Cmsemicon®

BAT32G157 User Manual

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

Rev.1.0.5

Please note the following CMS IP policy

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

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

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

Documentation Instructions

This manual is the technical reference manual for the BAT32G157 microcontroller product. The technical reference manual is the application instruction material on how to use this series of products, including the structure, function description, working mode and register configuration of each functional module.

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

The data sheet information is as follows:

BAT32G157xx: BAT32G157_datasheet_vx.x.x. pdf

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

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

Chapter 1 CPU

1.1 Overview

This Chapter provides a brief introduction to the features and debugging features of the ARM Cortex-M0+ kernel on which this product is built. Please refer to the ARM documentation for details.

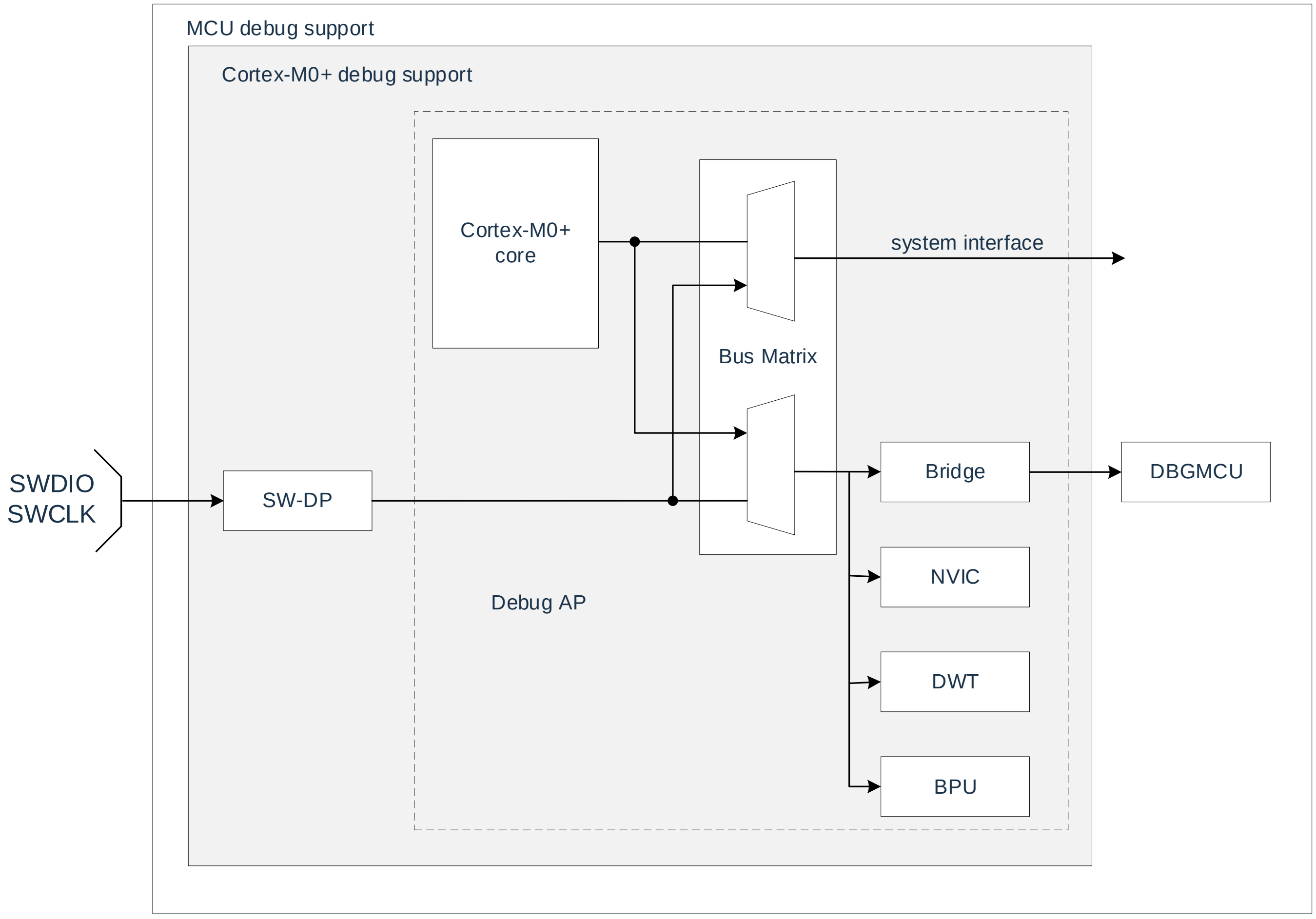
1.2 Cortex-M0+ core features

- ARM Cortex-M0+ processors are 32-bit RISC cores with a 2-stage pipeline that only supports privileged modes
- 32 cycle hardware multiplier
- Nested Vector Interrupt Controller (NVIC)
  - 1 unshielded interrupt (NMI)
  - Supports 32 masking interrupt requests (IRQs)
  - 4 interrupt priority
- The system timer SysTick is a 24-bit countdown timer that can be selected for fCLK or fIL count clocks
- Vector table offset register (VTOR)
  - The software can write VTOR to relocate the start address of the vector table to a different location
  - The default value for this register is 0x0000_0000, with low 8-bit write ignore, read to zero, that is, offset 256 bytes aligned.

1.3 Debug features

- 2-wire SWD debug interface
- Support for pausing, resuming, and single-step execution procedures
- Access processor's kernel register and special function register
- 4 hardware breakpoints (BPU)
- Unlimited software breakpoints (BKPT instruction)
- 2 data observation points (DWT)
- Access memory during kernel execution

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



Note: SWD does not work in deep sleep mode, please debug in active and sleep mode.

1.4 SWD interface pin

2 GPIO of the product can be used as SWD interface pins that are present in all packages.

Table 1-1 SWD debug port pin

SWD port name	Debug function	Pin assignment
SWCLK	Serial clock	PB03
SWDIO	Serial Data Input/Output	PH01

When you do not use the SWD feature, you can disable the SWD by setting the debug stop control register (DBGSTOPCR).

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

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

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

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

SWDIS	SWD debug interface disable
0	SWD debug interface enable. PH01 cannot be used as a GPIO in the state where the debugger is connected (because the ENO and DOUT for that IOBUF are now controlled by the debugger)
1	The SWD debug interface is disabled. PH01 can be used as GPIO

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

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

Note 1: The timer system peripheral module of the product includes: Universal timer unit Timer4/8

Note 2: The communication system peripheral module of this product includes: Communication Serial Communication Unit, Serial IICA

1.5 ARM reference document

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

For 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 Port Function

2.1 Port function

Refer to datasheet for each product family.

2.2 Port multiplexing function

This product supports up to 60 GPIOs, in order to facilitate the configuration of the multiplexing function, these 60 ports are divided into GRP0, GRP1 and GRP2 three groups, each group in addition to the default multiplexing function, can also be redirected in the group part of the multiplexing function. For the grouping method, refer to Table 2-1 Port Grouping Method.

In addition to the default multiplexing function, the 20 ports in GRP0 can arbitrarily redirect the multiplexing function of channel 0 ~ 3 of the universal timer TIMER4, serial interface UART0 and serial interface IICA0.

In addition to the default multiplexing function, the 20 ports in GRP1 can arbitrarily redirect the multiplexing function of channel 0 ~ 3 of the general-purpose timer TIMER8, serial interface UART1 and high-speed SPI serial port SPIHS0.

In addition to the default multiplexing function, the 20 ports in GRP2 can arbitrarily redirect the multiplexing function of the universal timer TIMER8 channel 4 ~ 7, serial interface UART2, serial interface IICA1 and buzzer output CLKBUZ1.

For details on the configuration of each multiplexing function, please refer to 2.3.10 Port output multiplexing function configuration register (PxxCFG), 2.3.11 Port input multiplexing function configuration register, 2.5 Register settings when using the multiplexing function.

Table 2-1 Port grouping method

Serial number	GRP0	GRP1	GRP2
0	PB00	PC03	PB01
1	PH4	PC04	PB02
2	PH3	PC05	PB03
3	PH2	PC06	PB04
4	PH1	PC07	PB05
5	PC14	PC12	PB06
6	PC15	PC13	PB07
7	PC08	PA04	PB08
8	PC09	PA05	PC00
9	PC10	PA06	PC01
10	PC11	PA07	PC02
11	PA00	PA08	PA11
12	PA01	PA09	PA12
13	PA02	PA10	PA13
14	PA03	PD00	PA14
15	PD07	PD01	PD02
16	PD08	PD12	PD03
17	PD09	PD13	PD04
18	PD10	PD14	PD05
19	PD11	PD15	PD06

2.3 Registers for controlling port function

The port is controlled through the following registers.

- Port mode register (PMxx)
- Port register (Pxx)
- Pull-up resistor selection register (PUxx)
- Pull-down resistor selection register (PDxx)
- Port output mode register (POMx)
- Port mode control register (PMCxx)
- Port set control register (PSETxx)
- Port clear control register (PCLRxx)
- Port status readback register (PREADxx)
- Port output multiplexing function configuration register (PxxCFG, see 2.3.10)
- Port input multiplexing function configuration register (see 2.3.11 for details)
- External interrupt port selection register (INTPnPCFG, n=0~7)

Note: The assigned registers and bits vary by product. For registers and bits assigned to each product, refer to Table 2-.

Unassigned bits must be initialized.

Table 2-2 PMxx, Pxx, PSETxx, PCLRxx, PUxx, PDxx, POMxx, PREADxx, PMCxx registers and bits

Port		Bit name									64 pin (-S) Note1	64 pin	48 pin (-S) Note1	48 pin
		PMxx register	Pxx register	PSETxx register	PCLRxx register	PUxx register	PDxx register	POMxx register	PREADxx register	PMCxx register				
Port A	00	PMA0	PA0	PSETA0	PCLRA0	PUA0	PDA0	POMA0	PREADA0	PMCA0	○	○	○	○
	01	PMA1	PA1	PSETA1	PCLRA1	PUA1	PDA1	POMA1	PREADA1	PMCA1	○	○	○	○
	02	PMA2	PA2	PSETA2	PCLRA2	PUA2	PDA2	POMA2	PREADA2	PMCA2	○	○	○	○
	03	PMA3	PA3	PSETA3	PCLRA3	PUA3	PDA3	POMA3	PREADA3	PMCA3	○	○	○	○
	04	PMA4	PA4	PSETA4	PCLRA4	PUA4	PDA4	POMA4	PREADA4	PMCA4	○	○	○	○
	05	PMA5	PA5	PSETA5	PCLRA5	PUA5	PDA5	POMA5	PREADA5	PMCA5	○	○	○	○
	06	PMA6	PA6	PSETA6	PCLRA6	PUA6	PDA6	POMA6	PREADA6	PMCA6	○	○	○	○
	07	PMA7	PA7	PSETA7	PCLRA7	-	-	POMA7	PREADA7	-	○	○	○	○
	08	PMA8	PA8	PSETA8	PCLRA8	-	-	POMA8	PREADA8	-	○	○	○	○
	09	PMA9	PA9	PSETA9	PCLRA9	-	PDA9	POMA9	PREADA9	-	-	○	-	○
	10	PMA10	PA10	PSETA10	PCLRA10	PUA10	PDA10	POMA10	PREADA10	-	○	○	○	○
	11	PMA11	PA11	PSETA11	PCLRA11	PUA11	PDA11	POMA11	PREADA11	PMCA11	○	○	-	-
	12	PMA12	PA12	PSETA12	PCLRA12	PUA12	PDA12	POMA12	PREADA12	PMCA12	○	○	-	-
	13	PMA13	PA13	PSETA13	PCLRA13	PUA13	PDA13	POMA13	PREADA13	PMCA13	○	○	-	-
	14	PMA14	PA14	PSETA14	PCLRA14	PUA14	PDA14	POMA14	PREADA14	PMCA14	○	○	-	-

Note: 1. (-S) indicates that it is limited to BAT32G157xx-S series products.

2. PA07 is used as USB-DM by default, and can also be configured as a GPIO port, the configuration method refers to 2.3.13 USB_ DP, USB_DM port configuration registers (PMR, PRCR)

3. PA08 is used as USB-DP by default, and can also be configured as a GPIO port, the configuration method refers to 2.3.13 USB_ DP, USB_DM port configuration registers (PMR, PRCR)

Port		Bit name									64 pin (-S) Note 1	64 pin	48 pin (-S) Note 1	48 pin
		PMxx register	Pxx register	PSETxx register	PCLRxx register	PUxx register	PDxx register	POMxx register	PREADxx register	PMCxx register				
Port B	00	PMB0	PB0	PSETB0	PCLRB0	PUB0	PDB0	POMB0	PREADB0	-	○	○	○	○
	01	PMB1	PB1	PSETB1	PCLRB1	PUB1	PDB1	POMB1	PREADB1	PMCB1	○	○	-	-
	02	PMB2	PB2	PSETB2	PCLRB2	PUB2	PDB2	POMB2	PREADB2	PMCB2	○	○	-	-
	03	PMB3	PB3	PSETB3	PCLRB3	PUB3	PDB3	POMB3	PREADB3	PMCB3	○	○	-	○
	04	PMB4	PB4	PSETB4	PCLRB4	PUB4	PDB4	POMB4	PREADB4	PMCB4	○	○	-	○
	05	PMB5	PB5	PSETB5	PCLRB5	PUB5	PDB5	POMB5	PREADB5	-	-	○	-	○
	06	PMB6	PB6	PSETB6	PCLRB6	PUB6	PDB6	POMB6	PREADB6	PMCB6	○	○	○	○
	07	PMB7	PB7	PSETB7	PCLRB7	PUB7	PDB7	POMB7	PREADB7	PMCB7	○	-	○	-
	08	PMB8	PB8	PSETB8	PCLRB8	PUB8	PDB8	POMB8	PREADB8	PMCB8	○	-	○	-
Port C	00	PMC0	PC0	PSETC0	PCLRC0	PUC0	PDC0	POMC0	PREADC0	PMCC0	○	○	○	○
	01	PMC1	PC1	PSETC1	PCLRC1	PUC1	PDC1	POMC1	PREADC1	PMCC1	○	○	○	○
	02	PMC2	PC2	PSETC2	PCLRC2	PUC2	PDC2	POMC2	PREADC2	PMCC2	○	○	○	○
	03	PMC3	PC3	PSETC3	PCLRC3	PUC3	PDC3	POMC3	PREADC3	PMCC3	○	○	○	○
	04	PMC4	PC4	PSETC4	PCLRC4	PUC4	PDC4	POMC4	PREADC4	PMCC4	○	○	○	○
	05	PMC5	PC5	PSETC5	PCLRC5	PUC5	PDC5	POMC5	PREADC5	PMCC5	○	○	○	○
	06	PMC6	PC6	PSETC6	PCLRC6	PUC6	PDC6	POMC6	PREADC6	PMCC6	○	○	-	○
	07	PMC7	PC7	PSETC7	PCLRC7	PUC7	PDC7	POMC7	PREADC7	PMCC7	○	○	-	○
	08	PMC8	PC8	PSETC8	PCLRC8	PUC8	PDC8	POMC8	PREADC8	PMCC8	○	○	○	○
	09	PMC9	PC9	PSETC9	PCLRC9	PUC9	PDC9	POMC9	PREADC9	PMCC9	○	○	○	○
	10	PMC10	PC10	PSETC10	PCLRC10	PUC10	PDC10	POMC10	PREADC10	PMCC10	○	○	○	○
	11	PMC11	PC11	PSETC11	PCLRC11	PUC11	PDC11	POMC11	PREADC11	PMCC11	○	○	○	○
	12	PMC12	PC12	PSETC12	PCLRC12	PUC12	PDC12	POMC12	PREADC12	-	-	○	-	-
	13	PMC13	PC13	PSETC13	PCLRC13	PUC13	PDC13	POMC13	PREADC13	-	-	○	-	-
	14	PMC14	PC14	PSETC14	PCLRC14	PUC14	PDC14	POMC14	PREADC14	-	-	○	-	-
	15	PMC15	PC15	PSETC15	PCLRC15	PUC15	PDC15	POMC15	PREADC15	-	-	○	-	-

Port		Bit name									64 pin (-A) Note 1	64 pin	48 pin (-A) Note 1	48 pin
		PMxx register	Pxx register	PSETxx register	PCLRxx register	PUxx register	PDxx register	POMxx register	PREADxx register	PMCxx register				
Port D	00	PMD0	PD0	PSETD0	PCLRD0	PUD0	PDD0	POMD0	PREADD0	-	○	○	○	○
	01	PMD1	PD1	PSETD1	PCLRD1	PUD1	PDD1	POMD1	PREADD1	-	○	○	○	○
	02	PMD2	PD2	PSETD2	PCLRD2	PUD2	PDD2	POMD2	PREADD2	-	-	○	-	○
	03	PMD3	PD3	PSETD3	PCLRD3	PUD3	PDD3	POMD3	PREADD3	-	-	○	-	○
	04	PMD4	PD4	PSETD4	PCLRD4	PUD4	PDD4	POMD4	PREADD4	PMCD4	○	○	○	○
	05	PMD5	PD5	PSETD5	PCLRD5	PUD5	PDD5	POMD5	PREADD5	PMCD5	○	○	○	○
	06	PMD6	PD6	PSETD6	PCLRD6	PUD6	PDD6	POMD6	PREADD6	PMCC6	○	○	○	○
	07	PMD7	PD7	PSETD7	PCLRD7	PUD7	PDD7	POMD7	PREADD7	PMCD7	○	○	○	○
	08	PMD8	PD8	PSETD8	PCLRD8	PUD8	PDD8	POMD8	PREADD8	PMCD8	○	○	○	○
	09	PMD9	PD9	PSETD9	PCLRD9	PUD9	PDD9	POMD9	PREADD9	-	○	○	○	○
	10	PMD10	PD10	PSETD10	PCLRD10	PUD10	PDD10	POMD10	PREADD10	-	○	○	○	○
	11	PMD11	PD11	PSETD11	PCLRD11	PUD11	PDD11	POMD11	PREADD11	-	○	○	○	○
	12	PMD12	PD12	PSETD12	PCLRD12	PUD12	PDD12	POMD12	PREADD12	-	○	○	-	-
	13	PMD13	PD13	PSETD13	PCLRD13	PUD13	PDD13	POMD13	PREADD13	-	○	○	-	-
	14	PMD14	PD14	PSETD14	PCLRD14	PUD14	PDD14	POMD14	PREADD14	-	○	○	-	-
	15	PMD15	PD15	PSETD15	PCLRD15	PUD15	PDD15	POMD15	PREADD15	-	○	○	-	-
RESINB/PH00		-	PH0	-	-	PUH0	-	-	PREADH0	-	○	○	○	○
X1/PH01		PMH1	PH1	PSETH1	PCLRH1	PUH1	-	POMH1	PREADH1	-	○	○	○	○
X2/PH02		PMH2	PH2	PSETH2	PCLRH2	PUH2	-	POMH2	PREADH2	-	○	○	○	○
XT1/PH03		PMH3	PH3	PSETH3	PCLRH3	-	-	POMH3	PREADH3	-	○	○	○	○
XT2/PH04		PMH4	PH4	PSETH4	PCLRH4	-	-	POMH4	PREADH4	-	○	○	○	○

2.3.1 Port mode register (PMxx)

When a port is used as a digital channel, this is the register that sets its input/output in bits. After a reset signal is generated, each port defaults to the input state. When using a port for the multiplexing function, it must be set in accordance with “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.

Figure 2-1 Format of port mode register

symbol	15	14	13	12	11	10	9	8	offset addr	after reset	R/W
PMA	1	PMA14	PMA13	PMA12	PMA11	PMA10	PMA9	PMA8	0x010	FFFFH	R/W
	7	6	5	4	3	2	1	0			
	PMA7	PMA6	PMA5	PMA4	PMA3	PMA2	PMA1	PMA0			
PMB	1	1	1	1	1	1	1	PMB8	0x012	FFFFH	R/W
	PMB7	PMB6	PMB5	PMB4	PMB3	PMB2	PMB1	PMB0			
PMC	PMC15	PMC14	PMC13	PMC12	PMC11	PMC10	PMC9	PMC8	0x014	FFFFH	R/W
	PMC7	PMC6	PMC5	PMC4	PMC3	PMC2	PMC1	PMC0			
PMD	PMD15	PMD14	PMD13	PMD12	PMD11	PMD10	PMD9	PMD8	0x016	FFFFH	R/W
	PMD7	PMD6	PMD5	PMD4	PMD3	PMD2	PMD1	PMD0			
PMH	1	1	1	1	1	1	1	1	0x01E	FFFFH	R/W
	1	1	1	PMH4	PMH3	PMH2	PMH1	1*			

*RESINB/PH00 ports have input functions only.

PMmn	Selection of input/output modes for the Pmn port (m=A,B,C,D,H,N=0~15)
0	Output mode (used as output port (output buffer ON)).
1	Input mode (used as input port (output buffer OFF)).

Note: Unassigned bits must be initialized.

2.3.2 Port register (Pxx)

This is the register that sets the value of the port's output latch in bits. Reading this register in input mode yields the port level, and reading the value of the port's output latch in output mode.

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 2-2 Format of port register

symbol	15	14	13	12	11	10	9	8	address	after reset	R/W
PA	0	PA14	PA13	PA12	PA11	PA10	PA9	PA8	0x000	0000H (output latch).	R/W
	7	6	5	4	3	2	1	0			
	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0			
PB	0	0	0	0	0	0	0	PB8	0x002	0000H (output latch).	R/W
	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0			
PC	PC15	PC14	PC13	PC12	PC11	PC10	PC9	PC8	0x004	0000H (output latch).	R/W
	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0			
PD	PD15	PD14	PD13	PD12	PD11	PD10	PD9	PD8	0x006	0000H (output latch).	R/W
	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0			
PH	0	0	0	0	0	0	0	0	0x00E	0001H (output latch).	R/W
	0	0	0	PH4	PH3	PH2	PH1	PH0*			

*Since the RESINB/PH00 port only has input function, the PH0 bit is not writable, and the port level can only be read through this bit.

Pmn	m=A,B,C,D,H, n=0~15	
	Control of output data (output mode)	Reading of input data (input mode)
0	Output "0".	Input low.
1	Output "1".	Input high.

Note: 1. Unassigned bits must be initialized.

2.3.3 Port set control register (PSETxx)

This is the register in bits that sets the port output latch. After a reset signal is generated, the value of these registers becomes "0000H".

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

Figure 2-3: Format of port set control register

symbol	15	14	13	12	11	10	9	8	address	after reset	R/W
PSETA	0	PSETA14	PSETA13	PSETA12	PSETA11	PSETA10	PSETA9	PSETA8	0x060	0000H	W
	7	6	5	4	3	2	1	0			
	PSETA7	PSETA6	PSETA5	PSETA4	PSETA3	PSETA2	PSETA1	PSETA0			
PSETB	0	0	0	0	0	0	0	PSETB8	0x062	0000H	W
	PSETB7	PSETB6	PSETB5	PSETB4	PSETB3	PSETB2	PSETB1	PSETB0			
PSETC	PSETC15	PSETC14	PSETC13	PSETC12	PSETC11	PSETC10	PSETC9	PSETC8	0x064	0000H	W
	PSETC7	PSETC6	PSETC5	PSETC4	PSETC3	PSETC2	PSETC1	PSETC0			
PSETD	PSETD15	PSETD14	PSETD13	PSETD12	PSETD11	PSETD10	PSETD9	PSETD8	0x066	0000H	W
	PSETD7	PSETD6	PSETD5	PSETD4	PSETD3	PSETD2	PSETD1	PSETD0			
PSETH	0	0	0	0	0	0	0	0	0x06E	0000H	W
	0	0	0	PSETH4	PSETH3	PSETH2	PSETH1	0			

PSETmn	Set control of PMN port (m=A,B,C,D,H,n=0~15)
0	No action
1	The corresponding bit of Pmn is set to 1

Note: 1 Unassigned bits must be initialized.

2.3.4 Port clear control register (PCLRxx)

This is the register in bits that sets the port output latch. After a reset signal is generated, the value of the register becomes "0000H".

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

Figure 2-4: Format of port clear control register

symbol	15	14	13	12	11	10	9	8	address	after reset	R/W
PCLRA	0	PCLRA14	PCLRA13	PCLRA12	PCLRA11	PCLRA10	PCLRA9	PCLRA8	0x070	0000H	W
	7	6	5	4	3	2	1	0			
	PCLRA7	PCLRA6	PCLRA5	PCLRA4	PCLRA3	PCLRA2	PCLRA1	PCLRA0			
PCLRB	0	0	0	0	0	0	0	PCLRB8	0x072	0000H	W
	PCLRB7	PCLRB6	PCLRB5	PCLRB4	PCLRB3	PCLRB2	PCLRB1	PCLRB0			
PCLRC	PCLRC15	PCLRC14	PCLRC13	PCLRC12	PCLRC11	PCLRC10	PCLRC9	PCLRC8	0x074	0000H	W
	PCLRC7	PCLRC6	PCLRC5	PCLRC4	PCLRC3	PCLRC2	PCLRC1	PCLRC0			
PCLRD	PCLRD15	PCLRD14	PCLRD13	PCLRD12	PCLRD11	PCLRD10	PCLRD9	PCLRD8	0x076	0000H	W
	PCLRD7	PCLRD6	PCLRD5	PCLRD4	PCLRD3	PCLRD2	PCLRD1	PCLRD0			
PCLRH	0	0	0	0	0	0	0	0	0x07E	0000H	W
	0	0	0	PCLRH4	PCLRH3	PCLRH2	PCLRH1	0			

PCLRmn	Clear control of PMN port (m=A,B,C,D,H,n=0~15)
0	No action
1	The corresponding bit of PMN is cleared

Note: Unassigned bits must be initialized.

2.3.5 Pull-up resistor selection register (PUxx)

Select register for internal pull-up resistors. By setting this register, a port in input mode (PMmn=1) or an N-channel open-drain output mode can be pulled up in bits using an internal pull-up resistor. For ports set to output mode, independent of the setting of the pull-up resistor selection register, no internal pull-up resistors are connected. The same applies when used as an output port for multiplexing functions or when set to analog functions.

After generating the reset signal, the pull-up function of the five ports PA10, PB03, PD00, X1/PH01, RESINB/PH00 is turned on by default (PU A10, PUB3, PUD 0, PUH1, PUH0 reset value is "1"), the pull-up function of other ports is not turned on by default.

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.

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

symbol	15	14	13	12	11	10	9	8	address	after reset	R/W
PUA	0	PUA14	PUA13	PUA12	PUA11	PUA10	0	0	0x020	0400H	R/W
	7	6	5	4	3	2	1	0			
	0	PUA6	PUA5	PUA4	PUA3	PUA2	PUA1	PUA0			
PUB	0	0	0	0	0	0	0	PUB8	0x022	0008H	R/W
	PUB7	PUB6	PUB5	PUB4	PUB3	PUB2	PUB1	PUB0			
PUC	PUC15	PUC14	PUC13	PUC12	PUC11	PUC10	PUC9	PUC8	0x024	0000H	R/W
	PUC7	PUC6	PUC5	PUC4	PUC3	PUC2	PUC1	PUC0			
PUD	PUD15	PUD14	PUD13	PUD12	PUD11	PUD10	PUD9	PUD8	0x026	0001H	R/W
	PUD7	PUD6	PUD5	PUD4	PUD3	PUD2	PUD1	PUD0			
Tel	0	0	0	0	0	0	0	0	0x02E	0003H	R/W
	0	0	0	0	0	PUH2	PUH1	PUH0			

PUmn	Selection of the internal pull-up resistor for the Pmn port (m=A, B, C, D, H, N=0~15)
0	Internal pull-up resistors are not connected.
1	Connect an internal pull-up resistor.

Note: Unassigned bits must be initialized.

2.3.6 Pull-down resistor selection register (PDxx)

Selection register for internal pull-down resistors. By setting this register, ports in input mode (PMmn=1) or N-channel open-drain output mode can be pulled down in bits using internal pull-up resistors. For ports set to output mode, independent of the setting of the pull-down resistor selection register, no internal pull-down resistor is connected. The same applies when used as an output port for multiplexing functions or when set to analog functions.

After a reset signal is generated, the value of these registers becomes "000 0H".

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

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

symbol	15	14	13	12	11	10	9	8	address	after reset	R/W
PDA	0	PDA14	PDA13	PDA12	PDA11	PDA10	PDA9	0	0x030	0000H	R/W
	7	6	5	4	3	2	1	0			
	0	PDA6	PDA5	PDA4	PDA3	PDA2	PDA1	PDA0			
PDB	0	0	0	0	0	0	0	PDB8	0x032	0000 H	R/W
	PDB7	PDB6	PDB5	PDB4	PDB3	PDB2	PDB1	PDB0			
PDC	PDC15	PDC14	PDC13	PDC12	PDC11	PDC10	PDC9	PDC8	0x034	0000H	R/W
	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0			
PDD	PDD15	PDD14	PDD13	PDD12	PDD11	PDD10	PDD9	PDD8	0x036	0000 H	R/W
	PDD7	PDD6	PDD5	PDD4	PDD3	PDD2	PDD1	PDD0			

PDmn	Selection of the internal pull-down resistor for the Pmn port (m=A, B, C, D, N=0~15).
0	No internal pull-down resistors are connected.
1	Connect an internal pull-down resistor.

Note: Unassigned bits must be initialized.

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 of different potentials and I^{2}C with external devices of the same potential, the N-channel open-drain output mode can be selected for the SDAxx port.

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

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

Note: For the bit that sets the N-channel open-drain output mode (POMmn=1), no internal pull-up resistor is connected.

Figure 2-7: 8port output mode register

symbol	15	14	13	12	11	10	9	8	offset addr	after reset	R/W
POMA	0	POMA14	APPLE13	APPLE12	APPLE11	APPLE10	POMA9	POMA8	0x040	0000H	R/W
	7	6	5	4	3	2	1	0			
	POMA7	POMA6	POMA5	POMA4	POMA3	POMA2	POMA1	POMA0			
POMB	0	0	0	0	0	0	0	POMB8	0x042	0000H	R/W
	POMB7	POMB6	POMB5	POMB4	POMB3	POMB2	POMB1	POMB0			
POMC	POMC15	POMC14	POMC13	POMC12	POMC11	POMC10	POMC9	POMC8	0x044	0000H	R/W
	POMC7	POMC6	POMC5	POMC4	POMC3	POMC2	POMC1	POMC0			
POMD	POMD15	POMD14	POMD13	POMD12	POMD11	POMD10	POMD9	POMD8	0x046	0000H	R/W
	POMD7	POMD6	POMD5	POMD4	POMD3	POMD2	POMD1	POMD0			
POMH	0	0	0	0	0	0	0	0	0x04E	0000H	R/W
	0	0	0	POMH4	POMH3	POMH2	POMH1	0			

POMmn	Selection of the output mode of the Pmn port (m=A,B,C,D,H, N=0~15)
0	Normal output mode
1	N-channel open-drain output mode

Note: 1 Unassigned bits must be initialized.

2.3.8 Port mode control register (PMCxx)

The PMC registers set the port in bits for use as a digital input/output or as an analog channel.

After the reset signal is generated, PB 0 3 defaults to use as a digital channel (PMCB3 reset value is "0"), and the other ports default to analog channels. Ports without PMC registers have digital functions only and cannot be used as analog channels.

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.

Figure 2-9: 10port mode control register

symbol	15	14	13	12	11	10	9	8	address	after reset	R/W
	7	6	5	4	3	2	1	0			
PMCA	1	PMCA4	PMCA3	PMCA2	PMCA1	1	1	1	0x050	FFFFH	R/W
	1	PMCA6	PMCA5	PMCA4	PMCA3	PMCA2	PMCA1	PMCA0			
PMCB	1	1	1	1	1	1	1	PMCB8	0x052	FFF7H	R/W
	PMCB7	PMCB6	1	PMCB4	PMCB3	PMCB2	PMCB1	1			
PMCC	1	1	1	1	PMCC11	PMCC10	PMCC9	PMCC8	0x054	FFFFH	R/W
	PMCC7	PMCC6	PMCC5	PMCC4	PMCC3	PMCC2	PMCC1	PMCC0			
PMCD	1	1	1	1	1	1	1	PMCD8	0x056	FFFFH	R/W
	PMCD7	PMCD6	PMCD5	PMCD4	1	1	1	1			

PMCmn	Selection of digital input/output or analog input of PMN port (m=A,B,C,D, N=0~15)
0	Digital inputs/outputs (multiplexed functions other than analog inputs)
1	Analog input

Note: 1. Unassigned bits must be initialized.

2.3.9 Port read register (PREADxx)

This is a read-only register that can be read to get the port level when the port is a normal 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 2-11: Format of port read register

symbol	15	14	13	12	11	10	9	8	address	After	R/W
	7	6	5	4	3	2	1	0		reset	
PREADA	0	PREADA14	PREADA13	PREADA12	PREADA11	PREADA10	PREADA9	PREADA8	0x080	xxxxH	R
	PREADA7	PREADA6	PREADA5	PREADA4	PREADA3	PREADA2	PREADA1	PREADA0			
PREADB	0	0	0	0	0	0	0	PREADB8	0x082	xxxxH	R
	PREADB7	PREADB6	PREADB5	PREADB4	PREADB3	PREADB2	PREADB1	PREADB0			
PREADC	PREADC15	PREADC14	PREADC13	PREADC12	PREADC11	PREADC10	PREADC9	PREADC8	0x084	xxxxH	R
	PREADC7	PREADC6	PREADC5	PREADC4	PREADC3	PREADC2	PREADC1	PREADC0			
PREADD	PREADD15	PREADD14	PREADD13	PREADD12	PREADD11	PREADD10	PREADD9	PREADD8	0x086	xxxxH	R
	PREADD7	PREADD6	PREADD5	PREADD4	PREADD3	PREADD2	PREADD1	PREADD0			
PREADH	0	0	0	0	0	0	0	0	0x08E	xxxxH	R
	0	0	0	PREADH4	PREADH3	PREADH2	PREADH1	PREADH0*			

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

PREADmn	m=A,B,C,D,H,n=0~15
	Output mode/input mode
0	The port is low.
1	The port is high.

2.3.10 Port output multiplexing function configuration register (PxxCFG)

The port-output multiplexing function configuration register allows the output functions of a subset of peripheral modules to be redirected to different ports. The reset value of the port output multiplexing function configuration register is "00H", in which case the port is the default multiplexing function and GPIO function.

Register address = base address + offset address; The base address of the PxxCFG register is 0x40040900, and the offset address is shown in the figure below.

Figure 2-12: 213output multiplexing function configuration registers

Port group	Register name	Offset address	Function	R/W	Reset value
GRP0	PB00CFG[2:0]	0x000	PB00 port multiplexing output selection	R/W	00H
	PH04CFG[2:0]	0x001	PH04 port multiplexed output selection	R/W	00H
	PH03CFG[2:0]	0x002	PH03 port multiplexed output selection	R/W	00H
	PH02CFG[2:0]	0x003	PH02 port multiplexed output selection	R/W	00H
	PH01CFG[2:0]	0x004	PH01 port multiplexed output selection	R/W	00H
	PC14CFG[2:0]	0x005	PC14 port multiplexed output selection	R/W	00H
	PC15CFG[2:0]	0x006	PC15 port multiplexed output selection	R/W	00H
	PC08CFG[2:0]	0x007	PC08 port multiplexed output selection	R/W	00H
	PC09CFG[2:0]	0x008	PC09 port multiplexing output selection	R/W	00H
	PC10CFG[2:0]	0x009	PC10 port multiplexed output selection	R/W	00H
	PC11CFG[2:0]	0x00a	PC11 port multiplexed output selection	R/W	00H
	PA00CFG[2:0]	0x00b	PA00 port multiplexed output selection	R/W	00H
	PA01CFG[2:0]	0x00c	PA01 port multiplexing output selection	R/W	00H
	PA02CFG[2:0]	0x00d	PA02 port multiplexed output selection	R/W	00H
	PA03CFG[2:0]	0x00e	PA03 port multiplexing output selection	R/W	00H
	PD07CFG[2:0]	0x00f	PD07 port multiplexed output selection	R/W	00H
	PD08CFG[2:0]	0x010	PD08 port multiplexed output selection	R/W	00H
	PD09CFG[2:0]	0x011	PD09 port multiplexed output selection	R/W	00H
	PD10CFG[2:0]	0x012	PD10 port multiplexed output selection	R/W	00H
	PD11CFG[2:0]	0x013	PD11 port multiplexed output selection	R/W	00H

Port group	Register name	Offset address	Function	R/W	Reset value
GRP1	PC03CFG[3:0]	0x020	PC03 port multiplexed output selection	R/W	00H
	PC04CFG[3:0]	0x021	PC04 port multiplexed output selection	R/W	00H
	PC05CFG[3:0]	0x022	PC05 port multiplexed output selection	R/W	00H
	PC06CFG[3:0]	0x023	PC06 port multiplexed output selection	R/W	00H
	PC07CFG[3:0]	0x024	PC07 port multiplexed output selection	R/W	00H
	PC12CFG[3:0]	0x025	PC12 port multiplexed output selection	R/W	00H
	PC13CFG[3:0]	0x026	PC13 port multiplexed output selection	R/W	00H
	PA04CFG[3:0]	0x027	PA04 port multiplexing output selection	R/W	00H
	PA05CFG[3:0]	0x028	PA05 port multiplexed output selection	R/W	00H
	PA06CFG[3:0]	0x029	PA06 port multiplexed output selection	R/W	00H
	PA07CFG[3:0]	0x02a	PA07 port multiplexed output selection	R/W	00H
	PA08CFG[3:0]	0x02b	PA08 port multiplexed output selection	R/W	00H
	PA09CFG[3:0]	0x02c	PA09 port multiplexed output selection	R/W	00H
	PA10CFG[3:0]	0x02d	PA10 port multiplexed output selection	R/W	00H
	PD00CFG[3:0]	0x02e	PD00 port multiplexed output selection	R/W	00H
	PD01CFG[3:0]	0x02f	PD01 port multiplexed output selection	R/W	00H
	PD12CFG[3:0]	0x030	PD12 port multiplexed output selection	R/W	00H
	PD13CFG[3:0]	0x031	PD13 port multiplexed output selection	R/W	00H
	PD14CFG[3:0]	0x032	PD14 port multiplexed output selection	R/W	00H
	PD15CFG[3:0]	0x033	PD15 port multiplexed output selection	R/W	00H

Port group	Register name	Offset address	Function	R/W	Reset value
GRP2	PB01CFG[2:0]	0x040	PB01 port multiplexing output selection	R/W	00H
	PB02CFG[2:0]	0x041	PB02 port multiplexing output selection	R/W	00H
	PB03CFG[2:0]	0x042	PB03 port multiplexing output selection	R/W	00H
	PB04CFG[2:0]	0x043	PB04 port multiplexing output selection	R/W	00H
	PB05CFG[2:0]	0x044	PB05 port multiplexing output selection	R/W	00H
	PB06CFG[2:0]	0x045	PB06 port multiplexing output selection	R/W	00H
	PB07CFG[2:0]	0x046	PB07 port multiplexing output selection	R/W	00H
	PB08CFG[2:0]	0x047	PB08 port multiplexing output selection	R/W	00H
	PC00CFG[2:0]	0x048	PC00 port multiplexed output selection	R/W	00H
	PC01CFG[2:0]	0x049	PC01 port multiplexed output selection	R/W	00H
	PC02CFG[2:0]	0x04a	PC02 port multiplexed output selection	R/W	00H
	PA11CFG[2:0]	0x04b	PA11 port multiplexed output selection	R/W	00H
	PA12CFG[2:0]	0x04c	PA12 port multiplexed output selection	R/W	00H
	PA13CFG[2:0]	0x04d	PA13 port multiplexed output selection	R/W	00H
	PA14CFG[2:0]	0x04e	PA14 port multiplexed output selection	R/W	00H
	PD02CFG[2:0]	0x04f	PD02 port multiplexed output selection	R/W	00H
	PD03CFG[2:0]	0x050	PD03 port multiplexed output selection	R/W	00H
	PD04CFG[2:0]	0x051	PD04 port multiplexed output selection	R/W	00H
	PD05CFG[2:0]	0x052	PD05 port multiplexed output selection	R/W	00H
	PD06CFG[2:0]	0x053	PD06 port multiplexed output selection	R/W	00H

Figure 2-14 15port output multiplexing function configuration register

symbol	7	6	5	4	3	2	1	0	address	after reset	R/W
PxxCFG	0	0	0	0	Pxxcfg3	Pxxcfg2	Pxxcfg1	Pxxcfg0	See image above	12A.M.	R/W

Table 2-2 Selection method for port multiplexing outputs

By configuring the PxxCFG register of GRP0, five multiplexed output functions (TO00, TO01, TO02, TO03, TxD0/SDO00) can be redirected to any port of GRP0.

GRP0			
Serial number	Register name	Register configuration	Port function
0	PB00CFG[2:0]	3'h0	GPIO/default multiplexed output
		3'h1	TO00
		3'h2	TO01
		3'h3	TO02
		3'h4	TO03
		3'h5	TxD0/SDO00
		other	Configuration disabled
1	PH04CFG[2:0]	ditto	
2	PH03CFG[2:0]	ditto	
3	PH02CFG[2:0]	ditto	
4	PH01CFG[2:0]	ditto	
5	PC14CFG[2:0]	ditto	
6	PC15CFG[2:0]	ditto	
7	PC08CFG[2:0]	ditto	
8	PC09CFG[2:0]	ditto	
9	PC10CFG[2:0]	ditto	
10	PC11CFG[2:0]	ditto	
11	PA00CFG[2:0]	ditto	
12	PA01CFG[2:0]	ditto	
13	PA02CFG[2:0]	ditto	
14	PA03CFG[2:0]	ditto	
15	PD07CFG[2:0]	ditto	
16	PD08CFG[2:0]	ditto	
17	PD09CFG[2:0]	ditto	
18	PD10CFG[2:0]	ditto	
19	PD11CFG[2:0]	ditto	

By configuring GRP1's PxxCFG registers, eight output functions (TO10, TO 1 1, TO 1 2, TO 1 3, TxD1/ IrTxD/ SDO10, SPIHS0_SCKO, SPIHS0_MO, SPIHS0_SO) Redirect to any port of GRP1.

GRP1			
Serial number	Register name	Register configuration	Port function
0	PC03CFG[3:0]	4'h00	GPIO/default multiplexed output
		4'h01	TO10
		4'h02	TO11
		4'h03	TO12
		4'h04	TO13
		4'h05	TxD1/IrTxD/SDO10
		4'h06	SPIHS0_SCKO
		4'h07	SPIHS0_MO
		4'h08	SPIHS0_SO
		other	Configuration disabled
1	PC04CFG[3:0]	ditto	
2	PC05CFG[3:0]	ditto	
3	PC06CFG[3:0]	ditto	
4	PC07CFG[3:0]	ditto	
5	PC12CFG[3:0]	ditto	
6	PC13CFG[3:0]	ditto	
7	PA04CFG[3:0]	ditto	
8	PA05CFG[3:0]	ditto	
9	PA06CFG[3:0]	ditto	
10	PA07CFG[3:0]	ditto	
11	PA08CFG[3:0]	ditto	
12	PA09CFG[3:0]	ditto	
13	PA10CFG[3:0]	ditto	
14	PD00CFG[3:0]	ditto	
15	PD01CFG[3:0]	ditto	
16	PD12CFG[3:0]	ditto	
17	PD13CFG[3:0]	ditto	
18	PD14CFG[3:0]	ditto	
19	PD15CFG[3:0]	ditto	

Note: The three pins PC03, PC04 and PC05 are high-speed pins. When the SPIHS0_SCK, SPIHS0_ SIMO, SPIHS0_ MISO are configured to the three pins of PC03, PC04 and PC05, SPI0 enables high-speed communication

By configuring the PxxCFG register of GRP2, six output functions (TO14, TO15, TO 16, TO 17, TxD 2/SDO20) can be multiplexed, CLKBUZ1) redirects to any port of GRP2.

GRP2			
Serial number	Register name	Register configuration	Port function
0	PB01CFG[2:0]	3'h0	GPIO/default multiplexed output
		3'h1	TO14
		3'h2	TO15
		3'h3	TO16
		3'h4	TO17
		3'h5	TxD2/SDO20
		3'h6	CLKBUZ1
		other	Configuration disabled
1	PB02CFG[2:0]	ditto	
2	PB03CFG[2:0]	ditto	
3	PB04CFG[2:0]	ditto	
4	PB05CFG[2:0]	ditto	
5	PB06CFG[2:0]	ditto	
6	PB07CFG[2:0]	ditto	
7	PB08CFG[2:0]	ditto	
8	PC00CFG[2:0]	ditto	
9	PC01CFG[2:0]	ditto	
10	PC02CFG[2:0]	ditto	
11	PA11CFG[2:0]	ditto	
12	PA12CFG[2:0]	ditto	
13	PA13CFG[2:0]	ditto	
14	PA14CFG[2:0]	ditto	
15	PD02CFG[2:0]	ditto	
16	PD03CFG[2:0]	ditto	
17	PD04CFG[2:0]	ditto	
18	PD05CFG[2:0]	ditto	
19	PD06CFG[2:0]	ditto	

Configuration Instructions:

- When using the port's multiplexing function, the port must be configured in digital mode (PMCxx=0).
- When using the port's multiplexed output function, the port must be configured for output mode (push-pull or open-drain) (PMxx=0).
- When using the GPIO function or multiplexing function of PH01 and PH02 ports, make sure that the X1 oscillation mode and external clock input mode are not turned on. Refer to "Chapter 4 Clock Generation Circuit 4.3.1"
- When using the GPIO function or multiplexing function of PH03 and PH04 ports, make sure that the XT1 oscillation mode and external clock input mode are not turned on. Refer to "Chapter 4 Clock Generation Circuit 4.3.1"
- When using the port's multiplexed output function, you need to set the port output latch Pxx, 2.5 see Register settings when using the multiplexing function

- The clock port of IICA0/1 (SCLA0/1) and the data port of IICA (SDAA0/1) support bidirectional communication, and only SCLA0PCFG/SCLA1PCFG, SDAA0PCFG/SDAA1PCFG registers need to be configured when setting the redirection port, and there is no need to configure PxxCFG registers.
- The three pins PC03, PC04 and PC05 are high-speed pins. When the SPIHS0_SCK, SPIHS0_ SIMO, SPIHS0_ MISO are configured to the three pins of PC03, PC04 and PC05, SPI0 enables high-speed communication.

2.3.11 Port input multiplexing function configuration register

The port input multiplexing function configuration register allows the input functions of some peripheral modules to be redirected to different ports. The reset value of the port input multiplexing function configuration register is "00H".

Register address = base address + offset address; the base address of the register is 0x40040900, and the offset address is shown in the figure below.

Figure 216input multiplexing function configuration registers

Multiplexed group	Register name	Offset address	Function	R/W	Reset value
GRP0	TI00PCFG	0x080	TI00 redirects port selection	R/W	00H
	TI01PCFG	0x081	TI01 redirects port selection	R/W	00H
	TI02PCFG	0x082	TI02 redirect port selection	R/W	00H
	TI03PCFG	0x083	TI03 redirects port selection	R/W	00H
	RXD0PCFG	0x084	RxD0/SDI00 redirected port selection	R/W	00H
	SCLA0PCFG	0x085	SCLA0 redirects port selection	R/W	00H
	SDAA0PCFG	0x086	SDAA0 redirection port selection	R/W	00H
GRP1	TI10PCFG	0x0A0	TI10 redirects port selection	R/W	00H
	TI11PCFG	0x0A1	TI11 redirect port selection	R/W	00H
	TI12PCFG	0x0A2	TI12 redirect port selection	R/W	00H
	TI13PCFG	0x0A3	TI13 redirection port selection	R/W	00H
	RXD1PCFG	0x0A4	RxD1/IrRxD/SDI10 redirection port selection	R/W	00H
	SPIHS0_SCKIPCFG	0x0A5	SPIHS0_SCKI redirect port selection	R/W	00H
	SPIHS0_SIPCFG	0x0A6	SPIHS0_SI redirect port selection	R/W	00H
	SPIHS0_MIPCFG	0x0A7	SPIHS0_MI redirect port selection	R/W	00H
GRP2	TI14PCFG	0x0C0	TI14 redirection port selection	R/W	00H
	TI15PCFG	0x0C1	TI15 redirects port selection	R/W	00H
	TI16PCFG	0x0C2	TI16 redirect port selection	R/W	00H
	TI17PCFG	0x0C3	TI17 redirection port selection	R/W	00H
	RXD2PCFG	0x0C4	RXD2/SDI20 redirection port selection	R/W	00H
	SPIHS1_NSSPCFG	0x0C5	SPIHS1_NSS redirect port selection	R/W	00H
	SCLA1PCFG	0x0C6	SCLA1 redirect port selection	R/W	00H
	SDAA1PCFG	0x0C7	SDAA1 redirection port selection	R/W	00H

Figure 2-13 17port input multiplexing function configuration register

symbol	7	6	5	4	3	2	1	0	address	after reset	R/W
xxPCFG	0	0	0	xxpcfg[4:0]					See image above	12A.M.	R/W

By configuring the following registers, 7 input functions (TI00, TI01, TI02, TI03, RxD0/SDI00, SCLA0, SDAA0) redirects to any port of GRP0.

Table 2-3 Selection methods for multiplexing input ports

Serial number	Register name	Register configuration	Ports for multiplexing inputs
0	TI00PCFG	5'h00	No port multiplexing TI00
		5'h01	Redirect to PB00
		5'h02	Redirect to PH04
		5'h03	Redirect to PH03
		5'h04	Redirect to PH02
		5'h05	Redirect to PH01
		5'h06	Redirect to PC14
		5'h07	Redirect to PC15
		5'h08	Redirect to PC08
		5'h09	Redirect to PC09
		5'h0A	Redirect to PC10
		5'h0B	Redirect to PC11
		5'h0C	Redirect to PA00
		5'h0D	Redirect to PA01
		5'h0E	Redirect to PA02
		5'h0F	Redirect to PA03
		5'h10	Redirect to PD07
		5'h11	Redirect to PD08
		5'h12	Redirect to PD09
		5'h13	Redirect to PD10
		5'h14	Redirect to PD11
		other	Prohibit setting
1	TI01PCFG	ditto	
2	TI02PCFG	ditto	
3	TI03PCFG	ditto	
4	RXD0PCFG	ditto	
5	SCLA0PCFG	ditto	
6	SDAA0PCFG	ditto	

By configuring the following registers, 8 input functions (TI10, TI 1 1, TI 1 2, TI 13) can be multiplexed , RxD1/IrRxD/SDI10, SPIHS0_SCKI, SPIHS0_SI, SPIHS0_MI) Redirect to any port of GRP1.

Serial number	Register name	Register configuration	Ports for multiplexing inputs
0	TI10PCFG	5'h00	No port multiplexing TI10
		5'h01	Redirect to PC03
		5'h02	Redirect to PC04
		5'h03	Redirect to PC05
		5'h04	Redirect to PC06
		5'h05	Redirect to PC07
		5'h06	Redirect to PC12
		5'h07	Redirect to PC13
		5'h08	Redirect to PA04
		5'h09	Redirect to PA05
		5'h0A	Redirect to PA06
		5'h0B	Redirect to PA07
		5'h0C	Redirect to PA08
		5'h0D	Redirect to PA09
		5'h0E	Redirect to PA10
		5'h0F	Redirect to PD00
		5'h10	Redirect to PD01
		5'h11	Redirect to PD12
		5'h12	Redirect to PD13
		5'h13	Redirect to PD14
		5'h14	Redirect to PD15
		other	Prohibit setting
1	TI11PCFG	ditto	
2	TI12PCFG	ditto	
3	TI13PCFG	ditto	
4	RXD1PCFG	ditto	
5	SPIHS0_SCKIPCFG	ditto	
6	SPIHS0_SIPCFG	ditto	
7	SPIHS0_MIPCFG	ditto	

Note: The three pins PC03, PC04 and PC05 are high-speed pins. When the SPIHS0_SCK, SPIHS0_SIMO, SPIHS0_MI SO are configured to the three pins of PC03, PC04 and PC05, SPI0 enables high-speed communication.

By configuring the following registers, 8 input functions (TI14, TI 15, TI 16, TI17, RxD 2/SDI20,) can be multiplexed SPIHS0_NSS, SCLA1, SDAA1) redirects to any port of GRP2.

Serial number	Register name	Register configuration	Ports for multiplexing inputs
0	TI14PCFG	5'h00	No port multiplexing TI14
		5'h01	Redirect to PB01
		5'h02	Redirect to PB02
		5'h03	Redirect to PB03
		5'h04	Redirect to PB04
		5'h05	Redirect to PB05
		5'h06	Redirect to PB06
		5'h07	Redirect to PB07
		5'h08	Redirect to PB08
		5'h09	Redirect to PC00
		5'h0A	Redirect to PC01
		5'h0B	Redirect to PC02
		5'h0C	Redirect to PA11
		5'h0D	Redirect to PA12
		5'h0E	Redirect to PA13
		5'h0F	Redirect to PA14
		5'h10	Redirect to PD02
		5'h11	Redirect to PD03
		5'h12	Redirect to PD04
		5'h13	Redirect to PD05
		5'h14	Redirect to PD06
		other	Prohibit setting
1	TI15PCFG	ditto	
2	TI16PCFG	ditto	
3	TI17PCFG	ditto	
4	RXD2PCFG	ditto	
5	SPIHS1_NSSPCFG	ditto	
6	SCLA1PCFG	ditto	
7	SDAA1PCFG	ditto	

Configuration Instructions:

- When using the port's multiplexing function, the port must be configured in digital mode (PMCxx=0).
- When using the port's multiplex input function, the port must be configured for input mode (PMxx=1).
- For bidirectional multiplexing, the port must be configured in output mode (push-pull or open-drain) (PMxx=0). At this point, the input drive is configured for floating input mode.
- When using the GPIO function or multiplexing function of PH01 and PH02 ports, make sure that the X1 oscillation mode and external clock input mode are not turned on. Refer to "Chapter 4 Clock Generation Circuit 4.3.1"
- When using the GPIO function or multiplexing function of PH03 and PH04 ports, make sure that the XT1 oscillation mode and external clock input mode are not turned on. Refer to "Chapter 4 Clock Generation Circuit 4.3.1"

- IICA's clock port (SCLA0/1) and IICA's data port (SDAA0/1) support bidirectional communication, and only SCLA0PCFG, SDAA0PCFG, SCLA1PCFG, SDAA1PCFG registers need to be configured when setting the redirection port, and there is no need to configure PxxCFG registers or POMxx registers.
- The three pins PC03, PC04 and PC05 are high-speed pins. When the SPIHS0_SCK, SPIHS0_SIMO, SPIHS0_MI SO are configured to the three pins of PC03, PC04 and PC05, SPI0 enables high-speed communication.

2.3.12 External interrupt port selection register (INTPnPCFG)

This product supports 8-way external interrupt INTP0~7, each external interrupt can be redirected to multiple ports. By configuring an external interrupt port selection register (INTPnPCFG), it is possible to redirect the input functions of the INTPn to a different port. The reset value of the external interrupt port select register is "00H". (n=0~7)

Register address = base address + offset address; The base address of the INTPnPCFG register is 0x40040900, and the offset address is shown in the figure below.

Figure 2external interrupt port selection registers

Register name	Offset address	Function	R/W	Reset value
INTP0PCFG	0x0E0	INTP0 redirection port selection	R/W	00H
INTP1PCFG	0x0E1	INTP1 redirection port selection	R/W	00H
INTP2PCFG	0x0E2	INTP2 redirect port selection	R/W	00H
INTP3PCFG	0x0E3	INTP3 redirect port selection	R/W	00H
INTP4PCFG	0x0E4	INTP4 redirect port selection	R/W	00H
INTP5PCFG	0x0E5	INTP5 redirect port selection	R/W	00H
INTP6PCFG	0x0E6	INTP6 redirect port selection	R/W	00H
INTP7PCFG	0x0E7	INTP7 redirect port selection	R/W	00H

Figure 218external interrupt port selection register

symbol	7	6	5	4	3	2	1	0	address	after reset	R/W
INTPnPCFG	0	0	0	0	0	INTPnPCFG[2:0]			See image above	12A.M.	R/W

Serial number	Register name	Register configuration	INTP0 port selection
0	INTP0PCFG	3'h00	PC00
1		3'h01	PC01
2		3'h02	PC02
3		3'h03	PC03
4		3'h04	PC04
5		3'h05	PC05
6		3'h06	PC06
7		3'h07	PC07

Serial number	Register name	Register configuration	INTP1 port selection
0	INTP1PCFG	3'h00	PC12
1		3'h01	PC13
2		3'h02	PC14
3		3'h03	PC15
4		3'h04	PC08
5		3'h05	PC09
6		3'h06	PC10
7		3'h07	PC11

Serial number	Register name	Register configuration	INTP2 port selection
0	INTP2PCFG	3'h00	PA00
1		3'h01	PA01
2		3'h02	PA02
3		3'h03	PA03
4		3'h04	PA11
5		3'h05	PA12
6		3'h06	PA13
7		3'h07	PA14

Serial number	Register name	Register configuration	INTP3 port selection
0	INTP3PCFG	3'h00	PA04
1		3'h01	PA05
2		3'h02	PA06
3		3'h03	PA07
4		3'h04	PA08
5		3'h05	PA09
6		3'h06	PA10

Serial number	Register name	Register configuration	INTP4 port selection
0	INTP4PCFG	3'h00	PD00
1		3'h01	PD01
2		3'h02	PD12
3		3'h03	PD13
4		3'h04	PD14
5		3'h05	PD15
6		3'h06	PD02
7		3'h07	PD03

Serial number	Register name	Register configuration	INTP5 port selection
0	INTP5PCFG	3'h00	PD04
1		3'h01	PD05
2		3'h02	PD06
3		3'h03	PD07
4		3'h04	PD08
5		3'h05	PD09
6		3'h06	PD10
7		3'h07	PD11

Serial number	Register name	Register configuration	INTP6 port selection
0	INTP6PCFG	3'h00	PB00
1		3'h01	PH04
2		3'h02	PH03
3		3'h03	PH02
4		3'h04	PH01
5		3'h05	PB01
6		3'h06	PB02

Serial number	Register name	Register configuration	INTP7 port selection
0	INTP7PCFG	3'h00	PB03
1		3'h01	PB04
2		3'h02	PB05
3		3'h03	PB06
4		3'h04	PB07
5		3'h05	PB08

2.3.13 USB_ DP, USB_DM port configuration registers (PMR, PRCR)

USB_DP is multiplexed with PA08 and USB_DM is multiplexed with PA07, and the function of the port can be selected by setting the PMR register. It is used as a USB port by default.

Base address: 0x40040400

symbol	7	6	5	4	3	2	1	0	Offset addr	after reset	R/W
PMR	0	0	0	0	0	0	DMPMR	DPPMR	0x07D	03H	R/W

DMPMR	DPPMR	Port function	
		PA07/USB_DM	PA08/USB_DP
0	0	GPIO	GPIO
0	1	GPIO	GPIO
1	0	GPIO	GPIO
1	1	USB_DM	USB_DP

Note 1: When PA07/USB_DM and PA08/USB_DP are used as GPIOs (PA07 and PA08), the USB registers need to be kept initial.

Note 2: When PA07/USB_DM and PA08/USB_DP are used as USB terminals (USB_DM and USB_DP), the GPIO registers need to be kept initial.

Note 3: After reset, the PMR register can only be set once to switch PA07/USB_DM and PA08/USB_DP whether they are used as GPIO or USB terminals.

PRCR is a protection register for PMR to prevent USB _DM and USB _DP terminals from being accidentally switched to GPIO.

Base address: 0x40040400

symbol	7	6	5	4	3	2	1	0	Affset addr	after reset	R/W
PRCR	PRTKEY[7:1]							PRCR	0x07E	12A.M.	R/W

PRCR	PMR control register write protection
0	• The PMR register is not writable
1	• PMR registers are writable

PRTKEY[7:1]	Write protection of PRCR
78H	• PRCR is writable
other	• PRCR is not writable

2.4 Handling of unused ports

The handling of each unused port is as following Table 2-.

Table 2-5 Handling of each unused port

Port name	Input/Output	Recommended connection method when not in use
PA00~PA14	Input/Output	Input: The EV_{DD} or EV_{SS} are connected by resistance alone. Output: Open circuit.
PB00~PB08		
PC00~PC15		
PD00, PD15		
PH01~PH04		The V_{DD} or V_{SS} are connected by resistance alone.
RESETB/PH00	Input	The VDD is connected directly or through resistance·

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

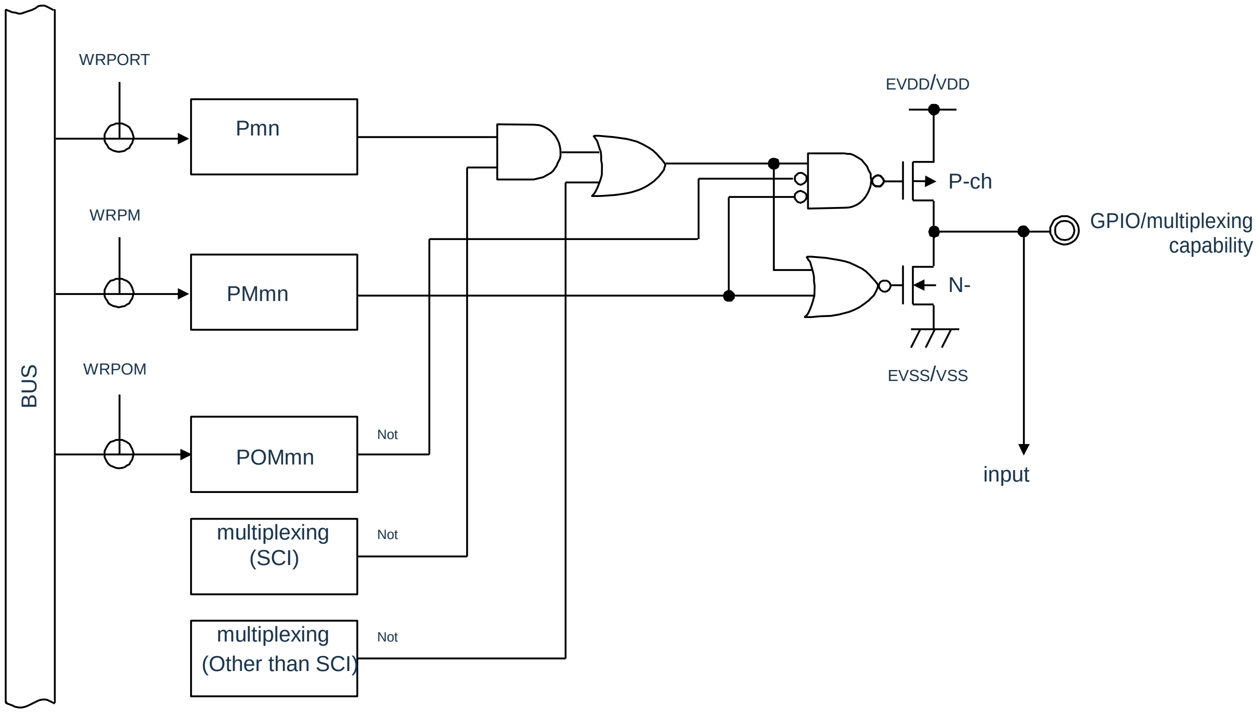
2.5 Register settings when using the multiplexing function

2.5.1 Basic ideas when using the multiplexed output function

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

The basic structure of the output circuit when used as a digital input/output is shown in Figure 2-19. The output of the SCI function multiplexed with the output latch of the port is input to the AND gate, the output of the AND gate is input to the OR gate, and the other inputs of the OR gate are connected to the multiplexed non Output of SCI functions (timer, RTC, clock/buzzer output, IICA, etc.). When such a port is used as a port function or a multiplexing feature, unused multiplexed features cannot affect the output of the functionality to be used. The basic idea of setting in this case is shown in Table2-.

Figure 2-19 220



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

2. The signal is High level (1) when the multiplexing function is not available.

3. The signal is Low level (0) when the multiplexing function is not available.

Table2-6 Basic principal for configuration

Pin Output Used	Output settings for unused multiplexing		
	port function	SCI output function	Output capabilities other than SCI
Port Output	—	High level output (1)	Low level output (0)
SCI output function	High(1)	—	Low level output (0)
Output capabilities other than SCI	Low(0)	High level output (1)	Low level output (0) Note

Note: Since it is possible to multiplex multiple output functions other than SCI on one port, it is necessary to set the output of the multiplexed function that is not used to a Low level (0). For specific configuration methods, please refer to "2.5.2, 2.5.3, 2.5.4.

2.5.2 Configuration method of multiplexed output function

Table 2-7 Configuration method of multiplexed output function

Feature name		Input/output	PxxCFP	PMCxx	PMxx	POMxx	Pxx	Remark
Analog channels		Input/output	4’h0	1	x	x	x	All analog functions are directed to fixed ports only and do not support configuration, see the data sheets for each product family
Digital GPIO		output	4’h0	0	0	0	0/1	
		N-channel open-drain output		0	0	1	0/1	
GRP0 port can be equipped with multiplexed output	TO00	output	4’h1	0	0	0	0	Can be redirected to any port of GRP0
	TO01	output	4’h2	0	0	0	0	Can be redirected to any port of GRP0
	TO02	output	4’h3	0	0	0	0	Can be redirected to any port of GRP0
	TO03	output	4’h4	0	0	0	0	Can be redirected to any port of GRP0
	TXD0/SDO00	output	4’h5	0	0	0	1	Can be redirected to any port of GRP0
GRP1 port can be equipped with multiplexed output	TO10	output	4'h 1	0	0	0	0	Can be redirected to any port on G RP1
	TO11	output	4'h 2	0	0	0	0	Can be redirected to any port on G RP1
	TO12	output	4'h 3	0	0	0	0	Can be redirected to any port on G RP1
	TO13	output	4'h 4	0	0	0	0	Can be redirected to any port on G RP1
	TxD1/IrTxD/SDO10	output	4'h 5	0	0	0	1	Can be redirected to any port on G RP1
	SPIHS0_SCKO	output	4'h 6	0	0	0	0	Can be redirected to any port on G RP1
	SPIHS0_MO	output	4'h 7	0	0	0	0	Can be redirected to any port on G RP1
	SPIHS0_SO	output	4'h 8	0	0	0	0	Can be redirected to any port on G RP1
GRP2 port can be equipped with multiplexed output	TO14	output	4'h 1	0	0	0	0	Can be redirected to any port of GRP2
	TO15	output	4'h 2	0	0	0	0	Can be redirected to any port of GRP2
	TO16	output	4'h 3	0	0	0	0	Can be redirected to any port of GRP2
	TO17	output	4'h 4	0	0	0	0	Can be redirected to any port of GRP2
	TxD2/SDO20	output	4'h 5	0	0	0	1	Can be redirected to any port of GRP2
	CLKBUZ1	output	4'h 6	0	0	0	0	Can be redirected to any port of GRP2
The multiplexed output of the fixed port	CLKBUZ0	output	PA00CFG=4’h0	0	0	0	0	By default, PA00 is used and cannot be mapped to other ports
	SDO01	output	PA12CFG=4’h0	0	0	0	1	By default, PA12 is used and cannot be mapped to other ports
	SCLKO01/SCL01	output	PA14CFG=4’h0	0	0	0	0	By default, PA14 is used and cannot be mapped to other ports
	RTC1HZ	output	PB00CFG=4’h0	0	0	0	0	By default, PB00 is used and cannot be mapped to other ports
	SCLKO21/SC	output	PB03CFG=4’h	0	0	0	1	By default, PB03 is used and

	L21		0					cannot be mapped to other ports
	SDO21	output	PB05CFG=4'h 0	0	0	0	1	By default, PB05 is used and cannot be mapped to other ports
	SPIHS1_SCKO	output	PC00CFG=4'h 0	0	0	0	0	By default, PC00 is used and cannot be mapped to other ports
	SPIHS1_MO	output	PC01CFG=4'h 0	0	0	0	0	By default, PC01 is used and cannot be mapped to other ports
	SPIHS1_SO	output	PC02CFG=4'h 0	0	0	0	0	By default, PC02 is used and cannot be mapped to other ports
	SCLKO11/SCL11	output	PC03CFG=4'h 0	0	0	0	1	By default, PC03 is used and cannot be mapped to other ports
	SDO11	output	PC04CFG=4'h 0	0	0	0	1	By default, PC04 is used and cannot be mapped to other ports
	DBWRB	output	PC10CFG=4'h 0	0	0	0	0/1	By default, P C10 is used, and cannot be mapped to other ports according to different modes PC10 needs to be set to different values, see the LCD BUS interface chapter for details
	DBRDB	output	PC11CFG=4'h 0	0	0	0	0/1	By default, PC11 is used, and cannot be mapped to other ports according to different modes PC11 needs to be set to different values, see the LCD BUS interface chapter for details
	InSB_VBUSEN	output	PD01CFG=4'h 0	0	0	0	0	By default, PD01 is used and cannot be mapped to other ports
	InSB_EXICEN	output	PD05CFG=4'h 0	0	0	0	0	By default, PD01 is used and cannot be mapped to other ports
	SSITXD0	output	PD13CFG=4'h 0	0	x	0	0	By default, PD13 is used, which cannot be mapped to other ports and has its own output control, and PMD bit13 does not need to be configured
	SSILRCK0O/SSIFS0O	output	PD14CFG=4'h 0	0	x	0	0	By default, PD14 is used, can not be mapped to other ports have their own output control, PMD bit14 does not need to be configured
	SSIBCK0O	output	PD15CFG=4'h 0	0	x	0	0	By default, PD15 is used, which cannot be mapped to other ports and has its own output control, and PMD bit15 does not need to be configured
	QSPCLK	output	PD02CFG=4'h 0	0	x	0	0	By default, PD02 is used, which cannot be mapped to other ports and has its own output control, and P MD bit2 does not need to be configured
	QSSL	output	PD03CFG=4'h 0	0	0	0	0	By default, PD03 is used and cannot be mapped to other ports
	ANDPWMO00	output	PD04CFG=4'h 0	0	0	0	0	By default, PD04 is used and cannot be mapped to other ports
	ANDPWMO01	output	PD05CFG=4'h 0	0	0	0	0	By default, PD05 is used and cannot be mapped to other

								ports
	ANDPWMO02	output	PD06CFG=4’h 0	0	0	0	0	By default, PD06 is used and cannot be mapped to other ports
	ANDPWMO03	output	PD07CFG=4’h 0	0	0	0	0	By default, PD07 is used and cannot be mapped to other ports
	ANDPWMO04	output	PD08CFG=4’h 0	0	0	0	0	By default, PD08 is used and cannot be mapped to other ports
	ANDPWMO05	output	PD09CFG=4’h 0	0	0	0	0	By default, PD09 is used and cannot be mapped to other ports
	ANDPWMO06	output	PD10CFG=4’h 0	0	0	0	0	By default, PD10 is used and cannot be mapped to other ports
	ANDPWMO07	output	PD11CFG=4’h 0	0	0	0	0	By default, PD11 is used and cannot be mapped to other ports
	InCOUT0	output	PH01CFG=4’h 0	0	0	0	0	By default, PH01 is used and cannot be mapped to other ports
	VCOUT1	output	PH02CFG=4’h 0	0	0	0	0	By default, PH02 is used and cannot be mapped to other ports

2.5.3 Configuration method of multiplexed input function

Table 2-4 Configuration method of multiplexed input function

Feature name		Input/output	xxxPCFP[4:0]	PMCxx	PMxx	POMxx	Pxx	Remark
Emulation function		Input/output	x	1	x	x	x	All analog functions are directed to fixed ports only and do not support configuration, see the data sheets for each product family
GPIO		Input	x	0	1	x	x	
Repurposeable multiplexed input to GRP0	TUE00	input	Configure TI00PCFG	0	1	x	x	Can be redirected to any port of GRP0
	TUE01	input	ConfigureT I01PCFG	0	1	x	x	Can be redirected to any port of GRP0
	TUE02	input	ConfigureT I02PCFG	0	1	x	x	Can be redirected to any port of GRP0
	TUE03	input	ConfigureT I03PCFG	0	1	x	x	Can be redirected to any port of GRP0
	RXD0/SDI00	input	Configure RXD0PCFG	0	1	x	x	Can be redirected to any port of GRP0
Redirectable multiplexed input to GRP1	TUE10	input	ConfigureT I10PCFG	0	1	x	x	Can be redirected to any port on G RP1
	TUE11	input	Configurethe T I11PCFG	0	1	x	x	Can be redirected to any port on G RP1
	TUE12	input	Configurethe T I12PCFG	0	1	x	x	Can be redirected to any port on G RP1
	TUE13	input	Configurethe T I13PCFG	0	1	x	x	Can be redirected to any port on G RP1
	RxD1/IrRxD/ SDI10	input	Configure RXD1PCFG	0	1	x	x	Can be redirected to any port on G RP1
	SPIHS0_SCKI	input	Configure the SPIHS0_SCKIPCFG	0	1	x	x	Can be redirected to any port on G RP1
	SPIHS0_SI	input	Configure the SPIHS0_SIPCFG	0	1	x	x	Can be redirected to any port on G RP1
	SPIHS0_MI	input	Configure the SPIHS0_MIPCFG	0	1	x	x	Can be redirected to any port on G RP1
Redirectable multiplexed input to GRP2	TUE14	input	Configure the TI14PCFG	0	1	x	x	Can be redirected to any port of GRP2
	TUE15	input	Configure the TI15PCFG	0	1	x	x	Can be redirected to any port of GRP2
	TUE16	input	Configure the TI16PCFG	0	1	x	x	Can be redirected to any port of GRP2
	TUE17	input	Configure the TI17PCFG	0	1	x	x	Can be redirected to any port of GRP2
	RxD2/SDI20	input	Configure RXD2PCFG	0	1	x	x	Can be redirected to any port of GRP2
	SPIHSI_NSS	input	Configure the SPIHS1_NSSPCFG	0	1	x	x	Can be redirected to any port of GRP2
Redirectable external interrupts	INTP0	input	Configure INTP0PCFG	0	1	x	x	Can be redirected to some ports, see section 2.3.11
	INTP1	input	Configure INTP1PCFG	0	1	x	x	Can be redirected to some ports, see section 2.3.11
	INTP2	input	Configure INTP2PCFG	0	1	x	x	Can be redirected to some ports, see section 2.3.11
	INTP3	input	Configure INTP3PCFG	0	1	x	x	Can be redirected to some ports, see section 2.3.11
	INTP4	input	Configure INTP4PCFG	0	1	x	x	Can be redirected to some ports, see section 2.3.11
	INTP5	input	Configure INTP5PCFG	0	1	x	x	Can be redirected to some ports, see section

								2.3.11
	INTP6	input	Configure INTP6PCFG	0	1	x	x	Can be redirected to some ports, see section 2.3.11
	INTP7	input	Configure INTP7PCFG	0	1	x	x	Can be redirected to some ports, see section 2.3.11
Multiplexed inputs for fixed ports	SS00	input	x	0	1	x	x	By default, PA03 is used and cannot be mapped to other ports
	SDI01	input	x	0	1	x	x	By default, PA13 is used and cannot be mapped to other ports
	SCLKI01	input	x	0	1	x	x	By default, PA14 is used and cannot be mapped to other ports
	SCLKI21	input	x	0	1	x	x	By default, PB03 is used and cannot be mapped to other ports
	SDI21	input	x	0	1	x	x	By default, PB04 is used and cannot be mapped to other ports
	SPIHS1_SCKI	input	x	0	1	x	x	By default, PC00 is used and cannot be mapped to other ports
	SPIHS1_SI	input	x	0	1	x	x	By default, PC01 is used and cannot be mapped to other ports
	SPIHS1_M1	input	x	0	1	x	x	By default, PC02 is used and cannot be mapped to other ports
	SCLKI11	input	x	0	1	x	x	By default, PC03 is used and cannot be mapped to other ports
	SDI11	input	x	0	1	x	x	By default, PC05 is used and cannot be mapped to other ports
	KR0	input	x	0	1	x	x	By default, PC06 is used and cannot be mapped to other ports
	KR1	input	x	0	1	x	x	By default, PC07 is used and cannot be mapped to other ports
	KR2	input	x	0	1	x	x	By default, PC12 is used and cannot be mapped to other ports
	KR3	input	x	0	1	x	x	By default, PC13 is used and cannot be mapped to other ports
	KR4	input	x	0	1	x	x	By default, PC14 is used and cannot be mapped to other ports
	KR5	input	x	0	1	x	x	By default, PC15 is used and cannot be mapped to other ports
	KR6	input	x	0	1	x	x	By default, PC08 is used and cannot be mapped to other ports
	KR7	input	x	0	1	x	x	By default, PC09 is used and cannot be mapped to other ports
	SSIRXD0	input	x	0	1	x	x	By default, PD12 is used and cannot be mapped to other ports
	SSILRCK0I/SSIFS01	input	x	0	1	x	x	By default, PD14 is used and cannot be mapped to other ports

	SSIBCK01	input	x	0	1	x	x	By default, PD15 is used and cannot be mapped to other ports
	AUDIO_CLK	input	x	0	1	x	x	By default, PD02 is used and cannot be mapped to other ports
	USB ID	input	x	0	1	x	x	By default, PD04 is used and cannot be mapped to other ports
	USB OVRCURA	input	x	0	1	x	x	By default, PD06 is used and cannot be mapped to other ports
	USB OVRCURB	input	x	0	1	x	x	By default, PD07 is used and cannot be mapped to other ports

2.5.4 Configuration method of the multiplexed bidirectional function

Table 2-5 Configuration method of the multiplexed bidirectional function

Feature name		Input/output	PxxCFP	PMCxx	PMxx	POMxx	Pxx	Remark
Redirectable to GRP0 reuse bidirectional	SCLA0	bidirectional	Configure SCLA0PCFG	0	0	1*	0	POMxx auto-1 can be redirected to any port of GRP0 and requires no software configuration
	SDAA0	bidirectional	Configure SDAA0PCFG	0	0	1*	0	POMxx auto-1 can be redirected to any port of GRP0 and requires no software configuration
Redirectable to GRP2 reuse bidirectional	SCLA1	bidirectional	Configure SCLA1PCFG	0	0	1*	0	POMxx auto-1 can be redirected to any port of GRP2 and requires no software configuration
	SDAA1	bidirectional	Configure SDAA1PCFG	0	0	1*	0	POMxx auto-1 can be redirected to any port of GRP2 and requires no software configuration
Multiplexing fixed ports in both directions	SDA01	bidirectional	PA13CFG=4'h0	0	0	1	1	PA13 is used by default and cannot be mapped to other ports
	SDA21	bidirectional	PB04CFG=4'h0	0	0	1	1	PB04 is used by default and cannot be mapped to other ports
	SDA11	bidirectional	PC05CFG=4'h0	0	0	1	1	PC05 is used by default and cannot be mapped to other ports
	DBD0	bidirectional	PC06CFG=4'h0	0	1	0	0	The default is PC06, PMC.bit6 must be set to 1'b1 that cannot be mapped to other ports
	DBD1	bidirectional	PC07CFG=4'h0	0	1	0	0	The default is PC07, PMC.bit7 must be set to 1'b1 that cannot be mapped to other ports
	DBD2	bidirectional	PC12CFG=4'h0	0	1	0	0	PC12 is used by default and cannot be mapped to other ports PMC.bit12 must be set to 1'b1
	DBD3	bidirectional	PC13CFG=4'h0	0	1	0	0	The default is PC13, which cannot be mapped to other ports PMC.bit13 must be set to 1'b1
	DBD4	bidirectional	PC14CFG=4'h0	0	1	0	0	The default is PC14, PMC.bit14 must be set to 1'b1

									that cannot be mapped to other ports
	DBD5	bidirectional	PC15CFG=4'h0	0	1	0	0	By default, PC15 is used, PMC.bit15 must be set to 1'b1 that cannot be mapped to other ports
	DBD6	bidirectional	PC08CFG=4'h0	0	1	0	0	By default, PC08 is used, PMC.bit8 must be set to 1'b1 that cannot be mapped to other ports
	DBD7	bidirectional	PC09CFG=4'h0	0	1	0	0	The default is PC09, PMC.bit9 must be set to 1'b1 that cannot be mapped to other ports
	QIO0	bidirectional	PD12CFG=4'h0	0	1	0	0	PD12 is used by default, PMD.bit12 must be set to 1'b1 that cannot be mapped to other ports
	QIO1	bidirectional	PD13CFG=4'h0	0	1	0	0	PD13 is used by default, PMD.bit13 must be set to 1'b1 that cannot be mapped to other ports
	QIO2	bidirectional	PD14CFG=4'h0	0	1	0	0	PD14 is used by default, PMD.bit14 must be set to 1'b1 that cannot be mapped to other ports
	QIO3	bidirectional	PD15CFG=4'h0	0	1	0	0	PD15 is used by default, PMD.bit15 must be set to 1'b1 that cannot be mapped to other ports

Chapter 3 System Structure

3.1 Overview

This product system consists of the following components:

- 2 AHB buses master:

- Cortex-M0+

- Enhanced DMA

- 5 AHB buses slaves:

- FLASH storage

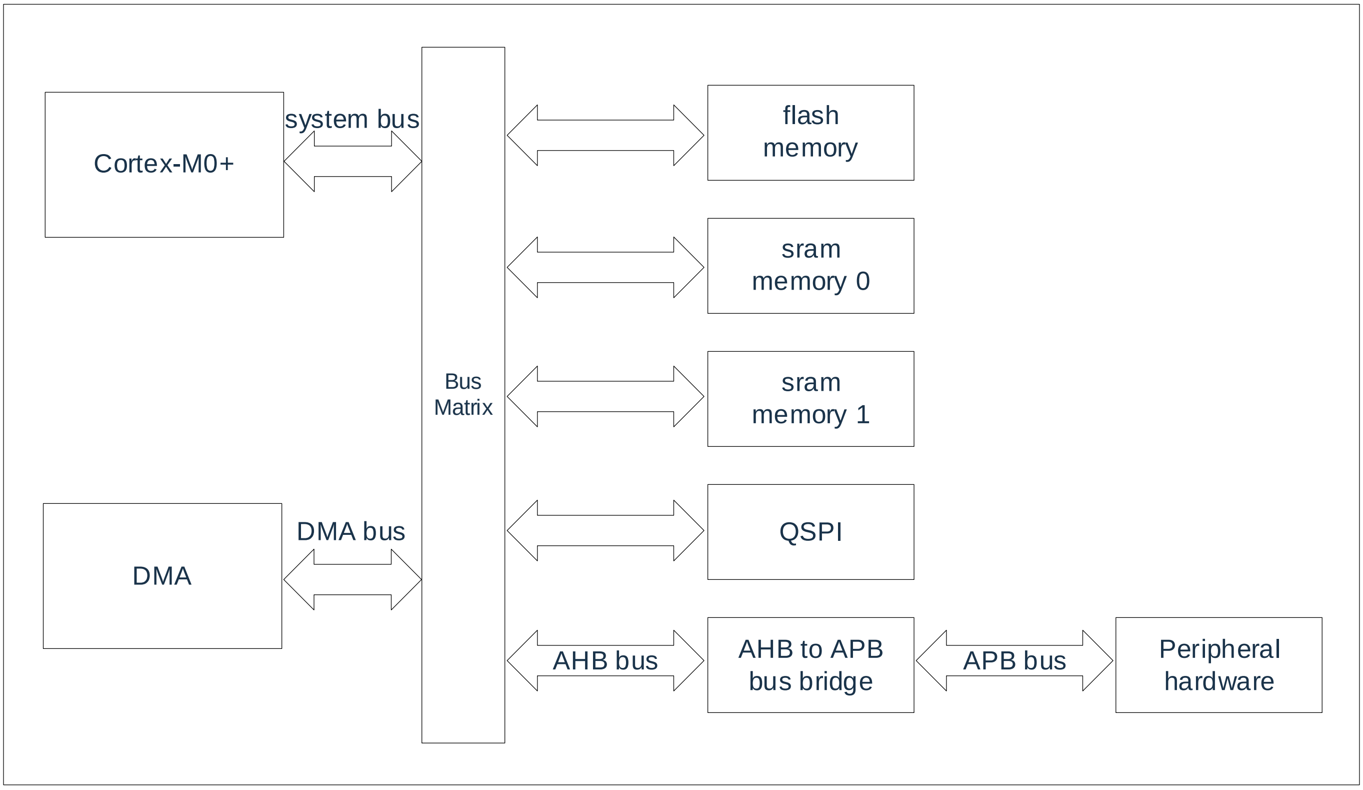
- SRAM memory 0

- SRAM memory 1

- QSPI

- AHB to APB Bridge, contains all APB interface peripherals

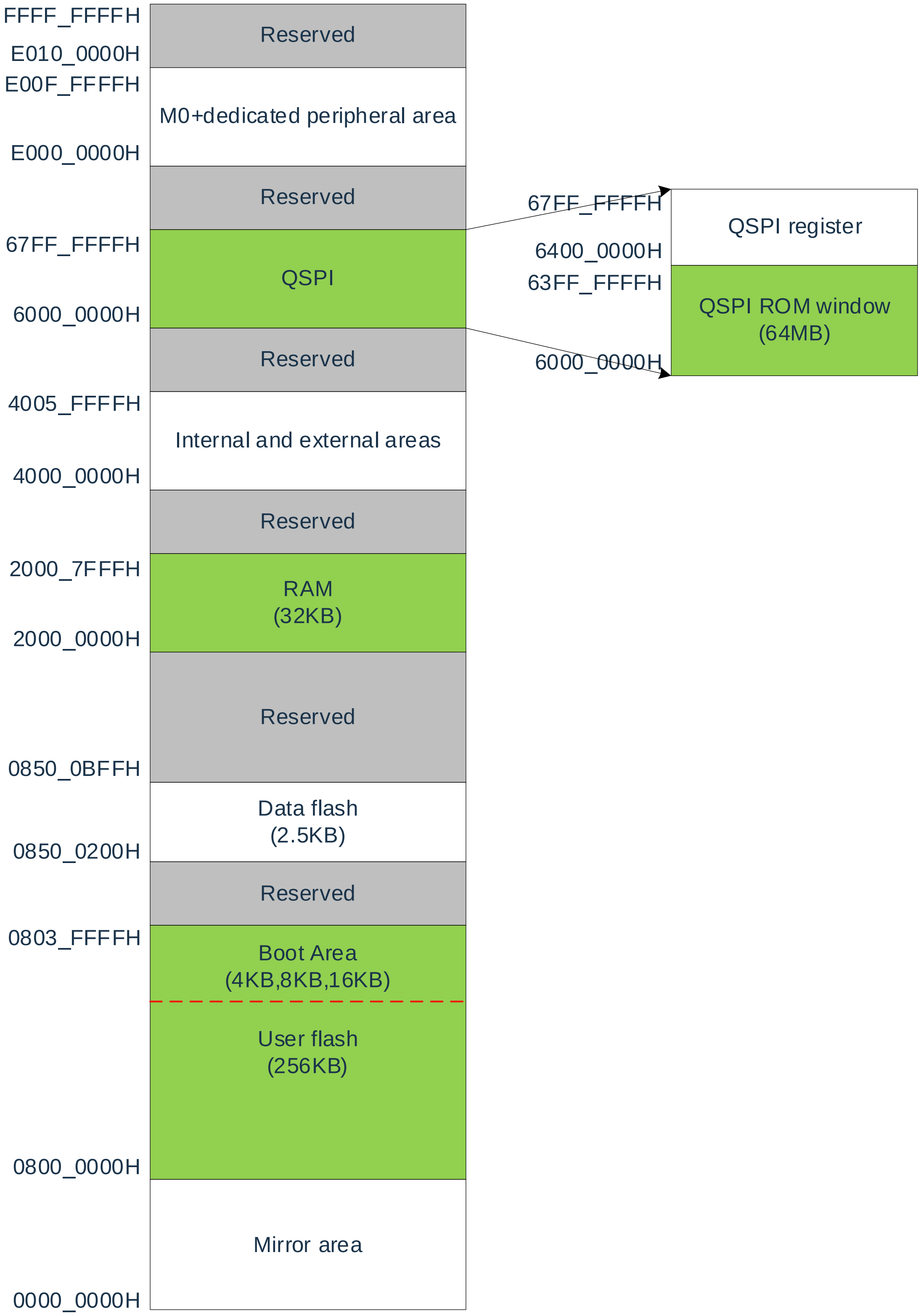
Figure3-1 System block diagram



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

3.2 System address partitioning

Figure3-2 Map of address area



#Note: The green area can be remap to the image area.

