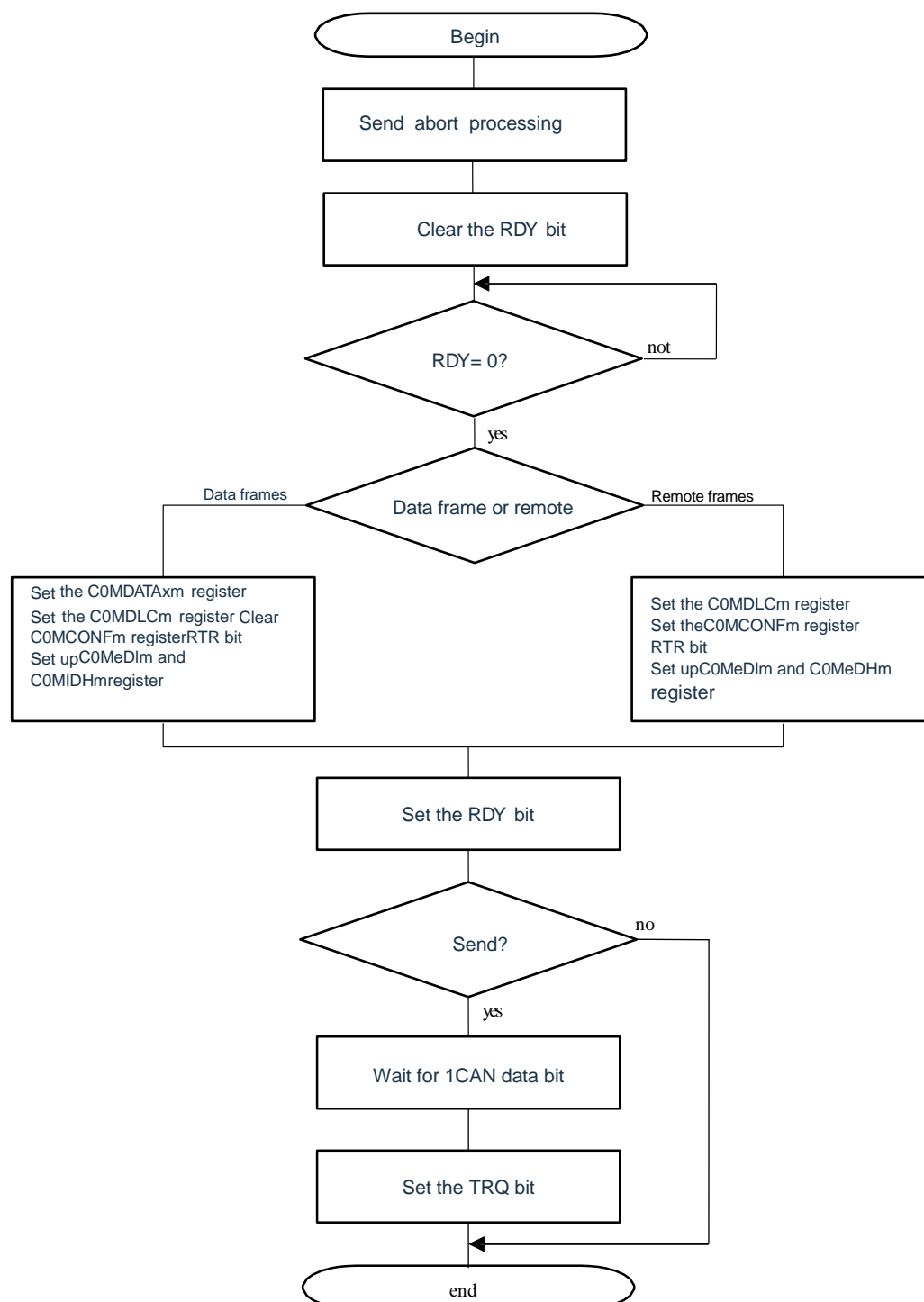
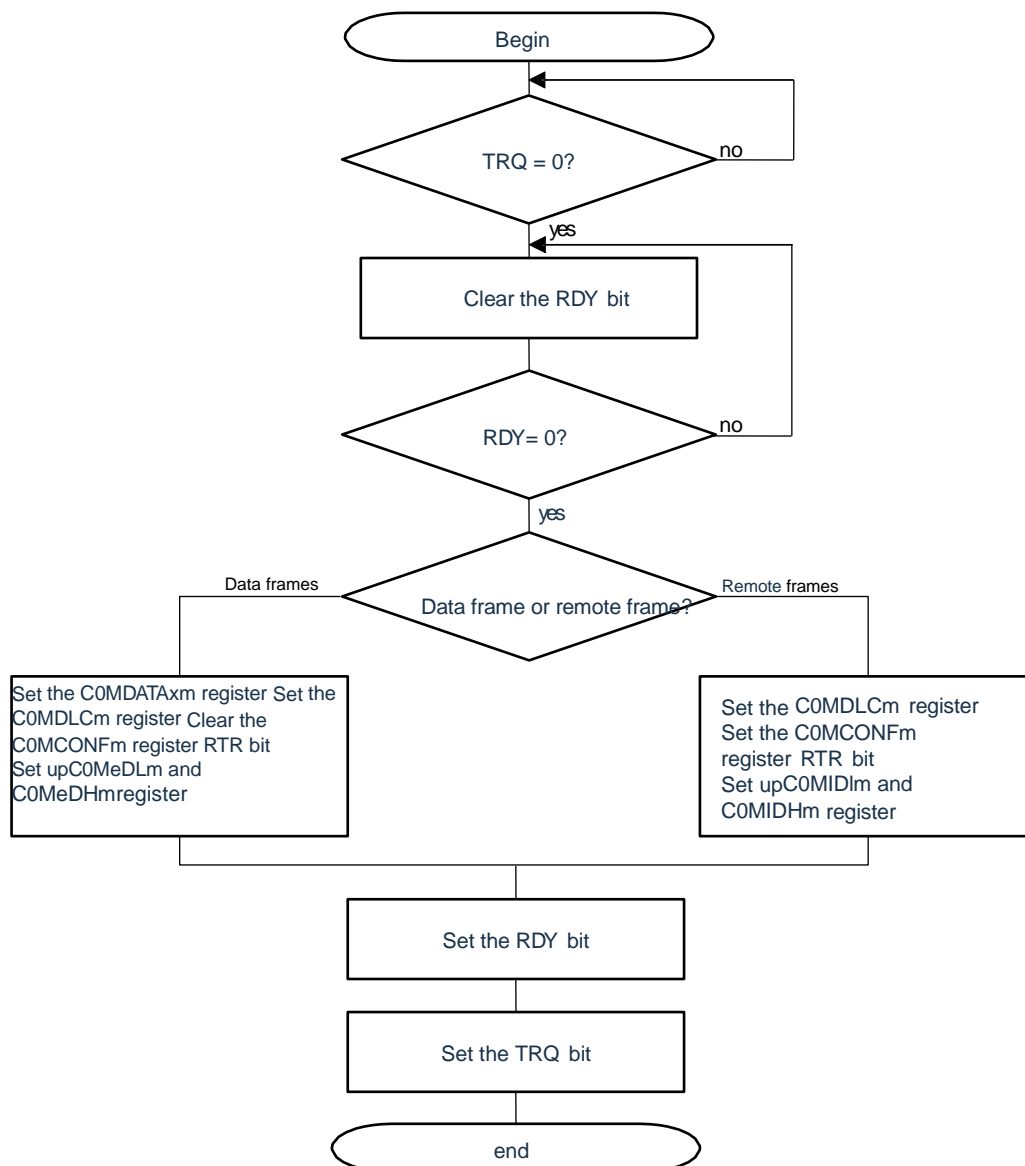


Figure 22-66 shows the transmit packet cache processing (CnMCONFm register MT [2:0] bit = 000B).

Figure 22-66. Packet sending processing

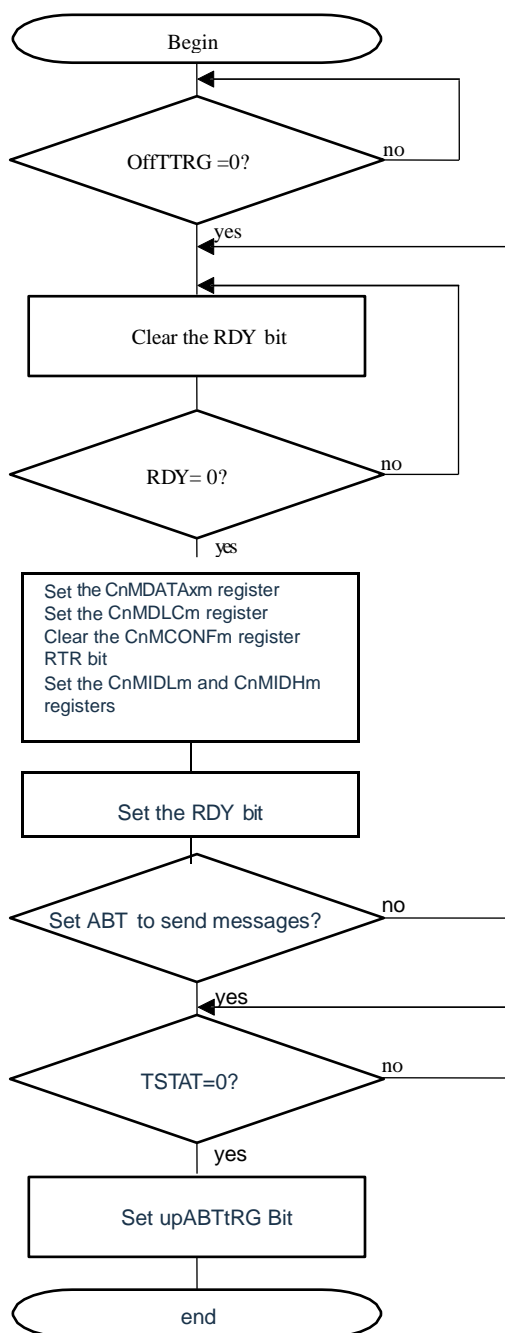


Note: 1 Set the TRQ bit after RDY settings

2. The RDY bit and the TRQ bit cannot be set at the same time

Fig.22-67 shows the cache processing of the transmit message (CnMCONFm register MT[2:0] bit = 000B).

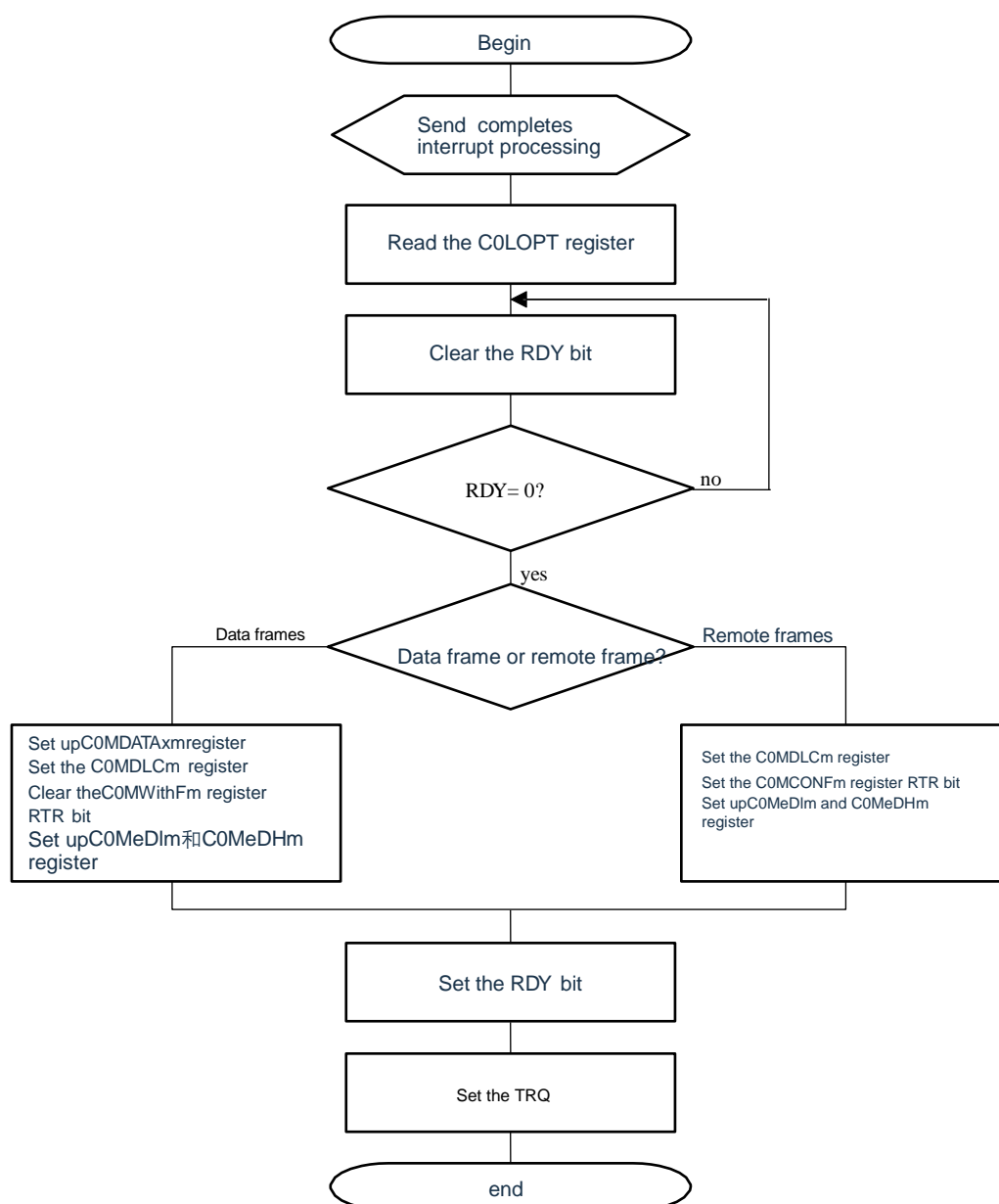
Fig.22-67. Abbot Packet sending processing



Note: After the TSTAT bit is cleared to 0, the ABTTRG bit should be set to 1. The TSTAT bit must be checked continuously and the ABTTRG bit set to 1. .

Note: This processing (using the normal operating mode of ABS) can only be applied to packet buffers 0 to 7. For packet buffers other than ABT packet buffers, see Figure 22-66.

Figure22-68. Send through interrupts (using the CnLOPT register).



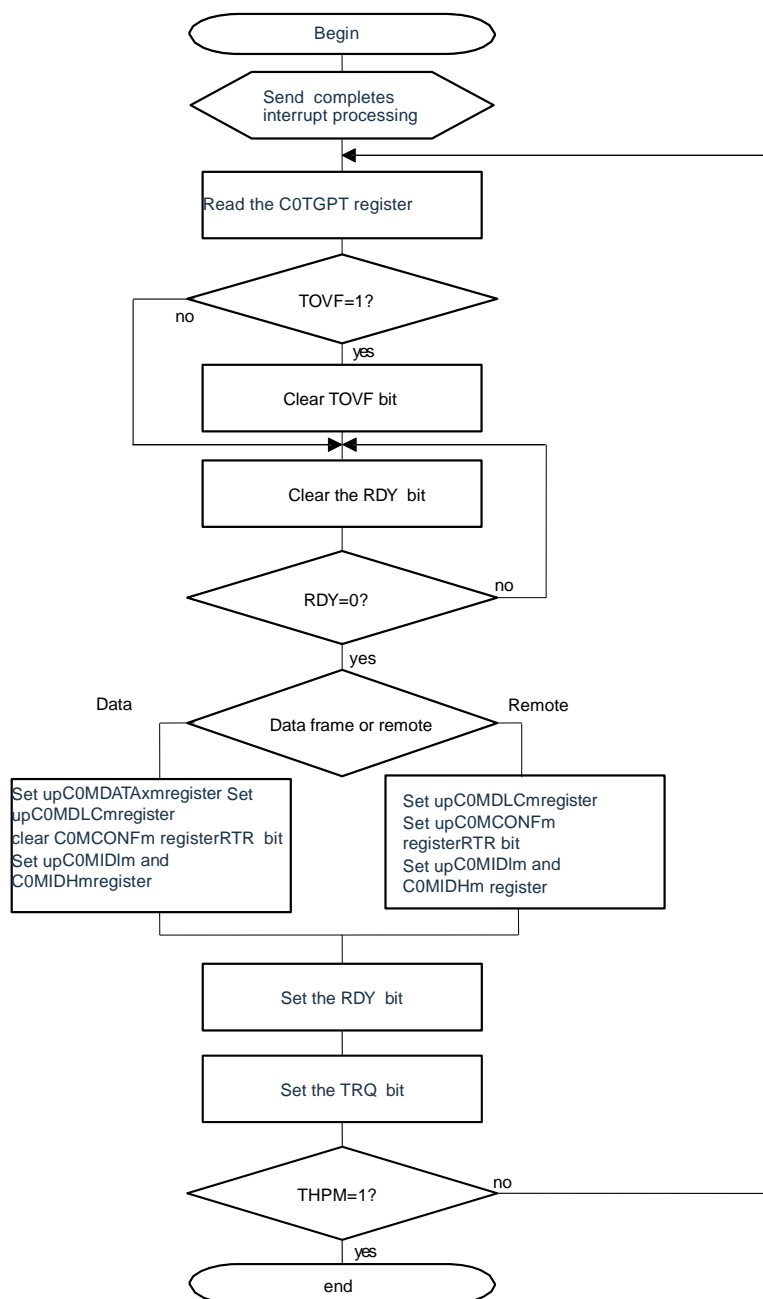
Note The 1.TRQ bit is set after the RDY setting

2. The RDY bit and the TRQ bit cannot be set at the same time

Note: Check the MBON flag at the beginning and end of the interrupt in order to check access to the message buffer and the TX history list register to prevent the execution of a pending sleep mode. If MBON is detected to be cleared, the result must be discarded and the operation processed again after MBON is set up again.

It is recommended to remove some sleep mode requests before handling TX interrupts.

Figure 22-69. Send via interrupt (using the CnTGPT register).



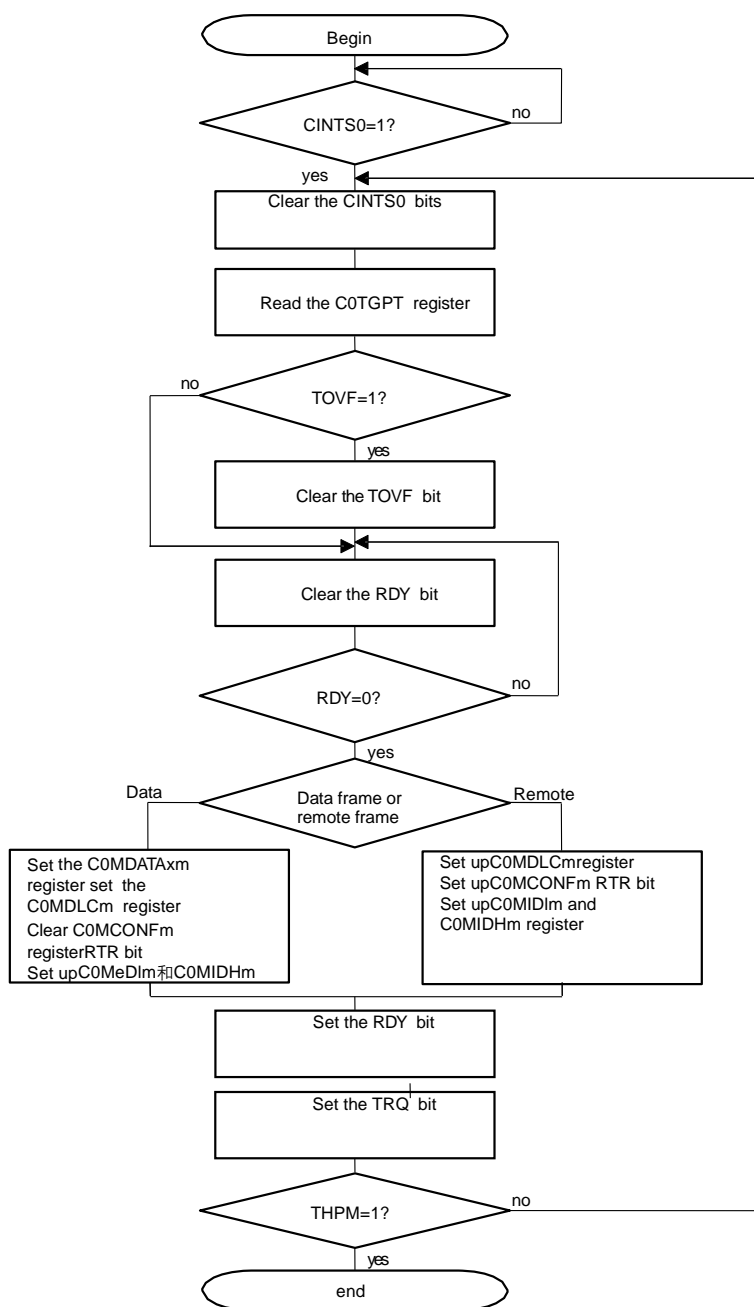
Note The setting of the TRQ bit is after the RDY bit is set

2. The RDY bit and the TRQ bit cannot be set at the same time

Note 1. Check the MBON flag at the beginning and end of the interrupt to check access to the message buffer and TX history list registers in case a suspended sleep mode is performed. If MBON is detected to be cleared, the result must be discarded and the operation processed again after MBON is set up again. It is recommended to remove some sleep mode requests before handling TX interrupts

2. If TOVF is set, the transmission history list is inconsistent. Consider scanning all configured transfer buffers to complete the transfer.

Figure 22-70. Transfer via software polling



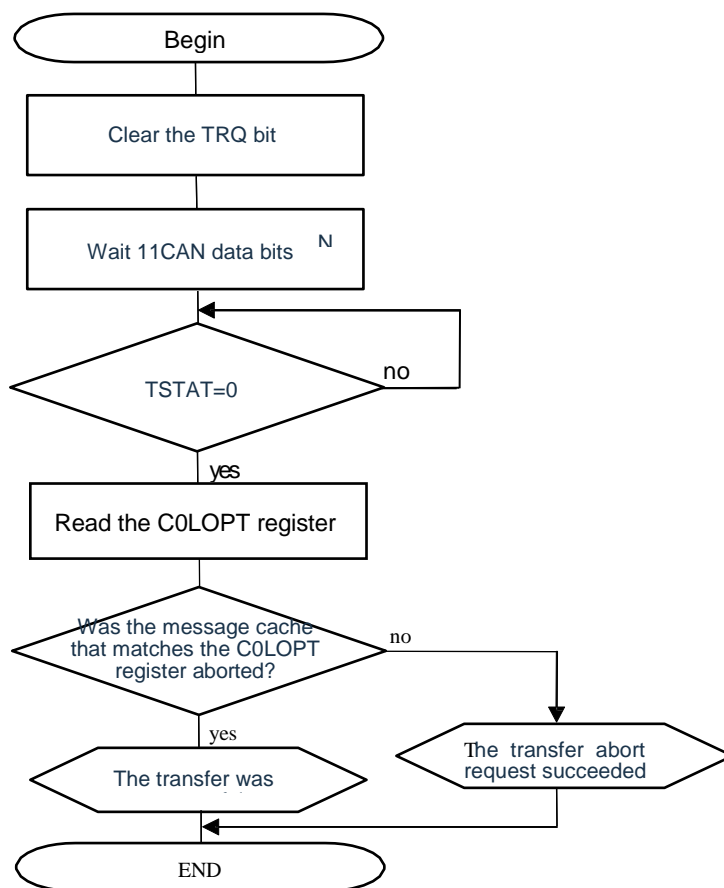
Note The setting of the TRQ bit is after the RDY bit is set

2. The RDY bit and the TRQ bit cannot be set at the same time

Note 1 Check the MBON flag at the beginning and end of the interrupt to check access to the message buffer and TX history list registers in case a suspended sleep mode is performed. If MBON is detected to be cleared, the result must be discarded and the operation processed again after MBON is set up again

2. If TOVF is set, the transmission history list is inconsistent. Consider scanning all configured transfer buffers to complete the transfer

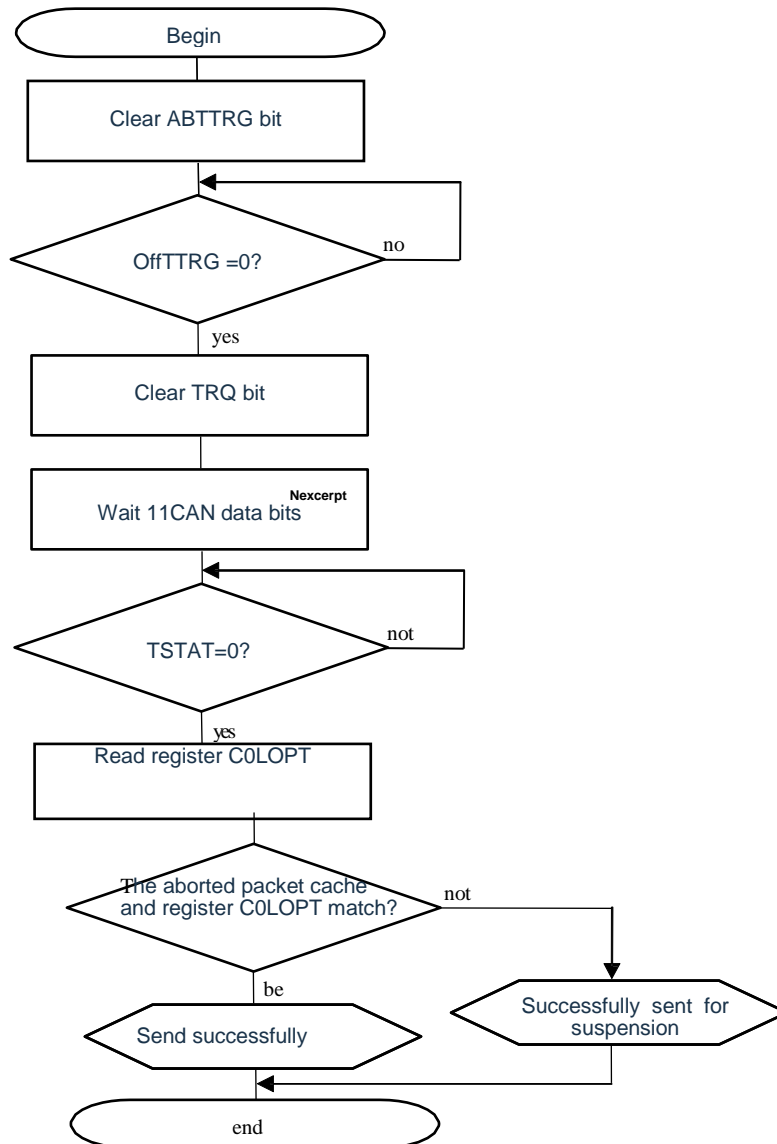
Figure 22-71 Transmission abort processing (except for normal operating mode with ABT).



Note Even if the TRQ bit is cleared, it is possible to start the transfer without aborting, because transfer requests to the protocol layer may have been accepted between 11 bits, inter-frame space (3 bits), and suspended transfers (8 bits).

- Notes:
1. Performs abort processing of the transfer request by clearing the TRQ bit instead of the RDY bit.
 2. Before you issue a sleep mode transition request, verify that there are no remaining transfer requests using this process.
 3. The user application can check the TSTAT bit periodically, or it can check after the transmission is complete and interrupted.
 4. While transfer abort processing is in progress, do not perform new transfer requests, including in other packet buffers.
 5. When a transmission continues in the same packet cache or only one packet cache is used, determining whether the transfer abort request is successful may cause a conflict. In this case, the history information indicated by the CnTGPT register is used to judge.

Figure 22-72 Transmit abort processing (except for ABT transmissions) (normal operation with ABT).

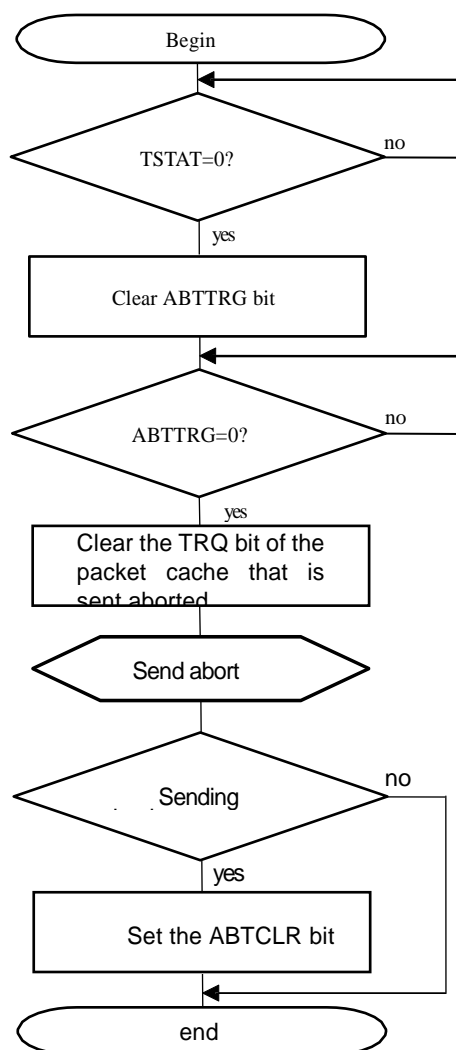


Note even though TRQ The bits are cleared, and it is also possible to initiate a send without aborting because 11Bit, inter-frame space (3bits) and suspend transfers (8bits) between which a transfer request to the protocol layer may have been accepted.

- Note:
1. Perform the transfer request abort processing by clearing the TRQ bit instead of the RDY bit.
 2. Before issuing a sleep mode transition request, verify that no transfer requests use this processing
 3. The user application can check the TSTAT bit periodically, or it can check after the transmission is complete and interrupted.
 4. While transfer abort processing is in progress, do not perform new transfer requests, including in other packet buffers.
 5. When the transmission of the same packet buffer is contiguous or uses only one packet buffer, judging whether the transfer abort request is successful may lead to a conflict. In this case, the historical information indicated by the CnTGPT register is used to judge.

Figure 22-73 shows that the transmission packet stops when the transmission ABT packet cache is aborted without skipping the recovery process

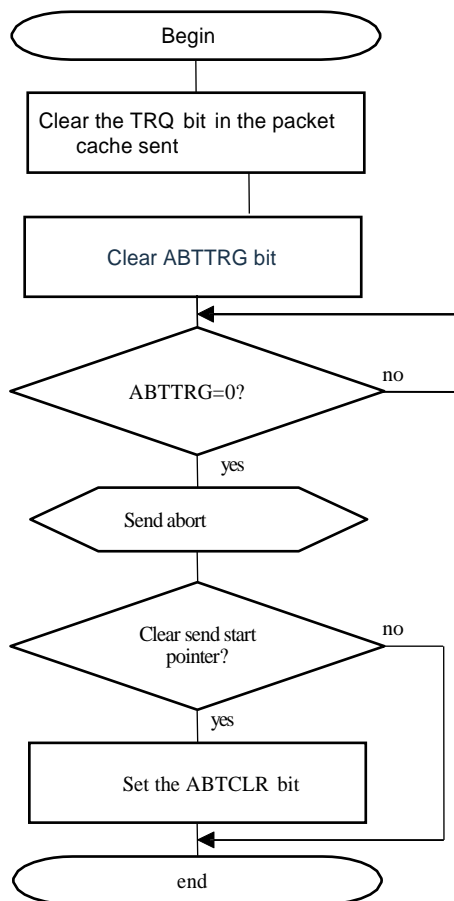
Figure 22-73. ABT transmits abort processing (normal mode of operation with ABT).



- Note: 1. Do not set any transfer requests when ABT transfer abort processing is in progress
2. After the ABTTRG bit is cleared (after ABT mode is aborted), a CAN sleep mode/CAN stop mode conversion request is issued, as shown in Figure 22-73 or Figure 22-74. When clearing a transfer request in a zone other than the ABT zone, follow the procedure shown in Figure 22-72.

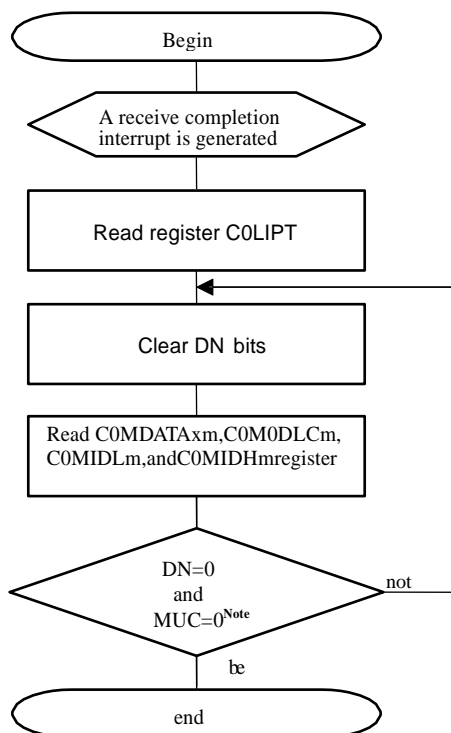
Figure 22-74 shows that the processing of the sending message stops skipping the recovery when the transmission ABT packet cache is aborted. .

Figure 22-74. ABT sends a request to abort processing (normal mode of operation with ABT).



- Note: 1. Do not set any transfer requests while ABT transfer abort processing is in progress.
2. Issue a CAN sleep mode/CAN stop mode request after ABTTRG is cleared (stop in ABT mode), following Figure 22-7322-73 Figure 22-74. When clearing a transfer request in an area other than the ABT area, follow the procedure shown in Figure 22-72. .

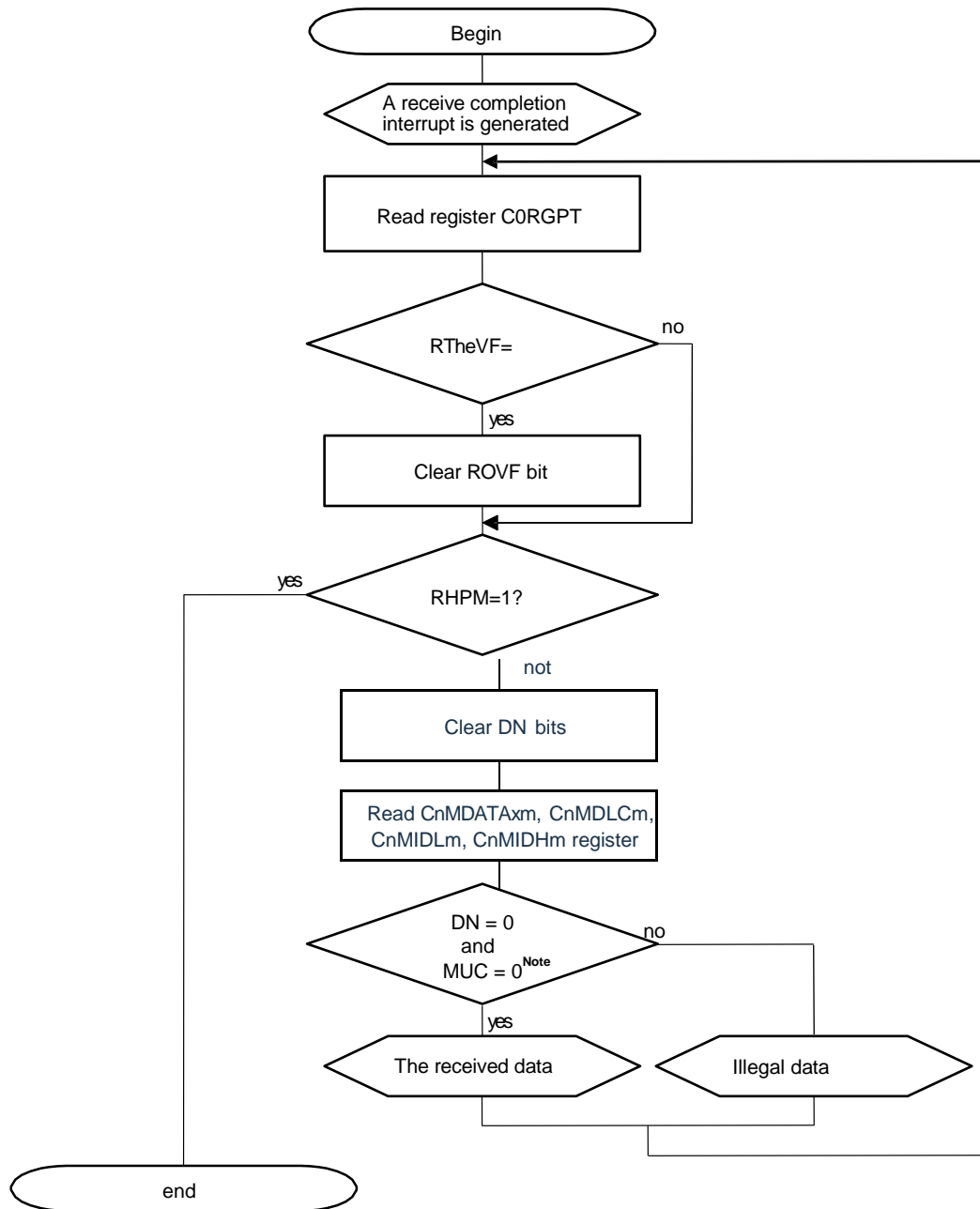
Figure 22-75. Receive via interrupt (using register CnLIPT).



Note Use read checks for MUC and DN bits

Note: To check the MBON flags at the beginning and end of an interrupt in order to check access to the message buffer and the receive history list register in case of a pending execution of sleep mode. If MBON is detected to be cleared, after setting MBON again, the actions and results of the processing must be discarded before processing. Before handling an RX interrupt, it is recommended to cancel any sleep mode requests.

Fig.22-76. Receive through interrupts(Use registers CnRGPT)

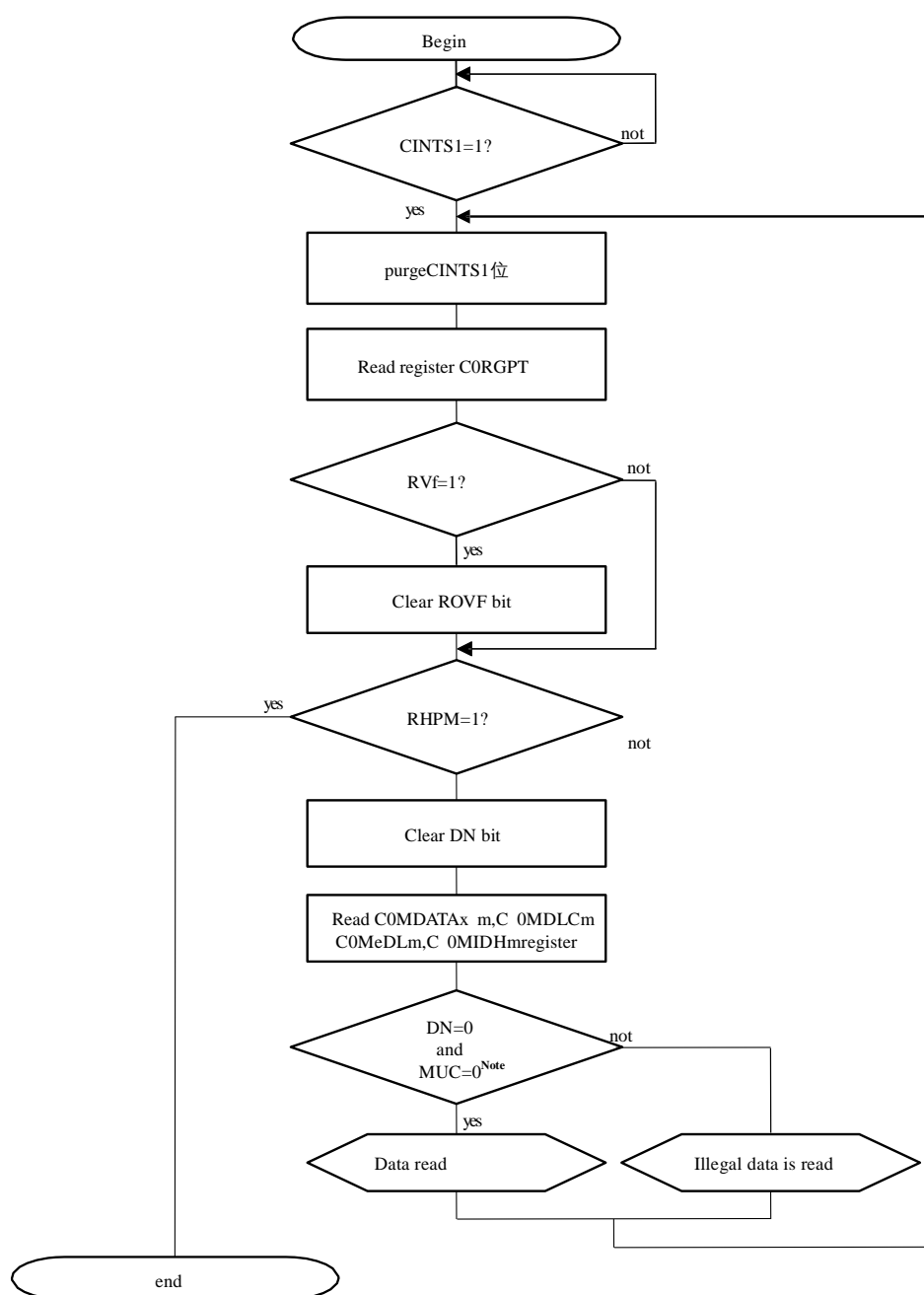


Note Uses Read Check MUC and DN Bits

Note 1 To check the MBON flags at the beginning and end of an interrupt in order to check access to the message buffer and the receive history list register in case of a pending execution of sleep mode. If MBON is detected to be cleared, after setting MBON again, the actions and results of the processing must be discarded before processing. Before handling the RX interrupt, it is recommended to cancel any sleep mode requests

2. If ROVF is set, the receive history list is inconsistent. Consider scanning all configured receive buffers for receive.

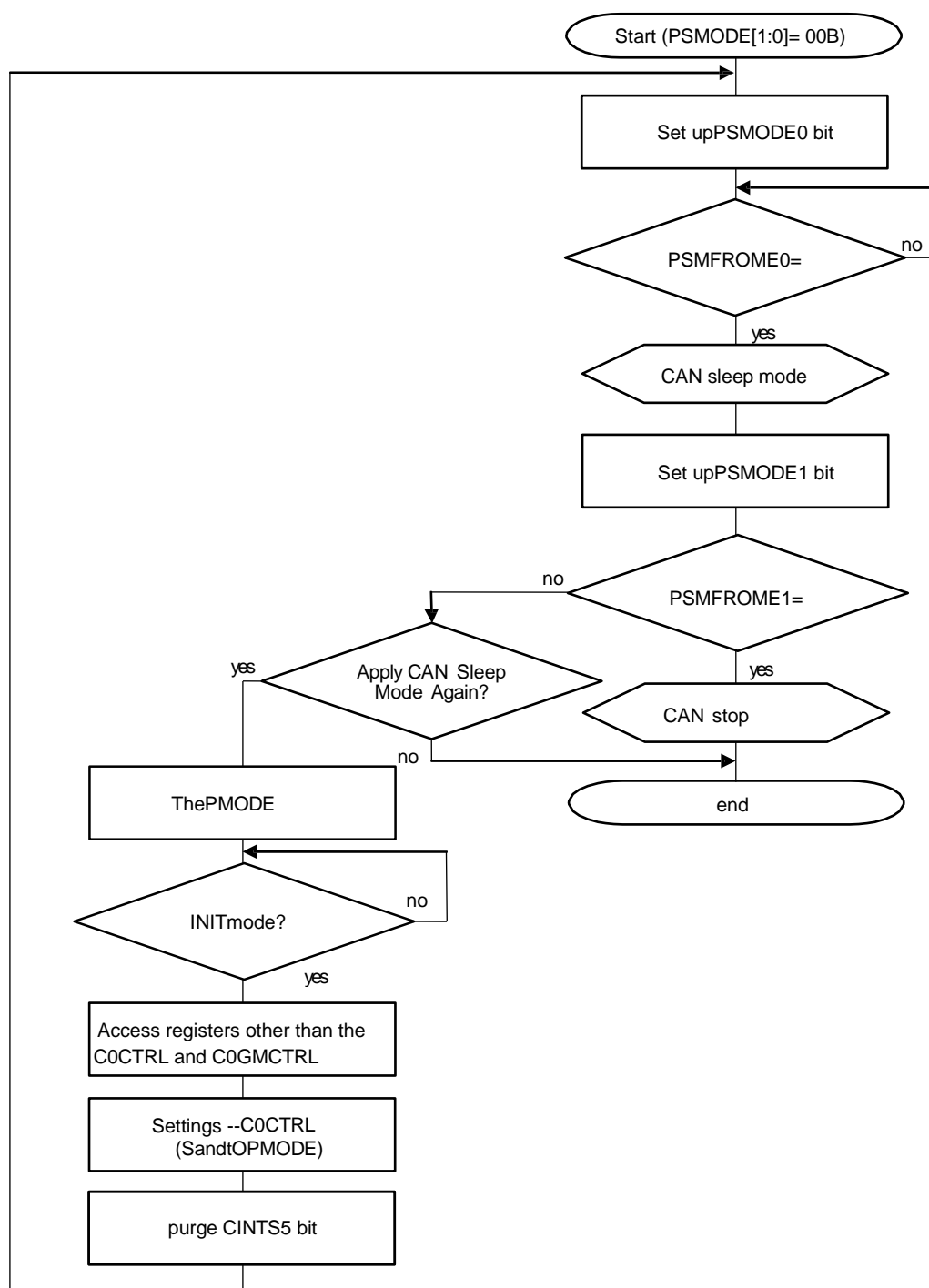
Figure 22-77. Receive through software polling



Note Use read checks for MUC and DN bits

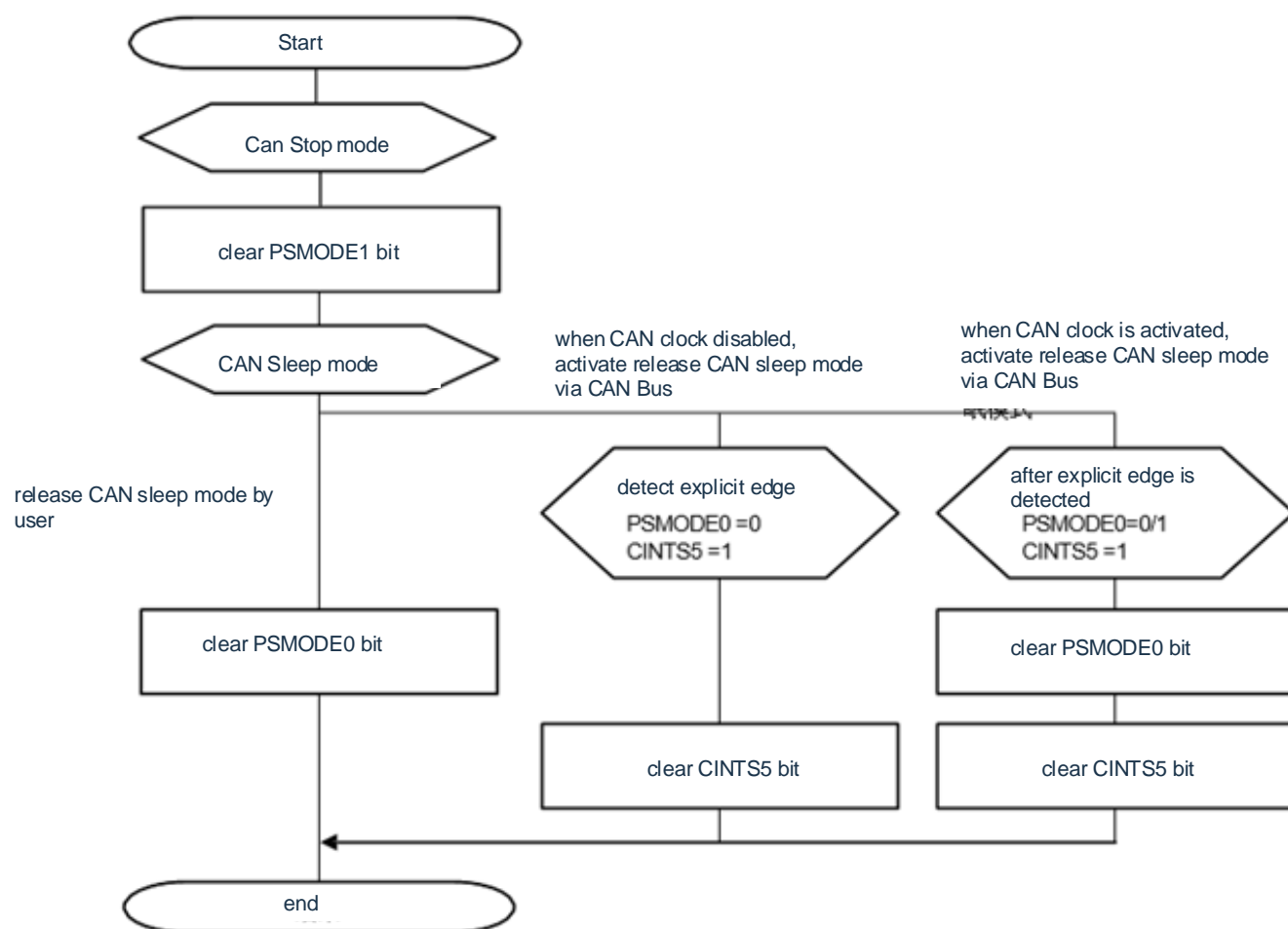
Note 1 Check the MBON flags for the start and end of polling in order to check access to the message buffer and the receive history list register in case a suspended sleep pattern is executed. If MBON is detected to be cleared, then after setting MBON again, the actions and results of the processing must be discarded before processing

2. If ROVF is set, the receive history list is inconsistent. Consider scanning all configured receive buffers for receive.

Figure 22-78. Set the CAN sleep/stop mode


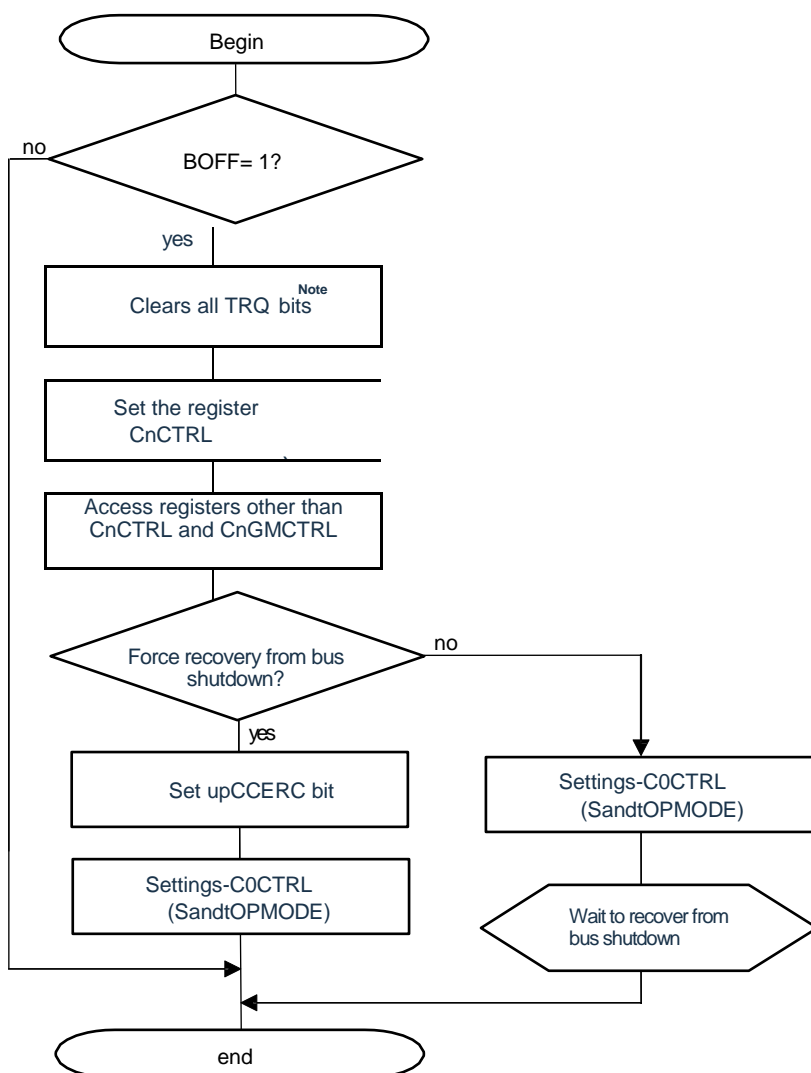
Note: To abort the transfer before requesting CAN sleep mode, perform according to Figure 22-71 or Figure 22-72.

Figure 22-79. Clear CAN sleep/stop mode



Note "CAN Clock Off": With CPU standby mode, the CAN module clock is turned off and the CAN module is in sleep mode.

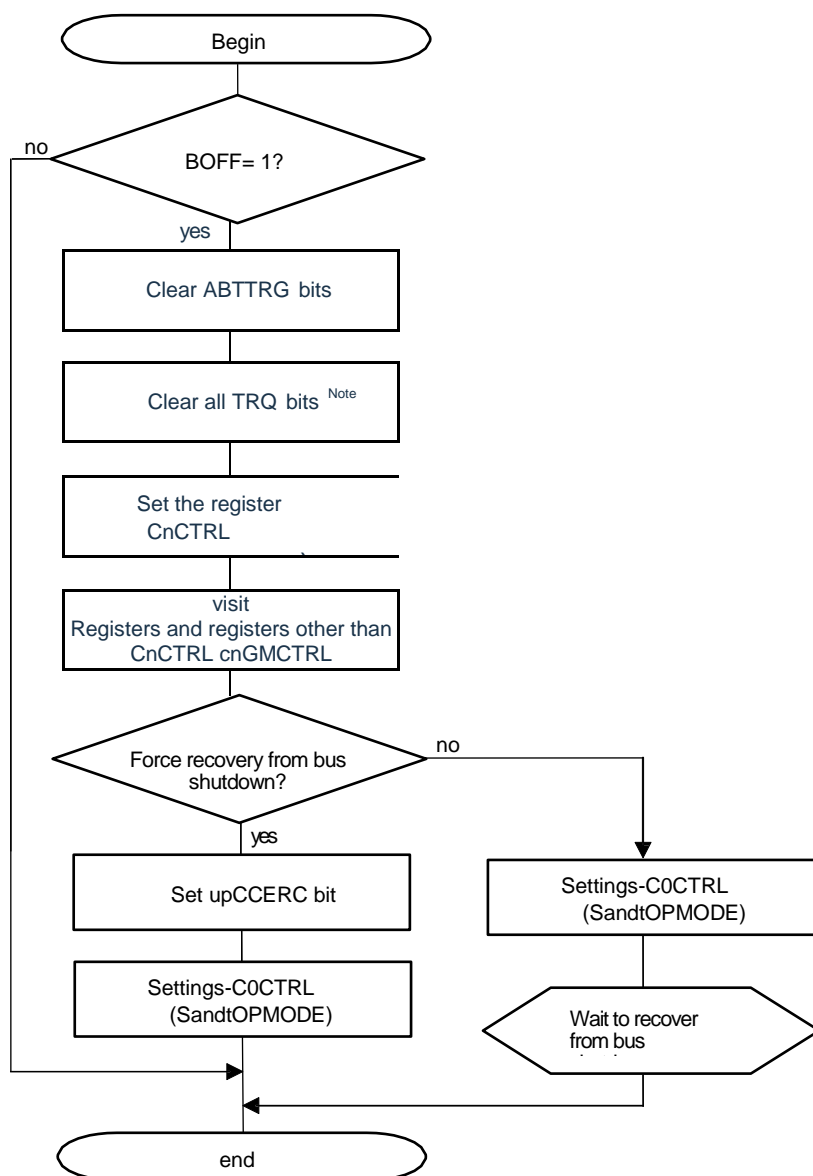
Fig.22-80. Bus shutdown resume (Normal mode operation with ABT)



Note When performing a reinitialization of the packet buffer by clearing the RDY bits before starting the bus shutdown recovery sequence, all TRQ bits are cleared.

Note When a request is made to transition from initialization mode to any other mode of operation to perform the bus shutdown recovery sequence again in the bus shutdown recovery sequence, the Receive Error counter is cleared. Therefore, it is necessary to detect the 11 consecutive recessive bits on the bus again 128 times.

Note OPMODE: Normal operating mode, Normal operating mode with ABT, Receive Only Mode, Single-Shot Mode, Self-Test Mode

Fig 22-81. Bus shutdown resume(Normal mode operation with ABT)


Note: When performing the reinitialization of the packet buffer by clearing the RDY bits before starting the bus shutdown recovery sequence, all TRQ bits are cleared.

Note When a request is made to transition from initialization mode to another mode of operation to perform the bus shutdown recovery sequence again in the bus shutdown recovery sequence, the Receive Error counter is cleared. Therefore, it is necessary to detect the 11 consecutive recessive bits on the bus again 128 times.

Note OPMODE: Normal operation mode, normal operation mode with ABT, receive only mode, single mode, self-test mode

Figure 22-82. Normal Shut down processing

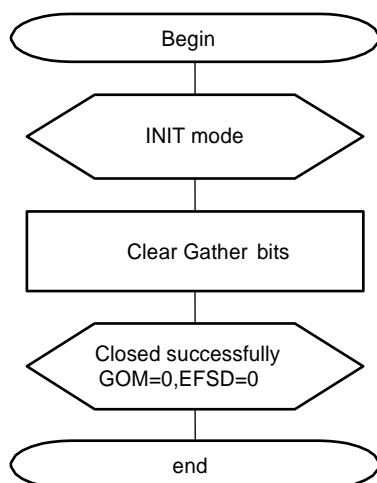
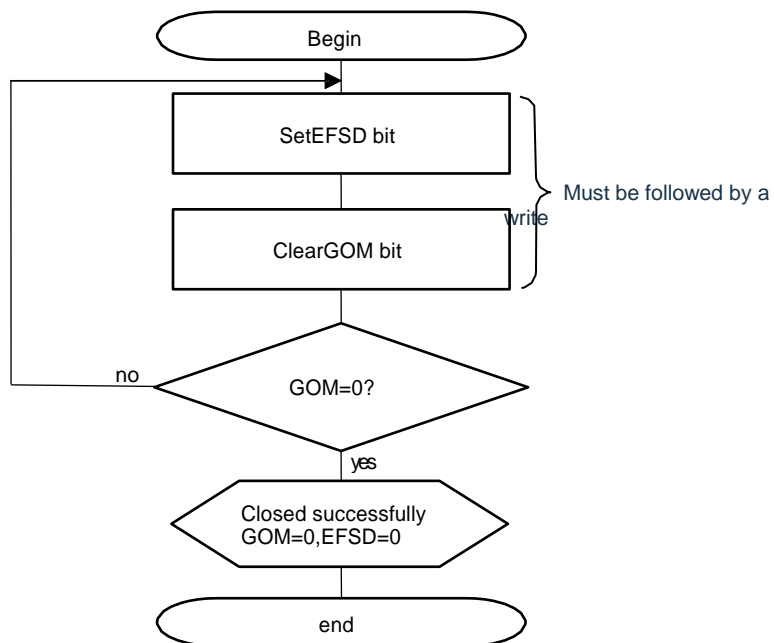


Figure 22-83. Force shutdown processing



Note: Between setting the EFSD bit and clearing the GOM bit, do not read or write any registers through the software.

Note that if an interrupt or DMA occurs, it is not considered sequential access and the forced close request is invalid

Figure 22-84. Error handling

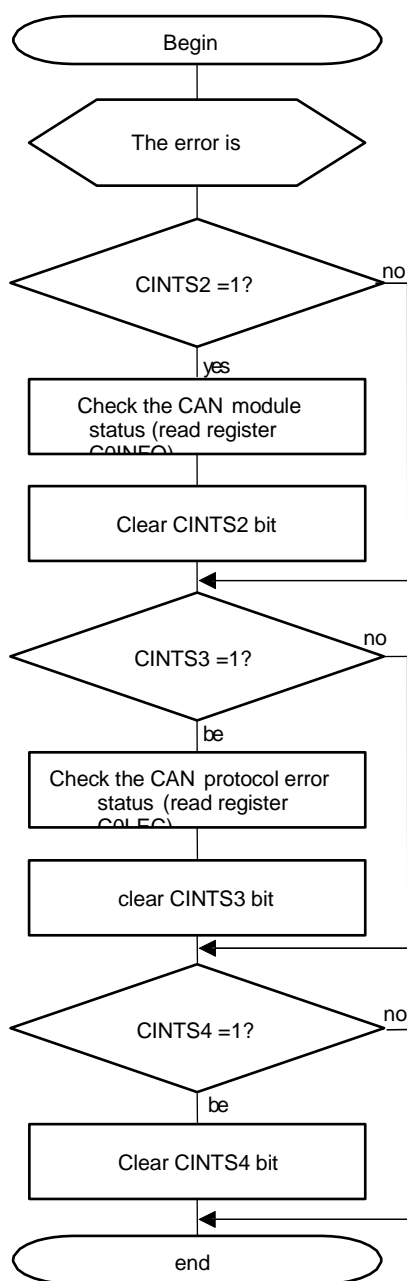
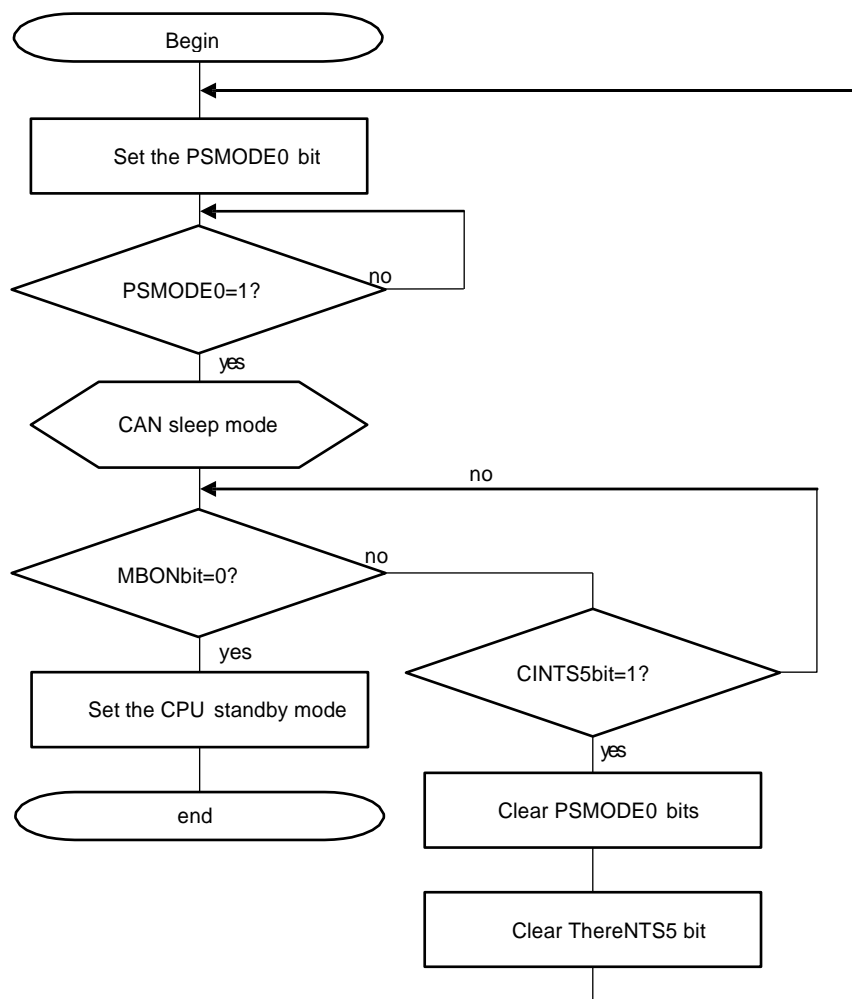
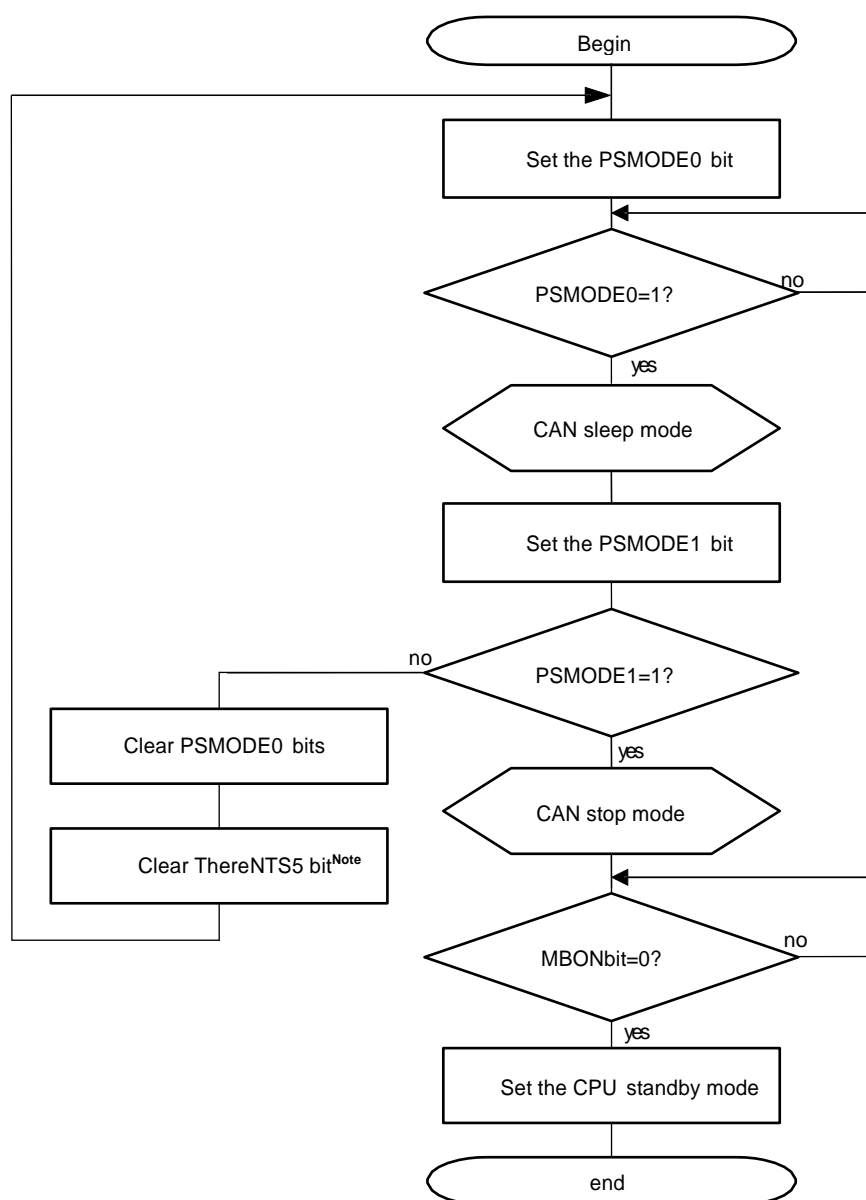


Figure 22-85. Set CPU standby (from CAN sleep mode).



Note: Before the CPU is set to enter CPU standby mode, check if CAN is in sleep mode. When CAN sleep mode is detected, until the CPU is set to standby mode, CAN sleep mode may be canceled by the wake-up of the CAN bus.

Figure 22-86. Set CPU standby (stop mode from CAN).



Note In interrupt wake-up

Note: THE CAN STOP MODE CAN ONLY BE RELEASED BY WRITING 01B TO THE PSMODE[1:0] BIT OF THE CNCTRL register, and not by a change in the CAN bus state.

Chapter 23 LCD bus interface

The LCD bus interface function is a proprietary function of BAT32A279.

The LCD bus interface is used to connect the internal bus system and the external LCD controller/driver.

The interface includes an asynchronous 8bit parallel data bus and two control lines.

The LCD bus interface supports bidirectional communication, which can be sent to or received from the LCD controller.

23.1 Functions of the LCD bus interface

The LCD bus interface has the following functions:

- Two different bus standards are supported:
 - Mode 80: The bus control read and write signals for this mode are separated.
 - Mode 68: The bus-controlled read and write signals in this mode are controlled by a pin with different levels.
- When the internal data bus accesses the LBDATA registers, data transfer begins.
- Supports 8/16 bit read and write operations
- The transmission speed can be controlled (up to 10MHz) through the following settings
 - Select the input clock
 - Set the transfer time
 - Set the waiting status
- The following two events can trigger DMA (DMA supports interrupt triggering).
 - Internal data transfer enablement
 - The external bus access is complete
- Flags are used to indicate the status of data registers, to send data to or receive data from the LCD controller.
- Support DMA read and write

Note: When LCD bus is used in the case of $EVDDx \leq VDD$, the LCD C/D associated registers must be set to the initial value (LCDON=0, SCOC=0, MDSET1-0=00, LCDPFx=0), otherwise normal operation is not guaranteed.

Note: When the LCD bus interface is multiplexed to a pin, the input/output mode of the pin is automatically switched by the LCD bus interface according to the read and write function.

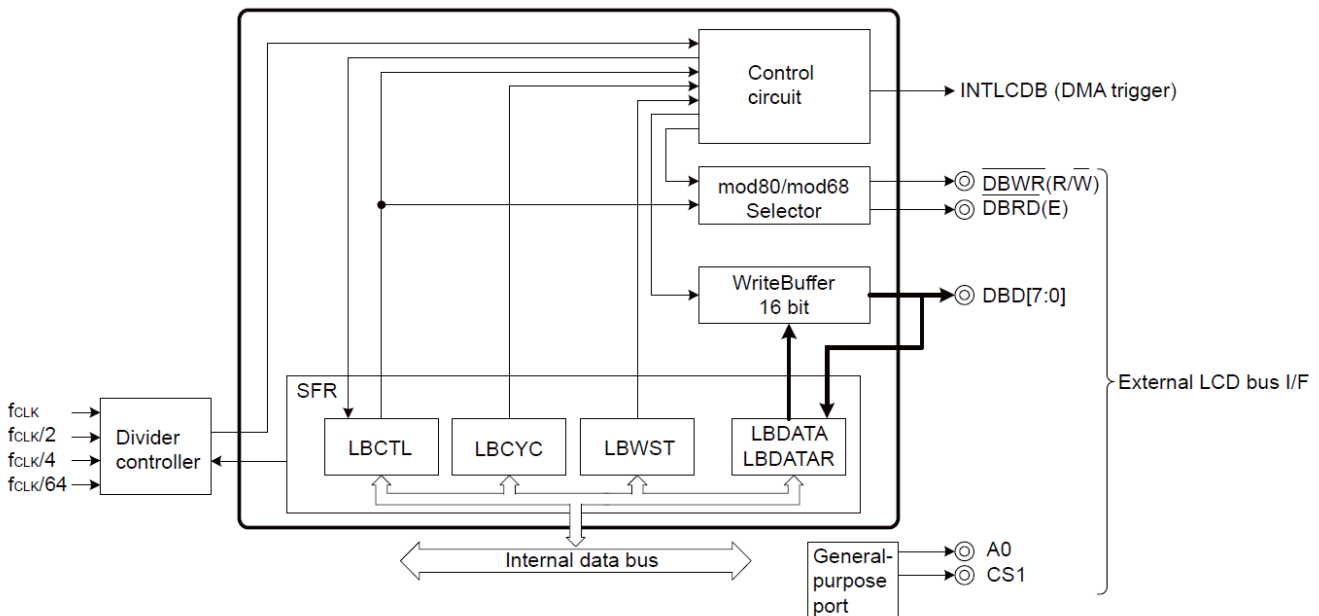
23.2 LCD bus interface configuration

The LCD bus interface includes the following configurations:

Table 23-1 LCD bus interface configuration

project	Configuration
Data I/O pins	8 pins (DBD7 to DBD0).
Control pins	$\overline{\text{DBWR}}$, $\overline{\text{DBRD}}$ (Mode 80(IMD=0)). R/W,E(Mode 68(IMD=1))
Data registers	LCD bus interface data register (LBDATA, LBDATAL). LCD bus interface read data registers (LBDATAR, LBDATARL).
Control registers	LCD Bus Interface Mode Register (LBCTL). LCD bus interface cycle register (LBCYC). THE LCD BUS INTERFACE WAITS STATUS REGISTER (LBWST). Port mode control register (PMCx). Port mode register (PMx). Port register (Px). Register 3 (PER3) is allowed outside

Figure 23-1 Block diagram of the LCD bus interface



23.2.1 LCD bus interface data register (LBDATA, LBDATAL).

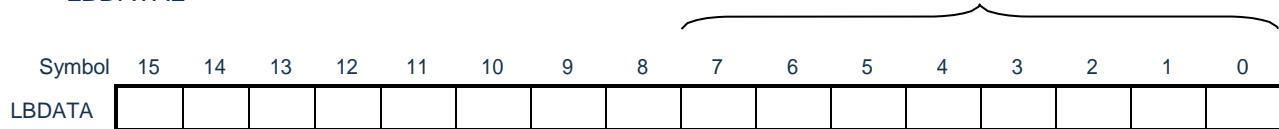
LBDATA is used to store data transmitted through the LCD bus interface and supports both 8-bit and 16-bit read-write.

The value of LBDATA after reset is 0000H

Figure 23-2 Format of the LCD bus interface data register (LBDATA, LBDATAL).

Address: 0x40045410 Reset value: 0000HRW

LBDATAL



The different operation methods of this register determine the transmission mode of the LCD bus interface.

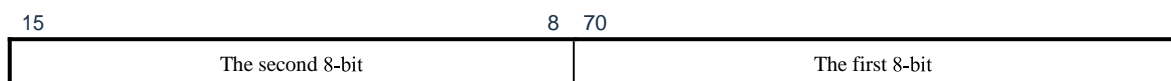
- 8-bit access:

When the register is operated at 8-bit, the LCD bus interface is transmitted at 8 bits.

- 16-bit access:

When a 16-bit operation is performed on the register, the operation is split into two 8-bit operations (8 bits lower and 8 bits higher), which are transmitted sequentially through the bus interface.

When the data is decomposed and transmitted continuously, the transmission sequence is as follows:



Write register LBDATA:

Writing to the LBDATA register immediately sets the LBCTL.BYF flag bit up.

If no LCD bus transfer is in progress at this time (LBCTL.TPF=0), the data is copied to the write cache, and the LBCTL.BYF flag bit is cleared to zero.

If a transfer is in progress at this time (LBCTL.TPF=1), the data is not copied to the write cache until the transfer is complete. As soon as the transfer is complete, the data is copied to the write cache and the LBCTL.BYF flag bit is cleared to zero. The data in LBDATA is copied to the write cache and the LCD bus interface is triggered immediately. At this point, if the LBCTL.TCIS=0, INTLCDB (DMA trigger source) is placed.

Read register LBDATA

The read operation of the LBDATA register is carried out by starting the LCD bus interface for read transmission. The data read out from this register is the data received during the last transmission of the LCDB bus interface.

Note 1 When accessing an LBDATA register, it can only point to the even address of the register, and the odd address of the register is prohibited.

2.LBCTL. The register can only be accessed if the BYF is 0.

23.2.2 LCD bus interface read data registers (LBDATAR, LBDATARL).

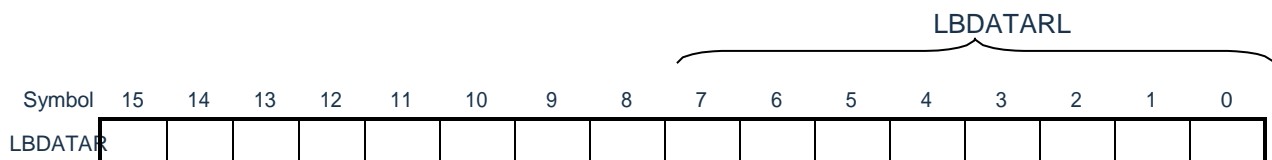
LBDATAR is a read-only register. It contains the data that was last read and transmitted through the LCD bus interface. Reading this register does not initiate a new read transfer on the LCD bus interface.

This register supports either a 16-bit read operation or an 8-bit read operation.

When a reset occurs, lbdatar has a reset value of 0000H

Figure 23-3. Format of the LCD bus interface read data registers (LBDATAR, LBDATARL).

Address: 0x40045412H reset value: 0000H R/W



By reading this register, the data transferred to the LBDATA register during the previous read operation can be obtained without having to start the LCD bus transfer again.

Reading the LBDATAR register does not change lbcTL. Status of the BYF flag and the LBCTL.TPF flag.

Note: When accessing the LBDATAR register, it can only point to the even address of the register, and the odd address of the register is prohibited.

23.3 Control registers for the LCD bus interface

The following 10 registers are used to control the LCD bus interface:

- LCD Bus Interface Mode Register (LBCTL).
- LCD bus interface cycle register (LBCYC).
- LCD BUS INTERFACE WAITS STATUS REGISTER (LBWST).
- Port mode control register (PMCx).
- Port mode register (PMx).
- Port register (Px).
- Peripheral enable register 1 (PER1).

23.3.1 Peripheral enable register 1 (PER1).

The PER1 register is a register that is set to enable or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use.

To use the LCD bus interface, bit1 (LCDBEN) must be set to "1".

The PER1 register is set via the 8-bit memory operation instruction.

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

Figure23-4 Perimeter Enable Register (PER3).

Address: 0x400208 after 1C reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
ONthe R3	Xx	Xx	Xx	Xx	Xx	Xx	Xx	LCDIn

LCDBEN	Control of the LCD bus interface clock
0	Stop providing the input clock. <ul style="list-style-type: none"> • Cannot write SFR for LCD bus interfaces used. • The LCD bus interface is reset.
1	An input clock is provided. <ul style="list-style-type: none"> • SFR that can read and write LCD bus interfaces.

23.3.2 LCD Bus Interface Mode Register (LBCTL).

LBCTL is used to control the operation of the LCD bus interface.

Set lbcTL via 8-bit memory operation instructions The reset value is 00H

Figure 23-5 Format of the LCD bus interface mode register (LBCTL).

Address: 0x40047400 Reset value: 00H R/W

Symbol	7	6	5	4	3	2	<1>	<0>
LBCTL	The	IMD	LBC1	LBC0	TCIS	0	TPF	BANDF
	The	Controls the level of signal "E" in 68 mode						
	0	E: Highly effective, data is read and written on the falling edge.						
	1	E: Low effective, data is read and written on the rising edge.						
	IMD	Selection of access modes for external bus interfaces						
	0	Mode 80 – The control signals are \overline{WR} and \overline{RD}						
	1	Mode 68 – The control signals are E and R/\overline{W}						
	LBC1	LBC0	Internal Clock Selection (SPCLK)					
	0	0	fCLK					
	0	1	fCLK/2					
	1	0	fCLK/4					
	1	1	fCLK/64					
	TCIS	INTLCDB (DMA Trigger) control						
	0	During write access to the bus interface, INTLCDB is generated as soon as data is moved from the LBDATA register to the write buffer. During read access to the bus interface, INTLCDB is generated as soon as the data in lbDATA and LBDATAR registers is available.						
	1	An interrupt occurs when the DATA transmission of the LCD bus interface is complete						
	TPF	External bus interface transmission flags						
	0	The external bus interface is idle						
	1	Data is being transferred on the external bus interface						
	BYF	Data register busy flag bit						
	0	Data can be read or written from LBDATA Data can be read from LBDATAR						
	1	Register LBDATA (LBDATAR) is busy						

Note 1.Bit2 must be set to 0.

- Although the LBCTL.TPF flag is used to identify the current state of LCD bus data transmission, sometimes reading this flag may also get an incorrect status.

Therefore, it is not recommended to use the method of polling the LBCTL.TPF flag, and it is recommended to use DMA transmission to load new LCD data into the LCD bus interface data register (LBDATAx).

23.3.3 LCB bus interface periodic control register (LBCYC).

LBCYC registers are used to control the cycle time of the LCD bus interface. The cycle time is the duration of one 8-bit data transmission per bus access. LBCYC uses 8-bit memory manipulation instructions for setup.

When resetting, the LBCYC register resets to 00H

Figure 23-6 LCB bus interface periodic control register (LBCYC).

Address: 0x40047401 Reset value: 02H R/W

Symbol	7	6	5	4	3	2	1	0
LBCYC	0	0	CANDC5	CANDC4	CANDC3	CANDC2	CANDC1	CANDC0

CYC5-CYC0	Cycle time
000000B	Disable settings
000001B	
000010B	Cycle time: 2 T
000011B	Cycle time: 3 T
:	:
111110B	Cycle time: 62 T
111111B	Cycle time: 63 T

Note: 1. T is the clock period of the selected clock (set by LBC1 and LBC0).

2. LBCYC≥2.

23.3.4 The LCB bus interface waits for control registers (LBWST).

LBWST is used to control the number of cycles in which the LCD bus interface waits for state. The number of periods of wait state determines the duration of \overline{DBWR} and \overline{DBRD} signals. This duration must be shorter than the cycle time.

LBWST is set using 8-bit memory manipulation instructions.

When resetting, lbWST resets to 00H.

Figure 23-7 The LCD bus interface waits for the format of the control register (LBWST).

Address: 0x40047402 Reset value: 00H R/W

Symbol	7	6	5	4	3	2	1	0
LBWST	0	0	0	WST4	WST3	WST2	WST1	WST0

WST4-WST0	Wait period
00000B	The wait period is not inserted
00001B	1 T
00010B	2 T
00011B	3 T
:	:
11110B	30 T
11111B	31 T

Note: 1.T is the clock period of the selected clock (set by LBC1 and LBC0).

2.WST≤CYC-2.

23.3.5 Pin-mode control registers

When using the LCD bus interface pins, the control registers for the multiplexed port function (port mode register (PMxx), port register (Pxx), and port mode control register (PMCxx)) must be set. For details, please refer to "2.3.1 Port Mode Register (PMxx)", "2.3.2 Port Register (Pxx).") and "2.3.6 Port Mode Control Registers (PMCxx)". For details, please refer to "2.5 Register Settings When Using the Multiplexing Function".

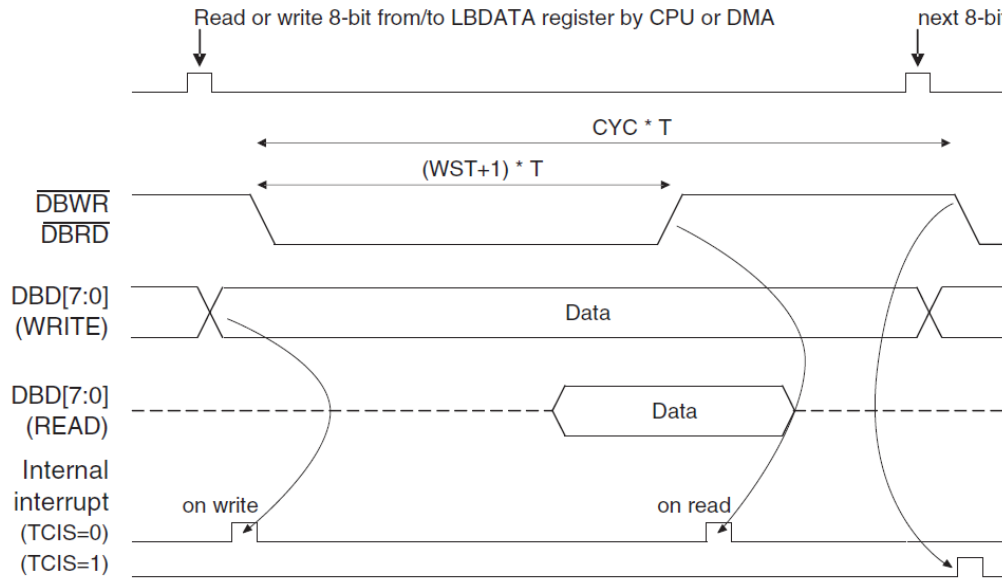
23.4 Runtime order

This section describes the timing of general operations, and then describes examples of sequential write and read operations.

23.4.1 Timing relationships

The following figure shows the general timing when using the 80 pattern. It illustrates the effect of LBCYC and LBWST register settings. The impact of LBCTL.TCIS on INTLCDB is also explained.

Figure23-8 LCD bus interface timing (mode 80).



In mode 80, \overline{DBWR} is the optional communication number for the write action, \overline{DBRD} is the optional communication number for the read action

- Note:**
- 1 T is the clock cycle of the internal clock SPCLK (configured by LBC1 and LBC0 bits).
 - 2 CYC is the number of clock cycles selected by LBCYC. LBCYC >2 is required.
 - 3 WST is the number of wait state clock cycles selected by LBWST and requires LBWST < (LBCYC-2).

The 68 mode and 80 mode signals have different meanings: DBWR is replaced with the read-write select communication number $\overline{R/W}$, and DBRD is replaced with the strobe enable signal E, which selects the communication number. The effective level of E is determined by LBCTL.EL.

23.4.2 LCD bus interface status

When the chip pins are configured for use with the LCD data bus interface DB [7:0], the input and output modes of the pins are automatically switched by the LCD module. After the pin is configured as DB[7:0], the pin is in input mode. Db[7:0] works in input mode while the bus interface is working during read access, and continues to remain in input mode after read access is complete. Db[7:0] works in output mode during the bus interface operation during write access, which also continues to remain in output mode after write access is complete.

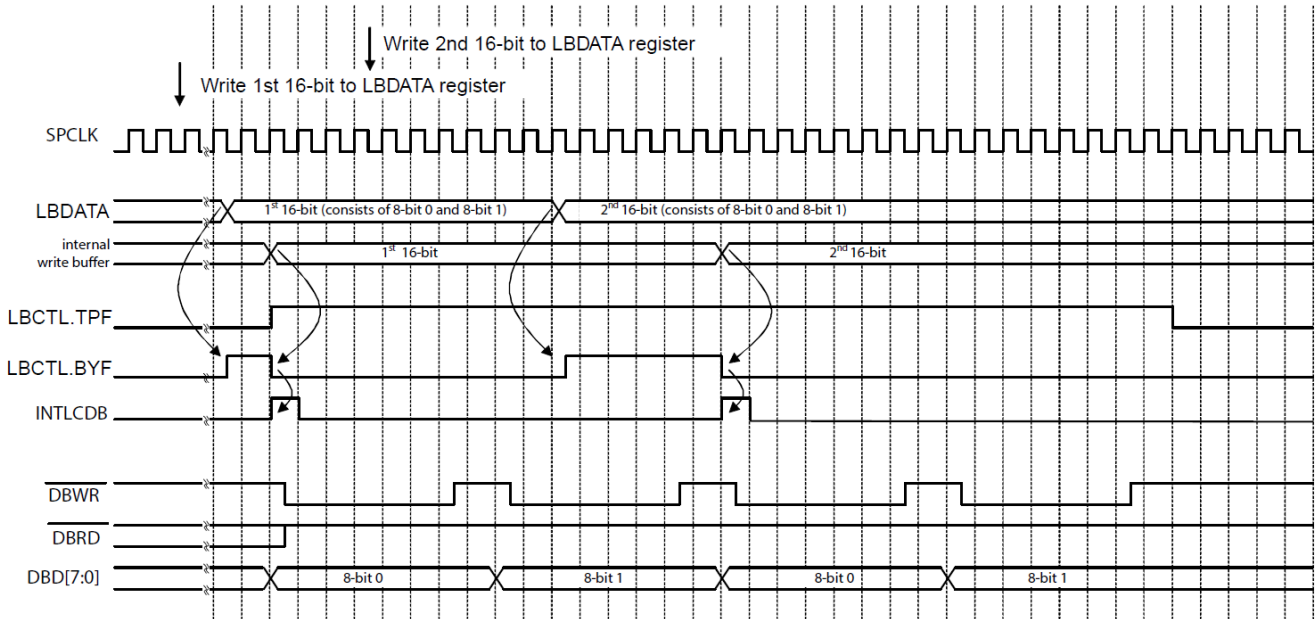
23.4.3 Write the LCD bus

This section describes typical 16-bit and 8-bit write timing for LCD buses.

(1). 16-bit write

A 16-bit write transfers two 8-bit data to an external LCD controller/driver.

Figure 23-9 Write contiguous 16 bits timing (80 mode: LBCTL. IMD=0)
LBWST=5, LBCYC=8, LBCTL. TCIS=0



Description: Timing diagrams are for feature description purposes only and have no association with the actual hardware implementation.

(a) The order of the run

<1> the first set of 16 bits of data from the LCD is written to the LBDATA registers. Operating the write interface registers over the internal bus requires some clocks. The busy flag bits then are set up until the data is copied to the write buffer.

<2> the contents of the LBDATA register are copied to the write buffer. LBCTL. The BYF flag bit is then cleared and an INTLDB valid signal for one clock cycle is output. Transmission on the LCD bus interface starts at 8 bits of data 0. The flag bit LBCTL.TPF is placed, indicating that a transmission is in progress.

<3> DMA triggered by INTLDB writes a second set of 16 bits of data to LBDATA. By querying the busy flag bit LBCTL.BYF, the CPU can also write this 16 bits of data. Operating the internal bus to write LBDATA registers and enable LBCTL.BYF to be set up also requires some clock cycles.

<4> Since the LCD bus interface is still in transit, the contents of the LBDATA register cannot be copied immediately to the write buffer, LBCTL. BYF remains in the state of being placed.

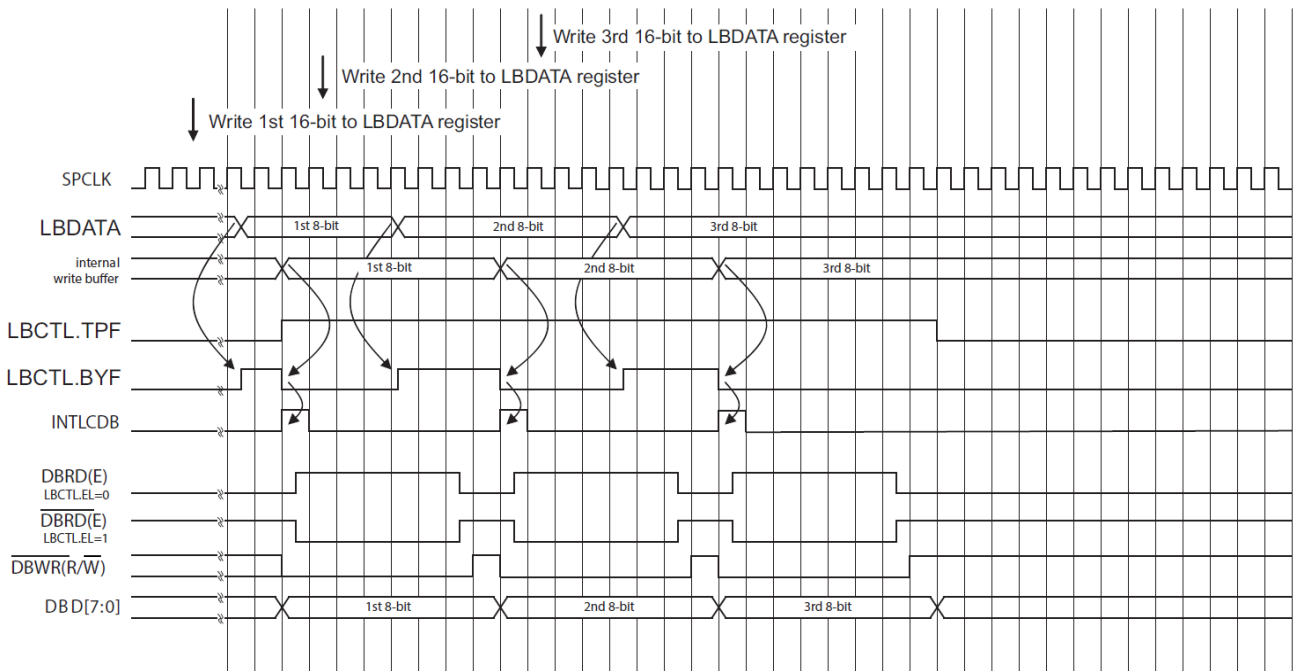
<5> After the LCD bus interface transfer ends, the write buffer is again populated with data from the LBDATA registers. Busy flag bit LBCTL. BYF is cleared to zero while outputting an INTLDB valid signal for one clock cycle.

Each time data is copied from LBDATA to the write buffer, a transfer to the external LCD controller is initiated.

(2). 8-bit write

Write 8 bits of data continuously and transfer these 8 bits to an external LCD controller/driver.

Figure23-10. 68 mode: LBCTL. IMD=1) Write consecutive 8bits timings
LBWST=5,LBCYC=8,LBCTL. TCIS=0



Description: Timing diagrams are for feature description purposes only and have no association with the actual hardware implementation.

(a) The order of the run

<1> the first set of 8 bits data of the LCD is written to the LBDATA register. Operating the write interface registers over the internal bus requires some clocks. The busy flag bits then are set up until the data is copied to the write buffer.

<2> the contents of the LBDATA register are copied to the write buffer. LBCTL. The BYF flag bit is then cleared and an INTLCDB valid signal for one clock cycle is output. Transmission on the LCD bus interface starts at 8 bits of data 0. The flag bit LBCTL.TPF is placed, indicating that a transmission is in progress.

<3> DMA triggered by INTLCDB writes a second set of 8 bits of data to LBDATA. By querying the busy flag bit LBCTL.BYF, the CPU can also write this 8 bits of data. Operating the internal bus to write LBDATA registers and enable LBCTL.BYF to be set up also requires some clock cycles.

<4> Since the LCD bus interface is still in the transmission state, the contents of the LBDATA register cannot be copied immediately to the write buffer, LBCTL. BYF remains in the state of being placed.

<5> after the LCD bus interface transfer is complete, the write buffer is again populated with data from the LBDATA registers. Busy flag bit LBCTL. BYF is cleared to zero while outputting an INTLCDB valid signal for one clock cycle.

Each time data is copied from LBDATA to the write buffer, a transfer to the external LCD controller is initiated.

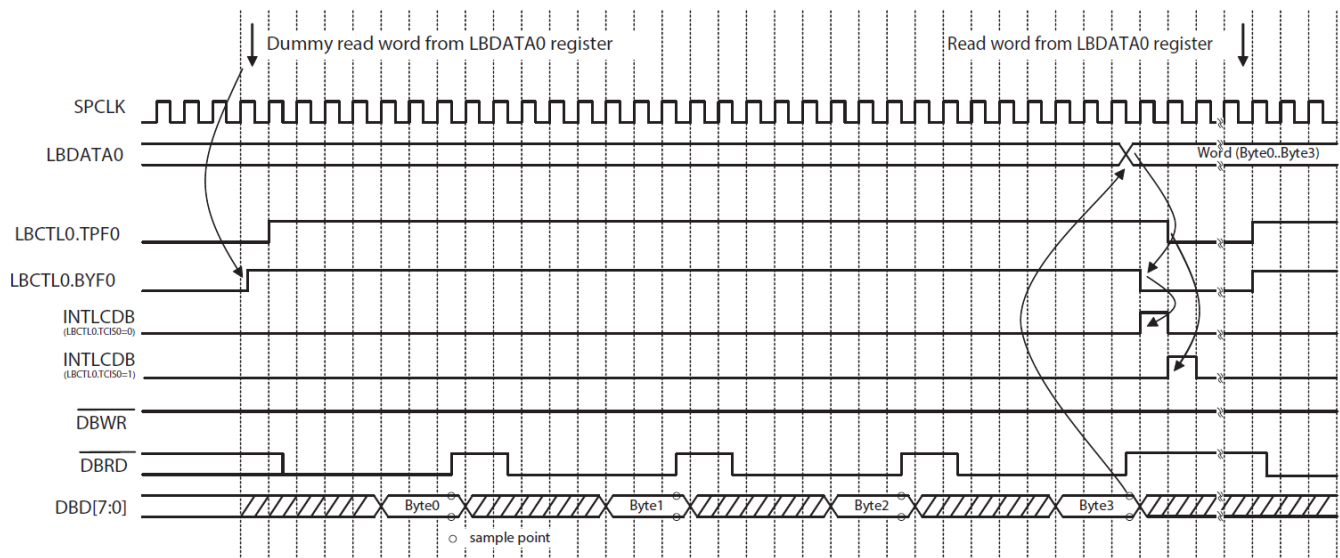
23.4.4 Read from the LCD bus

Supports reading data in 8-bit or 16-bit format from the LCD bus. The following shows a typical timing of reading 8 bits.

(1). 16-bit read

The figure below shows a 16 bits read action in 80 mode.

Figure 23-11(80 mode: LBTCTL. IMD=0): Read 16 bits timing
LBWST=5,LBCYC=8,LBCTL. TCIS=0 and 1



Description: Timing diagrams are for feature description purposes only and have no association with the actual hardware implementation.

(a) The order of the run

<1> a virtual read of the LBDATA register initiates a 4-byte read from an external LCD controller. Busy flag bit LBCTL. BYF was immediately put up. Transmitting flag bits LBCTL. TPF is placed on the rising edge of the clock. The data read from LBDATA belongs to the previous transfer data and can be ignored.

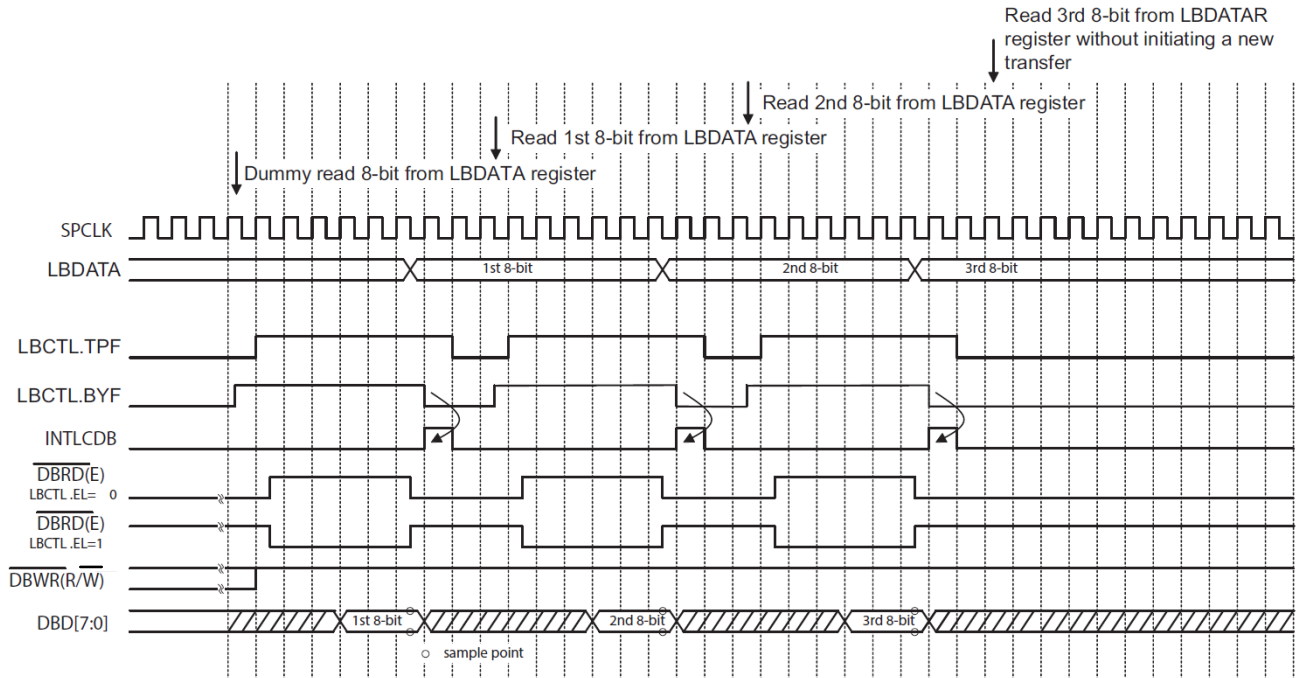
<2> the busy flag bit LBCTL.BYF is cleared when the last of the four bytes is sampled and the full word is available in the LBDATA register. The flag bit LBCTL.TPF remains in a pending state until the end of the cycle time of the last byte.