Peripheral address assignment

Table 3-1 Register group start address for peripheral

Start Address	Peripheral	Remark
0x4000_0000 - 0x4000_4FFF	Reserved	
0x4000_5000 - 0x4000_5FFF	DMA	
0x4000_6000 - 0x4000_6FFF	Interrupt control	
0x4000_7000 - 0x4001_8FFF	Reserved	
0x4001_9000 - 0x4001_AFFF	Reserved	
0x4001_B000 - 0x4001_BFFF	DBGREG	
0x4001_C000 - 0x4001_CFFF	DIV	
0x4001_D000 - 0x4001_FFFF	Reserved	
0x4002_0000 - 0x4002_03FF	FLASH control	
0x4002_0400 - 0x4002_0FFF	Clock control	
0x4002_1000 - 0x4002_1001	Watchdog timer	
0x4002_1002 - 0x4002_1800	Reserved	
0x4002_1800 - 0x4002_1BFF	High speed CRC	See Chapter 31 Safety Function for details
0x4002_1C00 - 0x4002_1FFF	FLASH control	
0x4002_2000 - 0x4003_FFFF	Reserved	
0x4004_0000 - 0x4004_0FFF	GPIO	
0x4004_1000 - 0x4004_15FF	Serial communication unit	
0x4004_1A00 - 0x4004_1CFF	Serial interface IICA0	
0x4004_1D00 - 0x4004_21FF	Universal timer unit	
0x4004_2200 - 0x4004_23FF	Reserved	
0x4004_2400 - 0x4004_2BFF	SPIHS0/1	
0x4004_2C00 - 0x4004_2FFF	Serial interface IICA1	
0x4004_3000 - 0x4004_31FF	reserve	
0x4004_3200 - 0x4004_32FF	Universal CRC	See Chapter 31 Safety Function for details
0x4004_3300 - 0x4004_33FF	Reserved	
0x4004_3400 - 0x4004_37FF	Linkage controller	
0x4004_3800 - 0x4004_3BFF	Comparator/amplifier	
0x4004_3C00 - 0x4004_3FFF	Reserved	
0x4004_4400 - 0x4004_47FF	EPWM	
0x4004_4800 - 0x4004_4EFF	Reserved	
0x4004_4F00 - 0x4004_4FFF	Real time clock	
0x4004_5000 - 0x4004_53FF	AD converter	
0x4004_5400 - 0x4004_54FF	LCDB	
0x4004_5500 - 0x4004_5AFF	Reserved	
0x4004_5B00 - 0x4004_5BFF	External interrupt control	
0x4008_0000 - 0x4008_FFFF	USB	
0x4009_0000 - 0x4009_FFFF	SSI	

Start Address	Peripheral	Remark
0x400A_0000 - 0x5FFF_FFFF	Reserved	
0x6400_0000 - 0x67FF_FFFF	QSPI	

3.3 Start configuration

The product can select three different boot modes through the configuration of user option bytes.

Table 3-2 Select addresses for boot mode

CONFIG[2:0] NOTE	The program starts the address setting
001	Boot zone starts
010	Extend flash area starts
011	The RAM zone starts
Other than the above	The main flash area boots

Depending on the selected boot address, the main flash memory area, boot area, extend flash and RAM can be accessed as follows:

Boot from the main flash memory area: The main flash memory area is mapped to the boot space (0x0000_0000), but can still access it at its original address (0x0800_0000), that is, the contents of the flash memory can be accessed in both address areas, 0x0000_0000 Or 0x0800_0000.

Boot from boot zone: The boot zone is mapped to the boot space (0x0000_0000), but can still be 3_C000H/0x080 at its original address (according to BOOTSIZE settings 0x0803_E000/0x0803_F000-0x080 3_FFFFH) to access it.

Boot from built-in SRAM: The SRAM zone is mapped to boot space (0x0000_0000), but can still access SRAM at its original address 0x2000_0000 the address zone it started.

Start from extension flash: The extendedFFLASH area is mapped to the launch space (0x0000_0000), but can still be accessed in the address zone starting at its original address 0x 6000_0000 exteend flash.

Note: For details on config and boot size, see "32.2 Format of user option bytes".

3.4 Start switching

If the program needs to switch between the boot areas, it needs to write the corresponding value using the control register BOOTCON, and then perform a software reset or generate a watchdog reset.

During power-on reset, external reset, and voltage reset, the BOOTCON reset value is 0x0000, and the software reset and watchdog reset cannot clear this register.

Table 3-3 BOOT control register (BOOTCON)

Address: 0x40021C04 after reset: 0000H R/W

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOOTCON	BOOTCON[15:0]															

BOOTCON	Boot zone settings
F155	If you switch from another AREA to the main flash memory area, you need to write 0xf155 to it and then perform a software reset or generate a watchdog reset
F1AA	If switching from another area to the BOOT zone, write 0xf1aa to it and then perform a software reset or generate a watchdog reset
F15A	If switching from anotherA REA to the extend flash area, write 0xf15A to it, and then perform a software reset or generate a watchdog reset
F1A5	If switching from anotherrea to the RAM area, write 0xf1a5 to it, and then perform a software reset or generate a watchdog reset

Chapter 4 Clock Generation Circuit

The presence of resonator connection pin/external clock input pin for the main system clock and the resonator connection pin/external clock input pin for the secondary system clock are different among products.

4.1 Function of clock generation circuit

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

(1) Main system clock

① X1 oscillating circuit

The X1 pin and X2 pin can oscillate a clock with f_X=1 to 20MHz by connecting the X1 pin and the X2 pin to the resonator.

② High speed internal oscillator (high-speed OCO)

Can oscillate from fHOCO=64MHz, 48MHz, 32MHz, 24MHz, 16MHz, 12MHz, 8MHz, 6MHz, 4MHz, 3MHz, 2MHz and 1MHz (TYP.) via option bytes (000C2H). When using high-speed internal oscillator clock as USB clock, fHOCO=48MHz must be selected. After the reset is removed, the CPU must start running with this high speed internal oscillator clock. The oscillation can be stopped by Inputing a deep sleep mode or by setting a HIOSTOP bit (bit0 of a CSC register). The frequency at which option byte settings can be changed through a HOCODIV (Frequency Selection Register) of a high speed internal oscillator. For frequency settings, refer to "Figure4-10 Format of ".

③ PLL circuit

The 2-way PLL circuit provides the system clock with the highest 64MHz clock and USB with the 48MHz clock through the frequency multiplication X1 oscillation circuit or the high-speed internal oscillator circuit. The PLL control register PLLCR/UPLLCR is used to control the PLL oscillation and stop.

In addition, the external main system clock (f_{EX}=1~20MHz) can be provided by the EXCLK/X2/P122 pin, and the input of the external main system clock can be set invalid by entering deep sleep mode or setting MSTOP bit.

Select the high-speed internal oscillator clock and PLL clock by setting the CKSELR bit (bit0 of the master system clock control register MCKC), and then switch the high-speed system clock (X1 clock or external master system clock) to the high-speed internal oscillator clock or PLL clock by setting the MCM0 bit (bit4 of the system clock control register CKC).

(2) Subsystem clock

- XT1 oscillating circuit

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

In addition, the external sub-system clock (f_{EXS}=32.768kHz) can be provided by the EXCLKS/XT2/PH4 pin, and the input of the external sub-system clock can be set invalid by setting the XTSTOP bit.

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

It can make the clock oscillate at f_{IL}=15 kHz (TYP.).
A low speed internal oscillator clock canbe used as a CPU clock.
The SysTick timer use a low-speed internal oscillator clock as an external reference clock.

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

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

Note: The low-speed internal oscillator clock (f_{IL}) can be selected as the count clock of the real-time clock only when a fixed period interrupt function is used.

Notes: f_X	: X1 clock oscillation frequency
f_{HOCO}	: Clock frequency of high speed internal oscillator
f_{IH}	: Clock frequency Note for high-speed internal oscillator
f_{EX}	: External master clock frequency
f_{PLL}	: PLL clock frequency
f_{XT}	:XT1 clock oscillation frequency
f_{EXS}	: external sub-system clock frequency
f_{IL}	: Clock frequency of low speed internal oscillator

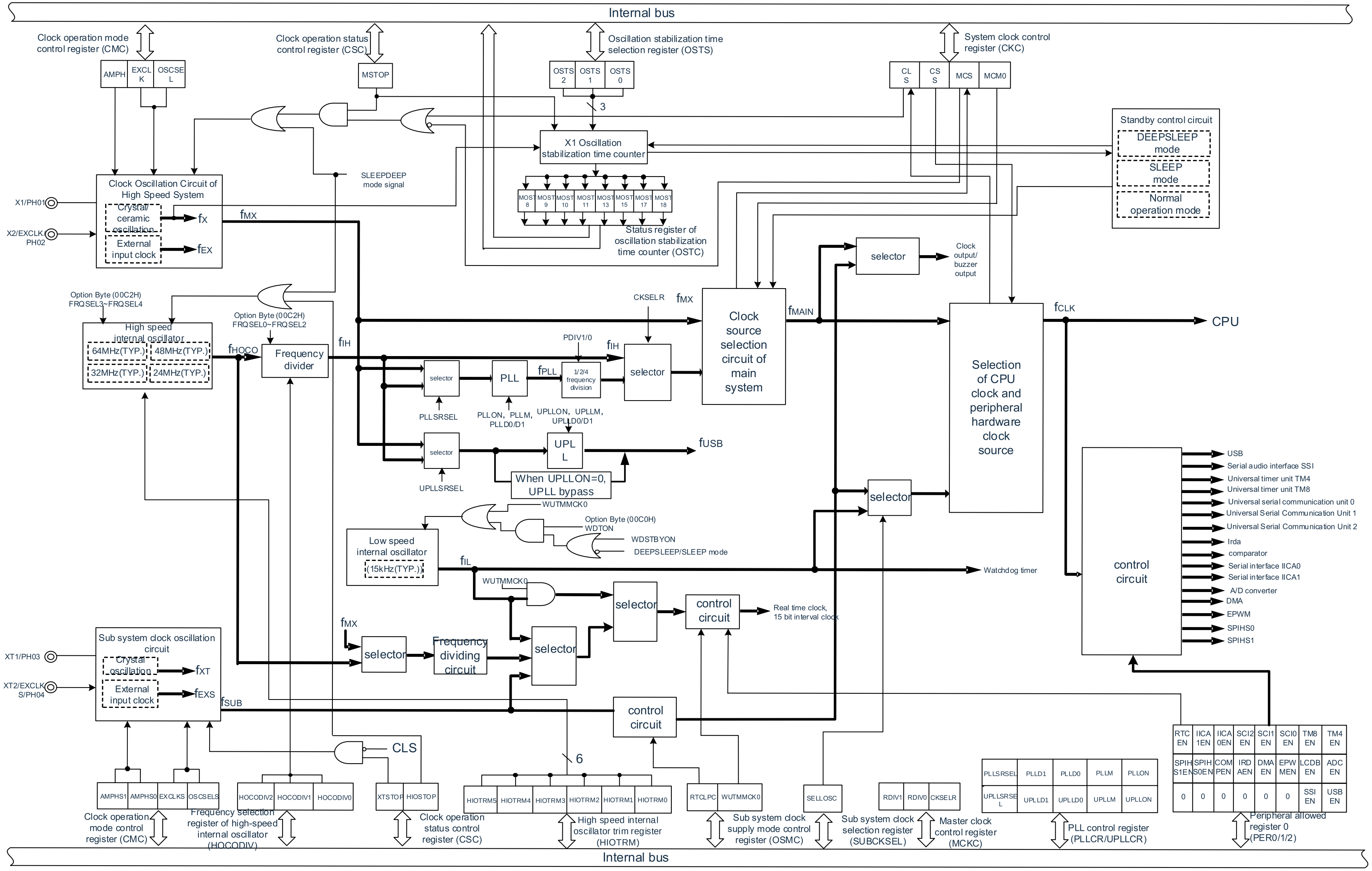
4.2 Structure of clock generating circuit

The clock generating circuit is composed of the following hardware.

Table 4-1 Structure of clock generation circuit

Item	Structure
Control register	Clock operation mode control register (CMC) System clock control register (CKC) Clock operation status control register (CSC) PLL control register for system clock (PLLCR) PLL control register for USB (UPLLCR) Oscillation stabilization time counter status register (OSTC) Oscillation stabilization time select register (OSTS) Peripheral enable registers 0, 1, 2 (PER0, PER1, PER2) Subsystem clock supply mode control register (OSMC) High-speed internal oscillator frequency selection register (HOCODIV) High-speed internal oscillator trim register (HIOTRM) Subsystem clock selection register (SUBCKSEL) Master system clock control register (MCKC)
Oscillating circuit	X1 oscillation circuit XT1 oscillation circuit High-speed internal oscillator Low-speed internal oscillator 2-channel PLL oscillator

Figure 4-1 Block diagram of clock generating circuit


Internal bus
Clock operation mode control register (CMC)
AMPH
EXCLK
OSCSEL
Clock operation status control register (CSC)
MSTOP
Oscillation stabilization time selection register (OSTS)
OSTS2
OSTS1
OSTS0
3
System clock control register (CKC)
CLS
CSS
MCS
MCM0
Standby control circuit
DEEPSLEEP mode
SLEEP mode
Normal operation mode
X1 Oscillation stabilization time counter
SLEEPDEEP mode signal
X1/PH01
X2/EXCLK/PH02
Clock Oscillation Circuit of High Speed System
Crystal/ceramic oscillation
External input clock
fX
fEX
fMX
MOST8
MOST9
MOST10
MOST11
MOST13
MOST15
MOST17
MOST18
Status register of oscillation stabilization time counter (OSTC)
selector
Clock output/buzzer output
Option Byte (00C2H) FRQSEL3~FRQSEL4
High speed internal oscillator
64MHz(TYP.)
48MHz(TYP.)
32MHz(TYP.)
24MHz(TYP.)
fHOCO
Option Byte (00C2H) FRQSEL0~FRQSEL2
Frequency divider
fIH
CKSELR
fMX
PDIV1/0
fIH
selector
PLL
fPLL
1/2/4 frequency division
selector
PLLSRSEL
PLLON, PLLM, PLLD0/D1
UPLLON, UPLLM, UPLLD0/D1
Clock source selection circuit of main system
fMAIN
Selection of CPU clock and peripheral hardware clock source
fCLK
CPU
selector
UPLL
fUSB
UPLLSRSEL
When UPLLON=0, UPLL bypass
WUTMMCK0
Option Byte (00C0H) WDTON
WDSTBYON
DEEPSLEEP/SLEEP mode
Low speed internal oscillator
(15kHz(TYP.))
fIL
Watchdog timer
selector
WUTMMCK0
selector
control circuit
Real time clock, 15 bit interval clock
fMX
selector
Frequency dividing circuit
selector
control circuit
USB
Serial audio interface SSI
Universal timer unit TM4
Universal timer unit TM8
Universal serial communication unit 0
Universal Serial Communication Unit 1
Universal Serial Communication Unit 2
Irda
comparator
Serial interface IICA0
Serial interface IICA1
A/D converter
DMA
EPWM
SPIHS0
SPIHS1
XT1/PH03
XT2/EXCLKS/PH04
Sub system clock oscillation circuit
Crystal oscillation
External input clock
fXT
fEXS
fSUB
CLS
6
AMPHS1
AMPHS0
EXCLKS
OSCSELS
Clock operation mode control register (CMC)
HOCODIV2
HOCODIV1
HOCODIV0
Frequency selection register of high-speed internal oscillator (HOCODIV)
XTSTOP
HIOSTOP
Clock operation status control register (CSC)
HIOTRM5
HIOTRM4
HIOTRM3
HIOTRM2
HIOTRM1
HIOTRM0
High speed internal oscillator trim register (HIOTRM)
RTCLPC
WUTMMCK0
Sub system clock supply mode control register (OSMC)
SELLOSC
Sub system clock selection register (SUBCKSEL)
RDIV1
RDIV0
CKSELR
Master clock control register (MCKC)
PLLSRSEL
PLLD1
PLLD0
PLLM
PLLON
UPLLSRSEL
UPLLD1
UPLLD0
UPLLM
UPLLON
PLL control register (PLLCR/UPLLCR)
RTCEN
IICA1EN
IICA0EN
SCI2EN
SCI1EN
SCI0EN
TM8EN
TM4EN
SPIHS1EN
SPIHS0EN
COMPEN
IRDAEN
DMAEN
EPWMEN
LCDBEN
ADCEN
0
0
0
0
0
0
SSIEN
USBEN
Peripheral allowed register 0 (PER0/1/2)

Remark f_X : X1 clock oscillation frequency

f_{HOCO} : Clock frequency of high speed internal oscillator

f_{IH} : Clock frequency of high-speed internal oscillator

f_{EX} : External master clock frequency

f_{MX} : high speed system clock frequency

f_{MAIN} : main system clock frequency

f_{XT} :XT1 clock oscillation frequency

f_{EXS} : external sub-system clock frequency

f_{SUB} : sub-system clock frequency

f_{CLK} : Clock frequency of the CPU/peripheral hardware

f_{IL} : Clock frequency of low speed internal oscillator

4.3 Registers for controlling clock generation circuit

The clock generation circuit is controlled by the following registers.

- Clock operation mode control register (CMC)
- System clock control register (CKC)
- Clock operation status control register (CSC)
- PLL control register for system clock (PLLCR)
- PLL control register for USB (UPLLCR)
- Oscillation stabilization time counter status register (OSTC)
- Oscillation stabilization time selection register (OSTS)
- Peripheral enable registers 0, 1, 2(PER0, PER1, PER2)
- Subsystem clock supply mode control register (OSMC)
- High-speed internal oscillator frequency selection register (HOCODIV)
- High-speed internal oscillator trim register (HIOTRM)
- Subsystem clock selection register (SUBCKSEL)
- Master clock control register (MCKC)

Note: The assigned registers and bits differ depending on the product. You must set an initial value for unassigned bits.

4.3.1 Clock operation mode control register (CMC)

This is a register that sets the operation mode of the X1/PH1, X2/EXCLK/PH2, XT1/PH3, XT2/EXCLKS/PH4 pin and selects the gain of the oscillating circuit.

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

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

Figure4-2 Format of clock operation mode control register (CMC)

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

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

EXCLK	OSCSEL	High speed system clock pin operation mode	X1/PH1 Pin	X2/EXCLK/PH2 Pin
0	0	Port mode	Input/output port	
0	1	X1 oscillation mode	Connect a crystal or ceramic resonator.	
1	0	Port mode	Input/output port	
1	1	External clock input mode	Input/output port	External clock input

EXCLKS	OSCELS	Subsystem clock pin operation mode	XT1/PH3 Pin	XT2/EXCLKS/PH4 pin
0	0	Port mode	Input/output port	
0	1	XT1 oscillation mode	Connect a crystal resonator.	
1	0	Port mode	Input/output port	
1	1	External clock input mode	Input/output port	External clock input

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

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

Note: The EXCLKS bit, OSCSELS bit, AMPHS1 bit, and AMPHS0 bit are initialized only when the power is reset, and remain unchanged when the other reset is reset.

Note 1. After the reset is removed, the CMC register can only be written 1 time through the 8-bit memory instruction. When using CMC registers with an initial value (00H), to prevent malfunctions when programs are out of control (if values other than "00H" cannot be recovered)

2. The CMC register must be set after the reset is removed and before X1 or XT1 oscillations are started by setting clock run status control registers.

3. The AMPH must be set to "1" when the X1 clock oscillates above 10 MHz.

4. AMPH bits, AMPHS1 bits, and AMPHS0 bits must be set after the reset is removed and under f_{IH} as f_{CLK} (switch f_{CLK} to f_{MX} or pre f_{SUB}).

5. The stability time of fXT must be counted by software.

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

Remark: f_X: X1 clock oscillation frequency

4.3.2 System clock control register (CKC)

This is a register that selects the CPU/peripheral hardware clock and the main system clock. The CKC register is set by an 8-bit memory operation instruction.

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

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

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

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

CLS	CPU/Peripheral Hardware Clock (f_{CLK}) Status
0	Main System Clock (f_{MAIN})
1	Secondary System Clock (f_{SUB})

CSS Note 2	CPU/Peripheral Hardware Clock (f_{CLK}) Selection
0	Main System Clock (f_{MAIN})
1	Secondary System Clock (f_{SUB})

MCS	Status of the main system clock (f_{MAIN})
0	high speed internal oscillator clock (f_{IH})
1	High speed system clock (f_{MX})

MCM0 Note 2	Main System Clock (f_{MAIN}) Operation Control
0	A high speed internal oscillator clock (f_{IH}) is selected as the main system clock (f_{MAIN}).
1	Select the high speed system clock (f_{MX}) as the primary system clock (f_{MAIN}).

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

2. It is prohibited to change the value of the MCM0 bit while the CSS bit is set to "1".

Remark: f_{HOCO}: clock frequency of high speed internal oscillator

f_{IH}: clock frequency of high-speed internal oscillator

f_{MX}: high speed system clock frequency

f_{MAIN}: main system clock frequency

f_{SUB}: sub-system clock frequency

Note: 1. Bit0~3 must be set to 0.

2. Provides CSS bit setting clocks for the CPU and peripheral hardware. If you change the CPU clock, change the peripheral hardware clock at the same time (except for real-time clocks, 15-bit interval timers, clock output/buzzer output, and watchdog timer). Therefore, if you want to change the clock on the CPU/peripheral hardware, you must stop the peripheral functions.

3. If the secondary system clock is used as the peripheral hardware clock, the A/D converter and IICA cannot be guaranteed. For operation characteristics of the peripheral hardware, refer to the electrical characteristics of the sections and datasheet for each peripheral hardware.

4.3.3 Clock operation status control register (CSC)

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

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

Figure 4-4 Format of clock operation status control register (CSC)

Address: 40020401H After reset: C0H R/W

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

MSTOP	Operation Control of High Speed System Clock		
	X1 oscillation mode	external clock input mode	port mode
0	Operation of X1 oscillator circuit	External clock valid for EXCLK pin	Input/output port
1	The X1 oscillation circuit stops	The external clock for the EXCLK pin is invalid	

XTSTOP	Operation Control of Sub-system Clock		
	XT1 oscillation mode	external clock input mode	port mode
0	XT1 Oscillation circuit operation	External clock valid for EXCLKS pin	Input/output port
1	XT1 oscillation circuit stop	The external clock for the EXCLKS pin is invalid	

HIOSTOP	Operation control of high speed internal oscillator clock
0	high speed internal oscillator operation
1	high speed internal oscillator stop

Note:

1. After the reset is removed, the CSC register must be set after setting the Clock Run Mode Control Register (CMC).
2. After the reset is removed and before the MSTOP bit set to "0", an oscillatory stabilization time selection register (OSTS) must be set. However, when using the OSTS register at the initial value, you do not need to set the OSTS register.
3. When X1 oscillation is started by setting the MSTOP bit, the oscillation stability time of the X1 clock must be confirmed by OSTC.
4. When you want to start XT1 oscillation by setting the XSTOP bit, you must wait for the oscillation stabilization time required by the secondary system clock through software.
5. The clock selected as the CPU/peripheral hardware clock (f_{CLK}) cannot be stopped through the CSC register.
6. Refer to Table 4-2 for register flag settings for stopping clock oscillation (invalid external clock input) and conditions before stopping.

Table 4-2 Clock stopping method

Clock	Condition before clock stops (invalid external clock input)	Set the flag for the CSC register
X1 clock	The CPU/peripheral hardware clock runs at a clock other than the high speed system clock. (CLS=0 and MCS=0, or CLS=1)	MSTOP=1
external main system clock		
XT1 clock	The CPU/peripheral hardware clock runs at a clock other than the secondary system clock. (CLS=0)	XTSTOP=1
external sub-system clock		
high speed internal oscillator clock	The CPU/peripheral hardware clock operates at a clock other than the high speed internal oscillator clock. (CLS=0 and MCS=1, or CLS=1)	HIOSTOP=1

4.3.4 PLL control register for system clock (PLLCR)

This is the register that controls the clock of the system to run with PLL. The PLLCR register is set by an 8-bit memory operation instruction.

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

Figure 4-5 Format of PLL control register for system clock (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	Input clock source selection of PLL
0	Select the high-speed internal oscillator clock F_{IH} as the input clock source of PLL
1	Select the external master system clock F_{MX} as the input clock source of PLL

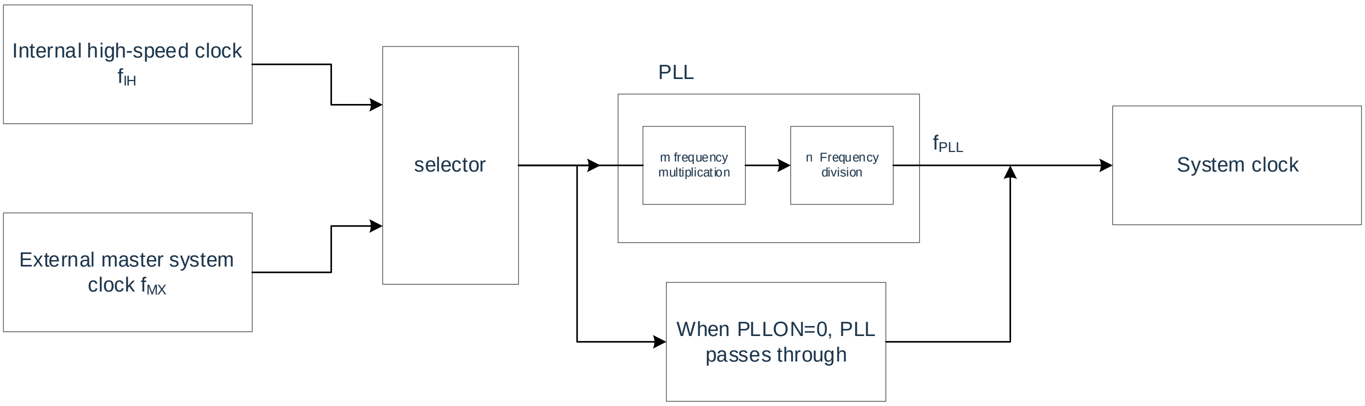
PLLD1	PLLD0	PLL frequency division selection
0	0	Undivided frequency
0	1	Frequency is divided by 2
1	x	Frequency is divided by 4

PLLM	PLL frequency multiplication selection
0	12x frequency
1	16x frequency

PLLON	PLL operation enable
0	PLL does not work
1	PLL is in operation

When using PLL as the system clock, the clock structure is shown in the figure below, where m is 12/16, determined by the setting value of PLLM, and n is 1/2/4, determined by the setting value of PLLD1/PLLD0.

Figure 4-6 Clock structure when PLL is used as system clock


Internal high-speed clock f_{IH}
External master system clock f_{MX}
selector
PLL
m frequency multiplication
n Frequency division
f_{PLL}
System clock
When PLLON=0, PLL passes through

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

4.3.5 PLL control register for USB (UPLLCR)

This is the register that controls the system clock to run with UPLL. The UPLLCR register is set by an 8-bit memory operation instruction.

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

Figure 4-7 Format of PLL control register for USB (UPLLCR)

Address: 40020C03H After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
UPLLCR	UPLLSRSEL	0	0	0	UPLLD1	UPLLD0	UPLLM	UPLLON

UPLLSRSEL	Input clock source selection of UPLL
0	Select the high-speed internal oscillator clock f_{IH} as the input clock source of UPLL
1	Select the external master system clock f_{MX} as the input clock source of UPLL

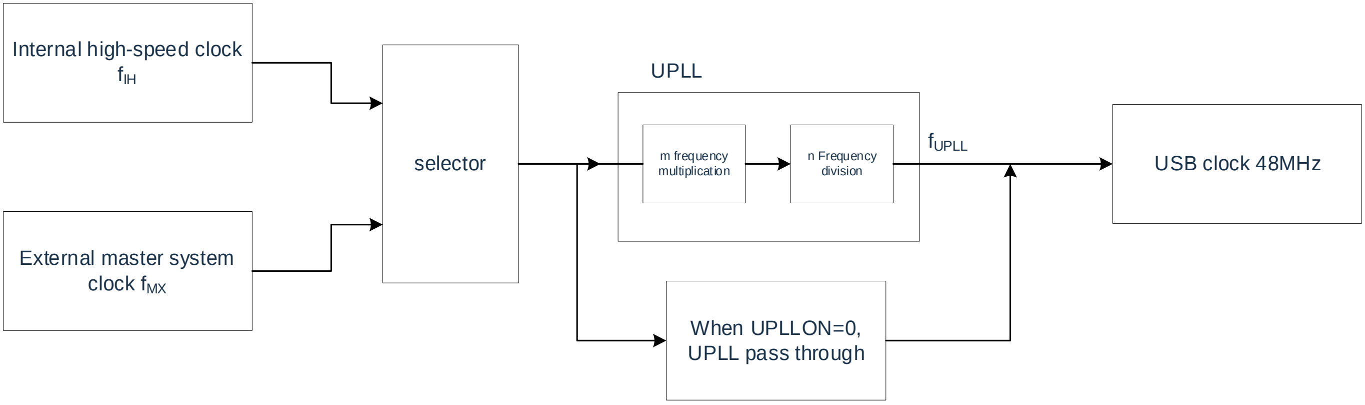
UPLLD1	UPLLD0	UPLL frequency division selection
0	0	Undivided frequency
0	1	Frequency is divided by 2
1	x	Frequency is divided by 4

UPLLM	UPLL frequency multiplication selection
0	12x frequency
1	16x frequency

UPLLON	UPLL work enable
0	PLL does not work
1	PLL is in operation

The clock structure of USB clock is shown in the figure below, where m is 12/16, determined by the setting value of PLLM, and n is 1/2/4, determined by the setting value of PLLD1/PLLD0.

Figure 4-8 Clock structure of USB clock


Internal high-speed clock f_{IH}
External master system clock f_{MX}
selector
UPLL
m frequency multiplication
n Frequency division
f_{UPLL}
USB clock 48MHz
When UPLLON=0, UPLL pass through

An example of USB clock setting is shown in the following table.

Internal high-speed clock fIH or external master system clock fMX	UPLL settings		USB clock
	m frequency multiplication	n frequency division	
4M	12 times	Divided by 1	48MHz
6M	16 times	Divided by 2	48MHz
8M	12 times	Divided by 2	48MHz

4.3.6 Oscillation stabilization time counter status register (OSTC)

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

When the CPU clock is a high speed internal oscillator clock or a sub-system clock and the oscillation of the X1 clock is started

When the CPU clock is a high speed internal oscillator clock and the sleep mode is released after transferring to deep sleep mode in the X1 clock oscillation state

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

By generating a reset signal, entering deep sleep mode or MSTOP bit (bit7 of clock running status control register (CSC) is 1.

Note: The oscillation steady-time counter starts counting when:

· When the X1 clock starts to oscillate (EXCLK, OSCSEL=0, 1→MSTOP=0)

· When deep sleep mode is released

Figure4-9 Format of oscillation stabilization time counter status register (OSTC)

Address: 40020402H After reset: 00H R

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

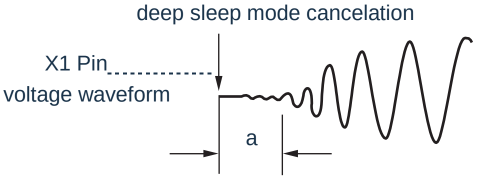
MOST8	MOST9	MOST10	MOST11	MOST13	MOST15	MOST17	MOST18	oscillation stabilization time status		
									f_X=10MHz	f_X=20MHz
0	0	0	0	0	0	0	0	Less than 28/f_X	less than 25.6μs	less than 12.8μs
1	0	0	0	0	0	0	0	At least $2^8/f_X$	At least 25.6μs	At least 12.8μs
1	1	0	0	0	0	0	0	At least $2^9/f_X$	At least 51.2μs	At least 25.6μs
1	1	1	0	0	0	0	0	At least $2^{10}/f_X$	At least 102μs	At least 51.2μs
1	1	1	1	0	0	0	0	At least $2^{11}/f_X$	At least 204μs	At least 102μs
1	1	1	1	1	0	0	0	At least $2^{13}/f_X$	At least 819μs	At least 409μs
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 time mentioned above, you change from MOST8 to "1" and remain in "1".

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

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

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



Note: f_X: X1 clock oscillation frequency.

4.3.7 Oscillation stabilization time select register (OSTS)

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

If the X1 clock is oscillated, the time set by the OSTS register is automatically waited after the X1 oscillation circuit (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 is switched to a deep sleep mode.

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

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

Figure4-10 Format of oscillation stabilization time select register (OSTS)

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

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

OSTS2	OSTS1	OSTS0	Selection of oscillation stabilization time		
				f_X=10MHz	f_X=20MHz
0	0	0	$2^8/f_X$	25.6μs	12.8μs
0	0	1	$2^9/f_X$	51.2μs	25.6μs
0	1	0	$2^{10}/f_X$	102μs	51.2μs
0	1	1	$2^{11}/f_X$	204μs	102μs
1	0	0	$2^{13}/f_X$	819μs	409μs
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

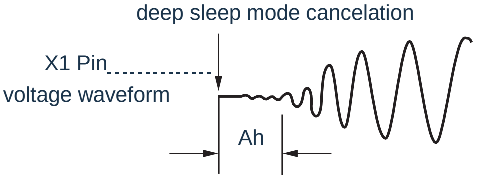
Note:1. To change the setting of the OSTS register, you must make the change before the MSTOP bit of the Clock Run Status Control Register (CSC) set to 0.

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

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

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

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



Note: f_X:X1 clock oscillation frequency

4.3.8 Peripheral enable registers 0, 1, 2 (PER0, PER1, PER2)

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

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

- Real-time clock, 15-bit interval timer
- A/D converter
- Serial interface IICA1
- Serial interface IICA0
- Universal serial communication unit 2
- Universal serial communication unit 1
- Universal serial communication unit 0
- Universal timer unit TM8
- Universal timer unit TM4
- High speed SPI unit 1
- High speed SPI unit 0
- Serial audio interface
- USB
- LCD BUS I/F
- D/A converter
- Comparator
- Enhanced DMA

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

After the reset signal is generated, the values of these registers change to '00H'.

Figure 4-11 Format of peripheral enable register 0 (PER0) (1/3)

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

symbol	7	6	5	4	3	2	1	0
PER0	RTCEN Note	IICA1EN	IICA0EN	SCI2EN	SCI1EN	SCI0EN	TM8EN	TM4EN

RTCEN	Control of an input clock of a real-time clock (RTC) and a 15-bit interval timer
0	Stop provide an input clock. · Cannot write real time clock(RTC) and the SFR used by the and 15-bit interval timers. · The real-time clock (RTC) and 15-bit interval timer are reset.
1	Provides an input clock. · Read and write to the SFR used by the real-time clock (RTC) and 15-bit interval timer.

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

Figure 4-11 Format of peripheral enable register 0 (PER0) (2/3)

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

symbol	7	6	5	4	3	2	1	0
PER0	RTCEN Note	IICA1EN	IICA0EN	SCI2EN	SCI1EN	SCI0EN	TM8EN	TM4EN

IICA1EN	Provides control of input clock for serial interface IICA1
0	Stop provide an input clock. · Cannot write the SFR used by IICA1. · IICA1 is in a reset state.
1	Provides an input clock. · Can read and write SFRs used by IICA1.

IICA0EN	Provides control of input clock for serial interface IICA0
0	Stop provide an input clock. · Cannot write the SFR used by IICA0. · IICA0 is in a reset state.
1	Provides an input clock. · Can read and write SFRs used by IICA0.

SCI2EN	Control of an input clock of a universal serial communication unit 2 is provided
0	Stop provide an input clock. · Cannot write the SFR used by Universal Serial Communication Unit 2. · The universal serial communication unit 2 is in a reset state.
1	Provides an input clock. · Read and write the SFR used by the Universal Serial Communication Unit 2.

SCI1EN	Control of an input clock of a universal serial communication unit 1 is provided
0	Stop provide an input clock. · Cannot write the SFR used by Universal Serial Communication Unit 1. · The universal serial communication unit 1 is in a reset state.
1	Provides an input clock. · Read and write the SFR used by the Universal Serial Communication Unit 1.

SCI0EN	Control of an input clock of a universal serial communication unit 0 is provided
0	Stop provide an input clock. · Cannot write the SFR used by the Universal Serial Communication Unit 0. · Universal serial communication unit 0 is in reset state.
1	Provides an input clock. · Read and write the SFR used by the Universal Serial Communication Unit 0.

Figure 4-11 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 Note	IICA1EN	IICA0EN	SCI2EN	SCI1EN	SCI0EN	TM8EN	TM4EN

TM8EN	Provide control of input clock of general timer unit TM8
0	Stop providing input clock. • SFR used by general timer unit TM8 cannot be written. • Universal timer unit TM8 is in reset state.
1	Provide input clock. • Can read and write SFR used by general timer unit TM8.

TM4EN	Provide control of input clock of general timer unit TM4
0	Stop providing input clock. • SFR used by general timer unit TM4 cannot be written. • Universal timer unit TM4 is in reset state.
1	Provide input clock. • Can read and write SFR used by general timer unit TM4.

Figure4-12 Format of the Peripheral Admission Register 1 (PER1) (1/2)

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

symbol	7	6	5	4	3	2	1	0
PER1	SPIHS1EN	SPIHS0EN	PGACMPEN	-	DMAEN	EPWMEN	LCDBEN	ADCEN

SPIHS1EN	Provide control of input clock of high-speed SPI unit 1
0	Stop providing input clock. • The SFR used by high-speed SPI unit 1 cannot be written. • High speed SPI unit 1 is in reset state.
1	Provide input clock. • Can read and write SFR used by high-speed SPI unit 1.

SPIHS0EN	Provide control of input clock of high-speed SPI unit 0
0	Stop providing input clock. • The SFR used by high-speed SPI unit 0 cannot be written. • High speed SPI unit 0 is in reset state.
1	Provide input clock. • Can read and write SFR used by high-speed SPI unit 0.

PGACMPEN	Control of an input clock of an amplifier and a comparator is provided
0	Stop provide an input clock. · SFRs used by amplifiers and comparators cannot be written. · The amplifier comparator is in a reset state.
1	Provides an input clock. · SFRs that can read and write to the amplifiers and comparators.

DMAEN	Provide input clock control of DMA
0	Stop providing input clock. • DMA cannot operate.
1	Provide input clock. • DMA can operate.

B4	Reserve
0	This bit is read as 0. The write value should be 0.

Figure4-12 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	SPIHS1EN	SPIHS0EN	PGACMPEN	-	DMAEN	EPWMEN	LCDBEN	ADCEN

EPWMEN	Provide control of input clock of EPWM
0	Stop providing input clock. • SFR used by EPWM cannot be written. • EPWM is in reset state.
1	Provide input clock. • Can read and write SFR used by EPWM.

LCDBEN	Provide control of input clock of LCDB
0	Stop providing input clock. • SFR used by LCDB cannot be written. • LCDB is in reset state.
1	Provide input clock. • Can read and write SFR used by LCDB.

ADCEN	Provide control of input clock of A/D converter
0	Stop providing input clock. • SFR used by A/D converter cannot be written. • A/D converter is in reset state.
1	Provide input clock. • Can read and write SFR used by A/D converter.

Figure 4-8 Format of peripheral enable register 2 (PER2)

Address: 4002081BH After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
PER1	0	0	0	0	0	0	SSIEN	USBEN

SSIEN	Provide control of input clock of serial audio interface
0	Stop providing input clock. • SFR used by serial audio interface cannot be written. • The serial audio interface is in reset state.
1	Provide input clock. • Can read and write SFR used by serial audio interface.

USBEN	Provide USB input clock control
0	Stop providing input clock. • SFR used by USB cannot be written. • USB is in reset state.
1	Provide input clock. • SFR that can read and write USB.

4.3.9 Subsystem clock supply mode control register (OSMC)

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

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

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

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

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

Figure4-9 Format of subsystem clock supply mode control register (OSMC)

Address: 40020423H After reset: 00H R/W

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

RTCLPC	Settings in Deep Sleep Mode and Sleep Mode in which the CPU is running on subsystem clock
0	Allow sub-system clock to be provided for peripheral functions (Refer to Table 27-1~Table 27-3 for allowable peripherals.
1	Stop providing that sub-system clock to peripheral function other than the real time clock and 15 bit interval timer.

WUTMMCK0	Selection of a real-time clock, a 15-bit interval timer and an operation clock of timer A
0	• The subsystem clock is the real-time clock and the runtime clock of the 15-bit interval timer.
1	• The low-speed internal oscillator clock is the real-time clock and the runtime clock of the 15-bit interval timer.

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

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

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

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

Figure4-10 Format of high-speed on-chip oscillator frequency select register(HOCODIV)

Address: 40021C20H After reset: Setting value for the FRQSEL2~FRQSEL0 bit of the option byte (000C2H) R/W

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

HOCODIV2	HOCODIV1	HOCODIV0	Selection of high-speed on-chip oscillator frequency			
			FRQSEL4=0		FRQSEL4=1	
			FRQSEL3=0	FRQSEL3=1	FRQSEL3=0	FRQSEL3=1
0	0	0	f_{IH}=24MHz f_{HOCO}=24MHz	f_{IH}=32MHz f_{HOCO}=32MHz	f_{IH}=48MHz f_{HOCO}=48MHz	f_{IH}=64MHz f_{HOCO}=64MHz
0	0	1	f_{IH}=12MHz f_{HOCO}=24MHz	f_{IH}=16MHz f_{HOCO}=32MHz	f_{IH}=24MHz f_{HOCO}=48MHz	f_{IH}=32MHz f_{HOCO}=64MHz
0	1	0	f_{IH}=6MHz f_{HOCO}=24MHz	f_{IH}=8MHz f_{HOCO}=32MHz	f_{IH}=12MHz f_{HOCO}=48MHz	f_{IH}=16MHz f_{HOCO}=64MHz
0	1	1	f_{IH}=3MHz f_{HOCO}=24MHz	f_{IH}=4MHz f_{HOCO}=32MHz	f_{IH}=6MHz f_{HOCO}=48MHz	f_{IH}=8MHz f_{HOCO}=64MHz
1	0	0	Disable setting.	f_{IH}=2MHz f_{HOCO}=32MHz	f_{IH}=3MHz f_{HOCO}=48MHz	f_{IH}=4MHz f_{HOCO}=64MHz
1	0	1	Disable setting.	f_{IH}=1MHz f_{HOCO}=32MHz	Disable setting.	f_{IH}=2MHz f_{HOCO}=64MHz
Others			Disable setting.			

Note 1. The HOCODIV register must be set in a state where the high speed internal oscillator clock ($_{fIH}$) is selected as the CPU/peripheral hardware clock (f_{CLK}).

2. After changing the frequency through the HOCODIV register, the frequency switch is performed after the following transition time:

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

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

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

Note: If the temperature and the voltage of the V_{DD} pin change after the correction accuracy, the frequency changes. In the case where the temperature and the voltage of the V_{DD} pin are varied, it is necessary to perform the correction before or periodically before the required frequency accuracy.

Figure4-11 Format of high-speed on-chip oscillator trimming register (HIOTRM)

Address: 40021C00H After reset: Note R/W

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

HIOTRM5	HIOTRM4	HIOTRM3	HIOTRM2	HIOTRM1	HIOTRM0	High-speed 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	maximum speed

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

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

4.3.12 Subsystem clock selection register (SUBCKSEL)

The SUBCKSEL register is used to select the subsystem clock fSUB and the low speed internal oscillator clock FIL.

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

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

Figure4-12 Format of subsystem clock selection register (SUBCKSEL)

Address: 40020407H After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
SUBCKSEL	0	0	0	0	0	0	0	SELLOSC

SELLOSC	Selection of Sub system Clock and Low-Speed Internal Oscillator Clock
0	• Select the subsystem clock.
1	• Select the low speed internal oscillator clock.

4.3.13 Master system clock control register (MCKC)

The MCKC register is used to control the master system clock.

MCKC register is set by 8-bit memory operation instruction.

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

Figure 4-13 Format of master system clock control register (MCKC)

Address: 40020C00H After reset: 00H R/W[Note1]

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

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

PDIV1	PDIV0	Frequency division selection of PLL clock
0	0	Undivided frequency
0	1	Frequency is divided by 2
1	0	Frequency is divided by 4
1	1	Frequency is divided by 8

CKSELR	Selection of High Speed Internal Oscillator Clock and PLL Clock
0	• Select the high-speed internal oscillator clock.
1	• Select the PLL clock.

Note1: Bit7 is read-only.

4.4 System clock oscillation circuit

4.4.1 X1 oscillation circuit

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

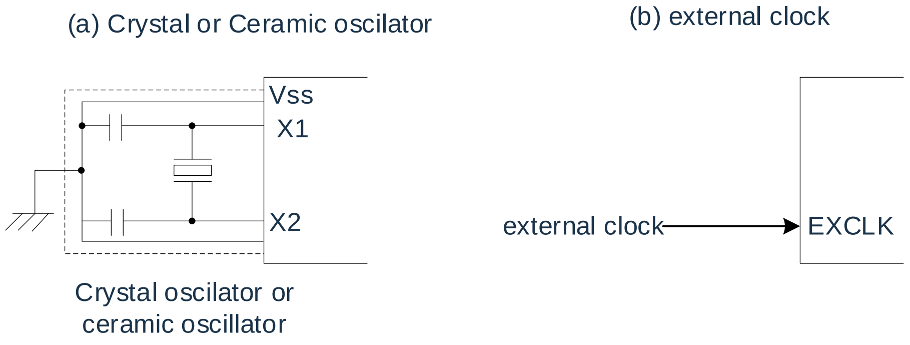
When using the X1 oscillator circuit, the bit7 and bit6 (EXCLK, OSCSEL) of the clock mode control register must be set:

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

When the X1 oscillator circuit is not used, it must be set to port mode (EXCLK, OSCSEL=0, 0). Also, when not used as an input/output port, refer to the "Table 2-5 Handling of each unused port"

Examples of the external circuit of the X1 oscillating circuit are as following Figure.

Figure4-14 Example of external circuit of an X1 oscillating circuit


(a) Crystal or Ceramic oscilator

(b) external clock

The notes are shown on the following page.

4.4.2 XT1 oscillation circuit

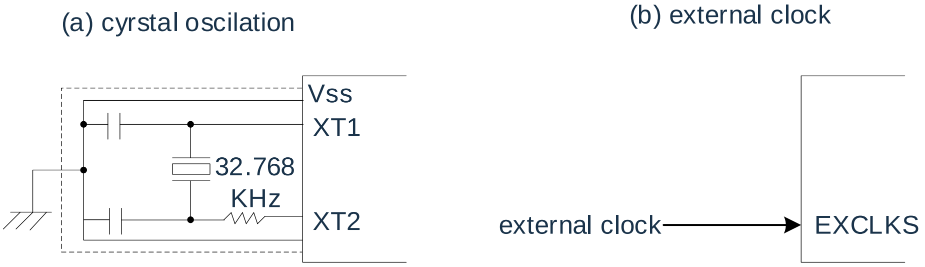
The XT1 oscillation circuit oscillates by a crystal resonator (32.768kHz (TYP.)) connecting the XT1 pin and XT2 pin. When the XT1 oscillating circuit is used, the bit4 (OSCSELS) of the clock operation mode control register (CMC) must be set "1" to input the external clock, and the EXCLKS pin must be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

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

- crystal oscillation: EXCLKS, OSCILS=0,1
- external clock input: EXCLKS, OSCILS=1,1

When the XT1 oscillator circuit is not used, it must be set to port mode (EXCLKS, OSCSELS=0, 0). Also, when not used as an input/output port, refer to “Table 2-5 Handling of each unused port”. Examples of the external circuit of the XT1 oscillating circuit are as following Figure.

Figure 4-15 Example of external circuit of XT1 oscillation circuit


(a) cyrstal oscilation

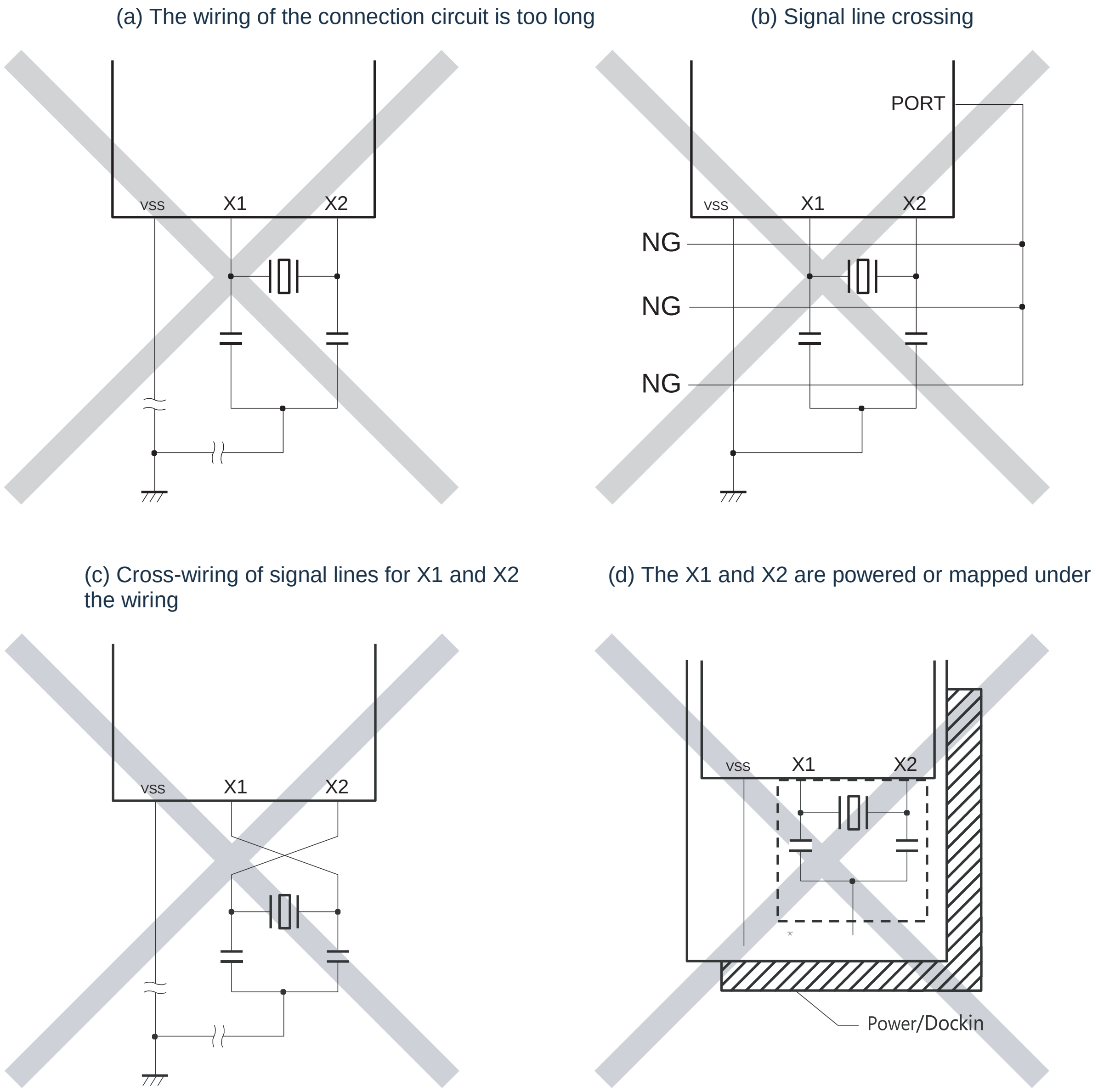
(b) external clock

Note: To avoid the effect of wiring capacitance or the like when using the X1 oscillator circuit and the XT1 oscillator circuit Figure and Figure The dotted section in routes:

- Cabling must be minimized.
- Cannot cross with other signal lines and cannot approach wiring through which a variable high current flows.
- The capacitor and V_{SS} contacts of the oscillating circuit must always be kept at the same potential, and the grounding pattern through which a large current flows must not be grounded.
- The signal cannot be removed from the oscillator circuit.

Incorrect resonator connection examples such as Figure.

Figure 4-16 Examples of incorrect resonator connections (1/2)



(a) The wiring of the connection circuit is too long

(b) Signal line crossing

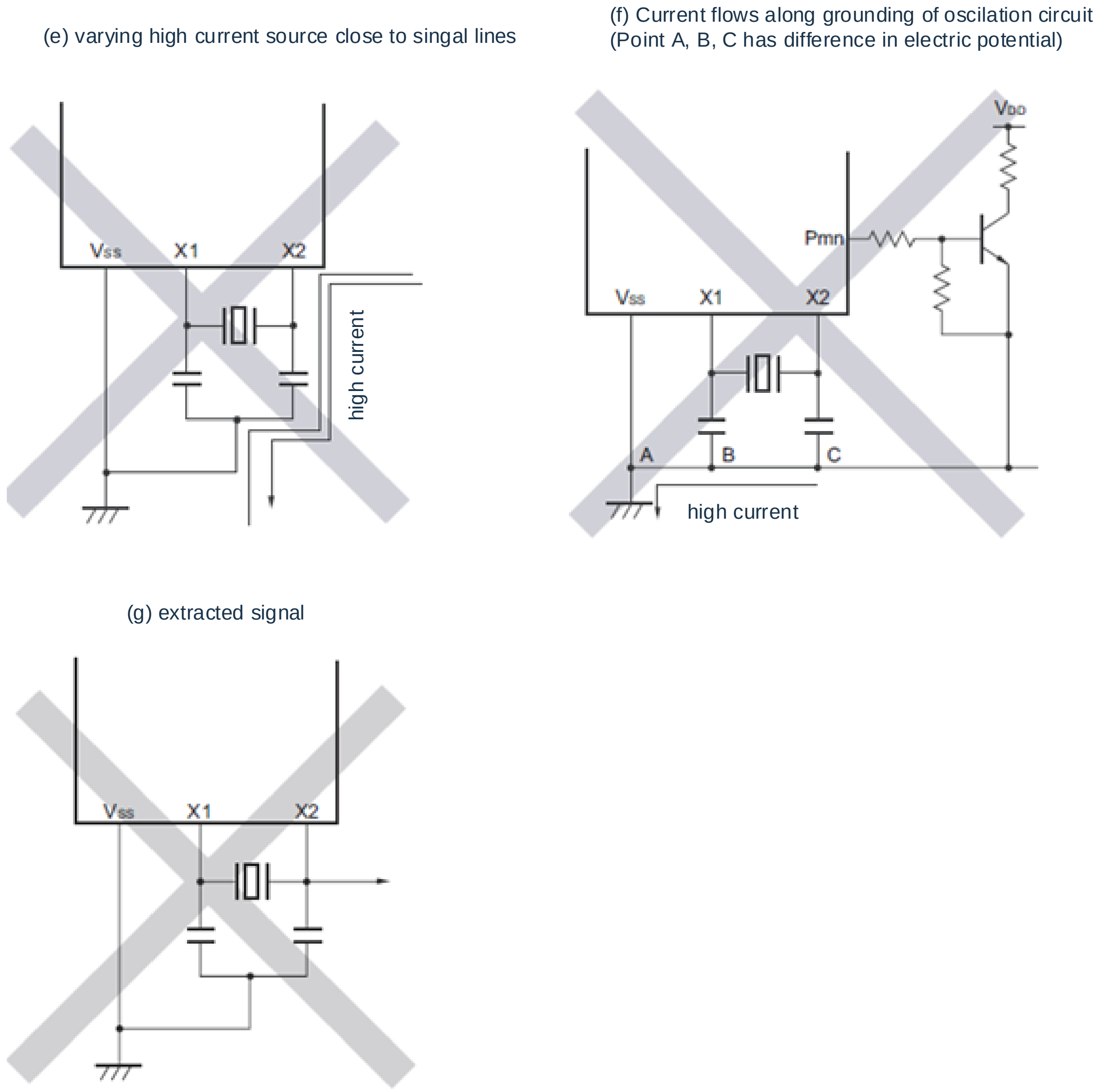
(c) Cross-wiring of signal lines for X1 and X2 the wiring

(d) The X1 and X2 are powered or mapped under

Note: In a multi-layered or dual-faceplate, you cannot configure power or map shapes below the X1 pin, X2 pin, and resonator cabling areas (dashed lines in the figure). The wiring may not produce a capacitive component and affect the oscillation characteristics.

Note: Using the subsystem clock, replace X1 and X2 with XT1 and XT2 respectively.

Figure 4-17　　Examples of incorrect resonator connections (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: When X2 and XT1 are in parallel, the crosstalk noise of X2 will be superimposed to XT1 and cause misoperation.

Remark: Using the subsystem clock, replace X1 and X2 with XT1 and XT2 respectively.

4.4.3 High-speed internal oscillator

The BAT32G157 has a built-in high speed internal oscillator. Frequency can be selected from 64MHz, 48MHz, 32MHz, 24MHz, 16MHz, 12MHz, 8MHz, 6MHz, 4MHz, 3MHz, 2MHz, 1MHz and via option bytes (000C2H). The oscillation can be controlled by the bit0 (HIOSTOP) of the clock running state control register (CSC).

After the reset is removed, the high speed internal oscillator automatically starts to oscillate.

4.4.4 Low-speed internal oscillator

The BAT32G157 has a built-in low-speed internal oscillator.

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

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

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

4.4.5 PLL

BAT32G157 is equipped with two PLL loops, one for system clock and the other for USB.

PLL can be used to multiply the master system clock. The PLL control register PLLCR/UPLLCR bit7 (PLLSRSEL/UPLLSRSEL) selects whether the PLL reference clock is an internal high-speed oscillator or an X1 oscillation clock. The bit 0 (PLLON/UPLLON) of PLLCR/UPLLCR controls whether PLL works.

When the bit0 (CKSELR) of the master system clock controller MCKC is "1", select the PLL clock as the master system clock. At this time, the bit4 (MCM0) of the CKC register must be set.

4.5 Operation of clock generation circuit

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

○ Main system clock f_{MAIN}

- High speed system clock f_{MX}

  X1 clock f_X

  External main system clock f_{EX}

- High speed internal oscillator clock f_{IH}
- PLL clock f_{PLL}

○ Sub-system clock f_{SUB}

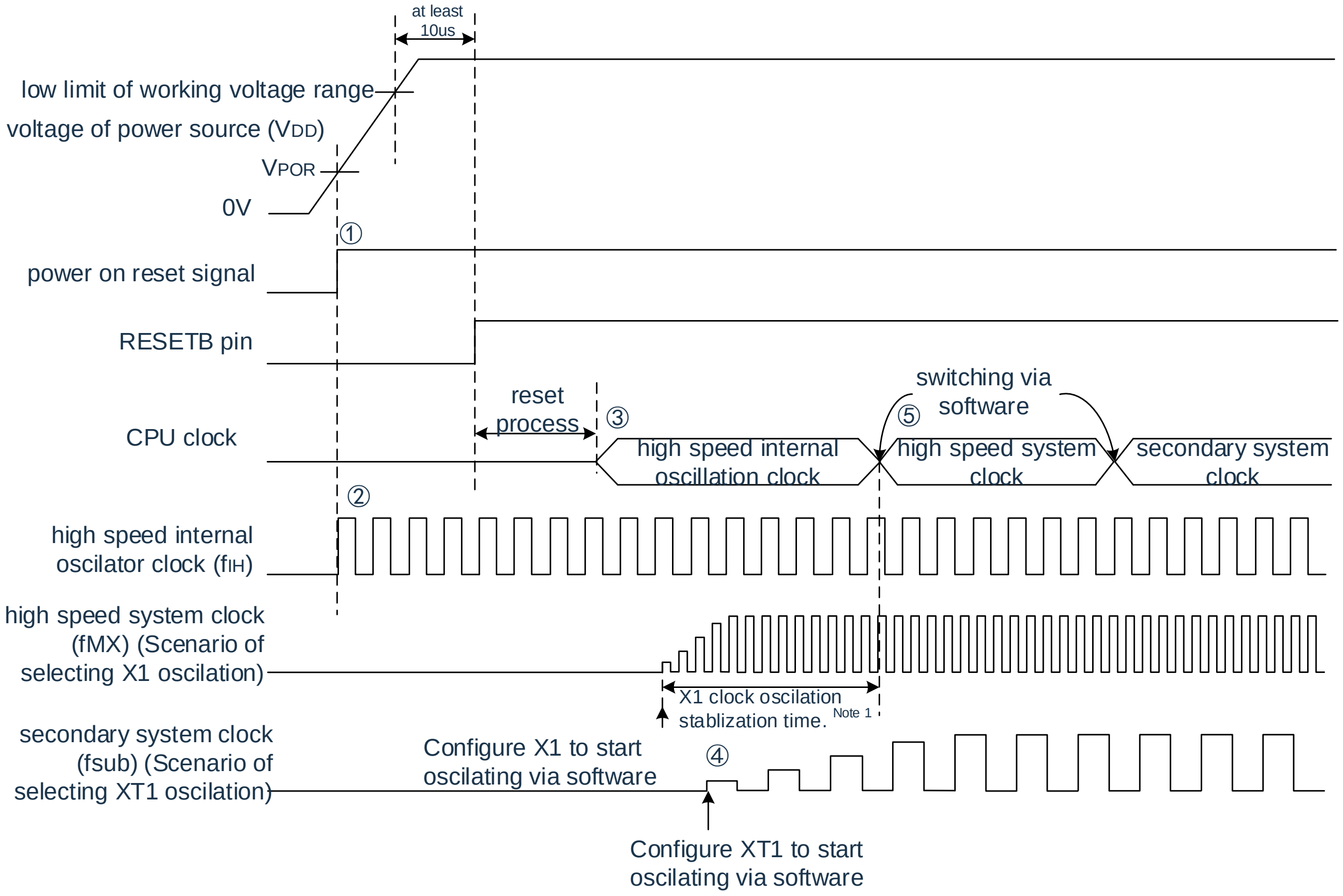
- XT1 clock f_{XT}
- External sub-system clockf_{EXS}

○ Low-speed internal oscillator clock f_{IL}

○ CPU/peripheral hardware clock f_{CLK}

After the BAT32G157 is released from reset, the CPU starts operation through the output of the high-speed internal oscillator.The operation of the clock generating circuit when the power is turned on is as shown in Figure.

Figure 4-18 Operation of clock generation circuit when power is turned on



① An internal reset signal is generated by the power-on reset (POR) circuit after the power is turned on. However, before reaching the operation voltage range shown by the AC characteristic of the data manual, the reset state is maintained by a voltage detection circuit or an external reset (the above figure is an example when an external reset is used).

② If that reset is release, the high speed internal oscillator will automatically start to oscillate.

③ After the reset is removed, a voltage stable waiting and reset process are performed, and then the CPU starts running with a high speed internal oscillator clock.

④ You must set the X1 or XT1 clock start oscillation by software (refer to "4.6.2 Example of X1 oscillation circuit setup" and "4.6.3 Example of X1 oscillation circuit setup").

⑤ If you want to switch the CPU clock to X1 clock or XT1 clock or PLL clock, you must wait for the clock oscillation to stabilize and then set the switch by software.(refer to"4.6.2 Example of X1 oscillation circuit setup" and "4.6.3 Example of X1 oscillation circuit setup").

NOTE:1. When the reset is removed, the X1 clock's oscillation stability time must be confirmed through OSTC of the oscillation stability time counter.

Note: If you use an external clock input by the EXCLK pin, you do not need an oscillatory steady-state wait time.

4.6 Clock control

4.6.1 Example of high speed internal oscillator set-up

The CPU/peripheral hardware clock (f_{CLK}) must run at high internal oscillator clock. High speed internal oscillator frequencies can be selected from 64MHz, 48MHz, 32MHz, 24MHz, 16MHz, 12MHz, 8MHz, 6MHz, 4MHz, 3MHz, 2MHz and 1MHz by FRQSEL0~FRQSEL4 bits of option bytes (000C2H). In addition, the frequency can be changed by a frequency selection register (HOCODIV) of a high speed internal oscillator.

[Settings for option bytes]

Address: 000C2H

Options bytes (000C2H)	7	6	5	4	3	2	1	0
	1	1	1	FRQSEL4 0/1	FRQSEL3 0/1	FRQSEL2 0/1	FRQSEL1 0/1	FRQSEL0 0/1

FRQSEL4	FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0	Frequency of high speed internal oscillator	
					fHOCO	fIH
1	1	0	0	0	64MHz	64MHz
1	0	0	0	0	48MHz	48MHz
0	1	0	0	0	32MHz	32MHz
0	0	0	0	0	24MHz	24MHz
0	1	0	0	1	32MHz	16MHz
0	0	0	0	1	24MHz	12MHz
0	1	0	1	0	32MHz	8MHz
0	0	0	1	0	24MHz	6MHz
0	1	0	1	1	32MHz	4MHz
0	0	0	1	1	24MHz	3MHz
0	1	1	0	0	32MHz	2MHz
0	1	1	0	1	32MHz	1MHz
Others					Disable setting.	

[Setting of the frequency selection register of the high-speed internal oscillator (HOCODIV)]

Address: 0x40021C20

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

HOCODIV2	HOCODIV1	HOCODIV0	Selection of Clock Frequency of High Speed Internal Oscillator			
			FRQSEL4=0		FRQSEL4=1	
			FRQSEL3=0	FRQSEL3=1	FRQSEL3=0	FRQSEL3=1
0	0	0	f_{IH}=24MHz f_{HOCO}=24MHz	f_{IH}=32MHz f_{HOCO}=32MHz	f_{IH}=48MHz f_{HOCO}=48MHz	f_{IH}=64MHz f_{HOCO}=64MHz
0	0	1	f_{IH}=12MHz f_{HOCO}=24MHz	f_{IH}=16MHz f_{HOCO}=32MHz	f_{IH}=24MHz f_{HOCO}=48MHz	f_{IH}=32MHz f_{HOCO}=64MHz
0	1	0	f_{IH}=6MHz f_{HOCO}=24MHz	f_{IH}=8MHz f_{HOCO}=32MHz	f_{IH}=12MHz f_{HOCO}=48MHz	f_{IH}=16MHz f_{HOCO}=64MHz
0	1	1	f_{IH}=3MHz f_{HOCO}=24MHz	f_{IH}=4MHz f_{HOCO}=32MHz	f_{IH}=6MHz f_{HOCO}=48MHz	f_{IH}=8MHz f_{HOCO}=64MHz
1	0	0	Disable setting.	f_{IH}=2MHz f_{HOCO}=32MHz	f_{IH}=3MHz f_{HOCO}=48MHz	f_{IH}=4MHz f_{HOCO}=64MHz
1	0	1	Disable setting.	f_{IH}=1MHz f_{HOCO}=32MHz	Disable setting.	f_{IH}=2MHz f_{HOCO}=64MHz
Others			Disable setting.			

4.6.2 Example of X1 oscillation circuit setup

The CPU/peripheral hardware clock (f_{CLK}) must run at high internal oscillator clock. Thereafter, if the X1 oscillating clock is changed, setting of the oscillating circuit and controlling the oscillation start are performed by the OSTS, the clock operation mode control register (CMC) and the clock operation state control register (CSC), and the oscillation stabilization is waited by the state register (OSTC) of the oscillation stabilization time counter. The X1 oscillation clock is set to f_{CLK} through the system clock control register (CKC) after waiting for oscillation stabilization.

[Register Settings] Registers must be set in the order (1) to (5).

① The OSCSEL bit of the CMC register is "1", and the AMPH bit is "1" when f_X is 10 MHz.

	7	6	5	4	3	2	1	0
CMC	EXCLK0	OSCSEL1	EXCLKS0	OSCILS0	0	AMPHS10	AMPHS00	AMPH0/1

② The oscillation stability time of the X1 oscillation circuit when the deep sleep mode is released is selected by the OSTS register.

Example) To wait at least 102 μs through a 10 MHz resonator, you must set
the following values.

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

③ The MSTOP bit of the CSC register is cleared "0" so that the X1 oscillation circuit starts oscillation.

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

④ The OSTC register is used to wait for the oscillation stabilization of the X1 oscillation circuit.

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

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

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

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

4.6.3 Example of X1 oscillation circuit setup

After the reset is released, the CPU/peripheral hardware clock (f_{CLK}) must run with the high-speed internal oscillator clock. After that, if it is changed to XT1 oscillation clock, the mode control register (OSMC), clock operation mode control register (CMC) and clock operation status control register (CSC) are provided through the subsystem clock to set the oscillation circuit and control the oscillation start, and the XT1 oscillation clock is set to f_{CLK} through the system clock control register (CKC).

Register Settings must be set in the order (1) to (5).

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

	7	6	5	4	3	2	1	0
OSMC	RTCLPC0/1	0	0	WUTMCK00	0	0	0	0

② The OSCSELS bit of the CMC register is "1" so that the XT1 oscillation circuit operates.

	7	6	5	4	3	2	1	0
CMC	EXCLK0	OSCSEL0	EXCLKS0	OSCILS1	0	AMPHS10/1	AMPHS00/1	AMPH0

AMPHS0 and AMPHS1 bits: An oscillation mode of the XT1 oscillation circuit is set.

③ The XTSTOP bit of the CSC register is cleared "0" so that the XT1 oscillation circuit starts to oscillate.

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

④ It is necessary to wait for that oscillation steady time required by the secondary system clock through software, timer function, etc.

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

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

4.6.4 Example of PLL Circuit Setup

After the reset is released, the CPU/peripheral hardware clock (f_{CLK}) must run with the high-speed internal oscillator clock. After that, if you want to shift to PLL clock, you can set PLL control register (PLLCR/UPLLCR) and main system clock control register (MCKC) to control PLL circuit.

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

① Set the PLLSRSEL/UPLLSRSEL bits to select the multiplier X1 oscillator clock or the high-speed internal oscillator clock.

Set the PLLD1/PLLD0/PLLM bits to determine the frequency division and multiplier times.

	7	6	5	4	3	2	1	0
PLLCR	PLLSRSEL0/1	0	0	0	PLLD10/1	PLLD10/1	PLLM0/1	PLLON0

	7	6	5	4	3	2	1	0
UPLLCR	UPLLSRSEL0/1	0	0	0	UPLLD10/1	UPLLD10/1	UPLLM0/1	UPLLON0

Note: 1. X1 oscillator clock or high speed internal oscillator clock is used as the input clock of PLL, the frequency range is: 4Mhz~8Mhz.

2. The dividing multiplier of PLL can be set freely according to the need, and the dividing multiplier of UPLL needs to be set according to the input clock frequency, so that the output clock is 48MHz.

② Set the RDIV1/RDIV0 bits of the MCKC register to select the system clock frequency division.

	7	6	5	4	3	2	1	0
MCKC	0	0	0	0	0	RDIV10/1	RDIV00/1	CKSELR0

③ Wait at least 1 µ s and then set PLLON/UPLLON to start PLL operation.

	7	6	5	4	3	2	1	0
PLLCR	PLLSRSEL0/1	0	0	0	PLLD10/1	PLLD10/1	PLLM0/1	PLLON1

	7	6	5	4	3	2	1	0
UPLLCR	UPLLSRSEL0/1	0	0	0	UPLLD10/1	UPLLD10/1	UPLLM0/1	UPLLON1

④ It must be set by software to wait at least 40µs (the required oscillation stabilization time of the PLL clock).

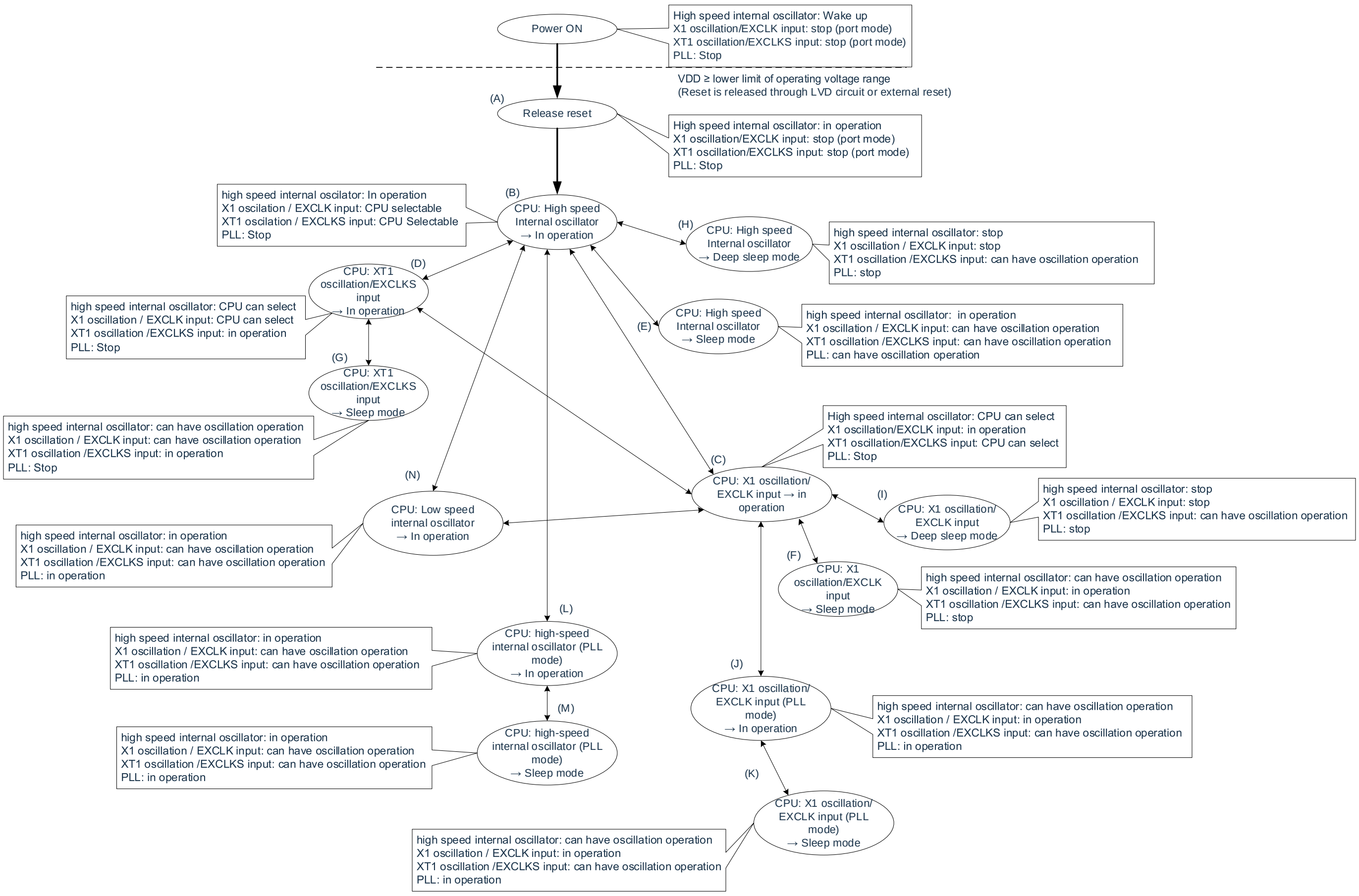
⑤ Set the CKSELR bit of the MCKC register to select the PLL clock as the system clock.