<3> then read the LBDATA registers to get the data transferred by the LCD controller and initiate the next new transfer.

(2). 8-bit read

The figure below shows the 8 bits read action in 68 mode.

Figure 23-12 (68 mode: LBTCTL. IMD=1): Read consecutive 8 bits timing
LBWST=4, LBCYC=7, LBCTL. TCIS=0



Description: Timing diagrams are for feature description purposes only and have no association with the actual hardware implementation.

(a) The order of the run

<1> a virtual read of the LBDATA register initiates the reading of 8 bits of data from an external LCD controller. Busy flag bit LBCTL. BYF was immediately put up. Transmitting flag bits LBCTL. The TPF is placed on the rising edge of the clock. The data read from LBDATA belongs to the previous transfer data and can be ignored.

<2> the busy flag bit LBCTL.BYF is cleared when data from the LCD bus interface is sampled into the LBDATA registers available. The interrupt signal INTLCD outputs the effective level of one clock cycle.

<3> performs a new read of LBDATA before the last transfer completed (no cycle time elapses). The busy flag bit LBCTL.BYF is immediately set up, but a new transfer begins after the previous transfer is complete. The in-transit flag LBCTL.TPF remains unchanged. The data read from LBDATA is the first 8bits LCD data.

<4>, the sampled data is available in LBDATA, and the busy flag bit LBCTL.BYF is cleared.
repeat steps 2 through 4 <5 > until the last 8 bits to be read are sampled.

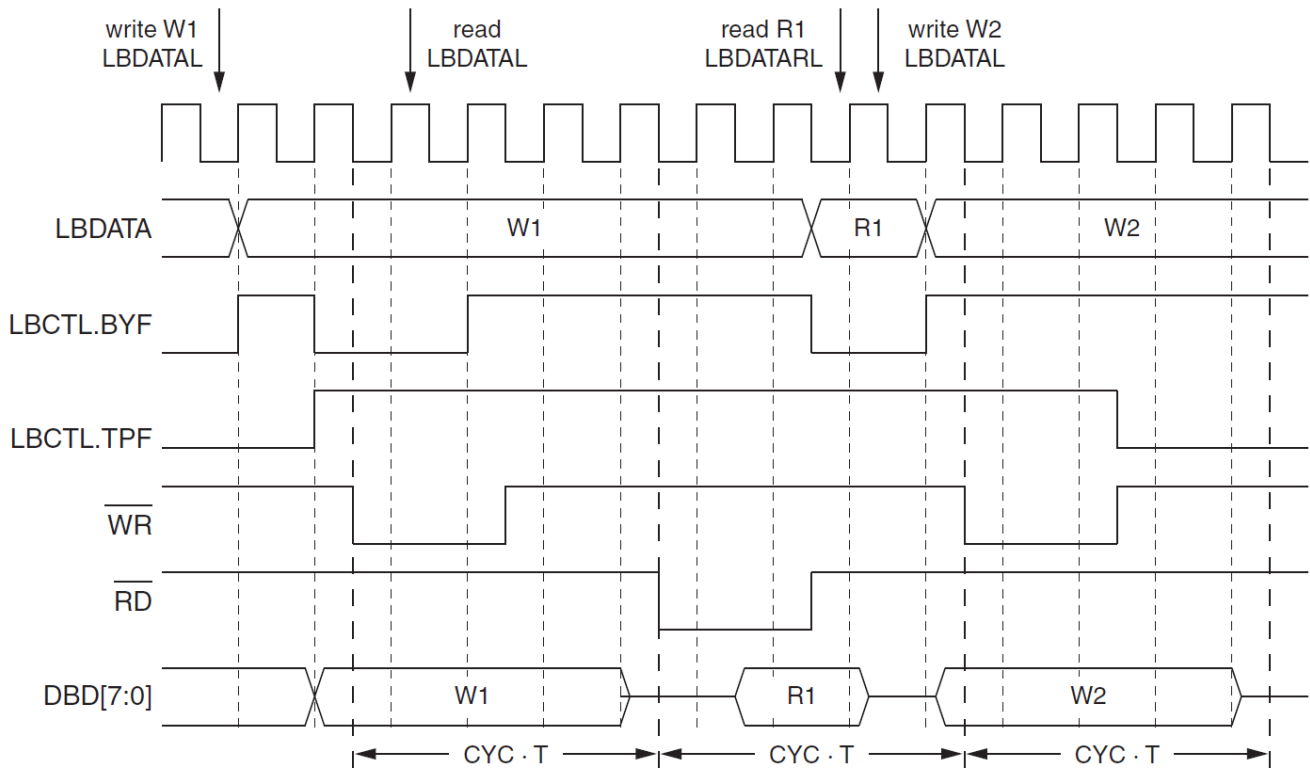
<6> the last 8 bits are not read from the LBDATA registers, but from LBDATAR to avoid triggering a new read transfer on the LCD bus.

23.4.5 Write-read-write timing on the LCD bus

Figure 23-13 shows an example of a write access to the LCD bus followed by a read access and vice versa. In 80 mode (LBCTL.IMD=0) 8 bits transfer as an example.

In 68 mode (LBCTL.IMD=1), the timing is the same when the RD strobe is considered to be a low effective E signal (LBCTL.EL=1).

Figure 23-13.(80 Mode: LBCTL.IMD=0): 8-bit write-read-write timing
LBWST=4,LBCYC=7,LBCTL.TCIS=0



23.5 Note for LCD bus interfaces

23.5.1 Write to LbDATA/LBDATAL registers

When the LCD data bus is in transit, a write operation to the LBDATAx register may cause a collision of data transfers. To avoid this, you must do the following:

Avoid writing LBDATAx registers simultaneously on the LCD bus transmission. When a transfer is in progress, in order to ensure that the LBDATAx registers are not written, the LBCTL.TCIS sets 1 to decide whether to write to the LBDATAx registers according to whether a bus interface interrupt is generated.

It is recommended to use DMA transfer to load new LCD data into the LCD Bus Interface Data Register (LBDATAx).

23.6 Example of LCD bus interface transmission

23.6.1 Example of transmission with an external LCD driver

Example 1

The BAT32A279 can be used as a master chip to provide a clock for display to the slave chip (LCD driver) via the CLKBUZ0 pin.

System Configuration:

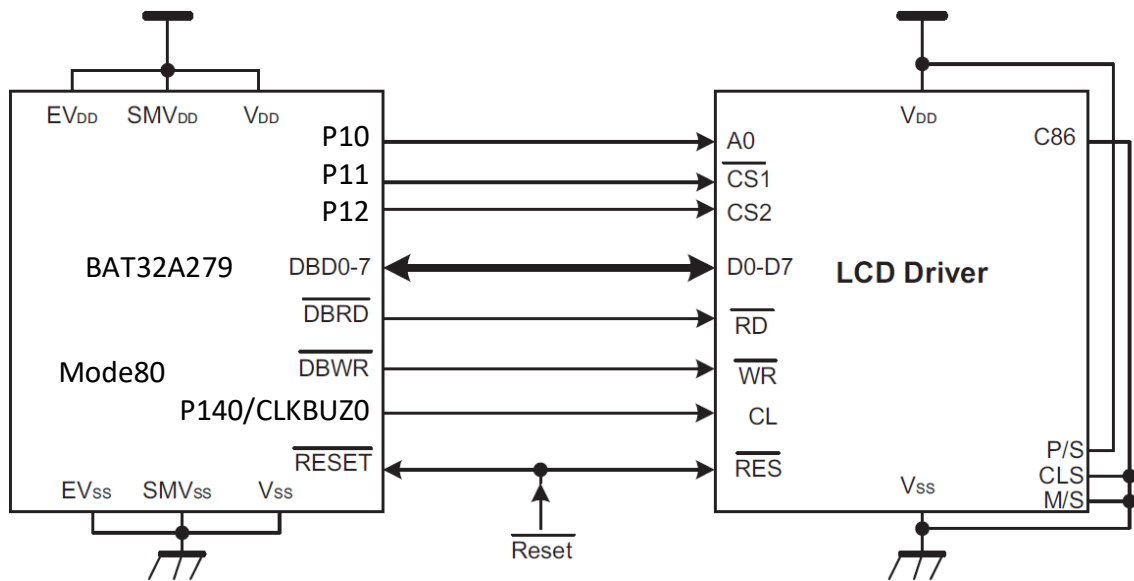
- The system clock is 32MHz and the LCDB access period is 8MHz ($f_{CLK}/4$).

68/80 mode

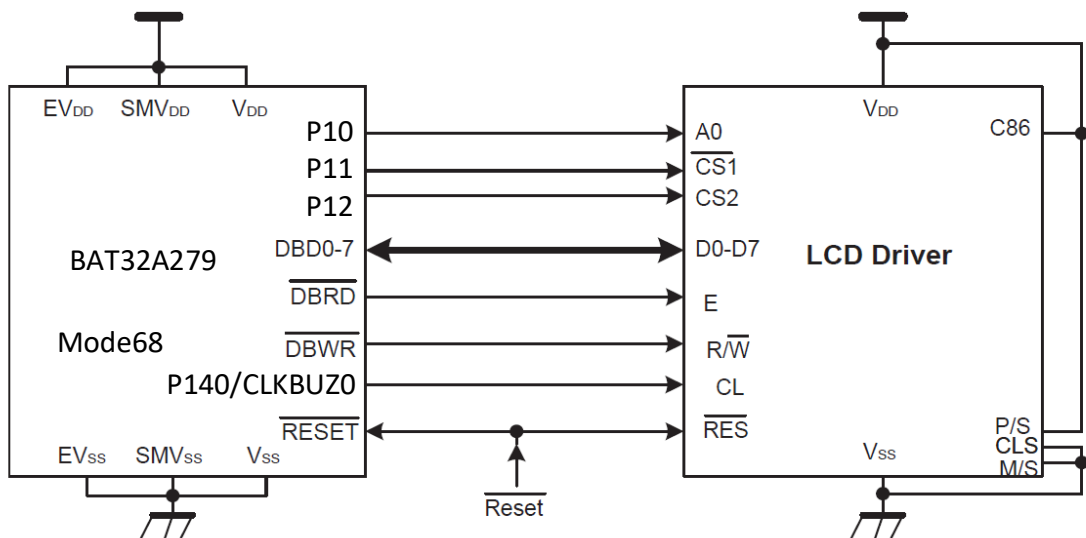
- CL is provided by CLKBUZ0 ($f_{CLK}/2^{11}=15.6\text{kHz}$).

Figure 23-14 Connection example 1

<Mode80>



<Mode68>



23-22 Connection example 1

No.	LCD driver pins	LCD drive function	Port name
1	A0	Used to indicate whether D0 to D7 is data or	P10 ^{note}
2	$\overline{\text{CS1}}$	Select 1	P11 ^{note}
3	CS2	Select 2	P12 ^{note}
4	D0 to D7	8-bit bidirectional data bus	DBD0 to DBD7
5	$\overline{\text{RD(E)}}$	Mode 80: Read the communication number	_DBRD
6	$\overline{\text{In R(R/ W)}}$	80 mode: Write and select the communication number	_DBWR
7	CL	Displays the clock	P140/CLKBUZ0
8	Nothing	reposition	_RESET

Note: Pins P10, P11, and P12 are used as A0, CS1 and CS2, which is just an example, can also use other pins.

Example 2

The BAT32A279 can be used as a master chip to provide a clock for display to the slave chip (LCD driver) via the CLKBUZ0 pin.

System Configuration:

- $f_{\text{CLK}}=6\text{MHz}$
- PCF21119x68 mode
- The f_{osc} is provided by the CLKBUZ0 pin ($f_{\text{CLK}}/16=375\text{kHz}$).

Figure 23-15 Connection Example 2

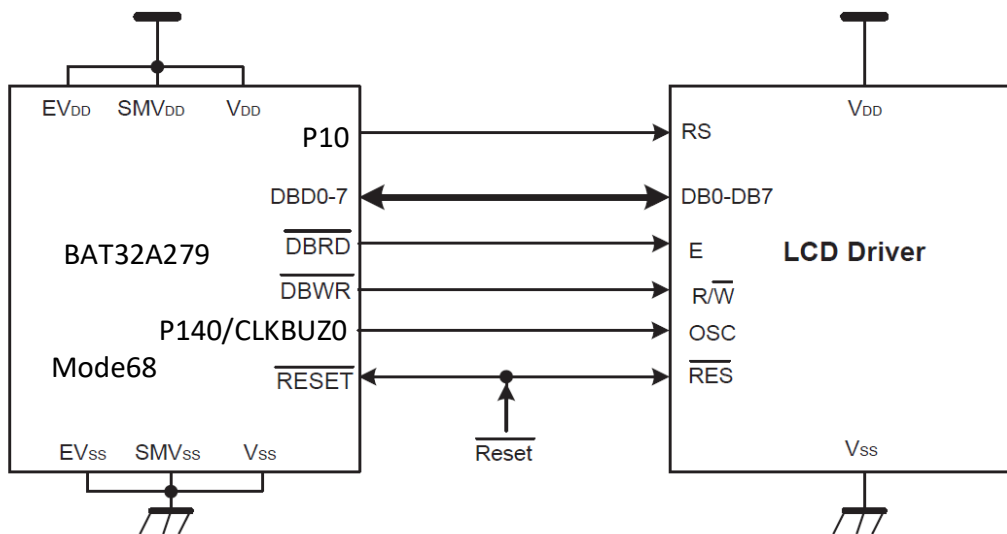


Table 23-3 Connection example 2

No.	LCD driver pins	LCD drive function	Port name
1	RS	Register selection	P10 ^{Note}
2	DB0 to DB7	8-bit bidirectional data bus	DBD0 to DBD7
3	And	Select the communication number	_DBRD
4	$\overline{\text{R/W}}$	Read/write control	_DBWR
5	OSC	Crystal or external clock input	P140/CLKBUZ0
6	Nothing	reposition	_RESET

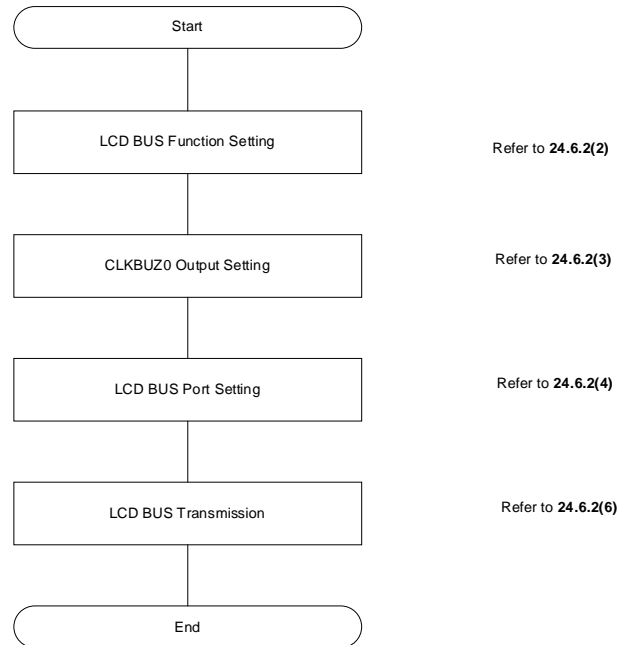
Note: Using pins P10 as RS is just an example, other pins can also be used.

23.6.2 The flow of LCD bus transmission

(1). Flowchart (recommended).

This flowchart is the operating procedure of the LCD bus transmission. A detailed explanation of each step is explained in the following subsections. (The reference section number on the right)

Figure23-16 LCD bus transmission

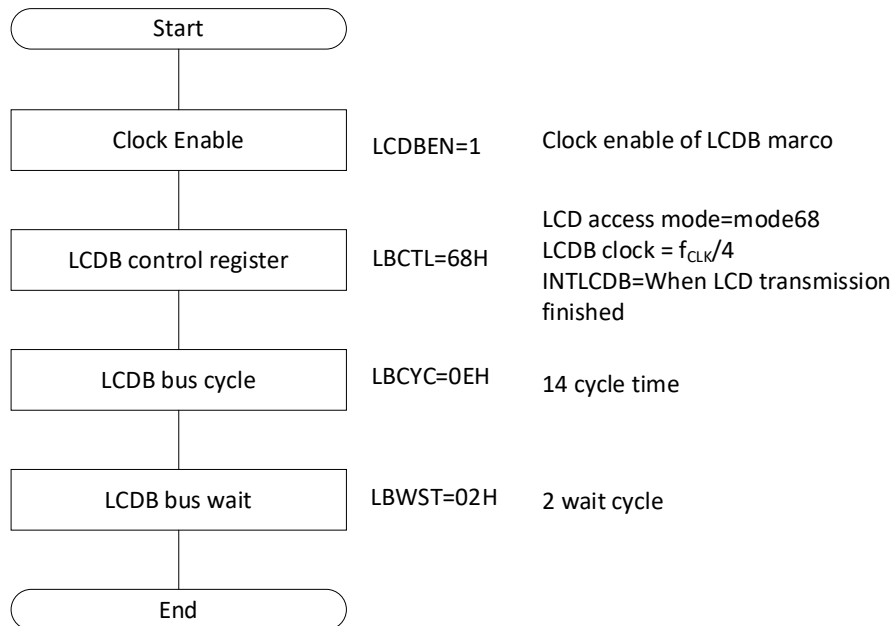


(2). LCD bus function settings

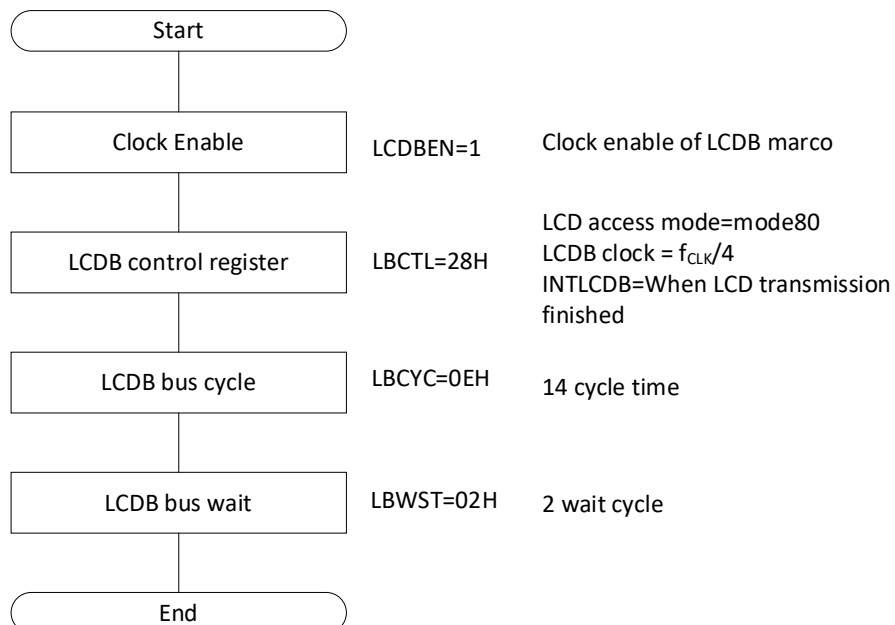
- Enables the clock of the LCDB module
- Set the access mode to 68 mode or mode 80
- Set the internal clock (for example: $f_{CLK}/4$).
- Set to generate INTLCDB when the LCD transfer ends
- Set the LCDB data transfer cycle (e.g., 14").
- Set the LCDB data transfer wait period (e.g., 2").

Figure23-17 LCD Bus Function Setup Flow

< 68 mode >



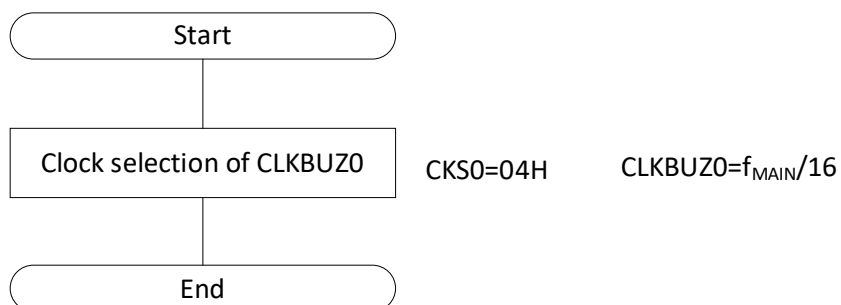
< 80 mode >



(3). PCL clock settings

- Set the CLKBUZ0 clock (e.g., $f_{\text{MAIN}}/16$).
- Set P140 as the output pin of CLKBUZ0

Figure 23-18 CLKBUZ0 clock setup flow



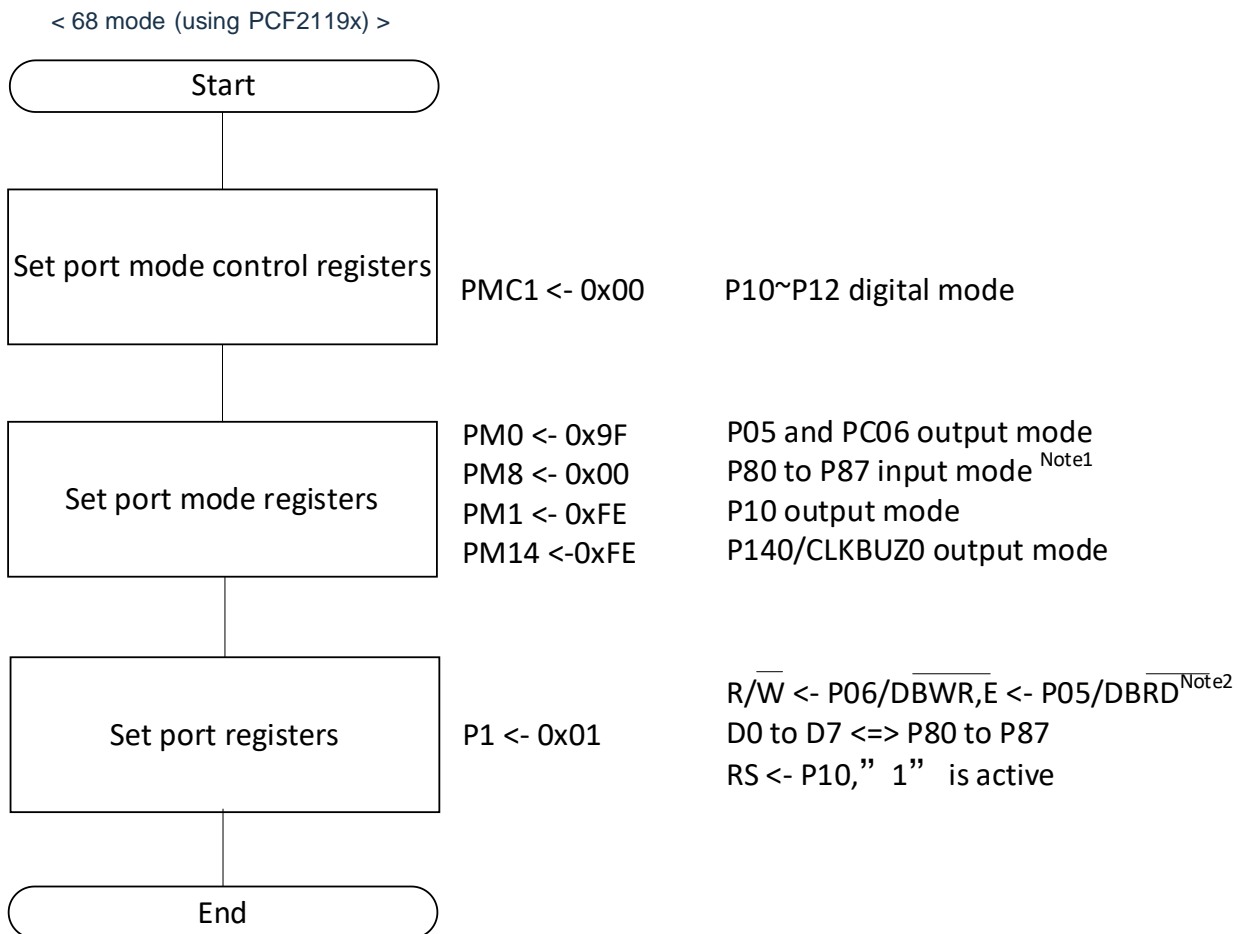
When the system clock is high, the appropriate CLKBUZ0 divideover needs to be set to meet PCF2119x ($f_{\text{OSC}}=120\sim 450\text{kHz}$) or S1D15E00 ($f_{\text{OSC}}=$. Specification for 40kHz (TYP).). See the LCD driver data sheet for details.

(4). LCD bus pin settings

The LCDB is used as a bidirectional bus interface for communicating with an external LCD driver chip with the following setup.

- Set the PMC registers to configure the pins to digital mode.
- Set the PM registers to set the pins \overline{DBWR} and \overline{DBRD} multiplexing to output mode.
- Set the pin registers to set the pin output latches \overline{DBWR} and \overline{DBRD} multiplexes to "1"
- Set the PM register to set the PIN for DBD0 to DBD7 multiplexing to input mode. (When the output is multiplexed from DBD0 to DBD7, the hardware automatically switches to the output mode, and does not require the user to set it himself.)
- Set the pin register to set the PIN output latch for DBD0 to DBD7 multiplexing to "0"

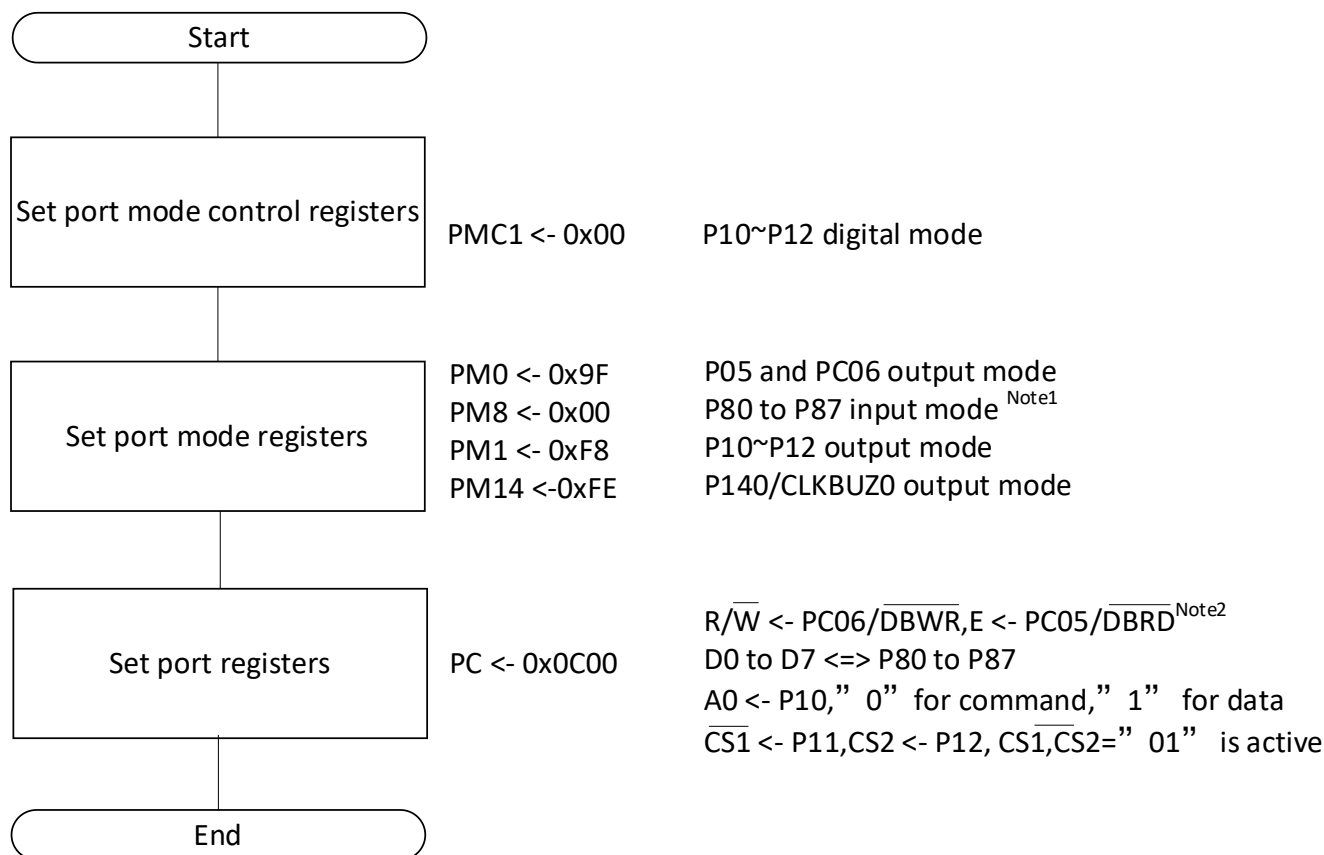
Figure 23-19 LCD bus pin setup flow



Note: 1 The P80 to P87 pins must be set to input mode to act as an LCD bidirectional communication bus.

2. When LBTL.bit7 (LBEL) = 0, the output latch registers of the P05 and P06 pins must be set to "1".

< 80 mode (using S1D15E00) >



Note: 1 The P80 to P 87 must be set to input mode to act as an LCD bidirectional communication bus.

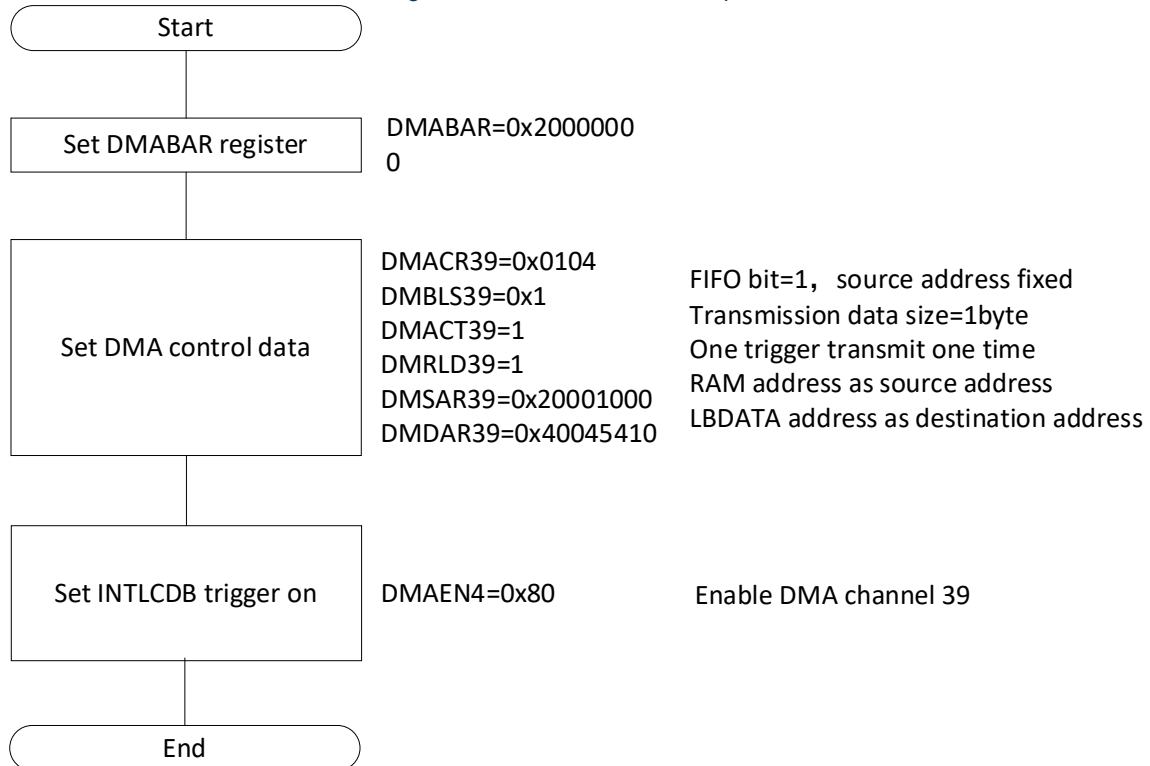
2. Because DBRD and DBWR are all active low, P05 And the P06 pin output latch register must be set to "1".

(5). DMA transport settings

- Set the DMABAR register
- Set the control data for DMA channel 39
- Set DMA channel 39 to boot

Note: The first time you transfer, you need to write the transfer content in the 0x200001000 to the LBDATA register through the normal write method.

Figure 23-20 DMA transfer setup flow



When reading data from the LCD driver chip to the LCD, set the RAM address to the destination address and the LBDATA address to the source address.

(6). LCD bus transmission

- **Use S1D15E00 (EPSON).**

Data/commands transmitted via DMA should be stored in advance in the RAM address. For example, the settings are as follows. Please refer to the S1D15E00 data sheet for details of the command.

The RAM address	value	command	description
0x20001000	A0H	ADC select SEG0→SEG131: 0(H) → column addresses →83(H).	LCD driver initialization (S1D15E0).
0x20001001	C0H	Universal Output Mode Selection COM Normal Scan Direction, COM0 → COM95	
0x20001002	A6H	Normal display/reverse display RAM data: High potential (normal) when lcd on	
0x20001003	A4H	Displays all points on/off Normal display mode	
0x20001004	61H	Duty cycle setting (2 byte). Set the duty cycle to 1/8 and the starting point of the block to 0 (COM0 to 3).	
0x20001005	00H		
0x20001006	81H	Electrical parameters (2 byte). Set the electrical parameter register 05H (small).	
0x20001007	05H		
0x20001008	40H	Temperature gradient settings Set the temperature gradient to 0.06%/°C	
0x20001009	8AH	Displays the starting row setting (2 byte). Set the display start line to 0 Set the page address The page address is set to 0 Set the column address The high 4 bits displayed are 0000B, and the low 4 bits are the default (0000B).	Displays the settings
0x2000100A	00H		
0x2000100B	B0H		
0x2000100C	10AH		
0x2000100D	7FH	Write the displayed data	The displayed data is: "E"
0x2000100E	49H		
0x2000100F	49H		
0x20001010	49H		
0x20001011	41H		
0x20001012	AFH	Turn on display	Displays Startup

- **Use PCF2119x (NXP Semiconductors).**

Only the initial procedure of the LCD driver chip is described here, and for the DMA section, please refer to the previous S1D15E00 example. For more information about commands, see the pcf2119x data sheet.

value	command	description
34H	Feature Settings ^{Note} 8 bits data length, 2-wire ×16 characters, 1:18 multi-drive mode	LCD driver initialization (PCF2119x).
34H	Feature Settings ^{Note}	
34H	Feature Settings ^{Note}	
34H	Feature Settings ^{Note}	
08H	Display controls Display, cursor, and character flashing off.	
01H	Clear the screen Determines the value.	
07H	Portal mode setting Address plus 1 to show shifts	

Note: Specify the same instruction to ensure that there is enough BF check time.

The LCD bus communication flow without DMA mode is as follows:

When $f_{OSC} = 450\text{kHz}$, it takes about $330\mu\text{s}$ (165 driver oscillator cycles) to complete the clean-screen command, and about $6\mu\text{s}$ (3 driver oscillator cycles) for other commands.

Perform a busy flag check operation on PCF2119x. The busy flag (BF) indicates a busy state, with BF maintaining "1" until initialization ends and busy lasting 2ms. When $RS=0$ and $R/\bar{W}=1$, the busy flag is output to pin DB7.

The DB7 pin of the LCD bus can be used as a busy flag to determine whether the internal operation of the driver chip is completed by reading the 7th bit of LBDATA/LBDATAR.

Chapter 24 Enhanced DMA

24.1 Features of the 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 24-1.

Table 24-1 Specifications of the DMA (1/2).

Item		specification
Startup Source		Up to 40 startup sources
Assignable control data		40 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. The value of the register continues after reloading into the DMCTj register.
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 24-5 DMA startup source and vector addresses".

Table 24-1 Specifications of the DMA (2/2).

project		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.
The transfer begins		If the DMAENi0~DMAENi7 position 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"> Place DMAENi0~DMAENi7 at position "0" (boot is prohibited). When the DMACTj register changes from "1" to "0" at the end of the data transfer
	Repeat pattern	<ul style="list-style-type: none"> Place DMAENi0~DMAENi7 at position "0" (boot is prohibited). Data transfer ends when the RPTINT bit is "1" (interrupts are allowed) and the DMCTj register changes from "1" to "0"

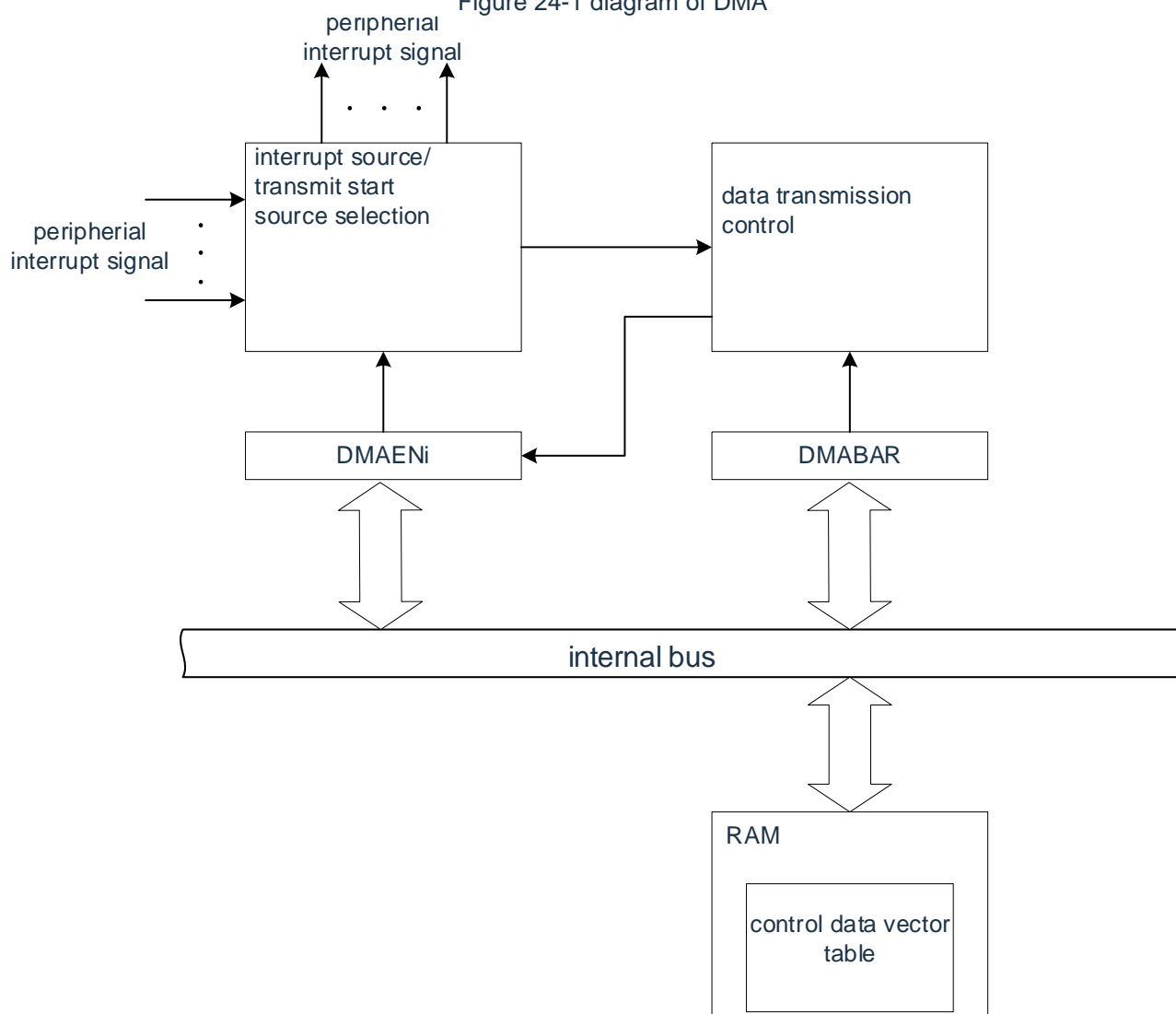
Note In deep sleep mode, the flash memory stops running and cannot be used as a DMA transfer source.

Remark i=0~4, j=0~39

24.2 Structure of the DMA

A block diagram of the DMA is Figure 24-1241

Figure 24-1 diagram of DMA



24.3 Control registers of the DMA

The registers that control the DMA are shown in Table 24-2.

Table 24-2 Control registers of the DMA

Register name	symbol
Peripheral enable register 1	PER1
DMA boot enable register 0	DMAEN0
DMA boot enable register 1	DMAEN1
DMA boot enable register 2	DMAEN2
DMA boot enable register 3	DMAEN3
DMA boot enable register 4	DMAEN4
DMA base address register	DMABAR

The control data for the DMA is shown in Table 24-3.

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 24-3 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

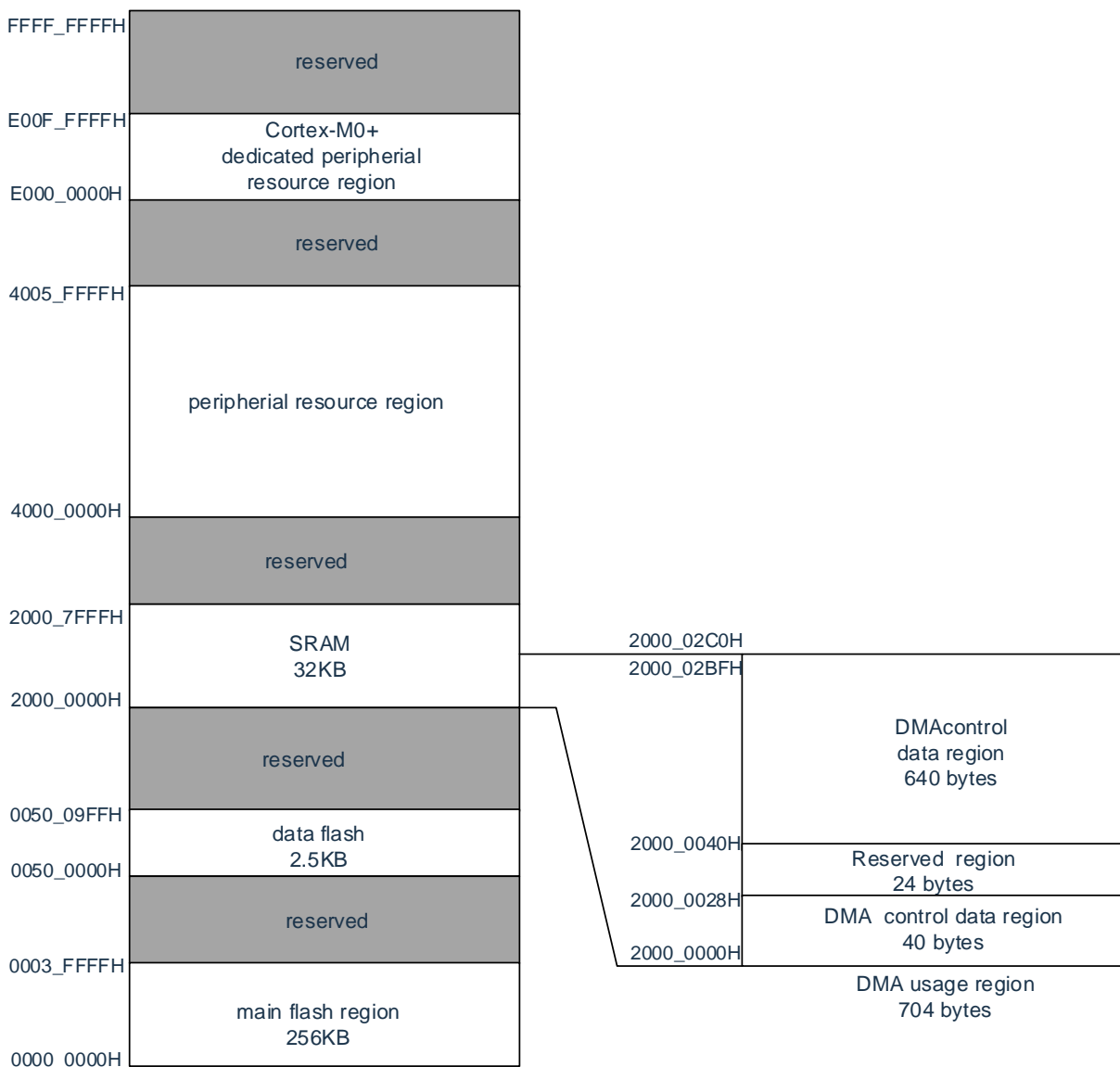
Note j = 0 ~ 39

24.3.1 DMA control data areas and DMA vector table areas allocation

The DMABAR register assigns the DMA control data and the 704-byte locale of the vector table to the RAM area.

An example of a memory image with the DMABAR register set to "20000000H" is shown in Figure 24-2. The 640 bytes of DMA control data area can be used as RAM in the space not used by the DMA.

Figure 24-2 memory image when the DMABAR register is set to "20000000H"



Note: The division of the system address area in this figure is BAS32G139 as an example.

24.3.2 Control data allocation

Starting from the start address, follow DMACR_j, DMBLS_j, DMACT_j, DMRLD_j, DMSAR_j, DMDAR_j (j=0~39) 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 24-3.

Note 1 The DMAEN_{i0}~DMAEN_{i7} bits of the corresponding DMAEN_i (i=0~4) must be "0" (No Boot) when changing DMCR_j, DMBLS_j, DMACT_j, DMRLD_j, DMSAR_j, Data from dmdaR_j registers.

2. DMACR_j, DMBLS_j, DMACT_j, DMRLD_j, DMSAR_j and DMSAR_j cannot be transmitted via DMA Access to DMDAR_j.

Figure 24-3 controls the allocation of data (DMABAR is set to 2000000H).