	7	6	5	4	3	2	1	0
MCKC	0	0	0	0	0	RDIV10/1	RDIV00/1	CKSELR1

4.6.5 State transition diagram of CPU clock

The CPU clock state transition diagram of this product is as shown in Figure.

Figure 4-19 State transfer diagram of CPU clock


Power ON
High speed internal oscillator: Wake up
X1 oscillation/EXCLK input: stop (port mode)
XT1 oscillation/EXCLKS input: stop (port mode)
PLL: Stop
VDD ≥ lower limit of operating voltage range
(Reset is released through LVD circuit or external reset)
(A)
Release reset
High speed internal oscillator: in operation
X1 oscillation/EXCLK input: stop (port mode)
XT1 oscillation/EXCLKS input: stop (port mode)
PLL: Stop
high speed internal oscilator: In operation
X1 oscilation / EXCLK input: CPU selectable
XT1 oscilation / EXCLKS input: CPU Selectable
PLL: Stop
(B)
CPU: High speed Internal oscillator → In operation
(H)
CPU: High speed Internal oscillator → Deep sleep mode
high speed internal oscillator: stop
X1 oscillation / EXCLK input: stop
XT1 oscillation /EXCLKS input: can have oscillation operation
PLL: stop
(D)
CPU: XT1 oscillation/EXCLKS input → In operation
high speed internal oscillator: CPU can select
X1 oscillation / EXCLK input: CPU can select
XT1 oscillation /EXCLKS input: in operation
PLL: Stop
(E)
CPU: High speed Internal oscillator → Sleep mode
high speed internal oscillator: in operation
X1 oscillation / EXCLK input: can have oscillation operation
XT1 oscillation /EXCLKS input: can have oscillation operation
PLL: can have oscillation operation
(G)
CPU: XT1 oscillation/EXCLKS input → Sleep mode
high speed internal oscillator: can have oscillation operation
X1 oscillation / EXCLK input: can have oscillation operation
XT1 oscillation /EXCLKS input: in operation
PLL: Stop
High speed internal oscillator: CPU can select
X1 oscillation/EXCLK input: in operation
XT1 oscillation/EXCLKS input: CPU can select
PLL: Stop
(C)
CPU: X1 oscillation/ EXCLK input → in operation
(I)
CPU: X1 oscillation/ EXCLK input → Deep sleep mode
high speed internal oscillator: stop
X1 oscillation / EXCLK input: stop
XT1 oscillation /EXCLKS input: can have oscillation operation
PLL: stop
(N)
CPU: Low speed internal oscillator → In operation
high speed internal oscillator: in operation
X1 oscillation / EXCLK input: can have oscillation operation
XT1 oscillation /EXCLKS input: can have oscillation operation
PLL: in operation
(F)
CPU: X1 oscillation/EXCLK input → Sleep mode
high speed internal oscillator: can have oscillation operation
X1 oscillation / EXCLK input: in operation
XT1 oscillation /EXCLKS input: can have oscillation operation
PLL: stop
(L)
CPU: high-speed internal oscillator (PLL mode) → In operation
high speed internal oscillator: in operation
X1 oscillation / EXCLK input: can have oscillation operation
XT1 oscillation /EXCLKS input: can have oscillation operation
PLL: in operation
(J)
CPU: X1 oscillation/ EXCLK input (PLL mode) → In operation
high speed internal oscillator: can have oscillation operation
X1 oscillation / EXCLK input: in operation
XT1 oscillation /EXCLKS input: can have oscillation operation
PLL: in operation
(M)
CPU: high-speed internal oscillator (PLL mode) → Sleep mode
high speed internal oscillator: in operation
X1 oscillation / EXCLK input: can have oscillation operation
XT1 oscillation /EXCLKS input: can have oscillation operation
PLL: in operation
(K)
CPU: X1 oscillation/ EXCLK input (PLL mode) → Sleep mode
high speed internal oscillator: can have oscillation operation
X1 oscillation / EXCLK input: in operation
XT1 oscillation /EXCLKS input: can have oscillation operation
PLL: in operation

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

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

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

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

(2). The CPU is transferred to the high speed internal oscillator clock operation (B).

Object state transition: (C) ➞ (B), (D) ➞ (B), (N) ➞ (B)

(Order in which SFR registers are set) →

Settings flag for SFR CPU clock after transfer	CSC register	Wait for stable oscillation accuracy	CKC register	
	HIOSTOP		MCM0	CSS
High speed internal oscillator clock	0	1μs	0	0

Not required in high-speed internal oscillator clock operation.

Object state transition: (L)➞(B)

(Order in which SFR registers are set) →

Settings flag for SFR CPU clock after transfer	MCKC register	Clock switching waiting	PLLCR register
	CKSELR		PLLON
High speed internal oscillator clock	0	Confirm CKSTR=0 of MCKC register	0

(3). The CPU transfers to a high-speed internal oscillator multiplied PLL action clock (L).

Object state transition: (B)➞(L)

(Order in which SFR registers are set) →

Settings flag for SFR CPU clock after transfer	PLLCR register				Waiting time	PLLCR register
	PLLSRSEL	PLLD1	PLLD0	PLLM		PLLON
High-speed internal oscillator multiplied PLL clock	0	0/1	0/1	0/1	1μs	1

→

PLL oscillation stabilization time	MCKC register	Clock switch confirmation
	CKSELR	
40μs	1	Confirm MCKC register CKSTR=1

Remark: 1.×: Ignore

2. (A) to (N) of Table 4-3 corresponds to (A) to (N) of Figure 4-23.

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

(4). The CPU shifts to a high-speed system clock multiplied PLL action clock (J).

Object state transition: (C)→(J)

(Order in which SFR registers are set) →

Settings flag for SFR / CPU clock after transfer	PLLCR register				Waiting time	PLLCR register
	PLLSRSEL	PLLD1	PLLD0	PLLM		PLLON
high-speed system clock multiplied PLL clock	1	0/1	0/1	0/1	1μs	1

→

PLL oscillation stabilization time	MCKC register	Clock switch confirmation
	CKSELR	
40μs	1	Confirm MCKC register CKSTR=1

(5). The CPU shifts to the high-speed system clock (C).

Object state transition: (B)→(C), (D)→(C), (N)→(C)

(Order in which SFR registers are set) →

Settings flag for SFR / CPU clock after transfer	CMC register[Note1]			OSTS register	CSC register	OSTC register	CKC register	
	EXCLK	OSCSEL	AMPH		MSTOP		MCM0	CSS
X1 clock: 1MHz≤f_X≤10MHz	0	1	0	Note 2	0	Need to confirm	1	0
X1 clock: 10MHz＜f_X≤20MHz	0	1	1	Note 2	0	Need to confirm	1	0
External Main Clock	1	1	×	Note 2	0	No need to confirm	1	0

Note: 1. After the reset is released, the clock operation mode control register (CMC) can only be written 1 time by 8-bit memory operation instructions.

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

- The oscillation stabilization time of the OSTC ≤ the oscillation stabilization time set by the OSTS register

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

Object state transition: (J) →(C)

(Order in which SFR registers are set) →

Settings flag for SFR / CPU clock after transfer	MCKC register	Clock switching wait	PLLCR register
	CKSELR		PLLON
X1 clock or external clock	0	Confirm MCKC register CKSTR=0	0

Remark: 1.×: Ignore

2. (A) to (N) of Table 4-3 corresponds to (A) to (N) of Figure 4-23.

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

(6). The CPU shifts to the subsystem clock (D).

Object state transition: (B)→(D), (C)→(D)

(Order in which SFR registers are set) →

Settings flag for SFR CPU clock after transfer	CMC register[Note]				CSC register	Oscillation stable waiting	CKC register
	EXCLKS	OSCSELS	AMPHS1	AMPHS0	XTSTOP		CSS
XT1 clock	0	1	0/1	0/1	0	Yes	1
External sub-clock	1	1	×	×	0	Yes	1

Object state transition: (N)->(D)

(Order in which SFR registers are set) →

Settings flag for SFR CPU clock after transfer	CMC register[Note]				CSC register	Oscillation stable waiting	SUBCKSEL register
	EXCLKS	OSCSELS	AMPHS1	AMPHS0	XTSTOP		SELLOSC
XT1 clock	0	1	0/1	0/1	0	Yes	0
External sub-clock	1	1	×	×	0	Yes	0

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

Remark: 1.×: Ignore

2. (A) to (N) of Table 4-3 corresponds to (A) to (N) of Figure 4-23.

Table4-3 Examples of CPU clock transfer and SFR register set-up (4/4)

(7). The CPU shifts to the low-speed internal oscillator (N).

Object state transition: (B)→(N), (C)→(N)

(Order in which SFR registers are set) →

Settings flag for SFR CPU clock after transfer	SUBCKSEL register	Oscillation stable waiting	CKC register	Clock switching wait
	SELLOSC		CSS	
Low-speed internal oscillator clock	1	Yes	1	CKC register CLS=1

(8). • The CPU is shifted to sleep mode (E) while the high-speed internal oscillator clock is running (B).

- The CPU is shifted to sleep mode (F) while running on the high-speed system clock (C).
- The CPU is shifted to sleep mode (G) while the subsystem clock is running (D).
- The CPU is shifted to sleep mode (K) while the high-speed system clock multiplied PLL action is running (J).
- The CPU is shifted to sleep mode (M) while the high-speed internal oscillator clock multiplied PLL action is running (L).

State Transition	Setting content
(B)→(E) (C)→(F) (D) →(G) (J)→(K) (L)→(M)	Execute WFI instructions.

(9). • The CPU is shifted to deep sleep mode (H) while the high-speed internal oscillator clock is running (B).

- The CPU is shifted to deep sleep mode (I) during high-speed system clock operation (C).

(Setting order) →

State Transition		Setting content		
(B)→(H)		Stop Peripheral functions that cannot be run in deep sleep mode.	—	The bit2 (SLEEPDEEP) of SCR register is set to 1, and the WFI instruction is executed.
(C)→(I)	X1 oscillation		Set the OSTS register.	
	External Clock		—	

Remark (A) to (N) of Table 4-3 corresponds to (A) to (N) of Figure 4-23.

4.6.6 Conditions before CPU clock transfer and post-transfer processing

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

Table4-4 Transfer of CPU clock (1/3)

CPU clock		Pre-transfer conditions	Post-transfer processing
Before transfer	After transfer		
High-speed internal oscillator clock	X1 clock	X1 oscillation is stable. • OSCSEL=1, EXCLK=0, MSTOP=0 • After oscillation stabilization time	If the oscillation of the high-speed internal oscillator is stopped (HIOSTOP=1), the operating current can be reduced.
	External main system clock	Set the external clock of the EXCLK pin input to valid. • OSCSEL=1, EXCLK=1, MSTOP=0	
	XT1 clock	X1 oscillation is stable. • OSCSELS=1, EXCLKS=0, XTSTOP=0 • After oscillation stabilization time	
	External subsystem clock	Set the external clock of the EXCLKS pin input to be valid. • OSCSELS=1, EXCLKS=1, XTSTOP=0	
	Low-speed internal oscillator clock	Allows for low-speed internal oscillator oscillation. • SELLOSC=1 • After oscillation stabilization time	
	PLL clock (multiplied high-speed internal oscillator clock)	Allows the PLL to work. • PLLON=1, CKSELR=1 • After oscillation stabilization time	The oscillation of the high-speed internal oscillator cannot be stopped.
Low-speed internal oscillator clock	High-speed internal oscillator clock	Allows high-speed internal oscillator oscillation. • HIOSTOP=0 • After oscillation stabilization time	Can stop F_{IL} oscillation (SELLOSC=0).
	External main system clock	Set the external clock of EXCLK pin input to valid. • OSCSEL=1, EXCLK=1, MSTOP=0	
	XT1 clock	Cannot be transferred.	—
	External subsystem clock	Cannot be transferred.	—
	PLL clock	Cannot be transferred.	—

Table4-4 Transfer of CPU clock (2/3)

CPU clock		Pre-transfer conditions	Post-transfer processing
Before transfer	After transfer		
X1 clock	High-speed internal oscillator clock	Allows high-speed internal oscillator oscillation. • HIOSTOP=0 • After oscillation stabilization time	Can stop the oscillation of X1 (MSTOP=1).
	External main system clock	Cannot be transferred.	-
	XT1 clock	XT1 oscillation is stable. • OSCSELS=1, EXCLKS=0, XTSTOP=0 • After oscillation stabilization time	Can stop the oscillation of X1 (MSTOP=1).
	External subsystem clock	Sets the external clock of the EXCLKS pin input to valid. • OSCSELS=1, EXCLKS=1, XTSTOP=0	Can stop the oscillation of X1 (MSTOP=1).
	Low-speed internal oscillator clock	Allows high-speed internal oscillator oscillation. • SELLOSC=1 • After oscillation stabilization time	Can stop the oscillation of X1 (MSTOP=1).
	PLL clock (multiplied X1 clock)	Allows the PLL to work. • PLLON=1, CKSELR=1 • After oscillation stabilization time	Cannot stop the oscillation of X1 (MSTOP=0).
External main system clock	High-speed internal oscillator clock	Allows high-speed internal oscillator oscillation. • HIOSTOP=0 • After oscillation stabilization time	Able to disable the input of the external main system clock (MSTOP=1).
	X1 clock	Cannot be transferred.	-
	XT1 clock	XT1 oscillation is stable. • OSCSELS=1, EXCLKS=0, XTSTOP=0 • After oscillation stabilization time	Able to disable the input of the external main system clock (MSTOP=1).
	External subsystem clock	Sets the external clock of the EXCLKS pin input to valid. • OSCSELS=1, EXCLKS=1, XTSTOP=0	Able to disable the input of the external main system clock (MSTOP=1).
	Low-speed internal oscillator clock	Allows low-speed internal oscillator oscillation. • SELLOSC=1 • After oscillation stabilization time	Able to disable the input of the external main system clock (MSTOP=1)
	PLL clock (multiplied external main system clock)	Allows the PLL to work. • PLLON=1, CKSELR=1 • After oscillation stabilization time	The input of the external main system clock cannot be set to invalid (MSTOP=0).

Table 4-4 Transfer of CPU clock (3/3)

CPU clock		Pre-transfer conditions	Post-transfer processing
Before transfer	After transfer		
XT1 Clock	High-speed internal oscillator clock	The high speed internal oscillator is oscillating and the high speed internal oscillator clock is selected as the main system clock. • HIOSTOP=0, MCS=0	Can stop the oscillation of XT1 (XTSTOP=1).
	X1 clock	The X1 oscillation is stable and the high-speed system clock is selected as the main system clock. system clock. • OSCSEL=1, EXCLK=0, MSTOP=0 • After oscillation stabilization time • MCS=1	
	External main system clock	Set the external clock of the EXCLK pin input to active and select the high-speed system clock as the main system clock. • OSCSEL=1, EXCLK=1, MSTOP=0 • MCS=1	
	External subsystem clock	Cannot be transferred.	—
	Low-speed internal oscillator clock	Cannot be transferred.	—
	PLL clock	Cannot be transferred.	—
External subsystem clock	High-speed internal oscillator clock	The high speed internal oscillator is oscillating and the high speed internal oscillator clock is selected as the main system clock. • HIOSTOP=0, MCS=0	Able to disable the input of the external subsystem clock
	X1 clock	X1 oscillation is stable and the high-speed system clock is selected as the main system clock. • OSCSEL=1, EXCLK=0, MSTOP=0 • After oscillation stabilization time • MCS=1	
	External main system clock	Set the external clock of the EXCLK pin input to active and select the high-speed system clock as the master system clock. • OSCSEL=1, EXCLK=1, MSTOP=0 • MCS=1	
	XT1 clock	Cannot be transferred.	—
	Low-speed internal oscillator clock	Cannot be transferred.	—
	PLL clock	Cannot be transferred.	—

4.6.7 Time required to switch CPU clock and main system clock

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

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

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

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

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

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

Table 4-6 Maximum number of clocks required for f_{IH}↔f_{MX}

Set value before switching		Set value after switch	
MCM0		MCM0	
		0 (f_{MAIN}=f_{IH})	1 (f_{MAIN}=f_{MX})
0 (f_{MAIN}=f_{IH})	fMX≥fIH		2 Clock
	fMX<fIH		2 f_{IH}/f_{MX} clocks
1 (f_{MAIN}=f_{MX})	fMX≥fIH	2 f_{MX}/f_{IH} clocks	
	fMX<fIH	2 clocks	

Table 4-7 Maximum number of clocks required for fMAIN↔fSUB

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

Remarks: 1.Table Table and Table The number of clocks in is the number of CPU clocks before the switch.

2. Table Table and Table The number of clocks in is the number of clocks rounded to the decimal portion.

Example of a master system clock switching from a high speed system clock to a high speed internal oscillator clock (f_{IH}=8MHz, f_{MX}=10 MHz)

2 f_{MX}/f_{IH}=2(10/8)=2.5→3 clocks

4.6.8 Conditions before clock oscillation stops

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

Table4-8 Condition and flag setting before clock oscillation stops

Clock	Condition before clock stops (invalid external clock input)	Flag setting of SFR register
high speed internal oscillator clock	MCS=1 or CLS=1 (CPU operates at a clock other than the high speed internal oscillator clock)	HIOSTOP=1
X1 clock	MCS=0 or CLS=1 (CPU runs at a clock other than the high speed system clock)	MSTOP=1
external main system clock		
XT1 clock	CLS=0 (CPU runs at a clock other than the secondary system clock)	XTSTOP=1
external sub-system clock		
PLL clock	CKSTR=0 (CPU runs at a clock other than the PLL clock)	PLLON=0
low speed internal oscillator clock	CLS=0 (CPU operates at a clock other than the low speed internal oscillator clock)	SELLOSC=0

Chapter 5 Hardware Divider

The hardware divider is dedicated hardware that supports 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 (2 complement) integer division calculation
- 32 bit signed divisor, 32 bit signed divisor
- 32-bit signed quotient and 32-bit signed remainder output
- Write division register automatic trigger division calculation
- Except 0 warning flag
- Indicates the BUSY flag in the operation
- Interrupt request with calculation end
- 4 or 8 CPU clock cycles per calculation
  - Spend 4 CPU clock cycles at double speed
  - Spend 8 CPU clock cycles in non-speed state

5.2 Description of features

When using hardware dividers, you need to set the Division Register (DIVIDEND) and then the Division Register (DIVISOR), because writing to the division register automatically trigger division calculations. You can know when the calculation is done by querying the BUSY bit of the STATUS or by using the interrupt at the end of the calculation. The results can be read out through the QUOTIENT and the remainder (REMAINDER) registers.

Note: Do not write divisor or divisor registers, nor read quotient or remainder registers during calculation, otherwise the results are unpredictable.

5.3 Registers for hardware divider

The register for the hardware divider is as follows:

Register Base Address: DIV_BASE = 4001_0000H;

Register name	Register description	R/W	Reset Value	Register address
DIVIDEND	division register	R/W	0000_000H	DIV_BASE+00H
DIVISOR	divisor register	R/W	0000_000H	DIV_BASE+04H
QUOTIENT	quotient register	R	0000_000H	DIV_BASE+08H
REMAINDER	remainder register	R	0000_000H	DIV_BASE+0CH
STATUS	state register	R	0000_000H	DIV_BASE+10H

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

5.3.1 Dividend register (DIVIDEND)

The dividend register is a register that holds the divisor and its value participates in the division operation as a 32-bit signed integer.

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

5.3.2 Divisor register (DIVISOR)

The divisor register is a register for storing divisors, whose value is a 32-bit signed integer participating in the division operation. 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 register (QUOTIENT)

The register stores the quotient of the division calculation result after the division calculation is completed, and the value is taken as 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 register (REMAINDER)

The register stores the remainder of the division result after division calculation, and the value is taken as 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 zero-division flag and the BUSY flag.

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

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

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

Chapter 6 Universal Timer Unit (Timer4/8)

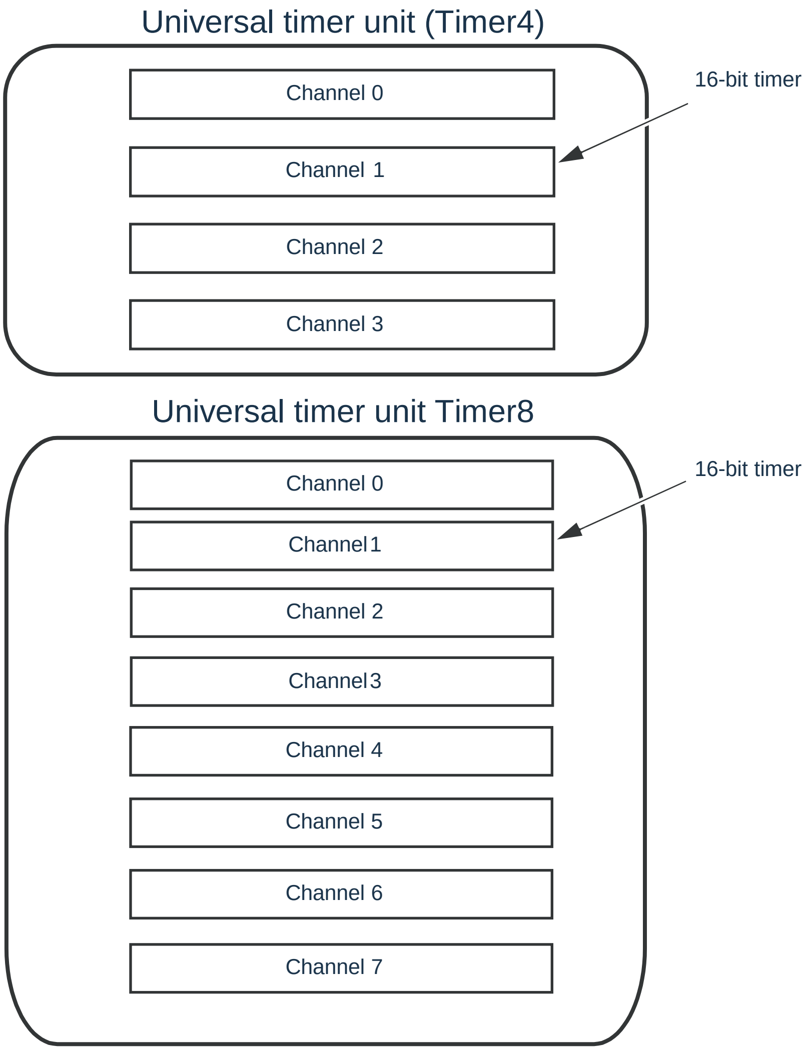
This product is equipped with two general-purpose timer units, Timer4, which contains 4 channels, and Timer8, which contains 8 channels. The number of channels of the universal timer unit varies depending on the product.

Description:

1. The symbol "m" in the following part of this chapter represents the unit number. This product is equipped with two general-purpose timers Timer4 and Timer8, so m=0,1.
2. The symbol "n" in the following of this chapter represents the channel number (in this chapter when m=0: n=0~3 / m=1: n=0~7), and the availability of timer input/output pins for each channel varies from product to product. For details, please refer to “Chapter 2 Port Function”.
3. The following contents of this chapter are limited to 64-pin products for description.

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

Each 16-bit timer is called a "channel" and can be used separately as a timer or combined with multiple channels for advanced timer functions.



For details on each feature, refer to the table below.

Independent channel operation function	Multi-channel linkage function
• Interval timer (→refer to 6.8.1) • Square wave output (→refer to 6.8.1) • External event counter (→refer to 6.8.2) • Frequency divider (→refer to 6.8.3) • Measurement of the input pulse interval (→refer to 6.8.4) • Measurement of the high and low level width of the input signal (→refer to 6.8.5) • Delay counter (→refer to 6.8.6)	• Single trigger pulse output (→refer to 6.9.1) • PWM output (→refer to 6.9.2) • Multiple PWM Output (→refer to 6.9.3)

The 16-bit timers of channel 1 and channel 3 of cell 0 can be used as two 8-bit timers (high and low). The functions of channel 1 and channel 3 can be used as 8-bit timers are:

- Interval timers (high 8-bit and low 8-bit timers)/square wave output (low 8-bit timers only)
- External event counters (low 8-bit timers only)
- Delay Counter (Low 8-bit timers only)

The LIN-bus communication can be realized through the coordination of the channel 3 of the unit 0 and the UART0 of the universal serial communication unit.

6.1 Function of universal timer unit

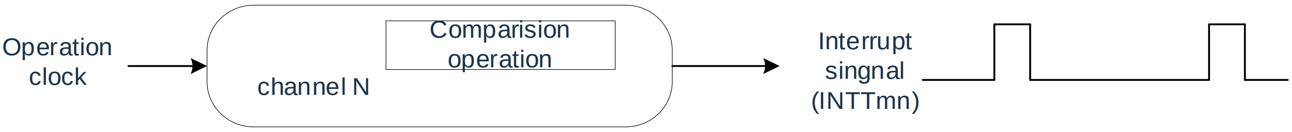
The universal timer unit has the following functions:

6.1.1 Independent channel operation

Independent channel operation function is independent of the other channel operation mode to use any channel function.

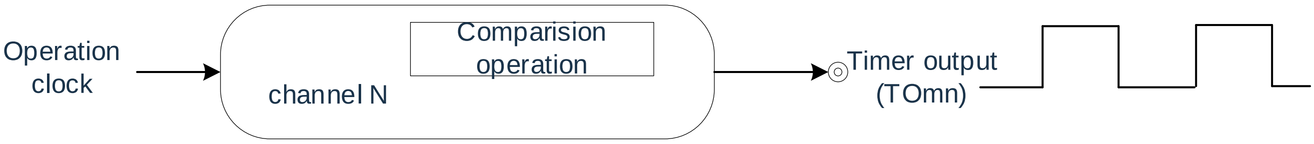
(1) Interval timer

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



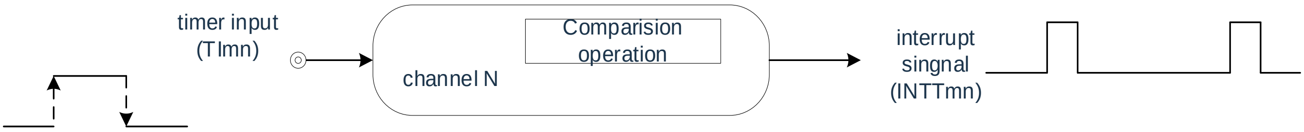
(2) Square wave output

When a INTTMmn interrupt is generated, an alternating operation is performed and a 50% duty cycle square wave is output from an output pin (TOmn) of the timer.



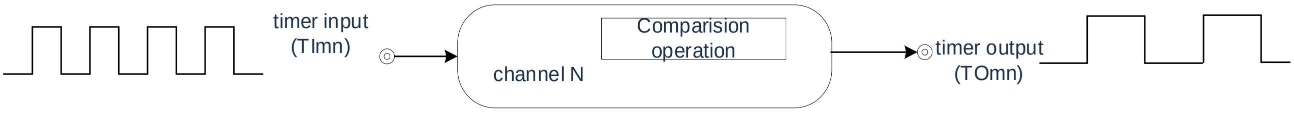
(3) External event counter

An effective edge of the input signal of the timer input pin (TImn) is counted, and if a prescribed number of times is reached, an event counter generating an interrupt can be used.



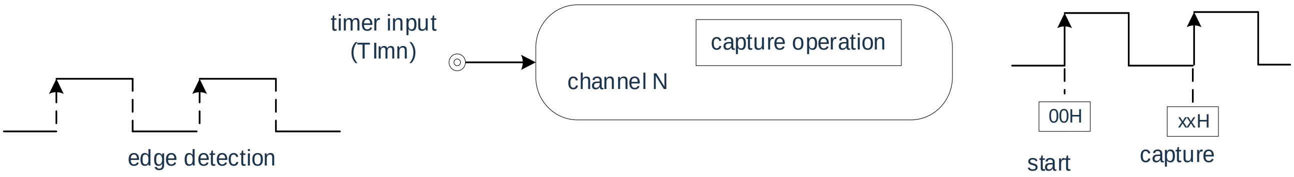
(4) Divider function (channel 0 of unit 0 only)

An input clock of a timer input pin (TI00) is frequency-divided and then output from an output pin (TO00).



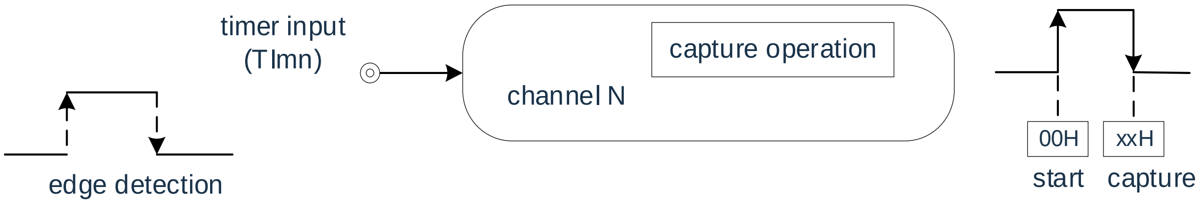
(5) Measurement of input pulse interval

An effective edge of the input pulse signal of the input pin (TImn) starts counting at a timer and captures the count value at the effective edge of the next pulse, thereby measuring the interval of the input pulse.



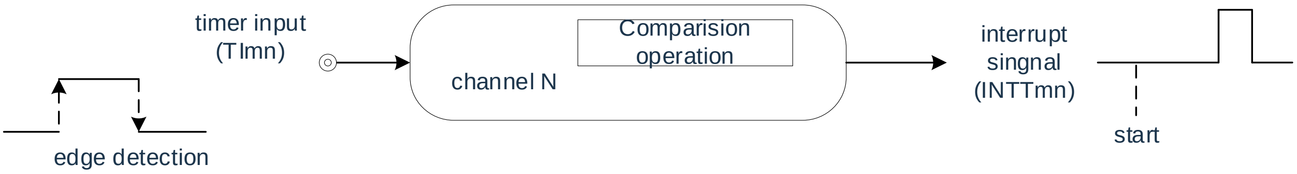
(6) Measurement of high and low level width of input signal

The input signal of the TImn is counted at one edge of the input pin at the timer and the count value is captured at the other edge, thereby measuring the high and low level width of the input signal.



(7) Delay counter

An effective edge of the input signal of the timer input pin (TImn) starts counting and an interrupt is generated after an arbitrary delay period.



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

2. Whether the timer input/output pins of each channel are different or not depends on the product. Refer to "Table 6-2, Product Has Timer Input/Output Pins" for details.

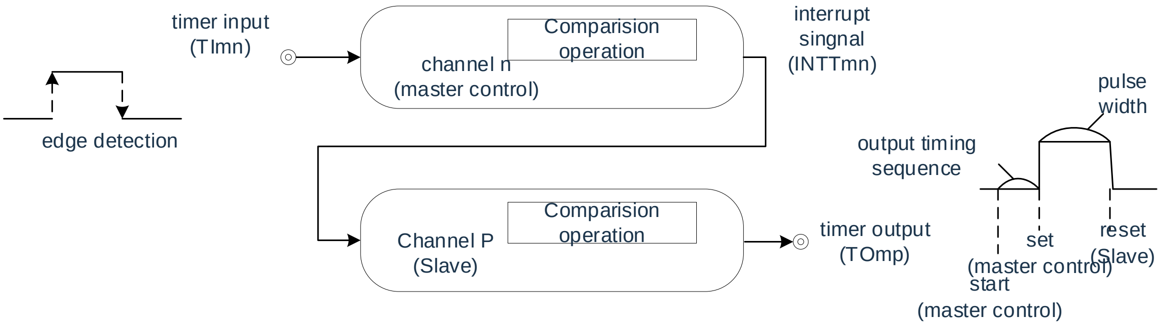
6.1.2 Multi-channel linkage operation function

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

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

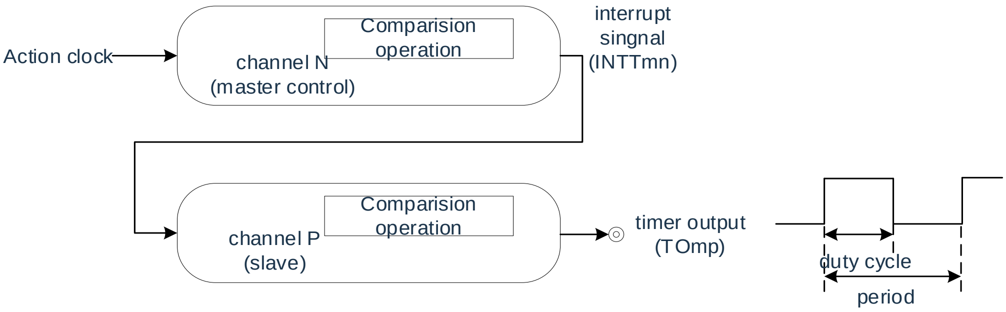
(1) Single trigger pulse output

The two channels are used in pairs, and a single trigger pulse with arbitrary output timing and pulse width is generated.



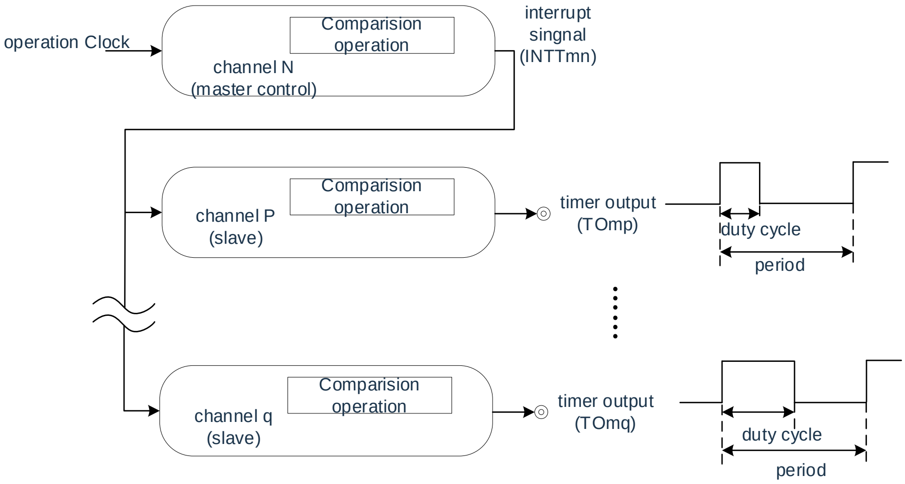
(2) PWM (Pulse Width Modulation) output

The two channels are used in pairs to generate pulses with arbitrary period and duty cycle.



(3) Multiple PWM (Pulse Width Modulation) output

The PWM signal can be generated at most 3+7 arbitrary duty cycles by extending the PWM function and using 1 master channel and multiple slave channels.



Note: For details on the Multi-Channel Coordinated Operation Functional Rules, refer to "6.4.1 Basic Rules for Multi-Channel Coordinated Operation Functions.

Note: m: Unit number (m=0,1)n: Channel number (N=0~3 when m=0, n=0~7 when m=1)
p,q: Slave channel number (when m=0: n＜p＜q≤3) (when m=1: 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 run function is the function of using the 16-bit timer channel as the 2 8-bit timer channels. Only channel 1 and channel 3 of unit 0 can be used.

Note: There are several rules when you use an 8-bit timer to run functions.

Refer to the "Basic Rules for 6.4.2 8-bit Timer Operation Functions (Channel 1 and Channel 3 only)" for details.

6.1.4 LIN-bus support (channel 3 of unit 0 only)

A universal timer unit is used to check whether a received signal in the LIN-bus communication is suitable for the LIN-bus communication table.

(1) Detection of wake-up signal

A low level width is measured by counting at the beginning of the falling edge of the input signal of the UART0 serial data input pin (RxD0) and capturing the counting at the rising edge. If that low level width is great than or equal to a fixed value, it is consider a wake-up signal.

(2) Detection of break field

After detecting the wake-up signal, the low level width is measured by counting the falling edge of the input signal of the UART0 serial data input pin (RxD0). If that low level width is great than or equal to a fixed value, it is consider as a break field.

(3) Measurement of pulse width of sync field

After detecting a break field, the low-level width and the high-level width of the input signal of the UART0 serial data input pin (RxD0) are measured. The baud rate is calculated based on the bit intervals of the sync field measured in this manner.

Note: Refer to "6.3.13 Input Switch Control Register (ISC)" and "6.8.5 Operation as Input Signal Level Width Measurement".

6.2 Structure of universal timer unit

The universal timer unit consists of the following hardware.

Table 6-1 Structure of universal timer unit

Item	Structure
Counter	timer count register mn (TCRmn).
Register	timer data register mn (TDRmn)
Timer input	TI00~TI03 Note 1, RxD0 pin (for LIN-bus)
Output of timer	TO00~TO03 Note 1, output control circuit
Control register	<Register for unit setting> · Peripheral enable register 0 (PER0) · Timer clock select register m (TPSm). · The timer channel enable status register m (TEm) · Timer channel start register m (TSm) TSm. · Timer channel stop register m (TTm) TTm. · Timer input select register 0 (TIS0) · Timer output enable register m (TOEm) · Timer output register m (TOm) · Timer output level register m (TOLm) · Timer output mode register m(TOMm)
	<Register per channel> · Timer mode register mn(TMRmn) · Timer state register mn(TSRmn) · Input switch control register (ISC) · Noise filter enable registers 1,2 (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 from product to product. For details, please refer to "Table 6-2 Timer Input/Output Pins for Each Product".

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

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

Whether the timer input/output pins of each channel of the universal timer unit are different depends on the product.

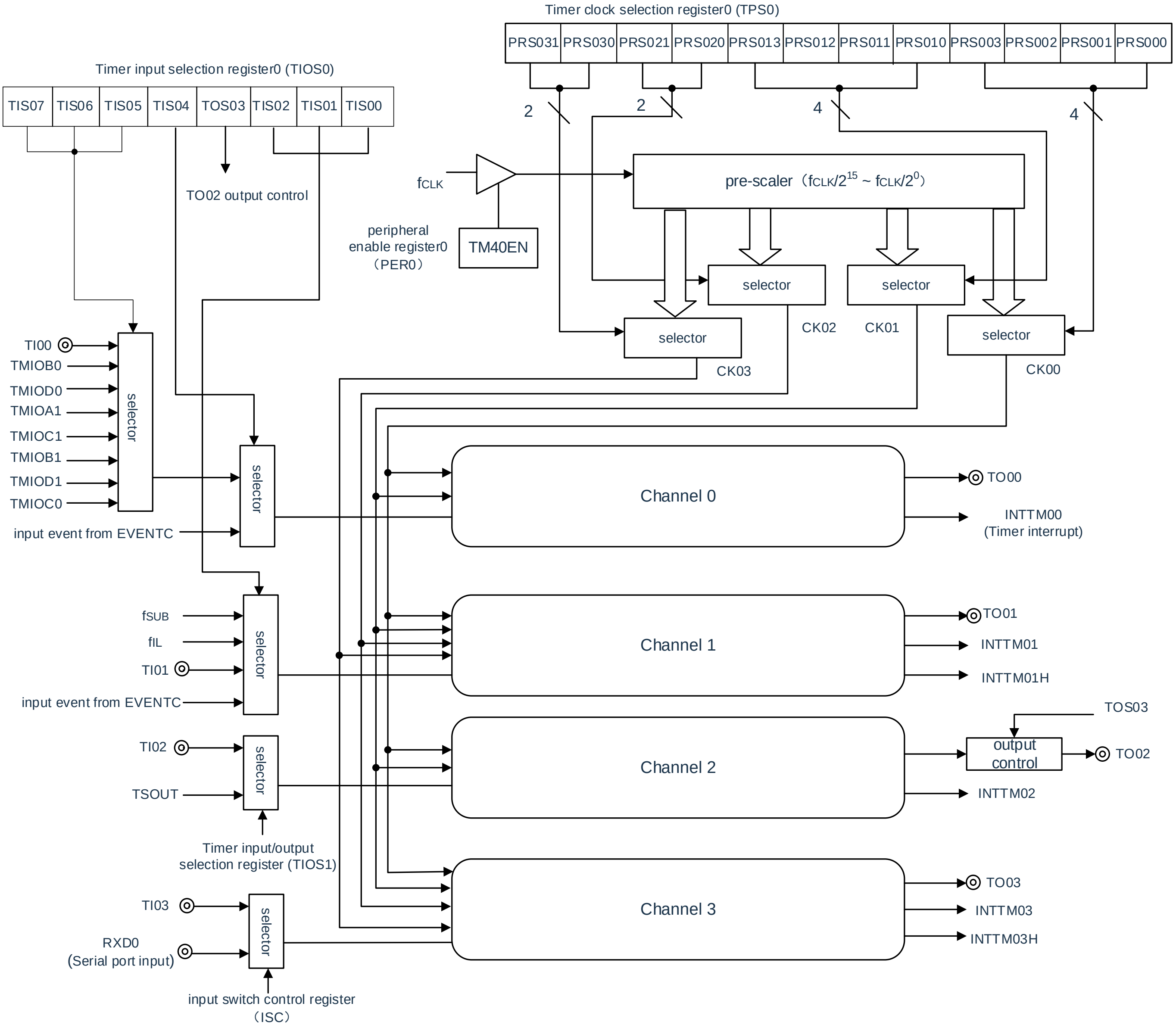
Table 6-2 Timer Input/output pins for each product

Channel of timer array unit		I/O Pins of 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
	Channel 4	TI14/TO14	TI14/TO14
	Channel 5	TI15/TO15	TI15/TO15
	Channel 6	TI16/TO16	TI16/TO16
	Channel 7	TI17/TO17	TI17/TO17

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

Block diagram of universal timer unit is shown in Figure 6-1.

Figure6-1 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 Universal timer unit register list

Unit 0 (Timer4) register base address: 0x40041C00

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

Unit 1 (Timer8) register base address: 0x40042000

Offset address	Register name	R/W	Bit width	Reset value
0x180	TCR10	R	16	FFFFH
0x182	TCR11	R	16	FFFFH
0x184	TCR12	R	16	FFFFH
0x186	TCR13	R	16	FFFFH
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	R/W	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 count clock. Count is increased or decreased in synchronization with the rising edge of the count clock.

The operation mode is selected by the MDmn3~MDmn0 bit of the timer mode register mn (TMRmn) to switch the increment and decrement count (refer to 6.3.3 timer mode register mn (TMRmn)).

Figure6-2 Table of timer count register mn (TCRmn)

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

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

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

The count value becomes "FFFFH" in the following cases.

- When a reset signal is generated
- When clearing the TM4EN/TM8EN bit of the Peripheral Enable Register 0 (PER0)
- The count of the slave channels in the PWM output mode ends
- The count of the dependent channels ends in a delayed count mode
- The count of the main/slave channels in the single-trigger pulse output mode ends
- Count end of slave channels in multiple PWM output mode

The count value becomes "0000H" in the following cases.

- When you enter the start trigger in capture mode
- At the end of the capture in capture mode

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

As shown below, the read value of the TCRmn register varies depending on the mode and state of operation.

Table 6-3 Read value of timer count register mn (TCRmn) in each running mode

Operation mode	Count Method	Read value [Note] for timer counter register mn (TCRmn)			
		The value when the Run Mode is changed after the reset is removed	Count Paused (TTmn=1)	Count Paused (TTmn=1) Changes the value when the run mode is changed	The value after a single count when waiting for the start of the trigger
interval timer mode	decremental count	FFFFFH	Value at Stop	indefinite value	—
capture mode	incremental count	0000H	Value at Stop	indefinite value	—
Event Counter Mode	decremental count	FFFFFH	Value at Stop	indefinite value	—
single count mode	decremental count	FFFFFH	Value at Stop	indefinite value	FFFFFH
Capture & Single Count Mode	incremental count	0000H	Value at Stop	indefinite value	Snap value for TDRmn register +1

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

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

6.2.3 Timer data register mn (TDRmn)

This is a 16-bit register that can be used for switching between capture and comparison functions. The operation mode is selected by the MDmn3~MDmn0 bit of the timer mode register mn(TMRmn) to switch the capture function and comparison function.

The TDRmn register can be rewritten at any time.

This register can be read and written in units of 16 bits.

Timer4 is in 8-bit timer mode (SPLIT bits of timer mode registers m1, m3 (TMRm1, TMRm3), TDRm1 registers and TDRm3 registers can be read and written in units of 8 bits, where TDRm1H and TDRm3H are used as high 8 bits and TDRm1L and TDRm3L are used as low 8 bits.

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

Figure 6-3 Table of timer data register mn(TDRmn) (n=0,2,4,5,6,7)

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

Figure 6-4 Table of timer data register mn(TDRmn) (n=1, 3)

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

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

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

The count is decremented from the set value of the TDRmn register, and an interrupt signal (INTTMmn) is generated when the count value becomes '0000H'. Holds the value of the TDRmn register until it is overwritten.

Notice: Even if you enter a capture trigger, the TDRmn register set to the comparison function does not capture the run.

(ii) the use of the timer data register mn(TDRmn) as a capture register

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

You can select the valid edge of the TImn pin as the capture trigger. A selection of capture triggers is set by a timer mode register mn (TMRmn).

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

6.3 Registers for controlling universal timer unit

The registers that control the universal timer units are as follows:

- Peripheral enable register 0 (PER0).
- Timer clock select register m (TPSm)
- timer mode register mn (TMRmn)
- timer status register mn (TSRmn)
- Timer channel enable state register m (TEm).
- Timer channel start register m (TSm)
- Timer channel stop register m (TTm).
- Timer Input-output select 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 assigned registers and bits differ depending on the product. You must set an initial value for unassigned bits.

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

6.3.1 Peripheral enable register 0 (PER0)

The PER0 register is a register that sets a clock that is allowed or prohibited to supply to each peripheral hardware. Reduce power consumption and noise by stopping clock supply to unused hardware.

You must set bit0 (TM4EN) to "1" when you want to use universal timer unit 0. You must set bit1 (TM8EN) to "1" when you want to use universal timer unit 1. The PER0 register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of the PER0 register changes to "00H".

Figure 6-5 Table for peripheral enable register 0 (PER0)

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

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

TM4EN	Control of an input clock of a universal timer unit 0
0	Stop provide an input clock. · Cannot write the SFR used by the universal timer unit 0. · The universal timer unit 0 is in a reset state.
1	Provides an input clock. · The SFR used by the universal timer unit 0 can be read and written

TM8EN	Control of an input clock of a universal timer unit 1
0	Stop provide an input clock. · Cannot write the SFR used by the universal timer unit 1. · The universal timer unit 1 is in a reset state.
1	Provides an input clock. · The SFR used by the universal timer unit 1 can be read and written

Note 1. To set a universal timer unit, you must first set the following register in the state with the TM4EN/TM8EN bit "1". When the TM4EN/TM8EN bit is "0", the control register value of the timer array unit is the initial value, neglecting the write operation (except for timer input-output selection register 0 (TIOS0), input switching control register (ISC), noise filter permit register 1 (NFEN1), port mode control register PMCx, port mode register PMx and port register Px).

· Timer status register mn(TSRmn)

· Timer channel enable status register m (TEm)

· Timer channel start register m (TSm).

· Timer channel stop register m (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 select register m (TPSm)

The TPSm register is a 16-bit register that selects two or four common runtime clocks (CKm0, CKm1, CKm2, CKm3). CKm0 is selected by bit3~0 of the TPSm register and CKm1 is selected by bit7~4 of the TPSm register. In addition, only channel 1 and channel 3 can select CKm2 and CKm3, select CKm2 by bit9~8 of TPSm register, and select CKm3 by bit13 and bit12 of TPSm register.

The TPSm register in the timer run can only be overridden in the following cases.

The case of PRSm00~PRSm03 bit can be rewritten (N=0~3 when m=0, n=0~7 when m=1):

Select CKm0 as the channel for the runtime clock (CKSmn1, CKSmn0=0, 0) all in the stopped state (TEmn=0). The case of PRSm10~PRSm13 bit can be rewritten (N=0~3 when m=0, n=0~7 when m=1):

Select CKm2 as the channel for the runtime clock (CKSmn1, CKSmn0=0, 1) all in the stopped state (TEmn=0). Can override PRSm20 and PRSm21 bits (n=1, 3):

Select CKm1 as the channel for the runtime clock (CKSmn1, CKSmn0=1, 0) all in the stopped state (TEmn=0). Can override PRSm30 and PRSm31 bits (n=1, 3):

Select CKm3 as the channel for the runtime clock (CKSmn1, CKSmn0=1, 1) all in the stopped state (TEmn=0).

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

Figure 6-6 Table of timer clock select register m (TPSm) (1/2)

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPSm	0	0	PRS m31	PRS m30	0	0	PRS m21	PRS m20	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00

PRS mk3	PRS mk2	PRS mk1	PRS mk0	Selection of operation clock $^{(CKmk)}$ (k=0,1) ^{Note}					
					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: In case of changing the clock selected as f_{CLK} (changing the value of the System Clock Control Register (CKC)), the universal timer unit (TTm=0,100FH) must be stopped. It is necessary to stop that universal timer unit even when selecting an effective edge of the run-time clock (f_{MCK}) or TImn pin input signal.

Notice: 1. bit15, 14,11,10 must be placed "0.

2. If you select fCLK as the runtime clock (CKmk) and set TDRnm to 0000H (n=0~3), you cannot use the universal timer unit.

Remark: 1 f_{CLK}: Clock frequency for CPU/peripheral hardware

2. The clock waveform selected by the TPSm register is high (m=1~15) with only 1 f_{CLK} period. Refer to “6.5.1 Count Clock (f_{TCLK})”.

Figure 6-7 Table of timer clock select register m (TPSm) (2/2)

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPSm	0	0	PRS m31	PRS m30	0	0	PRS m21	PRS m20	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00

PRSm21	PRSm20	Selection of operation clock (CKm2) Note					
			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

PRSm31	PRSm30	Selection of operation clock (CKm3) 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: In case of changing the clock selected as f_{CLK} (changing the value of the System Clock Control Register (CKC)), the universal timer unit (TTm=0,100FH) must be stopped. It is necessary to stop that universal timer unit even when selecting an effective edge of the run-time clock (f_{MCK}) or TImn pin input signal.

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

The interval time shown in Table 6-4 can be achieved by the interval timer function if channels 1 and 3 are used in 8-bit timer mode.

Table 6-4 The interval that the runtime clocks CKSm2 and CKSm3 can set

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: ○ Contains errors within 5%.

Note: 1.f_{CLK}: Clock frequency for CPU/peripheral hardware

2. Refer to the "6.5.1 Count Clock (f_{TCLK})" for details on the fCLK/2^r waveform selected by TPSm registers.

6.3.3 Timer mode register mn (TMRmn)

The TMRmn register is a register for setting channel n running mode, selecting f_{MCK}, counting clock, controlling/dependent,16bit/8 bit timer (only for channel 1 and channel 3), triggering and capturing, selecting the effective edge of timer input and running mode (interval, capture, event counter, single count, capture & single count).

Prevents the TMRmn register from being overwritten in the run (TEmn=1). However, bit7 and bit6 (CISmn1, CISmn0) can be rewritten in part of the functional operation (TEmn=1) (see "Independent Channel Operation Function of 6.8 Universal Timer Unit" and "Multi-channel Operation Function of 6.9 Timer Array Unit" for details).

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

Note: Bit11 of the TMRmn register varies by channel.

TMRm2, TMRm4, TMRm6	: MASTERmn bit (n=2, 4, 6)
TMRm1, TMRm3	: SPLITmn bit (n=1,3)
TMRm0, TMRm5, TMRm7	: Fixed as "0".

Figure 6-8 Table of timer mode register mn (TMRmn) (1/4)

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

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

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

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

CCSmn	Selection of channel n count clock (f_{TCLK})
0	Runtime clocks specified by CKSmn0 and CKSmn1 bits (f_{MCK})
1	Effective Edge of TImn Pin Input Signal · The case of Unit 0: Channel 0: Valid edge of input signal selected by TIS0 Channel 1: Valid Edges of the Input Signal Selected by TIS0
A count clock (f_{TCLK}) is used for the counter, the output control circuit, and the interrupt control circuit.	

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

Notice: 1. Bit 13, 5, 4 must be set to 0.

2. To change the clock selected as f_{CLK} (change the value of the System Clock Control Register (CKC), the timer array unit (TTm=0,10FFH) must be stopped even if the runtime clock (f_{MCK}) specified by the CKSmn0 bit and the CKSmn1 bit is selected or the valid edge of the TImn pin input signal is f_{TCLK}.

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

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

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

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

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

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

MASTERmn	Selection of independent channel operation/multi-channel simultaneous operation (slave or master) for channel n
0	A slave channel used as an independent channel operation function or a multi-channel linkage operation function.
1	The main control channel is used as the multi-channel linkage operation function.
Only channel 2,4,6 can be set as the master channel (MASTERmn=1). Channel 0 is fixed as '0' (because channel 0 is the highest bit channel, it is not related to this bit setting and is used as the master channel). For a channel that is used as a standalone channel operation function, the MASTERmn bit is set to "0".	

(Bit11 of TMRmn(n=1,3))

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

STSmn2	STSmn1	STSmn0	Settings for the start trigger and capture trigger of channel n
0	0	0	Only software triggers are active (no other trigger source is selected).
0	0	1	Use the valid edges entered by the TImn pin for the start trigger and capture trigger.
0	1	0	Use the two-sided edges of the TImn pin entries for the start trigger and capture trigger, respectively.
1	0	0	Use the interrupt signal of the main control channel (the case of a slave channel of the multi-channel linkage operation function).
Other than above			Disable setting.

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

Remark: m: Unit number (m=0, 1)n: Channel Number (N=0~3 when m=0, n=0~7 when m=1)

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

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

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

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

CISmn1	CISmn0	Effective Edge Selection for TImn Pins
0	0	Falling edge
0	1	Rising edge
1	0	Double edges (when measuring low level width) Start trigger: Falling Edge, Capture Trigger: Rising edge
1	1	Double edges (when measuring high level width) Start trigger: Rising edge, capture Trigger: falling edge
When STSmn2~STSmn0 bit is not '010B' and double edge is specified, CISmn1~CISmn0 set to '10B'.		

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

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

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

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

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

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

MD mn3	MD mn2	MD mn1	Settings for Channel n Running Mode	Corresponding function	Count run of TCR
0	0	0	interval timer mode	Interval Timer/Square Wave Output/ Divider Function/PWM Output (Master)	decremental count
0	1	0	capture mode	Measurement of input pulse interval	incremental count
0	1	1	Event Counter Mode	External event counters	decremental count
1	0	0	single count mode	Delay Counter/Single Trigger Pulse Output/PWM Output (Subordinates)	decremental count
1	1	0	Capture & Single Count Mode	Measurement of High and Low Level Width of Input Signal	incremental count
Other than above			Disable setting.		
The operation of the modes varies with the MDmn0 bit (see the following table).					

Run mode (settings for MDmn3~MDmn1 bits (see above table))	MD mn0	Settings to start counting and interrupts
· Interval timer mode (0, 0,0) · capture mode (0,1,0)	0	The timer interrupt does not occur at the start of the count (the output of the timer does not change).
	1	A timer interrupt is generated at the start of the count (the output of the timer also changes).
· Event counter mode (0,1,1)	0	The timer interrupt does not occur at the start of the count (the output of the timer does not change).
· Single Count Mode [Note 2] (1,0,0)	0	Invalid start trigger in count run. No interruption occurs at this time.
	1	The start of the count run valid triggers [note 3]. No interruption occurs at this time.
· Capture& Single Count Mode (1,1,0)	0	The timer interrupt does not occur at the start of the count (the output of the timer does not change). Invalid start trigger in count run. No interruption occurs at this time.

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

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

3. If a start trigger (TSmn=1) is generated in the run, the counter is initialized and counts are restarted (no interrupt requests are generated).

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

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 valid only in capture mode (MDmn3~MDmn1=010B) and capture & single count mode (MDmn3~MDmn1=110B. Refer to Table 6-5 for variations and set/clear conditions of OVF bits in each mode of operation.

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

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

Figure 6-12 Table of timer status register mn (TSRmn)

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

OVF	Counter overflow state for channel n
0	No overflow occurred.
1	Overflow occurred.
If the OVF bit is "1", clear this flag (OVF=0) when the next count does not overrun and the count value is captured.	

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

Table 6-5 OVF bit variation and set/clear conditions in each mode of operation

Timer operation mode	OVF bit	Set/Clear Criteria
· Capture Mode · Capture & Single Count Mode	Clear	No overflow at the time of capture
	Set	Overflow occurs during capture
· Interval timer mode · Event counter mode · Single count mode	Clear	— (not available)
	Set	

Note: Even if the counter overflows, the OVF bit does not change immediately, and changes occur in subsequent captures.

6.3.5 Timer channel enable status register m (TEm)

The TEm register is a register that represents the permitted or stopped state of operation of each channel timer.

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

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

A lower 8-bit of a TEm register can be read with a TEmL and through an 8-bit memory operation instruction. After the reset signal is generated, the value of the TEm register changes to "0000H".

Figure 6-13 Table of timer channel enable status register m (TEm)

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEm	0	0	0	0	TEHm3	0	TEHm1	0	0	0	0	0	TEm3	TEm2	TEm1	TEm0

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	TEm7	TEm6	TEm5	TEm4	TEm3	TEm2	TEm1	TEm0

m=1

TEHm3	A representation of the operational permit or stop state of a high 8-bit timer when channel 3 is in 8-bit timer mode
0	Idle Status
1	Run Enable Status

TEHm1	A representation of the operational permit or stop state of a high 8-bit timer when channel 1 is in 8-bit timer mode
0	Idle Status
1	Run Enable Status

TEmn	A representation of the running permit or stop state of channel n
0	Idle Status
1	Run Enable Status
When channels 1 and 3 are in 8-bit timer mode, TEm1 and TEm3 indicate the allowed or stopped states of low 8-bit timers.	

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

6.3.6 Timer channel start register m (TSm)

The TSm register initializes the timer count register mn (TCRmn) and sets the trigger register when each channel count operation starts. If each bit is "1", the counter bit of the timer channel allows state register m (TEm) to be "1. Because TSmn bits, TSHm1 bits, and TSHm3 bits are trigger bits, clear TSmn bits, TSHm1 bits, and TSHm3 bits immediately if it becomes run-enabled (TEmn, TEHm1, TEHm3=1).

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

The low 8 bits of the TSm register can be set with TSmL and through 8-bit memory. After the reset signal is generated, the value of the TSm register becomes "0000H".

Figure 6-14 Table of timer channel start register m (TSm)

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSm	0	0	0	0	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	Channel 3 is triggered when the operation of the high 8-bit timer in the 8-bit timer mode allows (start)
0	No trigger
1	Enter the TEHm3 bit to "1" and shift to the count enable state. If the count of the TCRm3 register is started in the count enable state, the interval timer mode is entered (Refer to Table 6-6 of "6.5.2 Start Timing of Counter").

TSHm1	Channel 1 is triggered when the operation of the high 8-bit timer in the 8-bit timer mode allows (start)
0	No trigger.
1	Set the TEHm1 to "1" to enter the count enable state. If the count of the TCRm1 register is started in the count enable state, the interval timer mode is entered (Refer to Table 6-6 of "6.5.2 Start Timing of Counter").

TSmn	Channel n operation enable (start) trigger
0	No trigger.
1	Set the TEmn bit to "1" and shift to the count enable state. The count start of the TCRmn register in the count enable state varies for each mode of operation (Refer to Table 6-6 of "6.5.2 Start Timing of Counter"). When channel 1 and 3 are in 8-bit timer mode, TSm1 and TSm3 allow the operation of the low 8-bit timer.

Note:

1. Bit15 ~ 12, 10, 8 ~ 4 must be set to 0.
2. When switching from a function that does not use the TImn pin inputs to a function that does use the TImn pin inputs, the following period of wait is required from the setting of the Timer Mode Register mn (TMRmn) to the setting of the TSmn (TSHm1, TSHm3) bits to "1":

TImn Pin Noise Filter Valid Time (TNFENmn=1): 4 Runtime Clocks (f_{MCK})

TImn Pin Noise Filter Invalid Time (TNFENmn=0): 2 Runtime Clocks (f_{MCK})

Remark: 1. The read value for 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 a trigger register that sets that count stop for each channel.

If each bit is" 1", the counter bits of the timer channel allow state register m (TEm) are cleared. Because the TTmn bit, TTHm1 bit, and TTHm3 bit are trigger bits, the TTmn bit, TTHm1 bit, and TTHm3 bit are cleared immediately if the idle state (TEmn, TEHm1, TEHm3=0) occurs.

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

A lower 8-bit of the TTm register can be set with the TTmL and through 8-bit memory operation instructions.

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

Figure 6-15 Table of timer channel stop register m (TTm)

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TTm	0	0	0	0	TTH m3	0	TTH m1	0	0	0	0	0	TTm3	TTm2	TTm1	TTm0

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	TTm7	TTm6	TTm5	TTm4	TTm3	TTm2	TTm1	TTm0

m=1

TTHm3	The High 8-bit Timer Stop Trigger When Channel 3 is in 8-bit Timer Mode
0	No trigger.
1	Clear TEHm3 bit '0' to count stop state.

TTHm1	The High 8-bit Timer Stop Trigger When Channel 1 is in 8-bit Timer Mode
0	No trigger.
1	Clear TEHm1 bit '0' to count stop state.

TTmn	Idle trigger for channel n
0	No trigger.
1	Clear the TEmn bit "0" and enter the count stop state. When channel 1 and 3 are in 8-bit timer mode, TTm1 and TTm3 are triggered to stop operation of low 8-bit timers.

Note: Bit15~12,10,8~4 must be 0.

Note: 1.TTm register always reads "0".

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

6.3.8 Timer input-output select register (TIOS0)

A channel 0 and a timer input of the channel 1 of the TIOS0 register selection unit 0 and a timer output of the channel 2. The TIOS0 register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of the TIOS0 register changes to "00H".

Figure 6-16 Table of timer input-output select register

Address: 0x40020474 After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
TIOS0	TIS07	TIS06	TIS05	TIS04	TOS03	TIS02	TIS01	TIS00

TIS07	TIS06	TIS05	Selection of timer input used by channel 0
0	0	0	Timer input pin (TI0) input signal
others			Disable setting.

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

TOS03	Enable of timer output for channel 2
0	Allow Output
1	Suppress output (output fixed to 0)

TIS02	TIS01	TIS00	Selection of timer input used by channel 1
0	0	0	Input signal of timer input pin (TI01)
0	0	1	Event input signal for EVENTC
0	1	0	Input signal of timer input pin (TI01)
0	1	1	
1	0	0	Low-speed internal oscillator clock ($_{fIL}$)
1	0	1	Secondary System Clock ($_{fSUB}$)
Other than above			Disable setting.

Note: 1. The high and low level width of the selected timer input needs to be greater than or equal to 1/f_{MCK}+10ns. Therefore, when f_{SUB} is selected as f_{CLK} (CSS=1 in CKC register), the TIS02 bit cannot be set to "1".

2. When selecting the event input signal of ELC through timer input selection register 0 (TIOS0), f_{CLK} must be selected through timer clock select register 0 (TPS0).

6.3.9 Timer output enable register m (TOEm)

The TOEm register is a register that sets the output of each channel timer to be allowed or disabled.

For the channel n, the value of the TOmn bit of the timer output register m (TOm) cannot be rewritten by software.

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

A lower 8-bit of the TOEm register can be set with the TOEmL and through an 8-bit memory operation instruction. After the reset signal is generated, the value of the TOEm register changes to "0000H".

Figure 6-17 Table of timer output enable register m (TOEm)

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOEm	0	0	0	0	0	0	0	0	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	Enable/Disable of timer output for channel n
0	Disables timer output. The operation of the timer is not reflected to the TOmn bit, and the output is fixed. Can write TOmn bits and output the level of the TOmn bit setting from the TOmn pin.
1	Enable timer output. The operation of the timer is reflected to the TOmn bit, and the output waveform is generated. Writes of TOmn bits are ignored.

Note: Bit15~4 must be set to 0.

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

6.3.10 Timer output register m (TOm)

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

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

The TOmn bit of this register can only be overwritten by software if timer output (TOEmn=0) is prohibited. When timer output is allowed (TOEmn=1), the override operation through the software is ignored and the value is changed only through the timer running.

To use the TOmn pins as port functions, you must set the TOmn bit to "0".

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

The low 8 bits of the TOm register can be set with TOmL and through 8-bit operation instructions. After the reset signal is generated, the value of the TOm register becomes "0000H".

Figure 6-18 Table of timer output register m (TOm)

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOm	0	0	0	0	0	0	0	0	0	0	0	0	TOm3	TOm2	TOm1	TOm0

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

Note: Bit15~4 must be set to 0.

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

6.3.11 Timer output level register m (TOLm)

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

When a timer output (TOEmn=1) is allowed and a multi-channel linkage operation function (TOMmn=1) is used, the setting and reset timing of the output signal of the timer reflect the reversed setting of each channel n by this register. This register has an invalid setting in the TOMmn=0 (Master Channel Output Mode).

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

A low 8-bit of the TOLm register can be set with TOLmL and through 8-bit memory instructions. After the reset signal is generated, the value of the TOLm register changes to "0000H".

Figure 6-19 Table of timer output level register m (TOLm)

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOLm	0	0	0	0	0	0	0	0	0	0	0	0	TOL m3	TOL m2	TOL m1	0

m=0

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOLm	0	0	0	0	0	0	0	0	TOL m7	TOL m6	TOL m5	TOL m4	TOL m3	TOL m2	TOL m1	0

m=1

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

Note: Bit15~4 and bit0 must be set to '0'.

Remark: 1. If you override the value of this register during a timer run, invert the output logic of the timer when the next timer output signal changes, not immediately after the override.

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

6.3.12 Timer output mode register m (TOMm)

The TOMm register is a register that controls the output mode of each channel timer. When used as a stand-alone channel operation function, the corresponding bit of the used channel is "0".

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

When a timer output (TOEmn=1) is allowed, the setting and reset timing of the timer output signal reflect the setting of each channel n performed by the register.

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

A low 8-bit of the TOMm register can be set with TOMmL and through 8-bit memory instructions. After the reset signal is generated, the value of the TOMm register changes to "0000H".

Figure 6-20 Table of timer output mode register m(TOMm)

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOMm	0	0	0	0	0	0	0	0	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 timer output mode of channel n
0	Master channel output mode (alternating output by timer interrupt request signal (INTTMmn))
1	Slave channel output mode (the output is set by timer interrupt request signal (INTTMmn) of master channel and reset by timer interrupt request signal (INTTMmp) of slave channel)

Note: Bit15~4 and bit0 must be set to '0'.

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

master channel number:

n=0, 2 when m=0, n=0, 2, 4 when m=1

slave channel number p:

n<p≤3 when m=0, n<p≤7 when m=1

(For details on the relationship between master and slave channels, refer to "6.4.1 Basic prinicpal of multi-channel linkage operation function").

6.3.13 Input switch control register (ISC)

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

Refer to "19.3.14 Input Switch Control Register (ISC)" for setting SSIE00 bits. The ISC register is set by an 8-bit memory operation instruction.

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

Figure 6-21 Table of input switch control register (ISC)

Address: 0x40040473 After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
ISC	SIE00	0	0	0	0	0	ISC1	ISC0

SIE00	SSI00 pin input settings for channel 0 in slave mode for CSI00 communication
0	SSI00 pin input is invalid.
1	SSI00 pin input is valid.

ISC1	Input switching of channel 3 of universal timer unit 0
0	Use the input signal of the TI03 pin as the input (usually running) of the timer.
1	The input signal of the RxD0 pin is used as the input of the timer (detecting the wake-up signal and measuring the low level width of the break field and the pulse width of the sync field).

ISC0	External Interrupt (INTP0) input switch
0	Use the input signal of the INTP0 pin as the input for the external interrupt (usually run).
1	Use the input signal of the RxD0 pin as the input of the external interrupt (detect wake-up signal).

Note: Bit6~2 must be set to '0'.

Note: When using LIN-bus for communication, you must select the input signal for the RxD0 pin by setting the ISC1 bit to "1".

6.3.14 Noise filter enable register (NFEN1/NFEN2)

The NFEN1/NFEN2 registers set whether the noise filter is used for the input signal of each channel timer input pin. For the pin that needs to eliminate noise, the corresponding bit must be set to "1" to make the noise filter active. When the noise filter is valid, it detects whether the 2 clocks are the same after synchronizing through the running clock (f_{MCK}) of the object channel; when the noise filter is invalid, it only synchronizes through the running clock (f_{MCK}) of the object channel [Note].

The NFEN1/NFEN2 register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of the NFEN1/NFEN2 register changes to "00H".

Note: For details, refer to “6.5.1(2) Selecting the active edge of TImn pin input signal (CCSmn=1)”, “6.5.2 Start timing of counter” and “6.7 Control of timer input (TImn)”.

Figure 6-22 Table of noise filter enable 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	Whether the TI03 pin or the input signal noise filter of the RxD0 pin is used or not
0	Noise filter is OFF
1	Noise filter is ON

TNFEN02	Whether the input signal noise filter of the TI02 Pin is used or not
0	Noise filter is OFF
1	Noise filter is ON

TNFEN01	Whether the input signal noise filter of the TI01 Pin is used or not
0	Noise filter is OFF
1	Noise filter is ON

TNFEN00	Whether the input signal noise filter of the TI00 Pin is used or not
0	Noise filter is OFF
1	Noise filter is ON

Note The applicable pin can be switched by setting the ISC1 bit of the input switch control register (ISC). ISC1=0: You can choose whether to use a noise filter for the TI03 pin. ISC1=1: You can choose whether to use a noise filter for the RxD0 pin.

TNFEN1n	Whether the input signal noise filter of the TI1n Pin is used or not
0	Noise filter is OFF
1	Noise filter is ON

n=0~7

Remark: Whether the timer input/output pins of channels are different or not depends on the product. Refer to "Table 6-2 Products with Timer Input/Output Pins".

6.3.15 Registers for controlling timer input/output pin port function

When using a universal timer unit, you must set the port-functional control registers (PMxx, Pxx, and PMCxx). Refer to "2.3.1 Port Mode Register (PMxx), "2.3.2 Port Register (Pxx)" and "2.3.6 Port Mode Control Register (PMCxx) " for details.

The set port mode registers (PMxx), port registers (Pxx), and port mode control registers (PMCxx) differ by product. Refer to "Register settings when 2.5 uses the multiplexing feature" for details.

When the multiplexed port of the timer output pin is used 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

(Example) Using PA01/TO00 as timer output

Set the PMCA01 bit of Port Mode Control Register 0 to "0".

Set the PMA01 bit of Port Mode Register 0 to "0".

Set the PA01 bit of Port Register 0 to "0".

When the multiplexed port of the timer input pin is used as the input of the timer, the bit of the port mode register (PMxx) set to 1. The bit of the port register (Pxx) can be "0" or "1".

(Example) Using PA00/TI00 as timer input

Set the PMCA00 bit of Port Mode Control Register0 to "0".

Set the PMA00 bit of Port Mode Register0 to "1".

Set the PA00 bit of Port Register0 to "0" or "1".

6.4 Basic rules for universal timer units

6.4.1 Basic prinicpal of multi-channel linkage operation function

The function of multi-channel linkage operation is a function which combines the main control channel (the reference timer which mainly counts the period) and the slave channel (the timer which follows the main control channel operation), and it needs to abide by several rules.

The basic rules for the multi-channel linkage operation function are as follows.

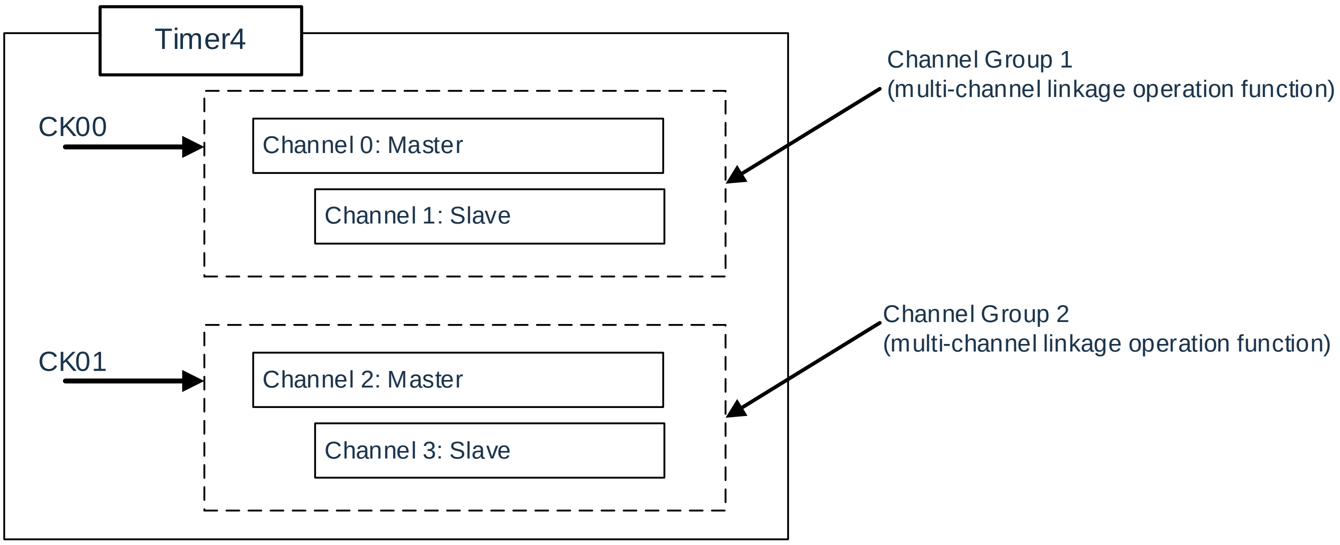
1) Only even channels (channel 0, channel 2) can be set as master channels.
2) Any channel other than channel 0 can be set as a slave channel.
3) You can only set the lower channel of the master channel as a slave channel.
   When setting channel 0 as the master channel, the channel starting channel 1 (channel 1, channel 2, channel 3) can be set as slave.
4) Multiple slave channels can be set for one master channel.
5) When using multiple master channels, you cannot set a slave channel across the master channel.
   When setting channel 0 and channel 2 as the master channel, channel 1 can be set as the slave channel of master channel 0.
6) The slave channel which is linked with the master channel needs to set the same runtime clock. The CKSmn0 bit and the value of CKSmn1 bit (bit15 and bit14 of timer mode register mn(TMRmn) of the slave channel interfaced with the master channel.
7) The main control channel can transmit INTTMmn, start software trigger and count clock to the low-level channel.
8) The slave channel can use the INTTMmn, start software trigger and count clocks of the master channel as the source clock, but cannot pass its INTTMmn, start software trigger
9) The master channel cannot use the INTTMmn, start software trigger, and count clocks of other high-level master channels as source clocks.
10) In order to start the channel to be linked at the same time, it is necessary to set the channel start trigger bit (TSmn) of the linked channel.
11) Only the full channel or master channel of the coordination can use the settings of the TSmn bits in the count run. You cannot use the settings of only the TSmn bits of the dependent channel.
12) In order to simultaneously stop the channel to be linked, it is necessary to simultaneously set the channel stop trigger bit (TTmn).
13) When the Coordinated Operation is running, you cannot select CKm2/CKm3 because the master channel and the slave channel require the same runtime clock.
14) The timer mode register m0 (TMRm0) is fixed to '0' without a master bit. However, because channel 0 is the highest-bit channel, channel 0 can be used as the master channel during the interaction run.

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

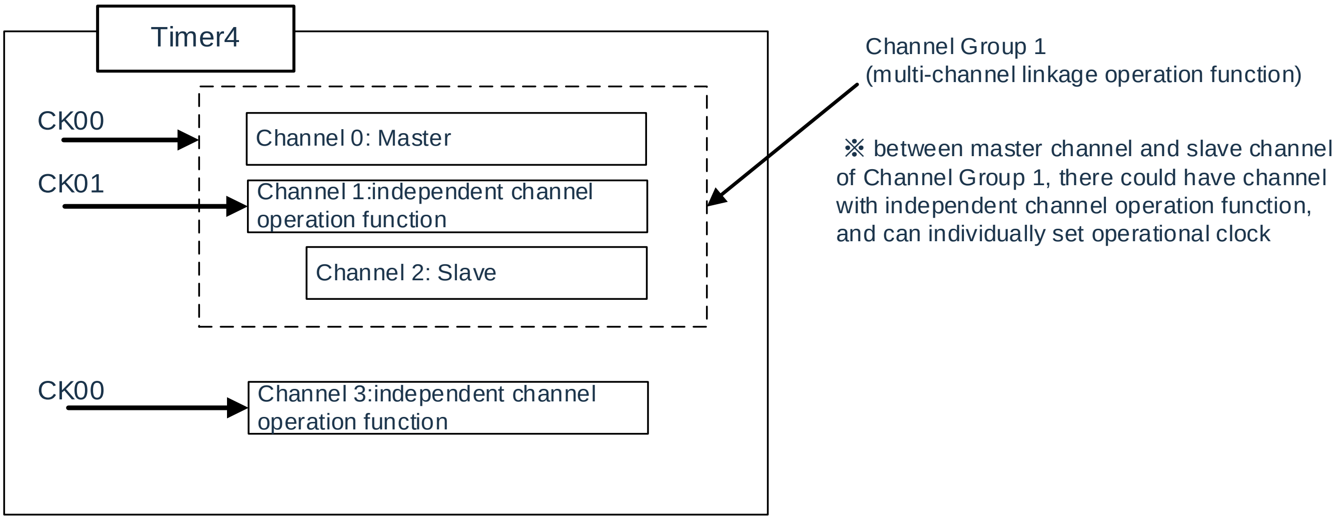
If two or more channels groups are set, the basic rules are not applicable to each other.

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

Example 1



Example 2


Timer4
CK00
CK01
Channel 0: Master
Channel 1:independent channel operation function
Channel 2: Slave
CK00
Channel 3:independent channel operation function
Channel Group 1
(multi-channel linkage operation function)

※ between master channel and slave channel of Channel Group 1, there could have channel with independent channel operation function, and can individually set operational clock

6.4.2 Timer channel start register m (TSm)

TSm register is the trigger register to initialize Timer Count Register mn (TCRmn) and set the start of each channel count operation. If each bit is set to "1", the corresponding bit of Timer Channel Enable Status Register m (TEm) is set to "1". Since the TSmn bit, TSHm1 bit and TSHm3 bit are trigger bits, the TSmn bit, TSHm1 bit and TSHm3 bit are cleared immediately if the operation is enabled (TEmn, TEHm1, TEHm3=1).

The TSm register is set by 16-bit memory operation instructions.

The TSm register can be set with TSmL and the lower 8 bits of the TSm register can be set by 8-bit memory operation instructions. After a reset signal is generated, the value of TSm register becomes "0000H".

Figure 6-23 Format of timer channel start register m (TSm)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSm	0	0	0	0	TSH m3	0	TSH m1	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	Operation enable (start) trigger for high 8-bit timer when channel 3 is 8-bit timer mode
0	No trigger.
1	Set the TEHm3 bit to "1" to enter the count permit state. If the TCRm3 register starts counting in the count-enable state, it enters the interval timer mode (refer to Table 6-6 of "6.5.2 Start timing of counter ").

TSHm1	Operation enable (start) trigger for high 8-bit timer when channel 1 is 8-bit timer mode
0	No trigger.
1	Set TEHm1 to "1" to enter the count-allowed state. If the count of TCRm1 register is started in the count allowed state, it enters the interval timer mode (refer to Table 6-6 of "6.5.2 Start timing of counter ").

TSmn	Operation enable (start) trigger for channel n
0	No trigger.
1	Set TEmn to "1" to enter the count-allowed state. The count start of the TCRmn register in the count allow state varies with each operation mode (refer to Table 6-6 of "6.5.2 Start timing of counter "). When channel 1 and channel 3 are in 8-bit timer mode, TSm1 and TSM3 are in the run-allowed (start) trigger state of the low 8-bit timer.

Note: 1. Bits 15 ~ 12, 10, 8 ~ 4 must be set to "0".

2. When switching from a function that does not use the TImn pin input to a function that uses the TImn pin input, the following period of waiting is required from setting the Timer Mode Registermn (TMRmn) until the TSmn (TSHm1, TSHm3) bits are set to "1":

When the TImn pin noise filter is valid (TNFENmn=1): 4 running clocks (f_{MCK})

When the TImn pin noise filter is invalid (TNFENmn=0): 2 running clocks (f_{MCK})

Note: 1. The read value of TSm register is always "0".
2. m: Unit number (m=0, 1) n: Channel number (when m=0: n=0 ~ 3, when m=1: n=0 ~ 7)

6.4.3 Basic principle for the 8-bit timer operation function (channel 1 and channel 3 of unit 0 only)

The 8-bit timer run function is the function of using the channel of the 16-bit timer as the channel of two 8-bit timers.

Only channel 1 and channel 3 can run functions with an 8-bit timer, which requires several rules.

The basic rules for the 8-bit timer operation function are as follows.