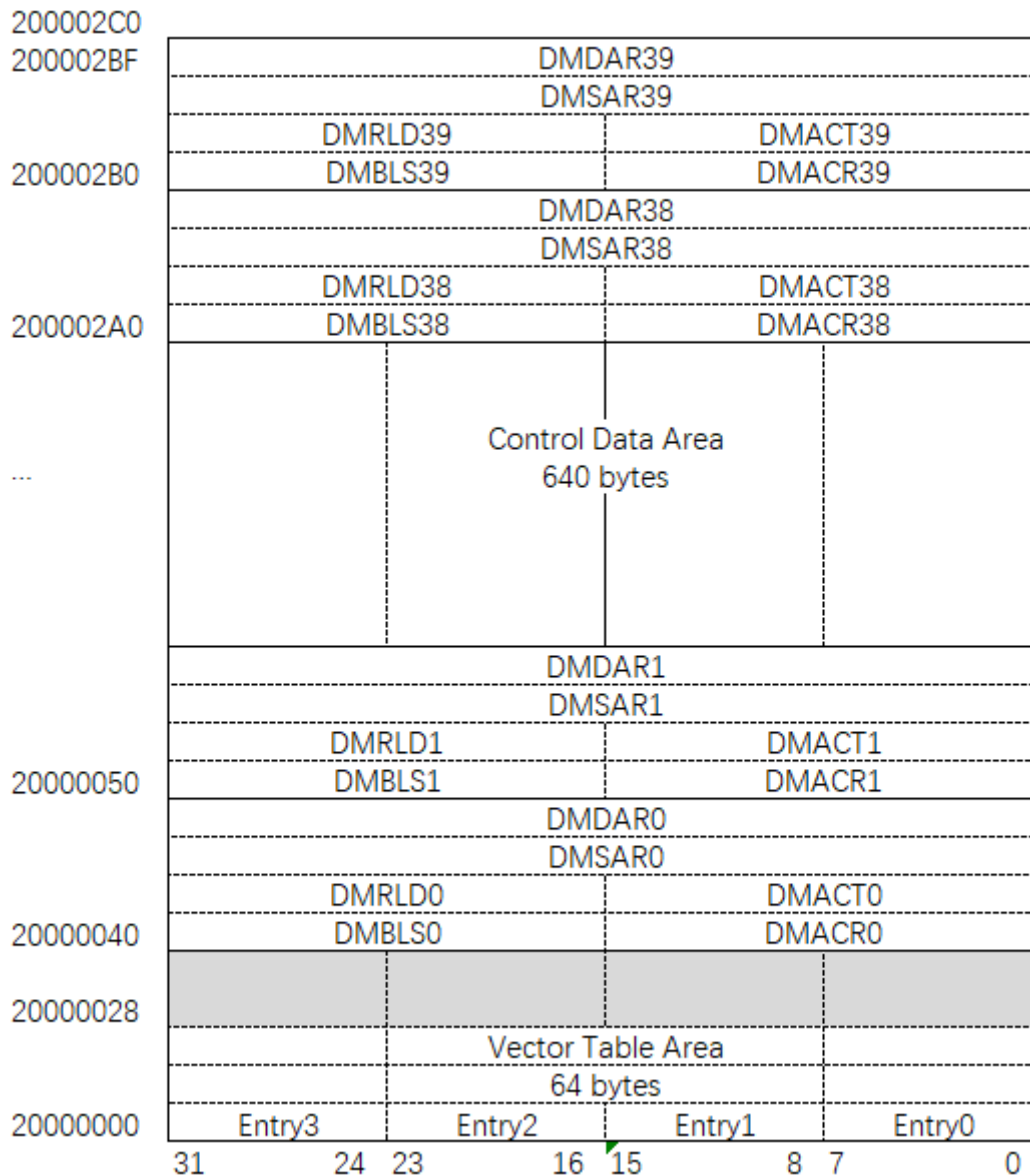


Table 24-4 controls the start address of the data

j	Address	j	Address
19	baseaddr+170H	39	baseaddr+2B0H
18	baseaddr+160H	38	baseaddr+2A0H
17	baseaddr+150H	37	baseaddr+290H
16	baseaddr+140H	36	baseaddr+280H
15	baseaddr+130H	35	baseaddr+270H
14	baseaddr+120H	34	baseaddr+260H
13	baseaddr+110H	33	baseaddr+250H
12	baseaddr+100H	32	baseaddr+240H
11	baseaddr+F0H	31	baseaddr+230H
10	baseaddr+E0H	30	baseaddr+220H
9	baseaddr+D0H	29	baseaddr+210H
8	baseaddr+C0H	28	baseaddr+200H
7	baseaddr+B0H	27	baseaddr+1F0H
6	baseaddr+A0H	26	baseaddr+1E0H
5	baseaddr+90H	25	baseaddr+1D0H
4	baseaddr+80H	24	baseaddr+1C0H
3	baseaddr+70H	23	baseaddr+1B0H
2	baseaddr+60H	22	baseaddr+1A0H
1	baseaddr+50H	21	baseaddr+190H
0	baseaddr+40H	20	baseaddr+180H

Note baseaddr: The setting value of the DMABAR register

24.3.3 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 24-5. The vector table of each startup source has 1 byte, saves the data from "00H" to "27H", and selects 1 from 40 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 "27H" corresponding to the startup source.

Note that the DMAENi0~DMAENi7 bits of the corresponding DMAENi (i=0~4) register must be "0" (Disable Startup) when changing the start address of the DMA control data area set in the vector table.

Figure 24-4 control the start address and vector table of the data
The case where the DMABAR register is set to "2000000H" (example).

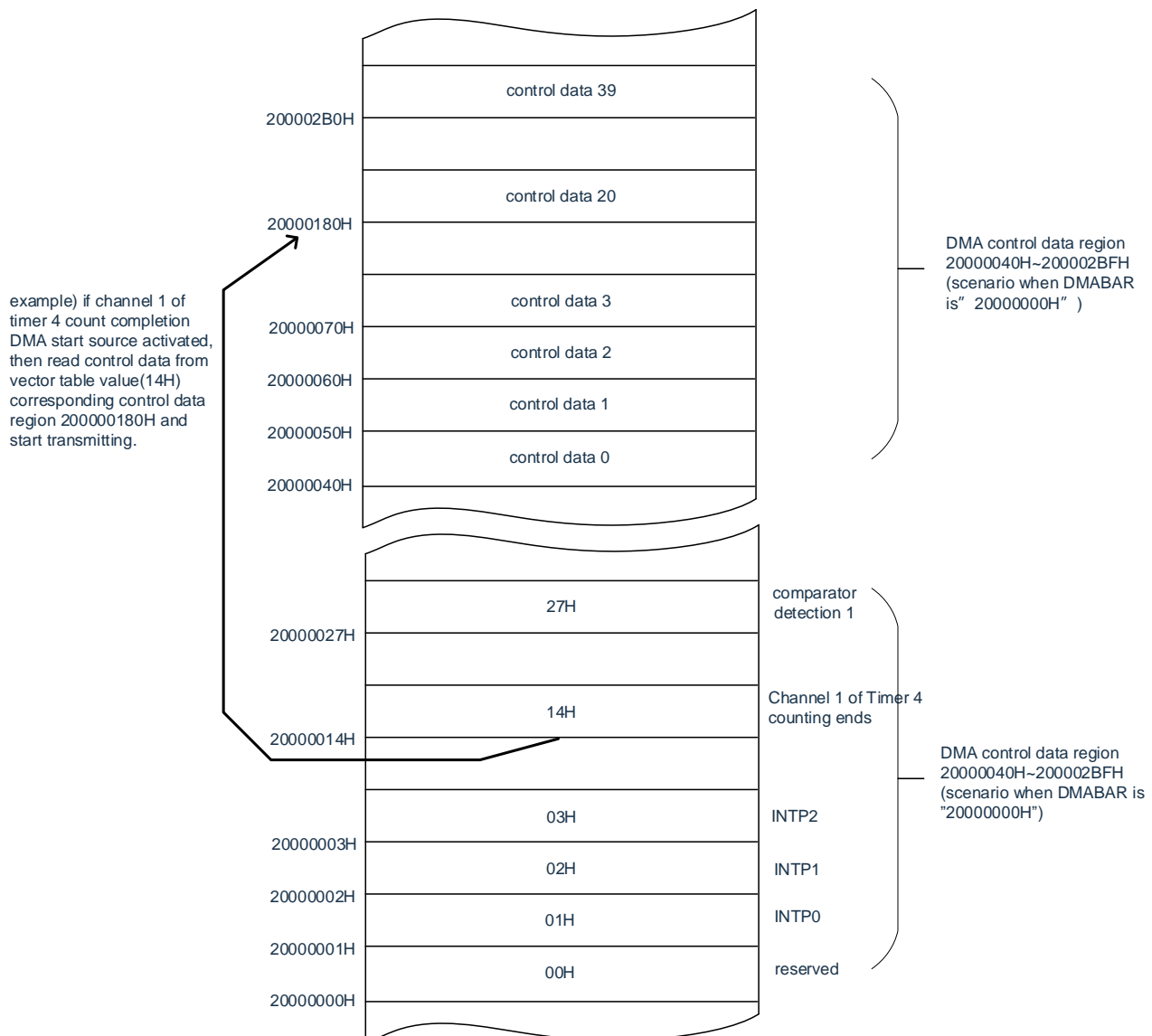


Table 24-5 DMA startup source and vector addresses

DMA Startup Source (interrupt request)	The source	Vector address	Priority
The LCD bus transfer ends with an interrupt ^{note}	0	The setting address of the DMABAR	<div>high</div> <div>↑</div> <div>↓</div> <div>lo</div>
INTP0	1	The setting address of the DMABAR	
INTP1	2	The setting address of the DMABAR	
INTP2	3	The setting address of the DMABAR	
INTP3	4	The setting address of the DMABAR	
INTP4	5	The setting address of the DMABAR	
INTP5	6	The setting address of the DMABAR	
INTP6	7	The setting address of the DMABAR	
INTP7	8	The setting address of the DMABAR	
Key input	9	The setting address of the DMABAR	
End of A/D conversion	10	The setting address of the DMABAR	
Transmit end received by UART0/Transmit End of CSI01 or	11	The setting address of the DMABAR register is +0BH	
Transmission end for UART0 sent/transfer end for CSI00 or	12	The setting address of the DMABAR register is +0CH	
Transmit end received by UART1/Transmit end of CSI11 or	13	The setting address of the DMABAR register is +0DH	
Transmission end of UART1 sent/transfer end of CSI10 or	14	The setting address of the DMABAR register is +0EH	
Transmit end received by UART2/Transmit End of CSI21 or	15	The setting address of the DMABAR register is +0FH	
Transmission end for UART2 send/transfer end for CSI20 or	16	The setting address of the DMABAR register is +10H	
IICA0 communication ended.	17	The setting address of the DMABAR	
High-speed SPI0 communication end note	18	The setting address of the DMABAR	
Timer4's channel 0 count or snap end	19	The setting address of the DMABAR	
Timer4's channel 1 count or snap end	20	The setting address of the DMABAR	
Timer4's channel 2 count or snap end	21	The setting address of the DMABAR	
Timer4's channel 3 count or snap end	22	The setting address of the DMABAR	
A 15-bit interval timer produces a counting	23	The setting address of the DMABAR	
Flash read and write erase ends	24	The setting address of the DMABAR	
High-speed SPI1 communication end note	25	The setting address of the DMABAR	
Overflow of TimerC	26	The setting address of the DMABAR	
The comparison of TimerM matches A0	27	The setting address of the DMABAR	
The comparison of TimerM matches B0	28	The setting address of the DMABAR	
The comparison of TimerM matches C0	29	The setting address of the DMABAR	
The comparison of TimerM matches D0	30	The setting address of the DMABAR	
The comparison of TimerM matches A1	31	The setting address of the DMABAR	
The comparison of TimerM matches B1	32	The setting address of the DMABAR	
The comparison of TimerM matches C1	33	The setting address of the DMABAR	
The comparison of TimerM matches D1	34	The setting address of the DMABAR	
The comparison of TimerB matches A	35	The setting address of the DMABAR	
The comparison of TimerB matches B	36	The setting address of the DMABAR	
TimerA underflow	37	The setting address of the DMABAR	
The comparator detects 0	38	The setting address of the DMABAR	
Comparator detection 1	39	The setting address of the DMABAR	

Note: LCDB, SPIHS0, SPIHS1 is BAT32A279 proprietary, so when BAT32A239 products, the startup source 0,18,25 is reserved.

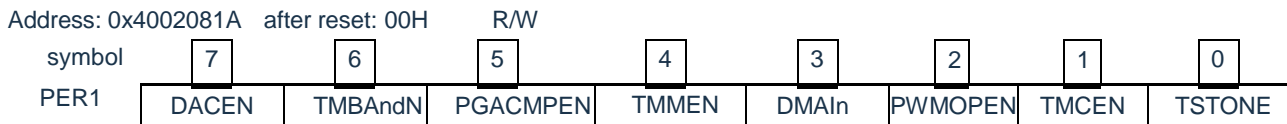
24.3.4 Peripheral enable register 1 (PER1).

The PER1 register is a register that is set to allow or disable clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking hardware that is not in use.

To use DMA, bit3 (DMAEN) must be set to "1".

The PER1 register is set via the 8-bit memory operation instruction. After generating a reset signal, the value of this register changes to "00H".

Figure 24-5 Peripheral enable the format of register 1 (PER1).



DMAEN	Provides control of the input clock of the DMA
0	Stop providing the input clock. •DMA cannot run.
1	An input clock is provided. • DMA can run.

24.3.5 DMA control register j (DMACRj) (j=0~39).

The DMACRj register controls the operating mode of the DMA.

Figure 24-6 DMA control register j (DMACRj).

Address: Refer to "24.3.2Control data".

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
	S		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

S	The choice of the length of the data to be transferred
00	8 bits
01	16 bits
10	32 bits
11	Prohibit settings

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/disable for chain transfers
0	Chain transfer is prohibited.
1	Chain transfer is allowed.
The DMACR39 register must be placed at chne position "0" (chain transfer is prohibited).	

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

Note that access to the DMACRj register cannot be made via DMA transfer.

24.3.6 DMA block size register j (DMBLSj) (j=0~39).

This register sets the block size of the data transfer that is started once.

Figure 24-7 DMA block size register j (DMBLSj).

Address: Refer to "24.3.2Control data". 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	The size of the transfer block		
	8-bit transmission	16-bit transmission	32-bit transmission
00H	Prohibit settings	Prohibit settings	Prohibit settings
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

Note 1. Access to the DMBLSj register cannot be made via DMA transfer.

24.3.7 DMA transmit count register j (DMACTj) (j=0~39).

This register sets the number of data transfers for the DMA. 1 is minus each time DMA transfer is initiated.

Figure 24-8 DMA Transfer Times Register j (DMACTj).

Symbol:	15	14	13	12	11	10	9	8
DMACTj	DMACTj15	DMACTj14	DMACTj13	DMACTj12	DMACTj11	DMACTj10	DMACTj9	DMACTj8
	7	6	5	4	3	2	1	0
	DMACTj7	DMACTj6	DMACTj5	DMACTj4	DMACTj3	DMACTj2	DMACTj1	DMACTj0

Address: Refer to "24.3.2Control data24.3.224.3.2Control data Control data allocation24.3.2Control data".
After reset: indefinite value R/W

DMACTj	Number of transfers
00H	Prohibit settings
01H	1 time
02H	2 times
03H	3 times
⋮	⋮
FDH	253 times
FEH	254 times
FFH	255 times
⋮	⋮
FFFFH	65535 times

Note 1 Access to the DMACTj register cannot be made via DMA transfer.

24.3.8 DMA number of transfers reloads register j (DMRLDj) (j=0~39).

This register sets the initial value of the number of transfer registers in repeat mode. In repeat mode, because the value of this register is reloaded into the DMACT register, the setting value must be the same as the initial value of the DMACT register.

Figure 24-9 DMA transfer times reload register j (DMRLDj).

Address: Refer to "24.3.2Control data". 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

Note 1. Access to the DMRLDj register cannot be made via DMA transfer.

24.3.9 DMA source address register j (DMSARj) (j=0~39).

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 24-10 DMA source address register j (DMSARj).

Address: Refer to "24.3.2Control data".

After reset: indefinite value R/W

symbol	31	30	29	28	27	26	25	24
DMSARj	DMSARj3 1	DMSARj3 0	DMSARj2 9	DMSARj2 8	DMSARj2 7	DMSARj2 6	DMSARj2 5	DMSARj2 4
	23	22	21	20	19	18	17	16
	DMSARj2 3	DMSARj2 2	DMSARj2 1	DMSARj2 0	DMSARj1 9	DMSARj1 8	DMSARj1 7	DMSARj1 6
	15	14	13	12	11	10	9	8
	DMSARj1 5	DMSARj1 4	DMSARj1 3	DMSARj1 2	DMSARj1 1	DMSARj1 0	DMSARj9	DMSARj8
	7	6	5	4	3	2	1	0
	DMSARj7	DMSARj6	DMSARj5	DMSARj4	DMSARj3	DMSARj2	DMSARj1	DMSARj0

Note 1 Access to DMSARj registers cannot be made via DMA transfer.

24.3.10 DMA destination address register j (DMDARj) (j=0~39).

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 24-11 DMA destination address register j (DMDARj).

Address: Refer to "24.3.2Control data".

After reset: indefinite value R/W

symbol	31	30	29	28	27	26	25	24
DMDARj	DMDARj3 1	DMDARj3 0	DMDARj2 9	DMDARj2 8	DMDARj2 7	DMDARj2 6	DMDARj2 5	DMDARj2 4
	23	22	21	20	19	18	17	16
	DMDARj2 3	DMDARj2 2	DMDARj2 1	DMDARj2 0	DMDARj1 9	DMDARj1 8	DMDARj1 7	DMDARj1 6
	15	14	13	12	11	10	9	8
	DMDARj1 5	DMDARj1 4	DMDARj1 3	DMDARj1 2	DMDARj1 1	DMDARj1 0	DMDARj9	DMDARj8
	7	6	5	4	3	2	1	0
	DMDARj7	DMDARj6	DMDARj5	DMDARj4	DMDARj3	DMDARj2	DMDARj1	DMDARj0

Note: Access to the DMDARj register cannot be made via DMA transfer.

24.3.11 DMA boot enable register i (DMAENi) (i=0~4).

This is the 8-bit register that controls enable or disables the startup of the DMA through each interrupt source. The interrupt source corresponds to the DMAENi0~DMAENi7 bits as shown in Table 24-6.

The DMAENi register can be set via an 8-bit memory operation instruction.

Note 1 The DMAENi0~DMAENi7 bit must be changed in places where the boot source to the bits is not generated.

2. Access to DMAENi registers cannot be carried out via DMA transmission.

3. The assigned function varies from product to product, and the position "0" without the assigned function must be placed.

Figure 24-12 DMA initiates the format of allowing registers i (DMAENi) (i=0~4).

address: 40005000H(DMAEN0), 40005001H(DMAEN1),
40005002H(DMAEN2), 40005003H(DMAEN3),
40005004H(DMAEN4)

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 boot enable i7
0	Disable startup.
1	Allow startup.
Depending on the condition under which the end-of-transmit interrupt occurs, the DMAENi7 bit becomes	

DMAENi6	DMA boot enable i6
0	Disable startup.
1	Allow startup.
Depending on the condition in which the end-of-transmit interrupt occurs, the DMAENi6 bit becomes "0"	

DMAENi5	DMA boot enable i5
0	Disable startup.
1	Allow startup.
Depending on the conditions under which the end-of-transmit interrupt occurs, the DMAENi5 bit becomes	

DMAENi4	DMA Boot enable i4
0	Disable startup.
1	Allow startup.
Depending on the condition in which the end-of-transfer interrupt occurs, the DMAENi4 bit becomes "0"	

DMAENi3	DMA boot enable i3
0	Disable startup.
1	Allow startup.
Depending on the conditions under which the end-of-transmit interrupt occurs, the DMAENi3 bit becomes	

DMAENi2	DMA boot enable i2
0	Disable startup.
1	Allow startup.
Depending on the condition in which the end-of-transfer interrupt occurs, the DMAENi2 bit becomes "0"	

DMAENi1	DMA boot enable i1
0	Disable startup.
1	Allow startup.
Depending on the condition in which the end-of-transmit interrupt occurs, the DMAENi1 bit becomes "0"	

DMAENi0	DMA boot enable i0
0	Disable startup.
1	Allow startup.
Depending on the condition under which the end-of-transfer interrupt occurs, the DMAENi0 bit becomes	

Table 24-6 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	INTP6	INTP5	INTP4	INTP3	INTP2	INTP1	INTP0	The LCD bus transfer ends with an interrupt
DMAEN1	Transmit end received by UART2/Transmit End for CSI21 or Transmit End for Buffer Null/IIC21	Transfer end for UART1 transmission/transmit end for CSI10 or transfer end for buffer null/IIC10	Transmit end received by UART1/Transmit End for CSI11 or Transmit End for Buffer Null/IIC11	Transfer end for UART0/transmit end for CSI00 or transfer end for buffer null/IIC00	Transmit end received by UART0/Transmit End for CSI01 or Transmit End for Buffer Null/IIC01	End of A/D conversion	KEY input	INTP7
DMAEN2	15-bit interval timer interrupt	The end of the count or the end of the capture of channel 3 of the timer array unit 0	The end of the count or the end of the capture of channel 2 of the timer array unit 0	The end of the count or the end of the capture of channel 1 of the timer array unit 0	The end of the count or the end of the snap of channel 0 of the timer array unit 0	High-speed SPI0 communication end note	IICA0 communication ended	Transfer end for UART2 transmission/transmit end for CSI20 or transfer end for buffer null/IIC20
DMAEN3	The comparison of TimerM	The comparison of timerM	The comparison of TimerM	The comparison of TimerM	The comparison of TimerM	Overflow of TimerC	High-speed SPI1 communication	Flash erase/write end
DMAEN4	Comparator detection 1	The comparator detects 0	TimerA Overflow	The comparison of TimerB	The comparison of TimerB	The comparison of TimerM	The comparison of TimerM	The comparison of TimerM

Note: LCDB, SPIHS0, SPIHS1 are proprietary to BAT32A279, so when BAT32A239 products are used, bit 0 of DMAEN0, bit 2 of DMAEN2 and bit 1 of DMAEN3 must be placed at 1'b0.

Note i = 0 ~ 4

24.3.12 DMAENi position register (DMSETi).

This is the DMA boot allowing the register DMAENi to set the position register, set the corresponding bit to 1 to set the corresponding position bit of DMAENi to 1.

Figure 24-13 Format of the DMAENi position register (DMSETi) (i=0~4).

address: 40005018H (DMSET0) , 40005019H (DMSET1) ,
4000501AH (DMSET2) , 4000501BH (DMSET3) ,
4000501CH (DMSET4)

after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
DMSETi	DMSETi7	DMSETi6	DMSETi5	DMSETi4	DMSETi3	DMSETi2	DMSETi1	DMSETi0

DMSETin	The position register for DMAENin
0	No assertion is generated.
1	Set the bit n of DMCENi to 1

Note: i=0~4, n=0~7

24.3.13 DMAENi reset register (DMCLRi).

This is the reset register that DMA initiates to allow register DMAENi, and setting the corresponding bit to 1 resets the corresponding bit of DMAENi to 0.

Figure 24-14 DMAENi reset register (DMCLRi) (i=0~4).

address: 40005020H (DMCLR0) , 40005021H (DMCLR1) ,
40005022H (DMCLR2) , 40005023H (DMCLR3) ,
40005024H (DMCLR4)

after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
DMCLRi	DMCLRi7	DMCLRi6	DMCLRi5	DMCLRi4	DMCLRi3	DMCLRi2	DMCLRi1	DMCLRi0

DMSETin	Reset register for DMAENin
0	No reset is generated.
1	Reset the bit n of DMCENi to 0

Note: i=0~4, n=0~7

24.3.14 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.

Note 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 "24.3.1DMA control data areas and DMA vector table areas" Note.
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 24-15 DMA Base Address Register (DMABAR).

Address: 40005008H After reset: 00000000H R/W

symbol	31	30	29	28	27	26	25	24
DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj
Rj	31	30	29	28	27	26	25	24
	23	22	21	20	19	18	17	16
DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj
	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8
DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	DMABARj	0	0
	15	14	13	12	11	10		
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	0

24.4 Operation of the 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 40 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=0~39) 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.

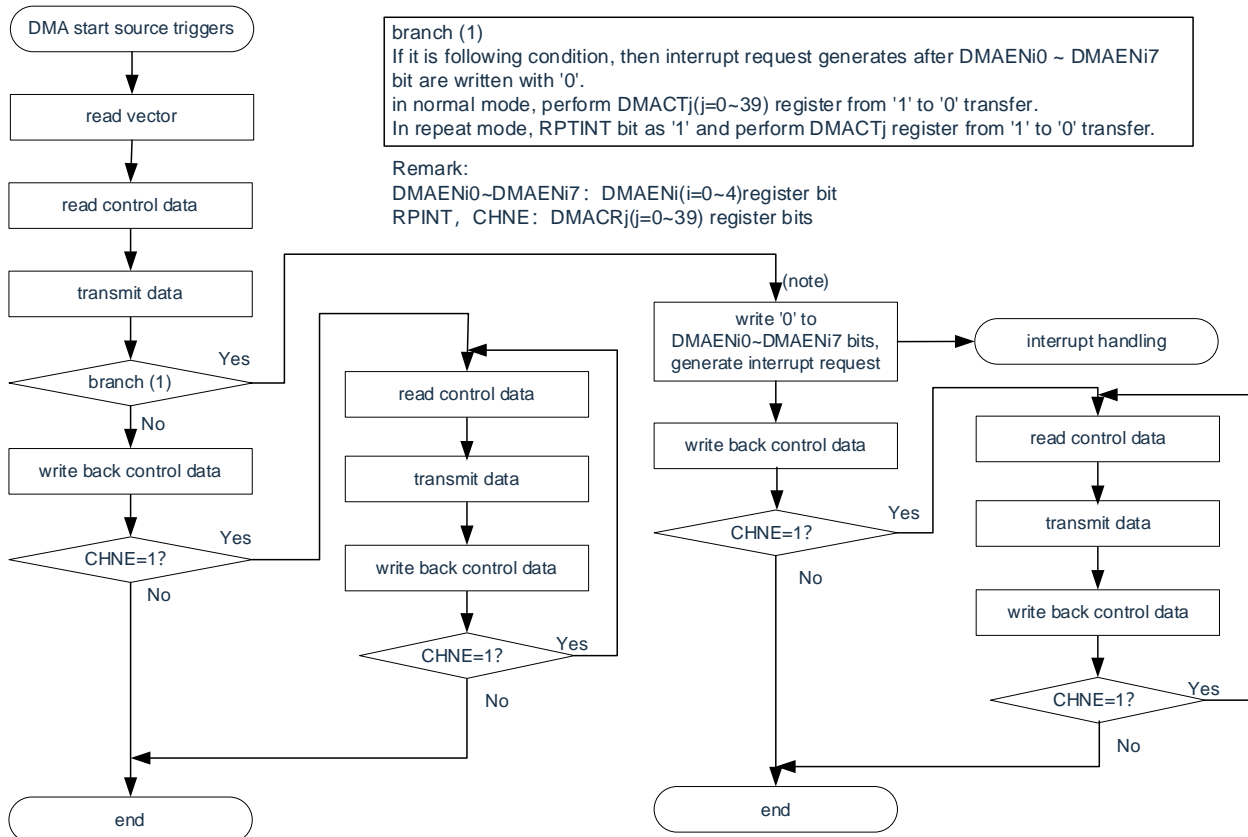
24.4.1 Startup Source

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=0~4) register. When the data transmission (in the case of chain transmission, continuous initial transmission) is set to the DMAENi0~DMAENi7 position of the corresponding DMAENi register in the DMA operation "0" (disables startup).

- In normal mode, a DMACTj (j=0~39) register is transferred to "0".
- In repeat mode, the RPTINT bit of the DMACRj register is "1" (interrupts are allowed) and the DMACTj register becomes "0".

The internal flowchart of the DMA is shown in Figure 24-16.

Figure 24-16 DMA's internal operating flowchart



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.

24.4.2 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=0~39) 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=0~4) register is generated Position "0" (disable startup).

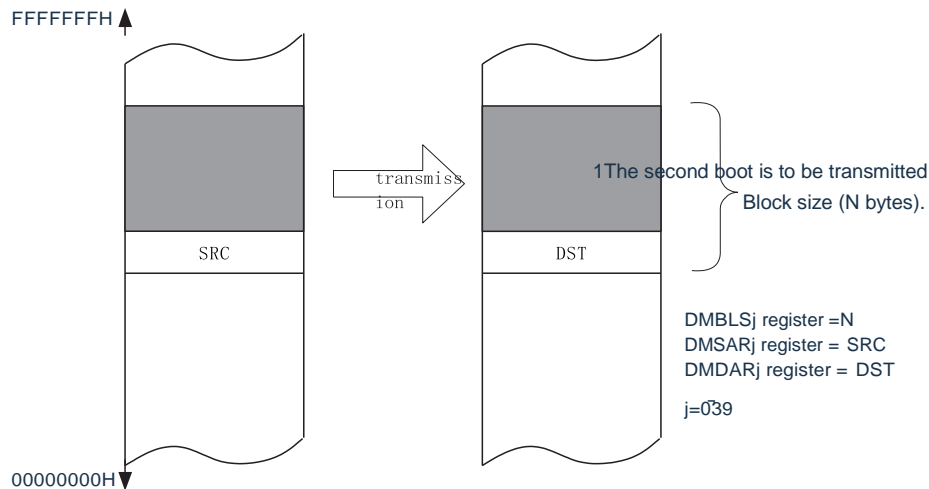
The register function and data transfer in normal mode are shown in Table 24-7 and Figure 24-17.

Table 24-7 Register functions 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
The number of DMA transfers reloads register j	DMRLDj	is not used ^{Note}
DMA source address register j	DMSARj	The address of the source from which the data is transmitted
DMA destination address register j	DMDARj	The address to which the data is transmitted

Note Initialization (00H) must be performed when parity error reset (RPERDIS=0) is allowed by the RAM parity error detection function. Note j = 0 ~ 39

Figure 24-17 Normal mode of data transfer



Settings of the DMACR				Control of the source address	Control of the destination	The source address after	The destination address after
DAMOD	SAMOD	RPTSEL	MODE				
0	0	X	0	fixed	fixed	SRC	Dst
0	1	X	0	Increasing	fixed	SRC+N	Dst
1	0	X	0	fixed	Increasing	SRC	DST+N
1	1	X	0	Increasing	Increasing	SRC+N	DST+N

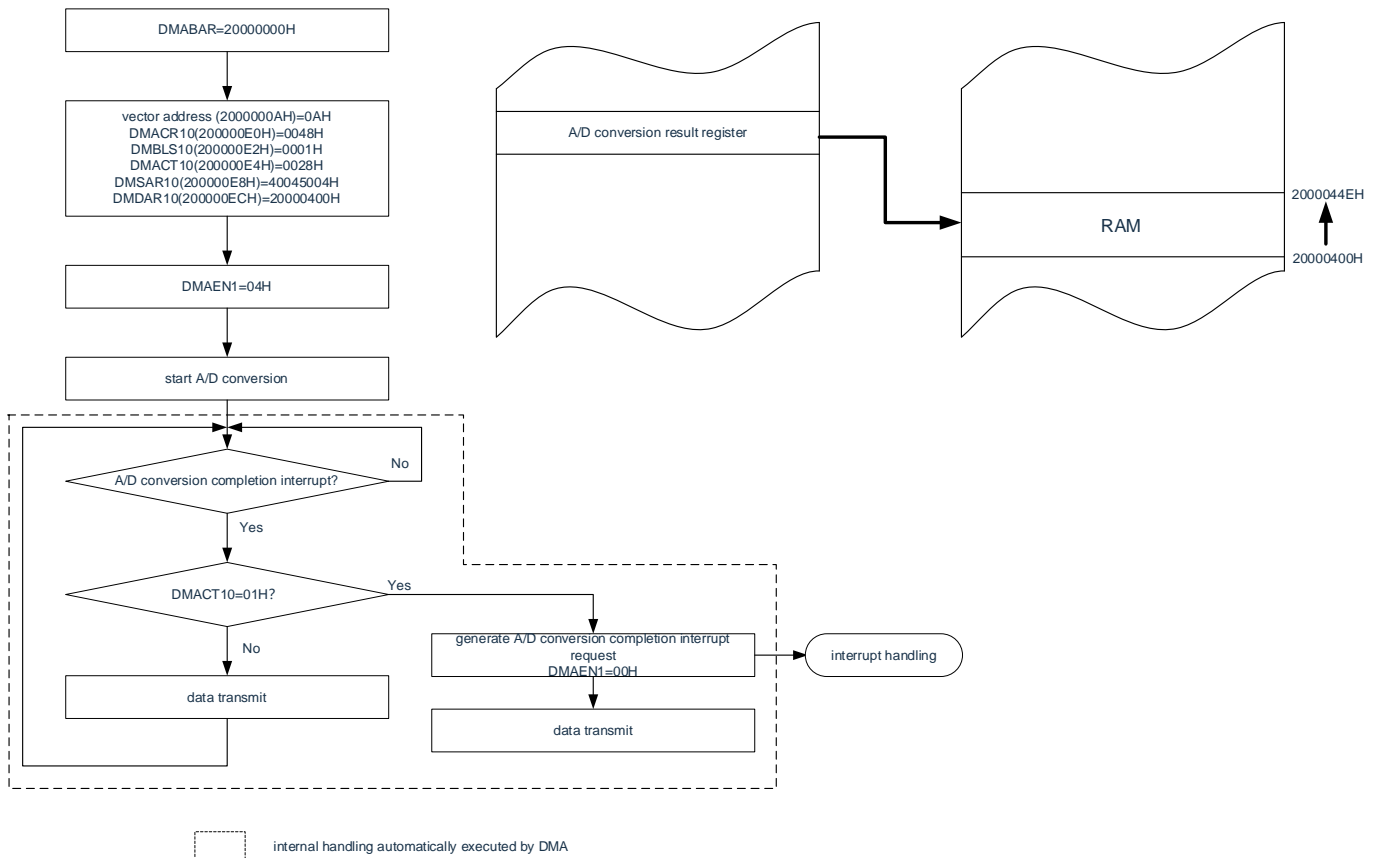
X: "0" or "1"

(1) Example 1 of the use of normal mode: Continuous A/D conversion results

The DMA is initiated via the A/D conversion end interrupt and the value of the A/D conversion result register is passed to RAM.

- Vector address assignment at 2000000AH, control data assignment at 200000E0H~2000000EFH.
- Transfer 2-byte data from the A/D conversion result registers (40045004H, 40045005H) 40 times to 20000400H of RAM ~80 bytes of ~2000044FH.

Figure 24-18 use of normal mode 1: Continuous A/D conversion results



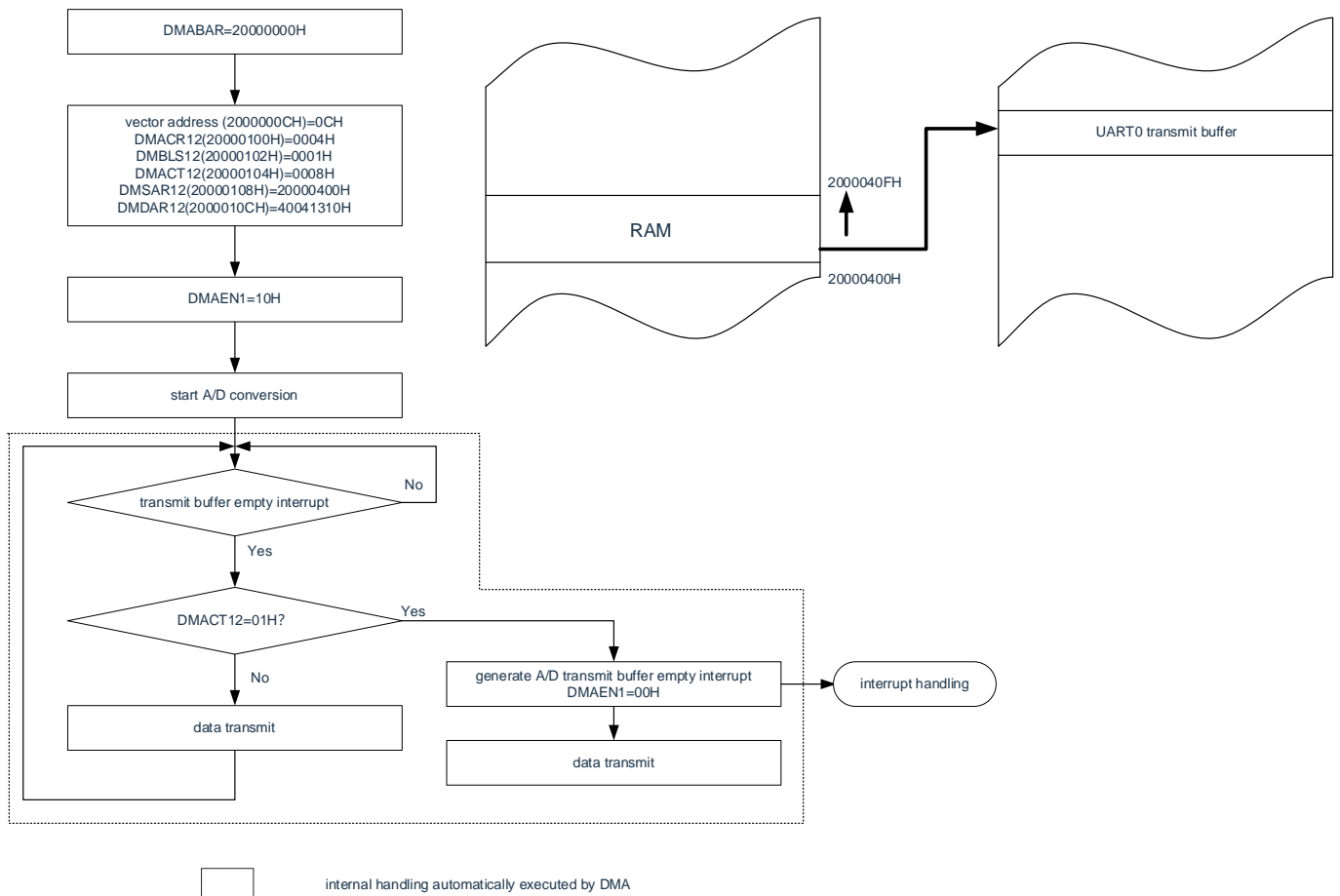
Because it is normal mode, the value of the DMRLD10 register is not used. However, when parity error reset (RPERDIS=0) is allowed to occur via the RAM parity error detection function, the DMRLD10 register must be initialized (0000H).

(2) Example 2 of the use of normal mode: UART0 transmits continuously

The DMA is started via an empty interrupt of the transmit buffer of UART0, and the value of the RAM is transmitted to the transmit buffer of UART0.

- Vector address assignment at 2000000CH, control data assignment at 20000100H~2000010FH.
- Transfer 8 bytes of RAM 20000400H~20000407H to the transmit buffer of UART0 (40041310H)。

Figure 24-19 Example of the use of normal mode 2: UART0 transmits continuously



Because it is normal mode, the value of the DMRLD12 register is not used. However, when parity error reset (RPERDIS=0) is allowed to occur via the RAM parity error detection function, the DMRLD12 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.

24.4.3 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=0~39) 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 DMACTRj register becomes "0", In the course of DMA operation, an interrupt request for the corresponding startup source is generated to the interrupt controller, and the DMAENi0 of the corresponding DMAENi (i=0~4) register is generated ~DMAENi7 position "0" (no boot). When the RPTINT bit of the DMACRj register is "0" (interrupt is prohibited), even if the data transfer of the DMACTRj register becomes "0", There is also no interrupt request, and the DMAENi0~DMAENi7 bits do not change to "0".

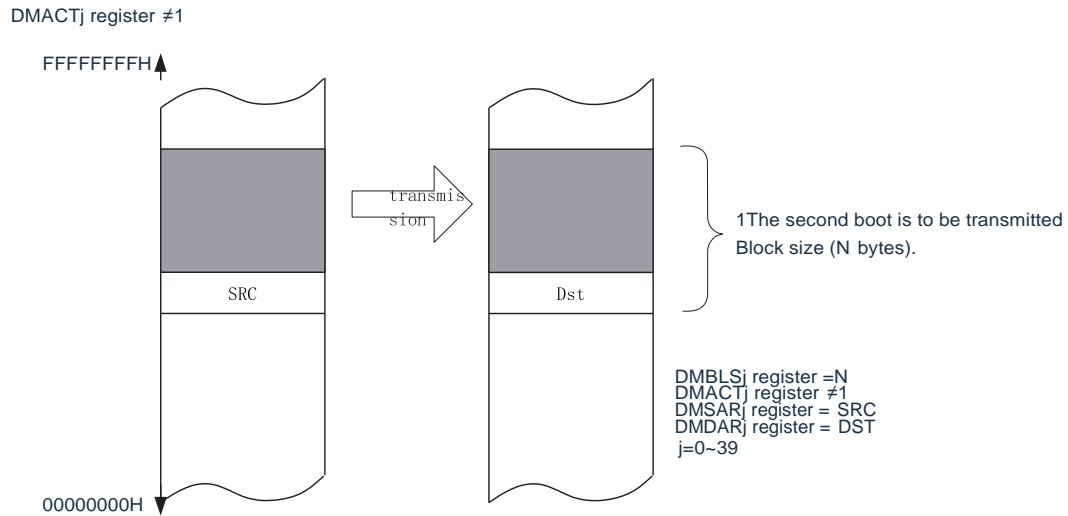
The repeating mode register function and data transfer are shown in Table 24-8 and Figure 24-20.

Table 24-8 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
The number of DMA transfers reloads register j	DMRLDj	Reload the value of this register into the DMACT register. (Initialization of 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	The address to which the data is transmitted

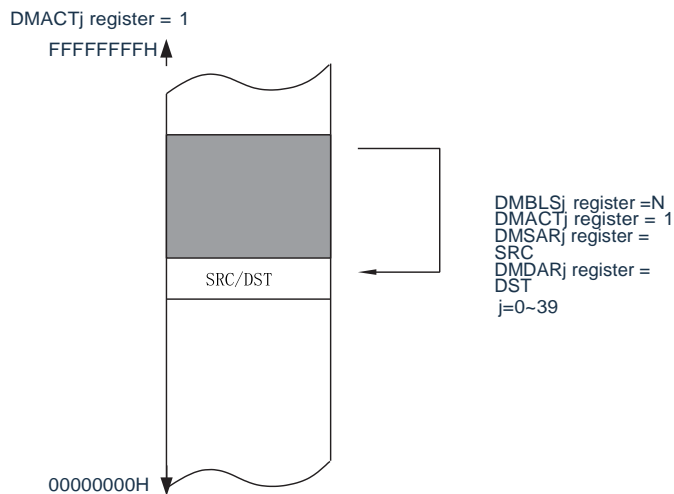
Note j = 0 ~ 39

Figure 24-20 repeating mode



Settings of the DMACR				Control of the source address	Control of the destination	The source address after	The destination address after
DAMOD	SAMOD	RPTSEL	MODE				
0	X	1	1	Duplicate	fixed	SRC+N	Dst
1	X	1	1	Duplicate	Increasing	SRC+N	DST+N
X	0	0	1	fixed	Duplicate	SRC	DST+N
X	1	0	1	Increasing	Duplicate	SRC+N	DST+N

X: "0" or "1"



Settings for the DMACR				Control of the source address	Control of the destination	The source address after	The destination address after
DAMOD	SAMOD	RPTSEL	MODE				
0	X	1	1	Duplicate areas	fixed	SRC	Dst
1	X	1	1	Duplicate	Increasing	SRC	DST+N
X	0	0	1	fixed	Duplicate	SRC	Dst
X	1	0	1	Increasing	Duplicate	SRC+N	Dst

SRC0: Initial value of source address DST0: Initial value X of destination address X: "0" or "1"

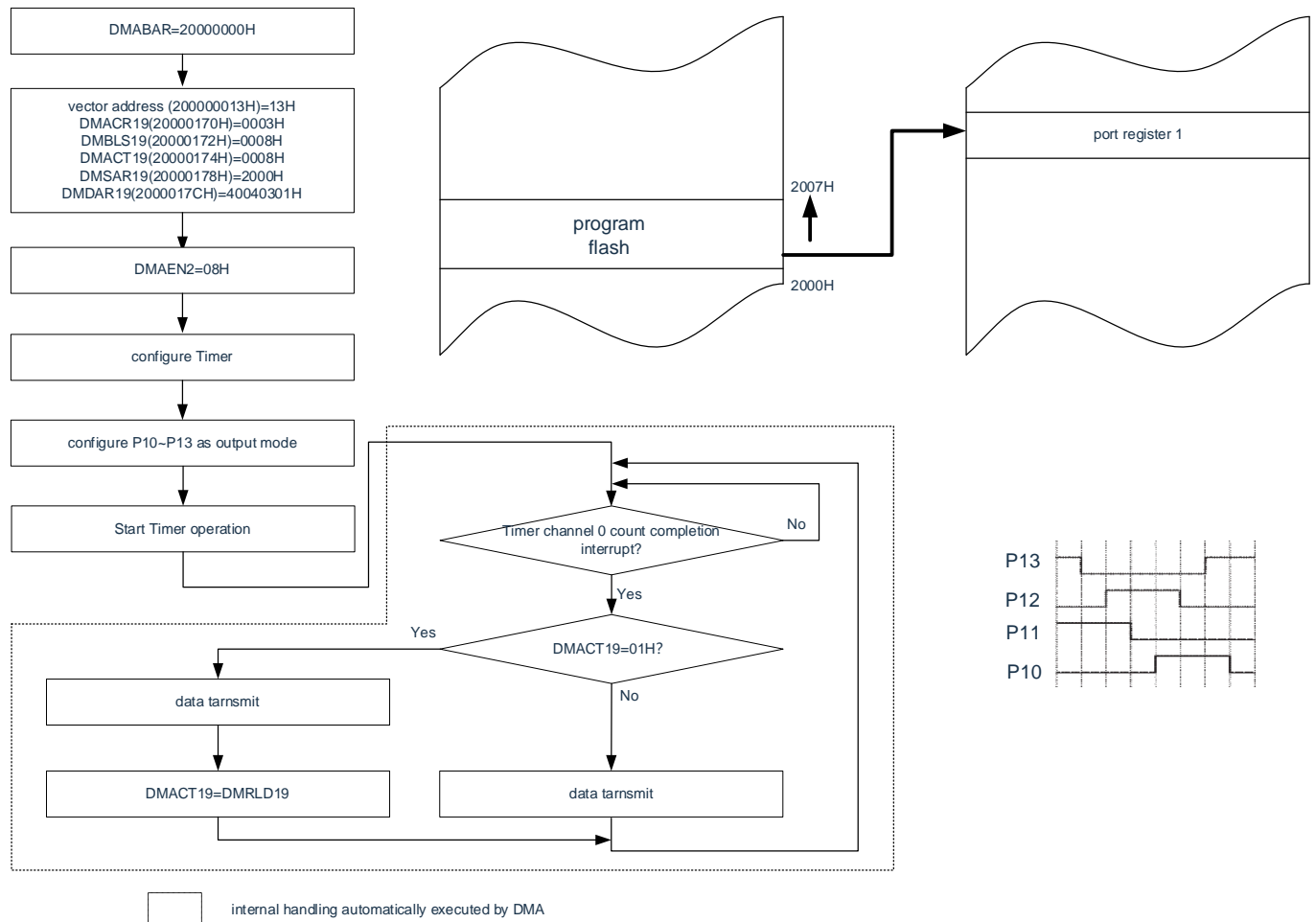
Note 1 When using the repeat pattern, the data length of the repeat zone must be set to within 65535 bytes.

(1) Example 1 of the use of repeat mode: Use the stepper motor of the port to control the pulse output

The DMA is initiated using the Timer4's Channel 0 Interval Timer feature, and the pattern of the motor control pulse saved in the code flash memory is transmitted to the universal port.

- Vector addresses are assigned at 20000013H, and control data is assigned at 20000170H~2000017FH.
- Transfer 8 bytes of code flash memory from 02000H to 02007H to port register 1 (40040301H).
- Disable repeat mode interrupts.

Figure 24-21 repeat mode example 1: Using the stepper motor of the port to control the pulse output



To stop the output, bit3 of DMAEN2 must be cleared after stopping the operation of the timer.

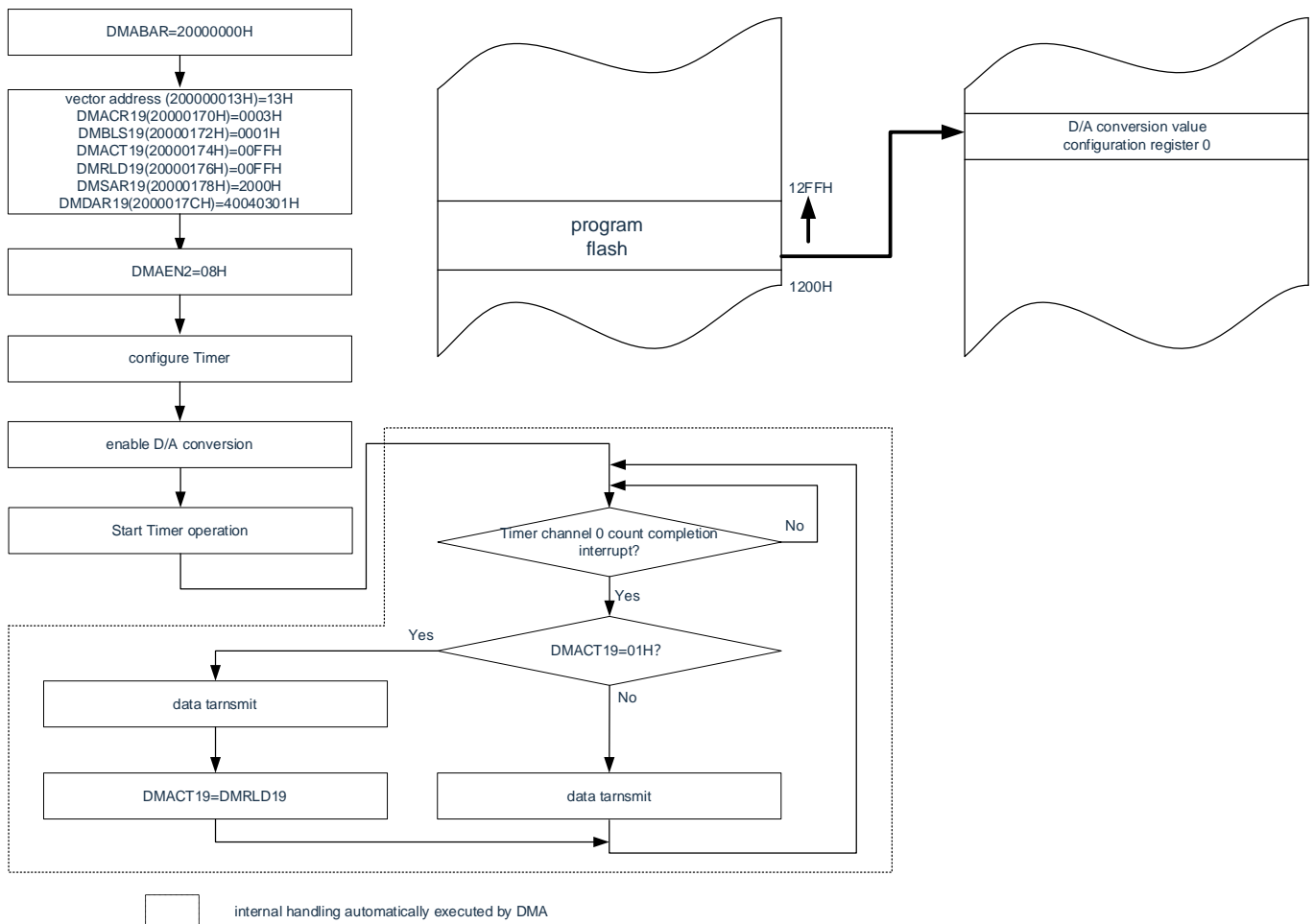
(2) Example 2 of the use of repeat mode: Sine wave output using an 8-bit D/A converter

Using the Channel 0 interval timer function using Timer4 and initiating the DMA by interrupt, transfer the sine wave meter saved in the data flash to the 8-bit D/A conversion value to set register 0 (40044734H) .