1) The 8-bit timer run function is available only for channel 1 and channel 3.
2) When used as an 8-bit timer, the SPLIT bit of timer mode register mn(TMRmn) is "1".
3) High-8-bit timers can be used as interval timer function.
4) The high 8-bit timer outputs INTTMm1H/INTTMm3H (the same as the run with the MDmn0 bit "1") at start.
5) The selection of the running clock of the high 8-bit timer depends on the setting of the CKSmn1 bit and the CKSmn0 bit of the low TMRmn register.
6) For a high 8-bit timer, the operation of the channel is started by operating the TSHm1/TSHm3 bit, and the operation of the channel is stopped. The status of the channel can be confirmed by TEHm1/TEHm3 bits.
7) The operation of the low-8-bit timer depends on the setting of the TMRmn register, which has three functions to support the operation:
   - Interval timer
   - External event counter
   - Delay count
8) For low-8 bit timers, channel operation is started by operating the TSm1/TSm3 bit, and channel operation is stopped by operating the TTm1/TTm3 bit. The status of the channel can be confirmed by TEm1/TEm3 bits.
9) The operation of the TSHm1/TSHm3/TTHm1/TTHm3 bit is not valid when the 16 bit timer runs. Channel 1 and channel 3 are run by operating TSm1/TSm3 and TTm1/TTm3 bits. The TEHm3 bit and the TEHm1 bit are unchanged.
10) The 8-bit timer function does not use the coordinated operation function (single trigger pulse, PWM, and multiple PWM).

Note: m: Unit number (m=0)n: Channel number (n=1,3)

6.5 Operation of counter

6.5.1 Count clock (f_{TCLK})

The counting clock (f_{TCLK}) of the universal timer unit can select any one of the following clock by the CCSmn bit of the timer mode register mn (TMRmn):

- CKSmn0-bit and CKSmn1-bit specified runtime clocks (f_{MCK})
- Effective edge of TImn pin input signal

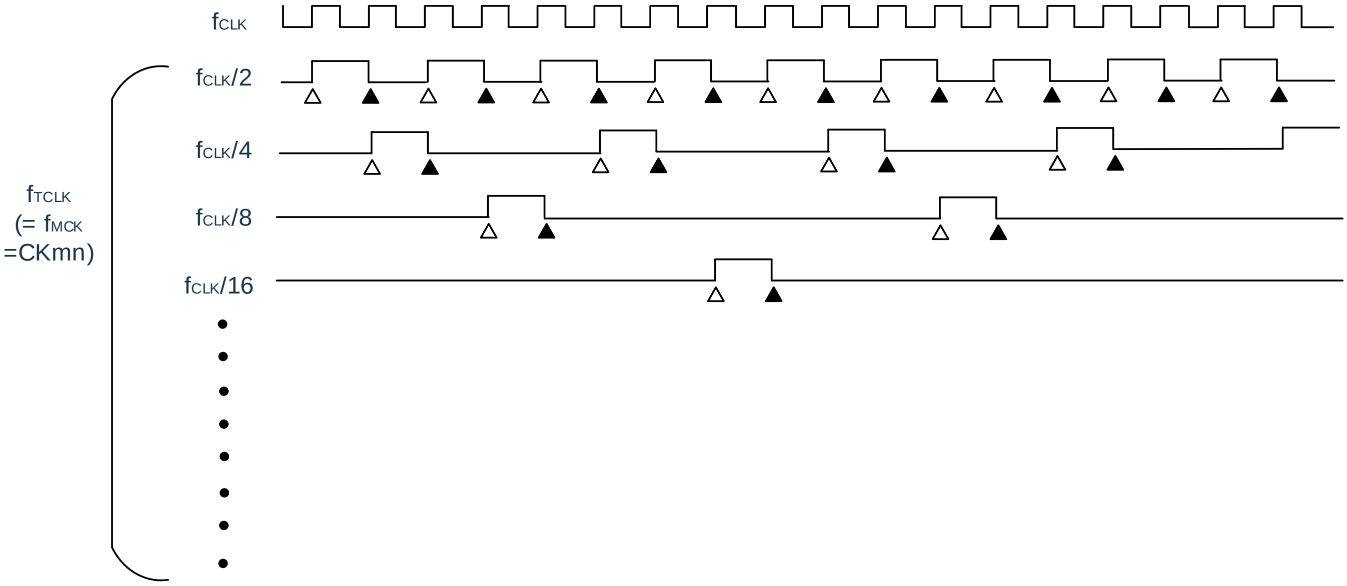
The universal timer unit is designed to run synchronously with the f_{CLK}, so the timing of the counting clock (f_{TCLK}) is as following.

(1) When selecting the running clock (f_{MCK}) specified by the CKSmn0 bit and CKSmn1 bit (CCSmn=0)

According to the setting of timer clock selection register m (TPSm), the count clock (f_{TCLK}) is f_{CLK}～$f_{CLK}/2^{15}$. However, when selecting the frequency division of the f_{CLK}, the TPSm register selects a clock that is only 1 f_{CLK} period of high level signal from the rising edge. Fixed to high level when f_{CLK} is selected.

In order to synchronize with f_{CLK} , timer counter register mn (TCRmn) counts after delaying 1 f_{CLK} clock from the rising edge of the counting clock.

Figure 6-24 Timing of f_{CLK} (f_{TCLK}) and count clock (when CCSmn=0)



Remarks: 1.△: Rising edge of counting clock

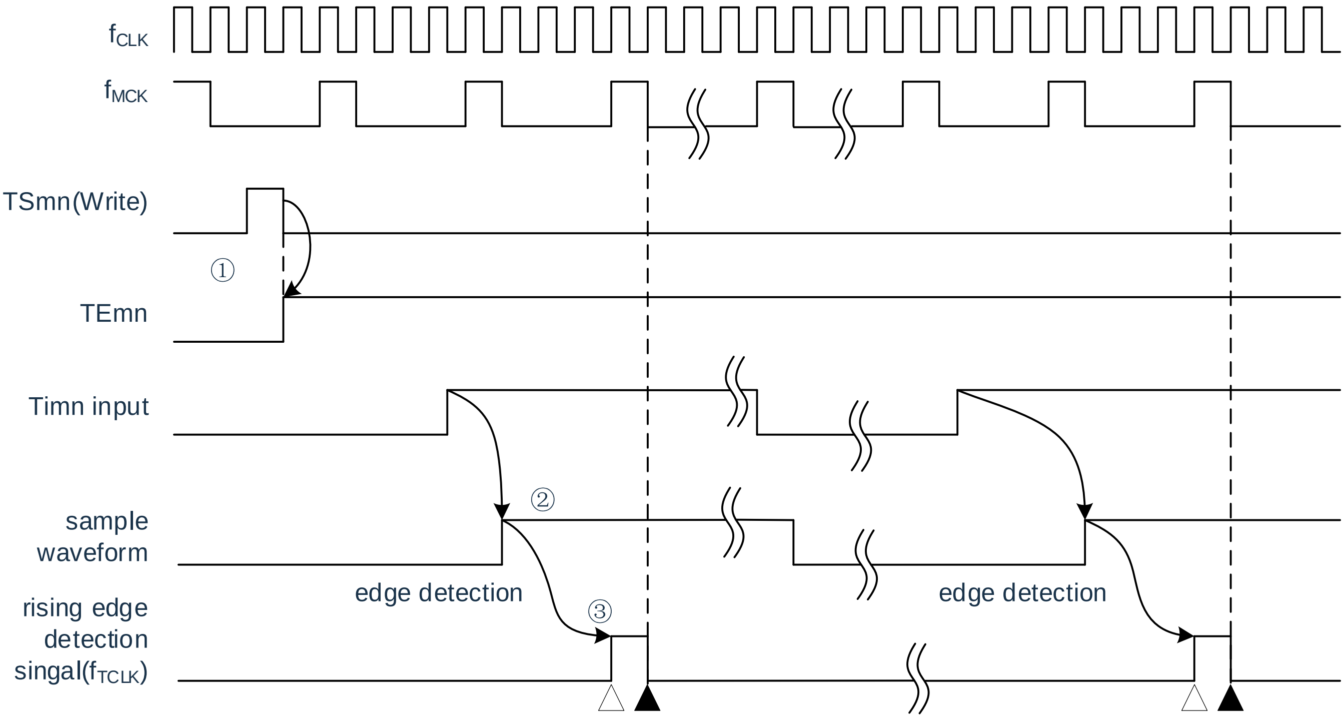
▲: Synchronization, Counter Increment/Decrement

2. f_{CLK}: Clock for CPU/peripheral hardware

(2) When selecting the active edge of the input signal on the TImn pin (CCSmn=1)

The count clock (f_{TCLK}) is a signal that detects an effective edge of the TImn pin input signal and is synchronized with the next f_{MCK} rising edge. In fact, this is a signal delayed by 1~2 f_{MCK} clocks compared to the input signal of the TImn pin (3~4 f_{MCK} clocks when using noise filters). In order to synchronize with f_{CLK}, timer counter register mn(TCRmn) counts after delaying 1 f_{CLK} from the rising edge of the counting clock, which is called "counting at the effective edge of TImn pin input signal".

Figure 6-25 Timing of the count clock (f_{TCLK}) (CCSmn=1, if no noise filter is used)



① The operation of the timer is started by setting the TSmn bit and the valid edge of the TImn input is awaited.

② Sample the rising edge of the TImn input through f_{MCK}.

③ A detection signal (counting clock) is output at a rising edge of the sampling signal.

Remarks: 1.△: Rising edge of counting clock

▲: Synchronization, Counter Increment/Decrement

2. f_{CLK}: CPU/Peripheral Hardware Clock

f_{MCK}: Operation clock of channel n

3. The same waveforms are measured for input pulse interval, input signal high and low level, delay counter, and TImn input with single trigger pulse output.

6.5.2 Start timing of counter

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

Execution from counting enable state up to the start of count register mn (TCRmn) is shown in Table 6-6.

Table 6-6 Operation from the count enable state until the timer count register mn (TCRmn) starts counting

Operation mode of the timer	Operation after setting TSmn bit to " 1"
Interval timer mode	No operation is performed from the detection of the start trigger (TSmn=1) until the count clock is generated. The value of the TDRmn register is loaded into the TCRmn register by the 1 count clock and decremented by the subsequent count clock (refer to "6.5.3 (1) operation of interval timer mode").
Event counter mode	Load the value of the TDRmn register into the TCRmn register by writing "1" to the TSmn bit. If an input edge of the TImn is detected, the count is decremented by a subsequent count clock. (Refer to "6.5.3(2) Operation of Event Counter Mode").
Capture mode	No operation is performed from the start of triggering of detection until the count clock is generated. The 0000H is loaded into the TCRmn register by the 1 count clock and counted incrementally by the subsequent count clock (reference to "6.5.3 (3) Operation of capture mode (interval measurement of input pulses)).
Single count mode	The first trigger is entered by writing "1" to the TSmn bit in the state where the timer stops running (TEmn=0) Pending status. No operation is performed from the start of triggering of detection until the count clock is generated. The value of the TDRmn register is loaded into the TCRmn register by the first count clock, and a subsequent count is performed Count clocks for decremental counts (refer to "6.5.3 (4) Operation of Single Count Mode").
Capture & single count mode	The first trigger is entered by writing "1" to the TSmn bit in the state where the timer stops running (TEmn=0) Pending status. No operation is performed from the start of triggering of detection until the count clock is generated. The '0000H' is loaded into the TCRmn register through the first count clock and is performed through a subsequent count clock Incremental Count (refer to " 6.5.3(5) Operation of Capture & Single Count Mode (Measurement of High Level Width) "

6.5.3 Operation of counter

The following describes the counter operation for each mode.

(1) Operation of interval timer mode

(1) Enter the running permission state (TEmn=1) by writing "1" to TSmn bits. The timer count register mn (TCRmn) maintains an initial value until a count clock is generated.

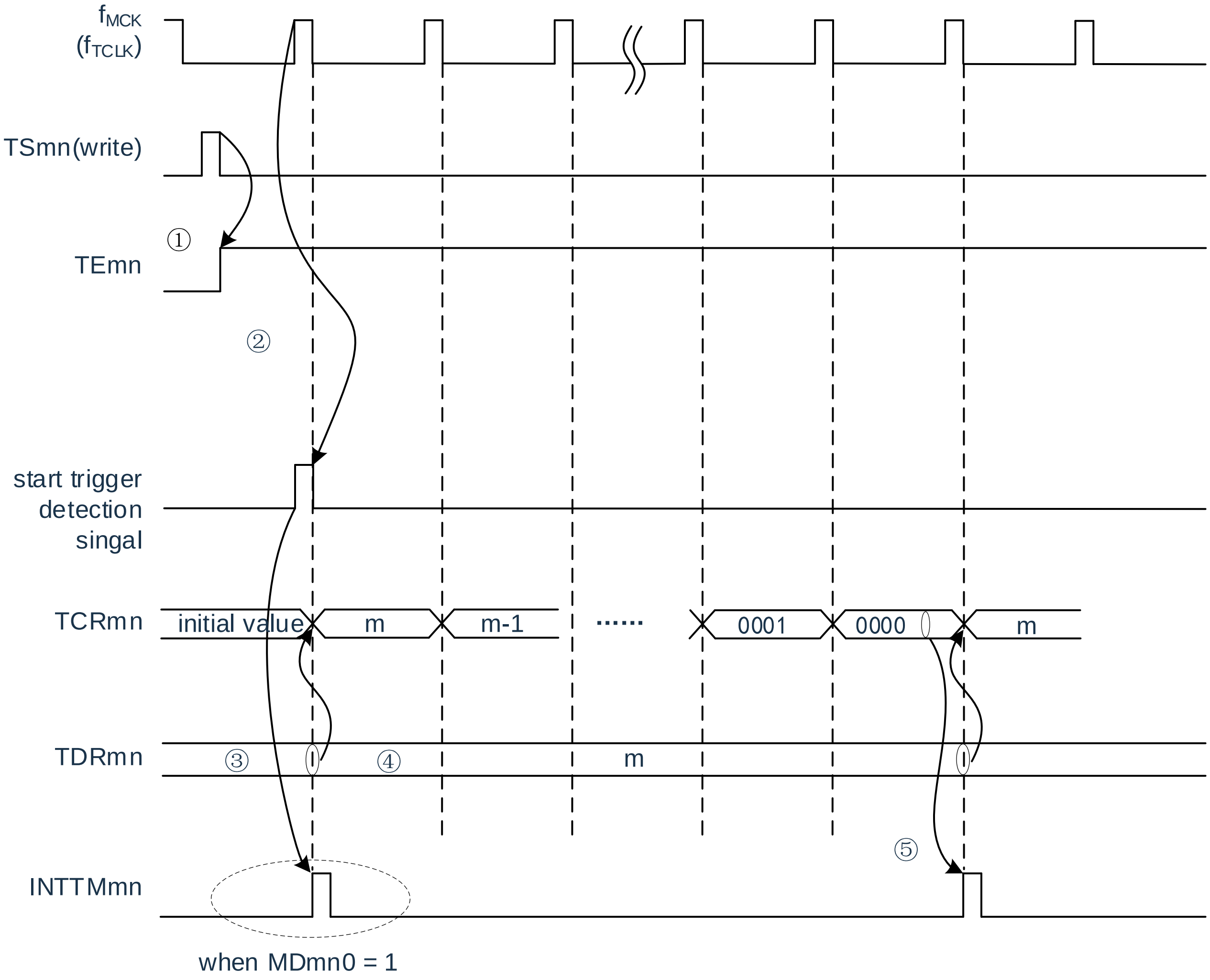
(2) A start trigger signal is generated by allowing the first count clock (f_{MCK}) after operation.

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

(4) The timer data register mn (TDRmn) is loaded into the TCRmn register by allowing the first counting clock to run.

(5) If the TCRmn register decrements to "0000H", INTTMmn is generated by the next counting clock (f_{MCK}) and continues counting.

Figure 6-26 Operation timing (interval timer mode)



Notice: Since the operation of the first count clock cycle is delayed after writing the TSmn bit and before the count clock is generated, an error of up to 1 clock cycle is generated. Also, if you need information about the start of the count timing, set the MDmn0 bit to "1" so that an interrupt can be generated when the counting starts.

Remark: f_{MCK}, the start trigger detection signal and INTTMmn are synchronized with fCLK and are valid for 1 clock.

(2) Operation of event counter mode

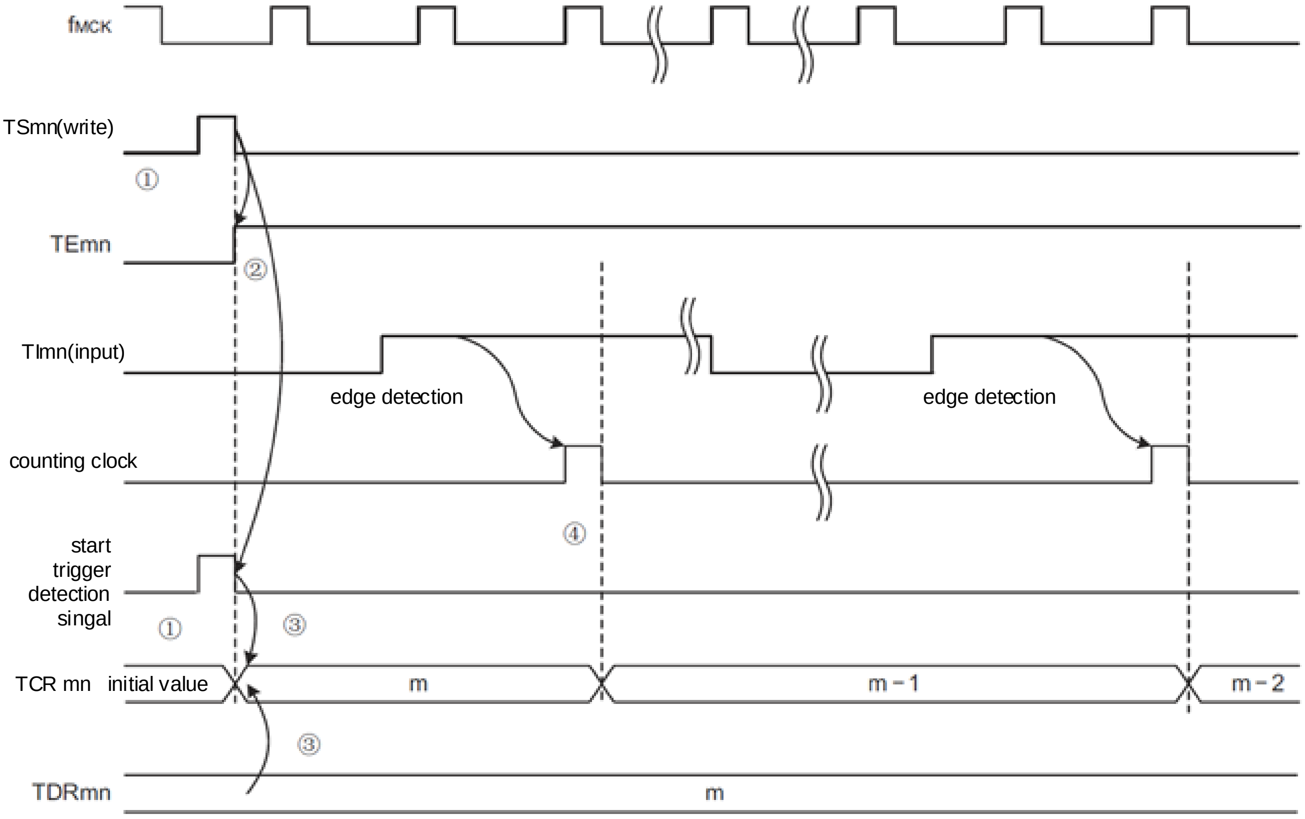
(1) The timer count register mn (TCRmn) maintains the initial value during the operation of the stop state (TEmn=0).

(2) Enter the running permission state (TEmn=1) by writing "1" to TSmn bits.

(3) Loading the value of timer data register mn (TDRmn) into the TCRmn register when both the TSmn bit and the TEmn bit become "1".

(4) Then, the value of the TImn register is decremented by the counting clock at the effective edge of the TCRmn input.

Figure 6-27 Operation timing (event counter mode)



Remark: This is the timing when no noise filter is used. If a noise filter is used, edge detection delays an additional 2 f_{MCK} cycles (3-4 cycles total) from the TImn input time. The error of 1 cycle is due to the TImn input being out of sync with the count clock (f_{MCK}).

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

① Enter the running permission state (TEmn=1) by writing "1" to the TSmn bit.

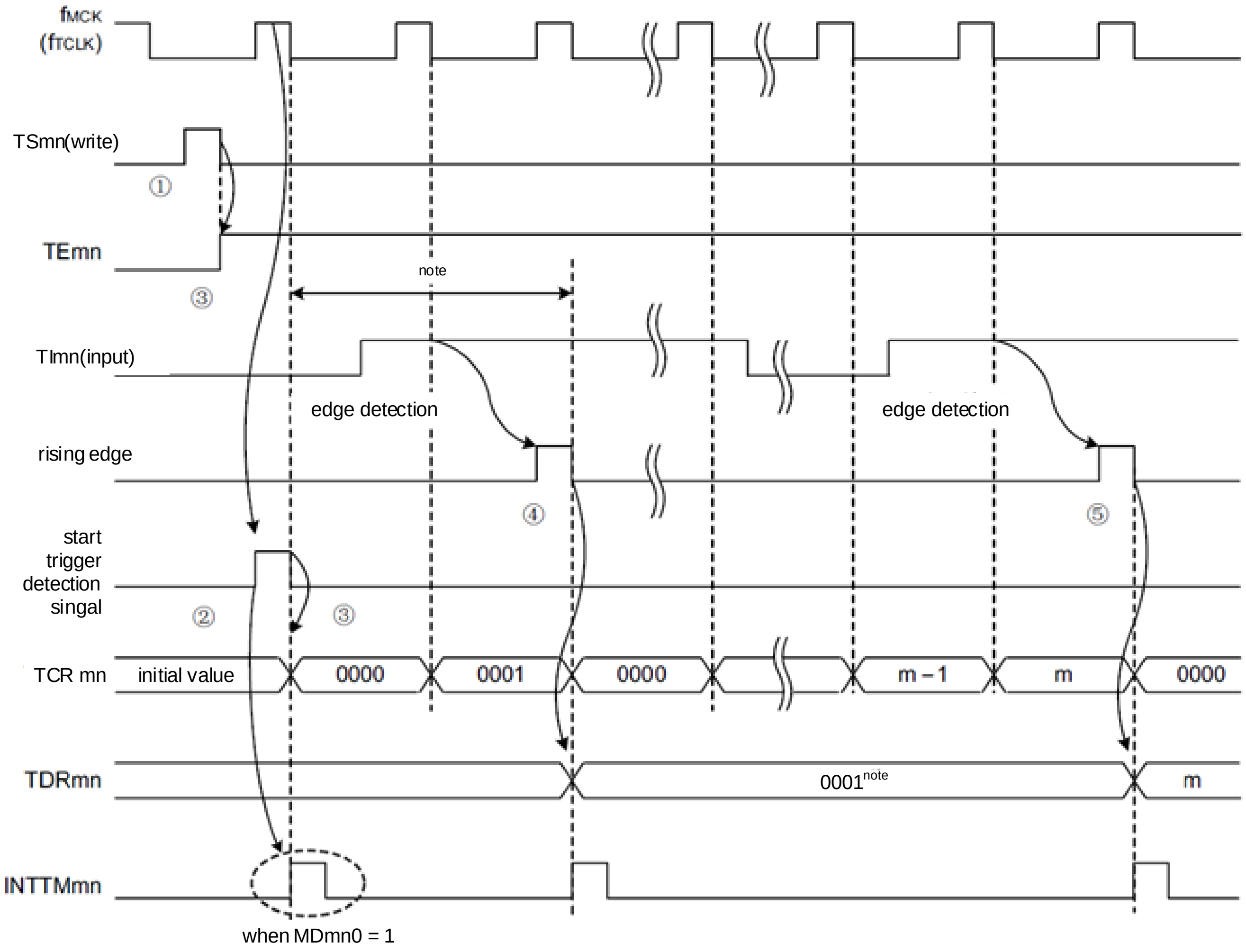
② The timer count register mn(TCRmn) keeps the initial value until the count clock is generated.

③ A start trigger signal is generated by allowing the first count clock (f_{MCK}) after operation. The '0000H' is then loaded into the TCRmn register and counts in capture mode (INTTMmn is generated by the start trigger signal when the MDmn0 bit is '1')

④ If the valid edge of the TImn input is detected, the value of the TCRmn register is captured to the TDRmn register and INTTMmn interrupt is generated. The catch value at this time is meaningless. The TCRmn register continues to count from '0000H'.

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

Figure 6-28 Operation timing (capture mode: interval measurement of input pulse)



Note: When the clock is input to the TImn (triggered) before starting, the count is started by detecting the trigger even if no edge is detected, therefore the capture value of the first capture is not pulse interval (in this example, 0001: 2 clock cycle interval), it shall be ignored.

Notice: Since the operation of the first count clock cycle is delayed after writing the TSmn bit and before the count clock is generated, an error of up to 1 clock cycle is generated. Also, if you need information about the start of the count timing, set the MDmn0 bit to "1" so that an interrupt can be generated when the counting starts.

Remark: This is the timing when no noise filter is used. If a noise filter is used, edge detection delays an additional 2 f_{MCK} cycles (3-4 cycles total) from the TImn input time. The error of 1 cycle is due to the TImn input being out of sync with the count clock (f_{MCK}).

(4) Operation of single count mode

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

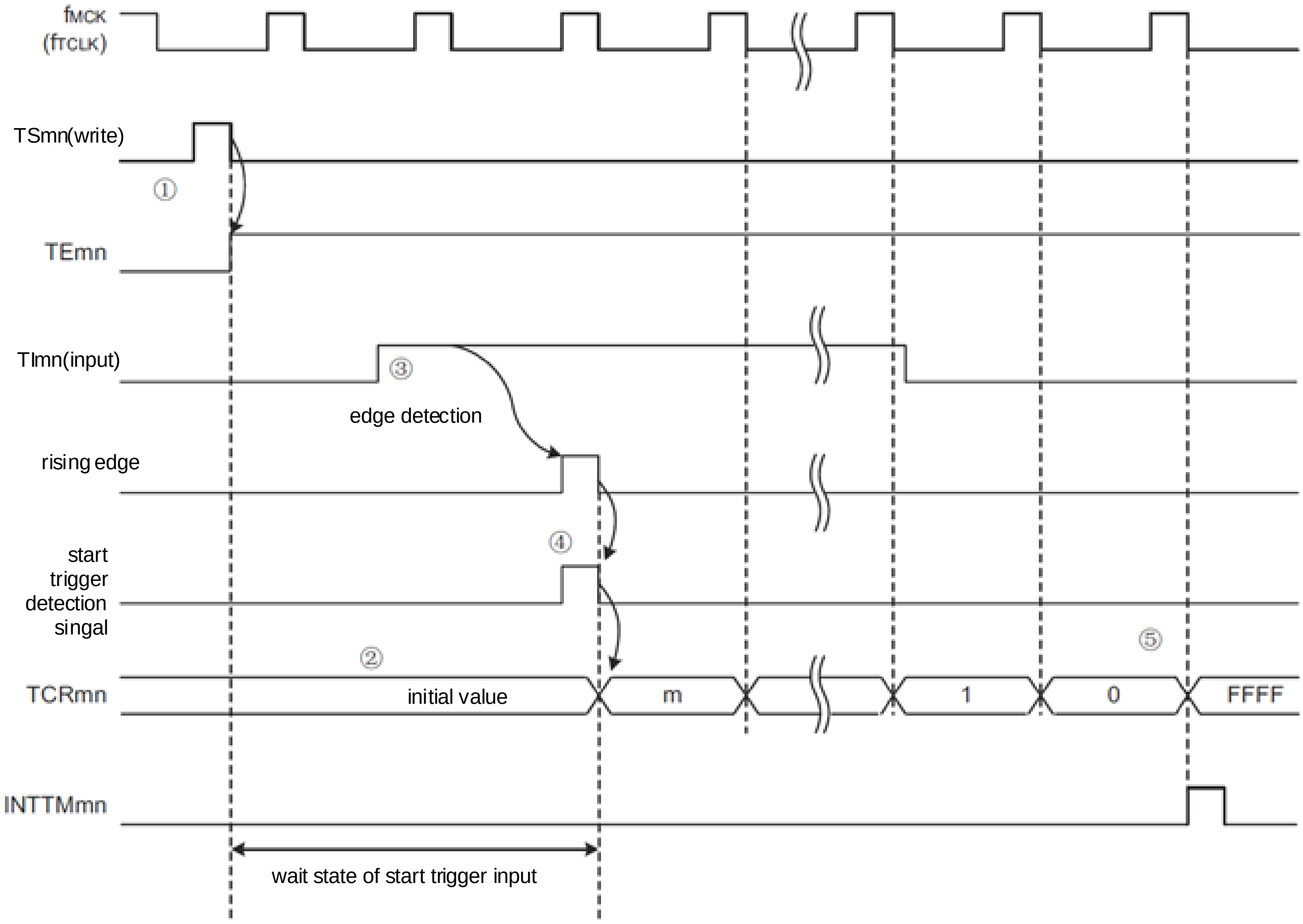
② The timer count register mn(TCRmn) keeps the initial value until the start trigger signal is generated.

③ Detect the rising edge of TImn input.

④ After the start trigger signal is generated, the value (m) of the TDRmn register is loaded into the TCRmn register and counting is started.

⑤ When the TCRmn register decrements to "0000H", INTTMmn interrupts are generated, and the TCRmn register changes to "FFFFH" to stop counting.

Figure 6-29 Operation timing (single count mode)



Remark: This is the timing when no noise filter is used. If a noise filter is used, edge detection delays an additional 2 f_{MCK} cycles (3-4 cycles total) from the TImn input time. The error of 1 cycle is due to the TImn input being out of sync with the count clock (f_{MCK}).

(5) Capture & single count mode operation (voltage high level width measurement)

① Enter the running allowed state (TEmn=1) by writing "1" with the TSmn bit of the timer channel start register m.

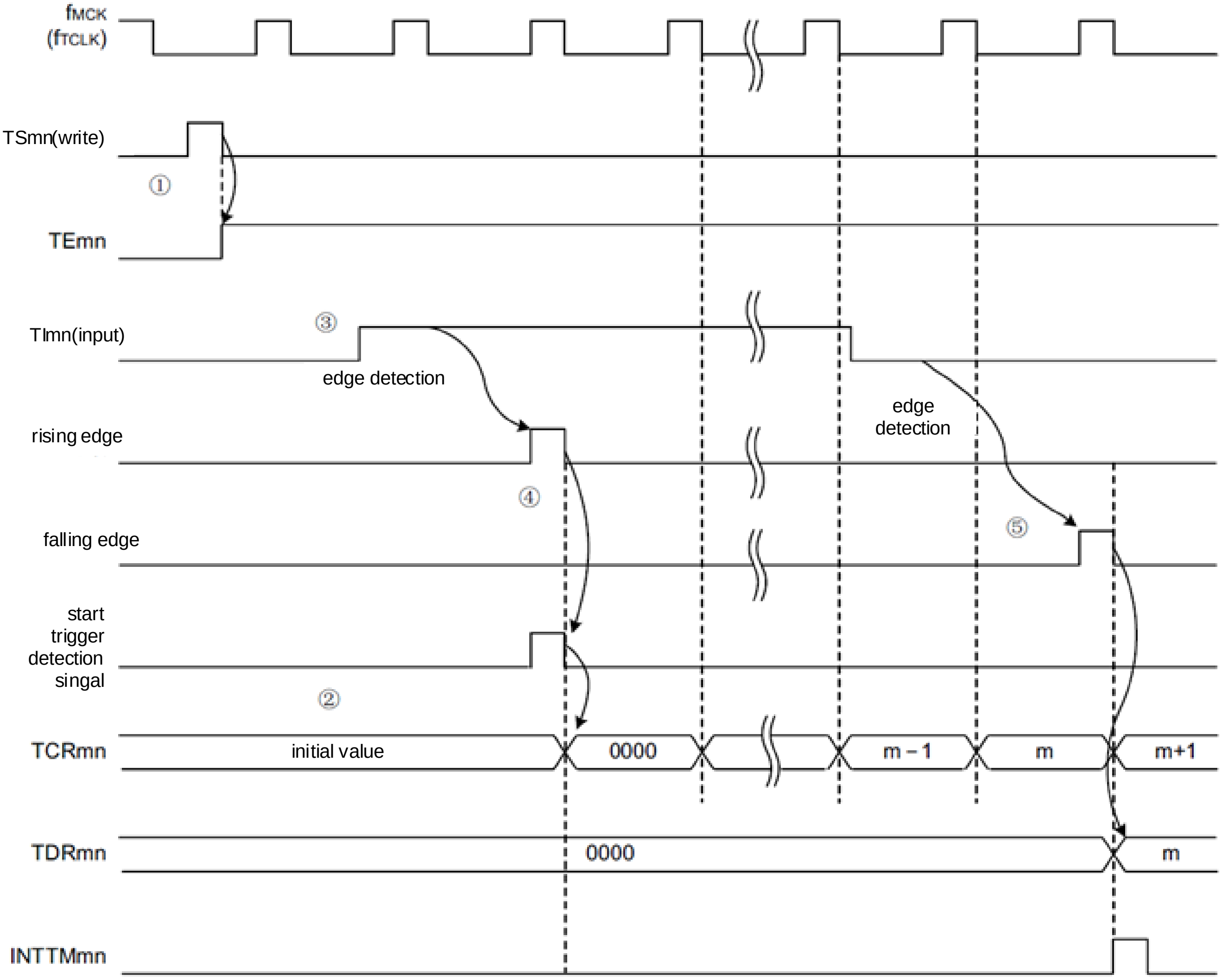
② The timer count register mn(TCRmn) keeps the initial value until the start trigger signal is generated.

③ Detect the rising edge of TImn input.

④ Loading the "0000H" into the TCRmn register after generating the start trigger signal and starting the 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 INTTMmn interrupt is generated.

Figure 6-30 Operation timing (capture & single count mode: high-level width measurement)

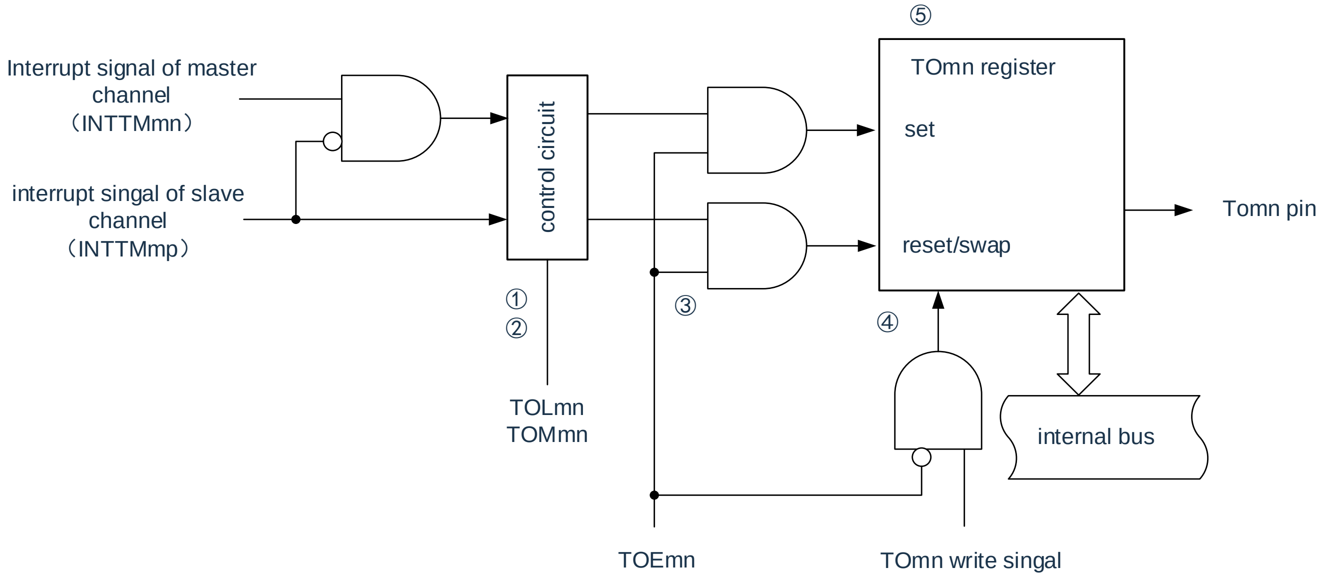


Remark: This is the timing when no noise filter is used. If a noise filter is used, edge detection delays an additional 2 f_{MCK} cycles (3-4 cycles total) from the TImn input time. The error of 1 cycle is due to the TImn input and count clock (f_{MCK}) being out of sync.

6.6 Control of channel output (TOmn pin)

6.6.1 Structure of TOmn pin output circuit

Figure 6-31 Structure of output circuit



The output circuit for the TOmn pin is described below.

① When the TOMmn bit is "0" (main channel output mode), the timer output level register m (TOLm) is ignored.

② When the TOMmn bit is "1" (slave channel output mode), the INTTMmn and INTTMmp (slave channel timer interrupt) are passed to TOm registers.

At this time, the TOLm register is valid and controls the following signal:

TOLmn=0 time: Forward Running (INTTMmn→ set, INTTMmp→ reset)

TOLmn=1 time: Invert run (INTTMmn→ reset, INTTMmp→ set)

When both INTTMmn and INTTMmp are generated (0% of PWM output), INTTMmp is preferentially used to mask INTTMmn.

③ The INTTMmn and INTTMmp are transferred to the TOm register in a state where timer output (TOEmn=1) is allowed. Write operation (TOmn write signal) is invalid for TOm register.

When the TOEmn bit is '1', the output of the TOmn pin is not changed except for the interrupt signal.

To initialize the output level of the TOmn pin, write values to the TOm register after being set to disable the timer output (TOEmn=0).

④ Write operations (TOmn write signals) of the TOmn bits of the object channel are valid in the state where timer output (TOEmn=0) is disabled. When the timer output is forbidden (TOEmn=0), INTTMmn and INTTMmp are not transferred to the TOm register.

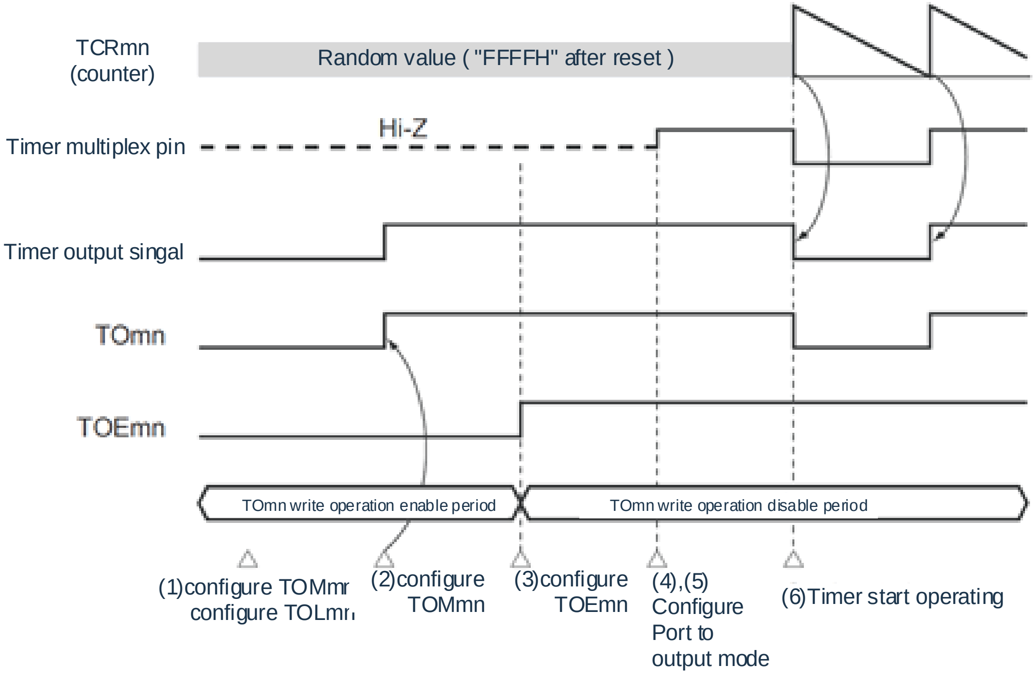
⑤ The TOm register can be read at any time and the output level of the TOmn pin can be confirmed.

Note m: Unit number (m=0, 1)
n: Channel number (n=0~3 when m=0, n=0~7 when m=1) (master channel: n=0,2,4,6)
p: Slave channel number
n=0: p=1, 2, 3
n=2: p=3

6.6.2 TOmn pin output configuration

The steps and status changes from the initial setting of the TOmn output pin to the start of the timer run are shown below.

Figure 6-32 State change from setting timer output to start of operation



①Set the operation mode output by the timer.

- TOMmn bit (0: Main control channel output mode,1: slave channel output mode)
- TOLmn bit (0: positive logical output, 1: negative logical output)

②Set the timer output signal to the initial state by setting the timer output register m (TOm).

③Write "1" to the TOEmn bit to allow timer output (writing to TOm register is disabled).

④Set the port as digital input/output through the port mode control register (PMCxx) (refer to “6.3.15”).

⑤Set the port input/output as output (refer to “6.3.15 register for controlling the timer input/output pin port function”).

⑥Allow the timer to run (TSmn=1).

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

6.6.3 Cautions for channel output operation

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

The operation of the timer (timer count register mn (TCRmn) and timer data register mn (TDRmn)) and the TOmn output circuit are independent. Therefore, the change of timer output register m (TOm), timer output permission register m (TOEm)TOLm does not affect the operation of timer. However, in order to output the expected waveform from the TOmn pin during the operation of each timer, the registers for each operation shown in 6.8 and 6.9 must be set.

If the setting values of TOEm register and TOLm register except TOm register are changed before and after the INTTMmn signal is generated for each channel.

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

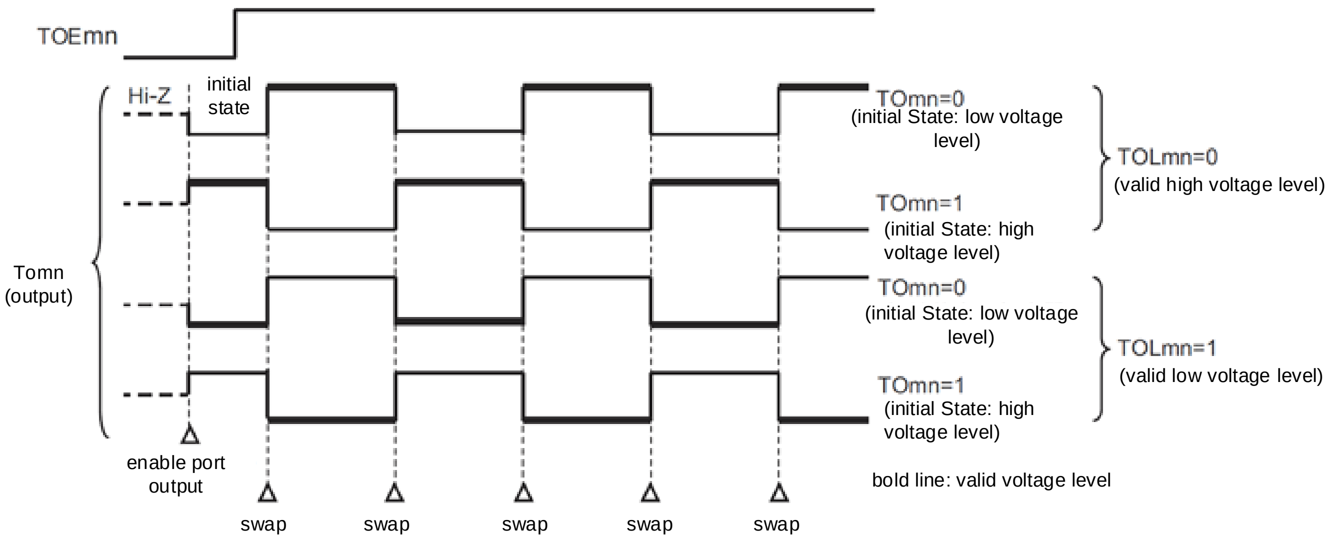
2) Initial level for the TOmn pin and output level after the timer starts to run

A timer output register m (TOm) is written before the port output is allowed and the timer output (TOEmn=0) is prohibited.

(a) The case when the operation starts in TOMmn=0 (Main Control Channel Output Mode)

The timer output level register m(TOLm) is not set in TOMmn=0. If operation of the timer is started after setting the initial level, the output level of the TOmn pin is inverted by generating alternating signals.

Figure 6-33 TOmn pin output state at alternate output (TOMmn=0)

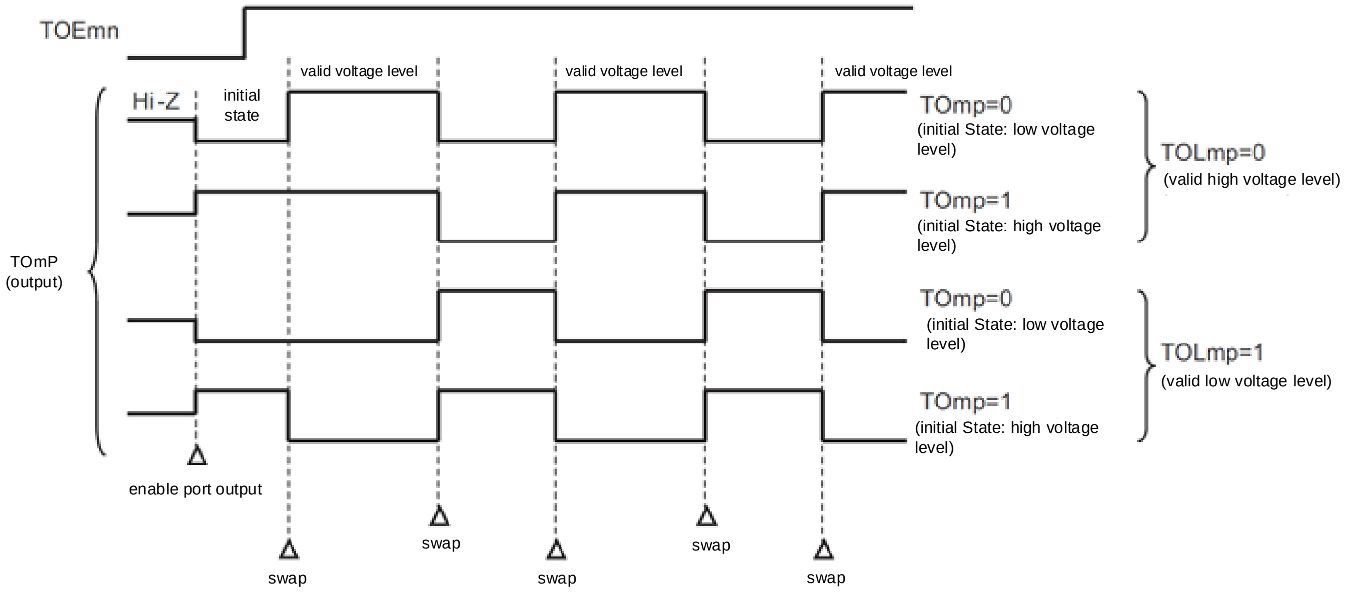

TOEmn
Hi-Z
initial state
Tomn (output)
TOmn=0 (initial State: low voltage level)
TOmn=1 (initial State: high voltage level)
TOLmn=0 (valid high voltage level)
TOmn=0 (initial State: low voltage level)
TOmn=1 (initial State: high voltage level)
TOLmn=1 (valid low voltage level)
enable port output
swap
swap
swap
swap
swap
bold line: valid voltage level

Note: 1. Swap: Output state of the inverted TOmn pin.
2. m: Unit number (m=0, 1); n: Channel number (n=0~3 when m=0, n=0~7 when m=1)

(b) Start of operation in slave channel output mode (TOMmn=1) (PWM output)

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

Figure 6-34 Output state of TOmn pin at PWM output (TOMmn=1)


TOEmn
Hi-Z
initial state
valid voltage level
valid voltage level
valid voltage level
TOmP (output)
TOmp=0
(initial State: low voltage level)
TOmp=1
(initial State: high voltage level)
TOLmp=0
(valid high voltage level)
TOmp=0
(initial State: low voltage level)
TOmp=1
(initial State: high voltage level)
TOLmp=1
(valid low voltage level)
enable port output
swap
swap
swap
swap
swap

Note: 1. Set: The output signal from the TOmp pin changes from an invalid level to an active level.

Reset: The output signal from the TOmp pin changes from the active level to the invalid level.

2.m: Unit number (m=0, 1)n: Channel number (p=1~3)

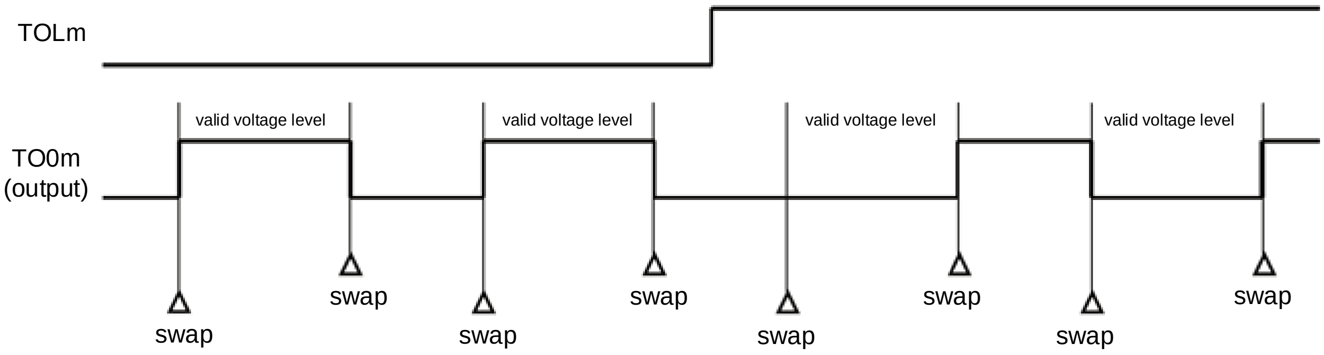
3) TOmn pin changes for slave channel output mode (TOMmn=1)

(a) The case of changing the setting of the timer output level register m (TOLm) in the timer operation

If you change the setting of the TOLm register during the timer run, the setting is valid when the TOmn pin change condition occurs. The output level of the TOmn pin cannot be changed by rewriting the TOLm register.

When the TOMmn bit is "1", the operation when the value of TOLm register is changed during the timer operation (TEmn=1) is shown below.

Figure 6-35 Run when changing the contents of the TOLm register in a timer run



Note: 1. Set: The output signal from the TOmn pin changes from an invalid level to an active level.
Reset: The output signal from the TOmn pin changes from the active level to the invalid level.
2. m: Unit number (m=0, 1); n: Channel Number (n=0~3 when m=0, n=0~7 when m=1)

(b) Set/reset timing

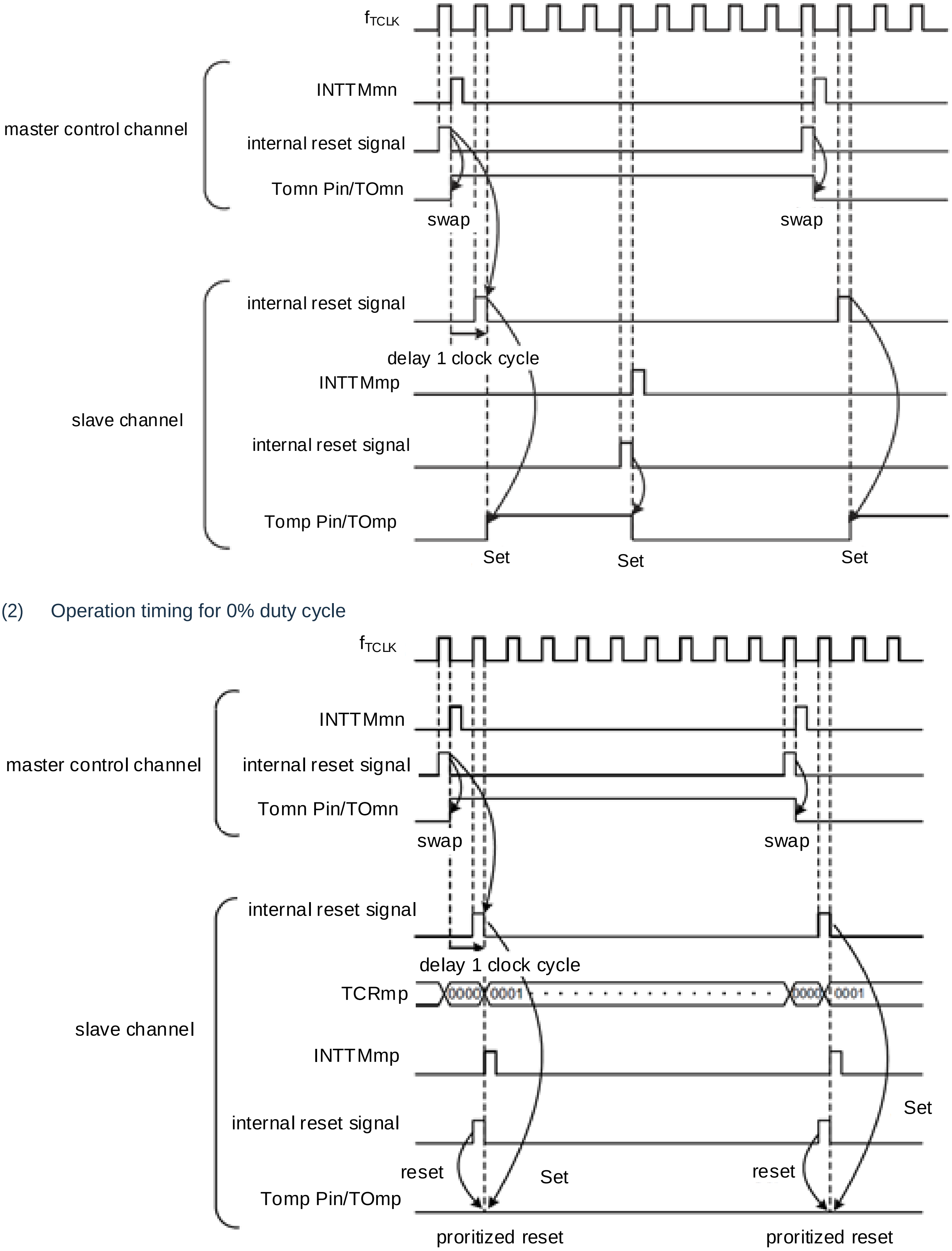
In order to achieve 0% and 100% output at PWM output, through sliave channel, the timing of TOmn pin /TOmn bit of master channel timer interrupt (INTTMmn) is delayed by 1 counting clock cycle.

When the setting condition and the reset condition are generated at the same time, the reset condition is prioritized.

The set/reset operation state when the master/slave channel is set as shown in Figure 6-35 in this manner.

Master channel: TOEmn=1, TOMmn=0, TOLmn=0
Slave channel: TOEmp=1, TOMmp=1, TOLmp=0

Figure 6-36 Set/reset timing operation status



(1) basic operation timing

(2) Operation timing for 0% duty cycle

Note 1. Internal reset signal: TOmn pin reset/swap signal

Internal set signal: set signal for the TOmn pin

2.m: Unit number (m=0, 1)

n: Channel number n=0~3 when m=0, n=0~7 when m=1 (master 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 TOmn bit

The timer output register m (TOm) has all channel setting bits (TOmn), so it can operate all channel TOmn bits.

Figure 6-37 Example of one-time operation of TO0n bit

Before

TO0	0	0	0	0	0	0	0	0	0	0	0	0	TO0 31	TO0 20	TO0 11	TO0 00
TOE0	0	0	0	0	0	0	0	0	0	0	0	0	TOE0 30	TOE0 20	TOE0 10	TOE0 01

Data to write

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

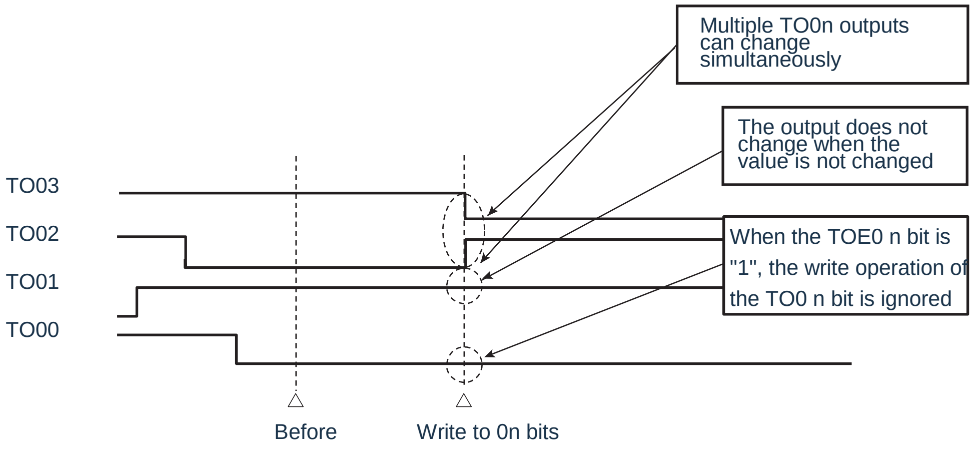
After writing

													↓	↓	↓	✕
TO0	0	0	0	0	0	0	0	0	0	0	0	0	TO0 30	TO0 21	TO0 11	TO0 00

Only TOmn bits with TOEmn bit '0' can be written, ignoring TOmn bits with TOEmn bit '1'.

TOmn with TOEmn bit "1" is not affected by write operation, even TOmn bit is ignored.

Figure 6-38 TO0n pin status at TO0n bit for one-time operation


Multiple TO0n outputs can change simultaneously
The output does not change when the value is not changed
When the TOE0 n bit is "1", the write operation of the TO0 n bit is ignored
TO03
TO02
TO01
TO00
Before
Write to 0n bits

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

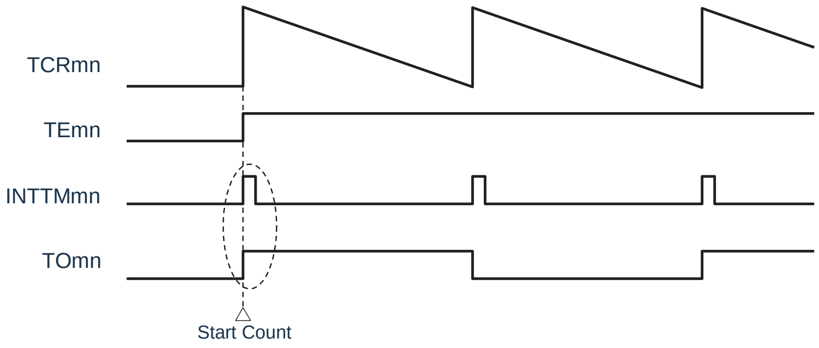
6.6.5 Timer interrupt and TOmn pin output when counting starts

In interval timer mode or capture mode, the MDmn0 bit of timer mode register mn (TMRmn) is set as the bit that generates timer interrupt when counting starts.

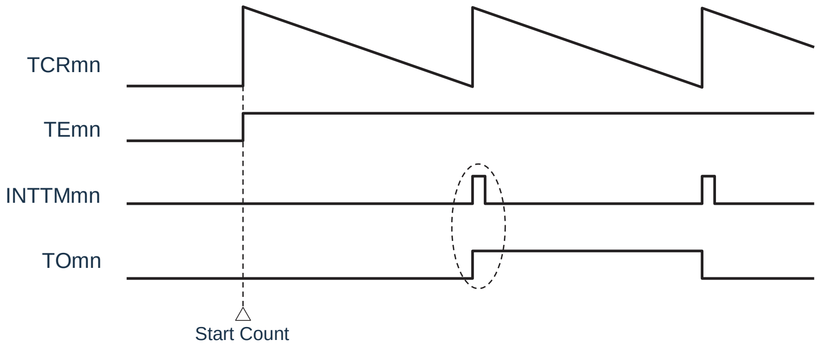
When the MDmn0 bit is '1', the start sequence of the count can be known by generating a timer interrupt (INTTMmn). In other mode, timer interrupt and TOmn output at that start of the count are not controlled. An example of a run when set to interval timer mode (TOEmn=1, TOMmn=0) is as follows.

Figure 6-39 Examples of timer interrupt and TOmn pin output when counting starts

(a) When MDmn0 bit is "1"



(b) When MDmn0 bit is '0'



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

When the MDmn0 bit is '0', no timer interrupt (INTTMmn) is output and TOmn is unchanged at the start of the count, and INTTMmn is output alternately by TOmn.

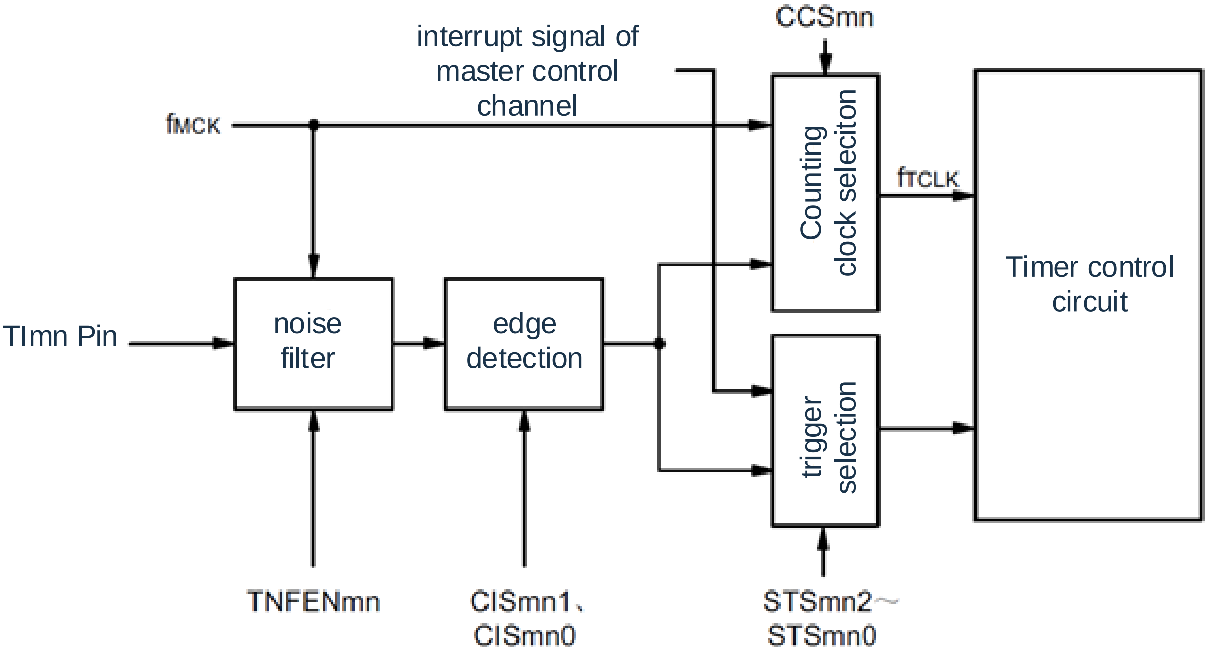
Note: m: Unit number (m=0, 1); n: Channel number (n=0~3 when m=0, n=0~7 when m=1)

6.7 Control of timer input (TImn)

6.7.1 Structure of TImn pin input circuit

The signal of the timer input pin is input to the timer control circuit through the noise filter and the edge detection circuit. For pins that need to be eliminated from noise, the corresponding pin noise filter must be set to be valid. The structure of the input circuit is as follows.

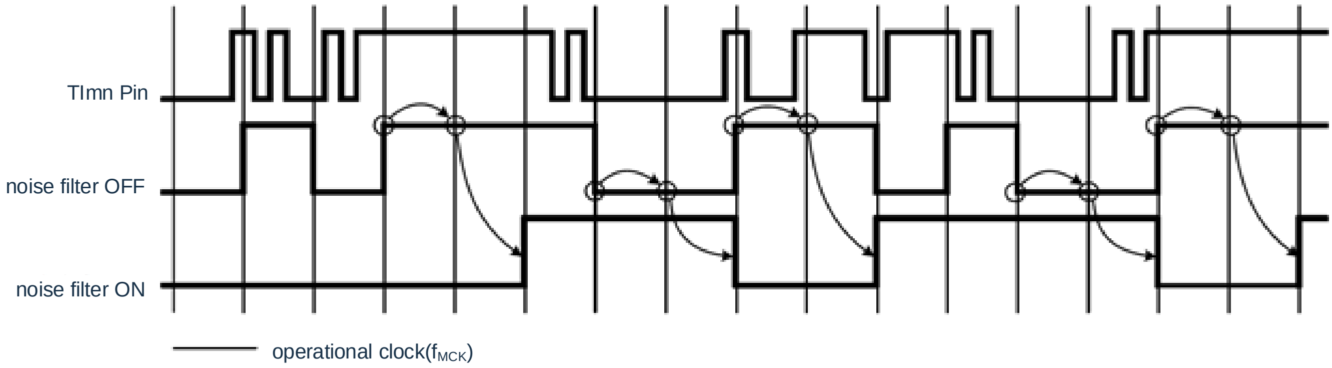
Figure 6-40(1) Structure of input circuit



6.7.2 Noise filter

When the noise filter is inactive, synchronization is performed only through the runtime clock (f_{MCK}) of channel n. When the noise filter is active, two clocks are detected after synchronization through the run-time clock (f_{MCK}) of channel n. The TM4mn input pin in that case of a noise filter ON or OFF, the waveform aft passing through the noise filter circuit is shown below.

Figure. 6-40 Sampling waveform of the TImn input pin when noise filter is ON or OFF



Note: The input waveform of the Tlmn pin is used to describe the operation of the noise filter ON or OFF. In actual use, the input must be made in accordance with the Tlmn input high and low level width shown in AC Characteristics.

6.7.3 Precautions when operating channel inputs

The noise filter circuit is not provided with a run-time clock when the timing input pin is not used. Therefore, the channel operation from the setting to use the timer input pin to the setting to input the timer corresponding to the pin allows triggering, requires the following waiting time.

(1) When the noise filter is OFF

If any of the timer mode register mn(TMRmn) bit12(CCSmn), bit9(STSmn1) and bit8(STSmn0) is set while all were '0', the operation enable of timer channel start register (TSm) must be triggered set after at least 2 execution clock (f_{MCK}) cycles.

(2) When the noise filter is ON

If any of the timer mode register mn(TMRmn) bit12(CCSmn), bit9(STSmn1) and bit8(STSmn0) is set while all were '0', the operation enable of timer channel start register (TSm) must be triggered set after at least 4 execution clock (f_{MCK}) cycles.

6.8 Independent channel operation function of universal timer unit

6.8.1 Operation as interval timer/square wave output

(1) Interval timer

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

Generation period for INTTMmn (timer interrupt) = count clock period (set value of TDRmn+1)

(2) Operation as square wave output

TOmn generates INTTMmn and outputs 50% duty cycle square wave at the same time of alternate output.

The period and frequency of the TOmn output square wave can be calculated by the following formula:

The Square Wave Period Output by TOmn = Count Clock Period (TDRmn Set Value +1)×2

The Square Wave Frequency from TOmn = Count Clock Frequency / {(TDRmn's Set Value +1) ×2}

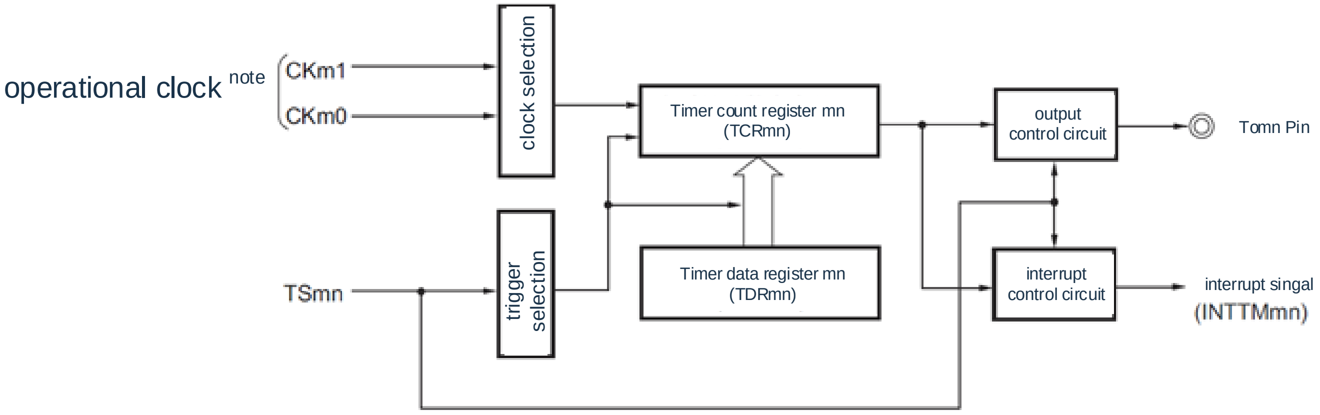
In interval timer mode, that timer count register mn (TCRmn) is use as the decrement counter.

After setting the channel start trigger bits (TSmn, TSHm1, TSHm3) of the timer channel start register m (TSm) to "1", the value of the timer data register mn (TDRmn) is loaded into the TCRmn register through one counting clock. At this time, if the MDmn0 bit of timer mode register n(TMRmn) is '0', the INTTMmn is not output and the TOmn is not output alternately. If the MDmn0 bit of the TMRmn register is "1", INTTMmn is output and TOmn is output alternately. The TCRmn register then decrements by a counting clock.

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

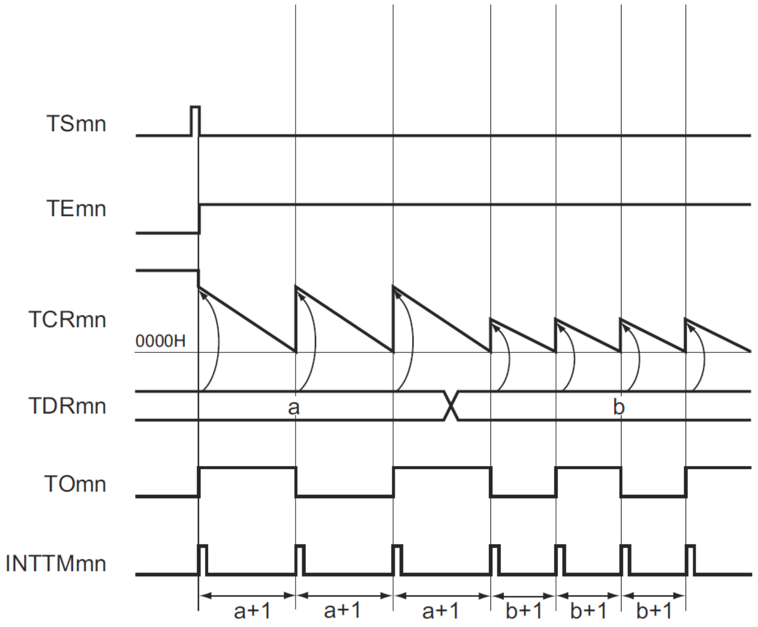
The TDRmn register can be overridden at any time, and the value of the overridden TDRmn register is valid from the next cycle.

Figure 6-41 Basic timing example as interval timer/square wave output operation (MDmn0=1)


operational clock note
CKm1
CKm0
clock selection
TSmn
trigger selection
Timer count register mn (TCRmn)
Timer data register mn (TDRmn)
output control circuit
interrupt control circuit
Tomn Pin
interrupt singal (INTTMmn)

Note: Clocks can be selected from CKm0, CKm1, CKm2, and CKm3 at Channels 1 and 3.

Figure 6-42 Basic timing example as interval timer/square wave output operation (MDmn0=1)



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

2. TSmn : Bit n for timer channel start register m (TSm)

TEmn : The timer channel allows the bit n of the state register m (TEm)

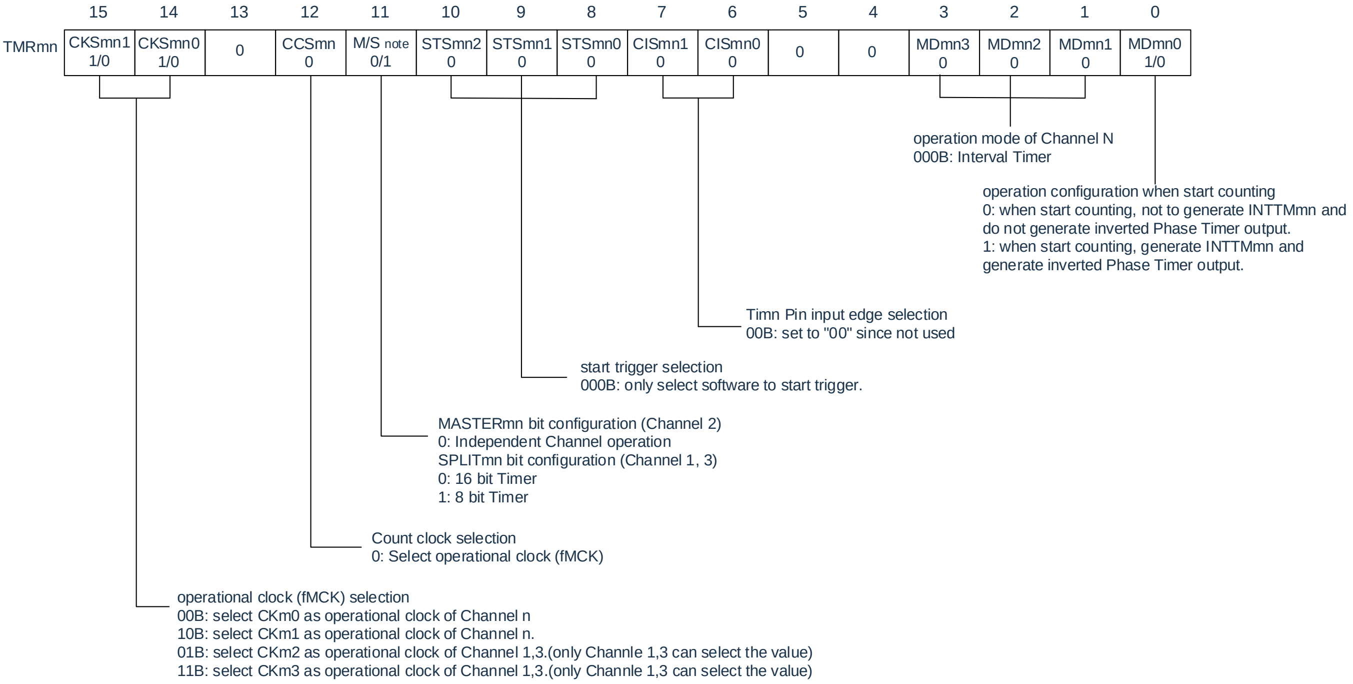
TCRmn: Timer count register mn (TCRmn)

TDRmn: Timer data register mn (TDRmn)

TOmn : TOmn pin output signal

Figure 6-43 Example of register setting content at interval timer/square wave output

(a) timer mode register mn (TMRmn)



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn	CKSmn1 1/0	CKSmn0 1/0	0	CCSmn 0	M/S note 0/1	STSmn2 0	STSmn1 0	STSmn0 0	CISmn1 0	CISmn0 0	0	0	MDmn3 0	MDmn2 0	MDmn1 0	MDmn0 1/0

operation mode of Channel N
000B: Interval Timer

operation configuration when start counting
0: when start counting, not to generate INTTMmn and do not generate inverted Phase Timer output.
1: when start counting, generate INTTMmn and generate inverted Phase Timer output.

Timn Pin input edge selection
00B: set to "00" since not used

start trigger selection
000B: only select software to start trigger.

MASTERmn bit configuration (Channel 2)
0: Independent Channel operation
SPLITmn bit configuration (Channel 1, 3)
0: 16 bit Timer
1: 8 bit Timer

Count clock selection
0: Select operational clock (fMCK)

operational clock (fMCK) selection
00B: select CKm0 as operational clock of Channel n
10B: select CKm1 as operational clock of Channel n.
01B: select CKm2 as operational clock of Channel 1,3.(only Channle 1,3 can select the value)
11B: select CKm3 as operational clock of Channel 1,3.(only Channle 1,3 can select the value)

(b) Timer output register m (TOm).

	bit
TOm	TOmn 1/0

0: Output "0" from TOmn.
1: Output "1" from TOmn.

(c) Timer output enable register m (TOEm).

	bit
TOEm	TOEmn 1/0

0: Stop the TOmn output from the count run.
1: Allow TOmn output from the count run.

(d) Timer output level register m (TOLm).

	bit
TOLm	TOLmn 0

0: "0" is set at TOMmn=0 (Master Channel Output Mode).

(e) Timer output mode register m (TOMm).

	bit
TOMm	TOMmn 0

0: Sets the master channel output mode.

Note: TMRm2 : MASTERmn bit

TMRm1, TMRm3 :SPLITmn bit

TMRm0: Fixed as "0".

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

Figure 6-44 Procedure for interval timer/square wave output function

	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)
	Set the TM4 mEN of the peripheral enable register 0 (PER0) to "1".	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 bit of the timer output mode register m (TOMm) is "0" (master channel output mode). Set TOLmn to "0". Set the TOmn bit to determine the initial level of the TOmn output. Set TOEmn to "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.
Start operation	(Set TOEmn bit to "1" only when using TOmn output and restarting) Set TSmn (TSHm1, TSHm3) to "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.
In operation	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 operation	Set TTmn (TTHm1, TTHm3) to "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 bit "0" and set the TOmn bit.	The TOmn pin outputs the level set by the TOmn bit.
TAU stop	To maintain the TOmn pin output level: Set TOmn to "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.
	Set the TM4 mEN of the PER0 register to "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)

Restart Operation

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

6.8.2 Operation as external event counter

It can be used as an event counter to count the valid edges (external events) of detected TImn pin inputs, and interrupt occurs if a specified count value is reached. The specified counter value can be calculated using the following formulas:

Specified count value=TDRmn's set value+1

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

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

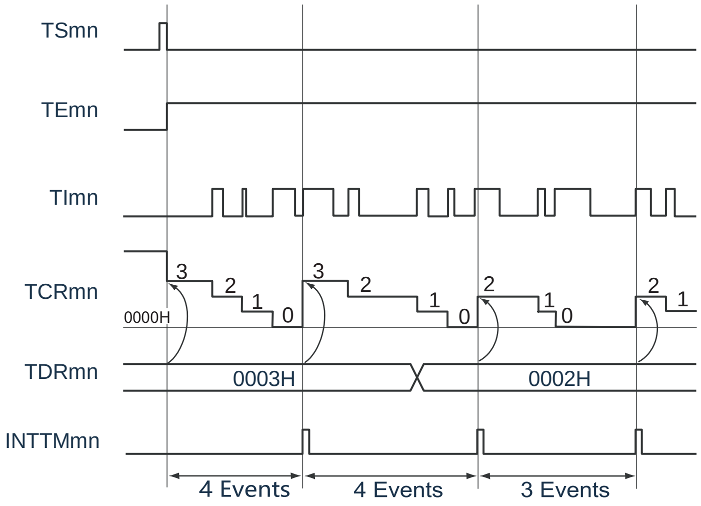
The TCRmn register decrements while detecting a valid edge of the TImn pin input. If the TCRmn becomes "0000H," the value of the TDRmn register is loaded again and the INTTMmn is output.

After that, the same operation continues.

Because the TOmn pin outputs an irregular waveform based on an external event, the output must be stopped by setting the TOEmn bit of the Timer Output Enable Register m (TOEm) to "0".

Can override the TDRmn register at any time, the value of the overridden TDRmn register is valid for the next count period.

Figure 6-45 Example of basic timing running as external event counter



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

2. TSmn : Bit n for timer channel start register m (TSm)

TEmn : Timer channel enable status register m (TEm) of bit n

TImn :TImn pin input signal

TCRmn: timer count register mn (TCRmn)

TDRmn: timer data register mn (TDRmn)

Figure 6-46 Example of register setting content in external event counter mode

(a) timer mode register mn (TMRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn	CKSmn1 1/0	CKSmn0 1/0	0	CCSmn 0	M/S note 0/1	STSmn2 0	STSmn1 0	STSmn0 0	CISmn1 0	CISmn0 0	0	0	MDmn3 0	MDmn2 0	MDmn1 0	MDmn0 1/0

operation mode of Channel N
000B: Interval Timer

operation configuration when start counting
0: when start counting, not to generate INTTMmn and do not generate inverted Phase Timer output.

Timn Pin input edge selection
00B: Detect falling edge
01B: Detect rising edge
10B: Detect both edges
11B: reserved

start trigger selection
000B: only select software to start trigger.

MASTERmn bit configuration (Channel 2)
0: Independent Channel operation
SPLITmn bit configuration (Channel 1, 3)
0: 16 bit Timer
1: 8 bit Timer

Count clock selection
0: Select operational clock (fMCK)

operational clock (fMCK) selection
00B: select CKm0 as operational clock of Channel n
10B: select CKm1 as operational clock of Channel n.
01B: select CKm2 as operational clock of Channel 1,3.(only Channle 1,3 can select the value)
11B: select CKm3 as operational clock of Channel 1,3.(only Channle 1,3 can select the value)

(b) Timer output register m (TOm).

	bit
TOm	TOmn 1/0

0: Output "0" from TOmn.

(c) Timer output enable register m (TOEm).

	bit
TOEm	TOEmn 1/0

0: Stop the TOmn output from the count run.
1: Allow TOmn output by count run.

(d) Timer output level register m (TOLm).

	bit
TOLm	TOLmn 0

0: "0" at TOMmn=0 (Master Channel Output Mode).

(e) Timer output mode register m (TOMm)m.

	bit
TOMm	TOMmn 0

0: Sets the master channel output mode.

Note: TMRm2, TMRm4, TMRm6: MASTERmn bit
TMRm1, TMRm3:SPLITmn bit
TMRm0, TMRm5, TMRm7: fixed as "0".

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

Figure 6-47 Procedure for external event counter function

	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)
	Set the TM4 mEN bit of the peripheral enable register 0 (PER0) to "1".	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. Set the TOEmn bit of the timer output enable register m (TOEm) to "0".	The channel is in an operational stop state. (Provide clocks, consume a portion of the Power)
Start operation	Set TSmn bit to "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.
In operation	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 operation	Set TTmn bit to "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	Set the TM4 mEN of the PER0 register to "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 frequency divider

A frequency divider capable of dividing the clock input by the TImn pin and used as the output of the TOmn pin. The frequency division clock frequency of TOmn output can be calculated by the following formula:

· Select a rising or a falling edge:

Divided clock frequency = Input clock frequency / {(Setting value of TDRmn +1)×2}

· Select both edges:

Divided clock frequency ≈ Input clock frequency / (Setting value of TDRmn +1)

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

After setting the channel start trigger bit (TSmn) of timer channel start register (TSm) to "1", the value of timer data register (TDRmn) is loaded into TCRmn register by detecting an effective edge of TImn. If the MDmn0 bit of the timer mode register (TMRmn) is "0", no output of INTTMmn and TOmn is not alternately output; If the TMRmn register has a MDmn0 bit of "1", INTTMmn is output and TOmn is output alternately.

The TCRmn register then decrements through valid edges entered by the TImn pin. If TCRmn becomes "0000H," TOmn will alternate output. At the same time, the value of the TDRmn register is loaded into the TCRmn register and continues to count.

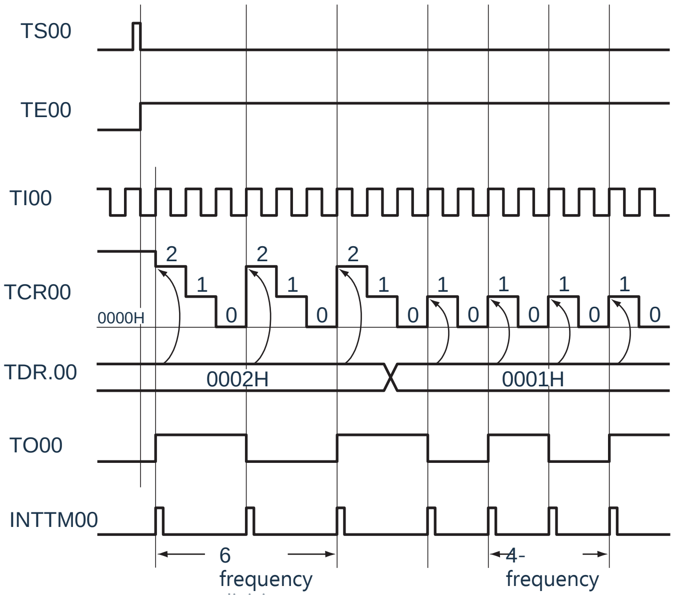
If the TImn pin is selected to input the double edge detection, the duty cycle error of the input clock will affect the frequency division clock cycle of the TOmn output.

The clock cycle output by TOmn contains a sampling error of one running clock cycle.

Clock cycle for TOmn output=Ideal TOmn output clock cycle ± Running clock cycle (error)

The TDRmn register can be overridden at any time, and the value of the overridden TDRmn register is valid during the next count.

Figure 6-48 Example of basic timing as divider (MDmn0=1)



Remark TSmn : Bit n of timer channel start register (TSm)

TEmn : Timer channel enable status register m (TEm) of bit n

TImn :TImn pin input signal

TCRmn: Timer count register (TCRmn).

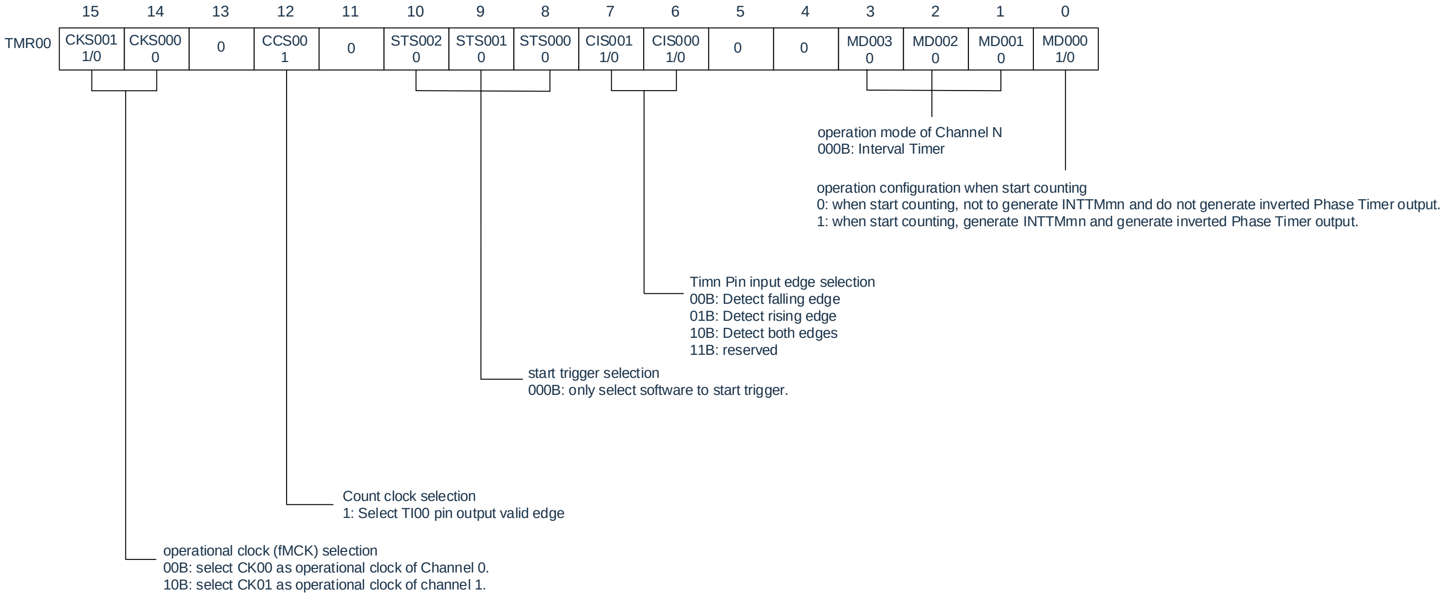
TDRmn: Timer data register (TDRmn).

TOmn : TOmn pin output signal

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

Figure 6-49 Example of register setting contents when operating as a divider (channel 0 of unit 0)

(a) Timer Mode Register 00 (TMR00)



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMR00	CKS001 1/0	CKS000 0	0	CCS00 1	0	STS002 0	STS001 0	STS000 0	CIS001 1/0	CIS000 1/0	0	0	MD003 0	MD002 0	MD001 0	MD000 1/0

operation mode of Channel N
000B: Interval Timer

operation configuration when start counting
0: when start counting, not to generate INTTMmn and do not generate inverted Phase Timer output.
1: when start counting, generate INTTMmn and generate inverted Phase Timer output.

Timn Pin input edge selection
00B: Detect falling edge
01B: Detect rising edge
10B: Detect both edges
11B: reserved

start trigger selection
000B: only select software to start trigger.

Count clock selection
1: Select TI00 pin output valid edge

operational clock (fMCK) selection
00B: select CK00 as operational clock of Channel 0.
10B: select CK01 as operational clock of channel 1.

(b) Timer output register 0 (TO0)

	bit 0
TO0	TO00 1/0

0: Output "0" by TO00.
1: Output "1" from TO00.

(c) Timer output enable register 0 (TOE0)

	bit 0
TOE0	TOE00 1/0

0: Stop the TO00 output from the count run.
1: Allow TO00 output by count runs.

(d) Timer output level register 0 (TOL0)

	bit 0
TOL0	TOL00 0

0: Place "0" in the master channel output mode (TOM00=0).

(e) Timer output mode register 0 (TOM0)

	bit
TOM0	TOM00 0

0: Sets the master channel output mode.

Figure 6-50　　Operating steps when frequency divider functions (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) valule 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' configure TO00 bit and confirm TO00 output initial voltage value. Set TOE00 bit to '1', aloow TO00 output. Set port register and port mode register to '0'.	TO00 pin in Hi-Z output state. When port mode register set to output mode and port register as '0', output TO00 initial configured voltage level. Because channel is in operation stopped state, thus TO00 remains unchange. 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 automtically 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 requried	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 measurement

It is possible to capture count value at that effective edge of TImn and measure the interval of the TImn input pulse. During the TEmn bit "1", the software operation (TSmn=1) can also be set to capture the trigger to capture count values.

The pulse interval can be calculated using the following equation:

TImn Input Pulse Interval = Period of the counting clock ((10000H x TSRmn:OVF) + (catch value of TDRmn+1))

Note: Since the TImn pin input is sampled by a running clock selected by the CKSmn bit of the timer mode register mn (TMRmn), an error occurs.

In capture mode, the timer count register mn (TCRmn) is used as the increment counter.

If the channel start trigger bit (TSmn) of the timer channel start register m (TSm) is set as '1', the TCRmn register starts counting from '0000H' by counting clock.

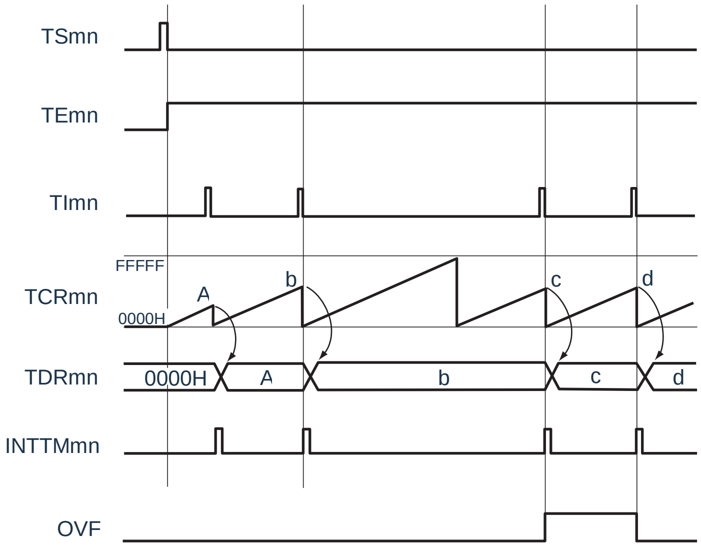
If the valid edge of the TImn pin input is detected, the count value of the TCRmn register is transferred (TDRmn) to the timer data register, TCRmn. At this time, if the counter overflows, set the OVF bit of the timer status register mn (TSRmn) to "1". If the counter does not overflow, clear the OVF bit. After that, the same operation continues.

When the count value is captured to the TDRmn register, the OVF bit of the TSRmn register is updated according to whether overflows occur during measurement.

Even if the counter performs a full count of 2 cycles or more, it is considered that an overrun occurs to set the OVF bit of the TSRmn register to “1”. However, the interval value cannot be measured normally through the OVF bit when an overflow occurs 2 or more times.

The STSmn2~STSmn0 bits of the TMRmn register set to "001B", and the effective edge of the TImn is used for starting trigger and capture trigger.

Figure 6-51 Example of basic timing operating as input pulse interval measurement (MDmn0=0)



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

2. TSmn : Bit n for timer channel start register m (TSm)

TEmn : Timer channel enable status register m (TEm) of bit n

TImn :TImn Pin Input Signal

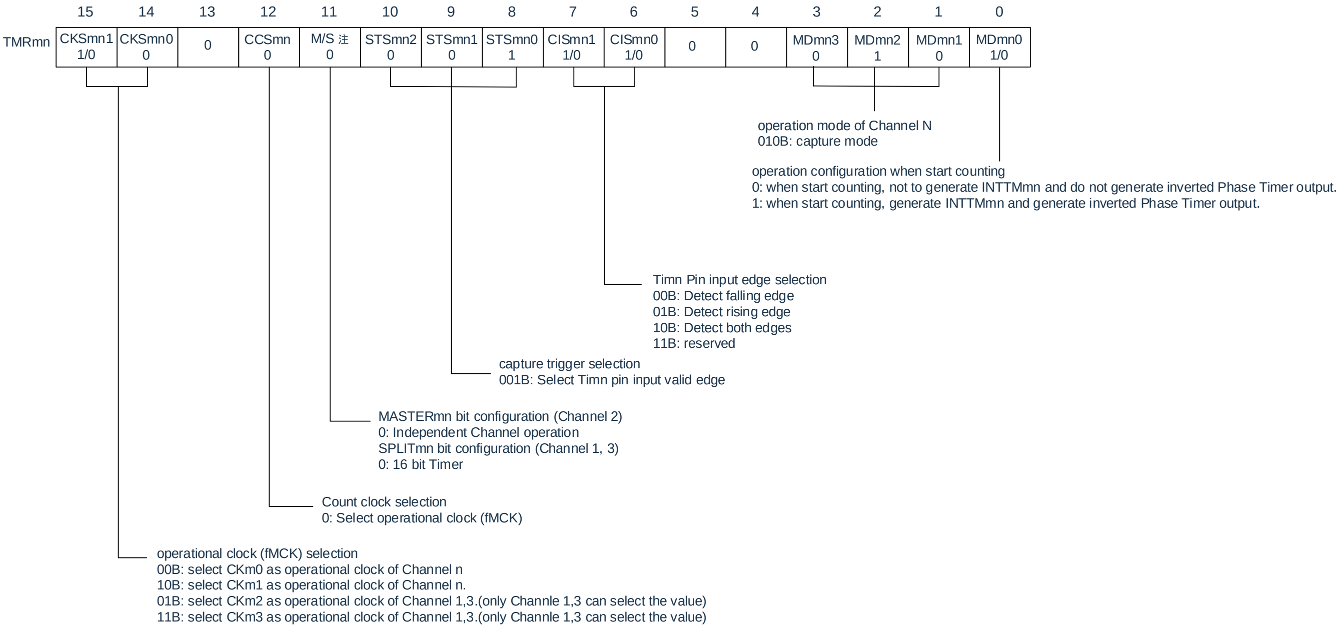
TCRmn : Timer count register mn (TCRmn)

TDRmn : Timer data register mn (TDRmn)

OVF : Bit 0 of timer state register mn (TSRmn).

Figure 6-52 Example of register contents setting in measuring input pulse interval

(a) Timer mode register mn (TMRmn)



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn	CKSmn1 1/0	CKSmn0 0	0	CCSmn 0	M/S 注 0	STSmn2 0	STSmn1 0	STSmn0 1	CISmn1 1/0	CISmn0 1/0	0	0	MDmn3 0	MDmn2 1	MDmn1 0	MDmn0 1/0

operation mode of Channel N
010B: capture mode

operation configuration when start counting
0: when start counting, not to generate INTTMmn and do not generate inverted Phase Timer output.
1: when start counting, generate INTTMmn and generate inverted Phase Timer output.

Timn Pin input edge selection
00B: Detect falling edge
01B: Detect rising edge
10B: Detect both edges
11B: reserved

capture trigger selection
001B: Select Timn pin input valid edge

MASTERmn bit configuration (Channel 2)
0: Independent Channel operation
SPLITmn bit configuration (Channel 1, 3)
0: 16 bit Timer

Count clock selection
0: Select operational clock (fMCK)

operational clock (fMCK) selection
00B: select CKm0 as operational clock of Channel n
10B: select CKm1 as operational clock of Channel n.
01B: select CKm2 as operational clock of Channel 1,3.(only Channle 1,3 can select the value)
11B: select CKm3 as operational clock of Channel 1,3.(only Channle 1,3 can select the value)

(b) Timer output register m (TOm)

	bit
TOm	TOmn 0

0: Output "0" from TOmn.

(c) Timer output enable register m (TOEm)

	bit
TOEm	TOEmn 0

0: Stop the TOmn output from the count run.

(d) Timer output level register m (TOLm)

	bit
TOLm	TOLmn 0

0: Place "0" in the main channel output mode (TOMmn=0).

(e) Timer output mode register m (TOMm)

	bit
TOMm	TOMmn 0

0: Sets the master channel output mode.

Note: TMRm2, TMRm4, TMRm6 : MASTERmn bit

TMRm1, TMRm3:SPLITmn bit

TMRm0, TMRm5, TMRm7: fixed as "0".

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

Figure 6-53 Procedure for the input pulse interval measurement function

restart operation

	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 aanytime. 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 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. 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. 0VF 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 (n=0~3 when m=0, n=0~7 when m=1)

6.8.5 Operation as voltage high and low level width measurement of input signal

Note: When used as LIN-bus support, the bit1 (ISC1) of the input switch control register (ISC) must be set to "1", and use RxD0 instead of TImn.

The signal width (high and low level width) of the TImn can be measured by starting the count at one edge input by the TImn pin and capturing the count at another edge. The signal width of TImn can be calculated using the following formula.

Signal width of TImn = period of counting clocks × ((10000H×HTSRmn:OVF) + (capture value of TDRmn+1))

Note: Since the TImn pin input is sampled by a running clock selected by the CKSmn bit of the timer mode register mn (TMRmn), an error occurs.

In capture&single count mode, the timer count register mn (TCRmn) is used as the increment counter. If the channel start trigger bit (TSmn) of the timer channel start register m (TSm) is set as "1", and the start edge of the TImn pin is detected waiting state.

If the start edge of the TImn pin input is detected (the rising edge of the TImn pin input when measuring the high level width), count incrementally starts from 0000H. Then, if an effective capture edge is detected (the falling edge of the TImn pin input when measuring the high level width), INTTMmn is output simultaneously. At this time, if the counter overflows, the OVF bit of the timer state register mn (TSRmn) is set. If the counter does not overflow, clear the OVF bit. The TCRmn register stops counting by changing its value to 'Pass to TDRmn register value +1' and enters the TImn pin start edge detection wait state. After that, the same operation continues.

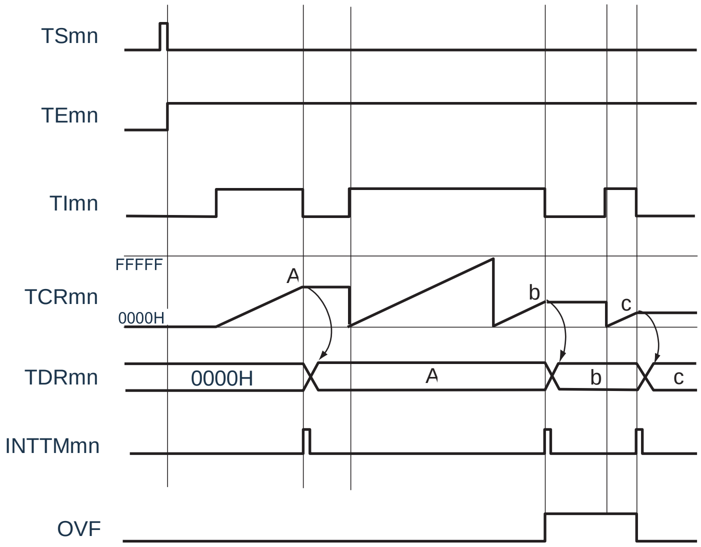
When the count value is captured to the TDRmn register, the OVF bit of the TSRmn register is updated according to whether overflows occur during measurement.

Even if the counter performs a full count of 2 cycles or more, it is considered that an overrun occurs to set the OVF bit of the TSRmn register to “1”. However, the interval value cannot be measured normally through the OVF bit when an overflow occurs 2 or more times.

It is possible to set whether a high level width or a low level width of a TImn pin is measured by a CISmn1 bit and a CISmn0 bit of a TMRmn register. This feature is designed to measure the input signal width of the TImn pin, so the TSmn = "1" cannot be used during TEmn.

CISmn1, CISmn0=10B of TMRmn register: A low level width is measured.
CISmn1, CISmn0=11B of TMRmn register: A high level width is measured.

Figure 6-54 Example of basic timing operating as high and low level width measurement of input signal



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

2. TSmn : Timer channel start register m (TSm) bit n

TEmn: Timer channel enable status register m (TEm) of bit n

TImn: TImn Pin Input Signal

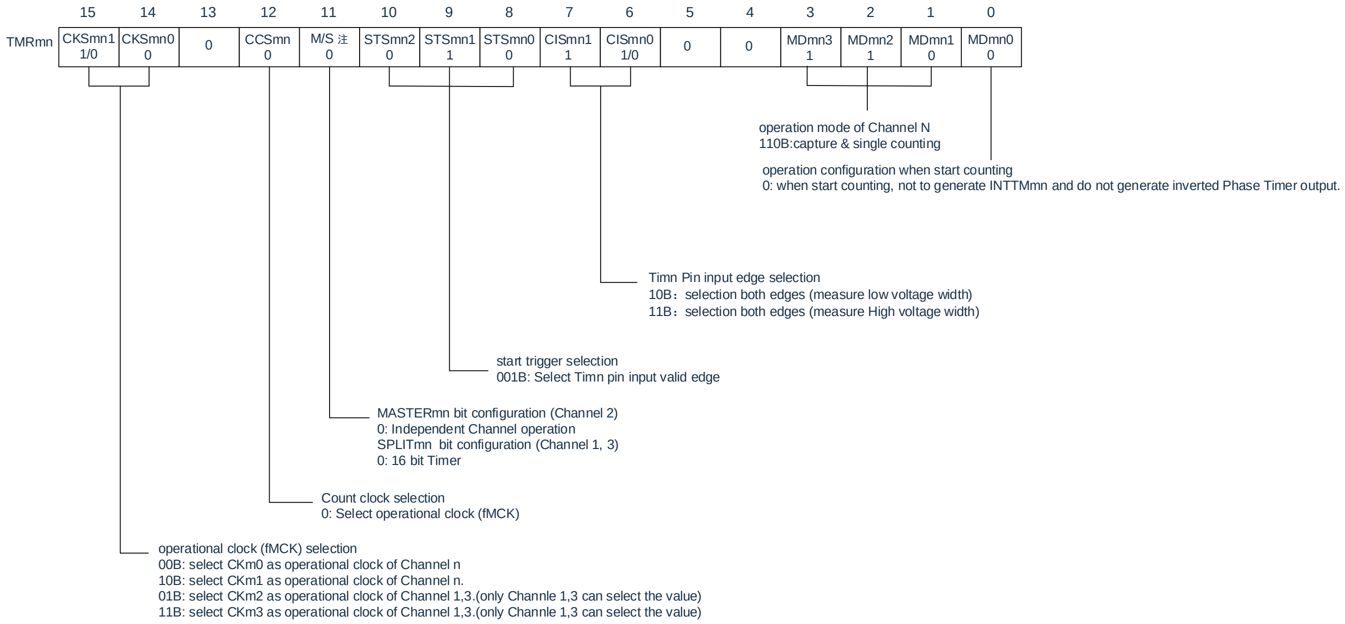
TCRmn: Timer count register mn (TCRmn)

TDRmn: Timer data register mn (TDRmn)

OVF: Bit0 for timer state register mn(TSRmn)

Figure 6-55 Example of register setting content when measuring high and low level width of input signal

(a) timer mode register mn (TMRmn)



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn	CKSmn1 1/0	CKSmn0 0	0	CCSmn 0	M/S 注 0	STSmn2 0	STSmn1 1	STSmn0 0	CISmn1 1	CISmn0 1/0	0	0	MDmn3 1	MDmn2 1	MDmn1 0	MDmn0 0

operation mode of Channel N
110B:capture & single counting

operation configuration when start counting
0: when start counting, not to generate INTTMmn and do not generate inverted Phase Timer output.

Timn Pin input edge selection
10B：selection both edges (measure low voltage width)
11B：selection both edges (measure High voltage width)

start trigger selection
001B: Select Timn pin input valid edge

MASTERmn bit configuration (Channel 2)
0: Independent Channel operation
SPLITmn bit configuration (Channel 1, 3)
0: 16 bit Timer

Count clock selection
0: Select operational clock (fMCK)

operational clock (fMCK) selection
00B: select CKm0 as operational clock of Channel n
10B: select CKm1 as operational clock of Channel n.
01B: select CKm2 as operational clock of Channel 1,3.(only Channle 1,3 can select the value)
11B: select CKm3 as operational clock of Channel 1,3.(only Channle 1,3 can select the value)

(b) Timer output register m (TOm)

	bit
TOm	TOmn 0

0: Output "0" from TOmn.

(c) timer output enable register m (TOEm)

	bit
TOEm	TOEmn 0

0: Stop the TOmn output from the count run.

(d) Timer output level register m (TOLm).

	bit
TOLm	TOLmn 0

0: Set "0" in the master channel output mode (TOMmn=0).

(e) Timer output mode register m (TOMm)m.

	bit
TOMm	TOMmn 0

0: Set master channel output mode.

Note: TMRm2, TMRm4, TMRm6 : MASTERmn bit

TMRm1, TMRm3:SPLITmn bit

TMRm0, TMRm5, TMRm7: fixed as "0".

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

Figure 6-56 Procedure for high and low level width measurement function of input signal

	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'. 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.
	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'. 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. 0VF 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 (n=0~3 when m=0, n=0~7 when m=1)

6.8.6 Operation as delay counter

Can start decrement counts with valid edge detection (external events) entered through the TImn pin, and generate INTTMmn at arbitrary set-up intervals

(Timer interrupt).

During the period when the TEmn bit is "1", the TSmn bit can be set to "1" by software to start decremental counting and generate INTTMmn (timer interrupt) at any set interval.

Interrupt generation cycles can be calculated using the following equation:

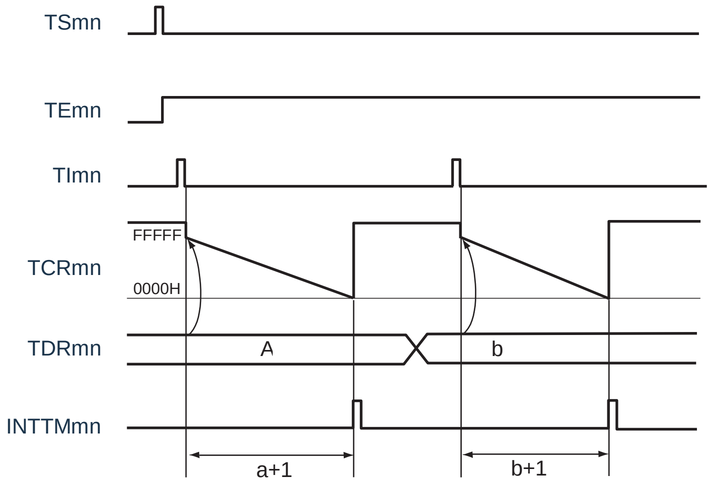
Generation period of INTTMmn (timer interrupt) = period of the counting clock×(set value of TDRmn+1)

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

if that channel start trigger bit (TSmn, TSHm1, TSHm3) of the timer channel start register m (TSm) is set as" 1",TEmn bit, TEHm1 bit, and TEHm3 bit become'1' and enter the valid edge detection wait state of TImn pin. The TImn register is started by valid edge detection of TCRmn pin input and the value of timer data register mn(TDRmn) is loaded. The TCRmn register counts down from the value of the mounted TDRmn register by counting the clock. If TCRmn becomes "0000H," INTTMmn is output and count is stopped before a valid edge of the next TImn pin input is detected.

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

Figure 6-57 Example of basic timing operating as delay counter



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

2. TSmn : Bit n for timer channel start register m (TSm)

TEmn : Timer channel enable status register m (TEm) of bit n

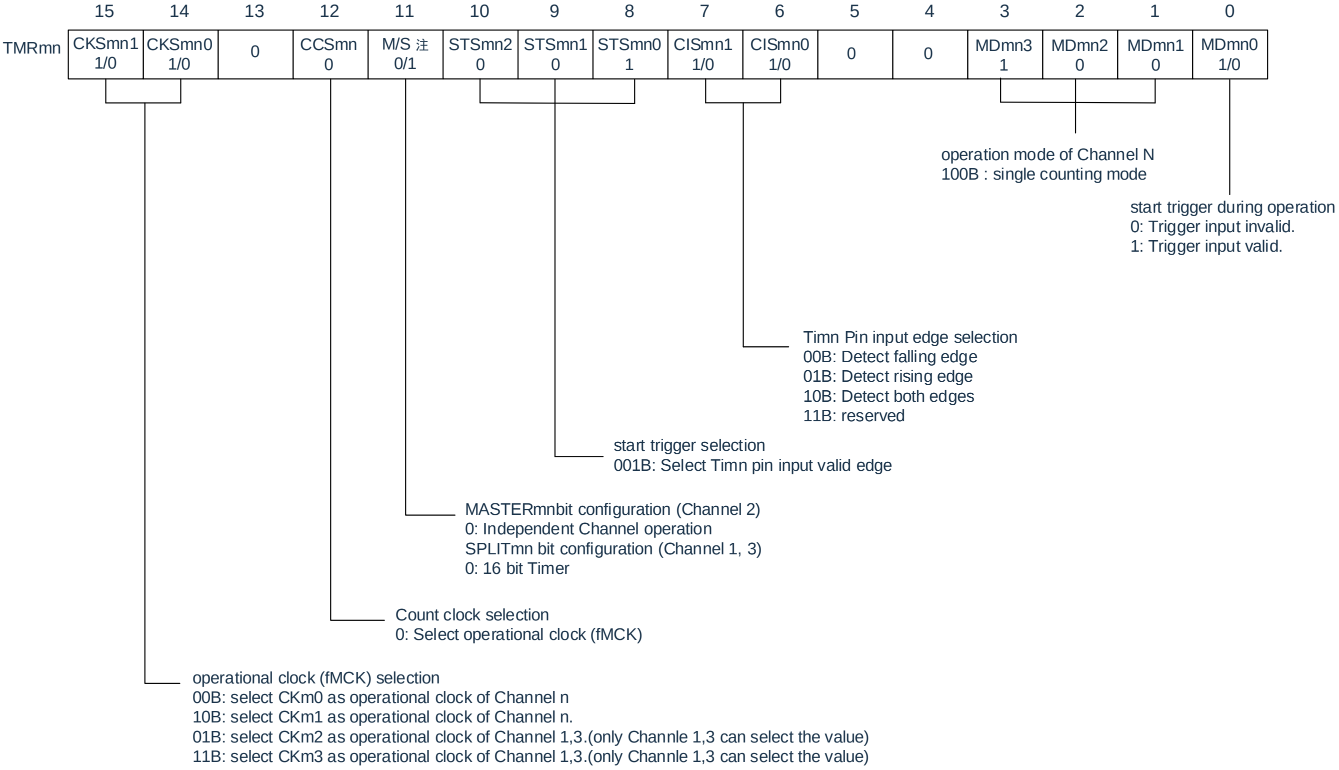
TImn :TImn pin input signal

TCRmn: Timer count register mn (TCRmn)

TDRmn: Timer data register mn (TDRmn)

Figure 6-58 Example of register contents setting for delay counter function

(a) Timer mode register mn (TMRmn)



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn	CKSmn1 1/0	CKSmn0 1/0	0	CCSmn 0	M/S 注 0/1	STSmn2 0	STSmn1 0	STSmn0 1	CISmn1 1/0	CISmn0 1/0	0	0	MDmn3 1	MDmn2 0	MDmn1 0	MDmn0 1/0

operation mode of Channel N
100B : single counting mode

start trigger during operation
0: Trigger input invalid.
1: Trigger input valid.

Timn Pin input edge selection
00B: Detect falling edge
01B: Detect rising edge
10B: Detect both edges
11B: reserved

start trigger selection
001B: Select Timn pin input valid edge

MASTERmnbit configuration (Channel 2)
0: Independent Channel operation
SPLITmn bit configuration (Channel 1, 3)
0: 16 bit Timer

Count clock selection
0: Select operational clock (fMCK)

operational clock (fMCK) selection
00B: select CKm0 as operational clock of Channel n
10B: select CKm1 as operational clock of Channel n.
01B: select CKm2 as operational clock of Channel 1,3.(only Channle 1,3 can select the value)
11B: select CKm3 as operational clock of Channel 1,3.(only Channle 1,3 can select the value)

(b) Timer output register m (TOm)

	bit
TOm	TOmn 0

0: Output "0" from TOmn.

(c) timer output enable register m (TOEm)

	bit
TOEm	TOEmn 0

0: Stop the TOmn output from the count run.

(d) Timer output level register m (TOLm)

	bit
TOLm	TOLmn 0

0: Place "0" in the main channel output mode (TOMmn=0).

(e) Timer output mode register m (TOMm)

	bit
TOMm	TOMmn 0

0: Sets the master channel output mode.

Note: TMRm2, TMRm4, TMRm6:MASTERmn bits
TMRm1, TMRm3:SPLITmn bit
TMRm0, TMRm5, TMRm7: fixed as "0".

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

Figure 6-59 Procedure for the delay counter function

	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 (TCRmn) 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 (n=0~3 when m=0, n=0~7 when m=1)

6.9 Multi-channel linkage operation function of universal timer unit

6.9.1 Operation as single trigger pulse output function

The two channels are used in pairs, and the single trigger pulse with arbitrary delay pulse width can be generated through the input of the TImn pin. The delay and pulse width can be calculated by the following formulas:

Delay={ Setting Value of TDRmn (Master) +2}×Count Clock Cycles
Pulse Width={ Setting Value of TDRmp (Slave) }×Count clock cycles

In a single count mode, that main control channel operate and count the delays. By detecting the start trigger, the timer count register mn (TCRmn) of the main control channel starts running and loads the value of timer data register mn (TDRmn). The TCRmn register counts down from the value of the mounted TDRmn register by counting the clock. If TCRmn becomes '0000H', INTTMmn is output and the count stops before the next start trigger is detected.

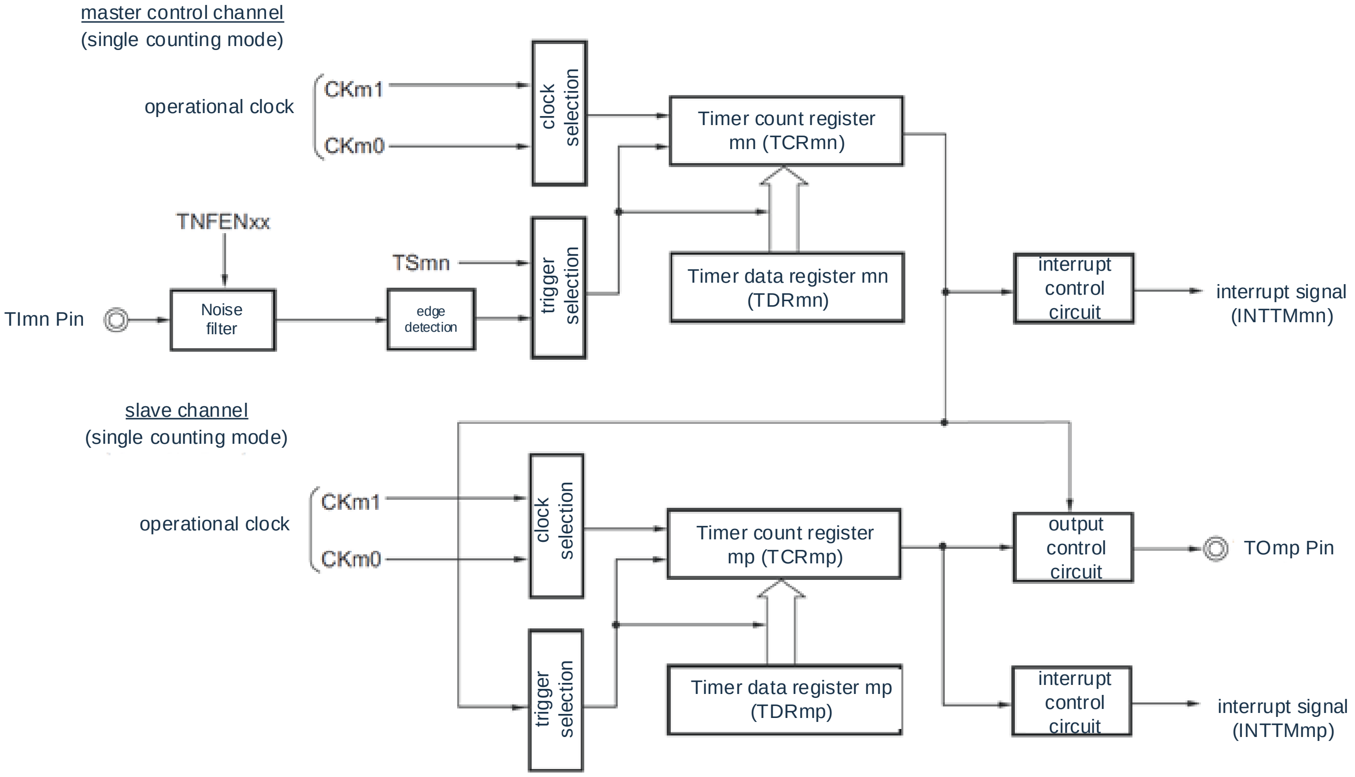
In a single count mode, the slave channel runs and counts the pulse width. The INTTMmn of the master channel is triggered as the start, the TCRmp register of the slave channel starts running and loads the value of the TDRmp register. The TCRmp register decrements the count from the value of the loaded TDRmp register by counting the clock. If the count value becomes "0000H," the INTTMmp is output and the count is stopped before the next start trigger (the INTTMmn of the primary channel) is detected. After the INTTMmn is generated from the main control channel and 1 counting clock passes, the output level of TOmp becomes effective level, and if TCRmp becomes "0000H".

The software operation (TSmn=1) can be output as a single trigger pulse as a start trigger without using TImn pin input.

Note: Because the loading timing of TDRmn register of master channel and TDRmp register of slave channel are different, if TDRmn register and TDRmp register are rewritten during counting, abnormal waveform may be output. The TDRmn register must be overridden after the generation of INTTMmn and the TDRmp register must be overridden after the generation of INTTMmp.

Note: m: Unit number (m=0, 1); n: Channel number (n=0, 2, 4, 6)
p: Slave channel number (n＜p≤3 when m=0, n＜p≤7 when m=1)

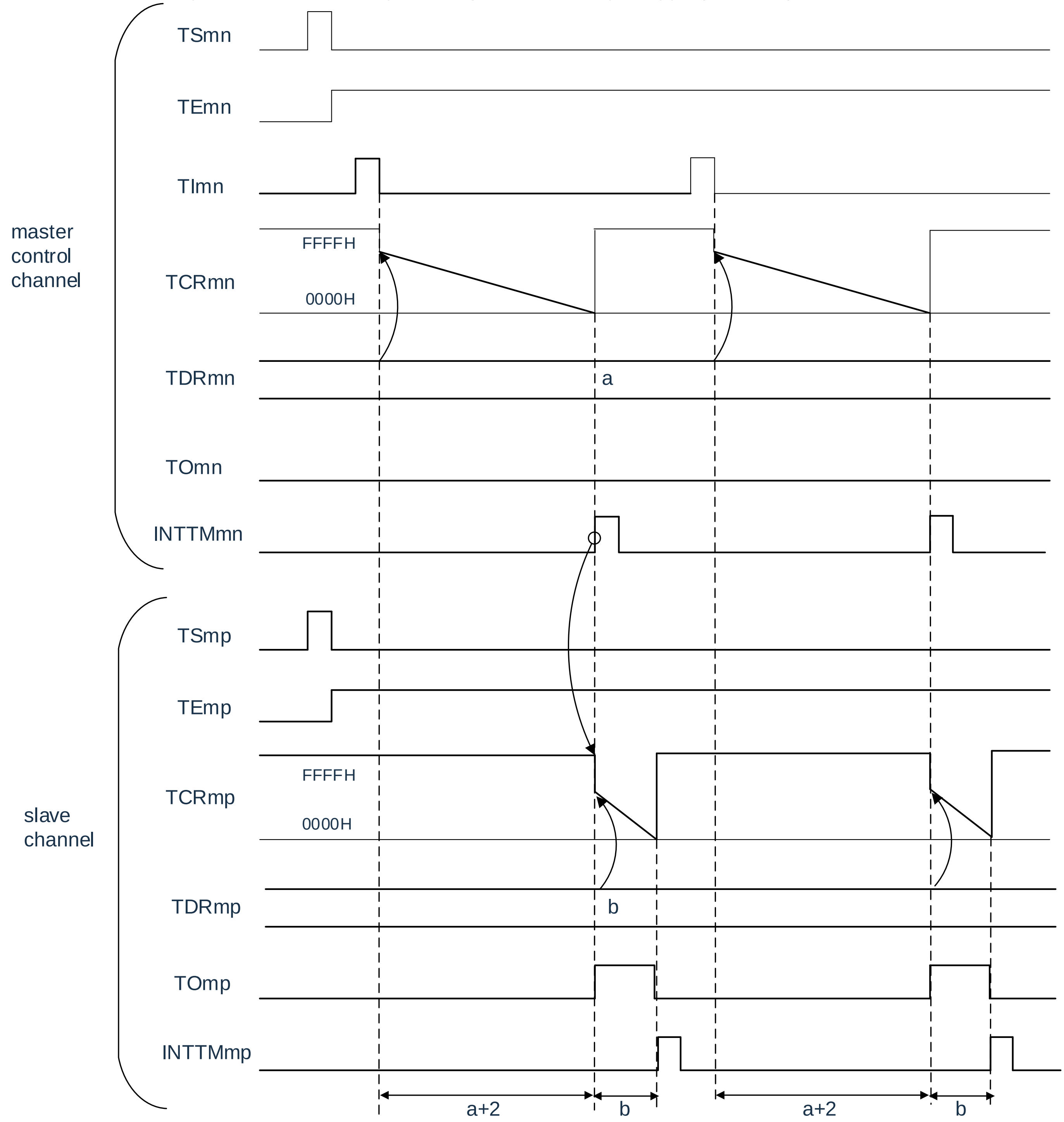
Figure 6-60 Block diagram for operating as single trigger pulse output function



Note: m: Unit number (m=0, 1); n: Channel number (n=0, 2, 4, 6)

p: Slave channel number (n＜p≦3 when m=0, n＜p≦7 when m=1)

Figure 6-61 Block diagram of operation as single trigger pulse output function



Remarks: 1. m: Unit number (m=0, 1); n: Channel number (n=0, 2, 4, 6)

p: Slave channel number (n<p≤3 when m=0, n<p≤7 when m=1)

2. TSmn, TSmp : bit n, p of the timer channel start register m (TSm).

TEmn, TEmp : timer channel allows bit n, p of status register m (TEm)

TImn, TImp : Input signal for TImn and TImp pins

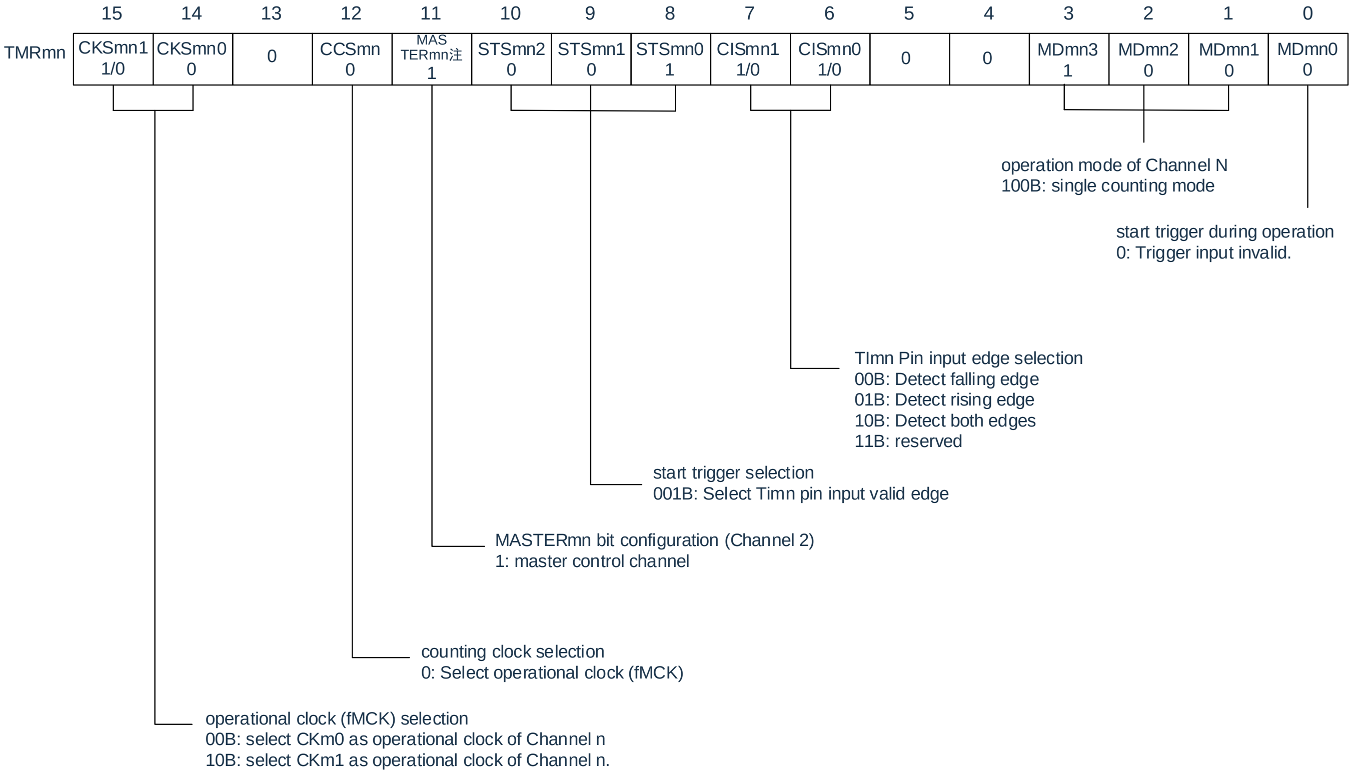
TCRmn, TCRmp : timer count registers mn, mp (TCRmn, TCRmp)

TDRmn, TDRmp : timer data register mn, mp (TDRmn, TDRmp)

TOmn, TOmp : output signal for TOmn and TOmp pins

Figure 6-62 Example of basic timing operating as a single trigger pulse output function (master channel)

(a) Timer mode register mn (TMRmn)



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn	CKSmn1 1/0	CKSmn0 0	0	CCSmn 0	MAS TERmn注 1	STSmn2 0	STSmn1 0	STSmn0 1	CISmn1 1/0	CISmn0 1/0	0	0	MDmn3 1	MDmn2 0	MDmn1 0	MDmn0 0

operation mode of Channel N
100B: single counting mode

start trigger during operation
0: Trigger input invalid.

TImn Pin input edge selection
00B: Detect falling edge
01B: Detect rising edge
10B: Detect both edges
11B: reserved

start trigger selection
001B: Select Timn pin input valid edge

MASTERmn bit configuration (Channel 2)
1: master control channel

counting clock selection
0: Select operational clock (fMCK)

operational clock (fMCK) selection
00B: select CKm0 as operational clock of Channel n
10B: select CKm1 as operational clock of Channel n.

(b) Timer output register m (TOm)

	bit	
TOm	TOmn 0	0: Output "0" from TOmn.

(c) timer output enable register m (TOEm)

	bit	
TOEm	TOEmn 0	0: Stop the TOmn output from the count run.

(d) Timer output level register m (TOLm)

	bit	
TOLm	TOLmn 0	0: "0" at TOMmn=0 (Master Channel Output Mode).

(e) Timer output mode register m (TOMm)

	bit	
TOMm	TOMmn 0	0: Set master channel output mode.

Note: TMRm2 :MASTERmn=1
TMRm0 : Fixed as "0".

Note: m: Unit number (m=0, 1); n: Channel number (n=0, 2, 4, 6)

Figure 6-63 Example of register contents setting for single trigger pulse output function (slave channel)

(a) timer mode register mp (TMRmp)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmp	CKSmp1 1/0	CKSmp0 0	0	CCSmp 0	M/S 注 0	STSmp2 1	STSmp1 0	STSmp0 0	CISmp1 0	CISmp0 0	0	0	MDmp3 1	MDmp2 0	MDmp1 0	MDmp0 0

operation mode of Channel P
100B: single counting mode

start trigger during operation
0: Trigger input invalid.

TImp Pin input edge selection
00B: set to "00" since not used

start trigger selection
100B: Select master control channel INTTMmn

MASTERmp bit configuration (Channel 2)
0: slave channel
SPLITmp bit configuration (Channel 1, 3)
0: 16 bit Timer

counting clock selection
0: Select operational clock (fMCK)

operational clock (fMCK) selection
00B: select CKm0 as operational clock of Channel p
10B: select CKm1 as operational clock of Channel p
※ same as master control channel configuration

(b) Timer output register m (TOm)

	bit p
TOm	TOmp 1/0

0: Output "0" from TOmp.
1: Output "1" by TOmp.

(c) timer output enable register m (TOEm)

	bit p
TOEm	TOEmp 1/0

0: Stop the TOmp output from the count run.
1: Allow TOmp output by count run.

(d) Timer output level register m (TOLm)

	bit p
TOLm	TOLmp 1/0

0: Positive Logical Output (High Level Valid)
1: Negative Logical Output (Low Level Valid)

(e) Timer output mode register m (TOMm)m.

	bit p
TOMm	TOMmp 1

1: Set slave channel output mode.

Note: TMRm2, TMRm4, TMRm6:MASTERmp bit
TMRm1, TMRm3: SPLITmp bit

Note: m: Unit number (m=0, 1); n: Channel number (n=0, 2, 4, 6)
p: Slave channel number (n＜p≦3 when m=0, n＜p≦7 when m=1)

Figure 6-64 Operating step for a 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.

Figure 6-65 Operating step for a single trigger pulse output function (2/2)

restart operation

Start operation	set TOEmp bit (slave) to '1' (only limit to restart operation). Set TSmn bit)(master control) and TSmp bit(slave) of timer channel start register m(TSm) both to '1'. Because TSmnn bit and TSmp bit are trigger bits, thus automatically return to '0'.	TEmn bit and Temp bit turn into '1' and master channel enter into start trigger (detect Timn pin input valid edge or set TSmn bit to '1') detection waiting state.Counter still in stop state.
	start master channel counting while detecting master channel start trigger. • Detect TImn pin input valid edge • set TSmn bit of master channel to"1"via software. Note.	master channel start counting
in operation	can only modify configure value of CISmn1 bit and CISmn0 bit of TMRmn register. Forbidden modifying TMRmn, TMRmp register and TOMmn bit, TOMmp bit, TOLmn bit and TOLmp bit configuration. Can read TCRmn register and TCRmp register anytime. Can not use TSRmn register and TSRmp register. can modify slave channel Tom regsiter and TOEm register configuration.	master channel load TDRmn register value into Timer technical register (TCRmn) via detecting start trigger (detecting Timn pin input valid edge or set TSmn bit of master channel to "1"), and perform decremental counting. If TCRmn counts till "0000H", then generating INTTMmn, and stop counting before next Timn pin input. Slave channel use INTTMmn of master channel as trigger, will load TDRmp register value into TCRmp regiter 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 votlage levle then stoop counting. Thereafter, the process repeats.
stop operation	set TTmn bit (master) and TTmp bit(slave) to '1'. Because TTmn bit and TTmp bit are trigger bits, thus automatically return to '0'.	TEmn bit and Temp bit turn into '0' and stop counting. TCRmn register and TCRmp register hold counted value and stop counting. T0mp output not initialized and remains unchanged.
	set TOEmp bit of slave channel to '0', and configure TOmp bit.	T0mp pin output T0mp configured voltage level.
timer 4 stop	Scenarios to maintain T0mp pin output voltage: set T0mp bit to '0' after set hold value to port register configuration. In case T0mp pin output voltage does not need to be held: no configuration requried	maintain T0mp pin 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. (TO00 bit turns into '0' and TO00 pin becomes port function)

Note: can not set TSmn bit of slave channel to '1'.

Remark: m: Unit number (m=0) n: Channel number (n=0, 2, 4, 6)

p: Slave channel number (n＜p≤3 when m=0, n＜p≤7 when m=1)

6.9.2 Operation as PWM function

The two channels are used in pairs, and the pulse with arbitrary period and duty cycle can be generated. The period and duty cycle of the output pulse can be calculated by the following formula:

Pulse period = { Setting value of TDRmn (Master) +1}×Count clock period
Duty Ratio [%]={Setting value of TDRmp (Slave)}/{Setting value of TDRmn (Master) +1}×100
0% output : Setting value of TDRmp (Slave)=0000H
100% output : Setting value of TDRmp (Slave) ≥{ Setting value of TDRmn (Master) +1}

Note The duty cycle exceeds 100% when the set value of TDRmp >{ Set value of TDRmn (Master) +1}, but 100% output.

The master channel is used as the interval timer mode. If the channel start trigger bit (TSmn) of the timer channel start register m (TSm) is "1", the interrupt (INTTMmn) is output, and the setting value of the timer data register mn (TDRmn) is loaded into the timer count register mn (TCRmn), and counting down by counting the clocks. When counting to "0000H", the value of the TDRmn register is loaded into the TCRmn register again after outputting INTTMmn, and the count is decremented. This operation is repeated after setting the channel stop trigger bit (TTmn) of the timer channel stop register m (TTm).

When used as a PWM function, the main control channel performs a decremental count, which is a PWM output (TOmp) period until 0000H. The slave channel is used as a single count mode. Starting with the INTTMmn of the main control channel, the value of the TDRmp register is loaded into the TCRmp register and is decremented until "0000H". When counted to "0000H," INTTMmp is output and the next trigger is waited (INTTMmn of the master channel).

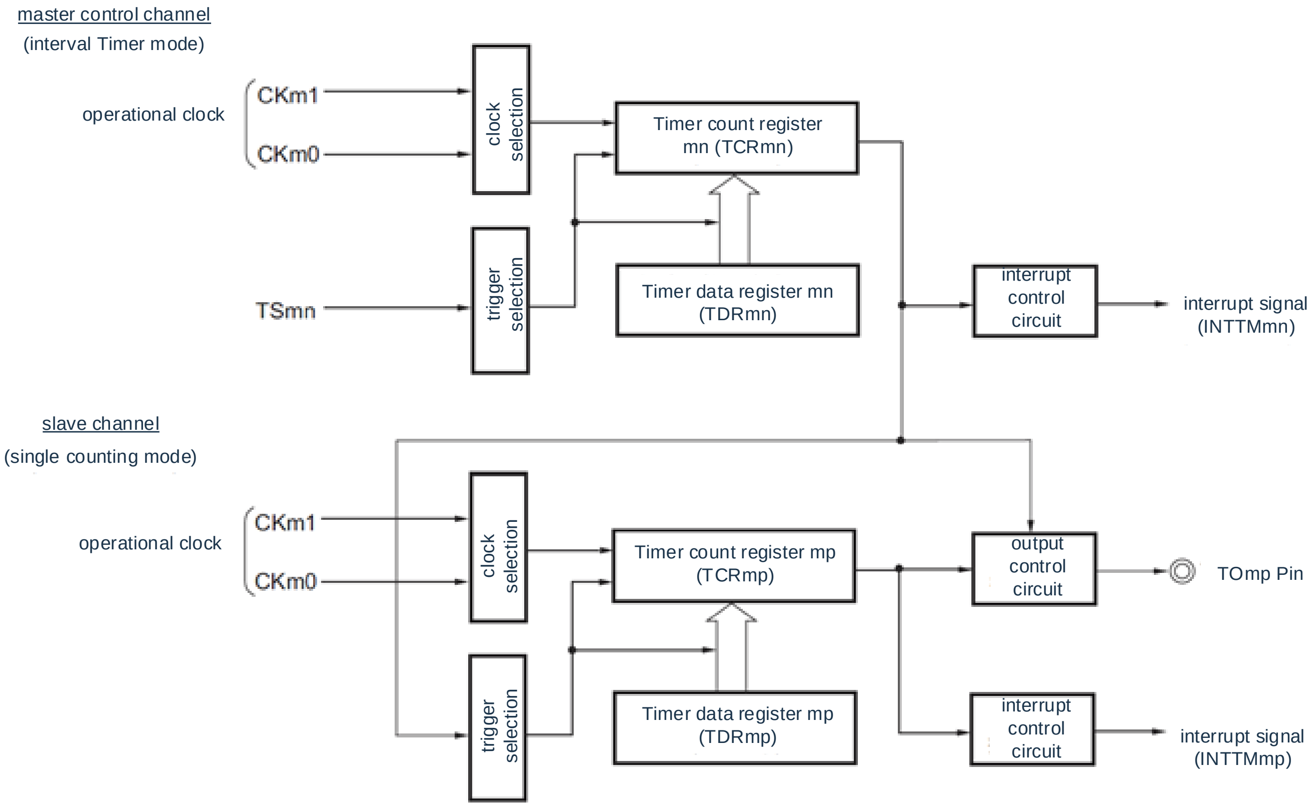
When used as a PWM function, the slave channel performs a decremental count, which is the duty cycle of the PWM output (TOmp) until '0000H'.

The PWM output (TOmp) becomes active after 1 clock generation of INTTMmn from the master channel and becomes invalid when the value of TCRmp register of slave channel is 0000H.

Note: When the timer data register mn (TDRmn) of the master channel and the TDRmp register of the slave channel are to be rewritten. Because the TDRmn register and the TDRmp register are loaded into the TCRmn register and the TCRmp register when the master channel generates the INTTMmn, the TOmp pin can not output the expected waveform if the INTTMmn is rewritten before and after the master channel generates the respectively. Therefore, to override both the master TDRmn register and the slave TDRmp register, you must override these 2 registers immediately after the master channel generates INTTMmn.

Note: m: Unit number (m=0, 1); n: Channel number (n=0, 2, 4, 6)
p: Slave channel number (n＜p≦3 when m=0, n＜p≦7 when m=1)

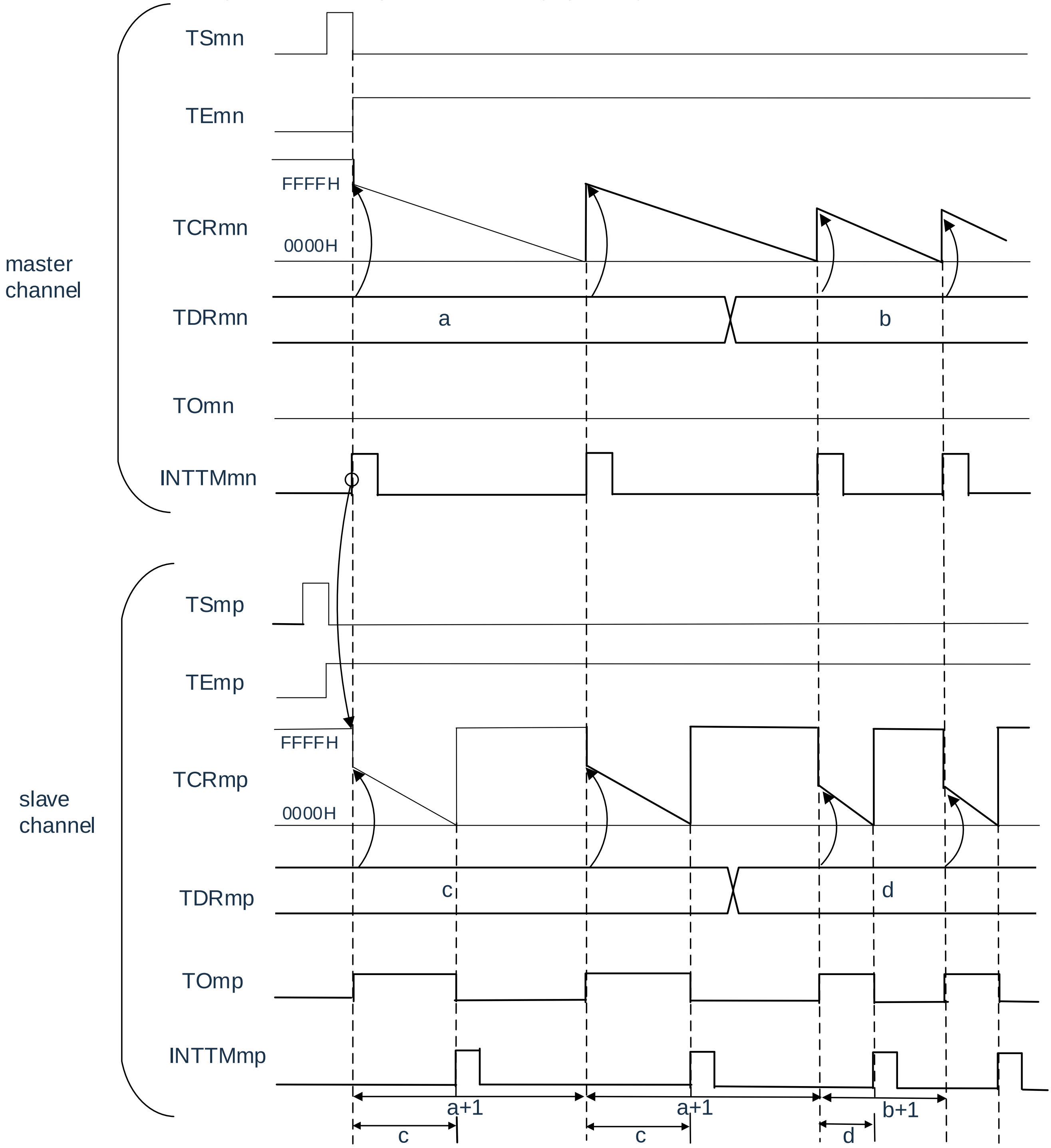
Figure 6-66 Block diagram for operation as PWM function



Note: m: Unit number (m=0, 1); n: Channel number (n=0, 2, 4, 6)

p: Slave channel number (n＜p≤3 when m=0, n＜p≤7 when m=1)

Figure 6-67 Example of basic timing operating as PWM function



Remarks: 1. m: Unit number (m=0, 1); n: Channel number (n=0, 2, 4, 6)

p: Slave channel number (n＜p≤3 when m=0, n＜p≤7 when m=1)

2. TSmn, TSmp: bitn, p of timer channel start register m (TSm).

TEmn, TEmp: bitn, p of timer channel enable status register m (TEm)

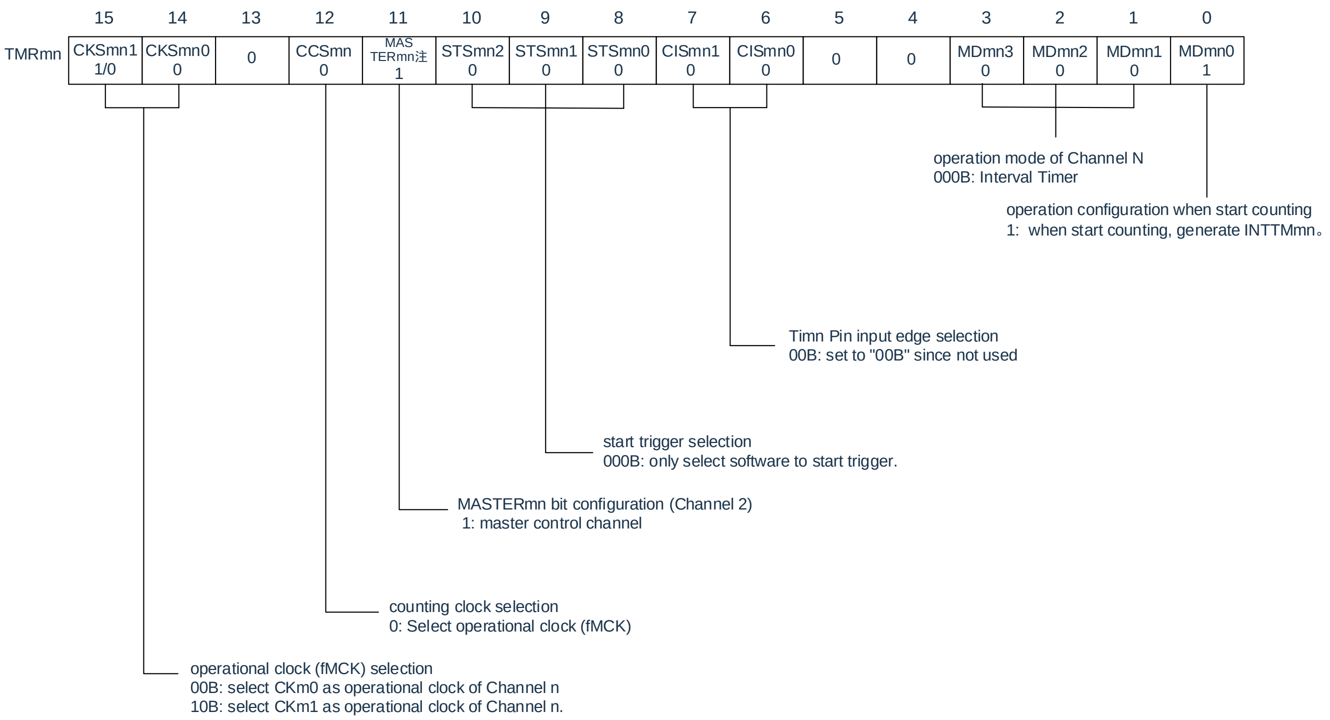
TCRmn, TCRmp: timer count registers mn, mp (TCRmn, TCRmp)

TDRmn, TDRmp: timer data register mn, mp (TDRmn, TDRmp)

TOmn, TOmp: output signal for TOmn Pin and TOmp Pin

Figure 6-68 Example of register contents setting for PWM function (master channel)

(a) Timer mode register mn (TMRmn)



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn	CKSmn1 1/0	CKSmn0 0	0	CCSmn 0	MAS TERmn注 1	STSmn2 0	STSmn1 0	STSmn0 0	CISmn1 0	CISmn0 0	0	0	MDmn3 0	MDmn2 0	MDmn1 0	MDmn0 1

operation mode of Channel N
000B: Interval Timer

operation configuration when start counting
1: when start counting, generate INTTMmn。

Timn Pin input edge selection
00B: set to "00B" since not used

start trigger selection
000B: only select software to start trigger.

MASTERmn bit configuration (Channel 2)
1: master control channel

counting clock selection
0: Select operational clock (fMCK)

operational clock (fMCK) selection
00B: select CKm0 as operational clock of Channel n
10B: select CKm1 as operational clock of Channel n.

(b) Timer output register m (TOm)

	bit
TOm	TOmn 0

0: Output "0" from TOmn.

(c) timer output enable register m (TOEm)

	bit
TOEm	TOEmn 0

0: Stop the TOmn output from the count run.

(d) Timer output level register m (TOLm)

	bit
TOLm	TOLmn 0

0: "0" is set at TOMmn=0 (Master Channel Output Mode).

(e) Timer output mode register m (TOMm)

	bit
TOMm	TOMmn 0

0: Sets the master channel output mode.

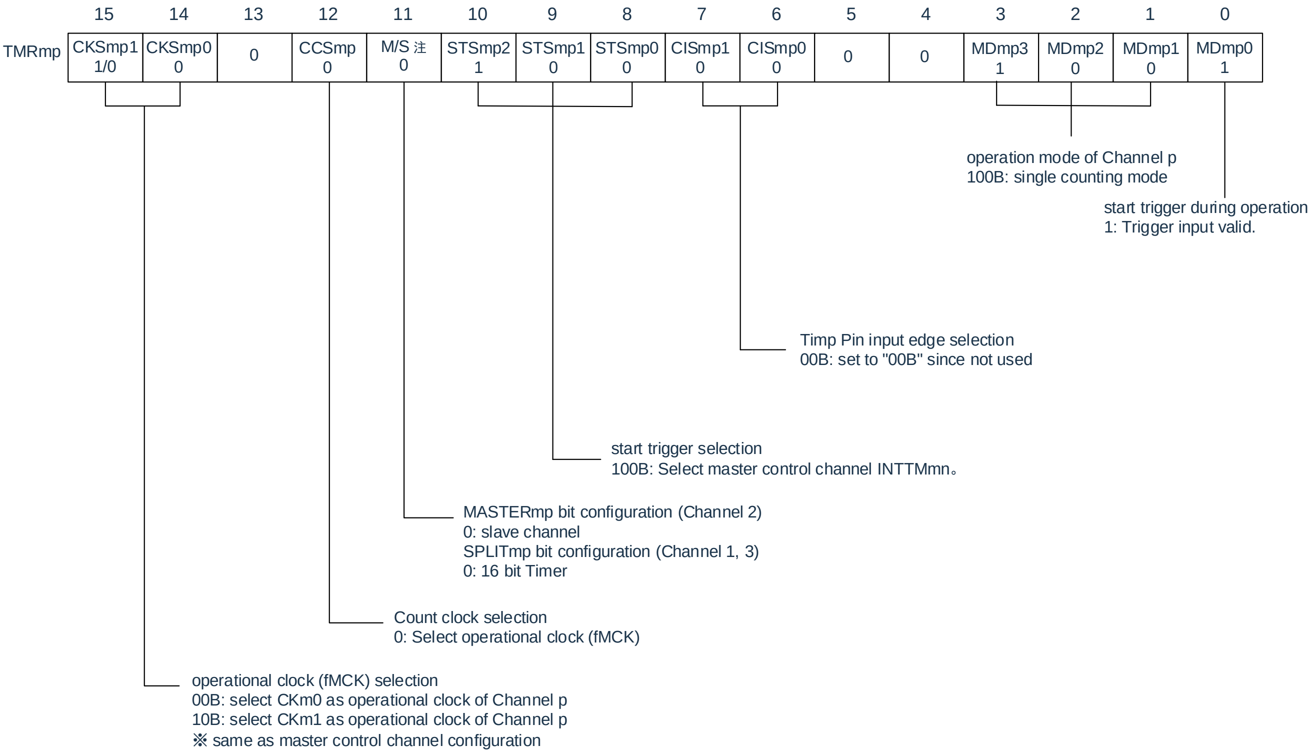
Note: TMRm2 :MASTERmn=1

TMRm0 : Fixed as "0".

Note: m: Unit number (m=0, 1); n: Channel number (n=0, 2, 4, 6)

Figure 6-69 Example of register contents setting for PWM function (slave channel)

(a) Timer mode register mp (TMRmp)



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmp	CKSmp1 1/0	CKSmp0 0	0	CCSmp 0	M/S 注 0	STSmp2 1	STSmp1 0	STSmp0 0	CISmp1 0	CISmp0 0	0	0	MDmp3 1	MDmp2 0	MDmp1 0	MDmp0 1

operation mode of Channel p
100B: single counting mode

start trigger during operation
1: Trigger input valid.

Timp Pin input edge selection
00B: set to "00B" since not used

start trigger selection
100B: Select master control channel INTTMmn。

MASTERmp bit configuration (Channel 2)
0: slave channel
SPLITmp bit configuration (Channel 1, 3)
0: 16 bit Timer

Count clock selection
0: Select operational clock (fMCK)

operational clock (fMCK) selection
00B: select CKm0 as operational clock of Channel p
10B: select CKm1 as operational clock of Channel p
※ same as master control channel configuration

(b) Timer output register m (TOm)

	bit p
TOm	TOmp 1/0

0: Output "0" from TOmp.
1: Output "1" by TOmp.

(c) Timer output enable register m (TOEm)

	bit p
TOEm	TOEmp 1/0

0: Stop the TOmp output from the count run.
1: Allow TOmp output by count run.

(d) Timer output level register m (TOLm)

	bit p
TOLm	TOLmp 1/0

0: Positive Logical Output (High Level Valid)
1: Negative Logical Output (Low Level Valid)

(e) Timer output mode register m (TOMm)

	bit p
TOMm	TOMmp 1

1: Sets the slave channel output mode.

Note: TMRm2, TMRm4, TMRm6:MASTERmp bit
TMRm1, TMRm3:SPLITmp bit

Note: m: Unit number (m=0, 1); n: Channel number (n=0, 2, 4, 6)
p: Slave channel number (n＜p≤3 when m=0, n＜p≤7 when m=1)

Figure 6-70 Procedure for for PWM functions (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 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 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.

Figure 6-71 Procedure for PWM functions (2/2)

restart operation

Start operation	set TOEmp bit (slave) to '1' (only limit to restart operation). Set TSmn bit)(master control) and TSmp bit(slave) of timer channel start register m(TSm) both to '1'. Because TSmnn bit and TSmp bit are 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 use INTTMmn of master channel as trigger, will load TDRmp register value into TCRmp regiter 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 votlage levle then stoop counting. Thereafter, the process repeats.
stop operation	set TTmn bit (master) and TTmp bit(slave) to '1'. Because TTmn bit and TTmp bit are trigger bits, thus automatically return to '0'.	TEmn bit and Temp bit turn into '0' and stop counting. TCRmn register and TCRmp register hold counted value and stop counting. T0mp output not initialized and remains unchanged.
	set TOEmp bit of slave channel to '0', and configure TOmp bit.	T0mp pin output T0mp configured voltage level.
timer 4 stop	Scenarios to maintain T0mp pin output voltage: set T0mp bit to '0' after set hold value to port register configuration. In case T0mp pin output voltage does not need to be held: no configuration requried	maintain T0mp pin 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 turns into '0' and T0mp pin becomes port function) (TO00 bit turns into '0' and TO00 pin becomes port function)

Remark m: Unit number (m=0, 1); n: Channel number (n=0, 2, 4, 6)

p: Slave channel number (n＜p≦3 when m=0, n＜p≦7 when m=1)

6.9.3 Operation as multiple PWM output function

This is a function of performing multiple PWM outputs with different duty cycles by extending the PWM function and using multiple slave channels.

For example, when two slave channels are used in pairs, the period and duty cycle of the output pulse can be calculated using the following equation:

Pulse period = { setting value of TDRmn (Master) +1×count clock period
Duty cycle 1[%]={setting value of TDRmp (Slave 1) }/{setting value of TDRmn (Master) +1}×100
Duty cycle 2[%]={setting value of TDRmq (Slave 2) }/{setting value of TDRmn (Master) +1}×100

Note: When {setting value of TDRmp (Slave 1)} > {setting value of TDRmn (Master) +1} or {setting value of TDRmq (Slave 2) } > {setting value of TDRmn (Master) +1}, the duty cycle exceeds 100% but is 100% output.

In interval timer mode, timer count register mn (TCRmn) of the main control channel runs and counts the cycles. In a single count mode, the TCRmp register of the slave channel 1 runs and counts the duty cycle and outputs a PWM waveform from the TOmp pin. Starting with the INTTMmn of the master channel, the timer data register mp (TDRmp) is loaded into the TCRmp register and decremented. If TCRmp becomes "0000H," the INTTMmp is output and counts are stopped before the next start trigger (INTTMmn of the master channel) is entered. After the INTTMmn is generated from the main control channel and 1 counting clock passes, the output level of TOmp becomes effective level, and if TCRmp becomes "0000H".

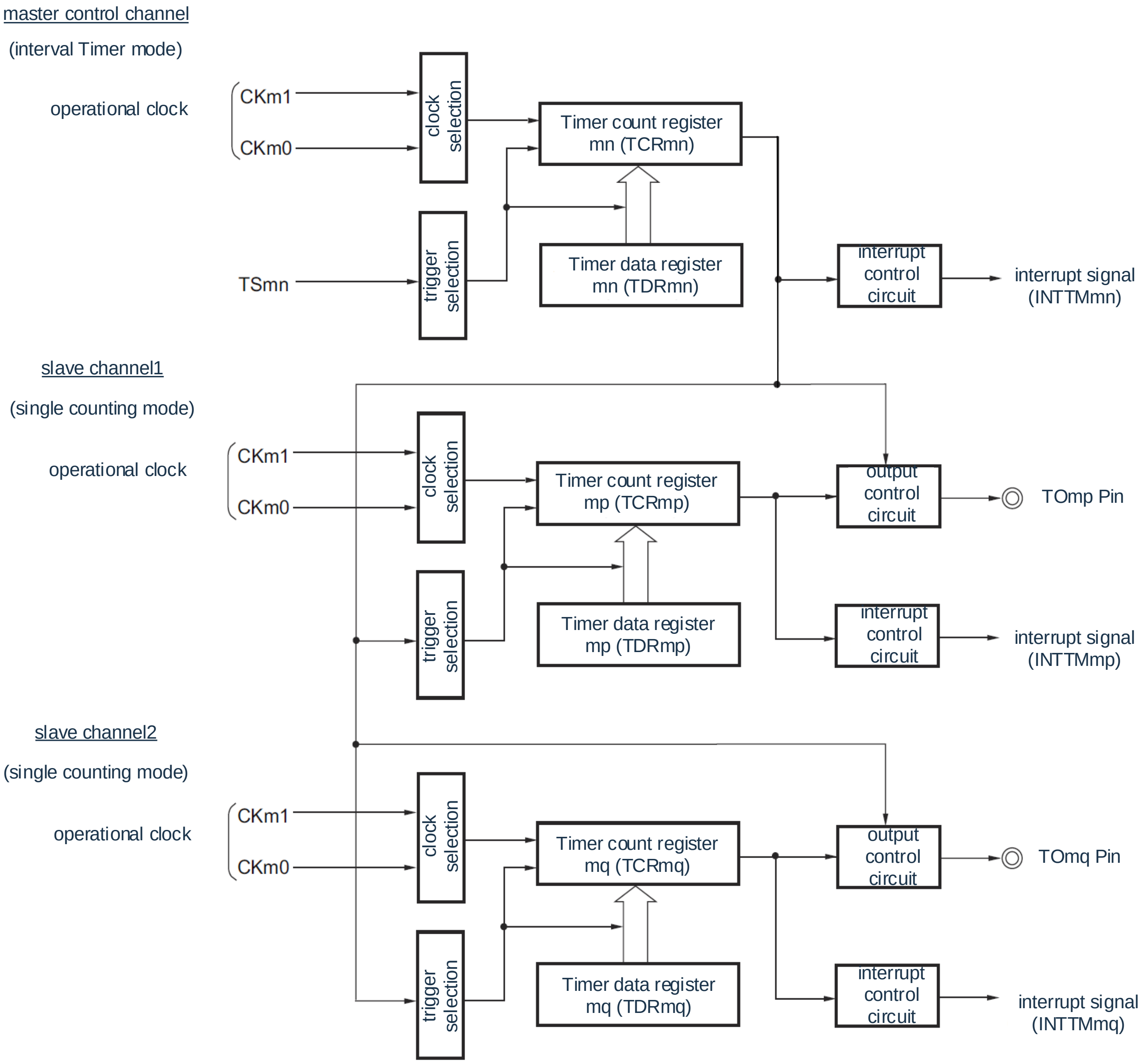
The same as the TCRmp register of the slave channel 1, in a single count mode, the TCRmq register of the slave channel 2 runs and counts the duty cycle and outputs a PWM waveform from the TOmq pin. Starting with the INTTMmn of the main control channel, the value of the TDRmq register is loaded into the TCRmq register and decremented. If TCRmq becomes "0000H," the INTTMmq is output and counts are stopped before the next start trigger (INTTMmn of the master channel) is entered. After the INTTMmn is generated from the main control channel and 1 counting clock passes, the output level of TOmq becomes effective level, and if TCRmq becomes "0000H".

When the channel 0 is used as the main control channel by such operation, up to three PWM signals can be output simultaneously.

Note: At least 2 write accesses are required when the timer data register mn(TDRmn) of the master channel and the TDRmp register of slave channel 1. Because the value of the TDRmn register and the TDRmp register are loaded into the TCRmn register and the TCRmp register when the master channel generates the INTTMmn, if the TOmp pin is rewritten before the master channel generates the INTTMmn and after the generation, the expected waveform cannot be output. Therefore, to override both the master TDRmn register and the slave TDRmp register, you must override both registers immediately after generating INTTMmn in the master channel (also applies to the slave TDRmq register).