Set the interval time of the timer to the output preparation time of D/A.

- Vector addresses are assigned at 20000013H, and control data is assigned at 20000170H~2000017FH.
- Transfer 255 bytes of data flash from 1200H to 12FEH to the D/A conversion value setting register 0 (40044734H) .
- Disable repeat mode interrupts.

Figure 24-22 repeat mode example 2: Sine wave output using an 8-bit D/A converter



To stop the output, bit3 of DMAEN2 must be cleared after stopping the operation of the timer.

24.4.4 Chain transfer

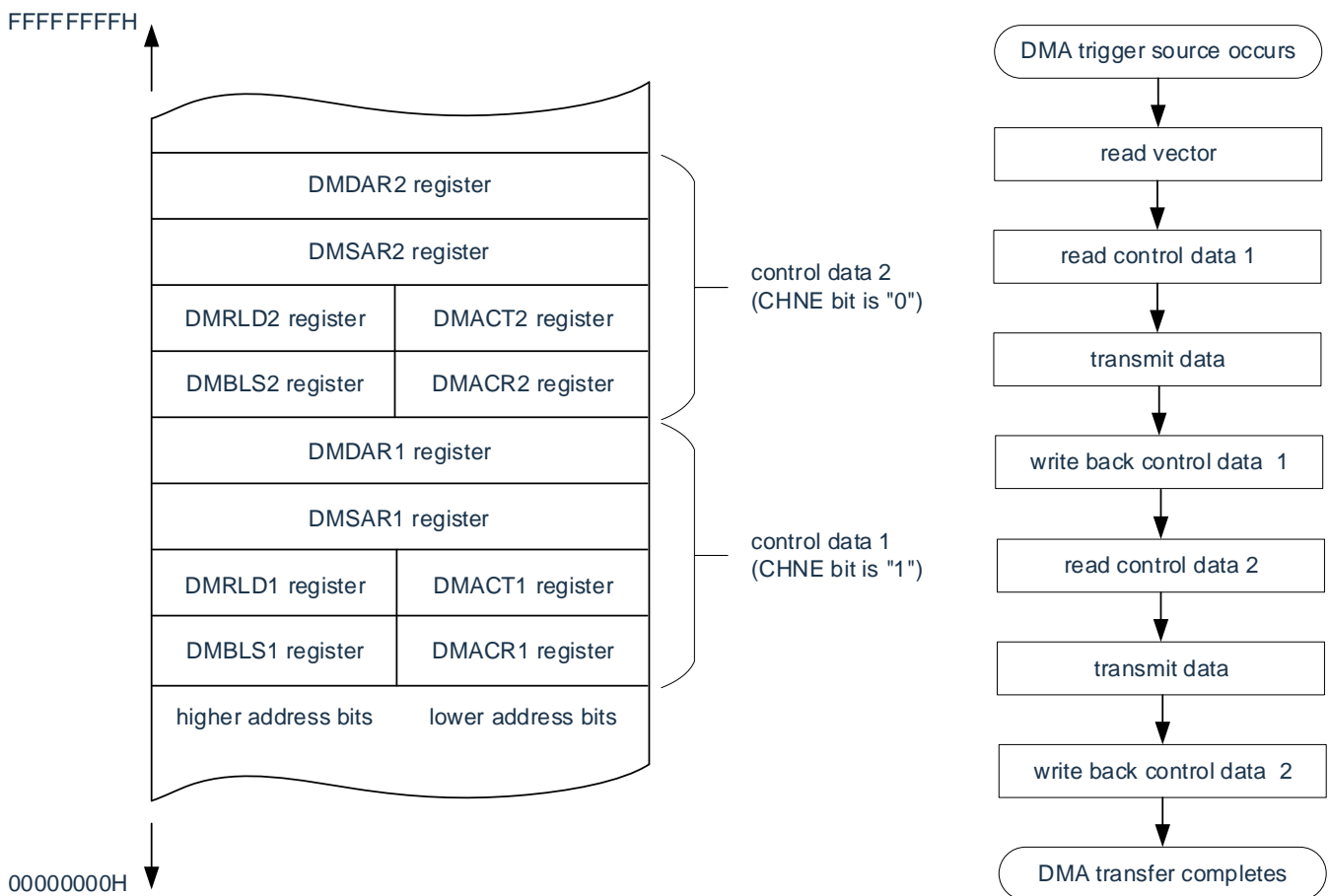
When the DMACRj (j=0~39) register has a CHNE bit of "1" (allow chain transfer), multiple data can be transferred continuously through one start source.

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 24-23.

Figure 24-23 chain transmission



Note 1 The DMACR39 register must be placed at chne position "0" (chain transfer is prohibited).

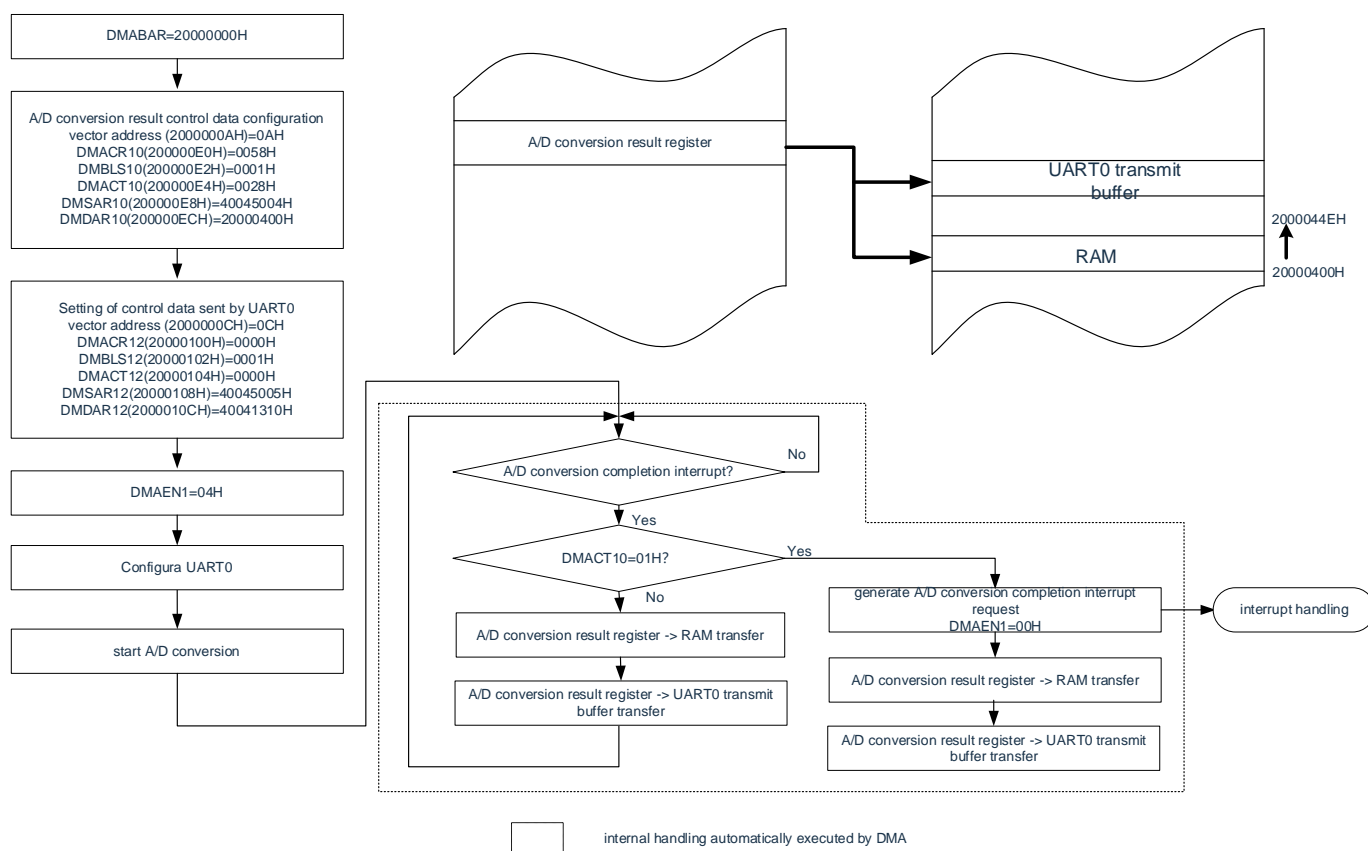
2. DMAENi0~DMAENi7 of the DMAENi (i=0~4) register after the second data transmission of the chain transmission The bit does not change to "0" (disables DMA startup) and does not generate interrupt requests.

(1) An example of the use of chain transfer: Continuously take the A/D conversion result for UART0 transmission

Start the DMA via the A/D conversion end interrupt, and transmit the A/D conversion result to RAM for UART0 transmission.

- Vector addresses are 2000000AH and 2000000CH, respectively.
- The control data of A/D conversion results is distributed at 200000E0H~200000EFH.
- The control data transmitted by UART0 is distributed at 20000100H~2000010FH.
- Transfer 2 bytes of data from the A/D conversion result registers (40045004H, 40045005H) to 20000400H~ of RAM 2000044FH, and the high bit 1 byte (40045005H) of the A/D conversion result register is transmitted to the UART Transmit buffer of 0 (40041310H).

Figure 24-24 example of chain transfer: Continuous A/D conversion results are used for UART0 transmission



24.5 Considerations when using DMA

24.5.1 The DMA controls the settings of the data and vector tables

- The DMA Base Address Register (DMABAR) must be changed with all DMA boot sources set to disable startup.
- The DMA Base Address Register (DMABAR) can only be rewritten once.
 - Must be "0" (prohibited) at the DMAENi0~DMAENi7 bits of the corresponding DMAENi (i=0~4) register DMA Startup) when changing DMCRj, DMBLSj, DMACTj, DMRLDj, DMSARj, the data of the DMDARj register.
 - Must be "0" (prohibited) at the DMAENi0~DMAENi7 bits of the corresponding DMAENi (i=0~4) register DMA Startup) when you change the start address of the DMA control data area that is set in the vector table.

24.5.2 The DMA controls the allocation of data areas and DMA vector table areas

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 occur via the RAM parity error detection feature, the DMRLD register must be initialized (0) even when using normal mode 000H)。

24.5.3 The number of execution clocks for the DMA

The execution of the DMA at startup and the number of clocks required are shown in Table 24-9.

Table 24-9 execution at startup and the number of clocks required

Read vector	Control data		Read the data	Write data
	read	Write back		
1	4	Note 1	Note 2	Note 2

Note 1 For the number of clocks required to write back control data, refer to "Table 24-10 number of clocks required to write back control data".

2. For the number of clocks required to read and write data, please refer to "Table 24-11 number of clocks required to read and write data".

Table 24-10 number of clocks required to write back control data

Settings of the DMACR				Address		Controls the writeback of registers				The 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	Do not write	Do not write	1
0	1	X	0	Increase	fixed	Write back	Write back	Write back	Do not write	2
1	0	X	0	fixed	Increase	Write back	Write back	Do not write	Write back	2
1	1	X	0	Increase	Increase	Write back	Write back	Write back	Write back	3
0	X	1	1	Duplicate areas	fixed	Write back	Write back	Write back	Do not write	2
1	X	1	1		Increase	Write back	Write back	Write back	Write back	3
X	0	0	1	fixed	Duplicate areas	Write back	Write back	Do not write	Write back	2
X	1	0	1	Increase		Write back	Write back	Write back	Write back	3

Note j=0~39, X: "0" or "1"

Table 24-11 number of clocks required to read and write data

Execution status	RAM	Code flash	Data flash	Special function registers (SFR)	Extended Special Function Register (2ndSFR).	
					No waiting	await
Read the data	1	2	4	1	1	1+ Wait for the number of bets
Write data	1	—	—	1	1	1+ Wait for the number of bets

24.5.4 Response time of the DMA

The DMA response time is shown in Table 24-12. 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 24-12 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
- Maximum response time for access to timerA registers in which wait occurs: Maximum response time + 1 clock for each condition

Note 1 clock: $1/f_{CLK}$ (f_{CLK} : CPU/peripheral hardware clock).

24.5.5 The startup source for the 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 the "23.3.3 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 ^{note} (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 produce an interrupt request through the comparator's one-edge detection (CnEDG=0) and an interrupt request is generated through the falling edge of the comparator (CnEPO=1) and $IVCMP < IVREF$ (Or internal reference voltage.) 1.45V)。
- (n=0, 1)

24.5.6 Running in standby mode

state	DMA runs
Sleep mode	Capable of operation (disables operation in low-power RTC mode).
Deep sleep mode	Can accept the DMA startup source and perform DMA transfer ^{note 1}

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 25 Linkage Controller (EVENTC).

25.1 Features 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.

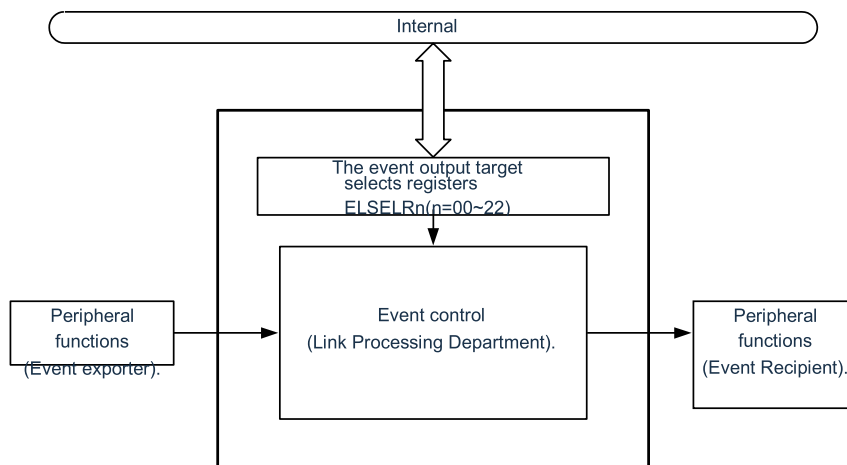
EVENTC has the following features:

- Depending on the product, event signals for 23 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 10 peripheral functions.

25.2 Structure of EVENTC

The block diagram of eventc is shown in Figure 25-1.

Figure 25-1 diagram of EVENTC



25.3 Control registers

The controller registers are shown in Table 25-1.

Table 25-1 control registers of the 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
Event output target selection register 16	ELSELR16
Event output target selection register 17	ELSELR17
Event output target selection register 18	ELSELR18
Event output target selection register 19	ELSELR19
Event output target selection register 20	ELSELR20
Event output target selection register 21	ELSELR21
Event output target selection register 22	ELSELR22

25.3.1 Output target selection register n(ELSELRn) (n=00~22).

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 25-22, ANDLRn register (n= The setting value of 00~2) and the corresponding operation of the link target peripheral function when it accepts an event are shown in Table 25-3.

Figure 25-2 Format of the event output target selection register n (ELSELRn).

address:40043400H(ELSELR00)~40043416H(ELSELR22) after reset :						00H	R/W	
symbol	7	6	5	4	3	2	1	0
ELSELRn	0	0	0	0	ELSELRn3	ELSELRn2	ELSELRn1	ELSELRn0

ELSELn3	ELSELn2	ELSELn1	ELSELn0	Selection of event links
0	0	0	0	Disable event links.
0	0	0	1	Select run ^{Note 1} for peripheral function ¹ that is
0	0	1	0	Select the linked peripheral function 2 for Run ^{Note 1}
0	0	1	1	Select the linked peripheral function 3 for Run ^{Note 1}
0	1	0	0	Select the linked peripheral function 4 for Run ^{Note 1}
0	1	0	1	Select the linked peripheral function 5 for Run ^{Note 1}
0	1	1	0	Select the linked peripheral function 6 for Run ^{Note 1}
0	1	1	1	Select the linked peripheral function 7 for Run ^{Note 1}
1	0	0	0	Select the linked peripheral function 8 for Run ^{Note 1}
1	0	0	1	Select the linked peripheral function 9 for Run ^{Note 1}
Other than the above				Prohibit settings.

Note 1 Refer to "Table 25-3 ELSELRn register (n=00~22) correspond to the operation of the link target peripheral function when it accepts an event".

Table 25-2 ELSELRn registers (n=00~22) correspond to peripheral functions

Register name	Event occurrence source (output source for event input n).	Contents
ELSELR00	External interrupt edge detection 0	INTP0
ELSELR01	External interrupt edge detection1	INTP1
ELSELR02	External interrupt edge detection2	INTP2
ELSELR03	External interrupt edge detection3	INTP3
ELSELR04	External interrupt edge detection4	INTP4
ELSELR05	External interrupt edge detection5	INTP5
ELSELR06	Key returns signal detection	INTKR
ELSELR07	RTC fixed cycle/alarm clock consistent detection	INTRTC
ELSELR08	The timer M input captures A0/compares match A 0	INTTMM0
ELSELR09	The timer M input captures B0/compares match B 0	INTTMM0
ELSELR10	The timer M input captures A1/compares match A1	INTTMM1
ELSELR11	The timer M input captures B1/compares match B1	INTTMM1
ELSELR12	Timer M underflow	TMM underflow signal
ELSELR13	Timer A underflow/end of pulse width measurement period/end of pulse period measurement	INTTMA
ELSELR14	The timer B input captures A/compares match A	INTTMB
ELSELR15	Timer B input captures B/compares match B	INTTMB
ELSELR16	Timer4 channel 00 count end/snap end	INTTM00
ELSELR17	Timer4 channel 01 of the count end/capture end	INTTM01
ELSELR18	Timer4 channel 02 of count end/end of capture	INTTM02
ELSELR19	Timer4 channel 03 of count end/capture end	INTTM03
ELSELR20	The comparator detects 0	INTCMP0
ELSELR21	Comparator detection 1	INTCMP1
ELSELR22	The stop vibration detection is interrupted	INTOSDC

Table 25-3 ELSELRn register (n=00~2 2) correspond to the operation of the link target peripheral function when it accepts an event

ELSELRn register ELSELRn3~ELSELRn0 bits	Link target No.	Link target peripheral functionality	Run when the event is accepted
0001B	1	A/D converter	Start A/D conversion.
0010B	2	Timer input for Timer4 channel 0 ^{Note 1}	Delay counter, measurement of input pulse interval, external event counter
0011B	3	Timer Input ^{Note 2} for the timer4 channel 1	Delay counter, measurement of input pulse interval, external event counter
0100B	4	Timer A	Count the sources
0101B	5	Timer B	Input capture for TBIO1
0110B	6	Timer M	TMIOD0 input capture, forced cutoff of pulse output
0111B	7	Timer M	TMIOD1 input capture, forced cutoff of pulse output
1000B	8	DA0 ^{Note 3}	Real-time output
1001B	9	DA1 ^{Note 3}	Real-time output

Note 1 To select the timer input of Timer4 channel 0 as the link target peripheral function, the operating clock of channel 0 must first be set to the operating clock of channel 0 through the timer clock selection register 0 (TPS0). fCLK, enable register 1 (NFEN1) to set the noise filter at the TI00 pin as OFF (TNFEN00=0), and the timer input used by channel 0 is set to the event input signal of the linkage controller via timer input select register 0 (TIS0).

2. To select the timer input of Timer4 channel 1 as the link target peripheral function, the operating clock of channel 1 must first be set to the operating clock of channel 1 through the timer clock selection register 0 (TPS0). fCLK, enable register 1 (NFEN1) to set the noise filter at the TI01 pin as OFF (TNFEN01=0), and the timer input used by Channel 1 is set to EVENTC via timer input select register 0 (TIS0). event input signal.

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 the deep sleep mode.

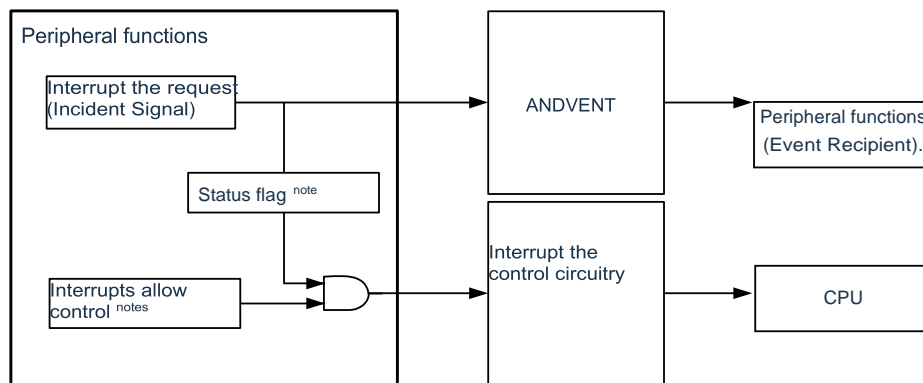
25.4 Operation of EVENTC

The path used by the event signal generated by each peripheral function as the interrupt request of the interrupt control circuit and the path used as the EVENTC event are independent of each other. Therefore, each event signal is independent of interrupt control and can be used as an event signal for the operation of the peripheral function of the event receiver.

The relationship between interrupt handling and EVENTC is shown in Fig.25-3. This figure uses the relationship between an interrupt request status flag and a peripheral function of the interrupt allow bit (which controls whether it is allowed or prohibited).

Peripheral functions that accept events through EVENTC operate according to the receiver peripheral functions after receiving the event (see Table Table 25-3 ELSELRn register (n=00~2 2) correspond to the operation of the link target peripheral function when it accepts an event").

Fig.25-3 Interrupt handling and EVENTC relationship



Note Some peripheral features do not have this feature.

The response to the peripheral functions that accept the event is shown in Table 25-4Table 25-44.

Table 25-4 Response of the peripheral function of the received event

The event accepts target No	The capabilities of the event link target	Run after the event is accepted	response
1	A/D converter	A/D conversion	EventC events become hardware-triggered directly for A/D conversions.
2	Timer4 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 fCLK cycles from the occurrence of an EVENTC event.
3	Timer4 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 fCLK cycles from the occurrence of an EVENTC event.
4	Timer A	Count the sources	EventC events become the counting source for timer A directly.
5	Timer B	Input capture for TBIOB	It is triggered from the time the EVENTC event has occurred after 2 or 3 fCLK cycles have elapsed.
6	Timer M	Input capture for TMIOD0	It is triggered after the event of an EVENTC event passes through the operating clock cycles of 2 or 3 timers M.
		Forced cutoff of the pulse output	It becomes a forced cut-off state after 2 or 3 timer M operating clock cycles from the occurrence of an EVENTC event.
7	Timer M	Input capture for TMIOD1	It is triggered after the event of an EVENTC event passes through the operating clock cycles of 2 or 3 timers M.
		Forced cutoff of the pulse output	It becomes a forced cut-off state after 2 or 3 timer M operating clock cycles from the occurrence of an EVENTC event.
8	Channel 0 of the D/A converter	Real-time output (channel 0).	The D/A conversion of channel 0 begins after 2 or 3 fCLK cycles from the time of the EVENTC event.
9	Channel 1 of the D/A converter	Real-time output (channel 1).	The D/A transition of channel 1 begins after 2 or 3 fCLK cycles from the time of the EVENTC event.

Chapter 26 Interrupt function

The Cortex-M0+ processor has a built-in nested vector interrupt controller (NVIC) that supports up to 32 interrupt request (IRQ) inputs, as well as an unshielded interrupt (NMI) input, as well as multiple internal exceptions.

The system extends 32 Interrupt Request (IRQ) inputs and 1 Unshielded Interrupt (NMI) input to support up to 96 interrupt sources, as well as one unshielded interrupt source. This user manual only describes the extended functions in this system, the Cortex-M0+ processor has built-in NVIC functions, please refer to the Cortex-M0+ processor user manual.

26.1 The types of interrupt function

There are two types of interrupt functions.

(1) Interrupts can be masked

This is an interrupt that is subject to masking control. If the interrupt mask flag register is not open, the interrupt request will not be responded to, even if it is generated.

It can generate a standby release signal to cancel deep sleep mode and sleep mode.

Maskable interrupts are divided into external interrupt requests and internal interrupt requests.

(2) Interrupts cannot be masked

This is an interrupt that does not accept masking control, and the CPU must respond once an interrupt request is generated.

26.2 Interrupt sources and structures

Suspend source column table reference Table 26-1.

Table 26-1 List of interrupt sources (1/4).

Interrupt handline	Interrupt source numbering	Interrupt source		Internal/External	Basic structure Type Note 1
		name	trigger		
Maskable	0	INTLVI	Voltage detection Note 2	interior	(A)
	1	INTP0	Detection of pin input edges	exterior	(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	INTST2/INT SSPI20/INTI C20	Transmission end of UART2 transmission or transmission end of buffer null interrupt/SSPI20 or transfer end of buffer null interrupt/IIC20	interior	(A)
	8	INTSR2/INT SSPI21/INTI IC21	Transmit end received by UART2 / Transmit end of SSPI21 or Buffer Null Interrupt / Transmit End of IIC21		
	9	INTSRE2	A communication error received by UART2 has occurred		
	10	INTST0/INT SSPI00/INTI C00	The end of the transmission sent by the UART0 or the transfer end of the buffer null interrupt/SSPI00 or the transfer end of the buffer null interrupt/IIC00		
	11	INTSR0/INT SSPI01/INTI IC01	Transmit end received by UART0 / Transmit end of SSPI01 or Transmit end of buffer null interrupt / IIC01		
	12	INTSRE0	A communication error received by UART0 has occurred		
	13	INTST1/INT SSPI10/INTI IC10	Transmission end for UART1 transmission or transfer end for buffer null interrupt/SSPI10 or buffer null interrupt/IIC10 transmission end		
	14	INTSR1/INT SSPI11/INTI C11	Transmit end received by UART1 / Transmit end of SSPI11 or Buffer Null Interrupt / Transmit End of IIC11		
	15	INTSRE1	A communication error received by UART1 has occurred		
	16	INTIICA0	IICA0 communication ended		
	17	INTTM00	The end of the count or the end of the snap for timer channel 00		
	18	INTTM01	The count of timer channel 01 is over or the capture is over		
	19	INTTM02	The count end of timer channel 02 or the end of the snap		
	20	INTTM03	The count end or capture end of timer channel 03		

Note 1 The basic composition types (A) to (D) correspond to (A) Figure 26-1D) of FIG. 261, respectively.

2. This is the case where the bit7 (LVIMD) of the voltage sense level register (LVIS) is placed at "0".

Table 25-1 List of interrupt sources (2/4).

Interrupt handling	Interrupt source	Interrupt source		Internal/External	Basic structure Type Note 1
		name	trigger		
Maskable	21	INTAD	End of A/D conversion	interior	(A)
	22	INTRTC	Fixed period /for real-time clocks Alarm clock consistent detection		
	23	INTKR	The key returns the detection of the signal	exterior	(C)
	24	INTCMP0	The comparator detects 0	interior	(A)
	25	INTCMP1	Comparator detection 1		
	26	INTTMA	Underflow of timer A		
	27	INTTMM0	Input capture, comparison matching, overflow, and underflow interrupts for timer M0		
	28	INTTMM1	Input capture, comparison matching, overflow, and underflow interrupts for timer M1		
	29	INTTMB	Timer B's input capture, comparison matching, overflow, and underflow interrupts		
	30	INTTMC	Overflow of timer C		
	31	INTFL	Flash programming is over		
	32	INTOSDC	The stop vibration detection is interrupted		
	33	INTP6	Detection of pin input edges	exterior	(B)
	34	INTP7	Detection of pin input edges		
	35	INTP8	Detection of pin input edges		
	36	INTP9	Detection of pin input edges		
	37	INTP10	Detection of pin input edges		
	38	INTP11	Detection of pin input edges		
	39	INTST3/INTSSPI30/INTIIC30	The end of the transmission sent by the UART 3 or the end of the transmission of the buffer null interrupt/SSPI30 or the transmit end of the buffer null interrupt/IIC30	interior	(A)
	40	INTSR3/INTSSPI31/INTIIC31	Transmit end of UART3 received / Transmit end of SSPI31 or Buffer Null Interrupt / Transmit End of IIC31		
	41	INTC0ERR	CAN0 error interrupt		
	42	INTSPI0	High-speed SPICHS0 transmission ends interrupt Note 2		
	43	reserved	-	-	-
	44	INTTM01H	The end of the count or the end of the capture of timer channel 01 (when the high 8-bit timer is operating).	interior	(A)
	45	reserved	-	-	-
	46	reserved	-	-	-
	47	INTTM03H	The count end or capture end of timer channel 03	interior	(A)

			(When operating with a high 8-bit timer).		
	48	INTDIV	The divider calculation ends		

Note 1 The basic composition types (A) to (D) correspond to (A) Figure 26-1D) of FIG. 261, respectively.

2. The high-speed SPICHS0 module is proprietary to the BAT32A279 product, so the interrupt source 42 is reserved when the BAT32A239 product is used.

Table 2 5-1 List of interrupt sources (3/4).

Interrupt handling	Interrupt source	Interrupt source		Internal/External	Basic structure Type Note
		name	trigger		
Maskable	49	INTTM10	The end of the count or the end of the snap for timer channel 10	interior	(A)
	50	INTTM11	The count of timer channel 11 is over, or snap ends		
	51	INTTM12	Timer channel 12 count ends or snap ends		
	52	INTTM13	Timer channel 13 count ends or snap ends		
	53	reserved	-	-	-
	54	INTIT	Detection of interval signals	interior	(A)
	55	INTC0REC	CAN0 receive end		
	56	INTC0WUP	CAN0 wake-up		
	57	INTC0TRX	CAN0 send ended		
	58	INTC1ERR	CAN1 error interrupt	interior	(A)
	59	reserved	-		
	60	reserved	-		
	61	INTC1REC	CAN1 receive ended		
	62	INTC1WUP	CAN1 wake-up		
	63	INTC1TRX	CAN1 send ends		
	64	INTOCR	High-speed internal oscillation self-adjustment ends		
	65	reserved	-		
	66	reserved	-		
	67	reserved	-		
	68	reserved	-		
	69	reserved	-		
	70	reserved	-		
	71	reserved	-		
	72	reserved	-		
	73	reserved	-		
	74	INTSPI1	High-speed SPICHS1 transmission ends interrupt ^{Note 2}	interior	(A)
	75	reserved	-		
	76	INTLCDB	The LCD bus transfer ends with an interrupt	interior	(A)
	77	reserved	-		
	78	reserved	-		
	79	reserved	-		
	80	INTIICA1	IICA1 communication ended	interior	(A)
	81	INTTM14	The timer channel 14 is counted or snapped		
	82	INTTM15	The end of the count of the timer channel 15 or the end of the capture		
	83	INTTM16	Timer channel 16 count ends or capture ends		
	84	INTTM17	The count of timer channel 17 is over or the snap ends	interior	(A)
	85	reserved	-		
	86	reserved	-		

Note 1 The basic composition types (A) to (D) correspond to (A) Figure 26-1D) of FIG. 261, respectively.

2. The high-speed SPICHS1 module and the LCDB module are proprietary to the BAT32A279 product, so the interrupt source 74, 76 is reserved when the BAT32A239 product is used.

Table 25-1 List of interrupt sources (4/4).

Interrupt handling	The interrupt	Interrupt source		Internal/External	Basic structure
		name	trigger		
Maskable	87	reserved	-		
	88	reserved	-		
	89	reserved	-		
	90	INTC2ERR	CAN2 error interrupt ^{note 3}	interior	(A)
	91	reserved	-		
	92	reserved	-		
	93	INTC2REC	CAN2 receives the end ^{note 3}	interior	(A)
	94	INTC2WUP	CAN2 wake-up ^{note 3}		
	95	INTC2TRX	CAN2 sends the end ^{note 3}		
Not Maskable	—	INTWDT	Watchdog timer interval interrupt ^{note 2}	interior	(D)

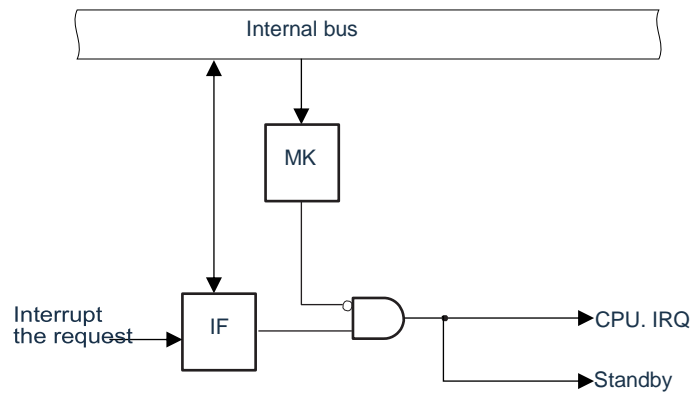
Note 1 The basic composition types (A) to (D) correspond to (A) Figure 26-1D) of FIG. 261, respectively.

2. This is the case where the bit7 (WDTINT) of the option byte (000C0H) is set to "1".

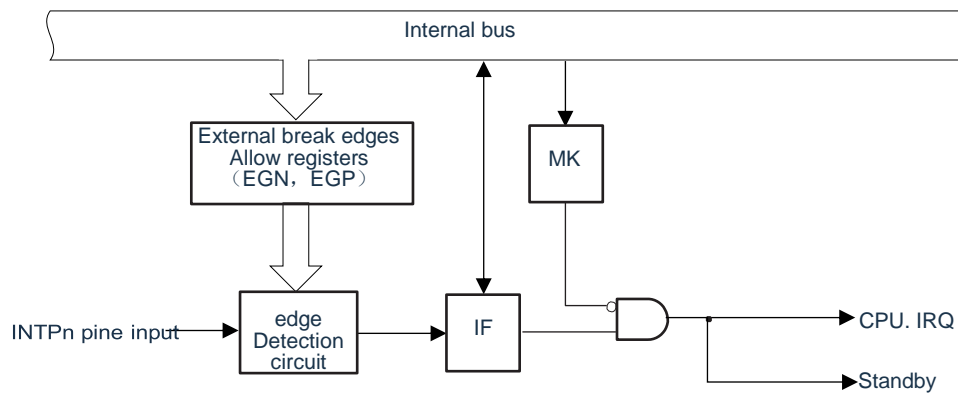
3. The CAN2 module is proprietary to the BAT32A279 product, so the interrupt sources 90, 93, 94, 95 are reserved when BAT32A239 products are used.

Figure 26-1 basic structure of the interrupt function

(A) Internally maskable interrupts

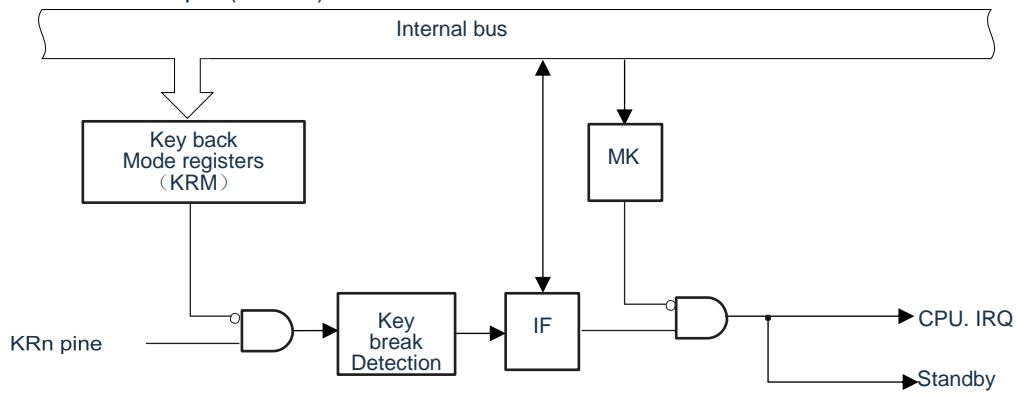


(B) Externally maskable interrupts (INTPn)



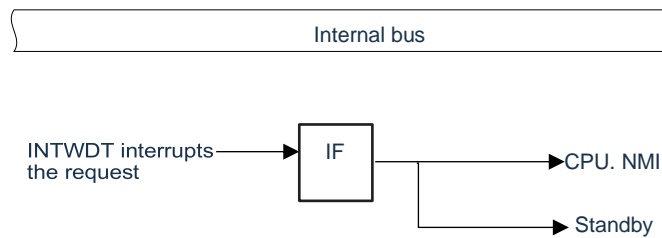
note $0 \leq n \leq 11$

(C)Externally maskable interrupts (INTKR)



note $0 \leq n \leq 7$

(D) Interrupts cannot be masked



Note: Unmasked interrupt request flag IF does not have a physical register and cannot generate an interrupt request by reading or writing registers on the bus.

26.3 Registers that control 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 rise edge enable registers (EGP0, EGP1).
- External interrupt drop edge enable registers (EGN0, EGN1).

26.3.1 Interrupt request flag register (IF00~IF31).

By occurring with a corresponding interrupt request or executing instruction, the interrupt request flag is set to "1". By generating a reset signal or executing an instruction, the interrupt request flag is cleared with "0".

Set the IF00L~IF31L, IF00H~IF31L, IF00H~IF31H, IF00T~IF31T registers via the 8-bit memory operation instructions or set the IF00~IF31 registers through the 32-bit memory operation instructions.

After generating a reset signal, the values of these registers become "0000_0000H".

Figure 26-2 Format of the 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_0000HR/W

	31	30	29	28	27	26	25	24
	Reserved							
	23	22	21	20	19	18	17	16
IFmT	Reserved							IFT
	15	14	13	12	11	10	9	8
IFmH	Reserved							IFH
	7	6	5	4	3	2	1	0
IFmL	Reserved							IFL

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.

IFmH	Interrupt request flag for interrupt sources numbered 32 to 63
0	No interrupt request signal is generated.
1	Generates an interrupt request and is in the interrupt request state.

IFmT	Interrupt request flag for interrupt sources numbered 64 to 95
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 the interrupt request flag register is shown Table 26-2.

2. Interrupt request flag register with CPU. The correspondence of the IRQ is Figure 26-4.

3. The interrupt request flag register does not self-clear, and the register must be written to 0 after the interrupt response.

26.3.2 Interrupt Mask Flag Register (MK00~MK31)

The interrupt masking flag setting enable or disables the corresponding maskable interrupt processing.

Set the MK00L~MK31L register via the 8-bit memory operation instruction, the MK00H~MK31H register, or the MK00~MK31 register through the 32-bit memory operation instruction.

After a reset signal is generated, the values of these registers become "FFFF_FFFF".

Figure 26-3 Format of the interrupt request masking register (MKm) (m=0~31).

Address: MK00:40006100H, MK01:40006104H, MK02:40006108H, MK03:4000610CH
 MK04:40006110H, MK05:40006114H, MK06:40006118H, MK07:4000611CH
 MK08:40006120H, MK09:40006124H, MK10:40006128H, MK11:4000612CH
 MK12:40006130H, MK13:40006134H, MK14:40006138H, MK15:4000613CH
 MK16:40006140H, MK17:40006144H, MK18:40006148H, MK19:4000614CH
 MK20:40006150H, MK21:40006154H, MK22:40006158H, MK23:4000615CH
 MK24:40006160H, MK25:40006164H, MK26:40006168H, MK27:4000616CH
 MK28:40006170H, MK29:40006174H, MK30:40006178H, MK31:4000617CH
 Reset value: FFFF_FFFFH/W

	31	30	29	28	27	26	25	24
	Reserved							
	23	22	21	20	19	18	17	16
MKmT	Reserved							MKT
	15	14	13	12	11	10	9	8
MKmH	Reserved							MKH
	7	6	5	4	3	2	1	0
MKmL	Reserved							MKL

MKmL	Interrupt handling control for interrupt sources numbered 0 to ³¹ Note 1
0	Interrupt handling is allowed.
1	Disable interrupt processing.

MKmH	Interrupt Handling Control for Interrupt Sources Numbers 32 to 63 ^{Note 2}
0	Interrupt handling is allowed.
1	Disable interrupt processing.

MKmT	Interrupt Handling Control for Interrupt Sources Numbers 64 to 95 ^{Note 2}
0	Interrupt handling is allowed.
1	Disable interrupt processing.

Note: 1. The correspondence between the interrupt source and the interrupt request mask register is shown Table 26-2

2. Interrupt request mask register with CPU. The correspondence of the Figure 26-4 shown in Figure 26-4

Table 26-2 Correspondence between interrupt sources and flag registers (1/2).

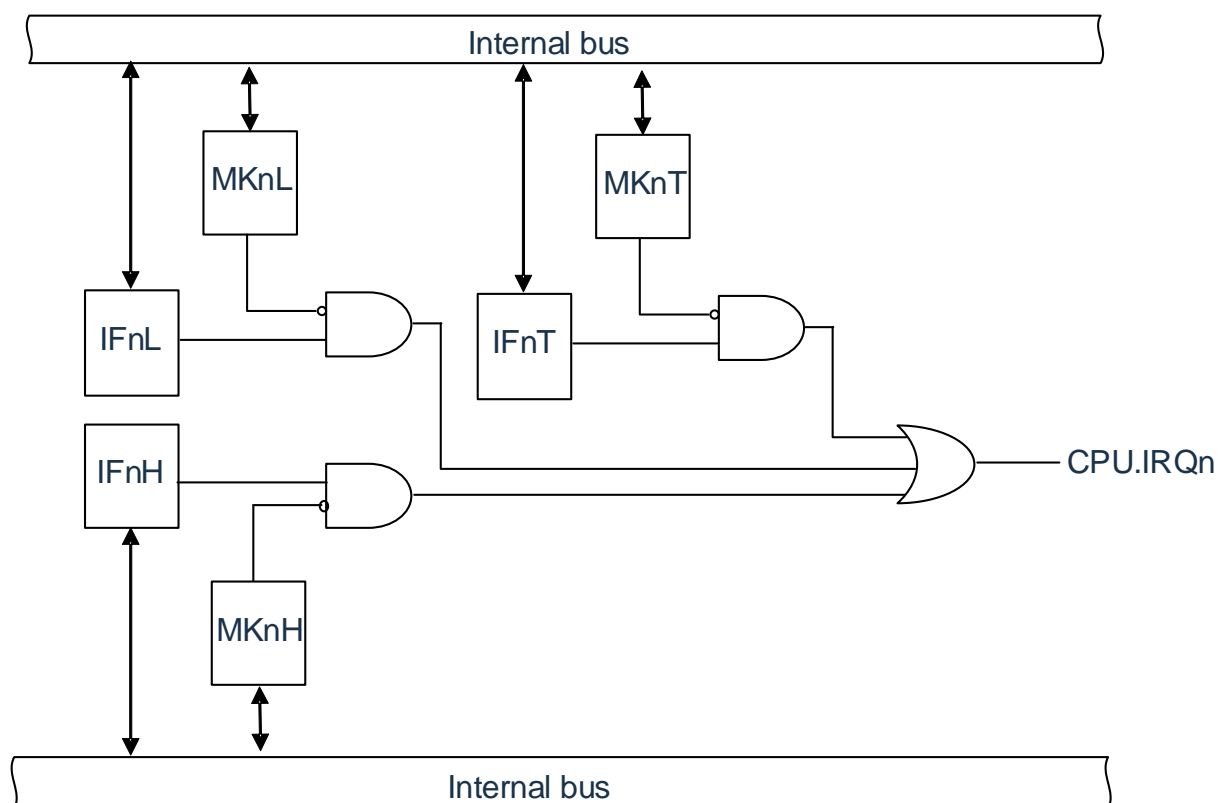
numbering	Interrupt source	Interrupt request flag	Interrupt mask flag register	numbering	Interrupt source	Interrupt request flag	Interrupt mask flag register
0	INTLVI	IF00.IFL	MK00.MKL	32	INTOSDC	IF00.IFH	MK00.MKH
1	INTP0	IF01.IFL	MK01.MKL	33	INTP6	IF01.IFH	MK01.MKH
2	INTP1	IF02.IFL	MK02.MKL	34	INTP7	IF02.IFH	MK02.MKH
3	INTP2	IF03.IFL	MK03.MKL	35	INTP8	IF03.IFH	MK03.MKH
4	INTP3	IF04.IFL	MK04.MKL	36	INTP9	IF04.IFH	MK04.MKH
5	INTP4	IF05.IFL	MK05.MKL	37	INTP10	IF05.IFH	MK05.MKH
6	INTP5	IF06.IFL	MK06.MKL	38	INTP11	IF06.IFH	MK06.MKH
7	INTST2/INTSSPI20/INTIIC20	IF07.IFL	MK07.MKL	39	INTST3/INTSSPI30/INTIIC30	IF07.IFH	MK07.MKH
8	INTSR2/INTSSPI21/INTIIC21	IF08.IFL	MK08.MKL	40	INTSR3/INTSSPI31/INTIIC31	IF08.IFH	MK08.MKH
9	INTSRE2	IF09.IFL	MK09.MKL	41	INTC0ERR	IF09.IFH	MK09.MKH
10	INTST0/INTSSPI00/INTIIC00	IF10.IFL	MK10.MKL	42	INTSPI0	IF10.IFH	MK10.MKH
11	INTSR0/INTSSPI01/INTIIC01	IF11.IFL	MK11.MKL	43	reserved	IF11.IFH	MK11.MKH
12	INTSRE0	IF12.IFL	MK12.MKL	44	INTTM01H	IF12.IFH	MK12.MKH
13	INTST1/INTSSPI10/INTIIC10	IF13.IFL	MK13.MKL	45	reserved	IF13.IFH	MK13.MKH
14	INTSR1/INTSSPI11/INTIIC11	IF14.IFL	MK14.MKL	46	reserved	IF14.IFH	MK14.MKH
15	INTSRE1	IF15.IFL	MK15.MKL	47	INTTM03H	IF15.IFH	MK15.MKH
16	INTIICA0	IF16.IFL	MK16.MKL	48	INTDIV	IF16.IFH	MK16.MKH
17	INTTM00	IF17.IFL	MK17.MKL	49	INTTM10	IF17.IFH	MK17.MKH
18	INTTM01	IF18.IFL	MK18.MKL	50	INTTM11	IF18.IFH	MK18.MKH
19	INTTM02	IF19.IFL	MK19.MKL	51	INTTM12	IF19.IFH	MK19.MKH
20	INTTM03	IF20.IFL	MK20.MKL	52	INTTM13	IF20.IFH	MK20.MKH
21	INTAD	IF21.IFL	MK21.MKL	53	reserved	IF21.IFH	MK21.MKH
22	INTRTC	IF22.IFL	MK22.MKL	54	INTIT	IF22.IFH	MK22.MKH
23	INTKR	IF23.IFL	MK23.MKL	55	INTC0REC	IF23.IFH	MK23.MKH
24	INTCMP0	IF24.IFL	MK24.MKL	56	INTC0WUP	IF24.IFH	MK24.MKH
25	INTCMP1	IF25.IFL	MK25.MKL	57	INTC0TRX	IF25.IFH	MK25.MKH
26	INTTMA	IF26.IFL	MK26.MKL	58	INTC1ERR	IF26.IFH	MK26.MKH
27	INTTMM0	IF27.IFL	MK27.MKL	59	reserved	IF27.IFH	MK27.MKH
28	INTTMM1	IF28.IFL	MK28.MKL	60	reserved	IF28.IFH	MK28.MKH
29	INTTMB	IF29.IFL	MK29.MKL	61	INTC1REC	IF29.IFH	MK29.MKH
30	INTTMC	IF30.IFL	MK30.MKL	62	INTC1WUP	IF30.IFH	MK30.MKH
31	INTFL	IF31.IFL	MK31.MKL	63	INTC1TRX	IF31.IFH	MK31.MKH

Table 26-2 Correspondence between interrupt sources and flag registers (2/2).

numbering	Interrupt source	Interrupt request flag	Interrupt mask flag register
64	INTOCRv	IF00.IFT	MK00.MKT
65	reserved	IF01.IFT	MK01.MKT
66	reserved	IF02.IFT	MK02.MKT
67	reserved	IF03.IFT	MK03.MKT
68	reserved	IF04.IFT	MK04.MKT
69	reserved	IF05.IFT	MK05.MKT
70	reserved	IF06.IFT	MK06.MKT
71	reserved	IF07.IFT	MK07.MKT
72	reserved	IF08.IFT	MK08.MKT
73	reserved	IF09.IFT	MK09.MKT
74	INTSPI1	IF10.IFT	MK10.MKT
75	reserved	IF11.IFT	MK11.MKT
76	reserved	IF12.IFT	MK12.MKT
77	reserved	IF13.IFT	MK13.MKT
78	reserved	IF14.IFT	MK14.MKT
79	reserved	IF15.IFT	MK15.MKT
80	INTIICA1	IF16.IFT	MK16.MKT
81	INTTM14	IF17.IFT	MK17.MKT
82	INTTM15	IF18.IFT	MK18.MKT
83	INTTM16	IF19.IFT	MK19.MKT
84	INTTM17	IF20.IFT	MK20.MKT
85	reserved	IF21.IFT	MK21.MKT
86	reserved	IF22.IFT	MK22.MKT
87	reserved	IF23.IFT	MK23.MKT
88	reserved	IF24.IFT	MK24.MKT
89	reserved	IF25.IFT	MK25.MKT
90	INTC2ERR	IF26.IFT	MK26.MKT
91	reserved	IF27.IFT	MK27.MKT
92	reserved	IF28.IFT	MK28.MKT
93	INTC2REC	IF29.IFT	MK29.MKT
94	INTC2WUP	IF30.IFT	MK30.MKT
95	INTC2TRX	IF31.IFT	MK31.MKT

Note: The SPIHS0, SPIHS1, CAN2 modules are proprietary to the BAT32A279 product, so interrupt sources 42, 74, 90, 93, 94, 95 are reserved when BAT32A239 products are used.

Figure 26-4 Flag registers and CPU. IRQ relationship



26.3.3 Enable registers for the external interrupt rising edge (EGP0, EGP1) and enable registers for the external interrupt falling edge (EGN0, EGN1).

These registers set the effective edge of INTP0~INTP11.

The EGP0, EGP1, EGN0, EGN1 registers are set by 8-bit memory operation instructions. After generating a reset signal, the value of these registers changes to "00H".

Figure 26-5 the enable registers for the rising edge of the external interrupt (EGP0, EGP1) and the enable registers for the falling edge of the external interrupt (EGN0, EGN1).

Address: 40045B38H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
EGP0	EGP7	EGP6	EGP5	EGP4	EGP3	EGP2	EGP1	EGP0

Address: 40045B39H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
EGN0	EGN7	EGN6	EGN5	EGN4	EGN3	EGN2	EGN1	EGN0

Address: 40045B3AH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
EGP1	0	0	0	0	EGP11	EGP10	EGP9	EGP8

Address: 40045B3BH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
EGN1	0	0	0	0	EGN11	EGN10	EGN9	EGN8

EGPn	EGNn	Effective edge selection for the INTPn pin (n=0~11).
0	0	Disables edge detection.
0	1	Drop the edge
1	0	Rising edge
1	1	Rising and falling edges

The ports corresponding to the EGPn bit and the EGNn bit are shown in Table Table 26-33.

Table 26-3 corresponds to the interrupt request signals for EGPn bits and EGNn bits

Detect the enable bits		Interrupt request signal	100 pins	80 pins	64 pins	48 pins
EGP0	EGN0	INTP0	○	○	○	○
EGP1	EGN1	INTP1	○	○	○	○
EGP2	EGN2	INTP2	○	○	○	○
EGP3	EGN3	INTP3	○	○	○	○
EGP4	EGN4	INTP4	○	○	○	○
EGP5	EGN5	INTP5	○	○	○	○
EGP6	EGN6	INTP6	○	○	○	○
EGP7	EGN7	INTP7	○	○	○	—
EGP8	EGN8	INTP8	○	○	○	○
EGP9	EGN9	INTP9	○	○	○	○
EGP10	EGN10	INTP10	○	○	○	○
EGP11	EGN11	INTP11	○	○	○	○

Notelf 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).

Note 1 For ports for edge detection, refer to "2.1 Port Capabilities".

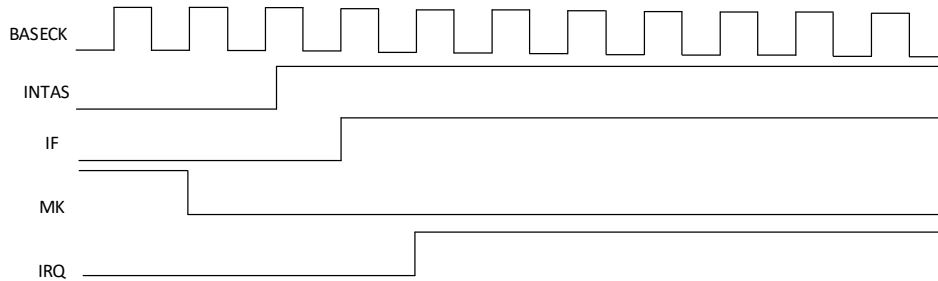
2.n=0~11

26.4 Interrupt the operation of processing

26.4.1 Acceptance of maskable interrupt requests

If the interrupt request flag is placed "1" and the masking (MK) flag of the interrupt request has been cleared "0", it enters a state that can accept the maskable interrupt request, and the interrupt request can be passed to the NVIC.

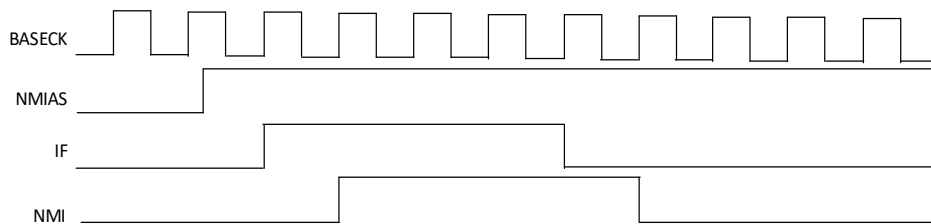
From the interrupt request flag set to 1 to the CPU's IRQ set to 1, only 1 clock is required.



26.4.2 Acceptance of unmaskable interrupt requests

If an unmaskable interrupt request is generated, the interrupt request flag will be placed "1" and passed directly to the NVIC.

From the interrupt request flag set to 1 to the CPU's NMI set to 1, only 1 clock is required.



Chapter 27 Key interrupt function

The number of channels for key interrupt input varies by product.

27.1 The function of the key interrupt

A key interrupt (INTKR) can be generated by giving the key interrupt input pin (KR0 to KR7) the input falling edge.

Table 27-1 key interrupt detection pin assignments

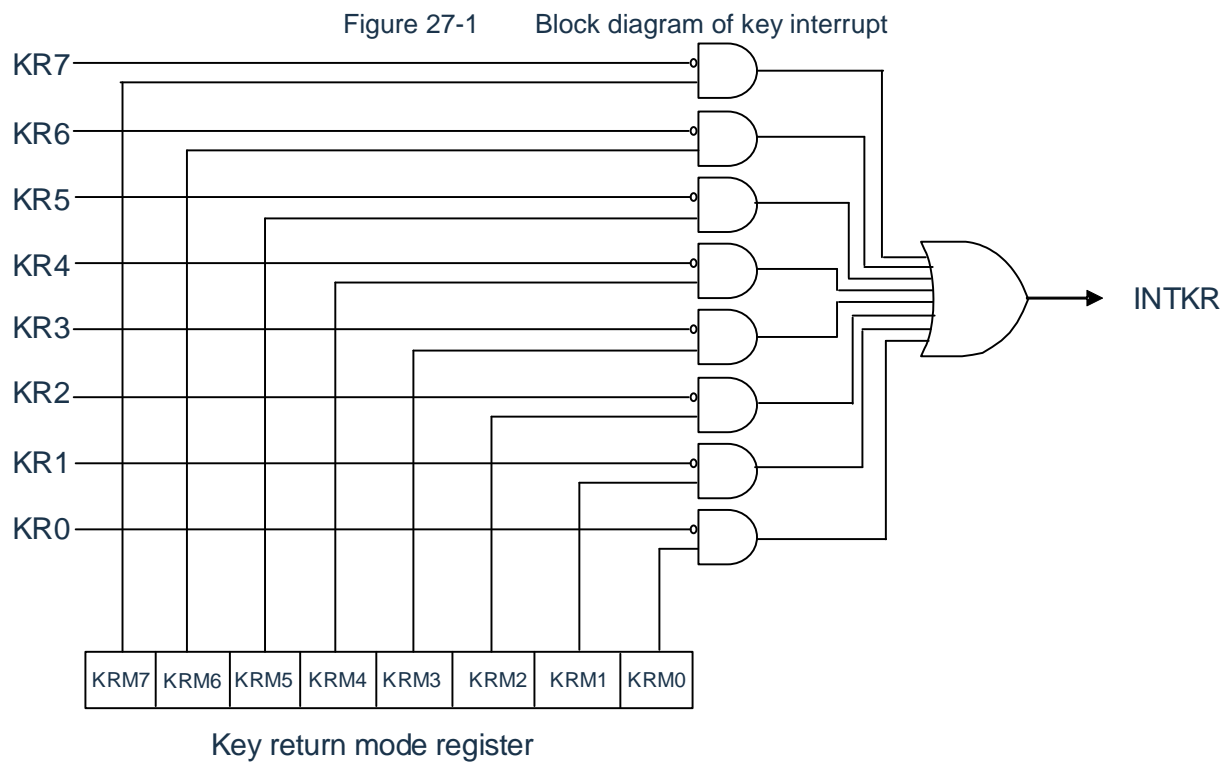
Key interrupt pin	Key return mode register (KRM).
KR0	KRM0
KR1	KRM1
KR2	KRM2
KR3	KRM3
KR4	KRM4
KR5	KRM5
KR6	KRM6
KR7	KRM7

27.2 The structure of the key interrupt

Key interrupts are made up of the following hardware.

Table 27-2 Structure of key interrupts

item	Control registers
Control registers	Key return mode register (KRM) Port mode register (PMx). Port mode control register (PMCx).



27.3 Control Registers of key interrupts

Interrupt function via the following registers control keys.

- Key Return Mode Register (KRM).
- Port Mode Register (PMx).

27.3.1 Key return mode register (KRM).

KRM0~KRM7 bit controls KR0~KR7 signal.

The KRM register is set by the 8-bit memory operation instruction.

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

Figure 27-2 format of the key return mode register (KRM).

Address: 40044B37H after reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
KRM	KRM7	KRM6	KRM5	KRM4	KRM3	KRM2	KRM1	KRM0

KRMn	Key interrupt mode control
0	Key interrupt signal is not detected.
1	Detects a key interrupt signal.

Note:

1. An internal pull-up resistor can be used by using the object position "1" of the pull-up resistor register (PUx) of the input pin by interrupting the key.
2. If the object position 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 then 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.

Note 1.n= 0~7

27.3.2 Port mode register (PMx).

When used as key interrupt input pins (KR0~KR7), the PMCx x bits must be set to "0" and PMxx respectively. The bits are placed "1" respectively. In this case, the output latch of Pxx can be "0" or "1".

The PM x register is set via the 8-bit memory operation instruction.

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

An internal pull-up resistor can be used in bits by a pull-up resistor select register (PU x).

For the format of the port mode register, see "2.3.1 Port Mode Register (PMxx)".

Chapter 28 Standby function

28.1 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.

Note 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 secondary system 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 run.
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 35 Options Bytes".

28.2 Sleep mode

28.2.1 The setting of the sleep mode

When the SLEEPDEEP bit of the SCR register is 0, execute the WFI instruction and enter sleep mode. In sleep mode, the CPU stops operating, but the values of the internal registers are still maintained, and the peripheral modules remain in the state they were in before they entered sleep mode. The status of peripheral modules, vibrators, etc. in sleep mode is shown in Table 28-1.

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.

Note When the interrupt mask flag is "0" (allow 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 move to sleep mode.

Table 28-1 Operating status in sleep mode (1/2).

The setting of the sleep mode project		A case in which wFI instructions are executed while the CPU is running on the main system clock		
		The CPU clocks at high speed internal oscillator (f _{IH}) runs	The CPU runs on an X1 clock (f _X).	The CPU takes the external master system clock (f _{EX}) run
System clock		Stop providing clocks to the CPU.		
The master system clock	f _{IH}	Continues running (cannot be stopped).	Disables operation.	
	f _X	Disables operation.	Continues running (cannot be stopped).	Cannot run.
	f _{EX}		Cannot run.	Continues running (cannot be stopped).
Subsystem clock	f _{XT}	Remain in the state before sleep mode.		
	f _{EXS}			
Low-speed internal oscillation device clock	f _{LI}	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.		
Timer4				
Real-time clock (RTC).				
1 5-bit interval timer				
Watchdog timer		Refer to "Chapter 14 Watchdog Timer".		
Timer A		Can run.		
Timer M				
Timer B				
Timer C				
Clock output/buzzer output				
A/D converter				
D/A converter				
Comparator				
Universal Serial Communication Unit (SCI).				
Serial Interface (IICA).				
aFCAN				
Data Transfer Controller (DMA).				
Linkage controller		Links between runnable function blocks.		
Power-on reset function		Can run.		
Voltage detection function				
External interrupts				
Key interrupt function				
CRC operations	High-speed CRC			

function	Universal CRC	It can be run when DMA is performed in the operation of the RAM area.
RAM parity function		It can be run when the DMA is performed.
SFR protection function		

Note Stop Running: Automatically stops running when you move to sleep mode.

Disable Run: Stops running before moving 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 time

f_{XT} : XT1 Clock

f_{EXS} : External Subsystem

Table 28-1 Operating status in sleep mode (2/2).

The setting of the sleep mode			The case of executing WFI instructions while the CPU is running at the	
			The CPU runs on an XT1 clock (fXT).	The CPU runs on an external subsystem
System clock			Stop providing clocks to the CPU.	
The master system clock	f _{IH}	Disables operation.		
	f _X			
	f _{EX}			
Subsystem clock	f _{XT}	Continues running (cannot be stopped).		Cannot run.
	f _{EXS}	Cannot run.		Continues running (cannot be stopped).
Low-speed internal oscillation device clock	f _{II}	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">• 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			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				
Watchdog timer			Refer to "Chapter 14 Watchdog Timer".	
Timer A			When RTCLPC=0, it can run (otherwise it is prohibited).	
Timer M				
Timer B				
Timer C				
Clock output/buzzer output				
A/D converter			Disables operation.	
D/A converter			When RTCLPC=0, it can run (otherwise it is prohibited).	
Comparator			Can run.	
Universal Serial			When RTCLPC=0, it can run (otherwise it is prohibited).	
Serial Interface (IICA).			Disables operation.	
aFCAN			When RTCLPC=0, it can run (otherwise it is prohibited).	
Data Transfer Controller			When RTCLPC=0, it can run (otherwise it is prohibited).	
Linkage controller			Links between runnable function blocks.	
Power-on reset function			Can run.	
Voltage detection function				
External interrupts				
Key interrupt function				
CRC operations	High-speed	Disables operation.		
	Universal CRC	It can be run when DMA is performed in the operation of the RAM area.		
RAM parity error detection			It can be run when the DMA is performed.	
SFR protection function				

Note Stop Running: Automatically stops running when you move to sleep mode.

Disable Run: Stops running before moving 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 time

f_{XT}: XT1 Clock f_{EXS}: External Subsystem

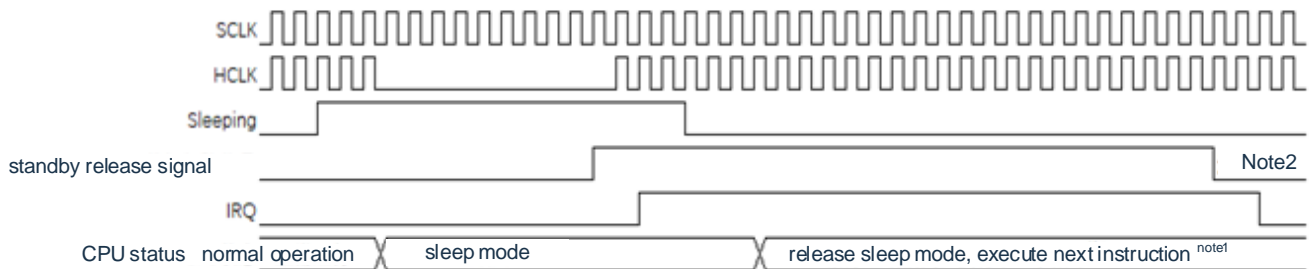
28.2.2 Exit from sleep mode

Sleep mode can be arbitrarily interrupted as well as external reset terminals, POR reset, low voltage detection reset, RAM parity error reset, WDT reset, software reset dismiss.

(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 28-1Dismiss 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.

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.

Note: Before entering sleep mode, only the shield bits corresponding to the interrupts expected to be used to lift 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 28-2Relieves sleep mode by resetting



Note 1: For the reset processing time, please refer to "Chapter 28 Reset Function". For reset processing time for power-on reset (POR) circuits and voltage detection (LVD) circuits, refer to Chapter 29 Power-on Reset Circuits.

28.3 Deep sleep mode

28.3.1 Settings for 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 27-2

Deep sleep mode can only be set if the CPU clock before setting is the main system clock.

Note 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 28-2 Operating status in deep sleep mode

Settings for deep sleep mode		A case in which wFI instructions are executed while the CPU is running on the main system clock	
		The CPU clocks at high speed internal oscillator (fIH) runs	The CPU runs on an X1 clock (fX). The CPU takes the external master system clock (fEX) run
Item			
System clock		Stop providing clocks to the CPU.	
	The master system clock	fIH	Stop it
		fX	
		fEX	
	Subsystem clock	fXT	Stay in the state before deep sleep mode.
		fEXS	
fII		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.	
1 5-bit interval timer			
Watchdog timer		Refer to "Chapter 14 Watchdog Timer".	
Timer A		• 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.	
Timer M		Disables operation.	
Timer B			
Timer C			
Clock output/buzzer output		When the subsystem clock is selected as the counting clock and the RTCLPC bit is "0", it can run (otherwise disables operation).	
A/D converter		Wake-up is possible.	
D/A converter		Can run (remain in the state before deep sleep mode is set).	
Comparator		Can operate (limited to cases where no digital filter is used).	
Universal 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	High-speed	Stop running.	

operations function	CRC	
	Universal CRC	
RAM parity function		
SFR protection function		

Note 1 Stop Running: Automatically stops running when you move to deep sleep mode.

Disable Run: Stops running before moving 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

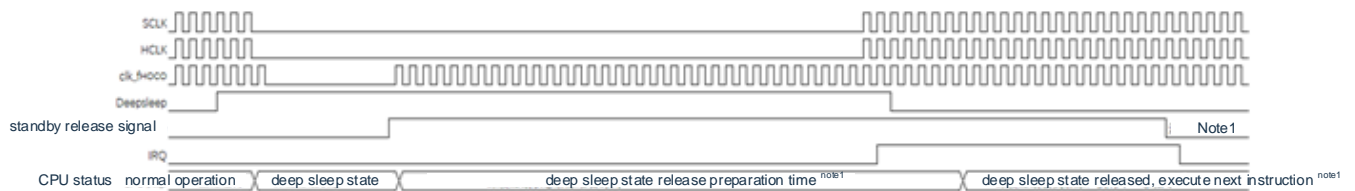
28.3.2 Exit from deep sleep mode

Deep sleep mode can be lifted in the following two ways.

(a) Unblocked by unmasked interrupt requests

If an unshielded interrupt request occurs, deep sleep mode is dismissed. After the oscillation settling time, if it is in a state that allows acceptance of interrupts, vector interrupts are processed. If you are in a state where you are prohibited from accepting interrupts, execute the instruction for the next address.

Figure 28-3 Exit from deep sleep mode by interrupt request



Note 1 Standby release signal: For details of the standby release signal, please refer to "The basic structure of the interrupt function in Figure 2 5-1".

2. Deep sleep state depreparation 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.

3. Wait: From the CPU The IRQ is valid enough to start executing the interrupt service procedure, which requires 14 clocks.

Note: 1. Before entering sleep mode, only the shielding bit corresponding to the interrupt expected to be used to lift sleep mode should be cleared.

2. When the CPU is running on a high-speed system clock (X1 oscillation) and the oscillation settling time after the deep sleep mode is deactivated, the CPU clock must be temporarily switched to a high-speed internal oscillator clock before executing the WFI instruction.

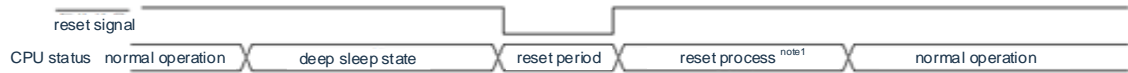
Note The oscillation accuracy of the high-speed internal oscillator clock is stable and waits for change due to temperature conditions and during deep sleep mode.

and waits for change due to

(b) Cancellation by generating a reset signal

Deep sleep mode is lifted by generating a reset signal. Then, as with the usual reset, the procedure is executed after the transfer to the reset vector address.

Figure 28-4 Exit from sleep mode by resetting



Note For reset processing time, please refer to "Chapter 28 Reset Function". For reset processing time for power-on reset (POR) circuits and voltage detection (LVD) circuits, refer to Chapter 29 Power-on Reset Circuits ".

Chapter 29 Reset function

The following 7 methods generate a reset signal.

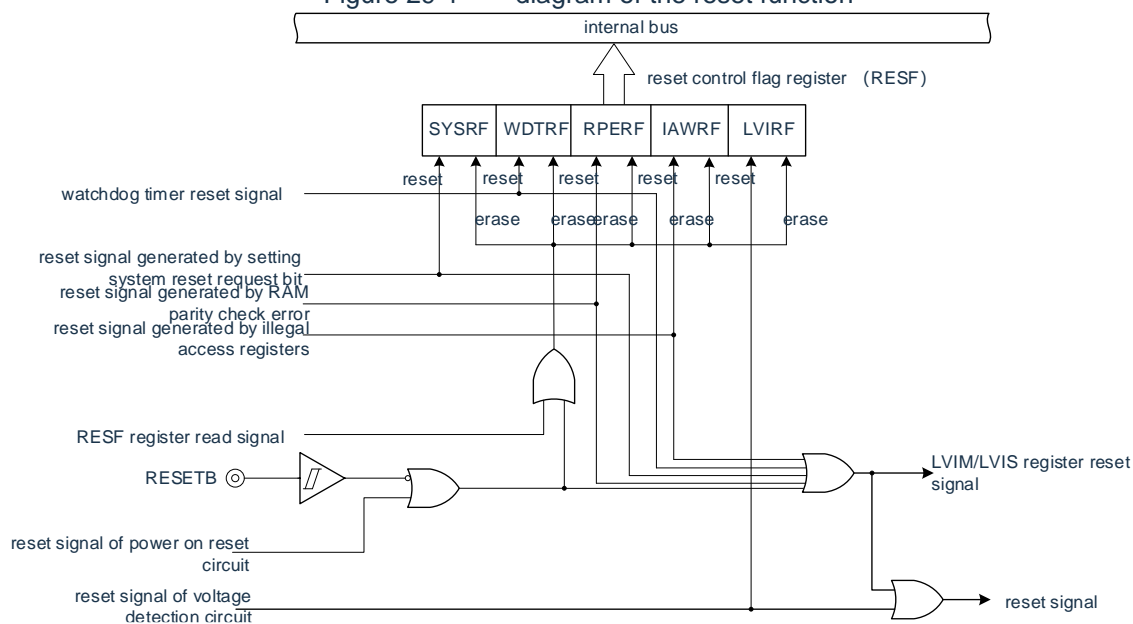
- (1) Input external reset via the RESETB pin.
- (2) An internal reset is generated by the program runaway detection of the watchdog timer.
- (3) An internal reset is generated by comparing the supply voltage and the sense voltage of the power-on reset (POR) circuit.
- (4) An internal reset is generated by comparing the supply voltage and the sense voltage of the voltage detection circuit (LVD).
- (5) Due to system reset requests register bit (AIRC.R. SYSRESETREQ) is set to 1 to produce an internal reset.
- (6) Internal reset due to RAM parity error.
- (7) Internal reset due to access to illegal memory.
- (8) The stop detection function selects a reset mode and detects the stop to produce an internal reset.

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 28-1.

- Note 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.
 3. If a reset occurs, each SFR is initialized so that the port pins become the following:
 - P40: High impedance during external reset or POR reset. High during other resets and after receiving the reset (pull-up resistance inside the connection).
 - P130: Outputs low during reset and after receiving reset.
 - Ports other than P40, P130: High impedance during reset and after receiving reset.

Figure 29-1 diagram of the reset function



Note that an internal reset of the LVD circuit does not reset the LVD circuit.

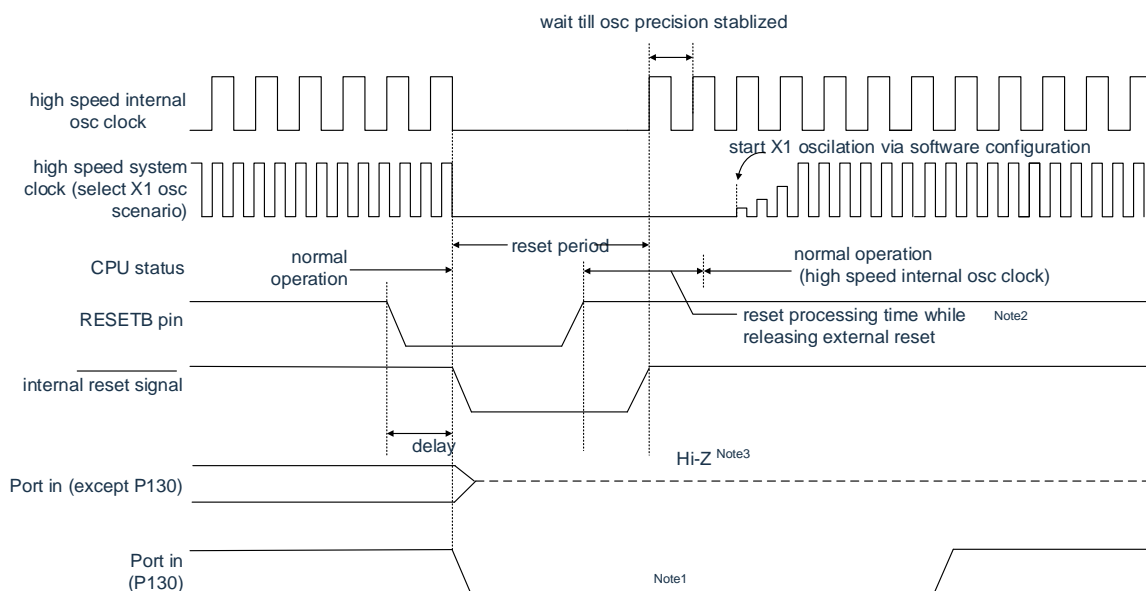
Note 1. LVIM: Voltage sense register

2. LVIS: Voltage sense level register

29.1 Reset timing

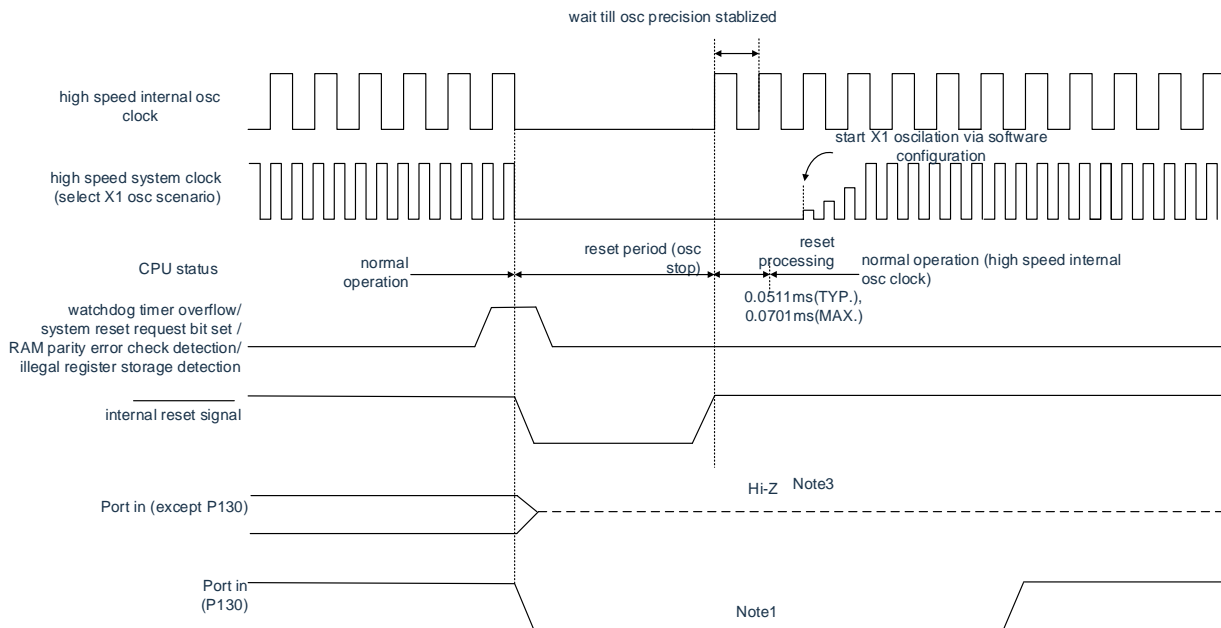
When the RESETB pin is input low, a reset is generated. The reset state is then released if the RESETB lead is entered high and the program begins with a high-speed internal oscillator clock after the reset process is complete.

Figure 29-2 timing of the 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 29-3 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 If a reset occurs, the P130 outputs a low level. Therefore, if the P130 is set to a high output before a reset occurs, the output of the P130 can be virtually output as a reset signal from an external device. To dismiss the reset signal from an external device, the P130 must be set to high via software.

2. Reset processing time when removing external reset:

1st time after POR released: 0.672ms (TYP.), 0.832ms(MAX.) (in the case of LVD).

0.399ms(TYP.), 0.519ms(MAX.) (In the case of not using LVD)

2nd time after the POR released: 0.531ms (TYP.), 0.675ms(MAX.) (in the case of LVD).

0.259ms(TYP.), 0.362ms(MAX.) (when LVD is not used).

When the supply voltage rises, a voltage stabilization wait time of 0.99ms (TYP.) is required before the reset processing time at the time of external reset is lifted, 2.30ms(MAX.) .

3. Port pin P40 changes to the following state:

- High impedance during external reset or POR reset.
- High during other resets and after receiving the reset (connects internal pull-up resistors). Note that the

watchdog timer is no exception, resetting when an internal reset occurs.

For resets resulting from voltage detection of POR circuits and LVD circuits, if $V_{DD} \geq V_{POR}$ or $V_{DD} \geq V_{LVD}$, the reset state is lifted, and after the reset processing begins execution with a high-speed internal oscillator clock. For details, please refer to "Chapter 30 Power-on Reset Circuits" and "Chapter 31" Chapter Voltage Detection Circuits".

Note V_{POR} : The POR supply voltage rises to the detection voltage
 V_{LVD} : LVD sense voltage

Table 29-1 Operational status during reset

Item			During reset
System clock			Stop providing clocks to the CPU.
The master system clock	f _{IH}	Stop running.	
	f _X	Stop operation (pins X1 and X2 are in input port mode).	
	f _{EX}	The clock input is invalid (the pin is in input port mode).	
Subsystem clock	f _{XT}	Can run.	
	f _{EXS}	The clock input is invalid (the pin is in input port mode).	
f _{II}		Stop running.	
CPU			
Code flash			Stop running.
RAM			Stop running.
Port (latch)			High impedance ^{note 1}
DIV			Stop running.
Timer array unit			
Timer A			
Timer M			
Timer B			
Timer C			
Real-time clock (RTC).			When the POR is reset, the operation is stopped. Can be run during other resets.
15-bit interval timer			Stop running.
Watchdog timer			
Clock output/buzzer output			
A/D converter			
D/A converter ^{Note 1}			
Comparator ^{Note 1}			
Universal Serial			
Serial Interface (IICA).			
aFCAN			
Data Transfer Controller			
Power-on reset function			Can perform inspection operations.
Voltage detection function			It can be operated when the LVD is reset. During other resets, the operation stops.
External interrupts			Stop running.
Key interrupt function			
CRC arithmetic	High-speed CRC		
	Universal CRC		
RAM parity function			
SFR protection function			

Note 1 The port pins P40, P130 change to the following state:

- P40: High impedance during external reset or POR reset. High during other resets (connects internal pull-up resistors).
- P130: Output low during reset.

remark f_{IH} : High-speed internal oscillator clock
 f_X : X1 oscillating clock
 f_{EX} : External master system clock
 f_{XT} : XT1 oscillating clock
 f_{EXS} : External subsystem clock
 f_{II} : Low speed internal oscillator clock

29.2 the registers of Confirmed reset source

29.2.1 Reset Control Flag Register (RESF).

The BAT32A2x9 microcontroller has multiple sources of internal reset generation. The Reset Control Flag Register (RESF) holds the reset source at which the reset request occurred. The RESF register can be read via an 8-bit memory operation instruction.

ClearSRF, WDTRF, CLMRF, via reset of RESET B input, reset of power-on reset (POR) circuit, and read of RESF registers RPERF, IAWRF, LVIRF marking. To determine the reset source, the value of the RESF register must be saved to arbitrary RAM and then judged by its RAM value.

Figure 29-4 Reset Control Flag Register (RESF) Format

Address: 40020440H	reset: indefinite value	Note 1	R					
Symbol	7	6	5	4	3	2	1	0
FRSR	SYSRF		0	WDTRF	CLMRF	RPERF	IAWRF	LVIRF

SYSRF	An internal reset request resulting from a system reset request bit being set
0	No internal reset request was generated or the RESF register was cleared.
1	Generates an internal reset request.

WDTRF	Internal reset request generated by the watchdog timer (WDT).
0	No internal reset request was generated or the RESF register was cleared.
1	Generates an internal reset request.

CLMRF	Internal reset request generated by the damping detection function
0	No internal reset request was generated or the RESF register was cleared.
1	Generates an internal reset request.

RPERF	Internal reset request generated by a RAM parity error
0	No internal reset request was generated or the RESF register was cleared.
1	Generates an internal reset request.

IAWRF	Internal reset request generated by accessing illegal memory
0	No internal reset request was generated or the RESF register was cleared.
1	Generates an internal reset request.

LVIRF	Internal reset request generated by the voltage sense circuit (LVD).
0	No internal reset request was generated or the RESF register was cleared.
1	Generates an internal reset request.

Note 1 Varies depending on the reset source. Please refer to Table 29-2.

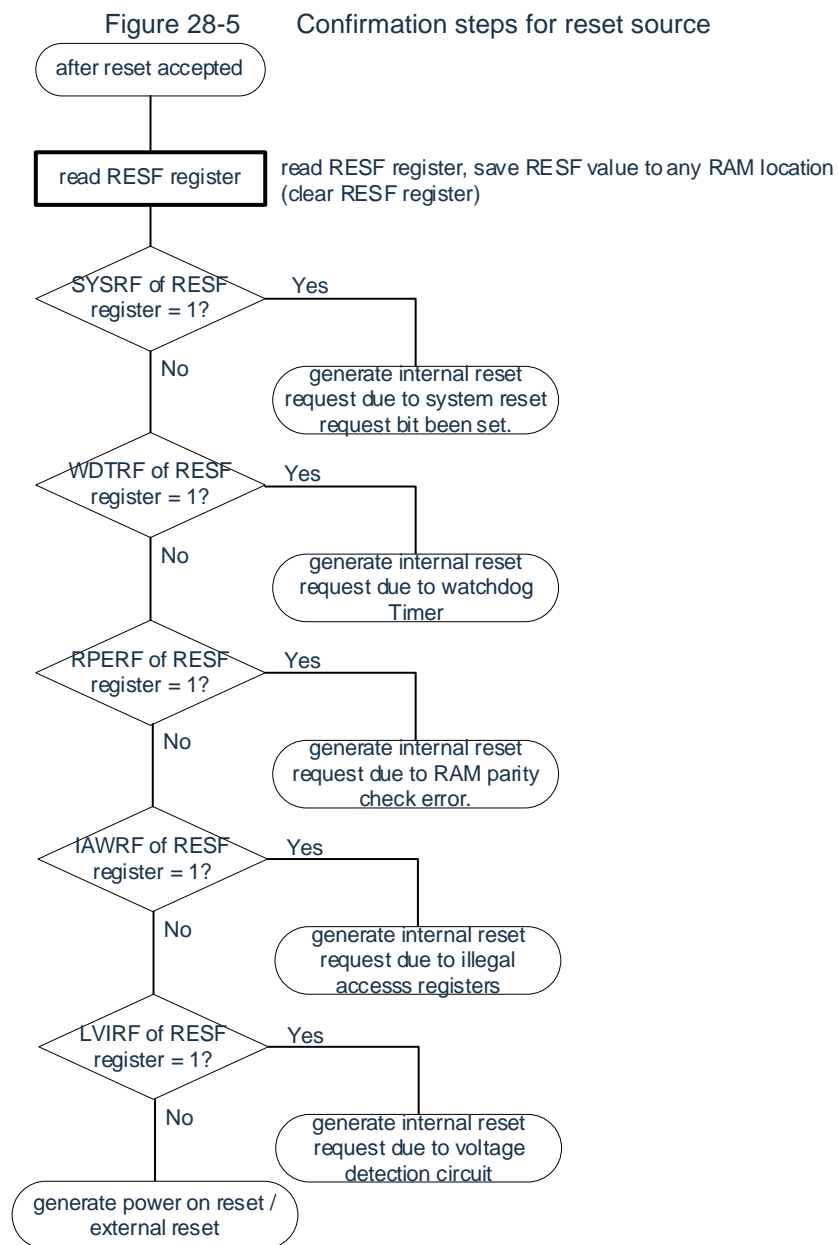
Note that in the case of ALLOWing RAM parity error reset (RPERDIS=0), when accessing data, the "RAM region used" must be initialized; When executing an instruction from a RAM region, the area of "RAM area +10 bytes used" must be initialized. By generating a reset, enter a state that allows the generation of a RAM parity error reset (RPERDIS=0). For details, please refer to "31.3.3 RAM Parity Error Detection Function".

The status of the RESF registers at the time of the reset request is shown in Table 29-2.

Table 29-2 RESF registers when a reset request occurs

<div>Reset the source flag</div>	RESETB input	Reset of POR generated	System reset requests a reset resulting from a position 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	Clear "0"	Clear "0"	Set "1"	keep	keep	keep	keep	keep		
WDTRF			keep	Set "1"						
CLMRF				keep	Set "1"	Set "1"				
RPERF					keep					
IAWRF					keep	Set "1"				
LVIRF					keep	keep	Set "1"			

The confirmation steps for resetting the source are shown in Figure 2 8-5.



Note that the above process is an example of a confirmation step.

Chapter 30 Power-on reset circuit

30.1 Function of the 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 the 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 Data sheet And characteristic 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.

Note When the power-on reset circuit generates an internal reset signal, the reset control flag register (RESF) is cleared to "00H".

Note 1 The BAT32A2x9 includes several hardware that generates an internal reset signal. When an internal reset signal is generated by a watchdog timer (WDT), voltage detection (LVD) circuit, system reset request position bit, RAM parity error, or access to illegal memory, Flags used to represent the reset source are assigned in the REF register; When an internal reset signal is generated due to the assertion of the WDT, LVD, system reset request bit, RAM parity error, or illegal memory access, the RSF register is not cleared to "00H" Instead, place the flag "1". For details on RESF registers, please refer to "Chapter 28 Reset Functions".

2. V_{POR} : POR supply voltage rises to detect voltage

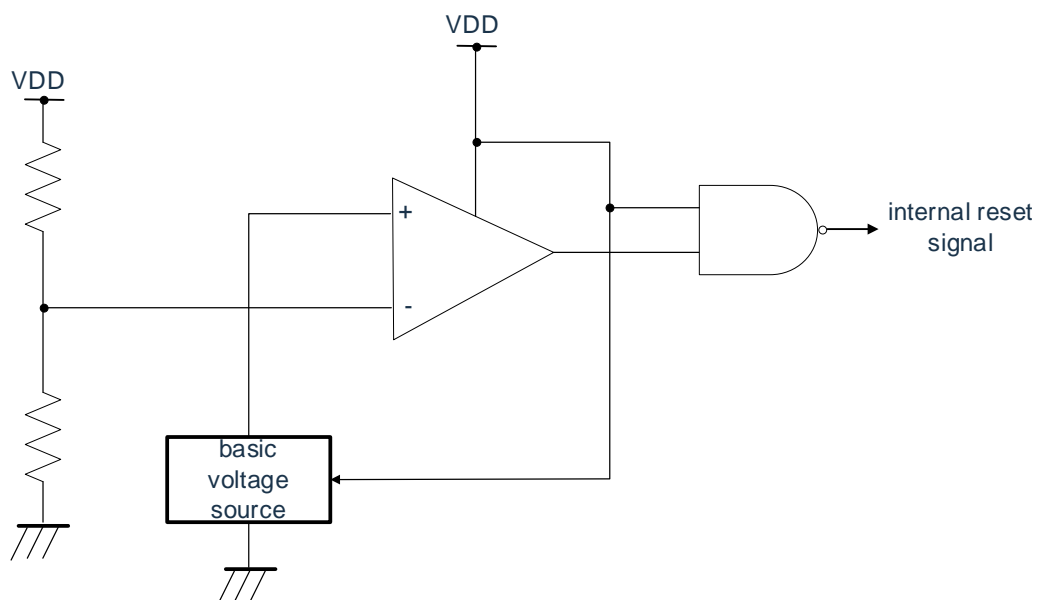
V_{PDR} : POR supply voltage drop sense voltage

For details, please refer to the POR circuit characteristics in the data sheet.

30.2 Structure of the power-on reset circuit

The block diagram of the power-on reset circuit is shown in Figure 29-1.

Fig. 29-1 Block diagram of the power-on reset circuit

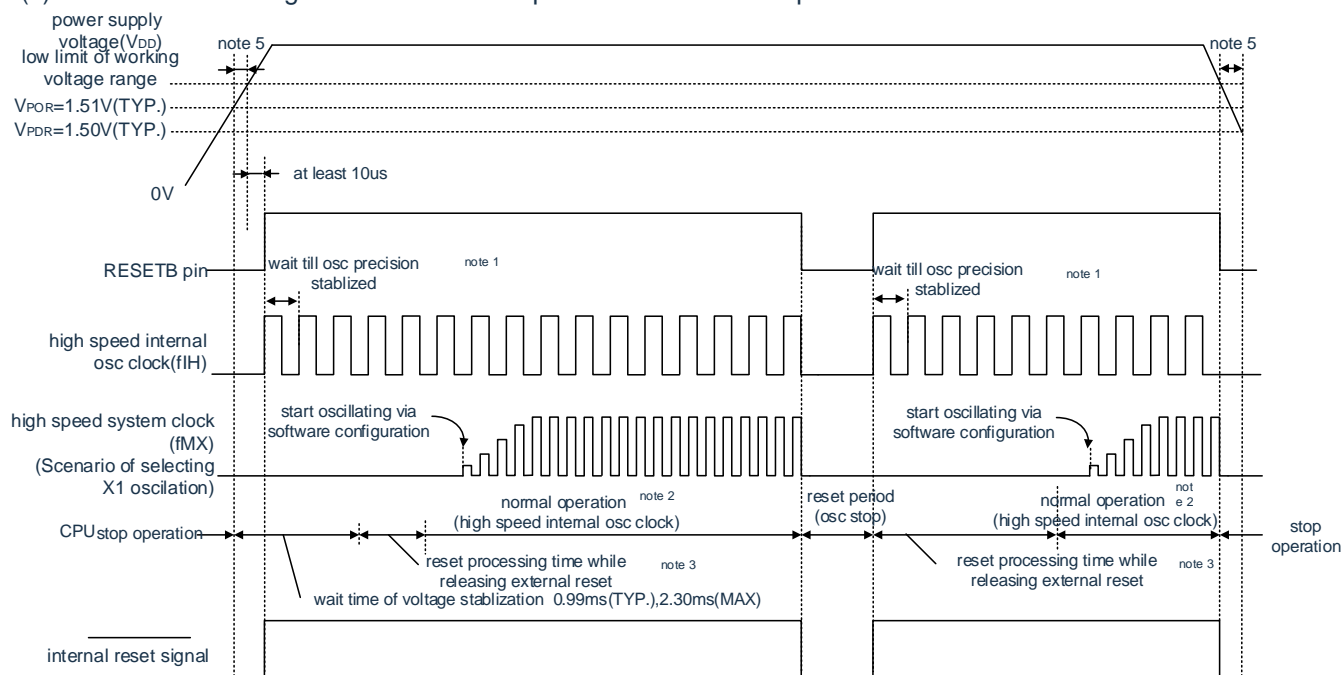


30.3 Operation of the power-on reset circuit

The timing of the generation of internal reset signals in the power-on reset circuit and the voltage detection circuit is as follows.

Fig. 29-2 Timing of the generation of internal reset signals in the power-on reset circuit and voltage detection circuit (1/3).

(1) The case of using an external reset input from the RESETB pin



Note 1 The internal reset processing time includes the oscillation accuracy stabilization wait time for the high-speed internal oscillator clock.

2. Can switch the CPU clock from a high-speed internal oscillator clock to a high-speed system clock or a sub-system clock. In the case of an X1 clock, the oscillation settling time must be switched after the oscillation settling time is confirmed by the state register (OSTC) of the oscillation settling time counter; In the case of using the XT1 clock, it is necessary to switch after confirming the oscillation stabilization time using the timer function, etc.

3. The time until the start of normal operation except for reaching V_{POR} (1.51V (TYP.)). After the "voltage stabilization wait time", the following "is required" after setting the RESETB signal high ("1"). Reset processing time when the external reset is lifted (the first time after the POR is released)". The reset processing time when the external reset is released is as follows:

1st time after POR released: 0.672ms (TYP.) , 0.832ms(MAX.) (in the case of LVD).

0.399ms(TYP.) , 0.519ms(MAX.) (when LVD is not used).

4. The reset processing time when the external reset is released after the second time after the POR is released as follows:

After 2nd time after the POR is released: 0.531ms (TYP.) , 0.675ms(MAX.) (in the case of LVD).

0.259ms(TYP.) , 0.362ms(MAX.) (when LVD is not used).

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.

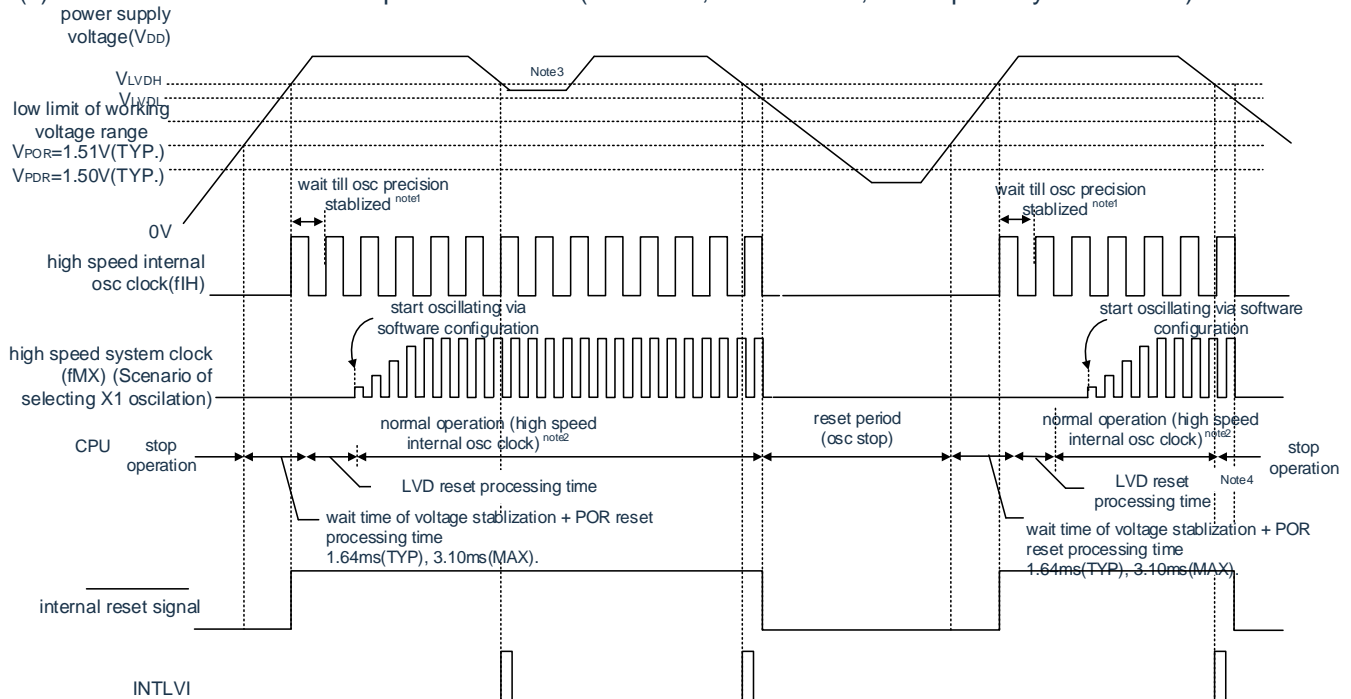
Note V_{POR} : The POR supply voltage rises to the detection voltage

V_{PDR} : POR supply voltage drop sense voltage

Note that when LVD is OFF, an external reset of the RESET B pin must be used. For details, please refer to "Chapter 30 Voltage Detection Circuits".

Figure 29-2 Timing of the generation of internal reset signals in the 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).