Note: m: Unit number (m=0, 1) n: Master channel number (n=0, 2, 4)
p: Slave channel number q: Slave channel number
n＜p＜q≤3 when m=0 (P and q are integers greater than n)
n＜p＜q≤7 when m=1 (P and q are integers greater than n)

Figure 6-72 Block diagram for operating the multiple PWM output function (output of 2 PWM cases)



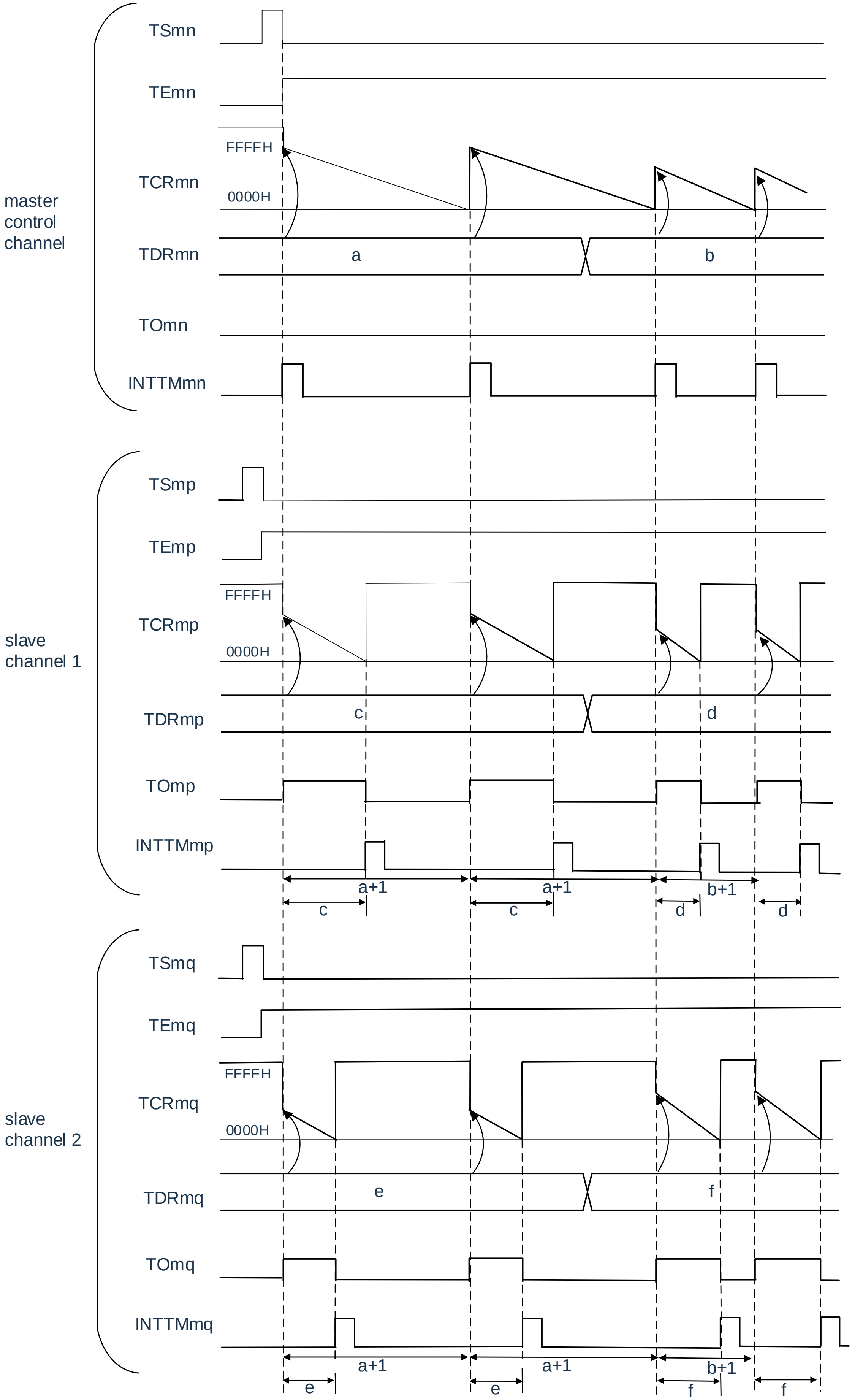
Note: m: Unit number (m=0, 1) n: Master channel number (n=0, 2, 4)

p: Slave channel number q: Slave channel number

n＜p＜q≤3 when m=0 (P and q are integers greater than n)

n＜p＜q≤7 when m=1 (P and q are integers greater than n)

Figure 6-73 Example of basic timing operating as multiple PWM output function (output two types of PWMs)



Remarks: 1. m: Unit number (m=0, 1) n: Master channel number (n=0, 2, 4)

p: Slave channel number q: Slave channel number

n＜p＜q≤3 when m=0 (P and q are integers greater than n)

n＜p＜q≤7 when m=1 (P and q are integers greater than n)

2. TSmn, TSmp, TSmq: bit n, p,q of timer channel start register m (TSm)

TEmn, TEmp, TEmq: bit n, p, q of timer channel enable status register m (TEm).

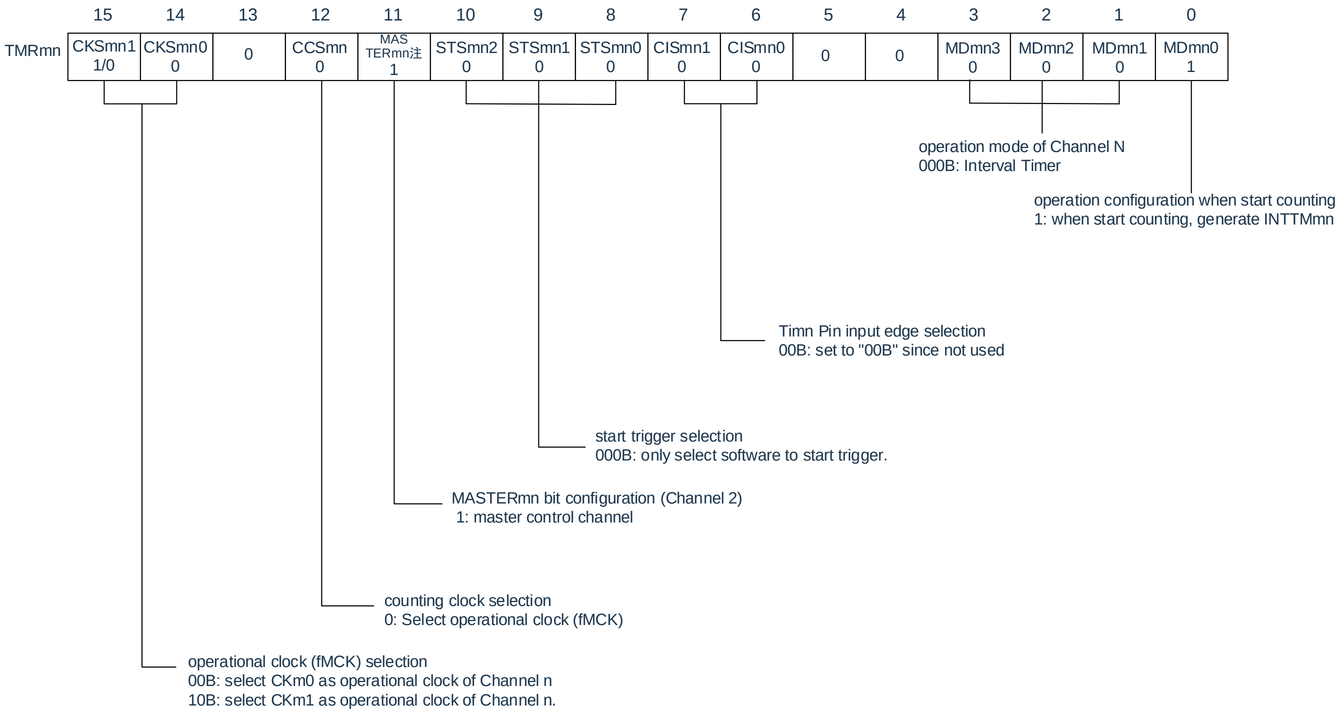
TCRmn, TCRmp, TCRmq: timer count registers mn, mp, mq (TCRmn, TCRmp, TCRmq)

TDRmn, TDRmp, TDRmq: timer data register mn, mp, mq (TDRmn, TDRmp,TDRmq)

TOmn, TOmp, TOmq: output signal of TOmn, TOmp, TOmq pins

Figure 6-74 Example of register contents setting for multiple PWM output function (master channel)

(a) Timer mode register mn (TMRmn)



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn	CKSmn1 1/0	CKSmn0 0	0	CCSmn 0	MAS TERmn注 1	STSmn2 0	STSmn1 0	STSmn0 0	CISmn1 0	CISmn0 0	0	0	MDmn3 0	MDmn2 0	MDmn1 0	MDmn0 1

operation mode of Channel N
000B: Interval Timer

operation configuration when start counting
1: when start counting, generate INTTMmn

Timn Pin input edge selection
00B: set to "00B" since not used

start trigger selection
000B: only select software to start trigger.

MASTERmn bit configuration (Channel 2)
1: master control channel

counting clock selection
0: Select operational clock (fMCK)

operational clock (fMCK) selection
00B: select CKm0 as operational clock of Channel n
10B: select CKm1 as operational clock of Channel n.

(b) Timer output register m (TOm)

	bit
TOm	TOmn 0

0: Output "0" from TOmn.

(c) Timer output enable register m (TOEm)

	bit
TOEm	TOEmn 0

0: Stop the TOmn output from the count run.

(d) Timer output level register m (TOLm)

	bit
TOLm	TOLmn 0

0: "0" is set at TOMmn=0 (Master Channel Output Mode).

(e) Timer output mode register m (TOMm)

	bit
TOMm	TOMmn 0

0: Sets the master channel output mode.

Note:		
	TMRm2, TMRm4	:MASTERmn=1
	TMRm0, TMRm5, TMRm7	: Fixed as "0".

Note: m: Unit number (m=0, 1) n: Master channel number (n=0, 2, 4)

Figure 6-75 Example of register setting content for multiple PWM output functions (slave channel) (output of 2 PWM cases)

(a) timer mode registers mp, mq (TMRmp, TMRmq)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmp	CKSmp1 1/0	CKSmp0 0	0	CCSmp 0	M/S 注 0	STSmp2 1	STSmp1 0	STSmp0 0	CISmp1 0	CISmp0 0	0	0	MDmp3 1	MDmp2 0	MDmp1 0	MDmp0 1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmq	CKSmq1 1/0	CKSmq0 0	0	CCSmq 0	M/S 注 0	STSmq2 1	STSmq1 0	STSmq0 0	CISmq1 0	CISmq0 0	0	0	MDmq3 1	MDmq2 0	MDmq1 0	MDmq0 1

operation mode of Channel p and q
100B:single counting mode

start trigger during ope
1: Trigger input valid.

Timp and TImq Pin input edge selection
00B: set to "00B" since not used

start trigger selection
100B: Select master control channel INTTMmn

MASTERmp bit and MASTERmq bit configuration
(Channel 2) 0: slave channel
SPLITmp bit and SPLITmq bit configuration
(Channel 1, 3) 0: 16 bit Timer

Count clock selection
0: Select operational clock (fMCK)

operational clock (fMCK) selection
00B: select CKm0 as operational clock of Channel p and q
10B: select CKm1 as operational clock of Channel p and q
※ same as master control channel configuration

(b) Timer output register m (TOm)

	bit q	bit p
TOm	TOmq 1/0	TOmp 1/0

0: Output "0" by TOmp and TOmq.
1: Output "1" by TOmp and TOmq.

(c) timer output enable register m (TOEm)

	bit q	bit p
TOEm	TOEmq 1/0	TOEmp 1/0

0: Stops the TOmp and TOmq output from the count run.
1: Allow TOmp and TOmq output from count runs.

(d) Timer output level register m (TOLm)

	bit q	bit p
TOLm	TOELq 1/0	TOELp 1/0

0: Positive Logical Output (High level valid)
1: Negative Logical Output (Low level valid)

(e) Timer output mode register m (TOMm)

	bit q	bit p
TOMm	TOMLq 1	TOMLp 1

1: Sets the slave channel output mode.

NOTE: TMRm2, 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
n＜p＜q≤3 when m=0 (P and q are integers greater than n)
n＜p＜q≤7 when m=1 (P and q are integers greater than n)

Figure 6-76 Procedure for multiple PWM output functions (output of 2 PWM cases) (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'. → Configure TOmp bit and Tomq bit, confirm TOmp and Tomq otuput initial voltage. → Set TOEmp bit and TOEmq to '1', enable TOmp and Tomq 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 and T0mq initial configured voltage level. 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.

Figure 6-77 Procedure for multiple PWM output functions (output of 2 PWM cases) (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 regiter 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 votlage levle 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 votlage levle 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 Tomp and TOmq 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 requried	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

n＜p＜q≤3 when m=0 (P and q are integers greater than n)

n＜p＜q≤7 when m=1 (P and q are integers greater than n)

6.10 Cautions when using the universal timer unit

6.10.1 Cautions when using timer output

According to the product, the pins to which the timer output function is assigned may also be assigned the output of other multiplexing functions. In this case, when using the timer output, it is necessary to set the initial value of the other multiplexing function output.

Please refer to the "Chapter 2 Port Function".

Chapter 7 EPWM Output Control Circuit

7.1 Function of EPWM output control circuit

Using the PWM output function of Timer, one DC motor or two stepper motors can be controlled. The output can be truncated by truncating the source CMP0 output, the INTP0 input, and the EVENTC event. The software allows you to select from four outputs: Hi-Z output, low output, high output, and anti-truncation output during forced truncation.

7.2 Structure of output control circuit

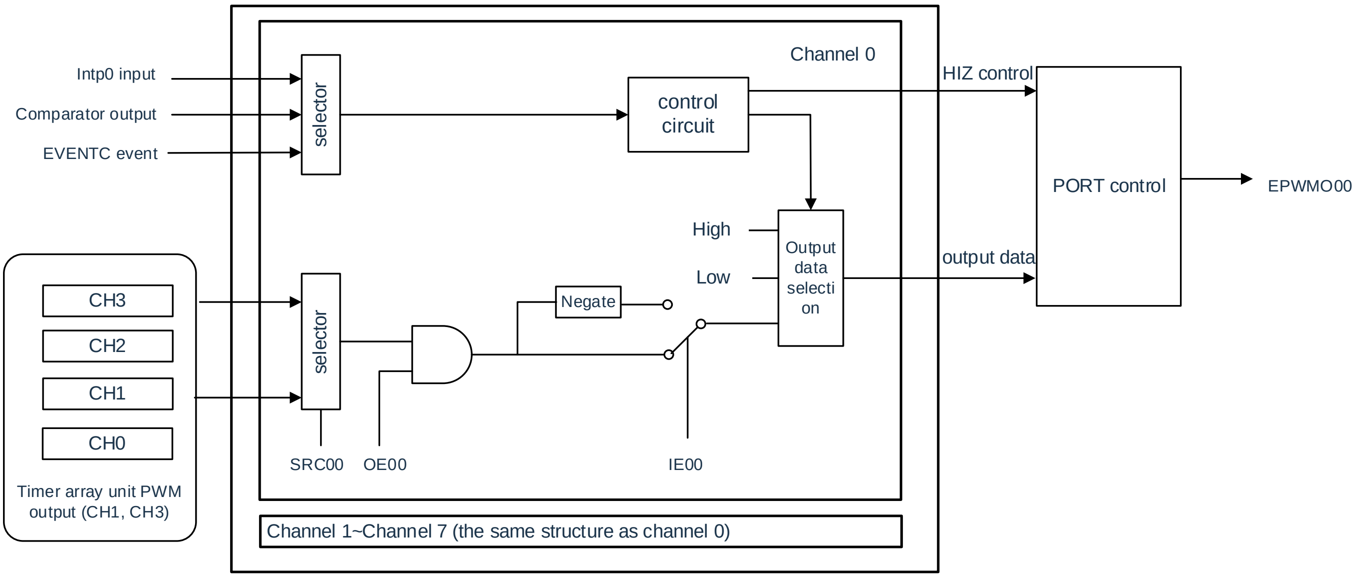
The EPWM output control circuit consists of the following hardware.

Table 7-1 Structure of output control circuit of EPWM

Item	Structure
Control register	EPWM input source selection register (EPWMSRC).
	EPWM output control register (EPWMCTL).
	EPWM force truncated input selection register (EPWMSTC).
	EPWM force truncated output selection register (EPWMSTL).
	EPWM Status Register (EPWMSTR).
Output	EPWM output(EPWMO00~EPWMOP07)

The block diagram of the EPWM output control circuit is shown in Figure 7-1.

Figure 7-1 Block diagram of EPWM output control circuit



7.3 Registers for controlling EPWM output control circuit

The real-time output control circuit is controlled by the following registers.

- Peripheral enable register 0 (PER1).
- EPWM input source select register (EPWMSRC).
- EPWM output control register (EPWMCTL).
- EPWM force truncated input select register (EPWMSTC).
- EPWM force truncated output select register (EPWMSTL).
- EPWM status register (EPWMSTR).
- Port mode register (PMxx).
- Port mode control register (PMCxx).
- Port register (Pxx).

7.3.1 Peripheral enable register 1 (PER1)

The PER1 register is a register that sets the clock that allows or disables clocking each peripheral hardware. Reduce power consumption and noise by stopping clocking unused hardware.

To use the EPWM function, EPWMEN must be set to “1”.

See “4.3.6 Peripheral Enable Registers 0, 1 (PER0, PER1)” for details.

7.3.2 EPWM input source selection register (EPWMSRC)

The EPWMSRC register selects the source clock of the input clock of the real-time output circuit. Select Timer's timer output TO01 or TO03 as the source clock and input to the EPWM.

The EPWMSRC register is set via an 8-bit memory operation command.

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

Figure 7-2 Format of EPWM input source selection register

Address: 0x40044400 After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
EPWMSRC	SRC07	SRC06	SRC05	SRC04	SRC03	SRC02	SRC01	SRC00

SRC0n	Select the source clock for the EPWM0n output
0	Select TO01
1	Select TO03

Remark: n: Channel number (n=0~7).

7.3.3 EPWM output control register (EPWMCTL)

The EPWMCTL register performs allowable control and reverse control of the waveform output of EPWMO00 to EPWMO03.

The EPWMCTL registers are set via 16-bit memory operation instructions.

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

Figure 7-3 Format of EPWM output control register (EPWMCTL)

Address: 0x40044408 After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EPWMCTL	IE07	IE06	IE05	IE04	IE03	IE02	IE01	IE00	OE07	OE06	OE05	OE04	OE03	OE02	OE01	OE00

OE0n	Control of EPWMO0n output
0	Disable output
1	Enable output

Remark: n: Channel number (n=0~7).

IE0n	Reverse control of EPWMO0n output
0	Not reversed
1	Reversed

Remark: n: Channel number (n=0~7).

7.3.4 EPWM force truncated input select register (EPWMSTC)

The EPWMSTC register makes the selection of the input source forced truncation.

The EPWMSTC register is set via 8-bit memory operation instructions.

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

Figure 7-4 Format of EPWM force truncated input select register (EPWMSTC)

Address: 0x40044404 After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
EPWMSTC	0	0	0	REL_SEL	HS_SEL	IN_EG	SC_SEL1	SC_SEL0

SC_SEL1	SC_SEL0	Selection of truncation sources[Note 1, 3, 4]
0	0	Do not select
0	1	Do not select
1	0	INTP0 terminal input
1	1	Event input from EVENTC

IN_EG	Source of force truncation/edge selection of force truncation output source[Note 1, 2]
0	Rising edge: Output force truncation Falling edge: Output force truncation released
1	Rising edge: Output force truncation released Falling edge: Output force truncation

HS_SEL	Output mode selection for forced truncation
0	Software release
1	Hardware release

REL_SEL	Release timing selection for forced output truncation
0	After the release signal generated by hardware or software occurs, the truncation is immediately released and the pulse output is restored.
1	After the release signal generated by hardware or software occurs, wait for the following timing: Select TO01 as the channel of the source clock: Truncation is released on the rising edge of the next TO01, and the pulse output is restored Select TO03 as the channel of the source clock: the cut-off is released on the rising edge of the next TO03 and the pulse output is restored

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

Note 2: Valid only when INTP0 input is selected.

Note 3: When using EVENTC to unenforce the cut-off, software dismiss must be selected (HS_SEL set to 1). There is no restriction when using I NTP0 input.

Note 4: The effective width of the input selected INTP0 must be greater than one clock cycle.

7.3.5 EPWM force truncated output select register (EPWMSTL)

The output state of the EPWMO terminal when the EPWMSTL register is forcibly truncated.

The EPWMSTL registers are set via 16-bit memory operation instructions.

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

Figure 7-5 Format of EPWM force truncated output select register (EPWMSTL)

Address: 0x4004440C After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EPWMSTL	IO71	IO70	IO61	IO60	IO51	IO50	IO41	IO40	IO31	IO30	IO21	IO20	IO11	IO10	IO01	IO00

IOn1	IOn0	Selection of terminal output when truncated
0	0	Truncation is prohibited
0	1	HI-Z output
1	0	Low level output
1	1	High level output

Remark: n: Channel number (n=0~7).

7.3.6 EPWM status register (EPWMSTR)

The EPWMSTR register clears the forced truncation signal and displays the truncation status. If the clear trigger bit HZCLR is set to “1”, the truancy state is released. When the truncation status indicates that the signal of the SHTFLG is high, it enters the forced truncation state. bit0 is write-only bit, and the read value is always “0”. bit7~1 is read-only.

The EPWMSTR registers are set via 8-bit memory operation instructions.

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

Figure 7-6 Format of EPWM status register (EPWMSTR)

Address: 0x4004410 After reset: 0000H R/W

Symbol	7	6	5	4	3	2	1	0
EPWMSTR	0	0	0	0	0	0	SHTFLG	

SHTFLG	Force truncation status flag
0	Normal output state
1	Force truncation state

HZCLR	Software clearance to force truncation signals
0	-
1	The software dismisses the truncation state

Notice: When the Output Selection Register (EPWMSTL) is set to disable cut-off by forcing truncation, the SHTFLG is set to “1” because of the input from an external truncation source, but truncation is not performed.

7.3.7 Control register for the port function of the EPWM output pin

When using the EPWM output, the control register (Port Mode Register (PMxx, PMCxx)) for the port function multiplexed with the EPWM output pin (EPWMOn pin) must be set. For details, refer to “2.3.1 Port Mode Register (PMxx)”.

When using the multiplexed ports of the EPWM pins as outputs of EPWMO, the bits of the port mode registers (PMxx, PMCxx) corresponding to each port must be set to “0”. In this case, the bit of the port register (Pxx) can be “0” or “1“.

For details, please refer to “2.5 Register Settings When Using the Multiplexing Function“.

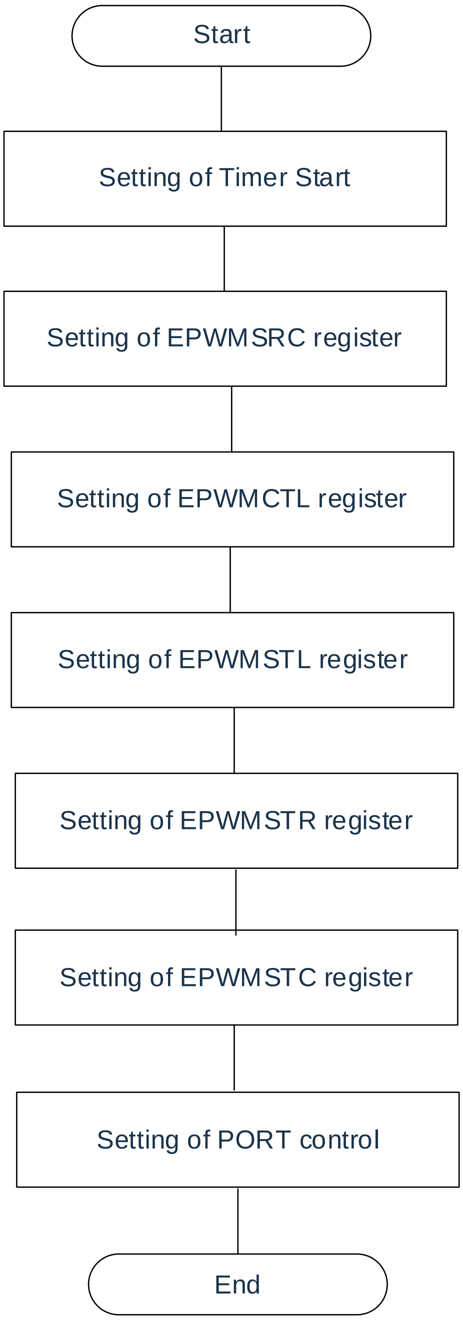
7.4 Operation of EPWM output control circuit

7.4.1 Initial setup

The timer waveform selects the TAU output (TO01, TO03) as the source clock through the EPWSRC register. The positive or inverting phase of the timer waveform can be fixed by setting the EPWMCTL register.

In the event of forced truncation, the Hi-Z output, low output, high output, or disable cut-off output can be selected through the setting of the EPWMSTL register.

Figure 7-7 Initial configuration flow of registers

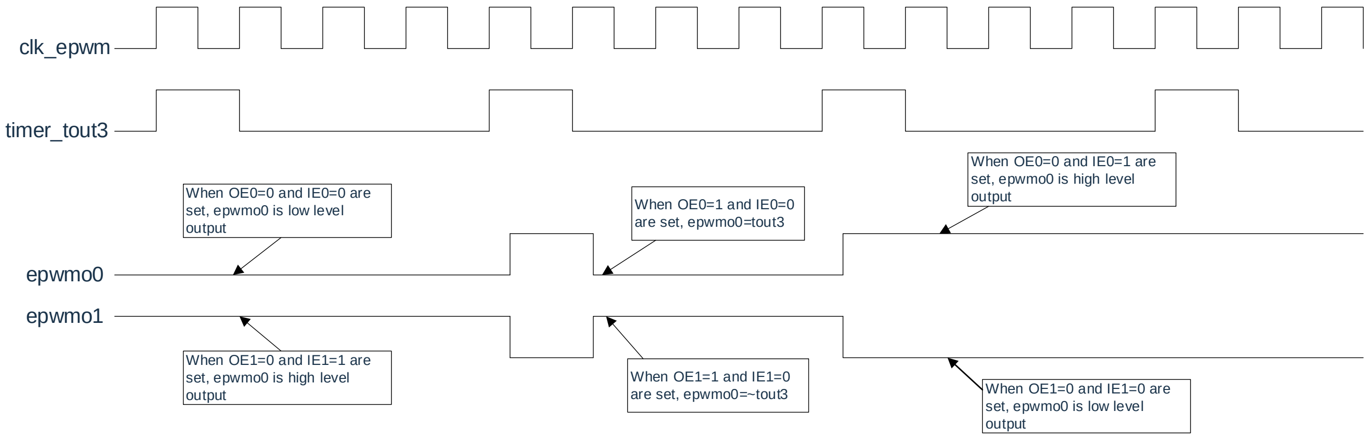

Start
Setting of Timer Start
Setting of EPWMSRC register
Setting of EPWMCTL register
Setting of EPWMSTL register
Setting of EPWMSTR register
Setting of EPWMSTC register
Setting of PORT control
End

7.4.2 Normal operation

Depending on the register settings, four output data can be selected, namely forward waveform output, inverted waveform output, low level output, and high level output. The EPWMCTL registers can be changed at runtime. Both OE0n bits and IE0n bits must be written at the same time.

For details, please refer to "Table 7-2 Operation Instructions for truncation signals".

Figure 7-8 Output timing diagram


clk_epwm
timer_tout3
epwmo0
epwmo1
When OE0=0 and IE0=0 are set, epwmo0 is low level output
When OE0=1 and IE0=0 are set, epwmo0=tout3
When OE0=0 and IE0=1 are set, epwmo0 is high level output
When OE1=0 and IE1=1 are set, epwmo0 is high level output
When OE1=1 and IE1=0 are set, epwmo0=~tout3
When OE1=0 and IE1=0 are set, epwmo0 is low level output

7.4.3 Force truncation processing

EPWM can select CMP0 output, INTP0, through the EPWMSTC register bit1,0 input, along with the E VENTC event, causes the EPWMO output to enter a forced truncation state.

(1) Occurrence of forced truncation

The INTP0 input and EVENTC events are truncated via the CMP0 output. By bit2(IN_EG) of EPWMSTC register, it can select the rising or falling edge and enter the truncated state after 1 to 2 clocks. For details, please refer to Figure 7-9.

(2) Release of forced truncation

a) Software release: When bit3 (HS_SEL) of EPWMSTC register is 0, the software release mode is used. Bit 0 (HZCLR) of EPWMSTR register is the clear bit of truncated status. When the truncated status flag SHTFLG is high, if the HZCLR bit is set to "1", the truncated status flag SHTFLG goes low and the forced truncated status is released.

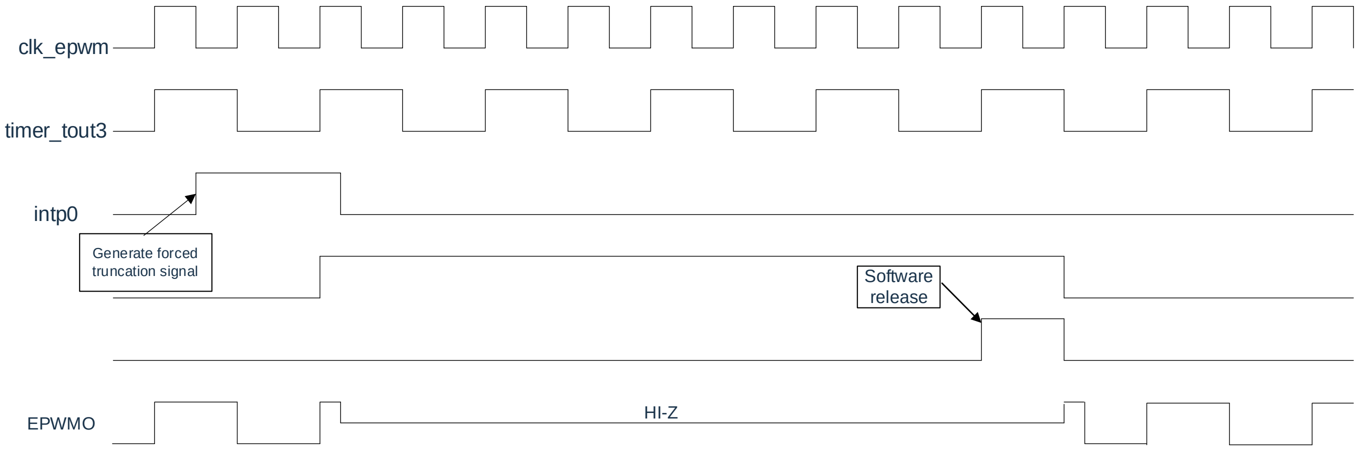
b) Hardware release: When bit3 (HS_SEL) of EPWMSTC register is 1, the hardware release mode is used. The forced truncation state is released by the edge of CMP0 output or INTP0 input.

Table 7-2 Table of operation Instructions for truncation signals

Bit	IOn1-0	OE0n	IE0n	SHTFLG	EPWM output pin
set value	00	1	0	*	Positive rotation waveform
	00	1	1	*	Invert the waveform
	01	*	*	*	Low level output
	10	*	*	*	High level output
	11	*	*	1	HI-Z output

Remark n=0～7

Figure 7-9 Timing diagram for generation and release of INTP0 truncation (HS_SEL=0, REL_SEL=0)



Notice: Short pulses may be generated when switching from “normal operation” to “Hi-Z”, “fixed low” or “fixed high” during forced cutoff caused by the cutoff signal INTP0, or when returning to the forced cutoff state by immediate release.

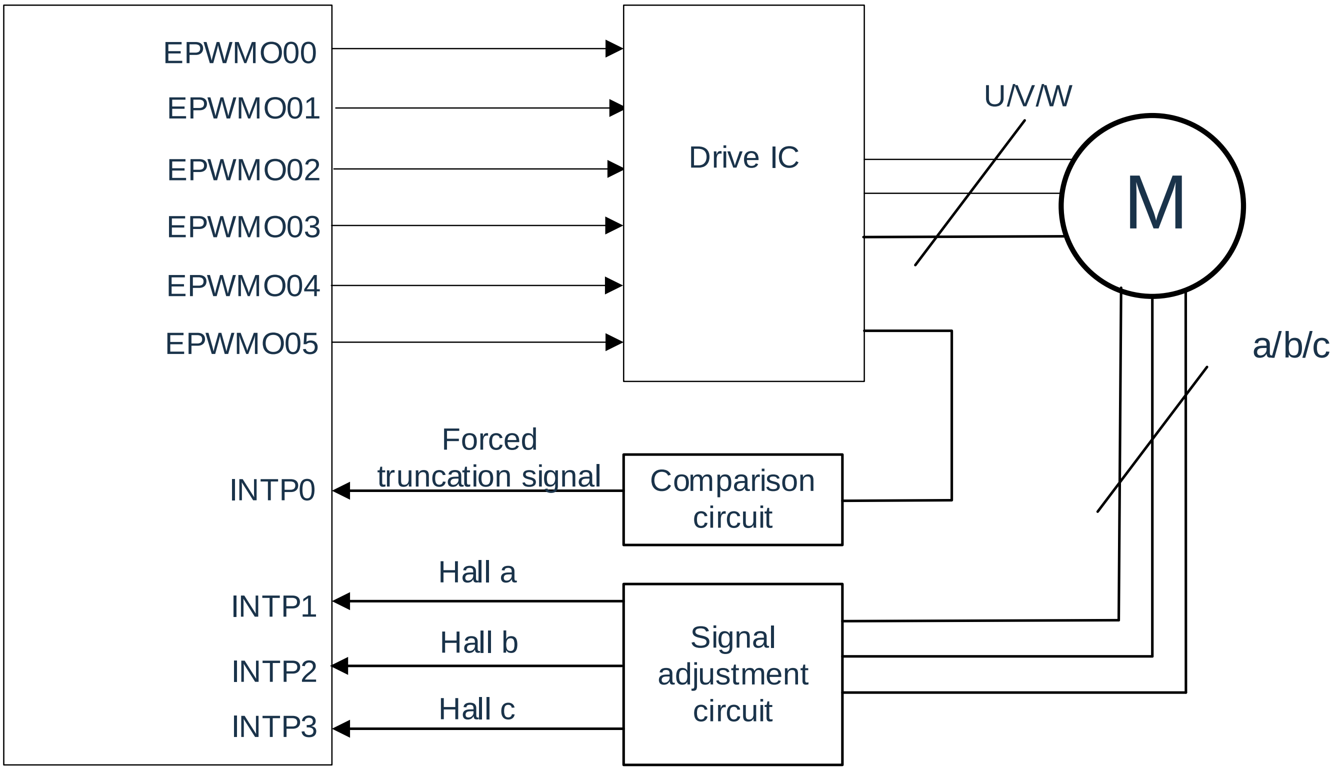
7.5 Control example of brushless DC motor

The following is an example of using the EPWM control function to control a brushless DC motor (hereinafter referred to as a BLDC motor).

7.5.1 Example of hardware connections

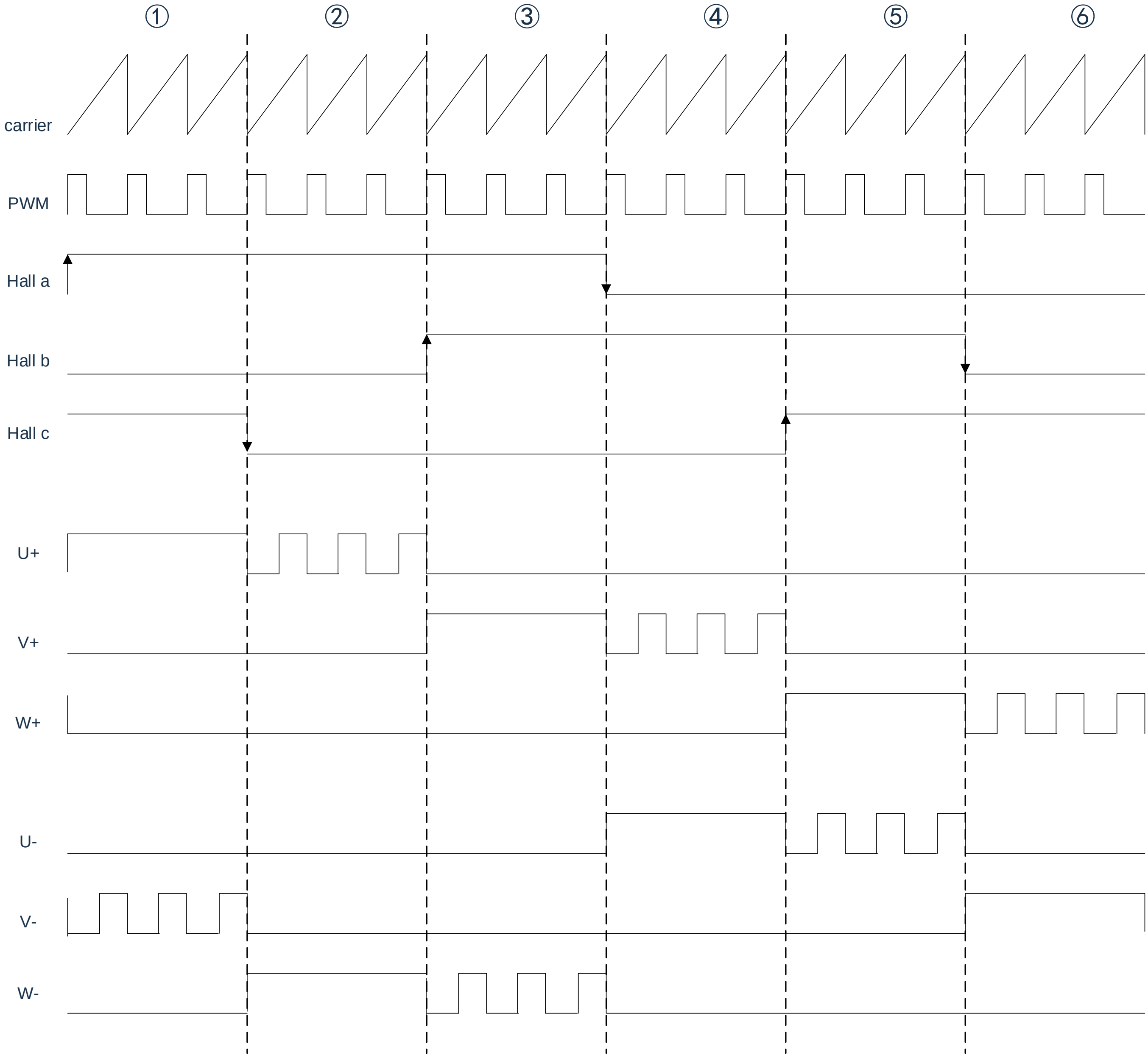
An example of a hardware connection for a brushless DC motor is shown in Figure 7-10. In this example, EPWMO00~EPWMO05 (output) is used for output control of BLDC motors, INTP1~INTP3(input) for the output signal of the Hall sensor, and INTP0 (input) is used to force a truncated signal.

Figure 7-10 Example of a hardware connection



7.5.2 Control timing of three-phase brushless DC motors

Figure 7-11 Control timing of a three-phase brushless DC motor



7.5.3 Example of register setting

In this example, the EPWM source selection registers (EPWMSRC) and EPWM control registers (EPWMCTL) are initialised to simultaneously output a waveform of positive rotation from EPWM00 to EPWM05 to the BLDC motor.

1. Set EPWMSRC5 to EPWMSRC0 in the EPWMSRC register to “0” and channel 1 of Timer as the input source of EPWMO00 ~ EPWMO05.

2. Set EPWMOE3 to EPWMOE0 in the EPWMCTL register to “1” to allow EPWMO03 ~ EPWM00 to be output. Set EPWMIE3 to EPWMIE0 of EPWMCTL register to “0”, EPWMO00 ~ EPWMO03 will be output in positive direction.

3. Set EPWMOE5 to EPWMOE4 in the EPWMCTL register to “1” to allow EPWMO05 to EPWM04 to be output. Set EPWMIE5 ~ EPWMIE4 in the EPWMCTL register to “1” to reverse the output of EPWMO04~ EPWMO05.

Table 7-3 Example of setting the EPWMCTL0 register

Description	Set value of the EPWMCTL
State ①: rising edge of Hall a Disable U+, U+ reverse outputs, enable V–, V–forward outputs.	0x0110
State ②: falling edge of Hall c Enable U+, U+ forward outputs, and disable W–, W– reverse outputs.	0x2001
State ③: rising edge of Hall b Disable V+, V+ reverse outputs, enable W–, W– forward outputs.	0x0220
State ④: falling edge of Hall a Enable V+, V+ forward outputs, disable U–, U–reverse outputs.	0x0802
State ⑤: rising edge of Hall c Disable W+, W+ reverse outputs, enable U–, U–forward outputs.	0x0408
State ⑥: falling edge of Hall b Enable W+, W+ forward outputs, disable V–, V–reverse outputs.	0x1004

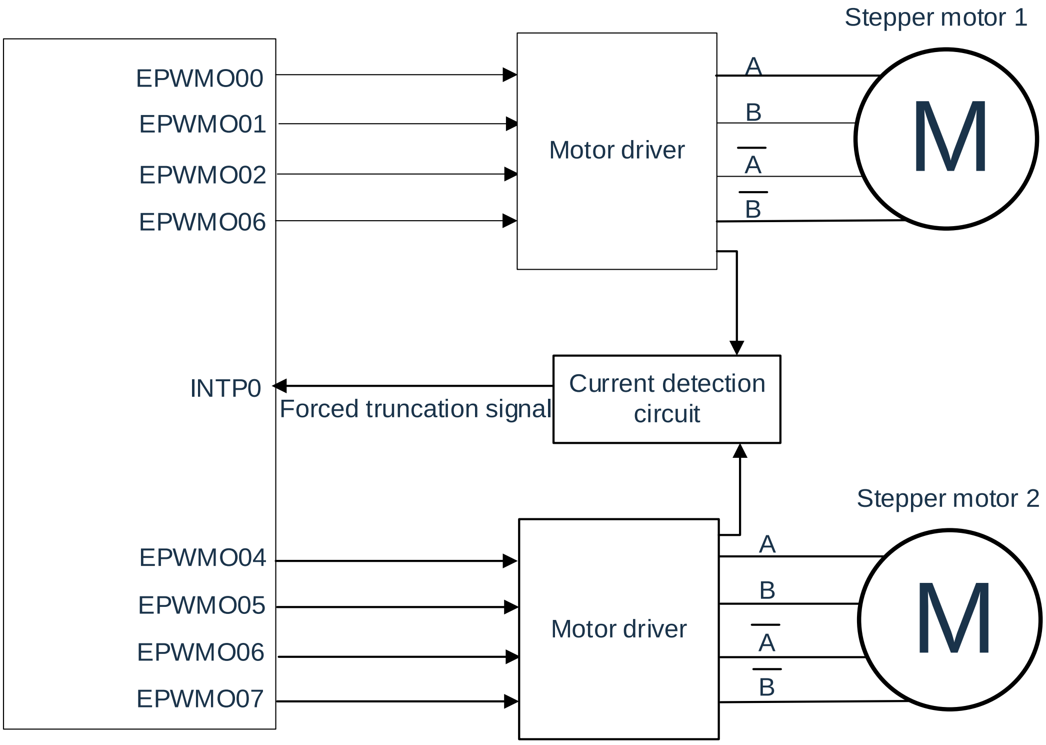
7.6 Example of stepper motor control

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

7.6.1 Example of a hardware connection

An example of a hardware connection to control two stepper motors is shown in Figure 7-12.

Figure 7-12 Example of a hardware connection



7.6.2 Control method

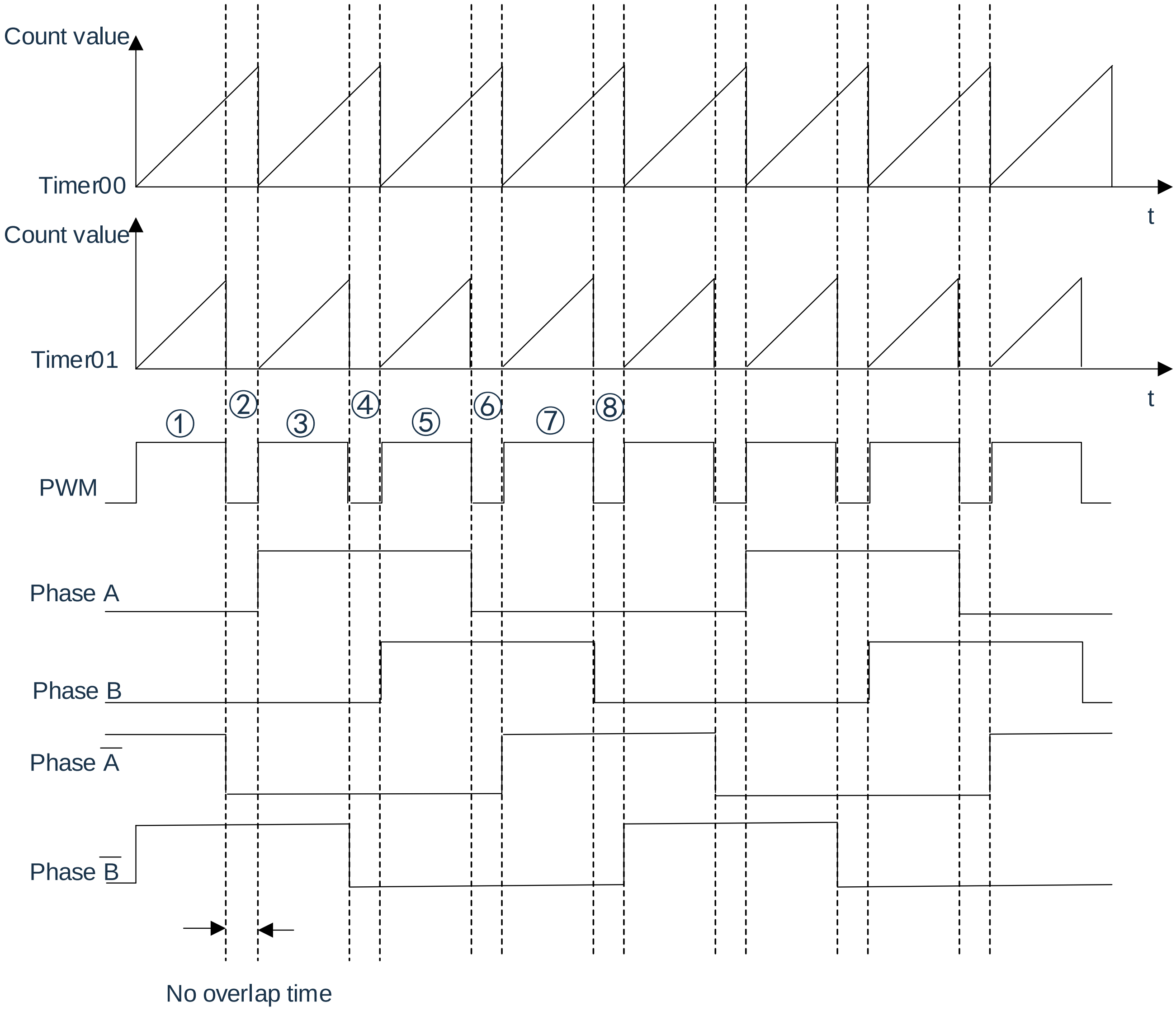
The stepper motor is rotated, reversed or stopped in two-phase excitation mode by using eight EPWMOs. Control the rotation speed via Timer's PWM mode.

In this example, Timer's CH0 and CH1 are used for the control of stepper motor 1, CH2 and CH3 are used for the control of stepper motor 2. If you combine 2 Timer channels, you can generate pulses of any period and duty cycle. CH0 and CH2 are the main control channels and operate as interval timer mode. CH1 and CH3 are slave channels and operate as single-count mode.

In addition, the cross-current prevention time (no overlapping time) is inserted when switching the output type.

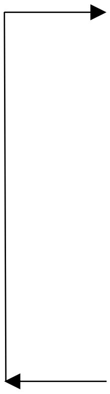
An example of a waveform for stepper motor control is shown in Figure 7-13.

Figure 7-13 Waveform example of step motor control



7.6.3 Example of register setting

Table 7-4 Example of setting the register that controls the stepper motor

State		Setting value of EPWMSRC	Setting value of EPWMCTL
	①	0x00	0x4400
	②	0x00	0x4000
	③	0x00	0x4100
	④	0x00	0x0100
	⑤	0x00	0x0300
	⑥	0x00	0x0200
	⑦	0x00	0x0600
	⑧	0x00	0x0400

Chapter 8 Real-Time Clock

8.1 Function of real-time clock

The real-time clock has the following functions.

- Holds counters for years, months, weeks, days, hours, minutes, and seconds up to a maximum of 99 years.
- Fixed cycle break (cycles: 0.5 seconds, 1 second, 1 minute, 1 hour, 1 day, 1 month)
- Alarm clock interrupt (alarm clock: week, hour, minute)
- 1Hz pin out capability

8.2 Structure of real-time clock

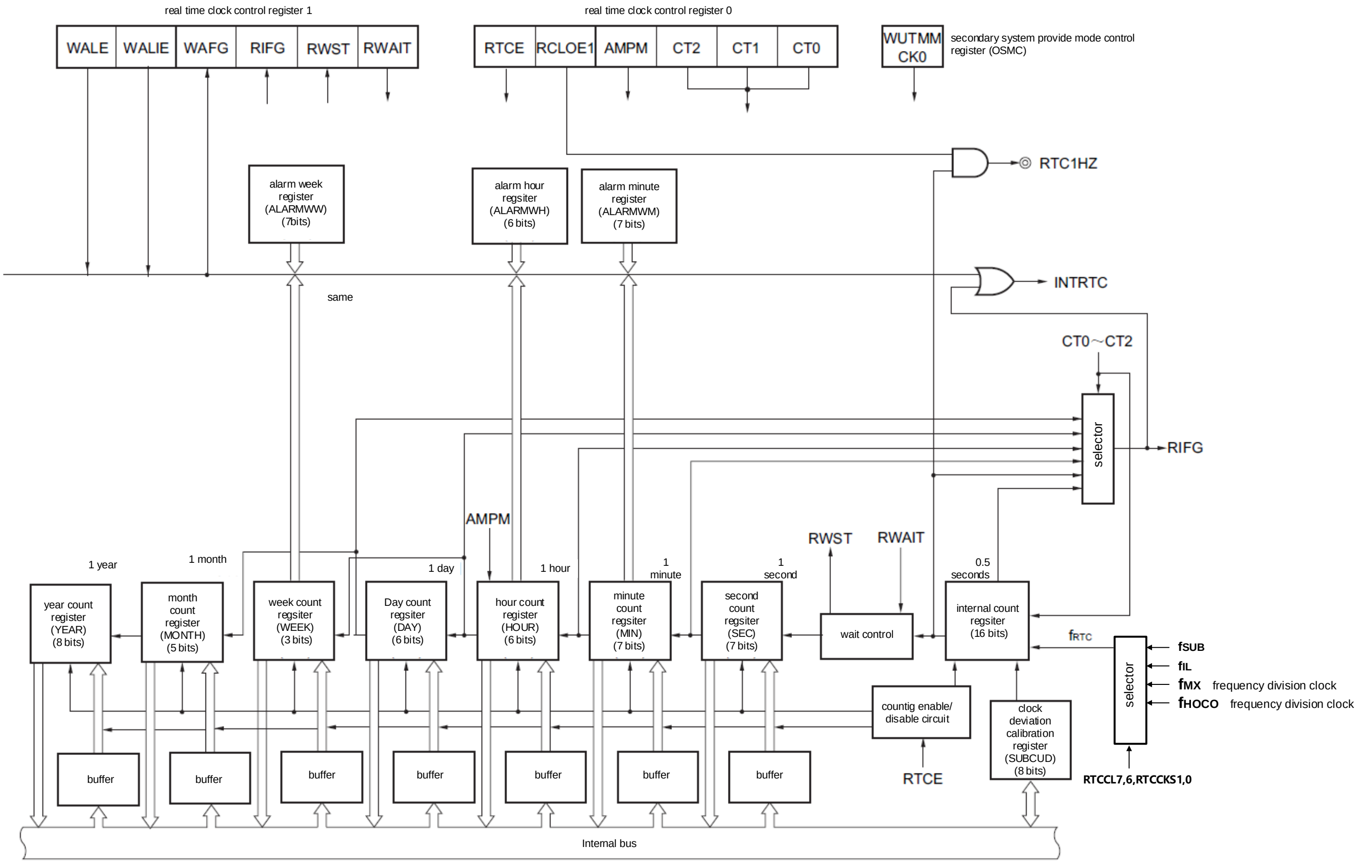
The real-time clock consists of the following hardware.

Table 8-1 Structure of real-time clock

Item	Structure
Counter	Internal Counter (16-bit)
Control register	Peripheral enable register 0 (PER0.bit7)
	Real-time clock selection register (RTCCL)
	Real-time clock control register 0 (RTCC0)
	Real-time clock control register 1 (RTCC1)
	Second count register (SEC)
	Minute count register (MIN)
	Hour count register (HOUR)
	Day count register (DAY)
	Week count register (WEEK)
	Month count register (MONTH)
	Year count register (YEAR)
	Clock error correction register (SUBCUD)
	Alarm clock minute register (ALARMWM)
	Alarm clock hour register (ALARMWH)
	Alarm clock week register (ALARMWW)

Note: The reset of the above RTC control register is only controlled by the POR reset.

Figure 8-1 Block diagram of real-time clock


real time clock control register 1
WALE
WALIE
WAFG
RIFG
RWST
RWAIT
real time clock control register 0
RTCE
RCLOE1
AMPM
CT2
CT1
CT0
WUTMM CK0
secondary system provide mode control register (OSMC)
RTC1HZ
alarm week register (ALARMWW) (7bits)
alarm hour regsiter (ALARMWH) (6 bits)
alarm minute register (ALARMWM) (7 bits)
same
INTRTC
CT0～CT2
selector
RIFG
AMPM
RWST
RWAIT
1 year
1 month
1 day
1 hour
1 minute
1 second
0.5 seconds
year count register (YEAR) (8 bits)
month count register (MONTH) (5 bits)
week count regsiter (WEEK) (3 bits)
Day count regsiter (DAY) (6 bits)
hour count regsiter (HOUR) (6 bits)
minute count regsiter (MIN) (7 bits)
second count regsiter (SEC) (7 bits)
wait control
internal count regsiter (16 bits)
fRTC
selector
fSUB
fIL
fMX frequency division clock
fHOCO frequency division clock
countig enable/ disable circuit
RTCE
clock deviation calibration register (SUBCUD) (8 bits)
RTCCL7,6,RTCCKS1,0
buffer
buffer
buffer
buffer
buffer
buffer
buffer
Internal bus

Note: Count years, months, weeks, days, hours, minutes and seconds only if you select f_{mx}/f_{hoco} clock (≈32, 768KHZ after every week) or the secondary system clock (f_{SUB}=32.768kHz) as the running clock for the real-time clock. When a low-speed internal oscillator clock (f_{IL}=15kHz) is selected, only a fixed cycle interrupt function is used.

The fixed cycle interrupt interval when selecting f_{IL} is calculated using the following equation:

Fixed period (value selected by the RTCC0 register) × f_{SUB}/f_{IL}

8.3 Register for controlling real-time clock

The real-time clock is controlled through the following registers.

- Peripheral enable register 0 (PER0).
- Real-time clock selection register (RTCCL)
- Real-time clock control register 0 (RTCC0)
- Real-time clock control register 1 (RTCC1)
- Second count register (SEC)
- Minute count register (MIN)
- Hour count register (HOUR)
- Day count register (DAY)
- Week count register (WEEK)
- Month count register (MONTH)
- Year count register (YEAR)
- Clock error correction register (SUBCUD)
- Alarm clock minute register (ALARMWM)
- Alarm clock hour register (ALARMWH)
- Alarm clock week register (ALARMWW)
- Port mode register (PMxx)
- Port mode control register (PMCxx)

8.3.1 Peripheral enable register 0 (PER0)

The PER0 register is a register that sets a clock that is allowed or prohibited to supply to each peripheral hardware. Reduce power consumption and noise by stopping clock supply to unused hardware.

You must set bit7 (RTCEN) to '1' when you want to use real-time clocks. The PER0 register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "00H".

Figure 8-2 Format of peripheral enable register 0 (PER0)

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

symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	XX	XX	XX	XX	XX	XX	XX

RTCEN	Control of an input clock of a real-time clock (RTC) and a 15-bit interval timer
0	Stop provide an input clock. · You cannot write the SFR used by the real-time clock (RTC) and 15-bit interval timers. · The real-time clock (RTC) and the 15-bit interval timer are reset.
1	Provides an input clock. · SFRs that can read and write real-time clocks (RTCs) and 15-bit interval timers.

Note:1. If you want to use the real-time clock, you must first set the RTCEN bit to "1" while the counting clock (f_{RTC}) oscillation is stable, and then set the following registers. When the RTCEN bit is "0", the write operation of the real-time clock control register is ignored, and the read values are initial (except RTCCL, port mode register, and port register).

- Real-time clock control register 0 (RTCC0)
- Real-time clock control register 1 (RTCC1)
- Second count register (SEC)
- Minute count register (MIN)
- Hour count register (HOUR)
- Day count register (DAY)
- Week count register (WEEK)
- Month count register (MONTH)
- Year count register (YEAR)
- Clock error correction register (SUBCUD)
- Alarm clock minute register (ALARMWM)
- Alarm clock hour register (ALARMWH)
- Alarm clock week register (ALARMWW)

2. By setting the RTCLPC bit in the Subsystem Clock Supply Mode Control Register (OSMC) to "1", the subsystem clock can be stopped for peripheral functions other than the real-time clock and 15-bit interval timer in deep sleep mode or sleep mode running with the subsystem clock.

8.3.2 Real-time clock selection register (RTCCL)

A real-time clock and a count clock of a 15-bit interval timer (fRTC) can be selected through RTCCL.

Figure 8-3 Format of real-time clock selection register (RTCCL)

Address: 0x4004047C After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
RTCCL	RTCCL7	RTCCL6	RTCCL5	0	0	0	RTCCKS1	RTCCKS0

RTCCL7	Selection of Clock Source for Real-time Clock and Counter Clock of 15-bit Interval Timer
0	Select a high speed system clock (fMX)
1	Select a high speed internal oscillator (fhoco)

RTCCKS1	RTCCKS0	RTCCL6	RTCCL5	Selection of running clock for real time clock, counting clock of 15-bit interval timer
0	0	x	x	Subsystem Clock (f_{SUB})
0	1			Low-speed internal oscillator clock (f_{IL}) (WUTMMCK0 must set to 1)
1	0	0	1	Main clock fmx/fhoco (via RTCCL7 selection)/1952
1	0	0	0	Main clock fmx/fhoco (via RTCCL7 selection)/1464
1	0	1	0	Main clock fmx/fhoco (via RTCCL7 selection)/976
1	1	0	0	Main clock fmx/fhoco (via RTCCL7 selection)/488
1	1	1	0	Main clock fmx/fhoco (via RTCCL7 selection)/244

8.3.3 Real-time clock control register 0 (RTCC0)

This is an 8-bit register that sets the start or stop of real-time clock operation, the control of RTC1HZ pins, the 12/24-hour system and fixed cycle interrupts.

The RTCC0 register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "00H".

Figure 8-4 Format of real-time clock control register 0(RTCC0)

Address: 0x40044F5D After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
RTCC0	RTCE	0	RCLOE1 Note	0	AMPM	CT2	CT1	CT0

RTCE	Real-time clock operation control
0	Stop the counter from running.
1	Start the counter running.

RCLOE1[note]	Output control of RTC1HZ Pin
0	Disables the output of the RTC1HZ pin (1Hz).
1	Allow RTC1HZ pin output (1 Hz).

AMPM	Selection of 12-hour system/24-hour system
0	12-hour system (for AM or PM)
1	24-hour system
· To change the value of the AMPM bit, the RWAIT bit (bit0 of Real-Time Clock Control Register 1 (RTCC1) must be overridden. If you change the value of the AMPM bit, the value of the HOUR register becomes the corresponding value of the time system you set. · Time frames are shown in Table 8-2.	

CT2	CT1	CT0	Selection of fixed cycle interrupt (INTRTC)
0	0	0	The fixed-cycle interrupt function is not used.
0	0	1	Once every 0.5 seconds (synchronized with seconds accumulation)
0	1	0	Once every 1 second (synchronized with seconds accumulation)
0	1	1	Once every minute (00 seconds per minute).
1	0	0	Once every hour (00 minutes and 00 seconds per hour).
1	0	1	Once a day (00:00:00 per day).
1	1	×	Once a month (1st of each month at 00:00:00 a.m.).
To change the CT2~CT0 bit value in counter run (RTCE=1), INTRTC must be set to disable interrupt handling by interrupt mask register, and RIFG and RTCIF flags must be cleared after override and then set to allow interrupt handling.			

Note: 1. You cannot change the RTCE bit when the RCLOE1 bit is '1'.

2. If the RTCE bit is "0", 1Hz is not output even if the RCLOE1 bit is set to "1".

Remark: ×: Ignore

8.3.4 Real-time clock control register 1 (RTCC1)

This is an 8-bit register that controls the alarm clock interrupt function and the counter wait. The RTCC1 register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "00H".

Figure 8-5 Format of real-time clock control register 1 (RTCC1) (1/2)

Address: 0x40044F5E After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
RTCC1	WALE	VALLE	0	WAFG	RIFG	0	RWST	RWAIT

WALE	Operation control of alarm clock
0	The consistent run is invalid.
1	Consistent operation is valid.
When setting the WALE bit with the WALIE bit "1" in the counter running (RTCE=1), INTRTC must be set to suppress interrupt handling by the interrupt mask register and the WAFG and RTCIF flags must be cleared after the override. To set each alarm register (WALIE flag of RTCC1 register, alarm minute register (ALARMWM), alarm hour register (ALARMWH) and the alarm week register (ALARMWW)), the WALE bit must be set to "0" (invalid for consistent operation).	

VALLE	Operation control of INTRTC (alarm clock interrupt) function
0	Interrupt consistently without an alarm clock.
1	Interrupt that alarm clock consistently occur.

WAFG	Alarm clock detection status flag
0	The alarm clock is out of sync.
1	Consistent alarm clock detected.
This is a status flag indicating that a consistent alarm clock has been detected. Valid only if WALE bit is '1', and becomes '1' after detecting that alarm clock is consistent one F_{RTC} clock has elapsed. Clear this flag by writing "0" to it. Invalid operation to write "1".	

Figure 8-5 Format of real-time clock control register 1 (RTCC1) (2/2)

RIFG	Fixed cycle interrupt status flag
0	No fixed cycle interrupt was generated.
1	Interrupt of a fixed cycle is generate.
This is a status flag indicating that a fixed cycle interrupt is generated. This flag is "1" when a fixed cycle interrupt is generated. Clear this flag by writing "0" to it. Invalid operation to write "1".	

RWST	Wait state flag for real-time clock
0	Counter is running.
1	In read-write mode for the counter.
This is the state indicating whether the setting for the RWAIT bit is valid. The count value must be read and written after confirming this flag as "1".	

RWAIT	Wait control of real-time clock
0	Set to counter run.
1	Set SEC~YEAR counter to stop running and enter read-write mode of counter.
This bit controls the operation of the counter. To read and write a count value, you must write "1" to this bit. Because the internal counter (16-bit) continues to run, the read and write must end within 1 second and then return to "0". The time required from the RWAIT bit set to "1" to the time the count value can be read and written (RWST=1) is at least 1 F_{RTC} clock. If an internal counter (16 bits) overflows when the RWAIT bit is "1", the overflow state is maintained and the count is incremented after the RWAIT bit becomes "0".	

Remark: 1. Fixed cycle interrupts and alarm clock consistent interrupts use the same interrupt source (INTRTC). When INTRTC interrupt occurs, which interrupt occurs can be judged by confirming fixed period interrupt state flag RIFG and alarm clock detection state flag WAFG.

2. If you write a second count register (SEC), clear the internal counter (16 bits).

8.3.5 Clock error correction register (SUBCUD)

This is a register capable of correcting clock speed with high accuracy by changing the overflow value from the internal counter (16 bits) to the second counter (SEC) (reference value: 7FFFH).

The SUBCUD register is set by a 16-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "00000H".

Figure 8-6 Format of clock error correction register (SUBCUD)

Address: 0x40044F34H After reset: 0000H R/W

symbol	15	14	13	12	11	10	9	8
SUBCUD	DEV	0	0	F12	F11	F10	F9	F8

7	6	5	4	3	2	1	0
F7	F6	F5	F4	F3	F2	F1	F0

DEV	Setting of Time Sequence for Correcting Clock Error
0	The clock error correction is performed when the second bits are "00", "20", and "40".
1	Clock error correction is only performed when the second bit is "00" (every 60 seconds).
Disable writing SUBCUD registers for the period shown: •DVE=0: Period of SEC=00H, 20H, 40H •DVE=1: Period of SEC=00H	

F12	Setting of clock error correction value
0	{(F11, F10, F9, F8, F7, F6, F5, F4, F3, F2, F1, F0)-1}×2 increase
1	{(/F11,/F10,/F9,/F8,/F7,/F6,/F5,/F4,/F3,/F2,/F1,/F0)+1}×2 Reduction
When (F12,F11,F10,F9,F8,F7,F6,F5,F4,F3,F2,F1,F0)=(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 Or (1,0,0,0,0,0,0,0,0,0,0,0,0,0,1), no clock error correction. Range of correction values: (F12=0)2,4,6,8,......,8186,8188 (F12=1)-2,-4,-6,-8,......,-8186,-8188	

Notice: "/" denotes the inverse of each.

The range of correction that can be performed by the clock error correction register (SUBCUD) is as follows.

	DEV=0 (correction every 20 seconds)	DEV=1 (correction every 60 seconds)
correctable range	-12496.9 ppm~12496.9 ppm	-4165.6 ppmto 4165.6 ppm
maximum quantization error	±1.53ppm	±0.51ppm
minimum resolution	±3.05ppm	±1.02ppm

Remark: The DEV bit must be set to "0" when the correction range exceeds -4165.6ppm~4165.6 ppm.

8.3.6 Second count register (SEC)

This is an 8-bit register that represents the value of the second meter in 0-59 decimal. An incremental count is performed by overflowing an internal counter (16 bits).

At write time, the data is first written to the buffer and then to the counter after passing up to 2 fRTC clocks. Decimal 00-59 must be set in BCD-code.

The SEC register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "00H".

Figure 8-7 Format of second count register (SEC)

Address: 0x40044F52 After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
SEC	0	SEC40	SEC20	SEC10	SEC8	SEC4	SEC2	SEC1

Notice: When you want to read and write this register in the counter run (RTCE=1), “8.4.3 Real-time clock counter reading and writing "in steps.

Remark: If you write a second count register (SEC), the internal counter (16 bits) is cleared.

8.3.7 Minute count register (MIN)

This is an 8-bit register that represents the minutes value in 0-59 (decimal). Incrementally counts by overflowing the second counter.

At write time, the data is first written to the buffer and then to the counter after passing up to 2 fRTC clocks. The overrun of the second count register is ignored during a write operation and set to a write value. Decimal 00-59 must be set in BCD-code.

The MIN register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "00H".

Figure 8-8 Format of minute count register (MIN)

Address: 0x40044F53 After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
MIN	0	MIN40	MIN20	MIN10	MIN8	MIN4	MIN2	MIN1

Notice: To read and write this register while the counter is running (RTCE=1), you must follow the steps described in “8.4.3 Real-time clock counter reading and writing".

8.3.8 Hour count register (HOUR)

This is an 8bit register that represents hourly values with 00-23 or 01-12, 21-32 decimal values. Incrementally count by overflowing the minutes counter.

At write time, the data is first written to the buffer and then to the counter after passing up to 2 fRTC clocks. The overflow of the minute count register is ignored during a write operation and set to a write value.

The decimal 00~23 or 01~12,21~32 must be set in BCD code according to the bit3(AMPM) setting of RTCC0.

If you change the value of the AMPM bit, the value of the HOUR register becomes the corresponding value of the time system that is set. The HOUR register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "12H".

However, if the AMPM bit is set to "1" after reset, the value of this register changes to "00H".

Figure 8-9 Format of hour count register (HOUR)

Address: 0x40044F54 After reset: 12H R/W

symbol	7	6	5	4	3	2	1	0
HOUR	0	0	HOUR20	HOUR10	HOUR8	HOUR4	HOUR2	HOUR1

Notice: 1. When the AMPM bit is selected as “0” (12-hour system), bit5 (HOUR20) of the Hour register is indicated AM(0) /PM(1).

2. To read and write this register while the counter is running (RTCE=1), you must follow the steps described in "8.4.3 Real-time clock counter reading and writing".

The relationship between the config value of the AMPM bit, the value of the hour count register (HOUR), and the time is shown in Table

Table 8-2 Representation of the time bits

24-hour representation (AMPM=1)		12-hour representation (AMPM=0)	
Time	HOUR register	Time	HOUR register
0	00H	12 a.m.	12H
1	01H	1 a.m.	01H
2	02H	2 a.m.	02H
3	03H	3 a.m.	03H
4	04H	4 a.m.	04H
5	05H	5 a.m.	05H
6	06H	6 a.m.	06H
7	07H	7 a.m.	07H
8	08H	8 a.m.	08H
9	09H	9 a.m.	09H
10	10H	10 a.m.	10H
11	11H	11 a.m.	11H
12	12H	12 p.m.	32H
13	13H	1 p.m.	21H
14	14H	2 p.m.	22H
15	15H	3 p.m.	23H
16	16H	4 p.m.	24H
17	17H	5 p.m.	25H
18	18H	6 p.m.	26H
19	19H	7 p.m.	27H
20	20H	8 p.m.	28H
21	21H	9 p.m.	29H
22	22H	10 p.m.	30H
23	23H	11 p.m.	31H

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 24 hours.

The bit5 of the HOUR register indicates AM/PM at the 12 hour representation. Morning (AM) is "0" and afternoon (PM) is "1.

8.3.9 Day count register (DAY)

This is an 8-bit register that represents the daily count value in 1-31 decimal. An incremental count is performed by overflowing the hour counter. The counter counts as follows.

- 01~31 (January, March, May, July, August, October, December)
- 01-30 (April, June, September, November)
- 01~29 (February, leap year)
- 01~28 (February, normal year)

At write time, the data is first written to the buffer and then to the counter after passing up to 2 f_{RTC} clocks. The overflow of the hour count register is ignored during the write operation and set to the write value. Decimal 01-31 must be set in BCD-code.

The day register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "01H".

Figure 8-10 Format of day count register (DAY)

Address: 0x40044F56H After reset: 01H R/W

symbol	7	6	5	4	3	2	1	0
DAY	0	0	DAY20	DAY10	DAY8	DAY4	DAY2	DAY1

Note: To read and write this register while the counter is running (RTCE=1), you must follow the steps described in “8.4.3 Real-time clock counter reading and writing".

8.3.10 Week count register (WEEK)

This is an 8-bit register that represents the day of the week value in 0-6 decimal. Increment counts in synchronization with the daily counter.

At write time, the data is first written to the buffer and then to the counter after passing up to 2 fRTC clocks. Must set decimal 00~06 with BCD code.

The WEEK register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "00H".

Figure 8-11 Format of the week count register (WEEK)

Address: 0x40044F55H After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
WEEK	0	0	0	0	0	WEEK4	WEEK2	WEEK1

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

Note: 1. The corresponding values of the MONTH and Day Count registers (DAY) are not automatically saved to the Day register (WEEK). The following settings must be made after the reset is removed:

2. To read and write this register while the counter is running (RTCE=1), you must follow the steps described in “8.4.3 Real-time clock counter reading and writing".

8.3.11 Month count register (MONTH)

This is an 8-bit register that represents the monthly count value in 1-12 decimal. The incremental count is performed by overflowing the daily counter.

At write time, the data is first written to the buffer and then to the counter after passing up to 2 fRTC clocks. The daily count register overflow is ignored during the write operation and set to the write value. Decimal 01-12 must be set in BCD code format.

The MONTH register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "01H".

Figure 8-12 Format of month count register (MONTH)

Address: 0x40044F57H After reset: 01H R/W

symbol	7	6	5	4	3	2	1	0
MONTH	0	0	0	MONTH10	MONTH8	MONTH4	MONTH2	MONTH1

Note: When you want to read and write this register in the counter run (RTCE=1), “8.4.3 Real-time clock counter reading and writing" The recorded steps are carried out.

8.3.12 Year count register (YEAR)

This is an 8-bit register that represents the annualized value in 0-99 decimal. Incrementing counts by overflowing the monthly counter (MONTH). 00,04,08,......, 92, and 96 are leap years.

At write time, the data is first written to the buffer and then to the counter after passing up to 2 fRTC clocks. The MONTH register overflow is ignored during the write operation and set to the write value. Decimal 00-99 must be set with BCD code. The YEAR register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "00H".

Figure 8-13 Format of year count register (YEAR)

Address: 0x40044F58H After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
YEAR	YEAR80	YEAR40	YEAR20	YEAR10	YEAR8	YEAR4	YEAR2	YEAR1

Note: To read and write this register while the counter is running (RTCE=1), you must follow the steps described in “8.4.3 Real-time clock counter reading and writing".

8.3.13 Alarm minute register (ALARMWM)

This is a register that sets alarm minutes.

The ALARMWM register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "00H".

Note: Decimal 00-59 must be set in BCD-code. If you set a value outside of the range, the alarm clock is not detected.

Figure 8-14 Format of alarm minute register (ALARMWM)

Address: 0x40044F5AH After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
ALARMWM	0	WM40	WM20	WM10	WM8	WM4	WM2	WM1

8.3.14 Alarm hour register (ALARMWH)

This is a register that sets alarm clock hours.

The ALARMWH register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "12H".

However, if the AMPM bit is set to "1" after reset, the value of this register changes to "00H".

Note: The decimal 00~23 or 01~12, 21~32 must be set in BCD code. If you set a value outside of the range, the alarm clock is not detected.

Figure 8-15 Format of alarm hour register (ALARMWH)

Address: 0x40044F5BH After reset: 12H R/W

symbol	7	6	5	4	3	2	1	0
ALARMWH	0	0	WH20	WH10	WH8	WH4	WH2	WH1

Note: When the AMPM bit is selected as "0" (12-hour system), the bit5 (WH20) of the ALARMWH register indicates AM(0)/PM(1).

8.3.15 Alarm clock week register (ALARMWW)

This is the register that sets the alarm week.

The ALARMWW register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "00H".

Figure 8-16 Format of alarm week register (ALARMWW)

Address: 0x40044F5CH After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
ALARMW	0	WW6	WW5	WW4	WW3	WW2	WW1	WW0

An example of setting an alarm clock time is shown below.

Alarm clock set time	Day							12-Hour Display				24-Hour Display			
	Sunday WW0	Monday WW1	Tuesday WW2	Wednesday WW3	Thursday WW4	Friday WW5	Saturday WW6	Hour 10	Hour 1	Minute 10	Minute 1	Hour 10	Hour 1	Minute 10	Minute 1
Every day 0:00 a.m	1	1	1	1	1	1	1	1	2	0	0	0	0	0	0
Every day 1:30 a.m.	1	1	1	1	1	1	1	0	1	3	0	0	1	3	0
Every day 11:59 a.m.	1	1	1	1	1	1	1	1	1	5	9	1	1	5	9
Monday-Friday 0:00 p.m	0	1	1	1	1	1	0	3	2	0	0	1	2	0	0
Sunday 1:30 p.m.	1	0	0	0	0	0	0	2	1	3	0	1	3	3	0
Monday, Wednesday, Friday 11:59 p.m.	0	1	0	1	0	1	0	3	1	5	9	2	3	5	9

8.3.16 Port mode register and port register

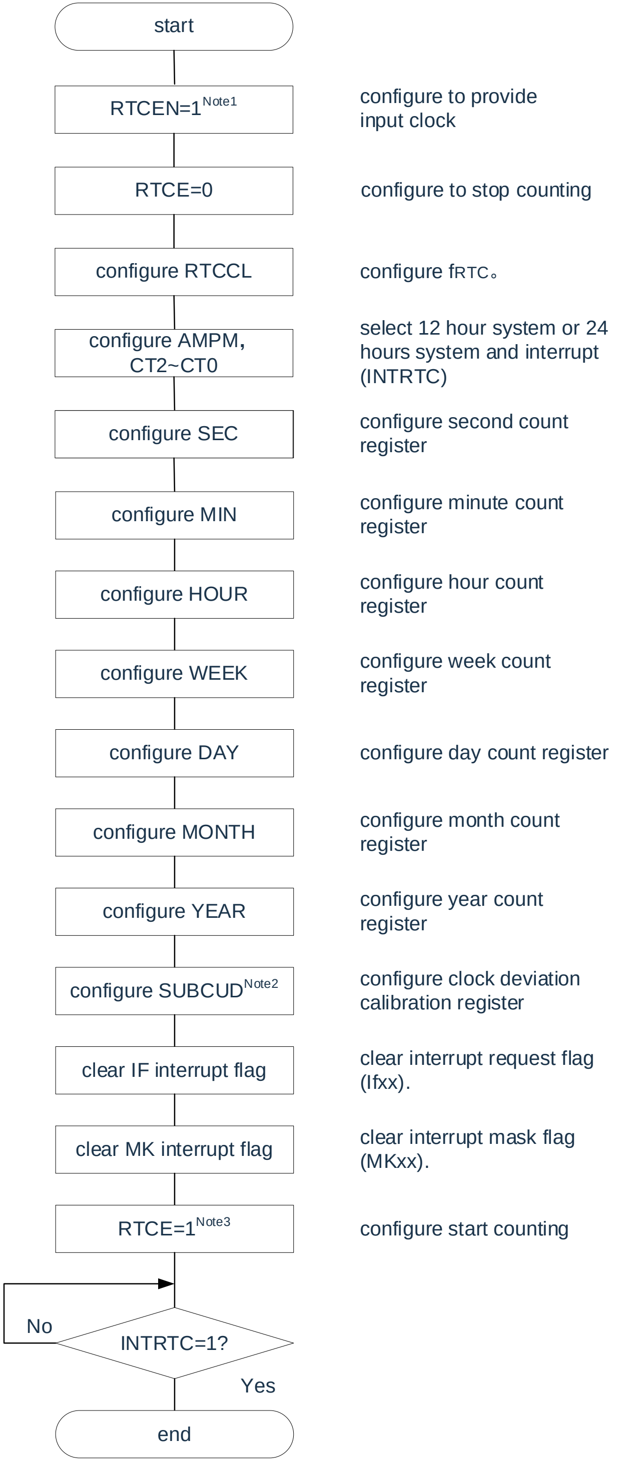
To output the multiplexed port of the RTC1HZ output pin with 1Hz, you must set "0" to the bit of the Port Mode Control Register (PMCxx), the bit of the Port Mode Register (PMxx), and the bit of the Port Register (Pxx) corresponding to each port.

The set port mode registers (PMxx), port registers (Pxx), and port mode control registers (PMCxx) differ by product. For more information, refer to “2.5 Register settings when using the multiplexing function”.

8.4 Operation of real-time clock

8.4.1 Start of real-time clock operation

Figure 8-17 Real-time clock start step


start
RTCEN=1Note1
configure to provide input clock
RTCE=0
configure to stop counting
configure RTCCL
configure fRTC。
configure AMPM，CT2~CT0
select 12 hour system or 24 hours system and interrupt (INTRTC)
configure SEC
configure second count register
configure MIN
configure minute count register
configure HOUR
configure hour count register
configure WEEK
configure week count register
configure DAY
configure day count register
configure MONTH
configure month count register
configure YEAR
configure year count register
configure SUBCUDNote2
configure clock deviation calibration register
clear IF interrupt flag
clear interrupt request flag (Ifxx).
clear MK interrupt flag
clear interrupt mask flag (MKxx).
RTCE=1Note3
configure start counting
No
INTRTC=1?
Yes
end

Note: 1. The RTCEN bit must first be set to "1" while the count clock (f_{RTC}) is oscillating and stable.

2. This is only a case where clock errors need to be corrected. Refer to the for how correction values are calculated "8.4.6 Example of clock deviation calibration for a real-time clock".

3. Please confirm the steps of "8.4.2 Shifting to sleep mode after starting operation" when the RTCE bit is " 1 " and is transferred to sleep mode without waiting for the INTRTC bit to "1".

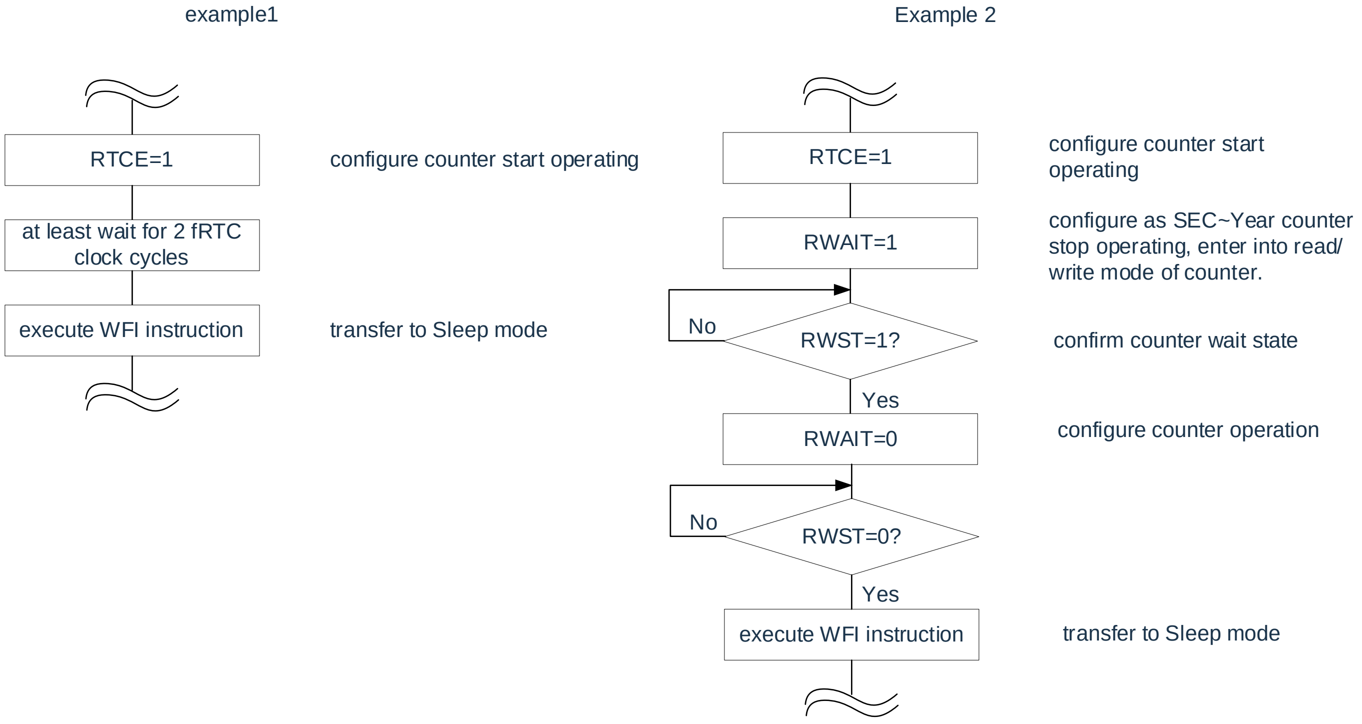
8.4.2 Shifting to sleep mode after starting operation

To transfer to sleep (including deep sleep) mode immediately after the RTCE set to “1”, one of the following treatments must be performed. However, after the RTCE set to “1” is taken, these processing is not required if you want to move to sleep mode after an INTRTC interrupt occurs.