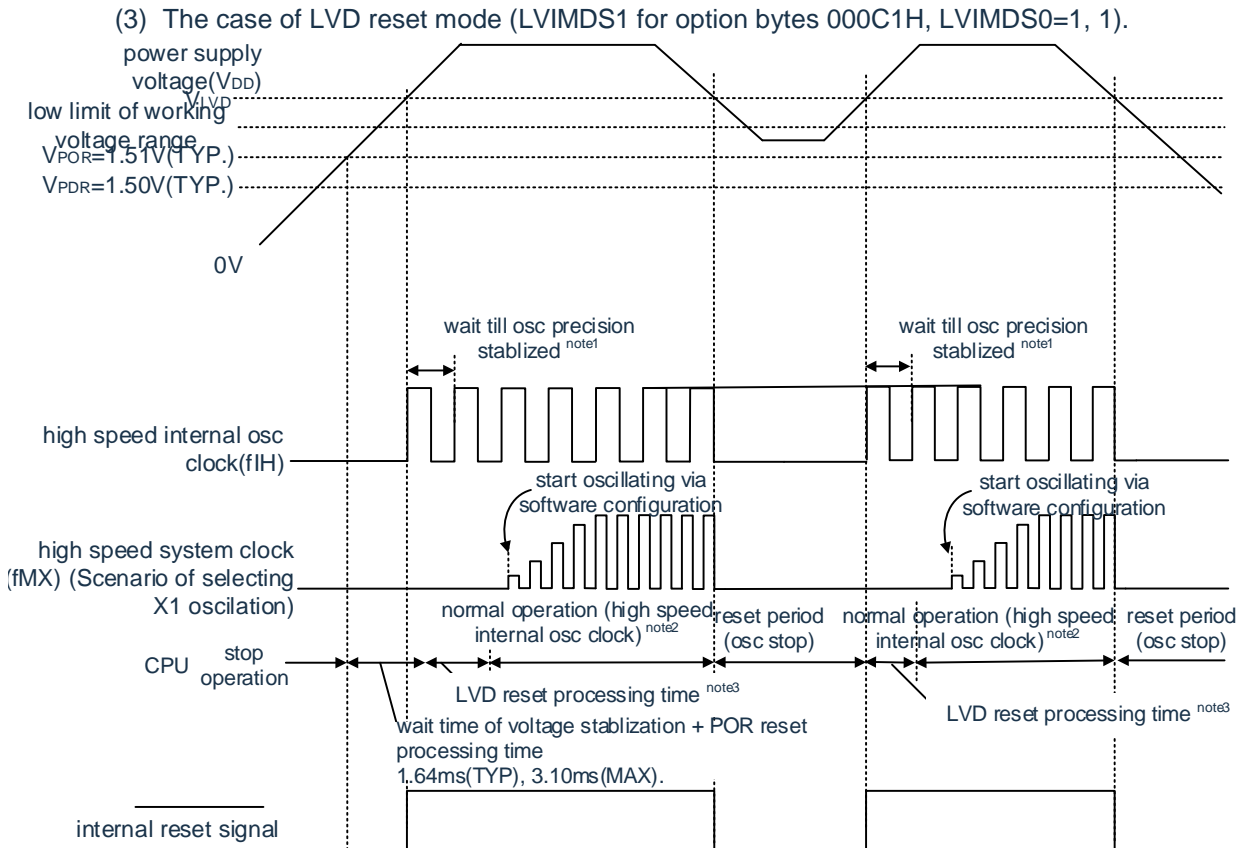


Note 1 The internal reset processing time includes the oscillation accuracy stabilization wait time for the high-speed internal oscillator clock.

- Can switch the CPU clock from a high-speed internal oscillator clock to a high-speed system clock or a sub-system clock. In the case of an X1 clock, the oscillation settling time must be switched after the oscillation settling time is confirmed by the state register (OSTC) of the oscillation settling time counter; In the case of using the XT1 clock, it is necessary to switch after confirming the oscillation stabilization time using the timer function, etc.
- After generating the interrupt request signal (INTLVI), the LVILV bit of the voltage detection level register (LVIS) and the LVIMD position "1" are automatically generated. Therefore, it is important to consider the possibility that the supply voltage will return to a high voltage sense voltage (V_{LV_{DH}}) or higher in a state not less than the low voltage sense voltage (V_{LV_{DL}}), and follow it after the INTLVI is generated "Setup steps for confirmation/reset of the operating voltage of Figure 30-7" and "Figure 30-8 interrupt." &Reset the initial setup step of the mode" to make the settings.
- 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 reaching the LVD detection level (V_{LV_{DH}}). LVD Reset Processing Time.
LVD reset processing time: 0ms~0.0701ms(MAX.).

Note V_{LV_{DH}}, V_{LV_{DL}} : LVD sense voltage
V_{POR} : The POR supply voltage rises to detect the voltage
V_{PDR} : POR supply voltage drop sense voltage

Figure 29-2 Timing of the generation of internal reset signals in the power-on reset circuit and voltage detection circuit (3/3).



Note 1 The internal reset processing time includes the oscillation accuracy stabilization wait time for the high-speed internal oscillator clock.

- Can switch the CPU clock from a high-speed internal oscillator clock to a high-speed system clock or a sub-system clock. In the case of an X1 clock, the oscillation settling time must be switched after the oscillation settling time is confirmed by the state register (OSTC) of the oscillation settling time counter; In the case of using the XT1 clock, it is necessary to switch after confirming the oscillation stabilization time using the timer function, etc.
- 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.).
- 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.)

Note 1 V_{LVDH} , V_{LVDL} : LVD sense voltage

V_{POR} : The POR power supply rises the sense voltage

V_{PDR} : The POR supply drops the sense voltage

- 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 The time of "Note 3" of "Figure 29-2(3) LVD bit mode case" is the same.

Chapter 31 Voltage detection circuit

31.1 The function of the 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.

- Compare supply voltage (VDD) with sense voltage (V_{LVDH} , V_{LVDL} , V_{LVD}). , which generates an internal reset or internal interrupt signal.
- The sense voltage of the supply voltage (V_{LVDH} , V_{LVDL}) can select 12 detection levels by the option byte (see "Section 34." Chapter Option Bytes").
- Can also be operated in deep sleep mode.
- 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/010C2H).

(a) Interrupt & Reset mode (LVIMDS1, LVIMDS0=1, 0).

Select 2 sense voltages (V_{LVDH} , V_{LVDL}) and high voltage sense level (V_{LVDH}) via option byte 000C1H) is used to de-reset or generate an interrupt, and a low voltage sense level (V_{LVDL}) is used to generate a reset.

(b) Reset mode (LVIMDS1, LVIMDS0=1, 1) of the option bytes

Use the 1 sense voltage (V_{LVD}) selected by option byte 000C1H to generate or de-reset.

(c) Interrupt mode (LVIMDS1, LVIMDS0=0, 1) of the options

Use the 1 sense voltage (V_{LVD}) selected by option byte 000C1H to generate an interrupt or de-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)	Break mode (LVIMDS1, LVIMDS0=0, 1)
When the operating voltage drops, it is detected $V_{DD} < V_{LVDH}$, generates an interrupt request signal; When detected $V_{DD} < V_{LVDL}$ when 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 detected $V_{DD} < V_{LVD}$, an internal reset is generated.	After a reset occurs, the LVD's internal reset state follows Continue until $V_{DD} \geq V_{LVD}$. When detected When $V_{DD} \geq V_{LVD}$, the internal reset of LVD is released. If detected after removing the internal reset of lvd $V_{DD} < V_{LVD}$ or $V_{DD} \geq V_{LVD}$ When, yes Generates an Interrupt Request Signal (INTLVI).

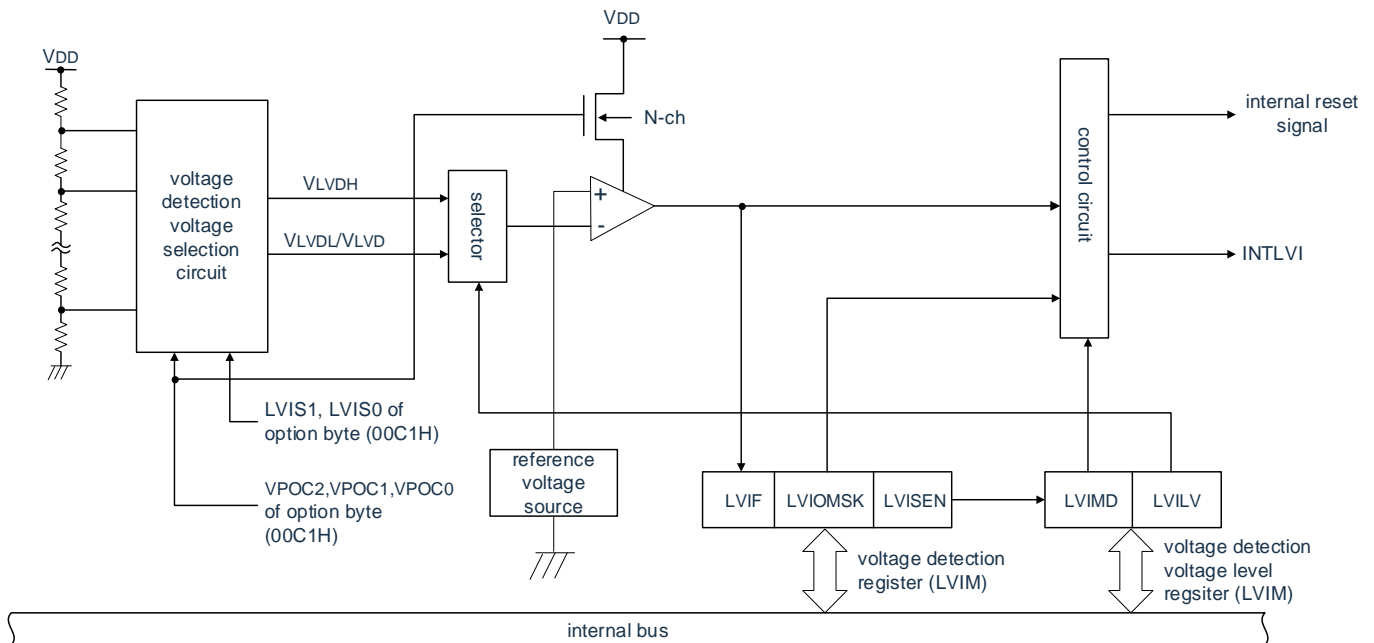
When running a voltage detection circuit, it is possible to confirm whether the supply voltage is greater than or equal to the sense voltage or less than the sense voltage by reading the voltage detection flag (LVIF: bit0 of the voltage detection register (LVIM)).

If a reset occurs, set bit0 (LVIRF) of the reset control flag register (RESF) to "1". For details on RESF registers, refer to Chapter 29 Reset Functions.

31.2 Structure of the voltage detection circuit

A block diagram of the voltage detection circuit is shown in Figure 31-1.

Figure 31-1 diagram of the voltage detection circuit



31.3 Control Registers of the voltage detection circuit

The voltage detection circuit is controlled by the following registers.

- Voltage Sense Register (LVIM).
- Voltage Sense Level Register (LVIS).

31.3.1 Voltage Sense Register (LVIM).

This register setting enable or disables the overriding of the voltage sense level register (LVIS) and confirms the shielding status of the LVD output. Set the LVIM registers via the 8-bit memory manipulation instructions.

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

Figure 31-2 Format of the Voltage Sense Register (LVIM).

Address: 40020441H reset value: 00H ^{Note 1} R/W ^{Note 2}

symbol	7	6	5	4	3	2	1	0
LVIM	LVISEN ^{note3}	0	0	0	0	0	LWOTBSP	LVIF

LVISEN ^{Note 3}	Enable/disable override settings for the Voltage Sense Level Register
0	Rewriting of the LVIS registers (LVIOMSK=0 (LVD output masking is invalid)) is prohibited.
1	Rewriting of LVIS registers is allowed (LVIOMSK=1 (LVD output masking is valid)).

LVIOMSK	Mask status flag for LVD output
0	LVD output masking is invalid.
1	LVD output mask valid ^{note 4} .

LVIF	Voltage detection flag
0	The supply voltage (V_{DD}) \geq sense voltage (V_{LVD}) or LVD is OFF.
1	Supply voltage (V_{DD}) < detect 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".

2. Bit0 and bit1 are read-only bits.

3. It can only be set when the interrupt & reset mode is selected (lvIMDS1 bits and LVIMDS0 bits of the option bytes are "1" and "0" respectively), the initial value cannot be changed in other modes.

4. Only when the interrupt & reset mode is selected (the LVIMDS1 bit and LVIMDS0 bits of the option byte are "1" and "0" respectively). The LVIOMSK bit automatically changes to "1" during the following periods, masking the reset or interrupt generated by LVD.

- Period of LVISEN=1
- Wait time from the time of the occurrence of an LVD interrupt until the LVD sense voltage stabilizes
- Wait time from changing the value of the LVILV bit until the LVD sense voltage stabilizes

31.3.2 Voltage Sense Level Register (LVIS).

This is the register that sets the voltage sense level.

Set the LVIS registers via the 8-bit memory manipulation instructions. After generating a reset signal, the value of this register changes to "00H/01H/81H" Note 1.

Figure 30-3 Format of the Voltage Sense Level Register (LVIS).

Address: FFFAAH	After reset: 00H/01H/81H Note 1							R/W
symbol	7	6	5	4	3	2	1	0
LVIS	LWEMD ^{note2}	0	0	0	0	0	0	LVILV ^{note2}

LVIMD ^{Note 2}	Operating mode for voltage detection
0	Break mode
1	Reset mode

LVILV ^{Note 2}	LVD detects the level
0	High voltage sense level (V_{LVDH}).
1	Low voltage sense level (V_{LVDL} or V_{LVD}).

Note 1 The reset value varies depending on the setting of the reset source and option bytes. When an LVD reset occurs, this register is not cleared to "00H".

When a reset other than LVD occurs, the values of this register are as follows:

- Option bytes for LVIMDS1, LVIMDS0=1, 0: 00H
- Option bytes for LVIMDS1, LVIMDS0=1, 1: 81H
- Option bytes for LVIMDS1, LVIMDS0=0, 1: 01H

2. Write "0" only if interrupt & reset mode is selected (LVIMDS1 bit and LVIMDS0 bits for option bytes are "1" and "0" respectively). In other cases, it cannot be set. In interrupt & reset mode, value substitution is performed automatically by generating a reset or interrupt.

Note 1 To rewrite the LVIS registers, it must be done in accordance with the steps in Figures 30-7 and 30-8.

2. Select the operating mode of LVD and the detection voltage of each mode (V_{LVDH} , V_{LVDL} , V) by option byte 000C1H LVD) . The format of the user option bytes (000C1H/010C1H) is shown in Table 30-1. For details on option bytes, refer to Chapter 33 Option Bytes.

Table 30-1 Format of User Option Bytes (000C1H/010C1H) (1/2).

Address: 000C1H/010C1H ^{Note}

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD settings (interrupt & reset mode).

Detect voltage			The setting value of the option byte						
IN _{LVDH}		IN _{LVDL}	VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode settings	
rise	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).

Detect voltage		The setting value of the option byte						
V _{LVD}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode settings	
rise	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.						

Note 1 For details on LVD circuits, refer to Chapter 30 Voltage Detection Circuits.

2. The detection voltage is TYP Value. For details, please refer to the LVD circuit characteristics in the data sheet.

Table 30-1 Format of User Option Bytes (000C1H) (2/2).

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

• LVD settings (interrupt mode).

Detect voltage		The setting value of the option byte						
V _{LVD}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode settings	
rise	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 as OFF(Use.)RESETB External reset of pins)

Detect voltage		The setting value of the option byte						
V _{LVD}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode settings	
rise	falling						LVIMDS1	LVIMDS0
—	—	1	x	x	x	x	x	1
—	—	Setting values other than those described above is prohibited.						

Note 1 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 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).

Note 1 x: Ignore

- For details of LVD circuits, please refer to "Chapter 30 Voltage Detection Circuits".
- The detection voltage is TYP Value. For details, please refer to the LVD circuit characteristics in the data sheet.

31.4 Operation of the voltage detection circuit

31.4.1 Used as a setting when reset mode

The operating mode (reset mode (RESET mode (LVIMDS1, LVIMDS0=1, 1)) and the sense voltage (V) are set by the option byte 000C1H_{LVD}). If you set the reset mode, it starts running in the following initial setup state.

- Set the bit7 (LVISEN) of the Voltage Sense Register (LVIM) to "0" (disables overwriting of the Voltage Sense Level Register (LVIS)).
- Set the initial value of the Voltage Sense Level Register (LVIS) to "81H". bit7 (LVIMD) is "1" (reset mode).
bit0 (LVILV) is "1" (voltage sense level: V_{LVD}).

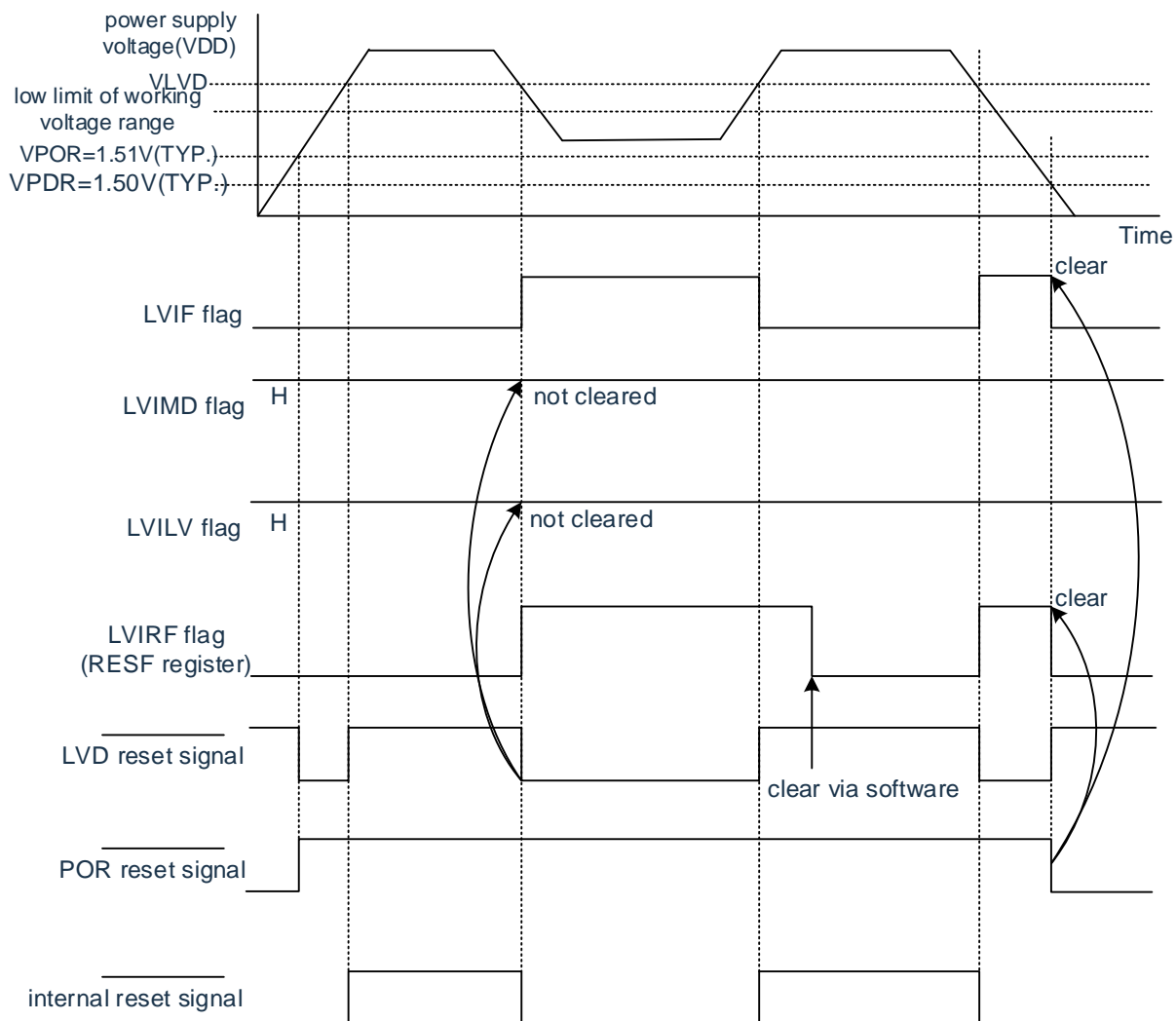
- Operation of LVD reset mode

When power is turned on, the reset mode (LVIMDS1, LVIMDS0=1, 1) exceeds the voltage sense level (V_{LVD}) at the supply voltage (V_{DD}) before maintaining the internal reset state of the LVD. If the supply voltage (V_{DD}) exceeds the voltage sense level (V_{LVD}), the internal reset is dismissed.

When the operating voltage drops, an internal reset of the LVD occurs if the supply voltage (V_{DD}) falls below the voltage sense level (V_{LVD}).

The timing of the generation of the internal reset signal in the LVD reset mode is shown in Figure 30-4.

Figure 30-4 Timing of generation of internal reset signals (LVIMDS1, LVIMDS0=1, 1) of the option bytes



Note V_{POR} : The POR supply voltage rises to the detection voltage
 V_{PDR} : POR supply voltage drop sense voltage

31.4.2 Used as a setting when breaking mode

The operating mode (interrupt mode (LVIMDS1, LVIMDS0=0, 1)) and the sense voltage (V) are set by the option byte 000C1H_{LVD}). If you set the break mode, it starts running in the following initial setup state.

- Set the bit7 (LVISEN) of the Voltage Sense Register (LVIM) to "0" (disables overwriting of the Voltage Sense Level Register (LVIS)).
- Set the initial value of the Voltage Sense Level Register (LVIS) to "01H". Bit7 (LVIMD) is "0" (interrupt mode).
bit0 (LVILV) is "1" (voltage sense level: V_{LVD}).

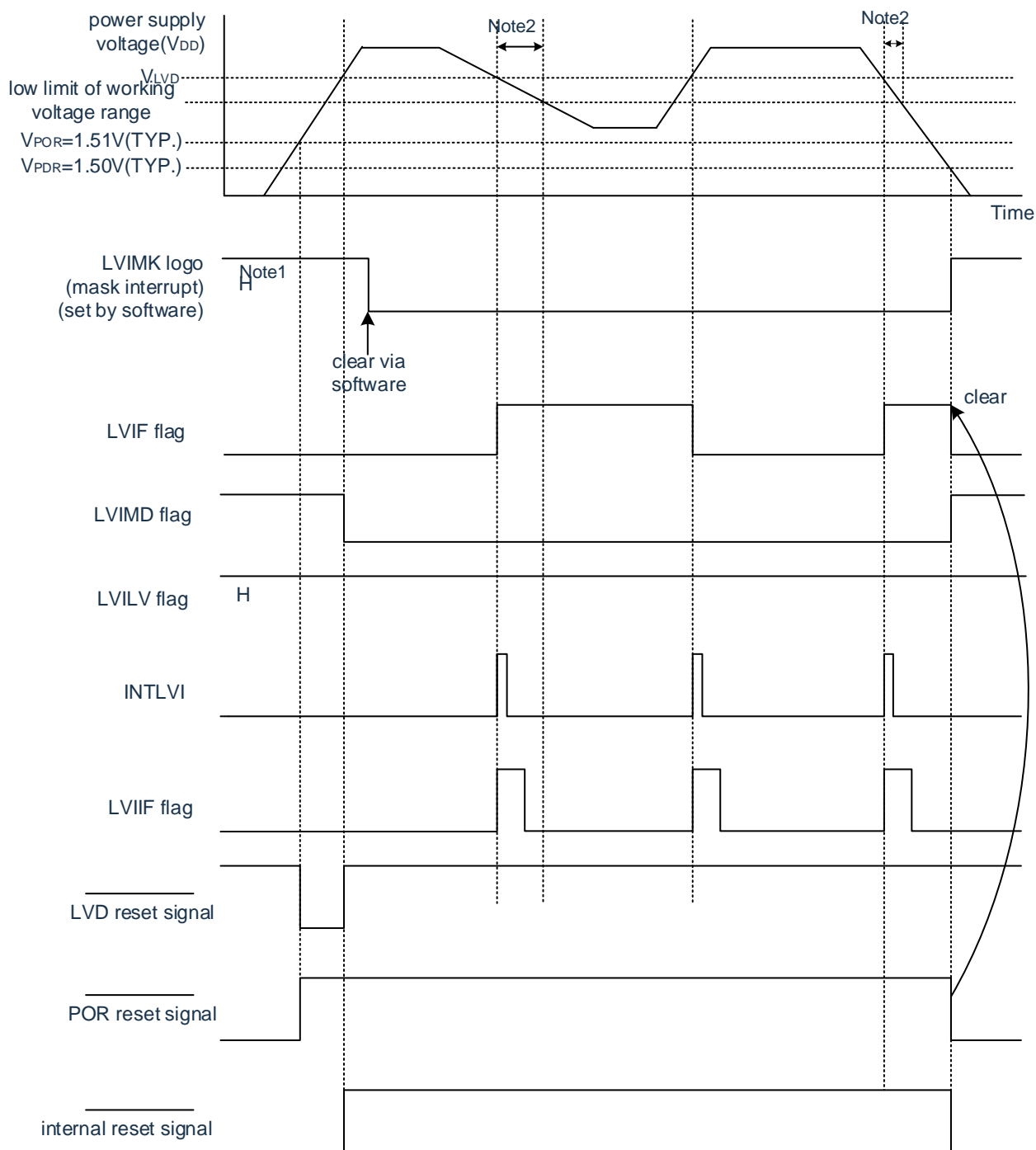
- Operation of LVD interrupt mode

After generating a reset, the interrupt mode (LVIMDS1, LVIMDS0=0, 1) exceeds the voltage sense level (V_{LVD}) at the supply voltage (V_{DD}) before maintaining the internal reset state of the LVD. If the supply voltage (V_{DD}) exceeds the voltage sense level (V_{LVD}), the internal reset of the LVD is released.

After removing the internal reset of lvd, if the supply voltage (VDD) exceeds the voltage sense level (VLVD), an interrupt request signal for LVD is generated (.D INTLVI). When the operating voltage drops, it must be reset by shifting in deep sleep mode 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 important to confirm whether the supply voltage has returned to the operating voltage range.

The timing of the generation of interrupt request signals in LVD interrupt mode is shown in Figure 30-5.

Figure 30-5 Timing of the generation of interrupt signals (LVIMDS1, LVIMDS0=0, 1) of the option bytes



Note 1 After generating a reset signal, the LVIMK flag changes to "1".

2. When the working voltage drops, it must be reset by the transfer of deep sleep mode or external reset before the working voltage is lower than the working voltage range shown in the AC characteristics of the data sheet. When restarting operation, you must confirm that the supply voltage has returned to the operating voltage range.

Note V_{POR} : The POR supply voltage rises to the detection voltage

V_{PDR} : POR supply voltage drop sense voltage

31.4.3 Used as a setting when interrupt & reset mode

The operating mode (interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0)) and the sense voltage (V) are set by the option byte 000C1H_{LV_{VDH}, V_{LVDL}}。

If you set the Interrupt & Reset mode, it starts running in the following initial setup state.

- Set the bit7 (LVISEN) of the Voltage Sense Register (LVIM) to "0" (disables overwriting of the Voltage Sense Level Register (LVIS))。
- Set the initial value of the Voltage Sense Level Register (LVIS) to "00H". Bit7 (LVIMD) is "0" (interrupt mode). bit0 (LVILV) is "0" (high voltage sense level: VLVDH).

- LVD interrupt & reset mode operation

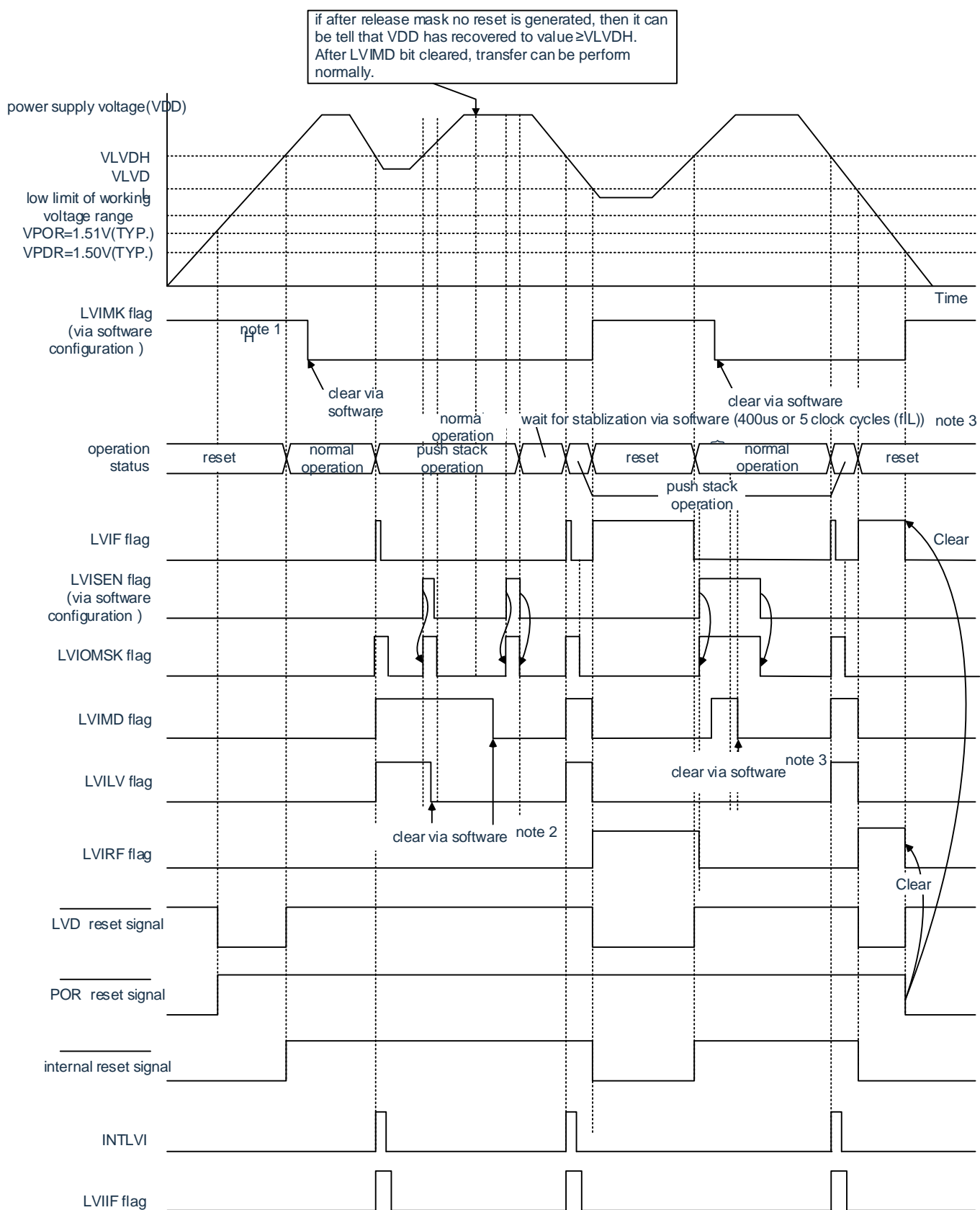
When powered on, interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0) exceeds the high voltage sense level (V_{DD}) at the supply voltage (V_{DD} V_{LVDH}) remains in the internal reset state of the LVD before. If the supply voltage (V_{DD}) exceeds the high voltage sense level (V_{LVDH}), the internal reset is dismissed.

When the operating voltage drops, if the supply voltage (V_{DD}) falls below the high voltage sense level (V_{LVDH}), an interrupt request signal (INTLVI) of the LVD is generated) and can perform arbitrary stack processing. Thereafter, if the supply voltage (V_{DD}) is below the low voltage sense level (V_{LVDL}), which results in an internal reset of the LVD. However, after the occurrence of INTLVI, even the supply voltage (V_{DD}) reverts to a high voltage sense voltage (V) in a state that is not lower than the low voltage sense voltage (V_{LVDL} · V_{LVDH}) or higher, and does not generate an interrupt request signal.

When using LVD interrupt & reset mode, it is necessary to follow the "Setup Steps for Confirmation/Reset of the Operating Voltage of Figure 30-7" and "Figure 30-8 In the initial setup steps of the interrupt & reset mode" steps shown in the flowchart are set up.

The timing of the internal reset signal and interrupt signal generation in LVD interrupt & reset mode is shown in Figure 30-6.

Figure 30-6 Reset & Interrupt Signal Generation Timing (LVIMDS1, LVIMDS0=1, 0) (1/2).



Note 1 After generating a reset signal, the LVIMK flag changes to "1".

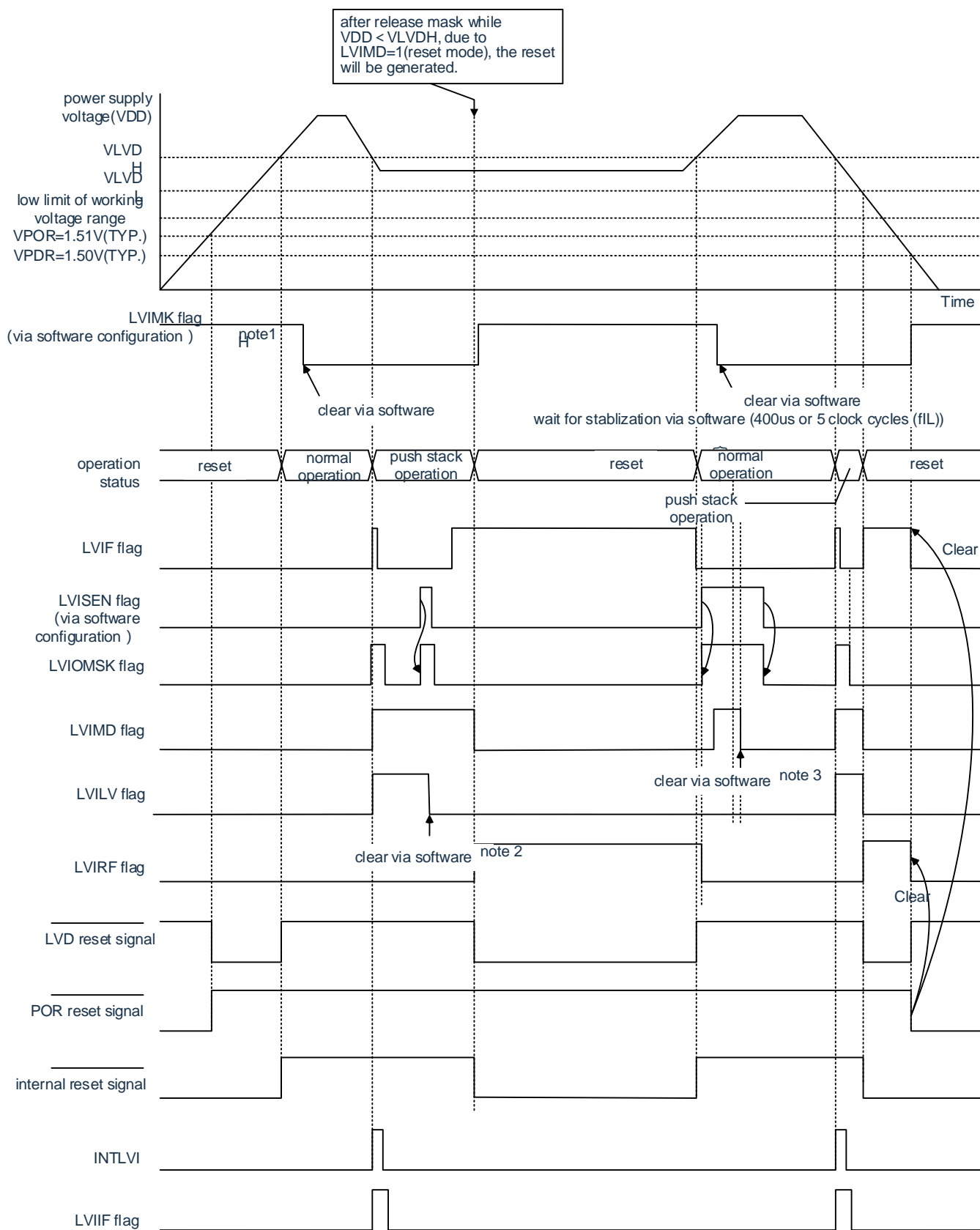
2. When using interrupt & reset mode, it must be set according to the "Setting Steps for Confirmation/Reset of the Working Voltage of Figure 30-7" after the interruption occurs.

3. When using the In progress & reset mode, it must be set according to the "Figure 30-8 Initial Setup Steps for Interrupt & Reset Mode" after the reset is lifted.

Note V_{POR} : The POR supply voltage rises to the detection voltage

V_{PDR} : POR supply voltage drop sense voltage

Figure 30-6 Timing of the generation of interrupt & reset signals (LVIMDS1, LVIMDS0=1, 0) (2/2).



Note 1 After generating a reset signal, the LVIMK flag changes to "1".

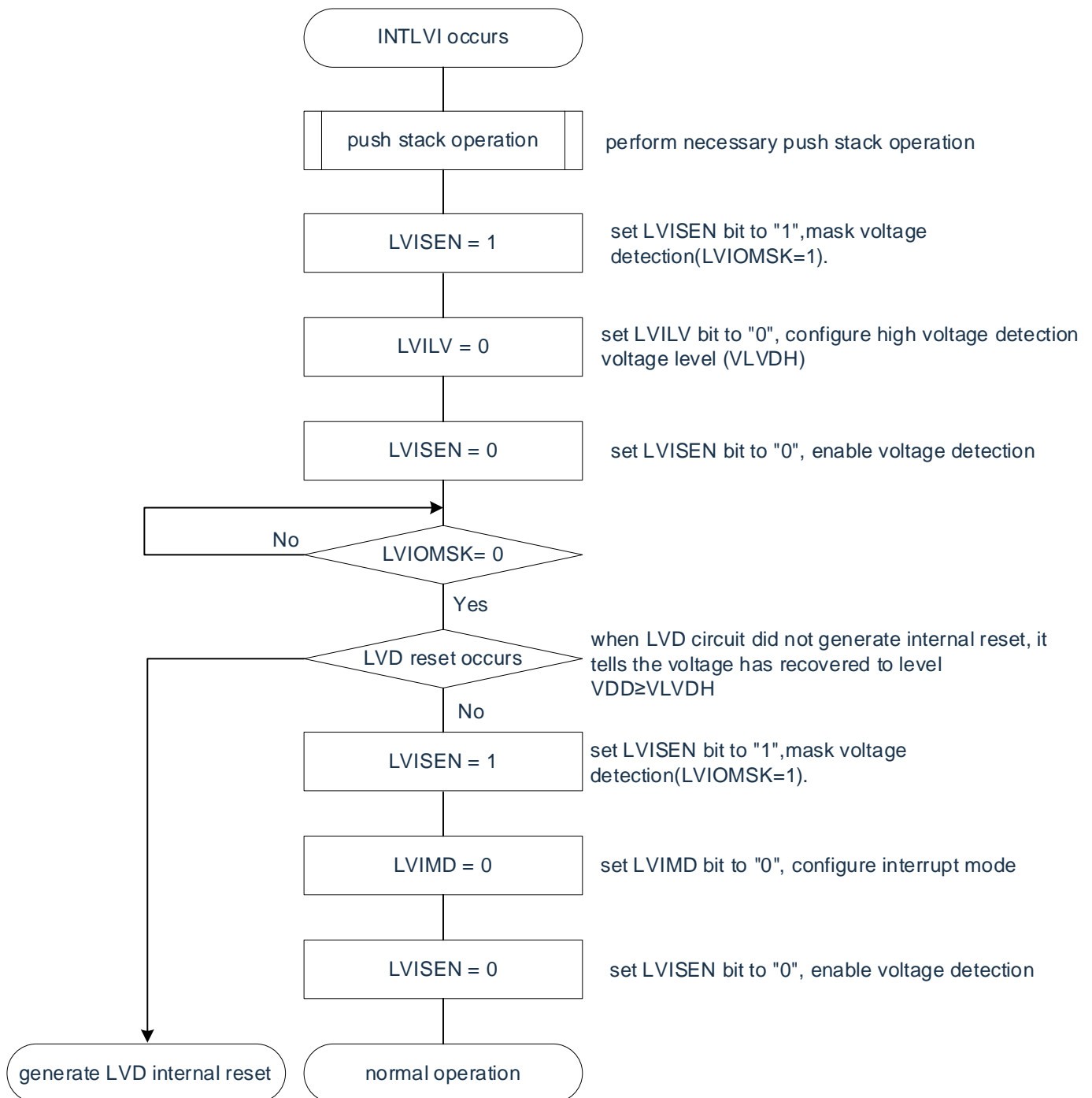
2. When using interrupt & reset mode, it must be set according to the "Setting Steps for Confirmation/Reset of the Working Voltage of Figure 30-7" after the interruption occurs.

3. When using interrupt & reset mode, it must be set according to "Figure 30-8 Initial Setup Steps of Interrupt & Reset Mode" after de-resetting.

Note V_{POR} : The POR supply voltage rises to the detection voltage

V_{PDR} : POR supply voltage drop sense voltage

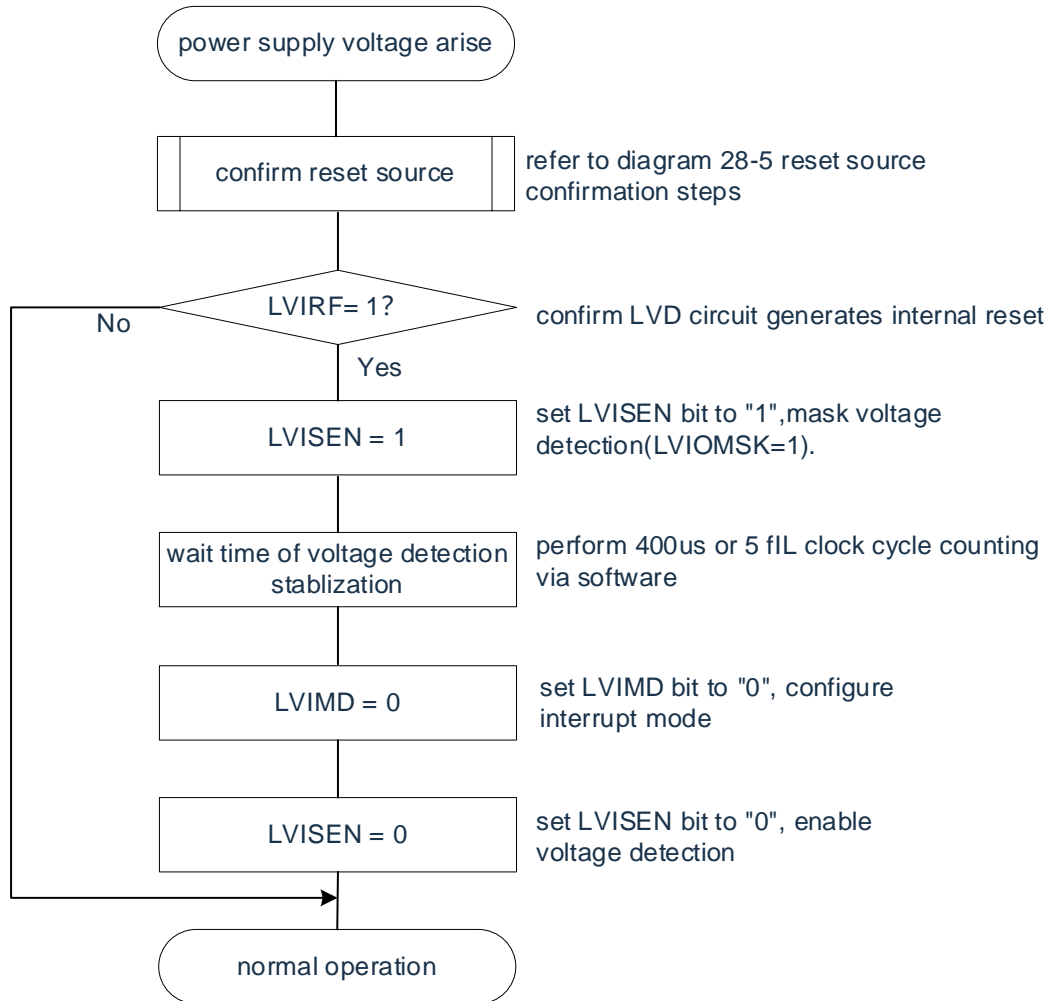
Figure 30-7 The setup steps for confirming/resetting the operating voltage



If you set the interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0), it is required just after the LVD reset (LVIRF=1) is lifted Voltage detection settling wait time for 400 us or 5 f_{IL} clocks. The LVIMD bit must be initialized with a clearance of "0" after waiting for the voltage detection to stabilize. During the counting of voltage detection stabilization wait times and when rewriting the LVIMD bit, the LVISEN position "1" must be used to shield the reset or interrupt generated by the LVD.

The initial setup steps for interrupt & reset mode are shown in Figure 30-8.

Figure 30-8 Initial setup steps for interrupt & reset mode



Note f_{IL} : Low-speed internal oscillator clock frequency

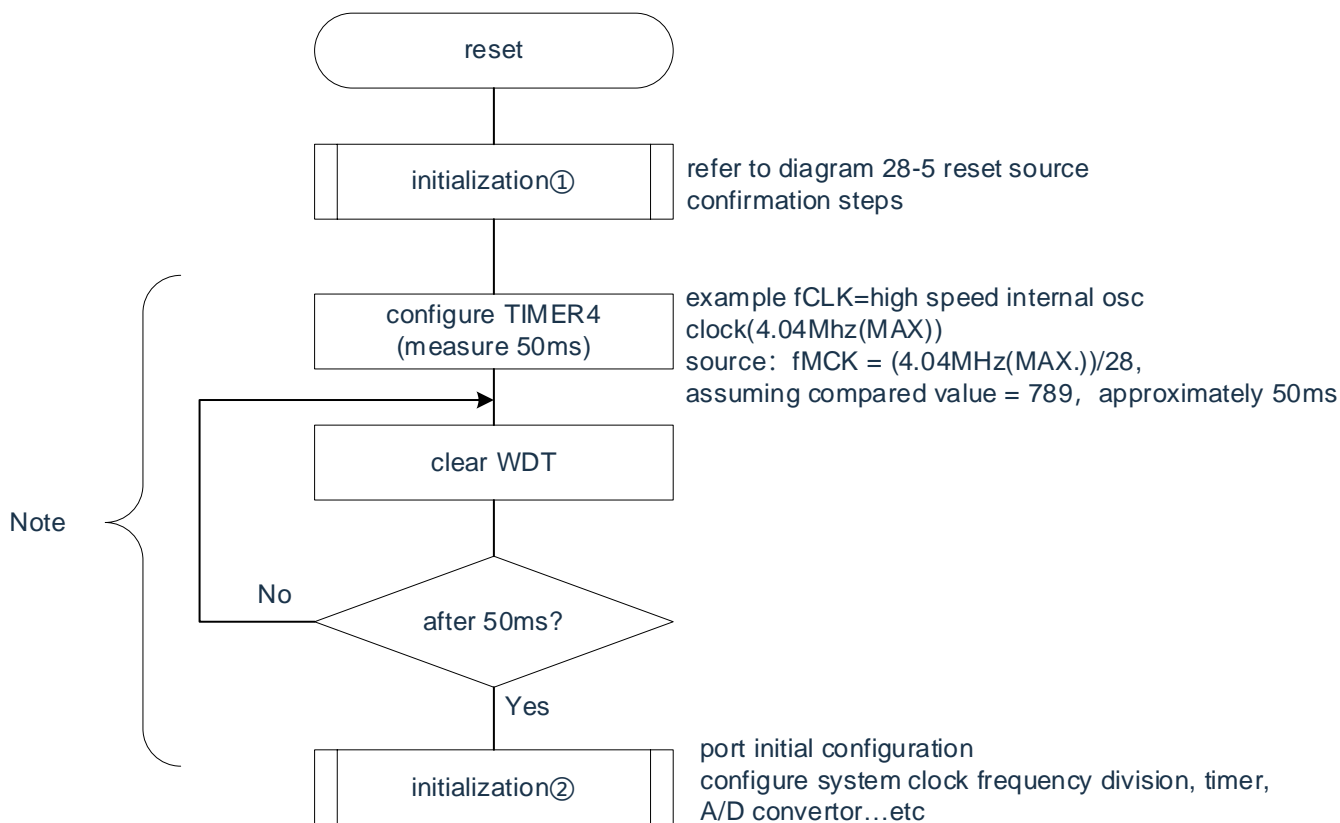
31.5 Considerations for voltage detection circuits

(1) About voltage fluctuations when power is turned on

For systems where the supply voltage (V_{DD}) 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.