· Transfer to sleep mode after at least 2 count clocks (fRTC) elapsed after the RTCE set to “1” (refer to Figure Example 1).

· After setting the RTCE bit to “1”, set the RWAIT bit to “1” and confirm that the RWST bit becomes “1” by polling. Then, set the RWAIT bit to “0” and poll again to make sure the RWST bit becomes “0”, then transfer to sleep mode (refer to Figure Example 2).

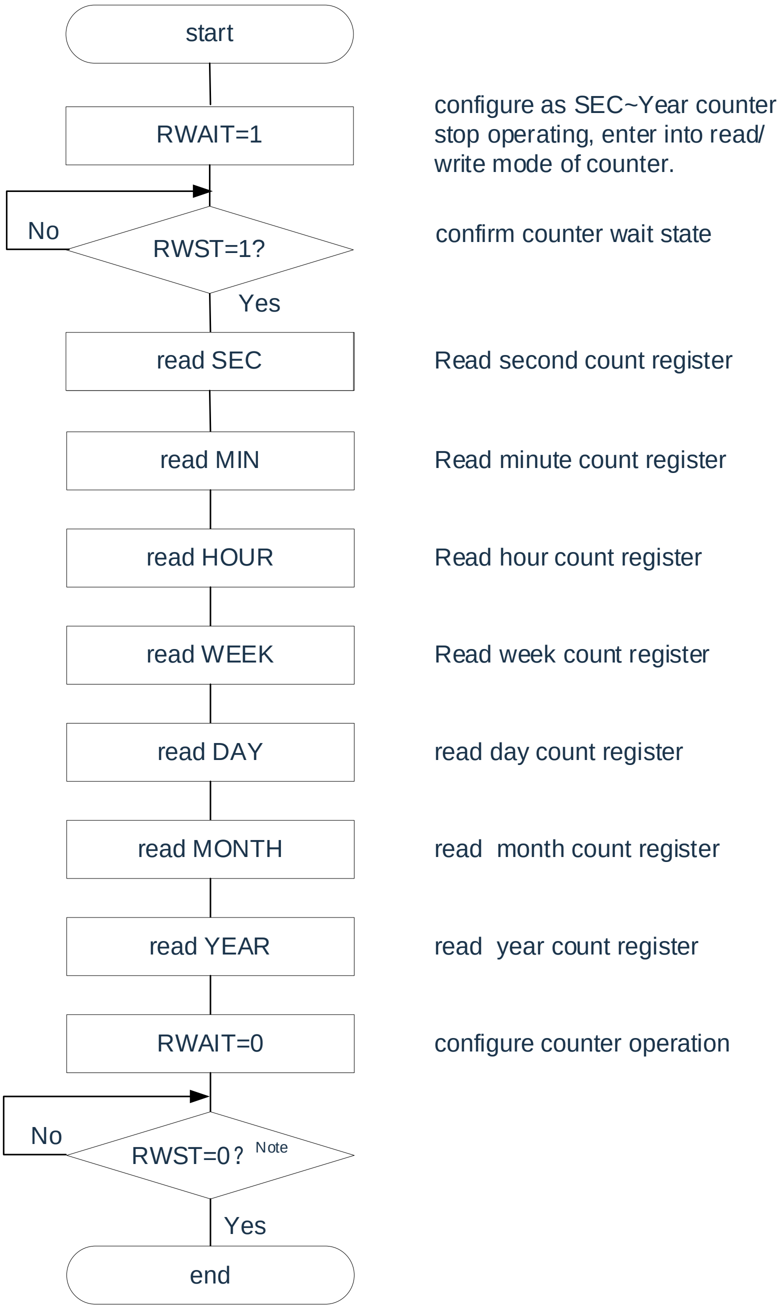
Figure 8-18 Procedure for shifting to sleep/deep sleep mode after setting RTCE bit to 1


example1
Example 2
RTCE=1
configure counter start operating
at least wait for 2 fRTC clock cycles
execute WFI instruction
transfer to Sleep mode
RTCE=1
configure counter start operating
RWAIT=1
configure as SEC~Year counter stop operating, enter into read/ write mode of counter.
No
RWST=1?
confirm counter wait state
Yes
RWAIT=0
configure counter operation
No
RWST=0?
Yes
execute WFI instruction
transfer to Sleep mode

8.4.3 Real-time clock counter reading and writing

Read or write the counter after setting “1” to RWAIT first. Set RWAIT to “0” after completion of reading or writing the counter.

Figure 8-19 Read operation steps of real-time clock counter

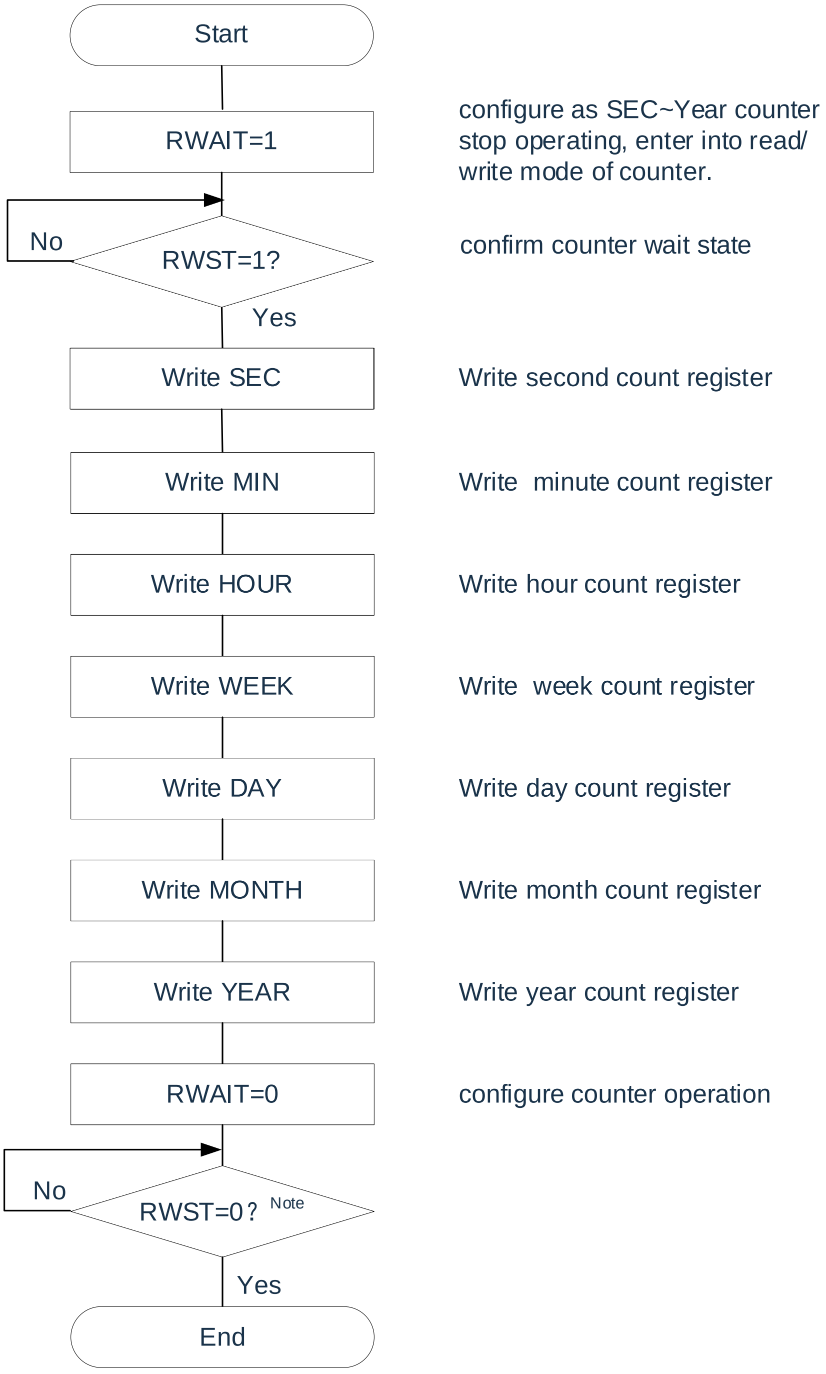

start
RWAIT=1
configure as SEC~Year counter stop operating, enter into read/ write mode of counter.
No
RWST=1?
confirm counter wait state
Yes
read SEC
Read second count register
read MIN
Read minute count register
read HOUR
Read hour count register
read WEEK
Read week count register
read DAY
read day count register
read MONTH
read month count register
read YEAR
read year count register
RWAIT=0
configure counter operation
No
RWST=0? Note
Yes
end

Note: You must verify that the RWST bit is "0" before moving to sleep mode.

Note: The processing of setting the RWAIT bit from "1" to "0" must be performed within 1 second.

Note: Do not limit the read order of seconds/minutes/hours/week/day/month/and year count register/s. It is possible to read only part of a register without read all of that register.

Figure 8-20 Read operation steps of real-time clock counter


Start
RWAIT=1
configure as SEC~Year counter stop operating, enter into read/ write mode of counter.
No
RWST=1?
confirm counter wait state
Yes
Write SEC
Write second count register
Write MIN
Write minute count register
Write HOUR
Write hour count register
Write WEEK
Write week count register
Write DAY
Write day count register
Write MONTH
Write month count register
Write YEAR
Write year count register
RWAIT=0
configure counter operation
No
RWST=0？ Note
Yes
End

Note: The RWST bit must be confirmed as '0' before being transferred to SLEEP mode.

Note: 1. The processing of setting the RWAIT bit from "1" to "0" must be performed within 1 second.

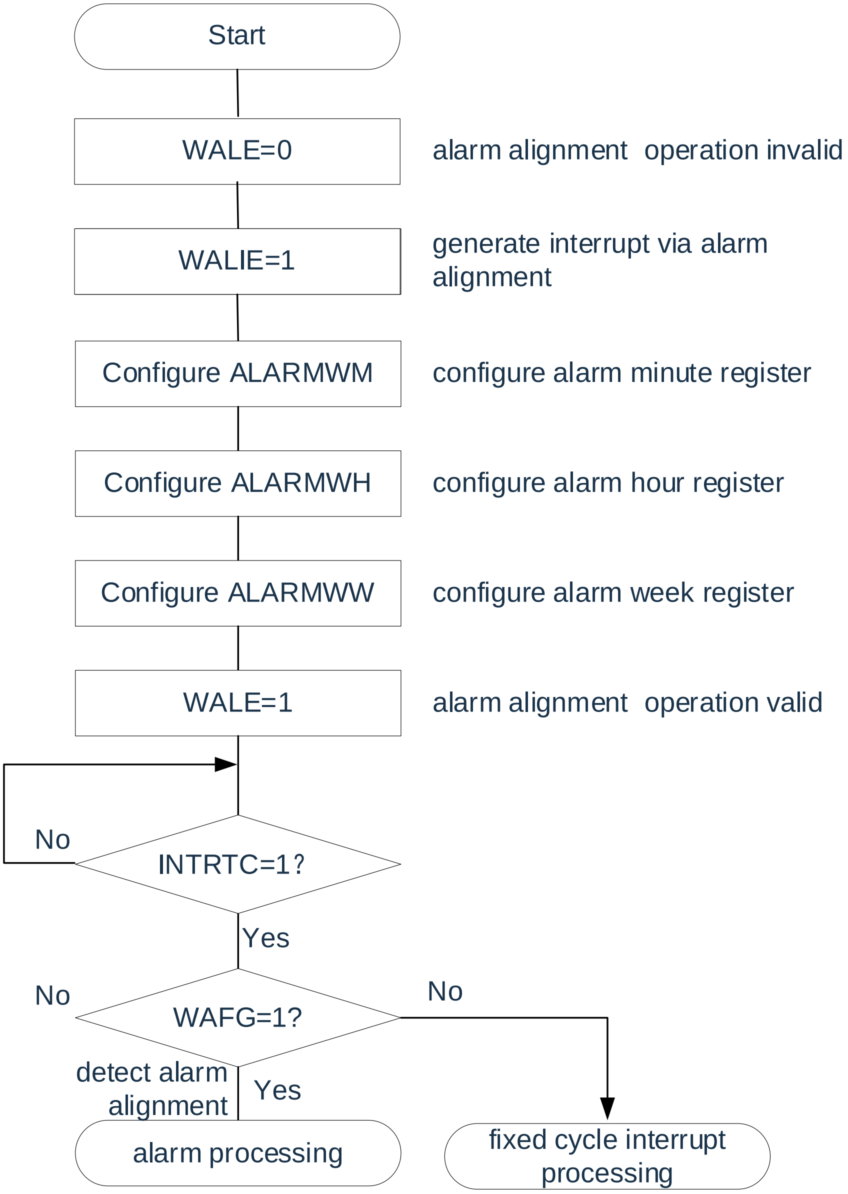
2. To override the SEC, MIN, HOUR, WEEK, DAY, MONTH, YEAR registers in the counter run (RTCE=1), INTRTC must be set by interrupt mask register to suppress interrupt handling for override, and the WAFG flag, RIFG flag, and RTCIF flag must be cleared after override.

Note: Do not limit the read order of seconds/minutes/hours/week/day/month/and year count register/s. It is possible to read only part of a register without read all of that register.

8.4.4 Alarm setting for real-time clock

You must first set the WALE to “0” (the alarm is not working) and then set the alarm time.

Figure 8-21 Alarm Clock Set-up Steps

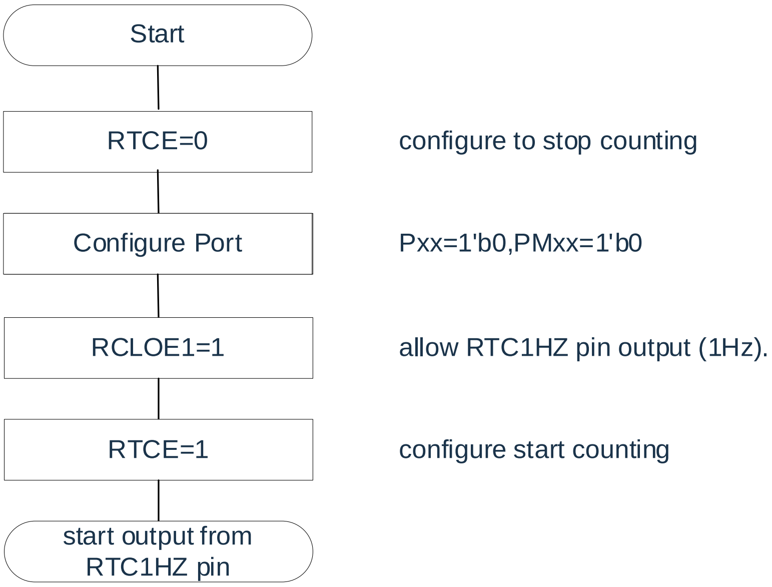

Start
WALE=0
alarm alignment operation invalid
WALIE=1
generate interrupt via alarm alignment
Configure ALARMWM
configure alarm minute register
Configure ALARMWH
configure alarm hour register
Configure ALARMWW
configure alarm week register
WALE=1
alarm alignment operation valid
No
INTRTC=1?
Yes
No
WAFG=1?
No
detect alarm alignment
Yes
alarm processing
fixed cycle interrupt processing

Note: 1. Write operation order of ALARMWM, ALARMWH, and ALARMWW is not restricted.

2. Fixed cycle interrupts and alarm clock consistent interrupts use the same interrupt source (INTRTC). When INTRTC occurs, it is possible to determine which interrupt occurs by confirming a fixed period interrupt status flag (RIFG) and an alarm detection status flag (WAFG).

8.4.5 1 Hz output of real-time clock

Figure 8-22 Set-up steps for 1Hz output


Start
RTCE=0
configure to stop counting
Configure Port
Pxx=1'b0,PMxx=1'b0
RCLOE1=1
allow RTC1HZ pin output (1Hz).
RTCE=1
configure start counting
start output from RTC1HZ pin

Note: 1. The RTCEN bit must first be set to "1" with the counting clock (fSUB) oscillating and stable.

8.4.6 Example of clock deviation calibration for a real-time clock

A clock speed correction can be performed with high accuracy by setting a value to a clock error correction register.

Example of calculation method of correction value

The correction value for 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, set 0 to DEV.

(When DEV=0)

Correction Value Note = 1 minute correction count value÷3 = (oscillation frequency ÷ target frequency -1) ×32768 ×60 ÷3

(When DEV=1)

Correction Value Note = 1 minute correction count value = (Oscillation Frequency ÷ target frequency -1) × 32768×60

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}×2

(Case of F12=1) Correction = -{(/F11,/F10,/F9,/F8,/F7,/F6,/F5,/F4,/F3,/F2,/F1,/F0)+1}×2

When (F12~F0)=(*,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,*), 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 (f_{RTC}).

Output frequency of the RTC1HZ pin when clock error correction register is initial value (00H) is 0×32768

3. The target frequency is the frequency corrected using the clock error correction register.

Correction example

Examples from 32767.4 Hz to 32768Hz (32767.4Hz+18.3ppm)

[Measurement of oscillation frequency]

The oscillating frequencies of the products are measured by outputting a signal of about 1Hz from the RTC1HZ pin when the clock error correction register (SUBCUD) is an initial value Note.

Note: Refer to the "8.4.5 1 Hz output of real-time clock" for RTC1Hz output.

[Calculation of correction values]

(Output frequency of the RTC1HZ pin is 0.999817 Hz)

Oscillation frequency=32768× 0.9999817≈32767.4Hz

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.

Correction value =1 minute correction count value= ÷(oscillation frequency ÷ target frequency-1) ×32768×60
= (32767.4÷32768-1) ×32768 ×60
=-36

[(Calculation of F12~F0) settings]

(Case of correction = -36)

F12=1 because the correction value is less than 0 (for faster cases). The correction values are calculated (F11 to F0).

-{(/F11~/F0)-1}×2=-36
(/F11~/F0)=17
(/F11~/F0)=(0,0,0,0,0,0,1,0,0,0,1)
(F11~F0)=(1,1,1,1,1,1,1,0,1,1,1,0)

Therefore, from 32767.4Hz to 32768Hz (32767.4Hz+18.3ppm), the following is true:

If the correction register is set by DEV=1 and correction value =-36 (bit12~0: 1,1,1,1,1,1,1,1,0,1,1,1,0) of the SUBCUD register, 32768Hz (0ppm).

Chapter 9 15-Bit Interval Timer

9.1 Function of 15-bit interval timer

An interrupt (INTIT) is generated at any time interval set in advance, which can be used for arousal from deep sleep mode.

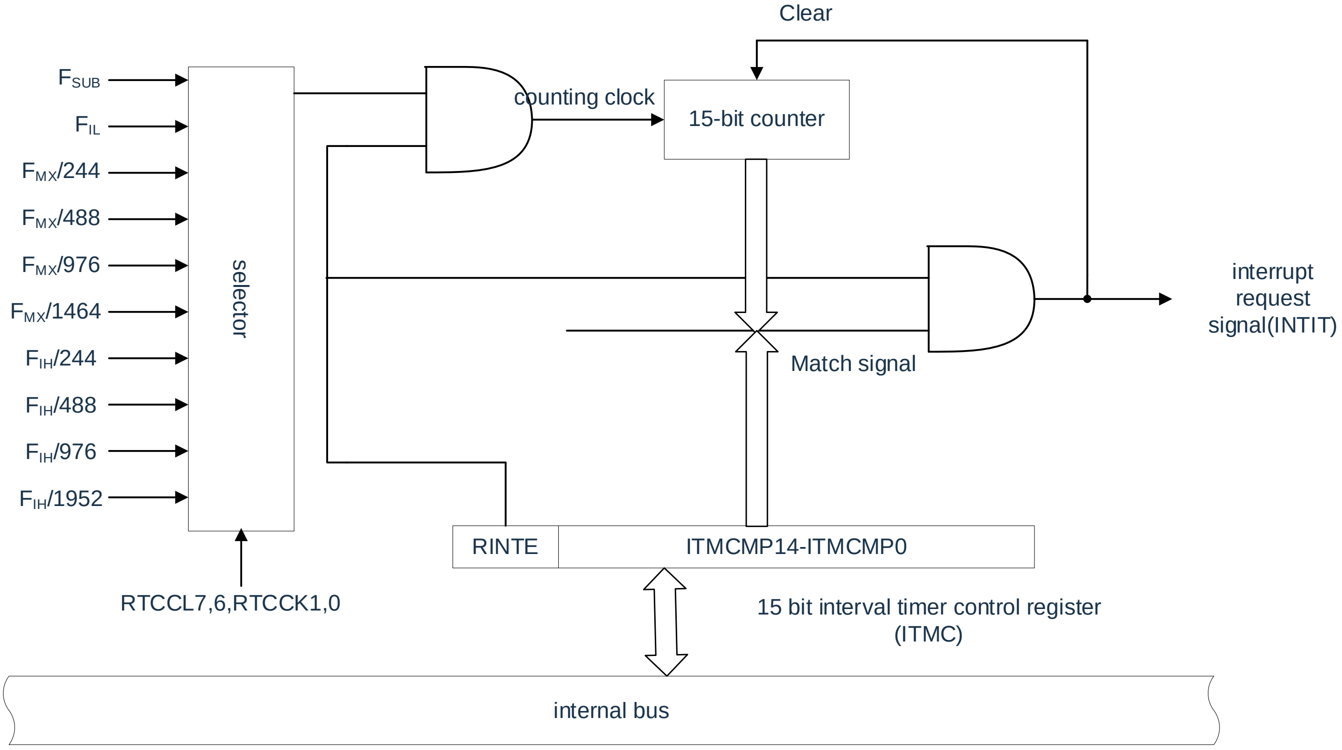
9.2 Structure of 15-bit interval timer

The 15-bit interval timer is composed of the following hardware.

Table 9-1 Structure of 15-bit interval timer

Item	Structure
counter	15-bit counter
control register	Peripheral enable register 0 (PER0).
	Real-time clock selection register (RTCCL)
	15-bit interval timer control register (ITMC)

Figure 9-1 Block diagram of 15-bit interval timer



9.3 Registers for controlling 15-bit interval timer

The 15-bit interval timer is controlled by the following registers.

- Peripheral enable register 0 (PER0).
- Real-time clock selection register (RTCCL)
- 15-bit interval timer control register (ITMC)

9.3.1 Peripheral enable register 0 (PER0)

The PER0 register is a register that sets a clock that is allowed or prohibited to supply to each peripheral hardware. Reduce power consumption and noise by stopping clock supply to unused hardware.

When using a 15-bit interval timer, bit7 (RTCEN) must be set to "1". The PER0 register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "00H".

Figure 9-2 Format of peripheral enable register 0 (PER0)

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

symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	XX	XX	XX	XX	XX	XX	XX

RTCEN	Control of an input clock of a real-time clock (RTC) and a 15-bit interval timer
0	Stop provide an input clock. •You cannot write the SFR used by the real-time clock (RTC) and 15-bit interval timers. •The real-time clock (RTC) and the 15-bit interval timer are reset.
1	Provides an input clock. •SFRs that can read and write real-time clocks (RTCs) and 15-bit interval timers.

9.3.2 Real-time clock selection register (RTCCL)

A real-time clock and a counter clock (fRTC) of a 15-bit interval timer can be selected through RTCCL.

Figure 9-3 Format of real-time clock selection register (RTCCL)

Address: 0x4002047C After reset: 00H R/W

RTCCL

RTCCL7	RTCCL6	RTCCL5	0	0	0	RTCCKS1	RTCCKS0

RTCCL7	Selection of Clock Source for Real-time Clock and Counter Clock of 15-bit Interval Timer
0	Select a high speed system clock (fMX)
1	Select a high speed internal oscillator (fhoco)

RTCCKS1	RTCCKS0	RTCCL6	RTCCL5	Selection of Running Clock of Real-time Clock and Counter Clock of 15-bit Interval Timer
0	0	x	x	Secondary System Clock (fSUB)
0	1			Low-speed internal oscillator clock (fIL) (must set WUTMMCK0 to 1)
1	0	0	1	Main clock fmax/fhoco (via RTCCL7 selection)/1952
1	0	0	0	Main clock fmax/fhoco (via RTCCL7 selection)/1464
1	0	1	0	Main clock fmax/fhoco (via RTCCL7 selection)/976
1	1	0	0	Main clock fmax/fhoco (via RTCCL7 selection)/488
1	1	1	0	Main clock fmax/fhoco (via RTCCL7 selection)/244

9.3.3 15-bit interval timer control register (ITMC)

This is a register that sets the start and stop of the 15-bit interval timer and compares the values. The ITMC register is set by a 16-bit memory operation instruction.

After the reset signal is generated, the value of this register changes to "7FFFH".

Figure 9-4 Format of 15-bit interval timer control register (ITMC)

Address: 0x40044F50 After reset: 7FFFH R/W

symbol	15	14~0
ITMC	RINTER	ITCMP14˜ITCMP0

RINTER	Operation control of 15-bit interval timer
0	Stop the counter from running (clear count).
1	Start the counter running.

ITCMP1 4~ITCMP0	15-bit interval timer comparison value setting
001H	These bits generate a fixed cycle interrupt for the "count clock cycle × (ITCMP set value +1)".
· The · The · The	
7 FFFH	
0000H	Disable from setting.
Example of interrupt period when ITCMP14~ITCMP0 is "0001H" or "7FFFH" ·ITCMP14~ITCMP0=0001H, count clock:fSUB=32.768kHz 1/32.768 [kHz]×(1+1)=0.06103515625 [ms]≈61.03 [μs] • ITCMP14~ITCMP0=7FFFH, count clock: fSUB=32.768kHz 1/32.768 [kHz]× (32767+1)=1000 [ms]	

Note:

1. When changing the RINTE bit from "1" to "0", you must override by setting INTIT to disable interrupt handling through the interrupt mask register. To restart (from "0" to "1"), you must set to allow interrupt handling after clearing the ITIF flag.
2. The read value of the RINTE bit is reflected after setting the 1 count clock of the RINTE bit.
3. After transferring from sleep mode to normal run mode, if ITMC register is to be set and transferred to sleep mode again, it must be transferred to sleep mode after confirming write value.
4. To change the setting of the ITCMP14~ITCMP0 bit, you must do it in the state with the RINTE bit "0".
5. However, it is possible to change the RINTE bit from "0" to "1" or from "1" to "0" while changing the setting of the ITCMP14 to ITCMP0 bits.

9.4 Operation of 15-bit interval timer

9.4.1 Operation timing of 15-bit interval timer

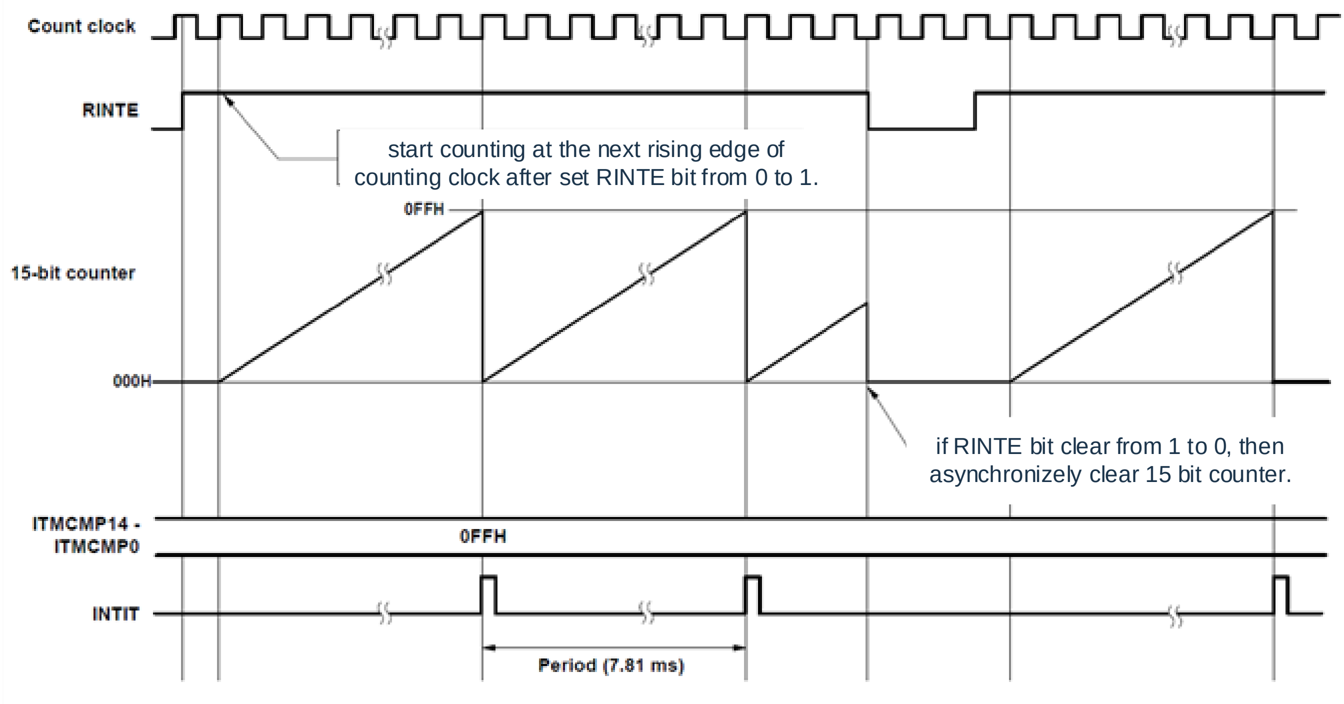
A 15-bit interval timer for repeated generation of interrupt requests (INTIT) is operated at intervals of ITCMP14~ITCMP0 bits. If the RINTE bit is set to "1", the 15-bit counter starts counting.

When the 15-bit count value is equal to the set value of the ITCMP14~ITCMP0 bit, the 15-bit count value is cleared '0' and continues to count, and an interrupt request signal (INTIT) is generated.

The basic operation of the 15-bit interval timer is as follows Figure

Figure 9-5 Operation timing of 15-bit interval timer

(ITCMP14~ITCMP0=0FFH, count clock: f_{SUB}=32.768kHz)

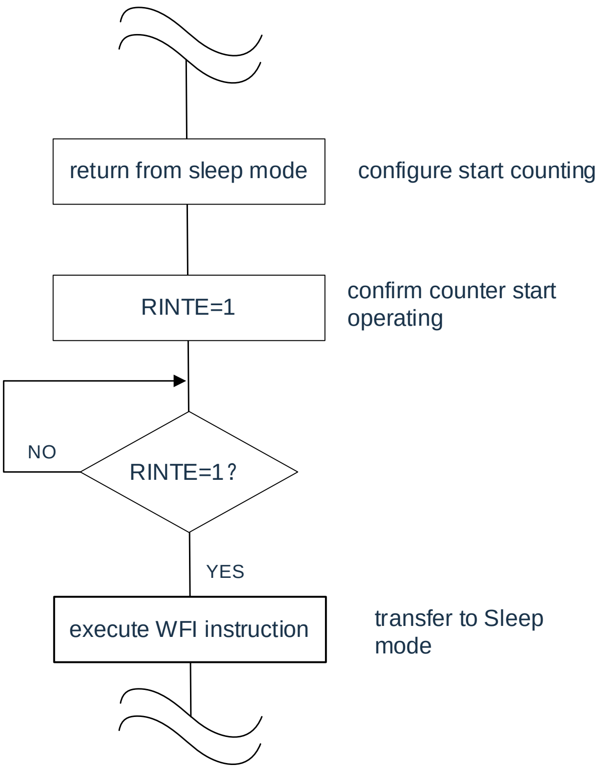

Count clock
RINTE
start counting at the next rising edge of counting clock after set RINTE bit from 0 to 1.
0FFH
15-bit counter
000H
if RINTE bit clear from 1 to 0, then asynchronizely clear 15 bit counter.
ITCMP14 - ITCMP0
0FFH
INTIT
Period (7.81 ms)

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

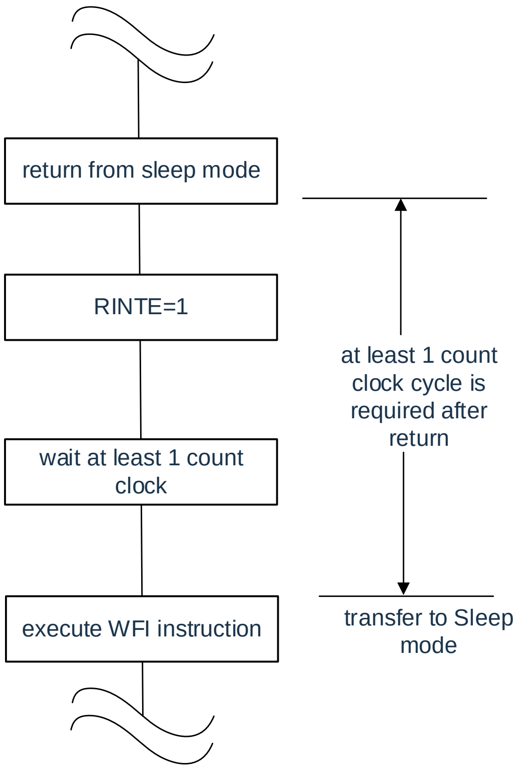
To set the RINTE bit to "1" and transfer to sleep mode again after returning from sleep mode, you must confirm that the write value of the RINTE bit is reflected after setting the RINTE bit to "1" or transfer to sleep mode after at least one count clock time has elapsed since returning.

- After setting the RINTE bit to "1", confirm the RINTE bit to "1" by polling, and then transfer to sleep mode (refer to Example 1 in the figure below).
- Transfer to sleep mode after setting the RINTE bit to "1" for at least 1 count clock (refer to example 2 in the figure below).

Example 1



Example 2



return from sleep mode
RINTE=1
at least 1 count clock cycle is required after return
wait at least 1 count clock
execute WFI instruction
transfer to Sleep mode

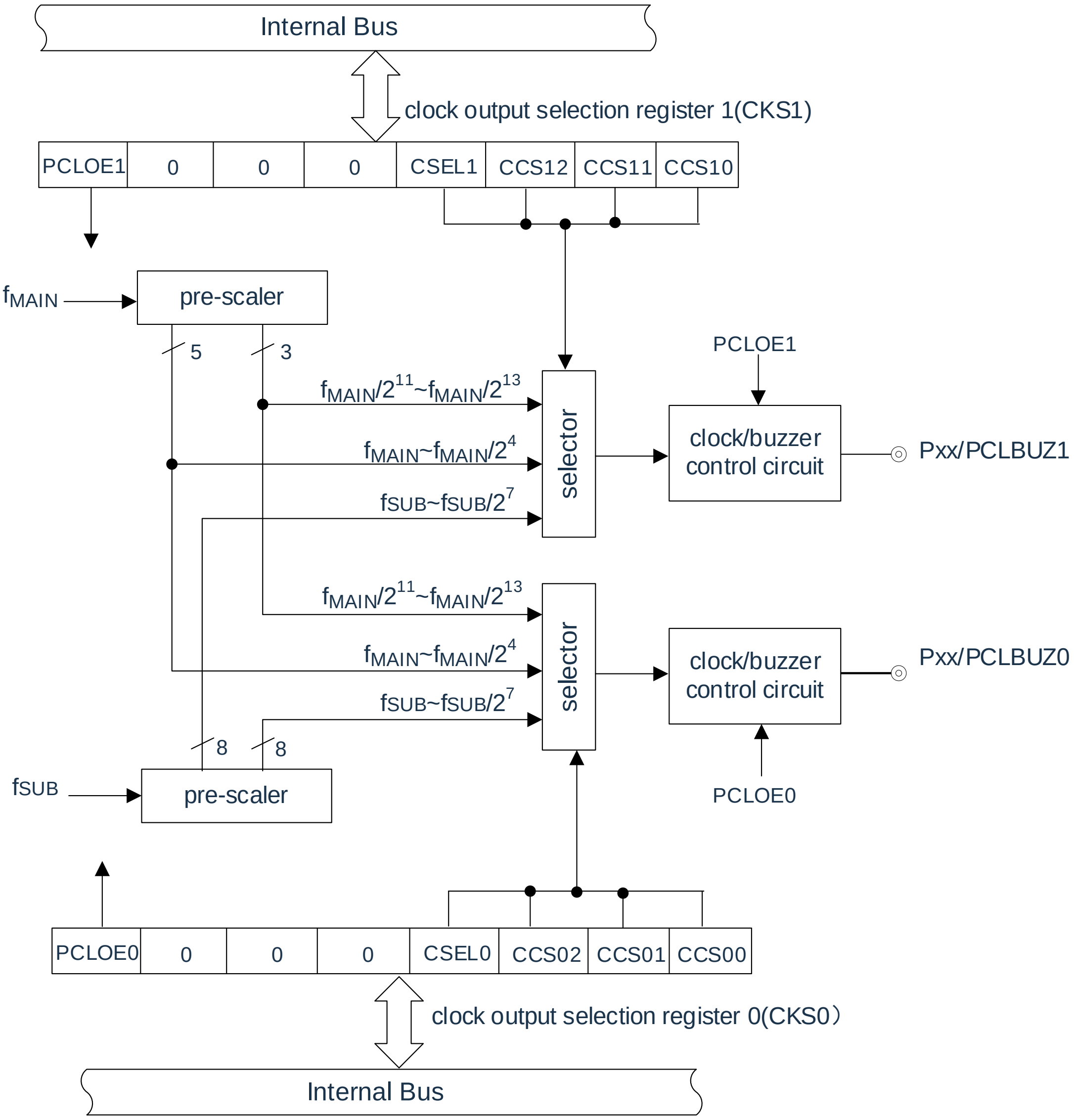
Chapter 10 Clock Output/Buzzer Output Control Circuit

10.1 Function of clock output/buzzer output control circuit

The output of the clock is the function of output to the peripheral IC clock, and the output of the buzzer is the function of output the frequency square wave of the buzzer.

The product has two clock output/buzzer output pins CLKBUZ0 and CLKBUZ1. The CLKBUZn pin outputs the clock selected by the clock output selection register n (CKSn). The block diagram of the clock output/buzzer output control circuit is shown in Figure 10-1. (n=0, 1)

Figure 10-1 Block diagram of clock output/buzzer output control circuit



Note: Refer to "AC Characteristics of the datasheet" for frequencies that can be exported from the CLKBUZ0 and CLKBUZ1 pins.

Notice: The subsystem clock (f_{SUB}) cannot be output from the CLKBUZn pin when the RTCLPC bit of the subsystem clock supply mode control register (OSMC) is "1" and in SLEEP mode where the CPU is running with the subsystem clock (f_{SUB}).

10.2 Structure of clock output/buzzer output control circuit

The clock output/buzzer output control circuit is composed of the following hardware.

Table 10-1 Register for clock output/buzzer output control circuit

Item	Register list
Control register	Clock output selection register n (CKSn) Port mode control register (PMCxx), Port Mode Register (PMxx), Port multiplexing control register (PxxCFG)

10.2.1 Clock output selection register n (CKSn)

This is a register that allows or disables the output of the clock output pin or CLKBUZn and sets the output clock.

Select the clock output from the CLKBUZn pin through the CKSn register. The CKSn register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "00H".

Figure 10-2 Format of clock output selection register n(CKSn)

Addresses: 0x40040FA5 (CKS0), 0x40040FA6 (CKS1) After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
CKSn	PCLOEn	0	0	0	CSELn	CCSn2	CCSn1	CCSn0

PCLOEn	CLKBUZn pin output enable/disable assignments
0	Disable output (default).
1	Enable output.

CSELn	CCSn2	CCSn1	CCSn0	Selection of CLKBUZn Pin Output Clock
0	0	0	0	f_{MAIN}
0	0	0	1	$f_{MAIN}/2$
0	0	1	0	$f_{MAIN}/2^2$
0	0	1	1	$f_{MAIN}/2^3$
0	1	0	0	$f_{MAIN}/2^4$
0	1	0	1	$f_{MAIN}/2^{11}$
0	1	1	0	$f_{MAIN}/2^{12}$
0	1	1	1	$f_{MAIN}/2^{13}$
1	0	0	0	f_{SUB}
1	0	0	1	$f_{SUB}/2$
1	0	1	0	$f_{SUB}/2^2$
1	0	1	1	$f_{SUB}/2^3$
1	1	0	0	$f_{SUB}/2^4$
1	1	0	1	$f_{SUB}/2^5$
1	1	1	0	$f_{SUB}/2^6$
1	1	1	1	$f_{SUB}/2^7$

Note: The output clock must be used in a range of less than 16 MHz. Refer to "AC Characteristics" of the datasheet for details.

Notice:

1. The output clock must be switched after it is set to disable output (PCLOEn=0).
2. When selecting the main system clock (CSELn=0), if you want to transfer to deep sleep mode, you must set the PCLOEn to "0" before executing WFI; When selecting a sub-system clock (CSELn=1), the PCLOEn can be set to '1' because the RTCLPC bit of the OSMC is '0'.
3. The secondary system clock (fSUB) cannot be output from the CLKBUZn pin in a sleep mode in which the RTCLPC bit of the OSMC is 1.

Remark:

1. n=0,1
2. f_{MAIN}: main system clock frequency
   f_{SUB}: sub-system clock frequency

10.2.2 Registers for controlling clock output/buzzer output pin port

This product can multiplex the clock output/buzzer output function CLKBUZ0/CLKBUZ1 into the port. To use the clock output/buzzer output multiplexing to function, you need to set the port register (Pxx), port mode register (PMxx), port mode control register (PMCxx) and port multiplexing function configuration register (PxxCFG).

The corresponding bits of the Port Register (Pxx), Port Mode Register (PMxx) and Port Mode Control Register (PMCxx) must be set to "0" for a multiplexed port configured as a clock output/buzzer output. For details, please refer to "Chapter 2 Port Function".

10.3 Operation of clock output/buzzer output control circuit

It can be use as clock output or buzzer output with 1 pin selection.

The CLKBUZ0 pin outputs a clock/buzzer selected by the clock output selection register 0 (CKS0).

The CLKBUZ1 pin outputs a clock/buzzer selected by the clock output selection register 1 (CKS1).

• Operation of output pin

The CLKBUZn pin follows the steps below to output:

Set the bits of Port Register (Pxx), Port Mode Register (PMxx) and Port Mode Control Register (PMCxx) corresponding to the port used as CLKBUZn pin to "0". Set the Port Multiplexing Function Configuration Register (PxxCFG).

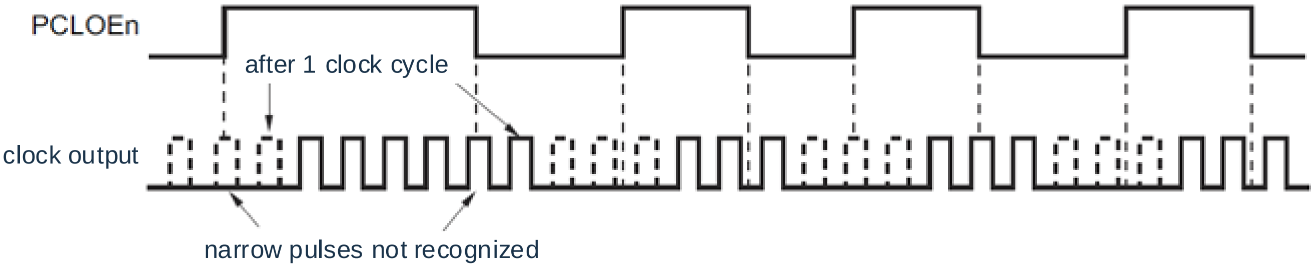
The output frequency (output is forbidden) is selected by bit0~3(CCSn0~CCSn2, CSELn) of clock output selection register (CKSn) of CLKBUZn pin.

Set the bit7 (PCLOEn) of the CKSn register to "1" to allow the clock/buzzer output.

Remark:

1. CLKBUZ0 is fixed and multiplexed to the PA00 port. When using CLKBUZ0, it is not necessary to set the port multiplexing function configuration register (PxxCFG).
2. A control circuit for clock output starts or stops clock output after allowing or disabling 1 clock output (PCLOEn bits). At this time, pulses of narrow width are not output. The timing of the output and the clock output by the PCLOEn bit is as followsFigure in the
3. n=0,1

Figure 10-3 Output timing for CLKBUZn pins



10.4 Cautions for clock output/buzzer output control circuit

When the main system clock is selected as the CLKBUZn output (CSELn=0), the output width of the CLKBUZn becomes narrower if the output clock of 1.5 CLKBUZn pins is shifted to the deep sleep mode after setting the stop output (PCLOEn=0).

Chapter 11 Watchdog Timer

11.1 Function of watchdog timer

The watchdog timer runs with the option byte (000C0H) setting count. The watchdog timer operates at a low speed internal oscillator clock (f_{IL}). The watchdog timer is used to detect program runaway. An internal reset signal is generated when it is detected that the program is out of control.

The following circumstances were determined as out of control of the procedure.

- When the watchdog timer counter overflows
- When the bit operation instruction is executed on the WDTE of the watchdog timer
- When writing data other than "ACH" to the WDTE register
- Write data to the WDTE register during window closure

When a reset occurs due to the watchdog timer, the bit4(WDTRF) of the reset control flag register (RESF) is set "1". For more information on RESF registers, refer to Chapter 28 Reset Features. Interval interruptions can be generated when the overflow time of 75%+1/2 f_{IL} is reached.

11.2 Structure of watchdog timer

The watchdog timer consists of the following hardware.

Table 11-1 Structure of watchdog timer

Project	structure
counter	Internal Counter (17-bit)
control register	Watchdog timer enable register (WDTE)

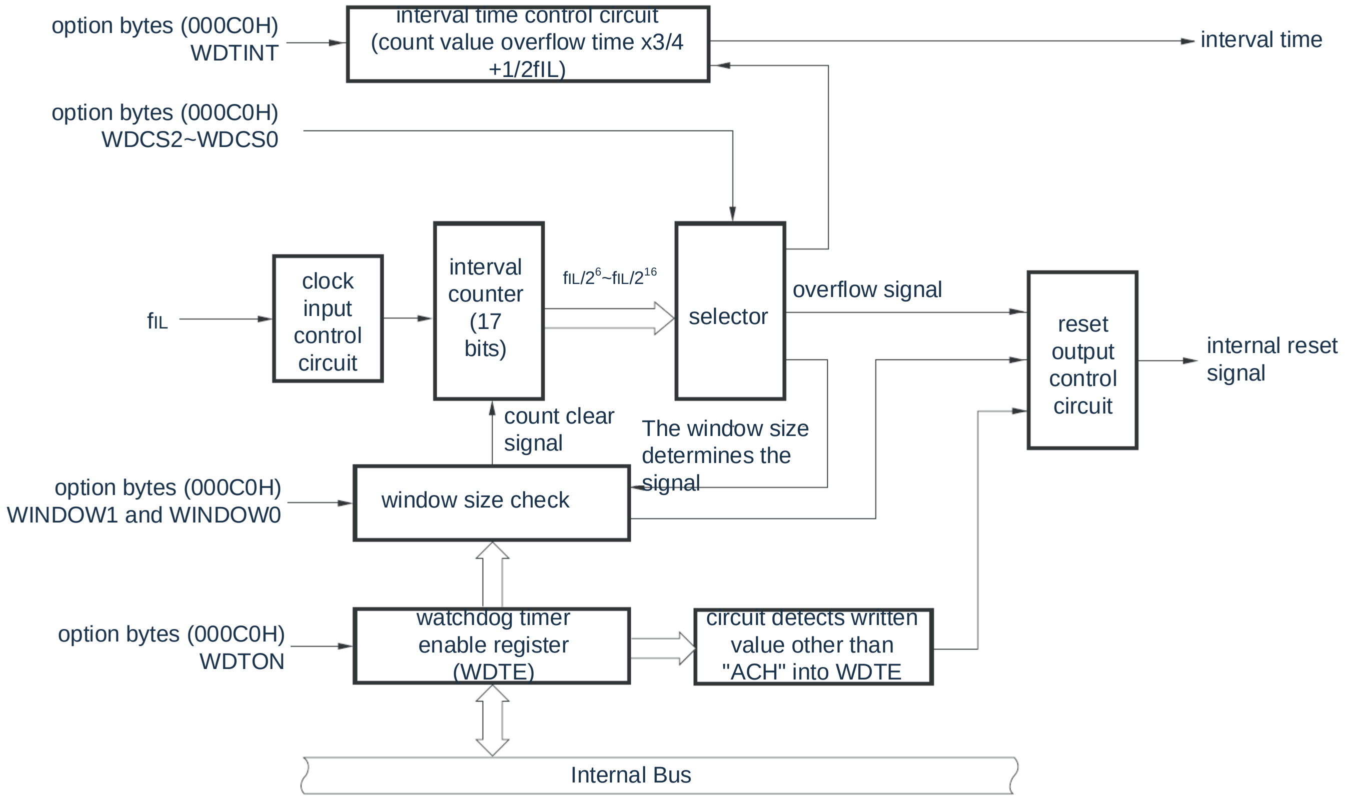
The option bytes control the operation of the counter and the setting of the overflow time, window open period and interval interrupts.

Table 11-2 Option bytes and the settings for the watchdog timer

Setting content of watchdog timer	Option Bytes (000C0H)
Setting of interval interrupt for watchdog timer	bit7 (WDTINT)
Settings during window opening	bit6 and bit5 (WINDOW1, WINDOW0)
Counter Operation Control of Watchdog Timer	bit4 (WDTON)
Setting of overflow time of watchdog timer	bit3~1 (WDCS2~WDCS0)
Counter operation control of watchdog timer (during sleep)	bit0 (WDSTBYON)

Note: For option bytes, refer to "Chapter 33 Option Bytes".

Figure 11-1 Watchdog timer block diagram



Note: f_{IL}: Clock frequency of low speed internal oscillator

11.3 Registers for controlling watchdog timer

The watchdog timer is controlled by an allowable register (WDTE) of the watchdog timer.

11.3.1 Watchdog timer enable register (WDTE)

By writing "ACH" to the WDTE register, the watchdog timer's counter is cleared and counting restarts. The WDTE register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to '9AH' or '1AH' note.

Figure 11-2 Format of watchdog timer enable register (WDTE)

Address: 0x40021001 After Reset: 9AH/1AH Note R/W

symbol	7	6	5	4	3	2	1	0
WDTE								

Note: The reset value of the WDTE register varies depending on the set value of the WDTON bit in the option byte (000C0H). For the watchdog timer to operate, the WDTON bit must be set to "1".

Set value for WDTON bit	Reset value for WDTE register
0 (Disable the watchdog timer to count)	1AH
1 (Enable the watchdog timer to count)	9AH

Note:

1. An internal reset signal is generated when a value other than "ACH" is written to the WDTE register.
2. An internal reset signal is generated when a bit operation instruction is executed on the WDTE register.
3. The read value of the WDTE register is "9AH/1AH" (different from "ACH").

11.3.2 LOCKUP control register (LOCKCTL) and its protection register (PRCR)

The LOCKCTL register is the configuration register for whether the Cortex-M0+ LockUp feature causes the watchdog timer to run, and the PRCR is its write-protected register.

Set the LOCKCTL, PRCR register via 8-bit memory operation instructions.

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

Figure 11-3 Format of LOCKUP control register (LOCKCTL)and its protection register (PRCR) (1/2)

Address: 40020405H After reset: 00H R/W

LOCKCTL								
	0	0	0	0	0	0	0	lockup_rst

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

Figure 11-3 Format of LOCKUP control register (LOCKCTL)and 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	• LOCKCTL registers are not writable
1	• LOCKCTL registers are writable

PRTKEY[7:1]	Write protection for PRCR
78H	• PRCR is writable
Others	• PRCR is not writable

11.4 Operation of watchdog timer

11.4.1 Operation control of watchdog timer

1) When you use the watchdog timer, set the following by 000C0H:
   - The bit4 (WDTON) of option bytes (000C0H) must be set to "1" to allow the watchdog timer to count (after reset, the counter starts) (see Chapter 33 option bytes for details).

WDTON	Counter of watchdog timer
0	Disable counting (stop counting after reset).
1	Allow count to run (count begins after reset is removed).

   - You must set the overrun time by bit3~1 (WDCS2~WDCS0) for option bytes (000C0H) (see for details) 11.4.2 and chapter 33).
   - You must set the window opening period (WINDOW1, WINDOW0) through bit6 and bit5 (,) of option bytes (000C0H) (see for details) 11.4.2 and chapter 33).

2) After the reset is removed, the watchdog timer starts counting.
3) After counting is started and before the overset time set in the option byte, the watchdog timer is cleared and counting starts again if the WDTE of the watchdog timer is written ACH.
4) Thereafter, the write operation of the WDTE register after the second reset must be performed during the window opening period. If that WDTE register is write during window close, an internal reset signal is generate.
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 when:
   - When a bit operation instruction is executed on a WDTE register
   - When writing data other than "ACH" to the WDTE register

Notice

1. Only when writing the WDTE of the watchdog timer for the first time after the reset is released, regardless of window opening.
2. It is possible to generate up to 2 fIL clock errors from writing "ACH" to the WDTE register to clearing the watchdog timer counter.
3. The watchdog timer can be cleared before the count value overflows.
4. As shown below, the watchdog timer runs in sleep or deep sleep mode differently depending on the setting value of bit0 (WDSTBYON) for option bytes (000C0H).

	WDSTBYON=0	WDSTBYON=1
sleep mode deep sleep mode	Stop the watchdog timer from running.	Keep the watchdog timer running.

When the WDSTBYON bit is '0', the watchdog timer is counted again after the sleep or deep sleep mode is canceled. At this point, clear the counter "0" and start counting.

When the deep sleep mode is released and the CPU is operated with an X1 oscillating clock, the CPU starts to operate after an oscillating steady time.

If that time from the deepsleep mode to the overturn of the watchdog timer is short, the overturn of the watchdog will occur within the oscillation stable time and reset. Therefore, after releasing the deep sleep mode through interval interruption, the watchdog timer is to be operated and cleared with the X1 oscillating clock.

11.4.2 Setting of overflow time of watchdog timer

Set the overflow time of the watchdog timer by bit3~1(WDCS2~WDCS0) of the option byte (000C0H).

An internal reset signal is generated when an overturn occurs. If the "ACH" is written to the WDTE of the watchdog timer during the window opening before the overage time, the count is cleared and counting restarts. The overflow times that can be set are shown below.

Table 11-3 Setting of overflow time of watchdog timer

WDCS2	WDCS1	WDCS0	Overflow time of watchdog timer (Case of fIL=20kHz(MAX.))
0	0	0	$2^6/f_{IL}$(3.2ms)
0	0	1	$2^7/f_{IL}$(6.4ms)
0	1	0	$2^8/f_{IL}$(12.8ms)
0	1	1	$2^9/f_{IL}$(25.6ms)
1	0	0	$2^{11}/f_{IL}$(102.4ms)
1	0	1	$2^{13}/f_{IL}$(409.6ms)
1	1	0	$2^{14}/f_{IL}$(819.2ms)
1	1	1	$2^{16}/f_{IL}$(3276.8ms)

Note: f_{IL}: The clock frequency of a low-speed internal oscillator.

11.4.3 Watchdog timer window settings during opening

Set the window opening period of the watchdog timer by bit6 and bit5 (WINDOW1, WINDOW0) of option bytes (000C0H). The window summary is as follows:

- If a watchdog timer's enable register (WDTE) is written "ACH" during window opening, the watchdog timer is cleared and counting restarts.
- During window closing, even if the WDTE register is written "ACH", an exception is detected and an internal reset signal is generated.

Note: Only when writing WDTE register for the first time after reset is released, the watchdog timer is cleared and counting is resumed, regardless of window opening.

The window opening period that can be set is shown below.

Table11-4 Watchdog timer window settings during opening

WINDOW1	WINDOW0	Watchdog timer window opening period
0	-	Disable from setting
1	0	75%
1	1	100%

Note: When the option byte (000C0H) has bit0 (WDSTBYON) of "0", it is independent of the WINDOW1 bit and WINDOW0 bit values, and the window is 100%.

Note: When setting the overspill time to ^{29}/fIL, the window close time and open time are shown below.

	Settings during window opening	
	75%	100%
Window Close Time	0~12.8 ms	None
Window Open Time	12.8~25.6ms	0~25.6 ms

< When window open period is 75%>

- Overflow time:
- $2^9/f_{IL}$(MAX.)=2^9/20kHz(MAX.)=25.6ms
- Window closed:
- 0～$2^9/f_{IL}$(MIN.)×(1–0.75)=0～2^9/10kHz×0.25=0～12.8ms
- Window opened:
- $2^9/f_{IL}$(MIN.)×(1–0.75)～$2^9/f_{IL}$(MAX.)=12.8～25.6ms

11.4.4 Setting of watchdog timer interval interrupt

Interval interrupts (INTWDTI) can be generated when 75%+1/2f_{IL} is reached by setting bit7 (WDTINT) of option bytes (000C0H).

Table 11-5 Setting of watchdog timer interval interrupt

WDTINT	Watchdog timer interval interrupt use/no use
0	Interrupt without interval.
1	Interval interruptions occur when the overset time reaches 75%+1/2 f_{IL}.

Note: When the deep sleep mode is released and the CPU is operated with an X1 oscillating clock, the CPU starts to operate after an oscillating steady time.

If that time from the deepsleep mode to the overturn of the watchdog timer is short, the overturn of the watchdog will occur within the oscillation stable time and reset. Therefore, when the deep sleep mode is released by interval interruption, the watchdog timer is to be operated and cleaned by X1 oscillating clock.

Remark: Counting continues even after the generation of the INTWDTI (continues until the "ACH" is written to the WDTE of the watchdog timer). If that" ACH" is not write to the WDTE register before the overrun time, an internal reset signal is generate.

11.4.5 Operation of watchdog timer during LOCKUP

When the lockup_rst bit of the LOCKUP control register LOCKCTL is set to 1, once the core enters the LOCKUP state, the low-speed internal oscillator begins to oscillate, the watchdog timer's timer automatically starts running, and the control bit of the overflow time (WDCS2~WDCS0) is set to 3'b010, which means that the overflow time is set to 12.8ms.

Chapter 12 Comparator

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

12.1 Function of comparator

The comparator has the following functions:

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

Table12-1 Comparator Feature Summary

Item	Content
CMP	• 2-channel comparators (CMP0 and CMP1) • The negative end of the comparator can select a reference voltage: Optional external pin input on the negative side of the CMP0, built-in reference voltage of CMP0 and internal reference voltage (1.45V) Optional external pin input (4) for the negative end of the CMP1, built-in reference voltage of CMP1 and internal reference voltage (1.45V) • The internal reference voltage of the negative end may be set (256 steps) • The front end of the CMP0 selects the output of the PGA • The front end of the CMP1 can select external pin inputs (4) • When the input voltage of the positive end > the input voltage of the negative end, the output high level When the input voltage of the positive end is less than the input voltage of the negative end, the output level is low • Filter width of digital filter is optional • output inversion function • The comparison result can be output from the VCOUT0 (VCOUT1) • An effective edge of the comparator output can be detected and an interrupt signal generated • Combined with Timer4 to output TIMER WINDOW • Supports comparator positive hysteresis, negative hysteresis, and bilateral hysteresis with hysteresis voltages of 20mV, 40mV, and 60mV

12.2 Structure of comparator

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

Figure 12-1 Block diagram of comparator 0

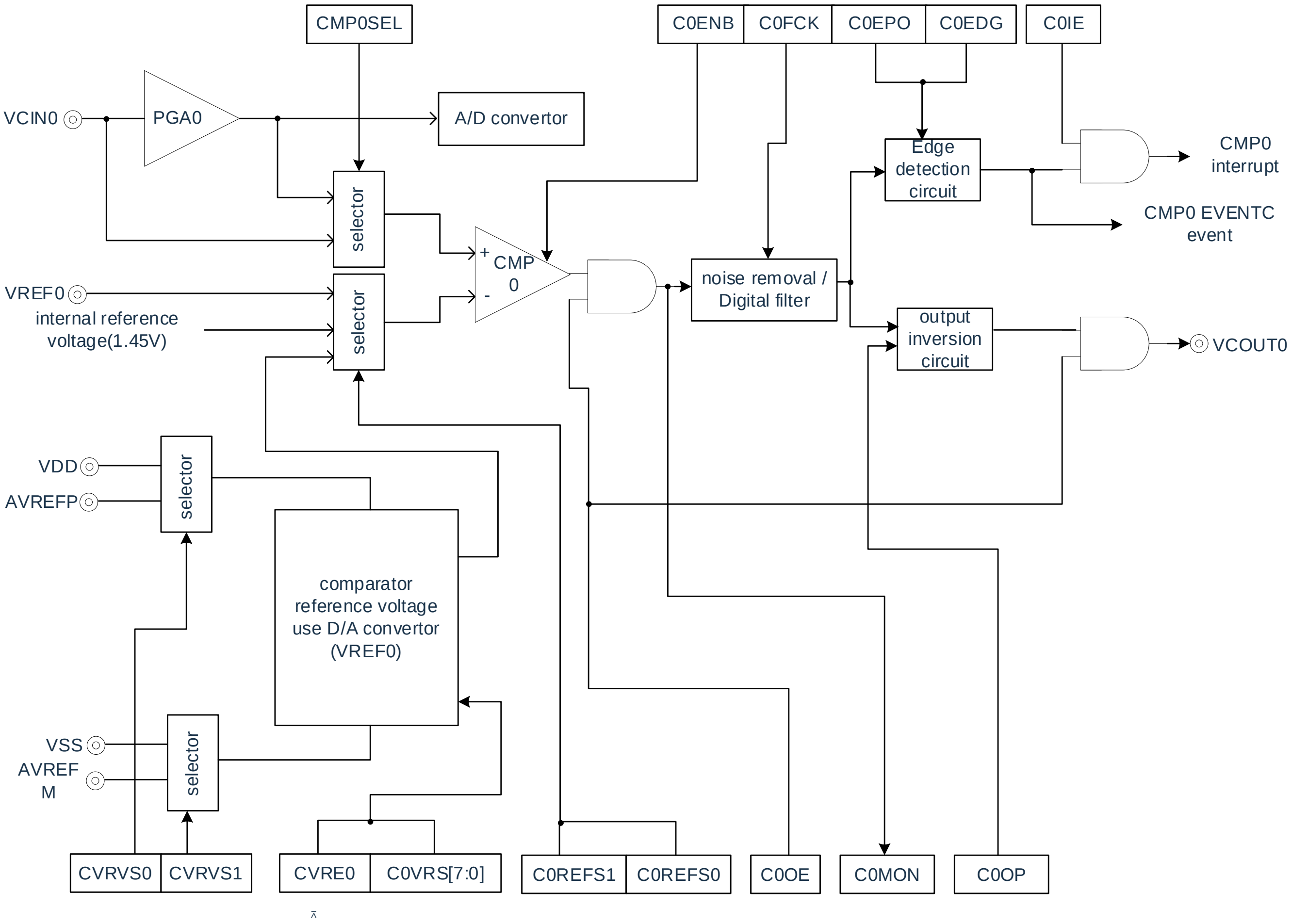
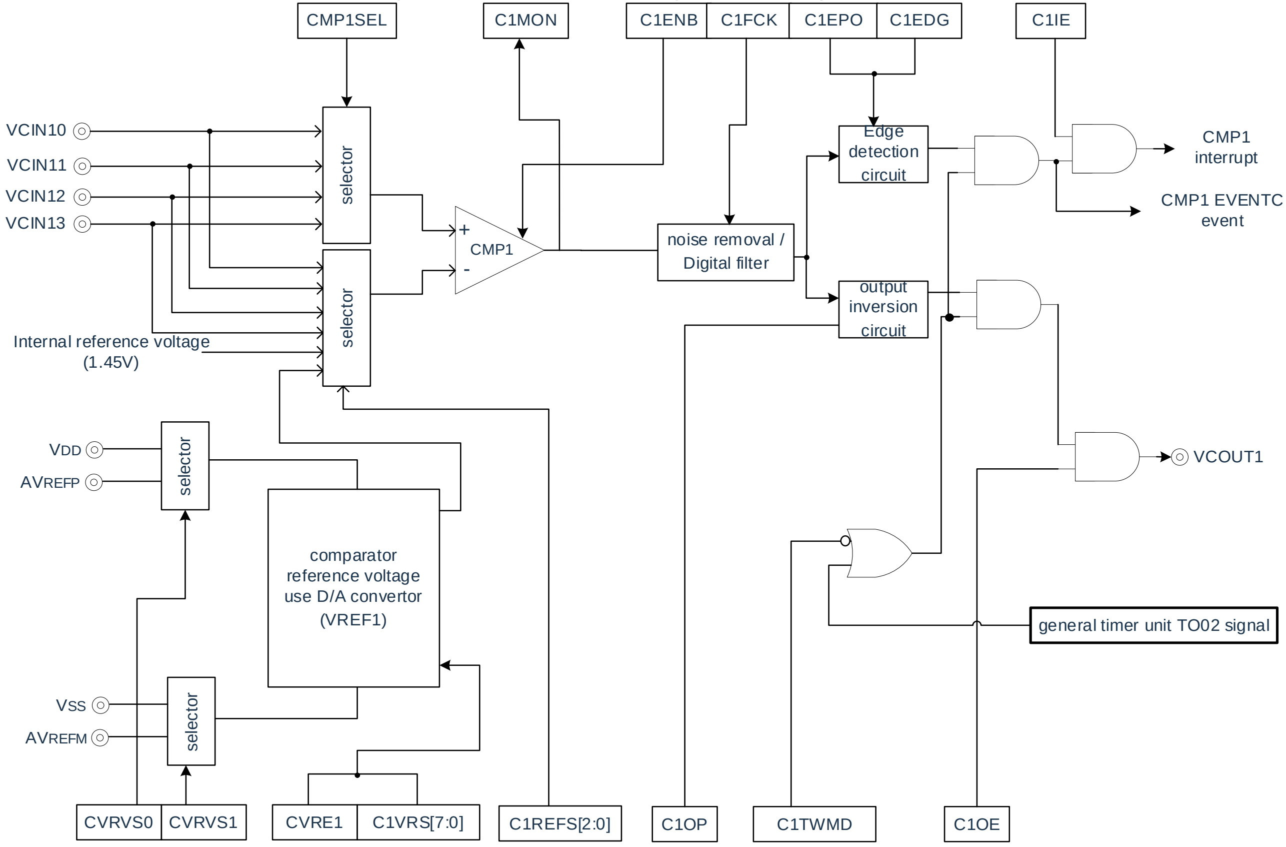


Figure 12-2 Block diagram of comparator 1



12.3 Registers for controlling comparator

The registers controlling the comparator are as follows.

Table 12-2 Registers for control comparator

Register name	Symbol
Peripheral enable register 1	PER1
Comparator mode configuration register	COMPMDR.
Comparator filter control register	COMPFIR
Comparator output control register	COMPOCR
Comparator built-in reference voltage control register	CVRCTL
Comparator built-in reference voltage selection register 0	C0RVM
Comparator built-in reference voltage selection register 1	C1RVM
Comparator 0 input selection control register	CMPSEL0
Comparator 1 input selection control register	CMPSEL1
Comparator 0 hysteresis control register	CMP0HY
Comparator 1 hysteresis control register	CMP1HY
Port mode control register	PMCxx
Port mode register	PMxx

12.3.1 Peripheral enable register 1 (PER1)

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

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

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

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

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

Address: 0x4002081A After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
PER1	XX	XX	PGACMPEN	XX	XX	XX	0	0

PGACMPEN	Control of comparator input clock
0	Stop provide an input clock. • Cannot write the SFR used by the comparator. • The comparator is in a reset state.
1	Provides an input clock. • SFR that can read and write to the comparator.

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

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

12.3.2 Comparator mode configuration register (COMPMDR)

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

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

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

- The CMP negative input selects the built-in reference voltage while the built-in reference voltage action stops (the CVREi bit of the CVRCTL register is "0")
- The input of the CMP0 selects the output of the PGA, and when the PGA action stops (the CMPSEL0 bit of the CVRCTL register is '1' and the PGAEN bit of the PGAEN register is '0')

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

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

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

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

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

C1MON	Comparator 1 Monitor Flag Notes 1, 2
0	VCIN1<Reference voltage of comparator 1, or comparator 1 is not running.
1	VCIN1> Reference voltage for comparator 1

C1ENB	Allow for comparator 1 to run
0	Disables comparator 1 operation.
1	Allow comparator 1 to run.

C0MON	Comparator 0 Monitor Flag Note 1,2
0	VCIN0<Reference voltage of comparator 0, or comparator 0 is not running.
1	VCIN0>Reference voltage of comparator 0

C0ENB	Allow for comparator 0 to run
0	Prevents comparator 0 from running.
1	Allow comparator 0 to run.

Note 1. Immediately after unreset becomes "0" (initial value), if both C0ENB and C1ENB bits are "0" after allowing comparator operation.

2. Ignore the write value for this bit.

12.3.3 Comparator filter control register (COMPFIR)

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

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

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

Address: 400438 41H. After reset: 00H R/W

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

C1EDG	Comparator 1 Edge Detection Selection Note 1
0	An interrupt request is generated by single edge detection of the comparator 1.
1	An interrupt request is generated by bilateral edge detection of the comparator 1.

C1EPO	Comparator 1 Edge Polarity Switching Note 1
0	An interrupt request is generated by the rising edge of the comparator 1.
1	An interrupt request is generated by the descent edge of the comparator 1.

C1FCK1	C1FCK0	Comparator 1 filter Selection Note 1
0	0	Comparator 1 has no filter.
0	1	Comparator 1 has a filter and samples it through the fCLK.
1	0	Comparator 1 has a filter and samples it by fCLK/8.
1	1	Comparator 1 has a filter and samples through fCLK/32.

C0EDG	Comparator 0 Edge Detection Selection Note 2
0	Interrupt requests are generated by single edge detection of comparator 0.
1	Interrupt requests are generated by bilateral edge detection of comparator 0.

C0EPO	Comparator 0 Edge Polarity Switching Note 2
0	An interrupt request is generated by the rising edge of the comparator 0.
1	An interrupt request is generated by the descent edge of the comparator 0.

C0FCK1	C0FCK0	Comparator 0 Filter Selection Note 2
0	0	Comparator 0 has no filter.
0	1	Comparator 0 has a filter, through the fCLK sampling.
1	0	Comparator 0 has a filter, and samples through fCLK/8.
1	1	Comparator 0 has a filter, and samples through fCLK/32.

Note:

1. If C1FCK1~C1FCK0 bits, C1EPO bits and C1EDG bits are changed, interrupt request and event signals output to EVENTC may occur. These bits must be changed after setting the EVENTC ELSELR14 register (output of unlinked comparator 1) to "0". In addition, the IF of the interrupt request flag register must be cleared "0".
   If C1FCK1~C1FCK0 bit is changed from '00B' (comparator1 has filter), the comparator 1 must use the interrupt request or the event signal output to EVENTC after 4 sampling before updating the output of the filter.

2. If you change the C0FCK1~C0FCK0 bits, C0EPO bits, and C0EDG bits, you may generate a interrupt request for comparator 0 and event signals output to EVENTC. These bits must be changed after setting the EVENTC ELSELR13 register (output of unlinked comparator 0) to "0". In addition, the IF of the interrupt request flag register must be cleared "0".

   If C0FCK1~C0FCK0 bit is changed from '00B' (comparator 0 without filter) to other values (comparator 0 with filter).

12.3.4 Comparator output control register (COMPOCR)

The COMPOCR register is a control register that sets the polarity of the comparator output, the permission/prohibition of the output, and the permission/prohibition of the interrupt output.

In the following cases, setting "1" to the CiOE bit of the COMPOCR register is prohibited (output enabled). (i=0,1)

- When the comparator action stops (the CiENB bit of the COMPMDR register is "0")
- The CMP negative input selects the built-in reference voltage while the built-in reference voltage action stops (the CVREi bit of the CVRCTL register is "0")
- The input of the CMP0 selects the output of the PGA, and the PGA action stops (the CMPSEL0 bit of the CVRCTL register is "1" and the PGAEN bit of the PGAEN register is "0")

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

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

Figure 12-6 Format of comparator output control register (COMPOCR)

Address: 40043842H. After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
COMPOCR	C1OTWMD	C1OP	C1OE	C1IE	0	C0OP	C0OE	C0IE

C1OTWMD	Comparator 1 TIMER WINDOW Output Mode Control Bit Note 1
0	Comparator 1 Normal Output Mode (controlled by C1OE)
1	Comparator 1TIMER WINDOW output mode (co-controlled by TO02 and C1OE)

C1OP	Selection of Output Polarity of VCOUT1
0	VCOUT1 is the output of comparator 1.
1	VCOUT1 is the inverted output of comparator 1.

C1OE	Allowable for VCOUT1 Pin Output Note 2
0	Disable VCOUT1 output to pin.
1	Allow VCOUT1 output to pin.

C1IE	Comparator 1 Allow for Interrupt Request Note 3
0	Disables interrupt request for comparator 1.
1	Allow interrupt requests for comparator 1.

C0OP	Selection of Output Polarity of VCOUT0
0	VCOUT0 is the output of comparator 0.
1	VCOUT0 is the inverted output of comparator 0.

C0OE	Allowable for VCOUT0 Pin Output Note 4
0	Disables VCOUT0 output to the pin.
1	Allow VCOUT0 output of to the pin Note 4, 8

C0IE	Comparator 0 Allow for Interrupt Request Note 5
0	Disable interrupt request for comparator 0.
1	Allow interrupt requests for comparator 0.

Note1. When comparator 1 uses the TIMER WINDOW mode, the bit7 (C1EDG) of register COMPFIR must be set to "1". The C1OE and C1OTWMD bits cannot be set at the same time. Set the C1OTWMD bit first, and then set the C1OE bit to "1".

Note2. When the result of comparator 1 is output to the pin, Pxx, PMxx, PMCxx of this pin must be set to 0.

Note3. If C1IE is changed from 0 to 1, the IF of the interrupt request flag register may become 1, so interrupt must be used after clearing IFL.

Note4. When the result of comparator 0 is output to the pin, Pxx, PMxx, PMCxx of this pin must be set to 0.

Note5. If C0IE is changed from 0 to 1, the IF of the interrupt request flag register may become 1, so interrupt must be used after IF.

12.3.5 Comparator built-in reference voltage control register (CVRCTL)

The CVRCTL register is a register that sets the built-in reference voltage permit/stop action of the comparator.

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

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

Note: The CVRVSi bit of the CVRCTL register is overridden when the built-in reference voltage stop action (CVREi=0).

Figure 12-7 Format of comparator built-in reference voltage control register (CVRCTL)

Address: 400438 43H. After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
CVRCTL	0	0	CVRE1	CVRVS1	0	0	CVRE0	CVRVS0

CVRE1	Control bit with built-in reference voltage 1
0	Disable operation of built-in reference voltage 1
1	Allow operation of built-in reference voltage 1

CVRVS1	Ground side selection bit with built-in reference voltage
0	Internal Reference Voltage ground Side Select Vss
1	Internal Reference Voltage ground Side Select AVREFM Note 1

CVRE0	Control bit with built-in reference voltage 0
0	Disable operation of built-in reference voltage 0
1	Allow operation of built-in reference voltage 0

CVRVS0	Power supply side selection bit with built-in reference voltage
0	Power Supply Side Selection VDD with Internal Reference Voltage
1	Internal Reference Voltage Power Supply Side Selection AVREFP Note 2

Note 1: AVREFM and VCIN13 multiplex the same port, so it is forbidden to set CVRVS1 bit to "1" when the port is used as input signal VCIN13 of CMP1.

Note 2: AVR EFP and VCIN12 multiplex the same port, so it is forbidden to set the CVRVS0 bit to "1" when the port is used as the input signal VCIN12 of CMP1.

12.3.6 Comparator built-in reference voltage selection register (CiRVM)

The CiRVM register is a register that sets the built-in reference voltage of the comparator.

When the built-in reference voltage stops (CVREi=0), rewrite the CiRVM register

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

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

Figure 12-8 Format of comparator built-in reference voltage selection register i (CiRVM)

Address: 400438434H (C0RVM), 400438435H (C1RVM), After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
CiRVM	CiRVS7	CiRVS6	CiRVS5	CiRVS4	CiRVS3	CiRVS2	CiRVS1	CiRVS0

CiRVS7	CiRVS6	CiRVS5	CiRVS4	CiRVS3	CiRVS2	CiRVS1	CiRVS0	Setting of built-in reference voltage of comparator
0	0	0	0	0	0	0	0	{(AVREFP or VDD)/256}x0
0	0	0	0	0	0	0	1	{(AVREFP or VDD)/256}x1
0	0	0	0	0	0	1	0	{(AVREFP or VDD)/256}x2
0	0	0	0	0	0	1	1	{(AVREFP or VDD)/256}x3
...								...
1	1	1	1	1	1	0	0	{(AVREFP or VDD)/256}x252
1	1	1	1	1	1	0	1	{(AVREFP or VDD)/256}x253
1	1	1	1	1	1	1	0	{(AVREFP or VDD)/256}x254
1	1	1	1	1	1	1	1	{(AVREFP or VDD)/256}x255

12.3.7 Comparator 0 input signal selection control register (CMPSEL0)

The CMPSEL0 register is a selection register for the input signal of the positive end and the negative end of the comparator 0.

When comparator 0 stops (C0ENB=0), override the CMPSEL0 register.

The CMPSEL0 register is set by an 8-bit memory operation instruction.

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

Figure 12-9 Format of comparator 0 input signal selection control register (CMPSEL0)

Address: 4004384AH After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
CMPSEL0	CMP0SEL	0	0	0	0	0	C0REFS1	C0REFS0

CMP0SEL	Comparator 0 positive input signal selection bit
0	Select External Pin (VCIN0 Pin)
1	Select PGA output signal

C0REFS1	C0REFS0	Comparator 0 negative input signal selection bit
0	0	Select the built-in reference voltage VREF0
0	1	Select Internal Reference Voltage (1.45V)
1	0	Select an external pin (IVREF0 pin)
1	1	Disable settings

12.3.8 Comparator 1 input signal selection control register (CMPSEL1)

The CMPSEL1 register is a selection register for the input signal of the positive end and the negative end of the comparator 1.

When comparator 1 stops (C1ENB=0), rewrite the CMPSEL1 register.

The CMPSEL1 register is set by an 8-bit memory operation instruction.

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

Figure 12-10 The format of the input signal selection control register (CMPSEL1) of comparator 1

Address: 4004384BH After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
CMPSEL1	CMP1SEL1	CMP1SEL0	0	0	0	C0REFS2	C0REFS1	C0REFS0

CMP1SEL1	CMP1SEL0	Comparator 1 positive input signal selection bit
0	0	Select External Pin (VCIN10 Pin)
0	1	Select External Pin (VCIN11 Pin)
1	0	Select External Pin (VCIN12 Pin)
1	1	Select External Pin (VCIN13 Pin)

C0REFS2	C0REFS1	C0REFS0	Negative input signal selection bit of comparator 1
0	0	0	Select the built-in reference voltage VREF1
0	0	1	Select Internal Reference Voltage (1.45V)
0	1	0	Select External Pin (VCIN10 Pin)
0	1	1	Select External Pin (VCIN11 Pin)
1	0	0	Select External Pin (VCIN12 Pin)
1	0	1	Select External Pin (VCIN13 Pin)
1	1	0	Disable from setting
1	1	1	Disable from setting

Note: When switching the analog input of the CMP1, the switching interval must be more than 3 us in order to prevent the through current before the two input signals.

12.3.9 Comparator 0 hysteresis control register (CMP0HY)

The CMP0HY register is the hysteresis function control register for comparator 0.

The CMP0HY register is rewritten when comparator 0 stops operating (C0ENB=0).

The CMP0HY register is set by 8-bit memory operation instructions.

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

Figure 12-11 Format of comparator 0 hysteresis control register (CMP0HY)

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	Hysteresis voltage selection bit for comparator 0
0	0	No hysteresis
0	1	20mV
1	0	40mV
1	1	60mV

C0HYSLS1	C0HYSLS0	Hysteresis mode selection bit for comparator 0
0	0	No hysteresis
0	1	Positive Hysteresis
1	0	Negative hysteresis
1	1	Bilateral hysteresis

12.3.10 Comparator 1 hysteresis control register (CMP1HY)

The CMP1HY register is the hysteresis function control register for comparator 1.

The CMP1HY register is rewritten when Comparator 1 stops operation (C1ENB=0).

The CMP1HY register is set by an 8-bit memory operation instruction.

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

Figure 12-12 Format of comparator 1 hysteresis control register (CMP1HY)

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	Hysteresis voltage selection bit for comparator 1
0	0	No hysteresis
0	1	20mV
1	0	40mV
1	1	60mV

C1HYSLS1	C1HYSLS0	Hysteresis mode selection bit for comparator 1
0	0	No hysteresis
0	1	Positive Hysteresis
1	0	Negative hysteresis
1	1	Bilateral hysteresis

12.3.11 Registers for controlling analog input pin port function

When using the VCIN0 pin, VCIN10-VCIN1 3 pin and VREF0 pin as analog inputs to the comparator, the bits of port mode register (PMxx) and PMCxx.

When using the functions of VCOUT0 and VCOUT1, port register (Pxx), port mode register (PMxx) and port mode control register (PMCxx) must be set. Please refer to Chapter 2 Port Function for details.