Fig. 30-9 Example of software processing when the supply voltage fluctuation near the LVD detection voltage does not exceed 50ms

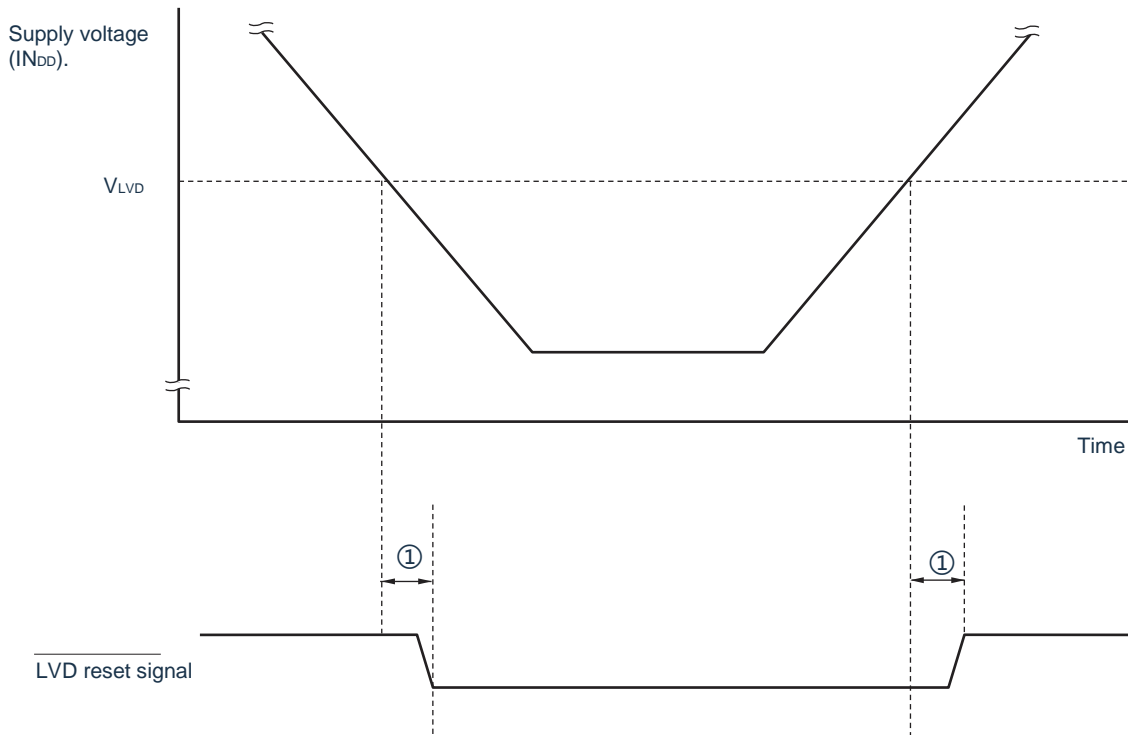


Note If the reset occurs again during this period, it is not transferred to initialization processing (2).

(2) Delay from generating the LVD reset source to generating or removing the LVD reset

From meeting the supply voltage (V_{DD}) < V_{LVD} Detection voltage (V_{LVD}) to produce LVD There is a delay until the reset is made. Again, from V_{LVD} Detection voltage (V_{LVD}) ≤ supply voltage (V_{DD}) to lift LVD A delay also occurs until the reset is made (see figure 30-10).

Fig. 30-10 Delay from generating the LVD reset source to generating or removing the LVD reset



(1): Detection delay (300us(MAX.)))

(3) About the situation where the LVD is plugged in when the power is turned on as OFF

When lvd is set to OFF, an external reset of the RESET B pin must be used.

When performing an external reset, a low of 10us must be input to the RESETB pin. If an external reset occurs while the supply voltage rises, the power must be switched on after inputting a low level to the RESETB pin and keeping it at least 10us low over the operating voltage range shown in the AC characteristics of the data sheet, and then entering the high level.

(4) About the case when the operating voltage drops when LVD is set to OFF and set to LVD interrupt mode

In the case of setting LVD to OFF and to LVD interrupt mode, if the operating voltage drops, it must be reset by transferring the deep sleep mode or external reset before the operating voltage falls below the operating voltage range indicated by the AC characteristics of the data sheet. When restarting operation, you must confirm that the supply voltage has returned to the operating voltage range.

Chapter 32 Security features

32.1 Overview of the security features

In order to be compliant to the IEC60730 and EC61508 safety standards, BAT32A239 has the following safety functions built in.

The purpose of this safety function is to safely stop work when a fault is detected through the self-diagnosis of the microcontroller.

(1) Flash CRC operation function (high-speed CRC, general-purpose CRC).

Detect data errors in flash memory by CRC operations. The following two CRCs can be used according to different uses and usage conditions.

- "High Speed CRC"... In the initializer, it is possible to stop the CPU from running and check the entire code flash area at high speed.
- "Generic CRC"... In CPU operation, it is not limited to the code flash memory area but can be used for multi-purpose inspection.

(2) RAM parity error detection

When reading RAM data, parity errors are detected.

(3) SFR protection function

Prevents rewriting the SFR due to CPU loss of control.

(4) Frequency detection function

Self-test of CPU/peripheral hardware clock frequency can be performed using a general-purpose timer unit.

(5) A/D test function

A/D that can pass through the positive (+) reference voltage, negative (–) reference voltage, analog input channel (ANI), temperature sensor output, and internal reference voltage output of the A/D converter. The conversion performs a self-test of the A/D converter.

(6) Digital output signal level detection function of input/output ports

When the input/output ports are in output mode, the output level of the pins can be read.

(7) Product unique identifier register (128-bit)

32.2 Registers used by the security function

The following registers are used for each function of the safety function.

Register name	The functions of the security function
<ul style="list-style-type: none"> Flash CRC control register (CRC0CTL). Flash CRC Operation Result Register (PGCRCL). 	Flash CRC operation function (High Speed CRC).
<ul style="list-style-type: none"> CRC Input Register (CRCIN). CRC Data Register (CRCD). 	CRC 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

The contents of each register are described in "32.3 Operation of security features".

32.3 Operation of security features

32.3.1 Flash CRC operation function (high-speed CRC).

The IEC60730 standard requires confirmation of data in flash memory and recommends CRC as a means of confirmation. This high-speed CRC checks the entire code flash area during the initial setup (initialization) procedure.

The high-speed CRC stops the CPU and reads 32-bit data from flash through a clock for operation. As a result, it is characterized by a shorter time to complete the inspection (e.g., 64KB flash memory: 512us@32MHz).

The CRC generates polynomials corresponding to CRC-16-CCITT's " $X^{16}+X^{12}+X^5+1$ ".

MsBs of bit31 bit0 are prioritized.

Note Because the generic CRC is LSB first, the results of the operation are different.

32.3.1.1 Flash CRC control register (CRC0CTL).

This is the register that sets the operational control and operation range of the high-speed CRC operator. The CRC0CTL register is set via the 8-bit memory manipulation instruction. After generating a reset signal, the value of this register changes to "00H".

Figure 32-1 Format of the flash CRC control register (CRC0CTL).

Address: 40021810H reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CRC0CTL	CRC0EN	FEA6	FEA5	FEA4	FEA3	FEA2	FEA1	FEA0

CRC0EN	Operational control of high-speed CRC operators
0	Stop 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)

Note: 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.

32.3.1.2 Flash CRC operation result register (PGCRCL).

This is the register that holds the results of the high-speed CRC operation.

Sets the PGCRCL register via the 16-bit memory manipulation instruction.

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

Figure 32-2 Flash CRC Operation Result Register (PGCRCL).

Address:	0x40021812	after reset:	0000H	R/W				
symbol	15	14	13	12	11	10	9	8
PGCRCL	PGCRC15	PGCRC14	PGCRC13	PGCRC12	PGCRC11	PGCRC10	PGCRC9	PGCRC8
	7	6	5	4	3	2	1	0
	PGCRC7	PGCRC6	PGCRC5	PGCRC4	PGCRC3	PGCRC2	PGCRC1	PGCRC0

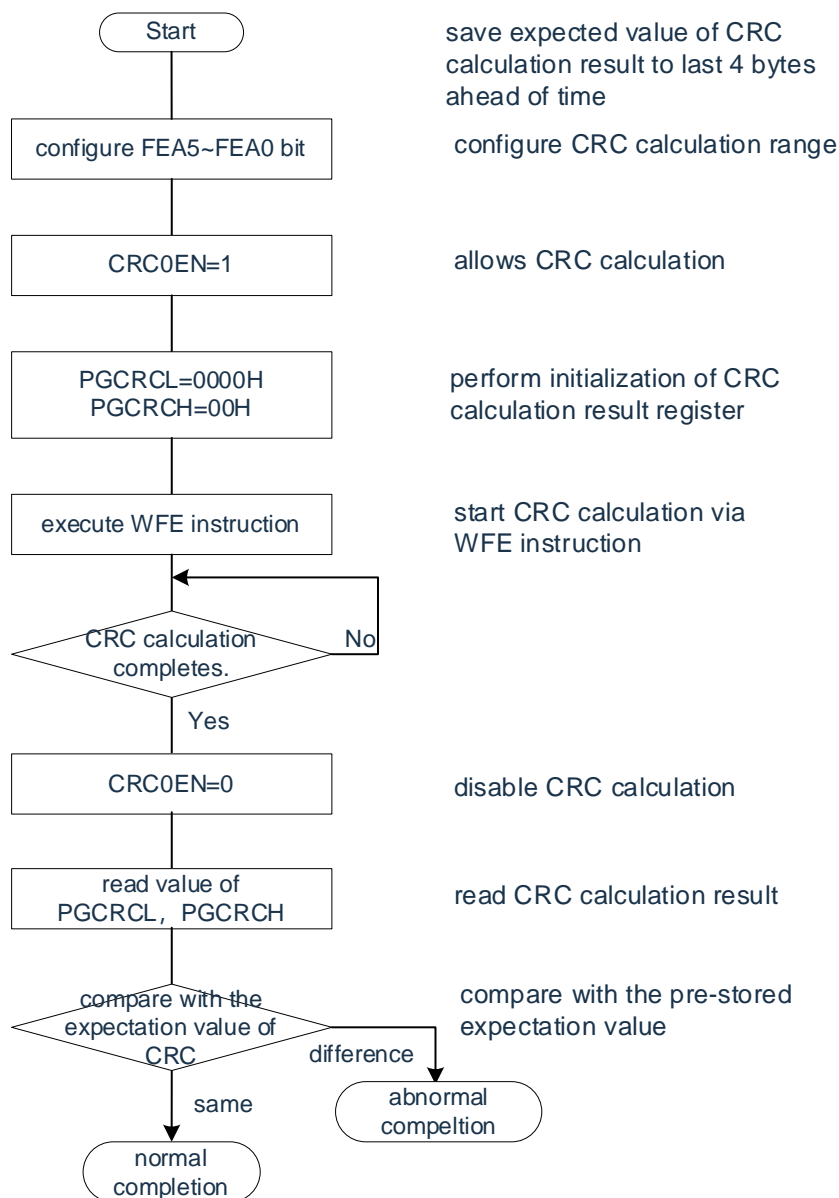
PGCRC15~0	The result of the operation of the high-speed CRC
0000H~FFFFH	Saves the calculation results of the high-speed CRC.

Note that the PGCRCL register can only be written if the CRC0EN (bit7 of the CRC0CTL register) bit is "1".

The flowchart of the flash CRC operation function (high-speed CRC) is shown in Figure 32-3.

< operation process >

Figure 32-3 the flash CRC operation function (high-speed CRC).



Note 1 Objects that operate only on code flash as CRC operations.

2. The expected value of the CRC operation must be saved in the area after the operation range in the code flash memory.

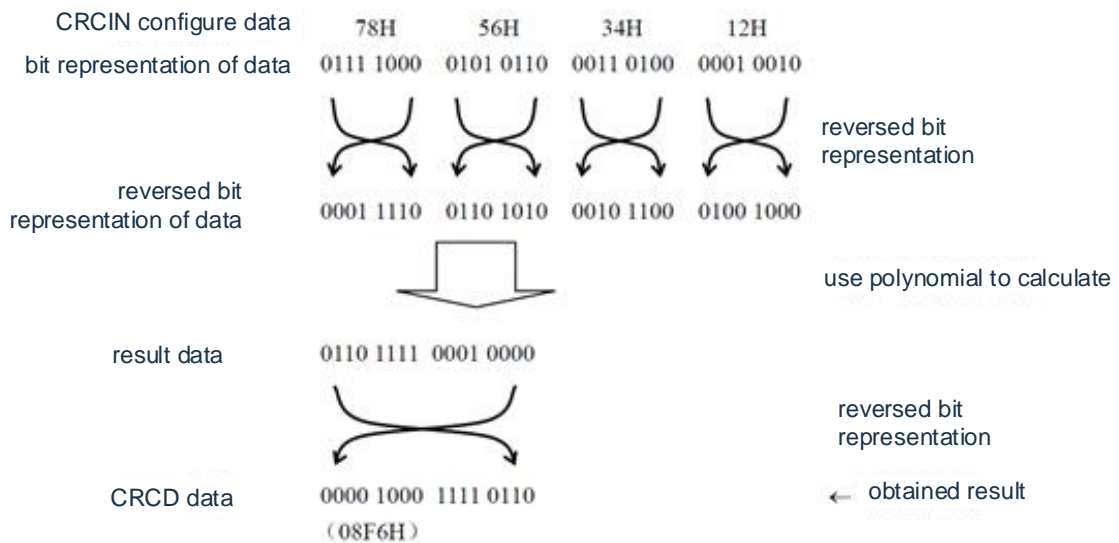
32.3.2 CRC operation function (universal CRC).

In order to ensure safety during operation, the IEC61508 standard requires that the data need to be confirmed even during CPU operation.

This universal CRC can perform CRC operations as a peripheral function during CPU operation. The universal 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.



Note that 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.

32.3.2.1 CRC Input Register (CRCIN).

This is the 8-bit register that sets the CRC calculation data for the universal CRC. The range that can be set is "00H~FFH".

The CRCN register is set via the 8-bit memory manipulation instruction. After generating a reset signal, the value of this register changes to "00H".

Figure 32-4 CRC Input Register (CRCIN).

Address: 400433ACH after reset: 00HR/W

Symbol	7	6	5	4	3	2	1	0
CRCIN								

bit7~0	function
00H~FFH	Data entry

32.3.2.2 CRC Data Register (CRCD).

This is the register that holds the results of the universal 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 via the 16-bit memory manipulation instruction.

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

Figure 32-5 CRC Data Register (CRCD).

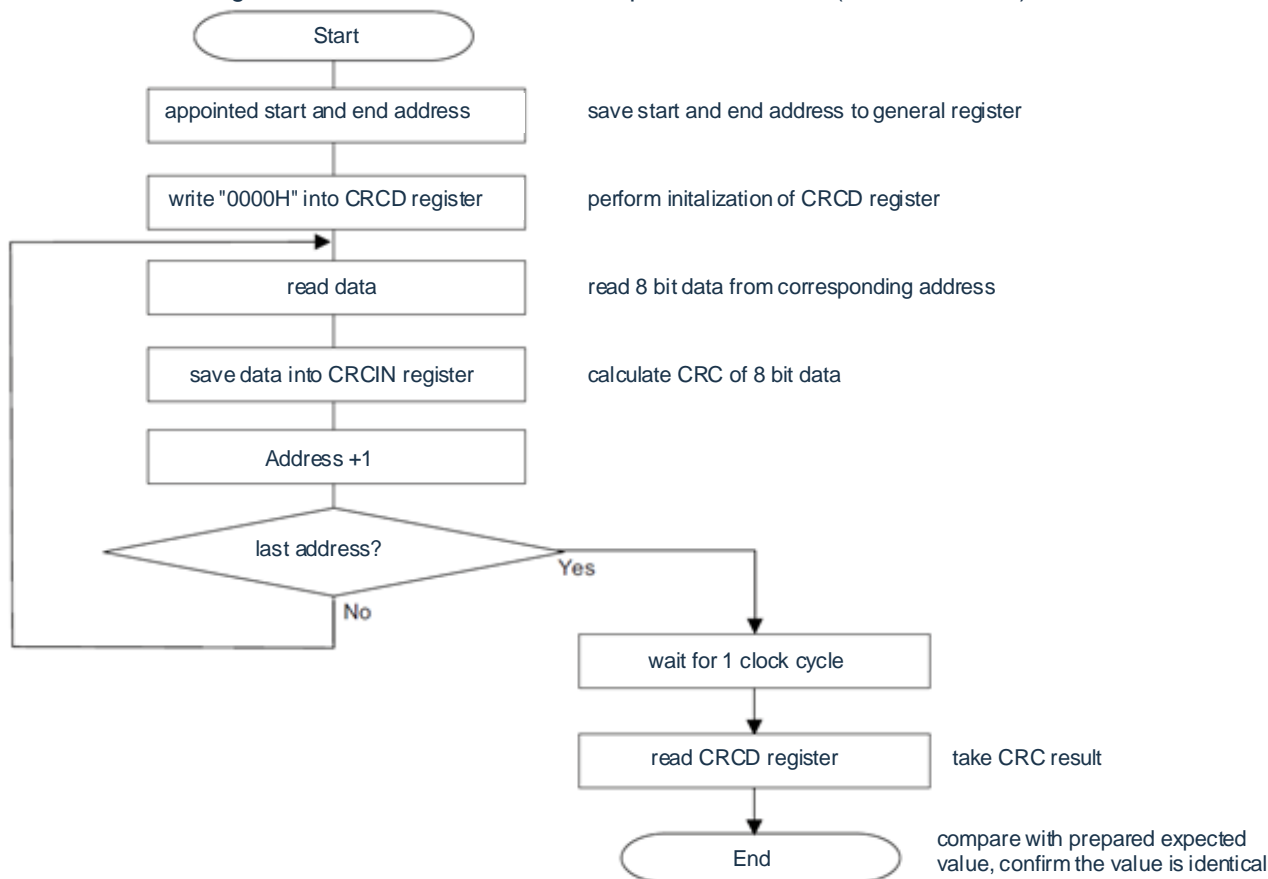
Address: 400432FAH		After reset: 0000H		R/W												
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCD																

Note 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 >

Figure 32-6 Flowchart of the CRC operation function (Universal CRC).



32.3.3 RAM parity error detection function

The IEC60730 standard requires validation of RAM data. Therefore, the BAT32A239'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.

32.3.3.1 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 via the 8-bit memory operation instruction. After generating a reset signal, the value of this register changes to "00H".

Figure 32-7 Ram parity error control register (RPECTL) format

Address: 40020425H after reset: 00HR/W

symbol	7	6	5	4	3	2	1	0
RPECTL	RPERDIS	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.

Note Attach parity bits when writing data, and check parity bits when reading data.

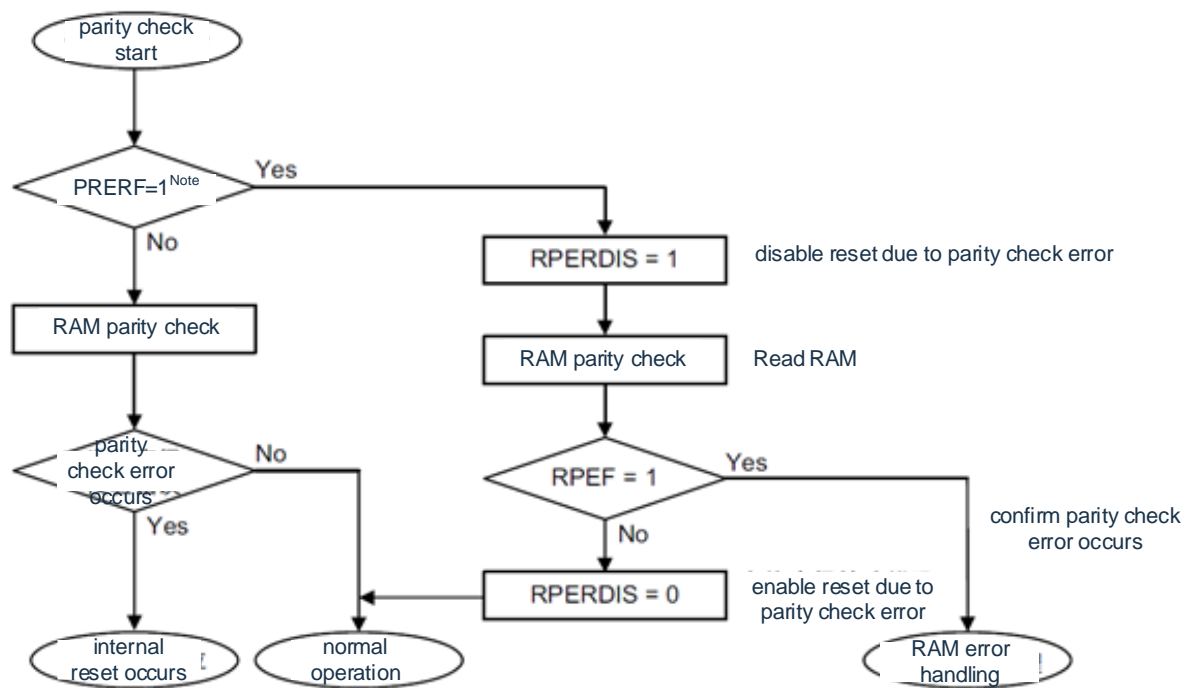
Therefore, to allow ram parity error reset (RPERDIS=0) to be generated, the "RAM region used" must be initialized when the data is accessed and before reading the data.

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.

Note 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 universal registers.

Figure 32-8 Ram parity process



Note For confirmation of internal reset of RAM parity errors, please refer to "Chapter 28 Reset Function".

32.3.4 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.

32.3.4.1 SFR Protects Control Registers (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 via the 8-bit memory operation instruction.

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

Figure 32-9 Format of the SFR Protection Control Register (SFRGD).

Address: 40040478H after reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
SFRGD	0	0	0	0	GCOMP	GAfterRT	GINT	GCSC
GCOMP		The comparator function controls the protection of registers						
0		Void. A control register capable of reading and writing comparator functions.						
1		Effective. The control register for the port function is invalid and can be read. [Protected SFR] COMPPMDR, COMPFIR, COMPOCR, CVRCTL, CxRVM, PGAxCTL, PGASHMD, CMPSELx						
GPORT		Port function for control register protection						
0		Void. A control register capable of reading and writing port functions.						
1		Effective. The control register for the port function is invalid and can be read. [Protected SFR] PMxx, PUxx, PIMxx, POMxx, PMCxx, ADPC, PIORx note						
GINT		Register protection of the interrupt function						
0		Void. Control register capable of reading and writing interrupt functions.						
1		Effective. The interrupt function's control register has an invalid write operation and can be read. [Protected SFR] IFxx, MKxx, PRxx, EGPx, EGNx						
GCSC		Protection of control registers for clock control functions, voltage detection circuitry, and RAM parity error detection functions						
0		Void. A control register capable of reading and writing clock control functions, voltage detection circuits, and RAM parity error detection functions.						
1		Effective. The control registers of the clock control function, voltage detection circuitry, and RAM parity error detection function are invalid and can be read. [Protected SFR] CMC, CSC, OSTs, CKC, PERx, OSMC, LVIM, LVIS, RPECTL						

Note Pxx (port registers) are not protected.

32.3.5 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 Timer4.

However, if a certain clock or 2 clocks stop oscillating, it is impossible to judge the ratio relationship of the 2 clocks.

< the clock > to compare

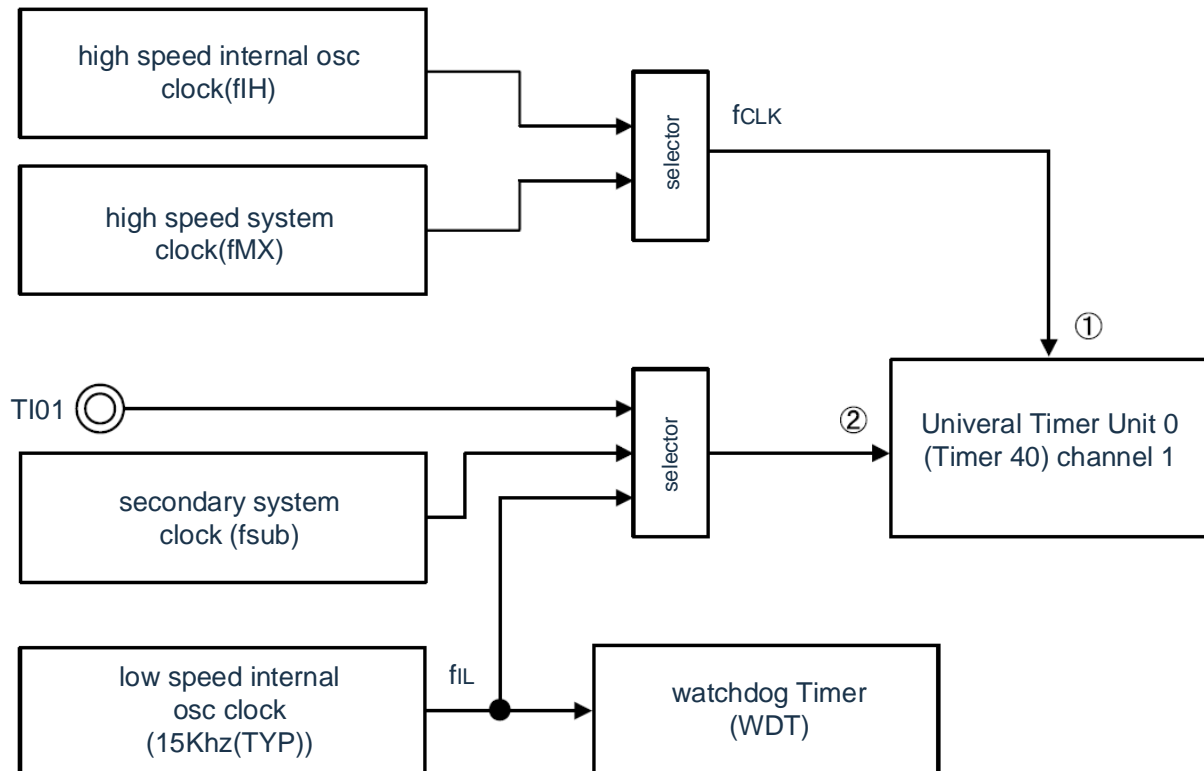
(1) CPU/peripheral hardware clock frequency (f_{CLK}):

- High speed internal oscillator clock (f_{IH}).
- High speed system clock (f_{MX}).

(2) Channel 1 input of Timer4:

- Timer input (TI01) for channel 1
- Low speed internal oscillator clock (f_{IL} : 15kHz (TYP.).)
- Sub-System Clock (f_{SUB}) ^{Note}

Figure 32-10 frequency detection function



When the measurement 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, please refer to "6.8.4 Operation as input pulse interval measurement".

Note Only products with built-in subsystem clocks can be selected.

32.3.5.1 The timer input selects register 0 (TIS0).

For a register description, refer to Section 6.3.8.

32.3.6 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.

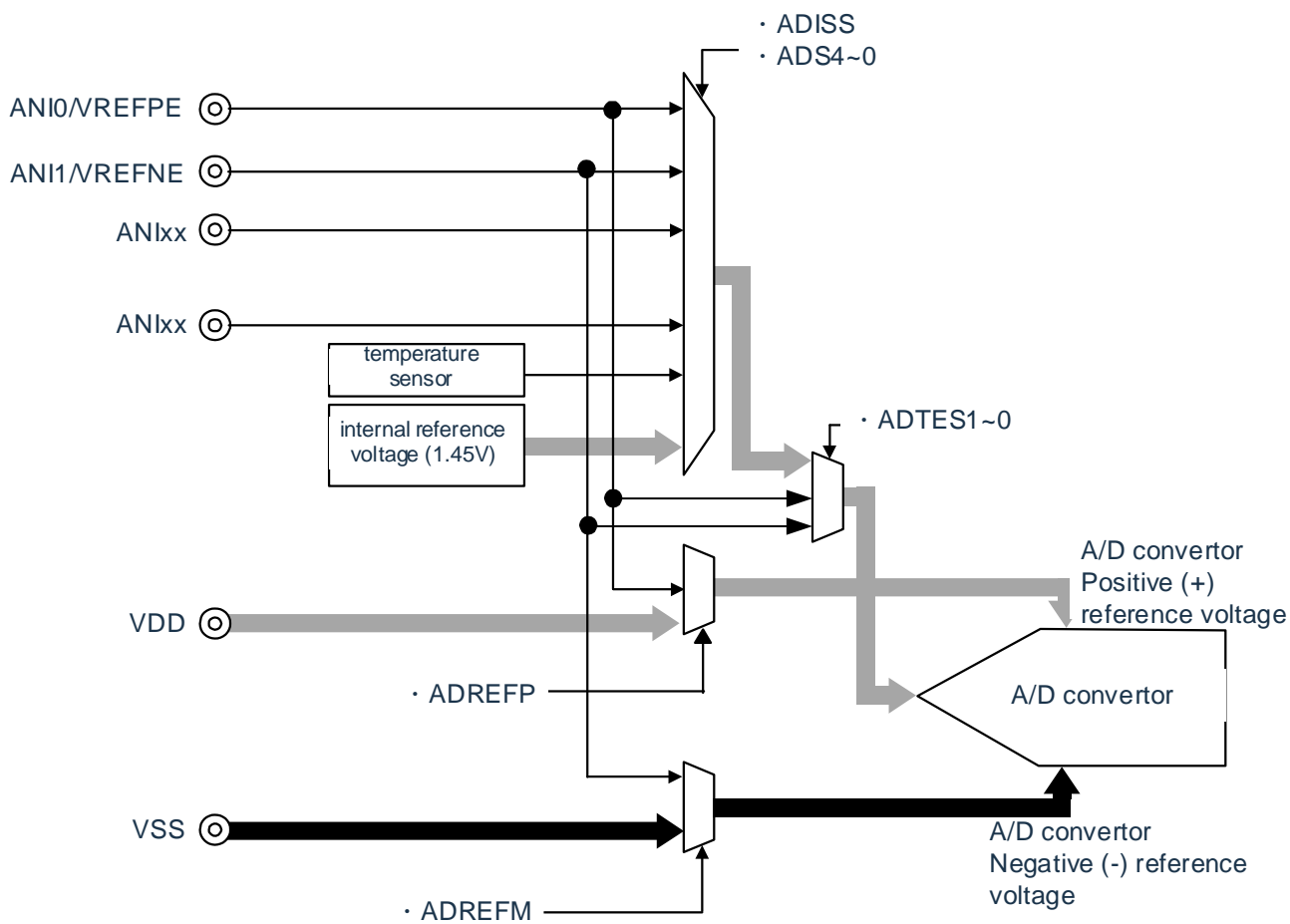
The analog multiplexer can be confirmed by following these steps:

- 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.

Note 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.

Figure 32-11 Structure of the A/D test function



32.3.6.1 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 a register description, refer to 15.2 10.

32.3.6.2 Analog input channel specified 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 a register description, refer to 15.2.7.

32.3.7 Digital output signal level detection function on input/output pins

The IEC60730 standard requires confirmation that the I/O function is normal.

Digital Output Signal Level Detection on Input/Output Pins Reads the Digital Output Level of the Pin when the Pin is in Output Mode.

32.3.7.1 Port Mode Select Register (PMS).

This register selects whether to read the value of the output latch of the port or the output level of the read pin when the pin is output mode (the PMmn bit of the port mode register (PMm) is "0").

Set the PMS registers via the 8-bit memory operation instructions.

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

Figure 32-12 format of port mode selection register (PMS).

Address: 4004087BH after reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
PMS	0	0	0	0	0	0	0	PMS0

PMS0	Selection of data to read when the pin is in output mode
0	Read the value of the Pmn register.
1	Read the digital output level of the pin.

Note 1 For pins that use the pulse output forced cutoff function of timer M to turn the pin into a high impedance state, if the digital output level of the pin is read, the read value is "0".

Note m = 0 to 7, 12 to 14
n=0~7

32.3.8 Product unique identifier registers

Product unique identifiers are ideal 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_084C

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

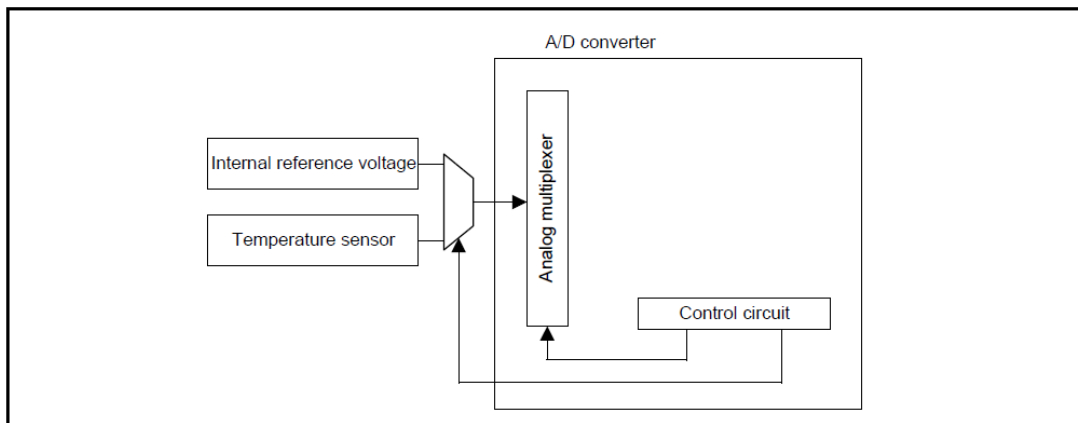
U_ID[127:96]

Chapter 33 Temperature sensor and internal reference voltage

33.1 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. Figure33-1 shows a block diagram of a temperature sensor.

Figure33-1 Temperature Sensor Block Diagram



33.2 Registers for temperature sensors

33.2.1 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 sensors¹ are automatically loaded when powered on or reset is initiated, with each chip having its own calibration data.

33.2.2 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 sensors², loaded automatically when powered on or reset started, with each chip having its own calibration data.

33.3 Instructions for using the temperature sensor

33.3.1 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

33.3.2 How to use the 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:

$$T = (V_s - V_1) / \text{slope} + 25 [^{\circ}\text{C}]$$

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:

$$T = (V_s - V_1) / \text{slope} + 25 [^{\circ}\text{C}]$$

Note: This method produces a lower temperature than the accuracy measured in method one.

33.4 Internal reference voltage

The Band-gap reference voltage (VBG) is an internal fixed reference voltage that is not affected by an external power supply. The VBG output can be converted by connecting internally to the ADC, so the VDD can be calculated from the ADC conversion result of the VBG.

Due to the process, the VBG of each chip is slightly different, so the calculation of VDD will be a little biased. This product has built-in A/D conversion result of VDD=3.0v when VDD=3.0v is built into the factory, and users can accurately calculate it through this value and the A/D conversion result of the current VBG. The voltage value of VDD.

33.4.1 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 for recording VDD=3. The A/D conversion result of VBG at 0v is automatically loaded when power is turned on or reset is started, and each chip has its own calibration data.

33.4.2 Instructions for using the internal reference voltage

In this product, the VDD CDR register stores the internal reference voltage conversion values measured at Ta=Tj=25°C and VDD=3.0v. Using this result, the user can calculate the current VDD voltage according to the following formula:

$$VDD = 3.0V * VDDCDR / CALVBG$$

CALVBG: Using an A/D converter, using VBG as the ADC input channel, the current VBG conversion result is obtained.

Chapter 34 Option bytes

34.1 Option byte functionality

BAT32A2x9 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.

34.1.1 User option bytes (000C0H~000C2H).

(1) 000C0H

- Operation of the 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
- The setting of the interval interrupt of the watchdog timer
 - Use or not use interval interrupts.

(2) 000C1H

- Setting of LVD operating mode
 - Interrupt & Reset mode
 - Reset mode
 - Break mode
 - LVD is OFF (using the external reset input of the RESETB pin).
- Setting of LVD detection level (VLVDH, VLVDL, VLVD).

Note 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, 48MHz, 64MHz.

34.1.2 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.

34.2 The format of the user option bytes

Figure 34-1 user option bytes (000C0H).

Address: 000C0H

Symbol 7 6 5 4 3 2 1 0

WDTINT	WINDOW1	WINDOW0	WDTON	WDCS2	WDCS1	WDCS0	WDSTBYON
--------	---------	---------	-------	-------	-------	-------	----------

WDTINT	The interval of the watchdog timer interrupts the use/non-use
0	Interval interrupts are not used.
1	When 75% of the overflow time is reached + $1/2f_{IL}$, an interval interrupt is generated.

WINDOW1	WINDOW0	The watchdog timer opens during ^{note 1}
0	-	Disable settings.
1	0	75%
1	1	100%

WDTON	Counter run control of the watchdog timer
0	Disables the operation of the counter (stops counting after de-reset).
1	Allow the counter to run (start counting after de-reset).

WDCS2	WDCS1	WDCS0	Overflow time of the watchdog timer ($f_{IL}=20\text{KHz}(\text{MAX.})$)
0	0	0	$2^6/f_{IL}$ (3.2ms)
0	0	1	$2^7/f_{IL}$ (6.4ms)
0	1	0	$2^8/f_{IL}$ (12.8ms)
0	1	1	$2^9/f_{IL}$ (25.6ms)
1	0	0	$2^{11}/f_{IL}$ (102.4ms)
1	0	1	$2^{13}/f_{IL}$ (409.6ms)
1	1	0	$2^{14}/f_{IL}$ (819.2ms)
1	1	1	$2^{16}/f_{IL}$ (3276.8ms)

WDSTBYON	Counter run control (sleep mode) of the watchdog timer
0	In sleep mode, stop the counter from running ^{Note 2} .
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.

Note f_{IL} : The clock frequency of the low-speed internal oscillator

Figure 34-2 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			The config value of the option byte						
INLVDH		INLVDL	VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode settings	
rising	falling	falling						LVIMDS1	LVIMDS0
2.09V	2.04V	1.84V	0	0	1	0	1	1	0
3.13V	3.06V					0	0		
2.61V	2.55V	1		0	1	0			
2.71V	2.65V				0	1			
3.75V	3.67V				0	0			
2.92V	2.86V	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.					

Note that you must write "1" to bit4.

Note 1 For details on LVD circuits, refer to Chapter 32 Voltage Detection Circuits.

2. The detection voltage is TYP Value. For details, please refer to the LVD circuit characteristics in the data sheet.

Figure 34-2 Format of user option bytes (000C1H) (2/4).

Address: 000C1H

76543210

VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0
-------	-------	-------	---	-------	-------	---------	---------

- LVD setting (reset mode).

Detect voltage		The config value of the option byte						
VLVD		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode settings	
rising	falling						LVIMDS1	LVIMDS0
2.09V	2.04V	0	0	1	0	1	1	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.						

Note that you must write "1" to bit4.

Note 1 For details on LVD circuits, refer to Chapter 32 Voltage Detection Circuits.

2. The detection voltage is TYP Value. For details, please refer to the LVD circuit characteristics in the data sheet.

Figure 34-2 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		The config value of the option byte						
VLVD		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode settings	
rising	falling						LVIMDS1	LVIMDS0
2.09V	2.04V	0	0	1	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.						

Note that you must write "1" to bit4.

Note 1 For details on LVD circuits, refer to Chapter 32 Voltage Detection Circuits.

2. The detection voltage is TYP Value. For details, please refer to the LVD circuit characteristics in the data sheet.

Figure 34-2 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

• LVD configuration (Interrupt mode)

Detect voltage		The config value of the option byte						
$V_{LCEO H}$		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode settings	
rising	falling						LVIMDS1	LVIMDS0
—	—	1	x	x	x	x	x	1
—		Setting values other than those described above is prohibited.						

Note 1 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).

Note 1 x: Ignore

- For details of LVD circuits, please refer to "Chapter 31 Voltage Detection Circuits".
- The detection voltage is TYP Value. For details, please refer to the LVD circuit characteristics in the data sheet.

Figure 34-3 user option bytes (000C2H).

Address: 000C2H

7	6	5	4	3	2	1	0
1	1	1	FRQSEL4	FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0

FRQSEL4	FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0	The clock frequency of the high-speed internal oscillator	
					f_{HOCO}	f_{IH}
1	1	0	0	0	64MHz	64MHz
1	0	0	0	0	48MHz	48MHz
0	1	0	0	0	32MHz	32MHz
0	0	0	0	0	24MHz	24MHz
0	1	0	0	1	32MHz	16MHz
0	0	0	0	1	24MHz	12MHz
0	1	0	1	0	32MHz	8MHz
0	0	0	1	0	24MHz	6MHz
0	1	0	1	1	32MHz	4MHz
0	0	0	1	1	24MHz	3MHz
0	1	1	0	0	32MHz	2MHz
0	1	1	0	1	32MHz	1MHz
Other than the above					Disable settings.	

Note 1 You must write "1" to bit7 to 5.

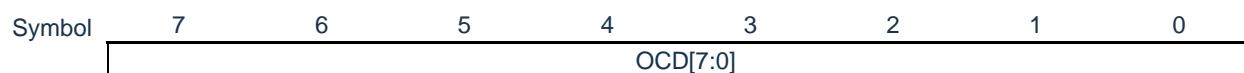
2. The operating frequency range and working voltage range vary according to the operating modes of the flash memory. For details, please refer to the AC characteristics of the data sheet.

34.3 The format of the flash data protection option bytes

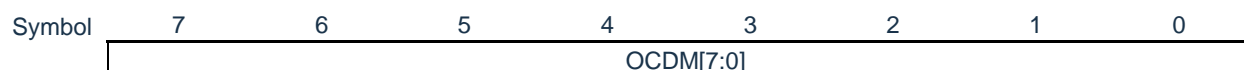
The format of the flash data protection option bytes is as follows.

Figure 34-4 Flash Data Protection Option Bytes (000C3H).

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.
Except for the above note		Allows read/write/erase operations on flash data via debugger

Chapter 35 FLASH control

Note: BAT32A239 products cannot use the PLL output as the system clock of the MCU during Flash scrubbing operations. BAT32A279 products do not have this limitation, and the PLL can be selected as the system clock of the MCU during Flash scrubbing operations.

35.1 FLASH control function description

The BAT32A239 product contains a 256KByte capacity flash memory, divided into 512 Selectors, each with a capacity of 512 Bytes. The BAT32A279 product contains a 512KByte capacity flash memory, divided into 512 Selectors, each with a capacity of 1Kbyte. Can be used as a program memory, a data memory. This module supports erasure, programming, and read operations of the memory. In addition, this module supports write protection for FLASH memory erasure and write protection for control registers.

35.2 FLASH memory structure

Memory structure of BAT32A239

FFFF_FFFFH	reserved
E00F_FFFFH	Cortex-M0+ specific resource region for peripherals
E000_0000H	reserved
4005_FFFFH	resource region for peripherals
4000_0000H	reserved
2000_3FFFFH	SRAM(max 32KB)
2000_0000H	reserved
0050_0BFFFH	data flash 2.5KB
0050_0200H	reserved
0003_FFFFH	main flash region (max 256KB)
0000_0000H	

The memory structure of BAT32A279

FFFF_FFFFH	reserved
E00F_FFFFH	Cortex-M0+ specific resource region for peripherals
E000_0000H	reserved
4005_FFFFH	resource region for peripherals
4000_0000H	reserved
2000_7FFFFH	SRAM(max 64KB)
2000_0000H	reserved
0050_5FFFFH	data flash 20KB
0050_1000H	reserved
0007_FFFFH	main flash region (max 512KB)
0000_0000H	

35.3 Controls registers of F LASH

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 Page Erase Time Control Register (FLSERCNT).
- Flash Write Time Control Register (FLPROCNT).
- Flash mode time control register (FLNVSCNT/FLPRVCNT/FLERVCNT).
- Flash Wipe And Write Protection Control Register (FLSECPR).

35.3.1 Flash Write Protection Register (FLPROT).

Flash protection registers are used to protect the FLASH operating control registers.

Address: 0x40020020 After reset: 00000000H R/W

symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLPROT	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	-	-	-	PRKEY[7:1]						WRP

WRP	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
Except for the above ^{note}	Rewriting WRP is not allowed

35.3.2 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

symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLOPMD1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	-	-	-	FLOPMD1[7:0]						

Address: 0x40020008 reset: 00H R/W

symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLOPMD2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	-	-	-	FLOPMD2[7:0]						

FLOPMD1	FLOPMD2	OPERATE
55	AA	Erase
AA	55	write
00	00	Read out
Except for the above note		Set Prohibit

35.3.3 Flash Erase Control Register (FLERMD).

Flash erase control register to set the type of FLASH erase operation.

Address: 0x4002000C 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, no hardware verification after erasure
1	0	sector erase, hardware validation after erasure
0	1	chip erasure note
1	1	Set Prohibit

Note: Chip Erase only erases the code flash area, not the data flash area. And chip wipe does not support hardware validation.

35.3.4 Flash Status Register (FLSTS).

The status register allows you to query the status of the FLASH controller.

Address: 0x40020000 reset: 00H

R/W

Symbol	7	6	5	4	3	2	1	0
FLSTS	0	0	0	0	0	EVF Note	0	OVF Note

OVF	Flash wipe operation finished flag
0	Flash wipe operation incomplete
1	The FLASH wipe operation is complete

Note: OVF requires the software to write "1" to clear. If not cleared, the next erase and write operation cannot be performed.

EVF	FLASH erases the hardware validation error flag
0	After flash wipe, no errors occur in the hardware check
1	After flash wipe, an error occurred in the hardware check

Note: EVF requires software to write "1" to clear.

35.3.5 Flash full-chip erase time control register (FLCERCNT).

The FLCERCNT register allows you to set the time for flash full-slice erase.

Address: 0x40020010 After reset: indefinite R/W

symbol

FLCERCNT

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
load	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	FLCERCNT[9:0]									

Load	Selection of erase time settings Note
0	Use the hardware-set erase time
1	Use the software-set erase time (F-LCERCNT [9:0]).

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 FLCERCNT.

FLCERCNT[9:0]	Software erase time setting
Chip erase time = $(\text{CERCNT} \times 2048 \times T_{\text{fclk}})$, which meets the hardware requirements of $> 20\text{ms}$	

35.3.6 Flash Page Erase Time Control Register (FLSERCNT).

The FLSERCNT register allows you to set the time for flash full-slice erase.

Address: 0x40020014 after reset: indefinite R/W

symbol

FLSERCNT

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
load	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	FLSERCNT[9:0]									

Load	Selection of erase time settings Note
0	Use the hardware-set erase time
1	Use the software-set erase time (FLSERCNT [9:0]).

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 FLSERCNT.

FLSERCNT[9:0]	Software erase time setting
sector erase time = (SERCNT*256*Tfclk), which meets the hardware requirements of $>4\text{ms}$	

35.3.7 Flash Write Time Control Register (FLPROCNT).

The FLPROCNT register allows you to set the time of flash WORD writes.

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 = (PROCNT*4*Tfclk), which meets the hardware requirements 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 = (PGSCNT*Tfclk), which meets the hardware requirements of $>5\mu\text{s}$	

35.3.8 Flash Wipe And Write Protection Control Register (FLSECPR).

When a Selector is protected, the wipe and write operations made to the Selector are invalid.

Address: 0x40020210 After reset: 00000000H R/W

symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLSECPR	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]				

KEY	Register SECPR write protection
5AA5F1	Rewriting of SECPR [3:0] is allowed
Other than the	Rewriting of SECPR [3:0] is not permitted

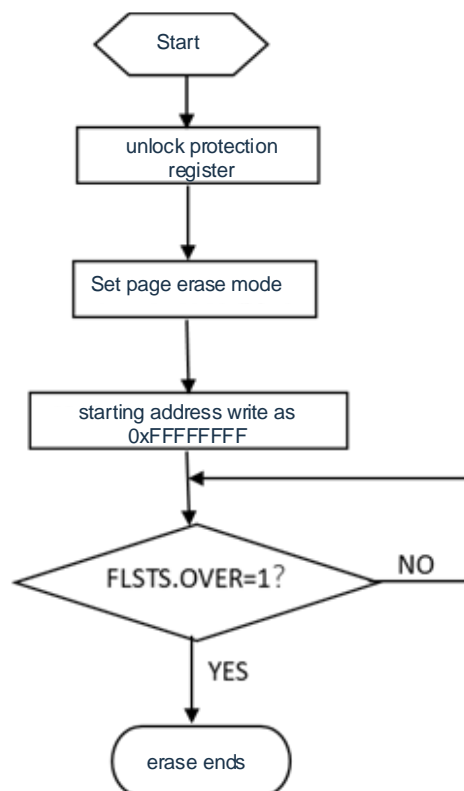
SECPR	Register SECPR write protection
0001	00_0000H~00_0FFFFH cannot be erased
0010	00_0000H~00_1FFFFH cannot be erased
0011	00_0000H~00_3FFFFH cannot be erased
0100	00_0000H~00_7FFFFH cannot be erased
0101	00_0000H~00_FFFFFH cannot be erased
0110	00_0000H~01_FFFFFH cannot be erased
0111	00_0000H~03_FFFFFH cannot be erased
1000	00_0000H~05_FFFFFH cannot be erased
1001	00_0000H~07_FFFFFH cannot be erased (all 512 sectors are not allowed to erase and write).
Other than the	sector is not protected and allows erasure and write

35.4 Flash operation method

35.4.1 Sector erase

sector erase, the erase time is implemented by hardware and can also be configured via FLSERCNT. The operation process is as follows:

- 1) Set up FLERMD. ERMD0 is 1'b0, select the sector erase mode, and set the value of ERMD 1 according to whether hardware verification is required;
- 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) If hardware verification after erase (ERMD1=1) is set, FLSTS.EV F can be determined by the software to check whether the verification is correct.
- 6) Before proceeding to the next operation, the software sets "1" to clear FLSTS.



35.4.2 Chip erase

chip erase, the erase time is 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.

35.4.3 Word program

word programming, write time is implemented by hardware and can also be configured via PROCNT. The operation process is as follows:

- 1) Set FLPROT to 0xF1 to de-protect FLOPMD. Then set FLOPMD1 to 0xAA and FLOPMD2 to 0x55
- 2) Writes the appropriate data to the destination address.
- 3) Software query status register FLSTS. OVF, when OVF=1, indicates that the write operation is complete.
- 4) Before proceeding to the next operation, the software sets "1" to clear FLSTS.

35.5 Flash read

The fastest finger frequency supported by flash built into this device is 32 MHz. When the HCLK frequency exceeds 32MHz, the hardware inserts a 1 wait period when the CPU accesses flash.

35.6 Considerations for FLASH operations

- 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 wipe and write 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.

Appendix Revision Record

Version	Date	Revisions
V1.00	May 2022	The first edition was made