12.4 Operation Instructions

Comparator 0 and Comparator 1 can be run independently.The CMP0 and PGA0 can be combined together. The set-up steps for the comparator's independent operation and interactivity are as below

Table12-3 Set-up steps for comparator-related registers

Step	Register	Bit	Set value
1	PGACTL	PGAVG0/1/2	Select gain [Note 3]
2	PGACTL	PVRVS	0 (Vss pin selection) [Note 3]
3	PGACTL	PGAEN	1 (Enable operation) [Note 3]
4	Wait for PGA stability time (minimum 10μs)		
5	COMPSELi	CMP0SEL/CMP1SELi1	comparator i positive input selection
6	COMPSELi	CiREFS	comparator i negative input selection
7	CiRVM	CiRVSn	Set the value of the built-in reference voltage
8	CVRCTL	CVRVSi	Select power and ground with built-in reference voltage
9	CVRCTL	CVREi	1 (built-in reference voltage i allowed to run)
10	Wait for stability time of reference voltage (minimum 20μs)		
11	Set VCIN0, VCIN1x, IVREF0 pin (input), PGAI [Note 3] to the analog input function. Function selection for PMCxx, VCIN0 pins, VCi and IVREFi pins. Set the PMCxx bit to "1" (analog input). Set the PMxx bit to "1" (input mode).		
12	COMPMDR.	CiENB	1 (Enable operation)
13	Wait for the stability time of the comparator (minimum 3μs)		
14	COMPFIR	CiFCK	Select a sampling clock using or without a digital filter.
		CiEOP, CiEDG	Select the edge detection criteria (rising, falling, or bilateral) for the interrupt request.
15	COMPOCR	CiOP, CiOE	Sets the output of the VCOUTi (select Polarity, set enable or disable output). Refer to "12.4.4 Output of comparator i (i=0,1)".
		CiIE	Sets the output that allows or disables the interrupt request. Refer to "12.4.2 Comparator i interrupts (i=0,1)".
		C1OTWMD	Set TIMER WINDOW output enable/disable for Comparator 1
16	MKxx [Note 1]	MKL	When using interrupts: select mask interrupt.
17	IFxx [Note 1]	IFL	When using interrupts: 0 (non-disruptive request: initialization) [Note 2]

Note 1. MKxx,IFxx is the comparator interrupt control register, refer to "Chapter 25 Interrupt Function" for details.

Note 2. After the comparator is set, an undesired interrupt request may occur during stable operation and the interrupt request flag must be initialized.

Note3. Comparator 0 must be set when interacting with PGA

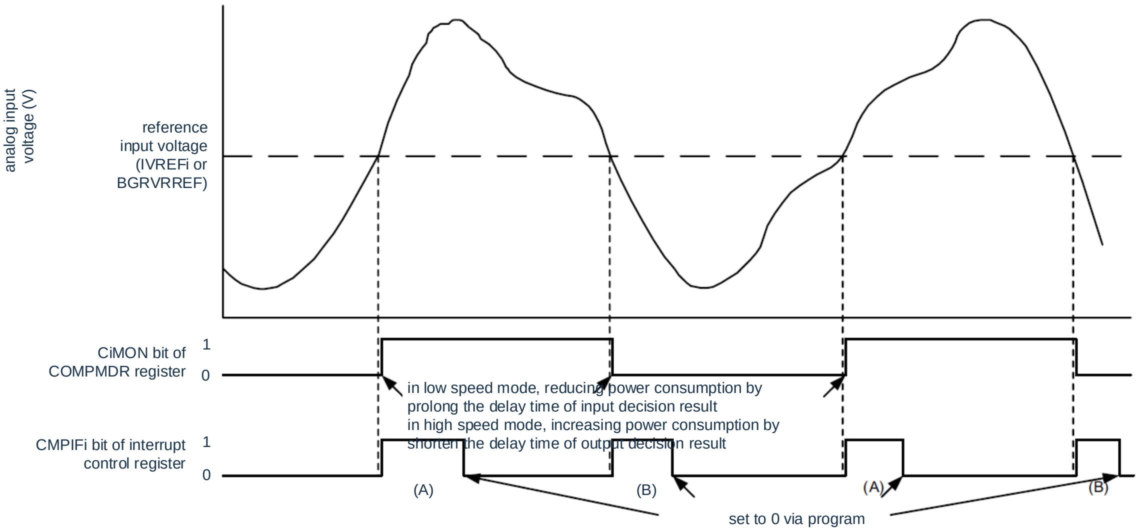
Remark: n=0-7, x=0-3

An example of operation of the comparator i (i=0,1) is as follows Figure 12-13. In the basic mode, when the analog input voltage is higher than the reference input voltage, the CiMON bit of the COMPMDR register is "1"; When the analog input voltage is lower than the reference input voltage, the CiMON bit is "0".

To use the comparator i interrupt, the CiIE bit of the COMPOCR register must be set to "1" (enable interrupt request). At this time, if the comparison result changes, an interrupt request of the comparator i is generated. For more information on interrupt requests, refer to "12.4.2 Comparator i interrupt (i=0,1)".

Figure 12-13 Operation example of comparator i (i=0,1) (basic mode)

- basic mode operation example


analog input voltage (V)
reference input voltage (IVREFi or BGRVRREF)
CiMON bit of COMPMDR register
1
0
in low speed mode, reducing power consumption by prolong the delay time of input decision result
in high speed mode, increasing power consumption by shorten the delay time of output decision result
CMPIFi bit of interrupt control register
1
0
(A)
(B)
(A)
(B)
set to 0 via program

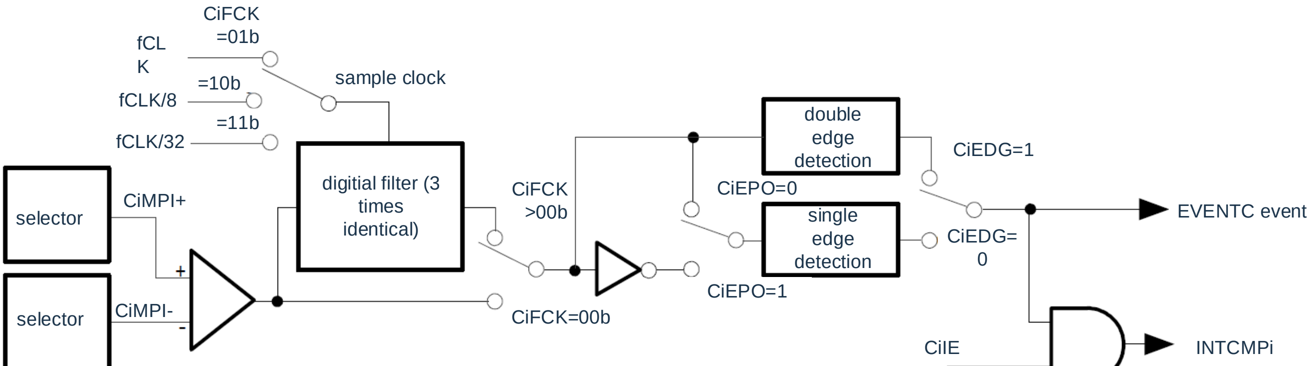
Note: The COMPFIR register has CiFCK1~CiFCK0 bit of '00B' (no filter) and CiEDG bit of '1' (two-sided edge) (CiEDG bit of '0' and CiEPO bit of '0' (rising edge) only CMPIFi changes in (A), CiEDG bit of '0' and CiEPO bit of '1' (falling edge).

12.4.1 Digital filter of comparator i (i=0,1)

The comparator i has a built-in digital filter, which can select the sampling clock through the CiFCK1~CiFCK0 bit of the COMPFIR register. The output signal of comparator i is sampled according to each sampling clock, and the digital filter outputs the sampling value.

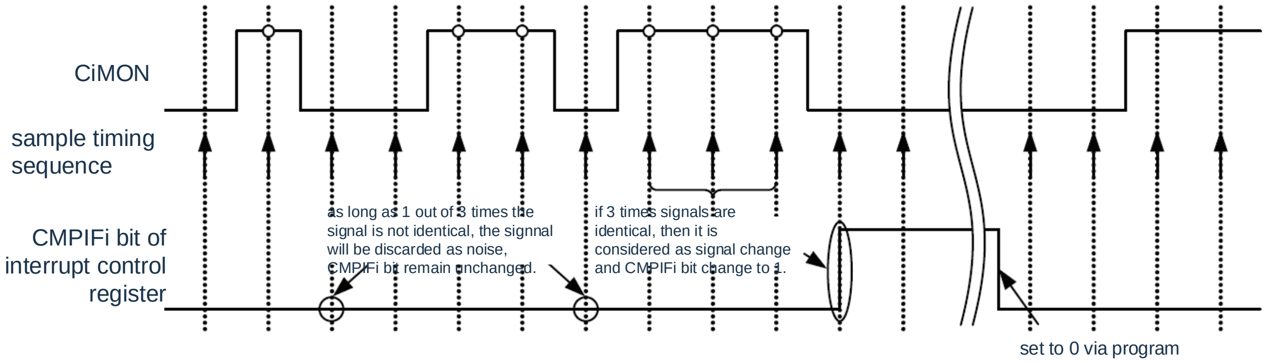
Figure is a result of the digital filter of the comparator i, Figure is an example of a digital filter and interrupt operation of a digital comparator i (i=0,1) of comparator i.

Figure 12-14 Comparator i digital filter (i=0,1) and edge detection structure



CiFCK，CiEPO，CiEDG: COMPFIR registers bits
CiIE: CIMPOCR register bits

Figure 12-15 Example of digital filter and interrupt operation for comparator i (i=0, 1)



as long as 1 out of 3 times the signal is not identical, the signnal will be discarded as noise, CMPIFi bit remain unchanged.

if 3 times signals are identical, then it is considered as signal change and CMPIFi bit change to 1.

Note: The above figure is an example of the COMPFIR register running when the CiFCK1~CiFCK0 bit is "01B" or "11B" (with a digital filter).

12.4.2 Comparator i interrupts (i=0,1)

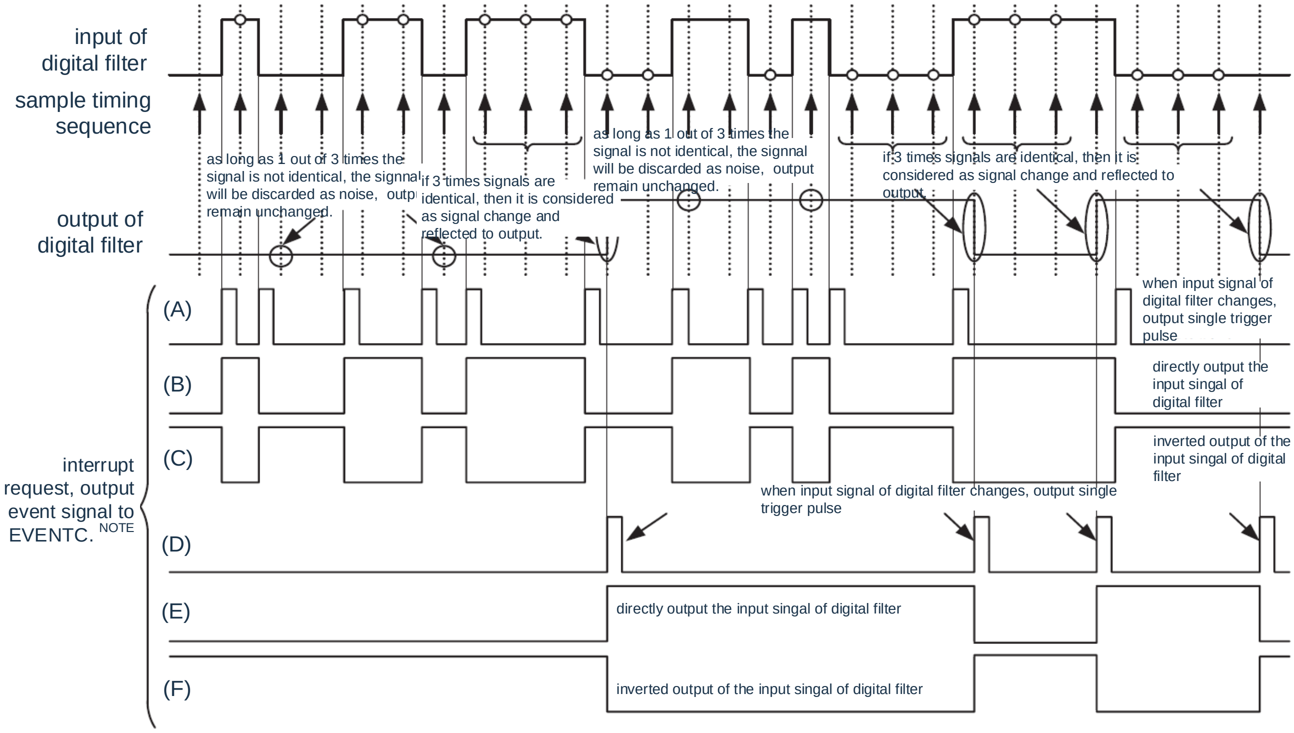
The comparator generates two interrupt requests for the comparator 0 and comparator 1. The comparator i interrupt has 1 priority specify flag, interrupt mask flag, interrupt request flag and interrupt vector respectively.

To use the comparator i interrupt, the CiIE bit of the COMPOCR register must be set to "1" (enable output of the interrupt request). The generation condition of the interrupt request is set through the COMPFIR register, and a digital filter can be added to the output of the comparator. The digital filter can select three kinds of sampling clock. Refer to "12.3.3 Comparator filter control register (COMPFIR)" and "12.3.4 Comparator output control register (COMPOCR) " for the setting of registers and the corresponding generation of interrupt requests.

12.4.3 Event signals output to the linkage controller (EVENTC)

An event signal output to the COMPFIR is generated by detecting the output edge of the digital filter set by the EVENTC register. However, unlike an interrupt request, the event signal is always output to the EVENTC regardless of the CiIE bit of the COMPOCR register. You must set the selection of the event output target and the stop of the event link through the EVENTC's ELSELR13 register and ELSELR14 register.

Figure 12-16 Operation of digital filters, interrupt requests and output of event signals to EVENTC


input of digital filter
sample timing sequence
as long as 1 out of 3 times the signal is not identical, the signnal will be discarded as noise, outp remain unchanged.
if 3 times signals are identical, then it is considered as signal change and reflected to output.
as long as 1 out of 3 times the signal is not identical, the signnal will be discarded as noise, output remain unchanged.
if 3 times signals are identical, then it is considered as signal change and reflected to output.
output of digital filter
(A)
when input signal of digital filter changes, output single trigger pulse
(B)
directly output the input singal of digital filter
(C)
inverted output of the input singal of digital filter
interrupt request, output event signal to EVENTC. NOTE
when input signal of digital filter changes, output single trigger pulse
(D)
(E)
directly output the input singal of digital filter
(F)
inverted output of the input singal of digital filter

Note: When the CiIE bit (i=0, 1) is '1', the interrupt request and the event signal output to the EVENTC are identical waveforms.

When the CiIE bit (i=0,1) is "0", only interrupt requests are fixed "0".

(A), (B), (C) waveforms are the case that the CiFCK bit (i=0,1) of the COMPFIR register is '00B' (without a digital filter), and the waveforms of (D), (E), and (F) are the case that the CiFCK bit (i=0,1) of the COMPFIR register is '01B', '10B', or '11B' (with a digital filter). (A), (D) are cases where the CiEDG bit is 1 (bilateral edges), (B), (E) is the CiEDG bit '0' and the CiEPO bit '0' (rising edge), (C), (F) is cases where the CiEDG bit is '0' and the CiEPO bit is '1' (falling edge).

12.4.4 Output of comparator i (i=0,1)

The CiOE bit of the COMPOCR register can be used to set whether the comparison result of the comparator is output to an external pin, and the CiOP bit of the COMPOCR register can be used to set the output polarity (positive or negative output). Refer to “12.3.4 Comparator output control register (COMPOCR)" for the corresponding register settings and comparator outputs.

To output the comparator's comparison results to the VCOUTi pin, you must set the port as follows (after reset, the port defaults to the input state):

① Set the mode of the comparator (Steps 2-5 of "Table Set-up steps for comparator-related registers").

② Sets the VCOUTi output of the comparator (sets the COMPOCR register, selects polarity, and allows output).

③ Set the port mode control register PMCxx corresponding to the output pin of VCOUTi to "0".

④ Set the bit of the port output latch register Pxx corresponding to the output pin of VCOUTi to "0".

⑤ Set the port direction register PMxx corresponding to the output pin of VCOUTi to output (output from the pin).

12.4.5 Stop and supply of comparator clock

In the case of stopping the comparator clock by setting the peripheral admission register 1 (PER1), you must follow these steps:

① Set the CiENB bit of the COMPMDR register to "0" (to stop the operation of the comparator).

② Set the IF bit of the interrupt request flag register to "0" (clear the interrupts that are not needed before the comparator stops running).

③ Set the PGACMPEN bit of PER1 register to "0".

If you stop the clock by setting the PER1 register, all internal registers for the comparator are initialized, so you set the register according to the steps in Table.

Note:

1. If the comparator n reference voltage selection bit (CnVRF) of the comparator mode setting register (COMPMDR) is set to '1', the output of the temperature sensor cannot be A/D converted.
2. If DMA start is permitted in one of the following states, DMA transfer is started and an interrupt is generated after transfer. Therefore, the monitor flag (CnMON) of the comparator must be confirmed as necessary to allow the DMA to start.
   - An interrupt request (CnEDG=0) is generated by unilateral edge detection of the comparator and an interrupt request (CnEPO=0) and IVCMP＞IVREF (or internal reference voltage 1.45V).
   - An interrupt request (CnEDG=0) is generated by unilateral edge detection of the comparator and an interrupt request (CnEPO=1) and IVCMP＜IVREF (or internal reference voltage 1.45V).

   (n=0, 1)

Chapter 13 Programmable Gain Amplifier (PGA)

13.1 Function of programmable gain amplifier

This product has a programmable gain amplifier (PGA0) with the following functions:

- PGA0 has 8 choices for amplification: 1/2/4/8/16/32/64/128
- PGA0 output with sample-and-hold circuit
- Supports single-ended/pseudo-differential inputs
- The output of the PGA1 can be selected as the analog input for the A/D converter

13.2 Registers for programmable gain amplifier

Table 13-1 Registers for controlling programmable gain amplifier

Peripheral enable register 1	PER1
programmable gain amplifier control register	PGA0CTL
Programmable gain amplifier sample hold control register	PGA0SH
Port mode control register	PMCxx
Port mode register	PMxx

13.2.1 Peripheral enable register 1 (PER1)

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

The bit5 (PGACMPEN) of this PER1 register must be set to '1' when using a programmable gain amplifier.

The PER1 register is set by 1-bit or 8-bit memory operation instructions.

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

Figure 13-1 Peripheral enable register 1 (PER1)

Adress: 0x4002081A Reset Value: 00H R/W

symbol	7	6	5	4	3	2	1	0
PER1	XX	XX	PGACMPEN	XX	XX	XX	XX	ADCEN

PGACMPEN	Control of input clock of comparator/programmable gain amplifier
0	Stop provide an input clock. The comparator or the programmable gain amplifier is in a reset state
1	Provides an input clock. Register readable and writable of comparator or programmable gain amplifier

Note:

Before configuring the comparator or the register of the programmable gain amplifier, confirm that the bit bit of the PGACMPEN is set to 1.

If PGACMPEN=0, writing to the comparator or the programmable gain amplifier control register is invalid, and all read-out values are default values. (except for Port Mode Register (PMXX) and Port Register PXX)

13.2.2 Programmable gain amplifier control register (PGAnCTL)

The PGA0CTL register are used to control the programmable gain amplifier to start working, stop working, and amplify.

The PGA0CTL register can be set by 1-bit or 8-bit memory instructions. After the reset signal is generated, the reset value of this register is 00H.

Figure 13-2 Format of PGA control register (PGA0CTL)

Adress: 0x40043846 Reset Value: 00H R/W

	7	6	5	4	3	2	1	0
PGA0CTL	PGA0EN	-	-	PGA0INHL	PGA0R1_N	PGA0VG2	PGA0VG1	PGA0VG0

PGA0EN	Programmable gain amplifier operation control
0	Amplifier stops working
1	Enable amplifier to work

PGA0R1_N	PGA0INHL	FUNCTION
1	0	"Differential" mode: High-side input is PGA0IN Low-side input is PGA0GND
	1	"Differential" mode: High-side input is PGA0GND Low-side input is PGA0IN
0	0	Single-ended mode: Positive (single-ended) input is PGA0IN
	1	Single-ended mode: Positive (single-ended) input is PGA0GND

PGA0VG2	PGA0VG1	PGA0VG0	PGA0 gain
0	0	0	1X
0	0	1	2X
0	1	0	4X
0	1	1	8x
1	0	0	16x
1	0	1	32x
1	1	0	64x
1	1	1	128x

13.2.3 Programmable gain amplifier sample hold control register (PGA0SH)

The PGA0SH register is used to control the programmable gain amplifier sample hold function. When PAG0 and ADC are linked, this register can be set to control the output of PGA0 to hold after a period of sampling time, while the ADC automatically starts conversion after the PGA output is held, and PGA0 returns to the sampling state after the conversion is completed.

For a detailed description of the PAG0SH register and the action status diagram of the ADC combined with the sample hold function of PGA0, refer to section 14.2.16 Programmable gain amplifier sample hold control register (PGA0SH) in the "AD Conversion" chapter.

13.2.4 Registers for controlling analog input pin port function

When using the PGA0IN pin and PGA0GND pin as the analog input of the programmable gain amplifier, the bit of the Port Mode Register (PMxx) and the bit of the Port Digital/Analog Control Register (PMCxx) corresponding to each port must be set to "1".

13.3 Operation of programmable gain amplifier

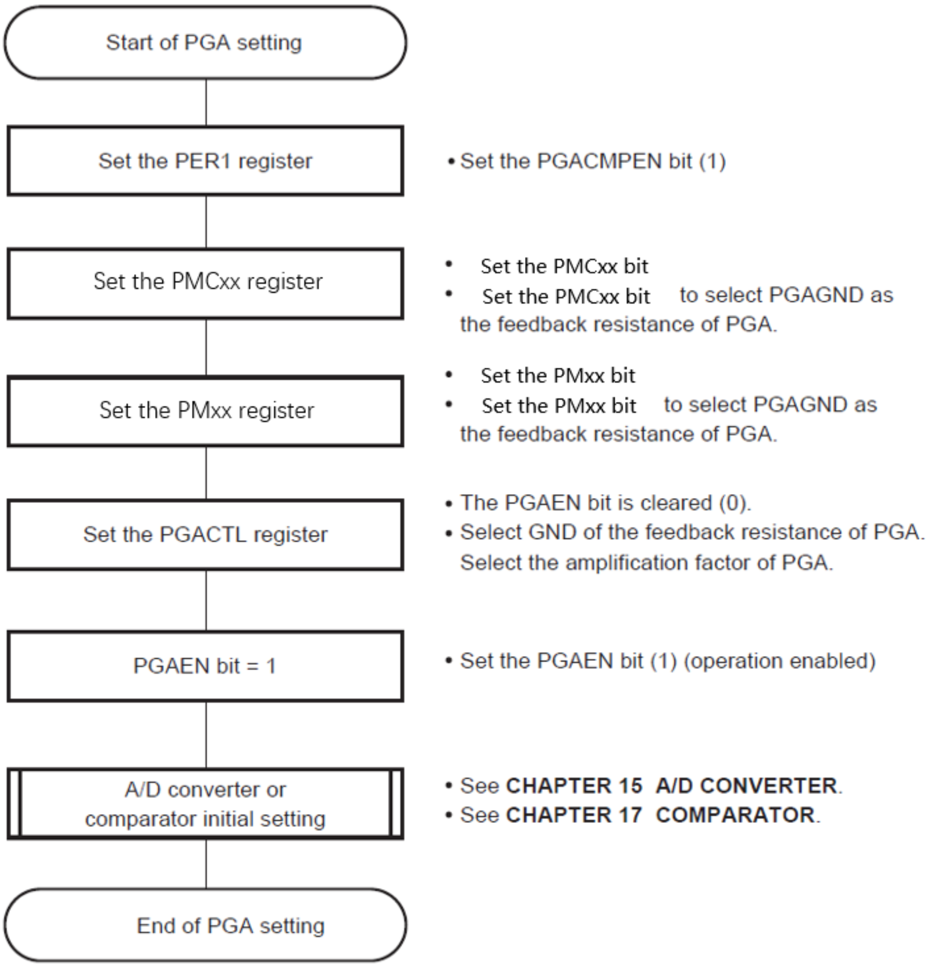
The analog voltage input by the PGA0IN pin is amplified, and the amplification gain has 8 choices.

The amplified voltage may be used for analog input of the A/D converter and positive input signal of the comparator 0 (CMP 0).

The steps for starting and stopping the programmable gain amplifier are as follows.

13.3.1 Starting operation steps of programmable gain amplifier

Take PGA0 as an example, set up as follows:

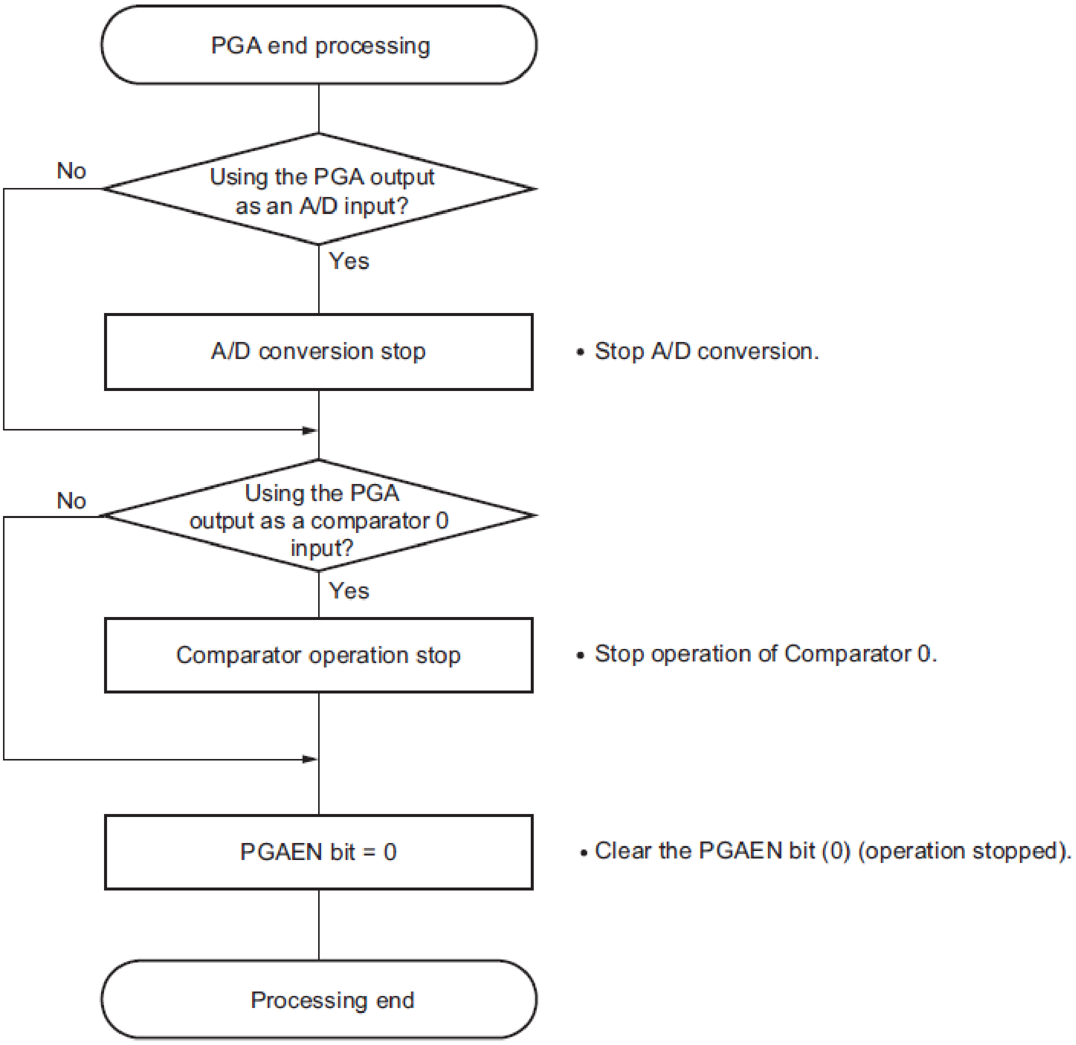

Start of PGA setting
Set the PER1 register
• Set the PGACMPEN bit (1)
Set the PMCxx register
• Set the PMCxx bit
• Set the PMCxx bit to select PGAGND as the feedback resistance of PGA.
Set the PMxx register
• Set the PMxx bit
• Set the PMxx bit to select PGAGND as the feedback resistance of PGA.
Set the PGACTL register
• The PGAEN bit is cleared (0).
• Select GND of the feedback resistance of PGA.
Select the amplification factor of PGA.
PGAEN bit = 1
• Set the PGAEN bit (1) (operation enabled)
A/D converter or comparator initial setting
• See CHAPTER 15 A/D CONVERTER.
• See CHAPTER 17 COMPARATOR.
End of PGA setting

Note 1. A PGA stabilization time of 10us is required after setting the PGAEN bit to 1. The A/D conversion is then initiated.

2. If you use the PGA's sample hold function, you need to set the PGA0SH register in the A/D converter initial setting.

13.3.2 Stopping operation steps of programmable gain amplifier

Take PGA0 as an example, set up as follows:


PGA end processing
Using the PGA output as an A/D input?
No
Yes
A/D conversion stop
• Stop A/D conversion.
Using the PGA output as a comparator 0 input?
No
Yes
Comparator operation stop
• Stop operation of Comparator 0.
PGAEN bit = 0
• Clear the PGAEN bit (0) (operation stopped).
Processing end

Note 1. When restarting PGA and A/D conversion or amplifier, 10us PGA stabilization time is required after setting the PGAEN bit to 1.

Chapter 14 A/D Converter

The number of analog input channels of the A/D converter varies depending on the product.

Pin Number	64pins	64pins(-S) Note1	48pins	48pins(-S) Note1
analog transmission access channel	33ch	34ch	27ch	25ch
	(ANI0~ANI12, ANI15~ANI34)	(ANI0~ANI33)	(ANI0~ANI12, ANI15~ANI26, ANI33, ANI34)	(ANI0~ANI20, ANI23~ANI26)

Note1: (-A) means only BAT32G157xx-S series products.

14.1 Function of A/D converter

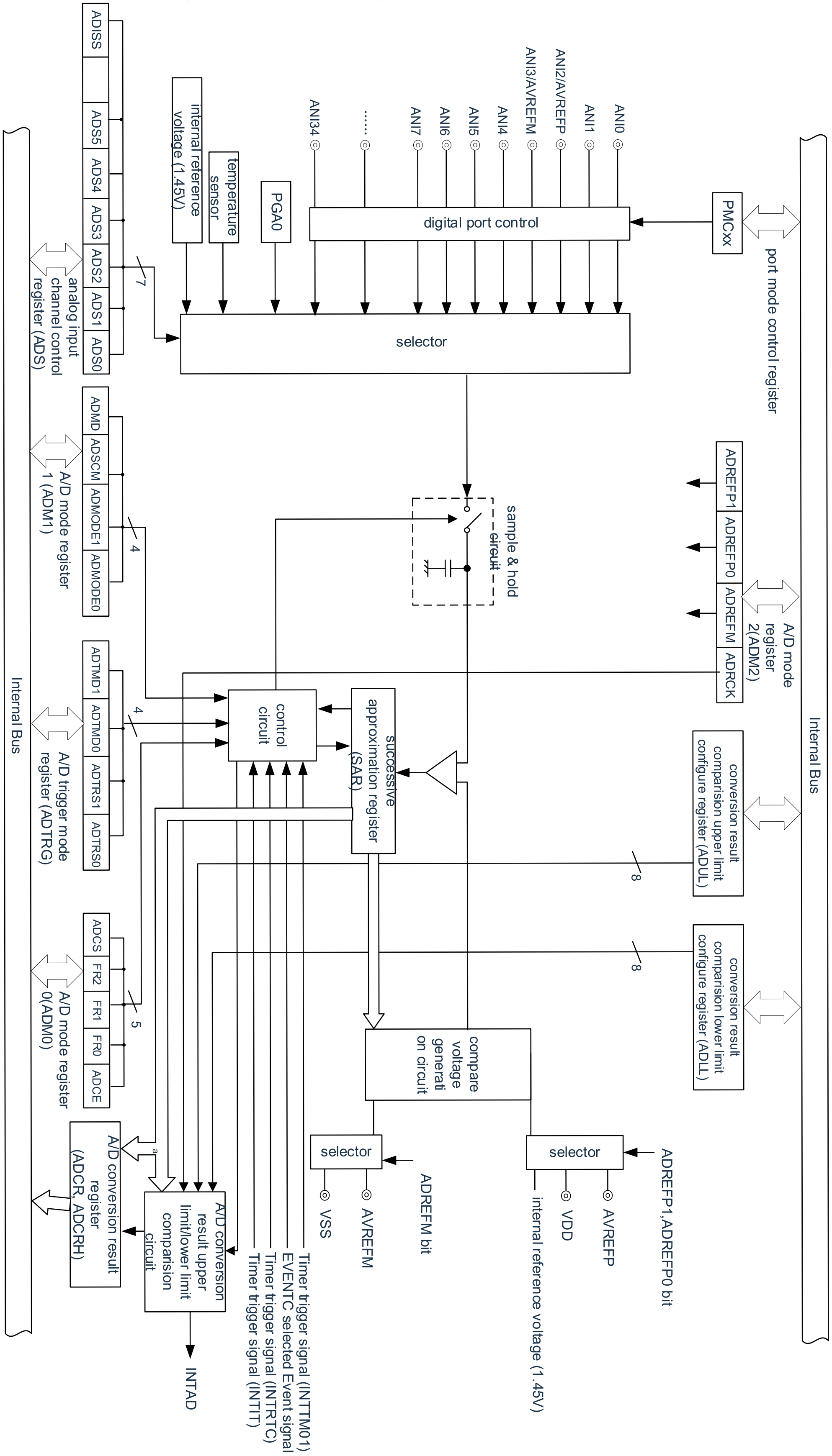
A/D converter is a converter that converts analog input to digital values, the A/D converter has the following functions.

- A/D Conversion of 12-bit Resolution
  The 12-bit resolution A/D conversion is repeated by selecting 1 channel analog input from ANI0~ANI34, PGA0 and temperature sensors. Each time an A/D transition is completed, an interrupt request (INTAD) is generated (a case of mode selection).

Various A/D conversion modes can be set by the following mode combination.

trigger mode	software trigger	The conversion is started by software operation.
	Hardware triggered no-wait mode	The conversion is started by detecting a hardware trigger.
	Hardware Trigger Wait Mode	In the conversion standby state where the A/D power is cut off, the power is turned on by detecting the hardware trigger, and the conversion starts automatically after the A/D power stabilization waiting time.
channel selection mode	selection mode	Select the analog input for 1 channel for A/D conversion.
	Scan mode	Analog inputs for the four channels are A/D converted sequentially. It can select 4 channels in ANI0~ANI15 as the analog input.
		Analog inputs for the three channels are A/D converted sequentially. It can select 3 channels in ANI0~ANI15 as the analog input.
		Analog inputs for the two channels are A/D converted sequentially. It can select 2 channels in ANI0~ANI15 as the analog input.
Conversion Mode	Single conversion Mode	Make 1 A/D conversion on the selected channel.
	continuous conversion mode	Performs a continuous A/D conversion on the selected channel until it is stopped by the software.
sampling time	Sampling clock 5.5~255 ADCLKs	The sampling time can be selected by the ADNSMP register, using 13.5 conversion clocks (f_{AD}).

Figure 14-1 Block diagram of A/D converter


Internal Bus
port mode control register
PMCxx
digital port control
ANI0
ANI1
ANI2/AVREFP
ANI3/AVREFM
ANI4
ANI5
ANI6
ANI7
......
ANI34
PGA0
temperature sensor
internal reference voltage (1.45V)
selector
ADISS
ADS5
ADS4
ADS3
ADS2
ADS1
ADS0
7
analog input channel control register (ADS)
sample & hold circuit
ADREFP1
ADREFP0
ADREFM
ADRCK
A/D mode register 2(ADM2)
ADMD
ADSCM
ADMODE1
ADMODE0
4
A/D mode register 1 (ADM1)
ADTMD1
ADTMD0
ADTRS1
ADTRS0
4
A/D trigger mode register (ADTRG)
control circuit
successive approximation register (SAR)
conversion result comparison upper limit configure register (ADUL)
8
conversion result comparison lower limit configure register (ADLL)
8
ADCS
FR2
FR1
FR0
ADCE
5
A/D mode register 0(ADM0)
compare voltage generati on circuit
selector
ADREFP1,ADREFP0 bit
AVREFP
VDD
internal reference voltage (1.45V)
selector
ADREFM bit
AVREFM
VSS
A/D conversion result register (ADCR, ADCRH)
a
A/D conversion result upper limit/lower limit comparison circuit
INTAD
Timer trigger signal (INTTM01)
EVENTC selected Event signal
Timer trigger signal (INTRTC)
Timer trigger signal (INTIT)
Internal Bus

Note: Please refer to 14.2.6 Analog input channel specification register (ADS) for the selection of analog input channel ANIx.

14.2 Registers for controlling 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_000H

Register Name	Register Description	R/W	Reset Value	Register Address
PER1	Peripheral Enable Register 1	R/W	00H	CSC_BASE+20H
ADM0	A/D converter mode register 0	R/W	00H	ADC_BASE+00H
ADM1	A/D converter mode register 1	R/W	00H	ADC_BASE+02H
ADM2	A/D converter mode register 2	R/W	00H	ADC_BASE+04H
ADTRG	A/D converter trigger mode register	R/W	00H	ADC_BASE+06H
ADS	Analog input channel specification register	R/W	00H	ADC_BASE+08H
ADLL	Conversion result comparison lower limit setting register	R/W	00H	ADC_BASE+0AH
ADUL	Conversion result comparison upper limit setting register	R/W	00H	ADC_BASE+0BH
ADNSMP	A/D converter sampling time control register	R/W	0dH	ADC_BASE+0CH
ADCR	12-bit A/D conversion result register	R	0000H	ADC_BASE+0EH
ADCRH	8-bit A/D conversion result register	R	00H	ADC_BASE+0FH
ADTES	A/D test register	R/W	00H	ADC_BASE+10H
ADNDIS	A/D converters charge-discharge control register	R/W	00H	ADC_BASE+11H
ADSMTWIT	A/D converter sampling time extension control register	R/W	00H	ADC_BASE+15H
ADFLG	A/D hard module status register	R	00H	ADC_BASE+16H
PGA0SH	Programmable gain amplifier sample hold register	R/W	00H	ADC_BASE+1EH
PMCn	Port mode control register	R/W	Note1	PORT_BASE+Note1

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

Note 1: When selecting a channel through the ADS register, you need to configure the PMC register of that channel pin as an analog channel.

14.2.1 Peripheral enable register 0 (PER0)

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

When you want to use an A/D converter, you must set the bit0 (ADCEN).

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

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

Figure 14-2 Format of peripheral enable register (PER1)

Adress:0x4002081A Reset value: 00H R/W

Symbol	7	6	5	4	3	2	1	0
PER1	XX	XX	XX	XX	XX	XX	XX	ADCEN

ADCEN	Control of input clock of A/D converter
0	Stop provide an input clock. •Cannot write the SFR used by the A/D converter. •The A/D converter is in a reset state.
1	Provides an input clock. •Read and write SFRs used by A/D converters.

Note 1.: When you want to set up an A/D converter, you must first read and write the following register in the state with the ADCEN bit "1". When the ADCEN bit is '0', the control register value of the A/D converter is the initial value, ignoring write operations (except for the port mode control register (PMCxx)).

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

14.2.2 A/D converter mode register 0 (ADM0)

Register used to set the A/D conversion clock, start of conversion, or stop. The ADM0 register is set by an 8-bit memory operation instruction.

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

Figure 14-3 Format of A/D converter mode register 0 (ADM0)

Reset value: 00H
R/W

	7	6	5	4	3	2	1	0
ADM0	ADCS	0	FR2	FR1	FR0	0	0	ADCE

ADCS	A/D conversion operation control
0	Stop the conversion run. [Read] Stop Transition Run/Standby State
1	Allow the transformation to run. [Read] When software triggers mode: Transition Health When hardware triggers wait mode: A/D power supply waits for steady state + transition running state

ADCE	A/D voltage comparator operation control Note 2
0	Stop the A/D voltage comparator operation.
1	Allows the A/D voltage comparator operation.

Note: 1. For details on FR2-FR0 A/D conversion, refer to Table 14-3 A/D Conversion.

2. The A/D converter needs 1μs stabilization time to start operation. In software-triggered mode or hardware-triggered no-wait mode, the conversion result is valid if the ADCE bit is set to "1" after at least 1μs and then the ADCS bit is set to "1". If the waiting time is less than 1μs and the ADCS bit is set to "1", the result of this conversion must be ignored. In the hardware-triggered wait mode, a wait time of 1 μs is guaranteed by design.

Note 1. You must change the FR2~FR0 bit under the transition stop state ADCS=0.

2. Prohibit ADCS=1, ADCE=0 settings.

3. Disable from setting ADCS=0, ADCE=0 status to ADCS=1, ADCE=1 through 8-bit operation instructions. You must follow the steps of the “14.5 A/D converter set-up flowchart”.

Table 14-1 Settings for ADCS and ADCE bits

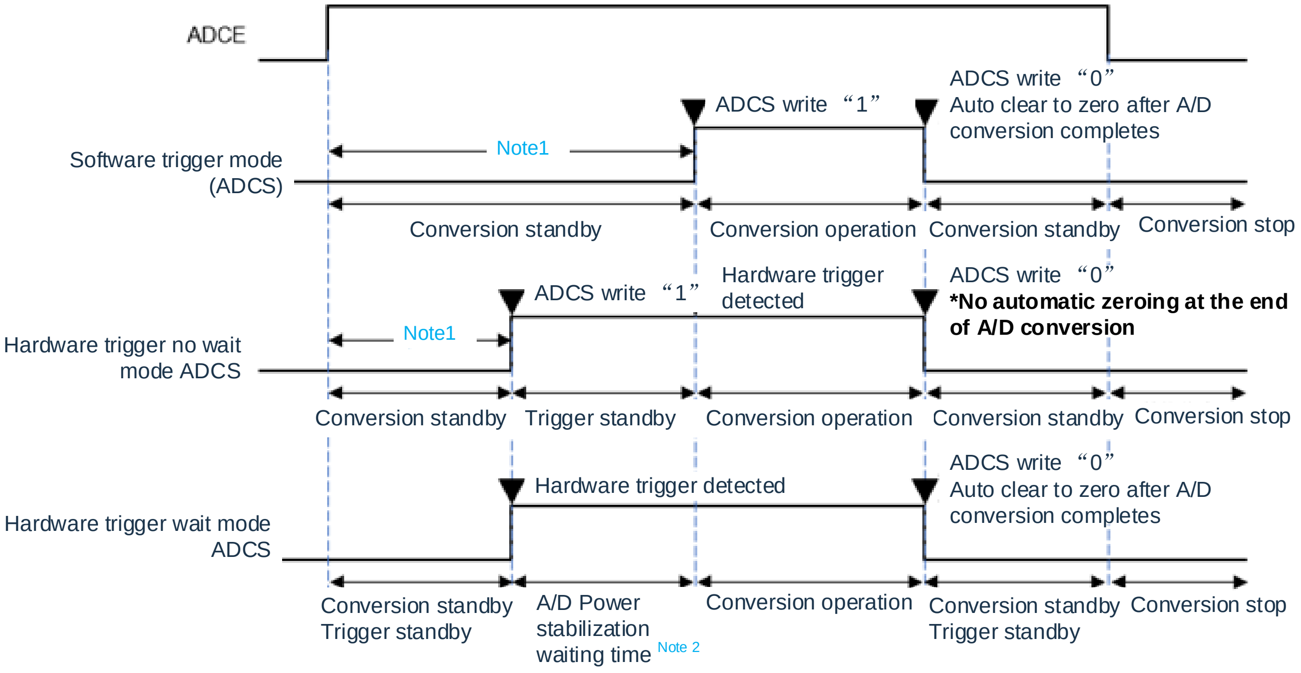
ADCS	ADCE	A/D conversion operation
0	0	Transition stop
0	1	Transition standby
1	0	Disable settings
1	1	Transition operation status

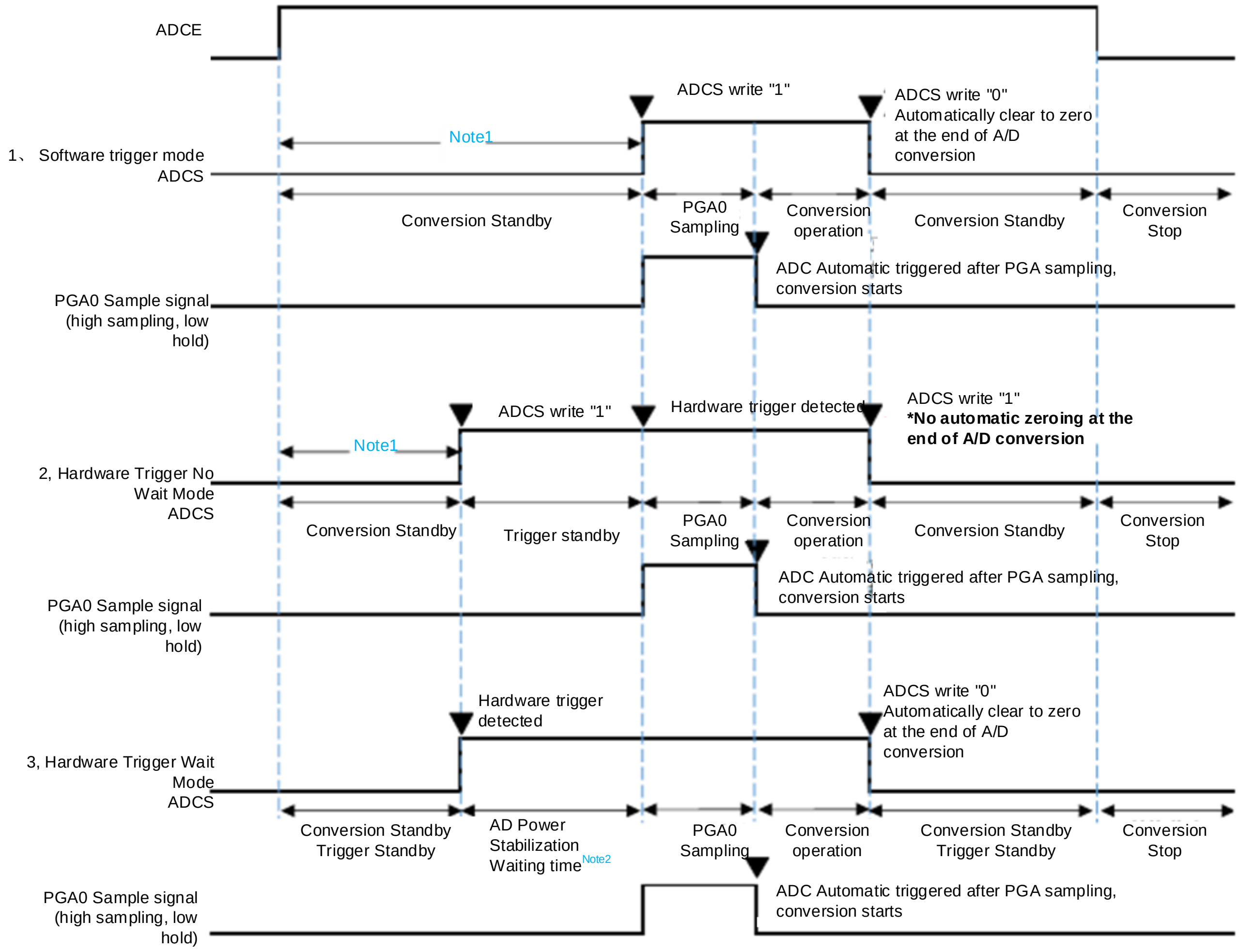
Table 14-2 Set and clear conditions for ADCS bits

A/D conversion mode			Set condition	Clear condition
Software trigger	select mode	continuous conversion mode	When writing "1" to ADCS bit	When writing "0" to ADCS bit
		Single Conversion Mode		• When writing "0" to ADCS bits • Automatically clears "0" at the end of the A/D conversion.
	Scan mode	continuous conversion mode		When writing "0" to ADCS bit
		Single Conversion Mode		• When writing "0" to ADCS bits • Automatically clears "0" when n channel transitions are set up.
Hardware triggered no-wait mode	select mode	continuous conversion mode	When I give ADCS a bit "1" hour	When writing "0" to ADCS bit
		Single Conversion Mode		• When writing "0" to ADCS bits
	Scan mode	continuous conversion mode		When writing "0" to ADCS bit
		Single Conversion Mode		• When writing "0" to ADCS bits
Hardware trigger wait mode	select mode	continuous conversion mode	When the input hardware triggers	When writing "0" to ADCS bit
		Single Conversion Mode		• When writing "0" to ADCS bits • Automatically clears "0" at the end of the A/D conversion.
	Scan mode	continuous conversion mode		When writing "0" to ADCS bit
		Single Conversion Mode		• When writing "0" to ADCS bits • Automatically clears "0" when 4 channel transitions are set up.

Note: n=2, 3, 4

Figure 14-4 Action state diagram with A/D modes




ADCE
1、 Software trigger mode ADCS
Note1
ADCS write "1"
ADCS write "0" Automatically clear to zero at the end of A/D conversion
Conversion Standby
PGA0 Sampling
Conversion operation
Conversion Standby
Conversion Stop
ADC Automatic triggered after PGA sampling, conversion starts
PGA0 Sample signal (high sampling, low hold)
2, Hardware Trigger No Wait Mode ADCS
Note1
ADCS write "1"
Hardware trigger detected
ADCS write "1"
*No automatic zeroing at the end of A/D conversion
Conversion Standby
Trigger standby
PGA0 Sampling
Conversion operation
Conversion Standby
Conversion Stop
ADC Automatic triggered after PGA sampling, conversion starts
PGA0 Sample signal (high sampling, low hold)
3, Hardware Trigger Wait Mode ADCS
Hardware trigger detected
ADCS write "0" Automatically clear to zero at the end of A/D conversion
Conversion Standby Trigger Standby
AD Power Stabilization Waiting time Note2
PGA0 Sampling
Conversion operation
Conversion Standby Trigger Standby
Conversion Stop
ADC Automatic triggered after PGA sampling, conversion starts
PGA0 Sample signal (high sampling, low hold)

Note: 1. In the software trigger mode or hardware trigger no wait mode, in order to stabilize the internal circuit, the rise time from ADCE bit to ADCS bit needs at least 1us (TBD).

2. In hardware triggered wait mode, A/D power supply stability time 1 s is guaranteed by design.

Notice:

1. To use the hardware trigger wait mode, setting the ADCS bit to "1" is prohibited (it is automatically switched to "1" when a hardware trigger signal is detected). However, the ADCS bit can be set to "0" in order to set the standby state for A/D conversion.
2. The ADCE bit must be overridden when the ADCS bit is "0" (Stop Switching/Switching Standby).
3. To end an A/D conversion, you must set at least the hardware trigger interval to:
   Hardware triggered no-wait mode: 2 fCLK clock + A/D transition times
   Hardware triggered wait mode: 2 fCLK clocks +A/D power supply steady wait time +A/D transition time

Remark: f_{CLK}: Clock frequency of CPU/peripheral hardware

Table 14-3 A/D conversion time selection (1/2)

(1) No A/D power steady wait time (software trigger mode/hardware trigger no wait mode)

Mode of A/D Converter Register 0 (ADM0)			Mode of A/D Converter Register 1 (ADM1)		Mode	Frequency of the conversion clock ADCLK (f_{AD})	12-bit resolution conversion time[Note2] ADC conversion time = (number of sampling clocks + number of successive comparison clocks)/ f_{AD}	
FR2	FR1	FR0	ADMODE [1]	ADMOD [0]			Number of ADC conversion clocks (13.5 sample clocks)	ADC conversion time (13.5 sample clocks)
0	0	0	0	0	High speed transform mode	f_{CLK}/32	45 ADCLKs (13.5 sample clocks +31.5 successive comparison clocks)	45/ f_{AD}
0	0	1				f_{CLK}/16		
0	1	0				f_{CLK}/8		
0	1	1				f_{CLK}/4		
1	0	0				f_{CLK}/2		
1	0	1				f_{CLK}/1		
0	0	0	1	1	Low current mode	f_{CLK}/32	54 ADCLKs (13.5 sample clocks +40.5 successive comparison clocks)	54/ f_{AD}
0	0	1				f_{CLK}/16		
0	1	0				f_{CLK}/8		
0	1	1				f_{CLK}/4		
1	0	0				f_{CLK}/2		
1	0	1				f_{CLK}/1		

Note: 1. To override FR2~FR0 bits, ADMODE[1:0] bits to different data, it must be done in the transition stop state (ADCS=0).

Note 2: Time required to perform one ADC conversion = (number of sampling clocks + number of successive comparison clocks) / f_{AD}

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

Remark: f_{CLK}: Clock frequency of CPU/peripheral hardware

Table 14-4 A/D conversion time selection (2/2)

(1) With A/D power steady wait time (hardware triggered wait mode Note 1).

Mode of A/D Converter Register 0 (ADM0)			Mode of A/D Converter Register 1 (ADM1)		Mode	Frequency of the conversion clock ADCLK (fAD)	A/D power supply stable waiting Time	Number of ADC conversion clocks	A/D power stabilization waiting time +ADC conversion time Note 2
FR2	FR1	FR0	ADMODE [1]	ADMOD [0]					
0	0	0	0	0	High speed transform mode	fCLK/32	1μs	45 ADCLKs (13.5 sample clocks +31.5 successive comparison clocks)	1μs +45/f_{AD}
0	0	1				fCLK/16			
0	1	0				fCLK/8			
0	1	1				fCLK/4			
1	0	0				fCLK/2			
1	0	1				fCLK/1			
0	0	0	1	1	Low current mode	fCLK/32	1μs	54 ADCLKs (13.5 sample clocks +40.5 successive comparison clocks)	1μs +54/f_{AD}
0	0	1				fCLK/16			
0	1	0				fCLK/8			
0	1	1				fCLK/4			
1	0	0				fCLK/2			
1	0	1				fCLK/1			

Note 1. In hardware trigger wait mode, the power stabilization time is guaranteed by hardware design and does not need to be set. And in continuous conversion mode, the A/D power stabilization wait time occurs only after the hardware trigger is detected for the 1st time.

Note 2. The time required for ADC conversion after hardware trigger = 1us + (number of sampling clocks + number of successive comparison clocks)/f_{AD}, where the number of sampling clocks can be adjusted through the ADNSMP register, the default is 13.5 ADCLK. The number of successive comparison clocks is determined by the conversion mode, 31.5 ADCLK in high speed conversion mode, 40.5 in low current mode. The fastest clock supported by ADCLK is 64MHz in high speed conversion mode and 27MHz in low current mode.

Note 3. When using the output of PLL as f_{CLK}, the hardware does not support hardware-triggered wait mode because the frequency of f_{CLK} is uncertain and the hardware cannot calculate the 1us power wait time.

Notice: 1. To override FR2~FR0 bits, ADMODE[1:0] bits to different data, it must be done in the transition stop state (ADCS=0).

2. The transition time in Hardware Triggered Wait Mode contains the A/D Power Supply Steady Wait time after the hardware trigger is detected.

Remark: f_{CLK}: Clock frequency of the CPU/peripheral hardware.

14.2.3 A/D converter mode register 1 (ADM1)

This is a register that sets the A/D conversion mode.

The ADM1 register is set by an 8-bit memory operation instruction.

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

Figure 14-5 Format of A/D converter mode register 1

Reset value: 00H
R/W

	7	6	5	4	3	2	1	0
ADM1	ADMD	SMODE1	SMODE0	0	ADSCM	0	ADMODE1	ADMODE0

ADMD	Setting of A/D conversion channel selection mode
0	Select mode
1	Scan mode

SMODE1	SMODE0	Number of channels to cycle in scan mode
0	0	4-channel scan
0	1	3-channel scan
1	0	2-channel scan
Others		Disable settings

ADSCM	Settings for A/D Conversion Mode
0	Continuous conversion mode
1	Single Conversion Mode

ADMODE1	ADMODE0	A/D conversion mode
0	0	high speed transform mode (ADCLK fastest clock is 64MHz)
1	1	low current mode (ADCLK fastest clock is 27MHZ)
Others		Disable settings

Note: Bit 6 ~ 4, 2 must be set to "0".

Note:1. To override the ADM1 register, it must be done in the transition stop state (ADCS=0).

2. In order to end A/D conversion normally, you must set the hardware trigger interval at least as follows:

Hardware triggered no-wait mode: 2 f_{CLK} clocks + A/D transition times
Hardware triggered wait mode: 2 f_{CLK} clocks +A/D power supply steady wait time +A/D transition time

3. For A/D conversion, the number of successive comparison clocks is determined by the conversion mode, which is 31.5 ADCLK for high speed conversion mode and 4 4. 0.5 ADCLK for low current mode, and the fastest clock supported by ADCLK is 64MHz for high speed conversion mode and 27MHz for low current mode. In practice, please configure the conversion mode and conversion clock frequency according to the "AC Characteristics" requirement in the data sheet.

Remark 1: f_{CLK}: Clock frequency of CPU/peripheral hardware

14.2.4 A/D converter mode register 2 (ADM2)

The ADM2 register is set by an 8-bit memory operation instruction.
After the reset signal is generated, the value of this register changes to "00H".

Figure 14-6 Format of A/D converter mode register 2 (ADM2)

Reset value: 00H
R/W

	7	6	5	4	3	2	1	0
ADM2	ADREFP1	ADREFP0	ADREFM	0	ADRCK	0	CHRDE	0

ADREFP1	ADREFP0	Selection of positive (+) reference voltage source for A/D converter
0	0	Supplied from V_{DD}
0	1	Supplied from AV_{REFP} external terminals.
1	0	Supplied from the internal reference voltage (1.45 V)
1	1	Settings are disabled.

ADREFM	Selection of negative (-) reference voltage source for A/D converter
0	Provided by VSS.
1	Supplied from AV_{REFM} external terminals.

ADRCK	Examination of the upper and lower limits of conversion results
0	An interrupt signal (INTAD) is generated when the ADLL register ⩽ADCR register ≤ ADUL register (AREA1).
1	An interrupt signal (ADCR) is generated when the ADCR register <ADLL register (AREA2) or the ADUL register <INTAD register (AREA3).
The generation range of the interrupt signal (INTAD) of AREA1~AREA3 is shown in Figure 14-8	

CHRDE	Output enable of channel identification in A/D converter scan mode
0	When scanning mode, channel numbers are not identified in the translation results
1	When scanning mode, the fourth bit of the converted result (ADCR register [15:12]) is the channel number for this result

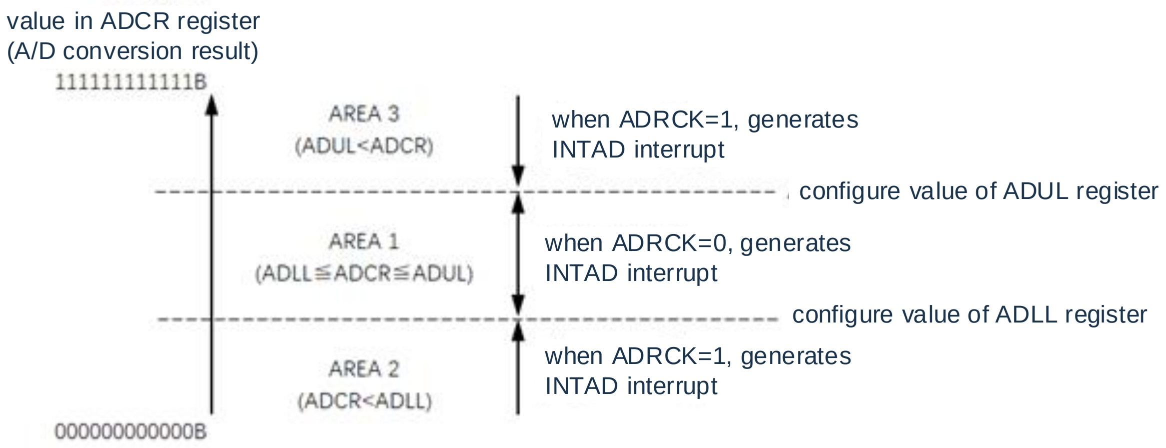


Figure 14-7 Interrupt signal generation range for ADRCK bits

Note 1. To override the ADM2 register, it must be done in the transition stop state (ADCS=0).

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

Remark: When INTAD does not occur, A/D conversion results are not saved to ADCR registers and ADCRH registers.

14.2.5 A/D converter trigger mode register (ADTRG)

This is a register that sets the A/D conversion trigger mode and hardware trigger signal.

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

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

Figure 14-8 Format of A/D converter trigger mode register (ADTRG)

Reset value: 00H
R/W

	7	6	5	4	3	2	1	0
ADTRG	ADTMD1	ADTMD0	0	0	0	0	ADTRS1	ADTRS0

ADTMD1	ADTMD0	Selection of trigger mode for A/D conversion
0	0	software trigger mode
0	1	
1	0	Hardware triggered no-wait mode
1	1	Hardware Trigger Wait Mode

ADTRS1	ADTRS0	Selection of hardware trigger signal
0	0	Timer channel 1 counts end or captures end interrupt signal (INTTM01)
0	1	ELC selected event signal
1	0	Real-time clock interrupt (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 A/D conversion normally, you must set the hardware trigger interval at least as follows:

Hardware triggered no-wait mode: 2 f_{CLK} clock + A/D conversion times
Hardware triggered wait mode: 2 f_{CLK} clocks +A/D power supply steady wait time +A/D conversion time

Remark 1.f_{CLK}: Clock frequency of the CPU/peripheral hardware
2. When using the output of the PLL as F_{CLK}, the hardware does not support the hardware-triggered wait mode because the frequency of F_{CLK} is uncertain and the hardware cannot calculate the 1us power wait time.

14.2.6 Analog input channel specification register (ADS)

This is a register that specifies the analog voltage input channel to be A/D converted.

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

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

Figure 14-9 Format of analog input channel specification register (ADS)

Reset value: 00H R/W

	7	6	5	4	3	2	1	0
ADS	ADISS	0	ADS5	ADS4	ADS3	ADS2	ADS1	ADS0

○ Select mode (ADM1.ADMD=0)

ADS register setting value		CH selection	Pin name
ADISS	ADS[5:0]		
0	6’h00	ANI0	PA00
0	6’h01	ANI1	PA01
0	6’h02	ANI2	PA02
0	6’h03	ANI3	PA03
0	6’h04	ANI4	PA04
0	6’h05	ANI5	PA05
0	6’h06	ANI6	PA06
0	6’h07	ANI7	PD04
0	6’h08	ANI8	PD05
0	6’h09	ANI9	PD06
0	6’h0a	ANI10	PD07
0	6’h0b	ANI11	PD08
0	6’h0c	ANI12	PB06
0	6’h0d	ANI13	PB07
0	6’h0e	ANI14	PB08
0	6’h0f	ANI15	PC00
0	6’h10	ANI16	PC01
0	6’h11	ANI17	PC02
0	6’h12	ANI18	PC03
0	6’h13	ANI19	PC04
0	6’h14	ANI20	PC05
0	6’h15	ANI21	PC06
0	6’h16	ANI22	PC07
0	6’h17	ANI23	PC08
0	6’h18	ANI24	PC09
0	6’h19	ANI25	PC10
0	6’h1a	ANI26	PC11
0	6’h1b	ANI27	PA11
0	6’h1c	ANI28	PA12
0	6’h1d	ANI29	PA13
0	6’h1e	ANI30	PA14
0	6’h1f	ANI31	PB01
0	6’h20	ANI32	PB02
0	6’h21	ANI33	PB03

0	6’h22	ANI34	PB04
0	6’h23	PGA0	PGA0
0	6’h3f	ANIxx all OFF	
1	6’h00	BGR(temperature sensor0)	
1	6’h01	BGR(1.45V)	
Other settings are prohibited			

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

2. The analog input channels of the A/D converters vary by product. Please refer to the datasheet for detailed channel specification information.

○ 4-channel scan mode (ADM1. ADMD=1)

ADISS	ADS[5:0]	Analog input channel			
		Scan 0	Scan 1	Scan 2	Scan 3
1’b0	6’h00	ANI0	ANI1	ANI2	ANI3
1’b0	6’h01	ANI1	ANI2	ANI3	ANI4
1’b0	6’h02	ANI2	ANI3	ANI4	ANI5
1’b0	6’h03	ANI3	ANI4	ANI5	ANI6
1’b0	6’h04	ANI4	ANI5	ANI6	ANI7
1’b0	6’h05	ANI5	ANI6	ANI7	ANI8
1’b0	6’h06	ANI6	ANI7	ANI8	ANI9
1’b0	6’h07	ANI7	ANI8	ANI9	ANI10
1’b0	6’h08	ANI8	ANI9	ANI10	ANI11
1’b0	6’h09	ANI9	ANI10	ANI11	ANI12
1’b0	6’h0A	ANI10	ANI11	ANI12	ANI13
1’b0	6’h0B	ANI11	ANI12	ANI13	ANI14
1’b0	6’h0C	ANI12	ANI13	ANI14	ANI15
Others		Disable from setting.			

○ 3-channel scan mode (ADM1. ADMD=1)

ADISS	ADS[5:0]	Analog input channel		
		Scan 0	Scan 1	Scan 2
1’b0	6’h00	ANI0	ANI1	ANI2
1’b0	6’h01	ANI1	ANI2	ANI3
1’b0	6’h02	ANI2	ANI3	ANI4
1’b0	6’h03	ANI3	ANI4	ANI5
1’b0	6’h04	ANI4	ANI5	ANI6
1’b0	6’h05	ANI5	ANI6	ANI7
1’b0	6’h06	ANI6	ANI7	ANI8
1’b0	6’h07	ANI7	ANI8	ANI9
1’b0	6’h08	ANI8	ANI9	ANI10
1’b0	6’h09	ANI9	ANI10	ANI11
1’b0	6’h0A	ANI10	ANI11	ANI12
1’b0	6’h0B	ANI11	ANI12	ANI13
1’b0	6’h0C	ANI12	ANI13	ANI14
1’b0	6’h0D	ANI13	ANI14	ANI15
Others		Disable from setting.		

○ 2-channel scan mode (ADM1. ADMD=1)

ADISS	ADS[5:0]	Analog input channel	
		Scan 0	Scan 1
1’b0	6’h00	ANI0	ANI1
1’b0	6’h01	ANI1	ANI2
1’b0	6’h02	ANI2	ANI3
1’b0	6’h03	ANI3	ANI4
1’b0	6’h04	ANI4	ANI5
1’b0	6’h05	ANI5	ANI6
1’b0	6’h06	ANI6	ANI7
1’b0	6’h07	ANI7	ANI8
1’b0	6’h08	ANI8	ANI9
1’b0	6’h09	ANI9	ANI10
1’b0	6’h0A	ANI10	ANI11
1’b0	6’h0B	ANI11	ANI12
1’b0	6’h0C	ANI12	ANI13
1’b0	6’h0D	ANI13	ANI14
1’b0	6’h0E	ANI14	ANI15
Others		Disable settings.	

Note

1. A/D conversion can be performed by ADS for ports that are set by the PMCxx register as analog inputs.
2. To override the ADISS bit, it must be done in the transition stop state (ADCS=0, ADCE=0).
3. When AV$_{REFP}$ is used as the positive (+) reference voltage of the A/D converter, ANI2 cannot be selected as the A/D conversion channel.
4. When AV$_{REFM}$ is used as the negative (–) reference voltage of the A/D converter, ANI3 cannot be selected as the A/D conversion channel.
5. After setting the ADISS bit to "1", the result of the first conversion cannot be used. For detailed set-up procedures, refer to "14.5.4 Settings when selecting the output voltage/internal reference voltage of temperaturesensor"
6. The ADISS bit cannot be set to "1" when shifting to deep sleep mode or when shifting to sleep mode while the CPU is running on the subsystem clock.

14.2.7 12-bit A/D conversion result register (ADCR)

This is a 16-bit register that holds the A/D conversion result, which is readable only. Each time the A/D conversion ends, the Translation Result Note is loaded from the Successive Approximation Register (SAR) Note.

The high 4-bit readout value of the register is fixed to '0' when selecting mode, and the channel number of this conversion result can be configured by ADM2.CHRDE=1.

The ADCR register is read by a 16-bit memory operation instruction.

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

Note: A/D conversion results are not saved if their values are not within the set values of the A/D conversion results comparison function (set through the ADRCK bit and the ADUL/ADLL register.

Figure 14-10 Format of 12-bit A/D conversion result register (ADCR)

Reset value: 0000H R

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCR	ADCH3	ADCH2	ADCH1	ADCH0	ADCR[11:0]											

Note 1. If only 8-bit resolution A/D conversion results are required, the high 8-bit conversion results can be read through the ADCRH register.

2. When 16-bit access to the ADCR register, the high 12 bits of the translation result can be read in sequence from bit11.

○ Select mode (ADM1.ADMD=0)

The readout value of ADCH0~3 is fixed to 4'b0000

○ Scan mode (ADM1.ADMD=1) and ADM2.CHRDE=1,ADCH0~3 read-out values are as follows:

ADCH3	ADCH2	ADCH1	ADCH0	Conversion channel ID
0	0	0	0	ANI0
0	0	0	1	ANI1
0	0	1	0	ANI2
0	0	1	1	ANI3
0	1	0	0	ANI4
0	1	0	1	ANI5
0	1	1	0	ANI6
0	1	1	1	ANI7
1	0	0	0	ANI8
1	0	0	1	ANI9
1	0	1	0	ANI10
1	0	1	1	ANI11
1	1	0	0	ANI12
1	1	0	1	ANI13
1	1	1	0	ANI14
1	1	1	1	ANI15

14.2.8 8-bit A/D conversion result register (ADCRH)

This is an 8-bit register that holds the A/D conversion result, holding a high 8-bit note with 12-bit resolution Note.

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

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

Note: A/D conversion results are not saved if their values are not within the set values of the A/D conversion results comparison function (set through the ADRCK bit and the ADUL/ADLL register).

Figure 14-11 Format of 8-bit A/D conversion result register (ADCRH)

Reset value: 00HR

	7	6	5	4	3	2	1	0
ADCRH								

Note: The result of the conversion must be read after the conversion is completed and before the ADM0, ADS registers are configured. Otherwise, you may not be able to read the correct conversion results.

14.2.9 Conversion result comparison lower limit setting register (ADUL)

This is the setting register for checking the upper limit value of the A/D conversion result.

The A/D conversion result is compared with the value of the ADUL register, and the ADRCK in the mode register 2 (ADM2) of the A/D converter

The generation of the interrupt signal (INTAD) is controlled within the setting range of the bit (refer to Figure 14-7 ADRCK bit interrupt signal generation range). The ADUL register is set by an 8-bit memory manipulation instruction.

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

Note 1. Only the higher 8 bits of the 12-bit A/D conversion result register (ADCR) are compared with the ADUL register and the ADLL register.

2. Rewrite the value of the ADUL register and ADLL register while conversion is stopped (ADCS = 0).

3. Rewrite the value of the ADUL register and ADLL register while ADUL＞ADLL.

Figure 14-12 Format of conversion result comparison upper limit setting register (ADUL)

Reset value: FFH R/W

	7	6	5	4	3	2	1	0
ADUL	ADUL7	ADUL6	ADUL5	ADUL4	ADUL3	ADUL2	ADUL1	ADUL0

14.2.10 Conversion result comparison lower limit setting register (ADLL)

This is the set-up register used to check the lower limit of the A/D conversion result.

The A/D conversion results and ADLL register value are compared, and interrupt signal (INTAD) generation is controlled in the range specified by the ADRCK bit of A/D converter mode register 2 (ADM2) (shown in Figure 14-7: ADRCK bit interrupt signal generation range). The ADLL register is set by an 8-bit memory manipulation instruction.

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

Figure 14-13 Format of conversion result comparison lower limit setting register (ADLL)

Reset value: 00H R/W

	7	6	5	4	3	2	1	0
ADLL	ADLL7	ADLL6	ADLL5	ADLL4	ADLL3	ADLL2	ADLL1	ADLL0

Note:1. Only the higher 8 bits of the 12-bit A/D conversion result register (ADCR) are compared with the ADUL register and the ADLL register.

2. Rewrite the value of the ADUL register and ADLL register while conversion is stopped (ADCS = 0).

3. Rewrite the value of the ADUL register and ADLL register while ADUL＞ADLL.

14.2.11 A/D converter sampling time control register (ADNSMP)

This register controls the A/D sampling time.
The ADNSMP register is set by an 8-bit memory operation instruction.
After the reset signal is generated, the value of this register changes to '0dH'.

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

Reset value: 0dH R/W

	7	6	5	4	3	2	1	0
ADNSMP	ADNSMP[7:0]							

Sample clock setting:

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

Note: Rewrite the value of the ADNSMP register while conversion is stopped (ADCS=0).

Time required to perform an ADC conversion:

High-speed conversion mode: ADC conversion time = (number of sampling clocks + number of successive comparison clocks (31.5))/F_{AD}

Low-current conversion mode: ADC conversion time = (number of sampling clocks + number of successive comparison clocks (40.5))/F_{AD}

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

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

Sampling time calculation equations: Number of sampling clocks /f_{AD} ⩾ recommended sampling time

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

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

14.2.12 A/D sample time extension register (ADSMPWAIT)

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

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

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

Figure 14-15 Format of A/D sample time extension register (ADSMPWAIT)

Reset Value: 00H R/W

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

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

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

14.2.13 A/D test register (ADTES)

This register is used to set the test mode of the A/D converter.
The ADTES register is set by an 8-bit memory operation instruction.
After the reset signal is generated, the value of this register changes to "00H".

Figure 14-16 Format of A/D test register (ADTES)

Reset Value: 00H R/W

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

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

14.2.14 A/D status register (ADFLG)

This register represents the state of the A/D converter.
The ADFLG register is read by an 8-bit memory operation instruction.
After the reset signal is generated, the value of this register changes to "00H".

Figure 14-17 Format of A/D status register (ADFLG)

Reset value: 00HR

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

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

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

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

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

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

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

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

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

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

Figure 14-18 Format of A/D charge/discharge control register (ADNDIS)

Reset value: 00HW

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

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

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

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

14.2.16 Programmable gain amplifier sample and hold control register (PGA0SH)

The PGA0SH register is used to control the programmable gain amplifier sample hold function. When PAG0 and ADC are linked, this register can be set to control the output of PGA0 to hold after a period of sampling time, while the ADC automatically starts conversion after the PGA output is held, and PGA0 returns to the sampling state after the conversion is completed.

Before operating the PGA0SH register, bit 0 (ADCEN) of the peripheral enable register PER1 must be set to "1".

The PGA0SH register can be set by a 16-bit memory operation instruction. The reset value of this register is 0000H after the reset signal is generated.

Figure 14-19 Foramt of programmable gain amplifier sample and hold control register (PGA0SH)

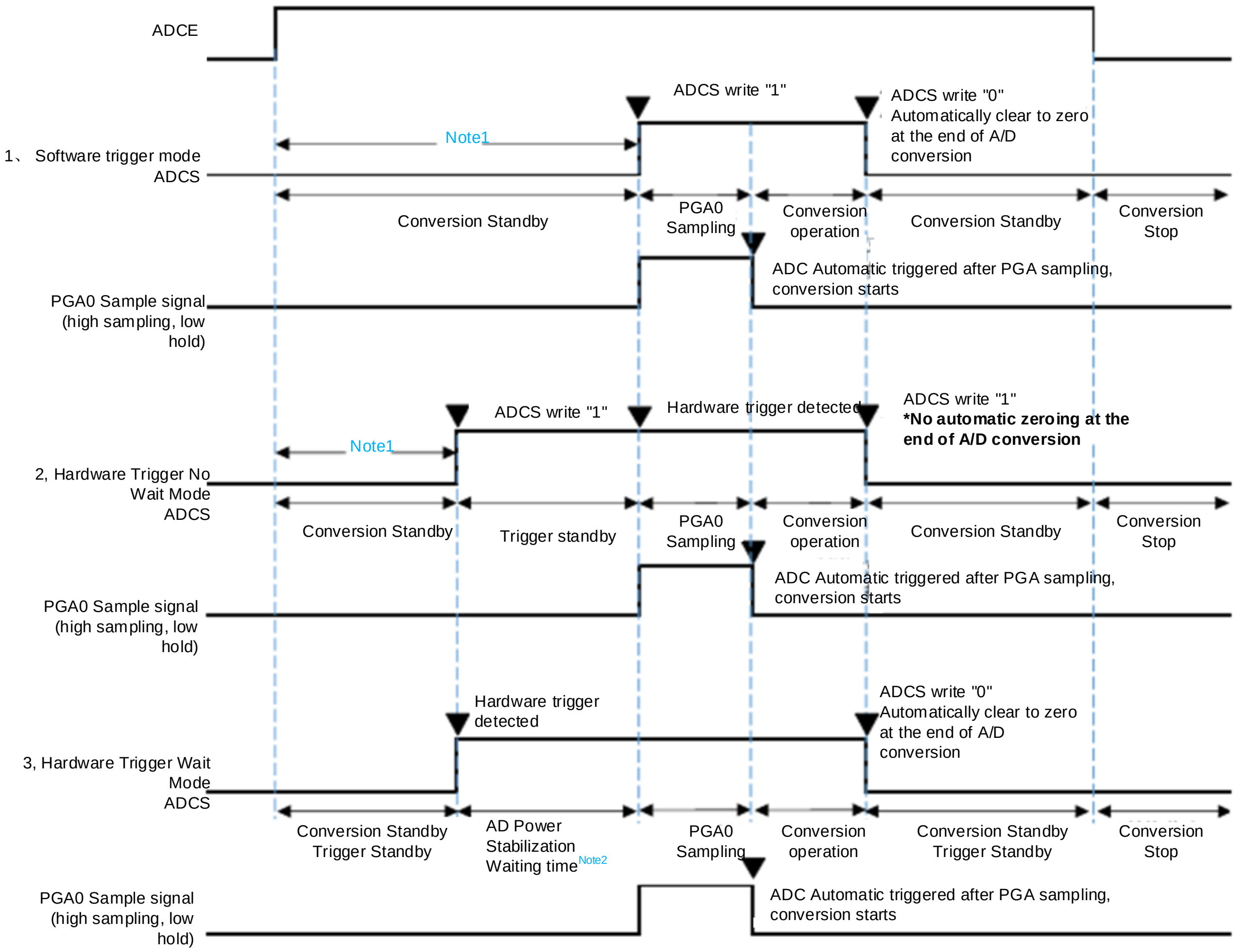
Address: 0x4004501E
Reset value: 00H R/W

	15	14	13	12	11	10	9	8
PGA0SH	PGA0SHEN	-	-	-	-	-	PGA0SH9	PGA0SH8
	7	6	5	4	3	2	1	0
	PGA0SH7	PGA0SH6	PGA0SH5	PGA0SH4	PGA0SH3	PGA0SH2	PGA0SH1	PGA0SH0

PGA0SHEN	Programmable gain amplifier sample hold control
0	Sample and hold function is disabled, and PAG0 is in normal amplified mode.
1	Sample and hold function is enabled, The ADC conversion starts after the PGA output is held, and the PGA sampling time is selected by PGA0SH9~0.

PGA0SH9~0	PGA0 sampling time selection
0	1 x T_{AD} (Action clock period of ADC)
1	2 x T_{AD}
2	3 x T_{AD}
3	4 x T_{AD}
……	
0x3fe	1203 x T_{AD}
0x3ff	Prohibit settings

Figure 14-20 Action status diagram when using various modes of A/D (converting PGA0 channel and using PGA sample hold function)


ADCE
1、 Software trigger mode
ADCS
Note1
ADCS write "1"
ADCS write "0"
Automatically clear to zero at the end of A/D conversion
Conversion Standby
PGA0 Sampling
Conversion operation
Conversion Standby
Conversion Stop
ADC Automatic triggered after PGA sampling, conversion starts
PGA0 Sample signal (high sampling, low hold)
2, Hardware Trigger No Wait Mode
ADCS
Note1
ADCS write "1"
Hardware trigger detected
ADCS write "1"
*No automatic zeroing at the end of A/D conversion
Conversion Standby
Trigger standby
PGA0 Sampling
Conversion operation
Conversion Standby
Conversion Stop
ADC Automatic triggered after PGA sampling, conversion starts
PGA0 Sample signal (high sampling, low hold)
3, Hardware Trigger Wait Mode
ADCS
Hardware trigger detected
ADCS write "0"
Automatically clear to zero at the end of A/D conversion
Conversion Standby
Trigger Standby
AD Power Stabilization Waiting time Note2
PGA0 Sampling
Conversion operation
Conversion Standby
Trigger Standby
Conversion Stop
ADC Automatic triggered after PGA sampling, conversion starts
PGA0 Sample signal (high sampling, low hold)

14.2.17 Registers for controlling analog input pin port function

When using the ANIx pin as the analog input of the A/D converter, the port must be configured as an analog channel by setting the corresponding Port Mode Control Register (PMCxx) bit to "1". For details, please refer to "Chapter 2 Port Function".

14.3 Input voltage and conversion results

The analog input voltage of the analog input pin (ANIx) and the theoretical A/D conversion result register (ADCR) are related.

$$ADCR=INT(\frac{V_{AIN}}{AV_{REF}}\times4096+0.5) \text{ or } (ADCR-0.5)\times\frac{AV_{REF}}{4096}\leq V_{AIN}<(ADCR+0.5)\times\frac{AV_{REF}}{4096}$$

INT(): Function that returns the integer part of the value in parentheses

$_{VAIN}$: analog input voltage

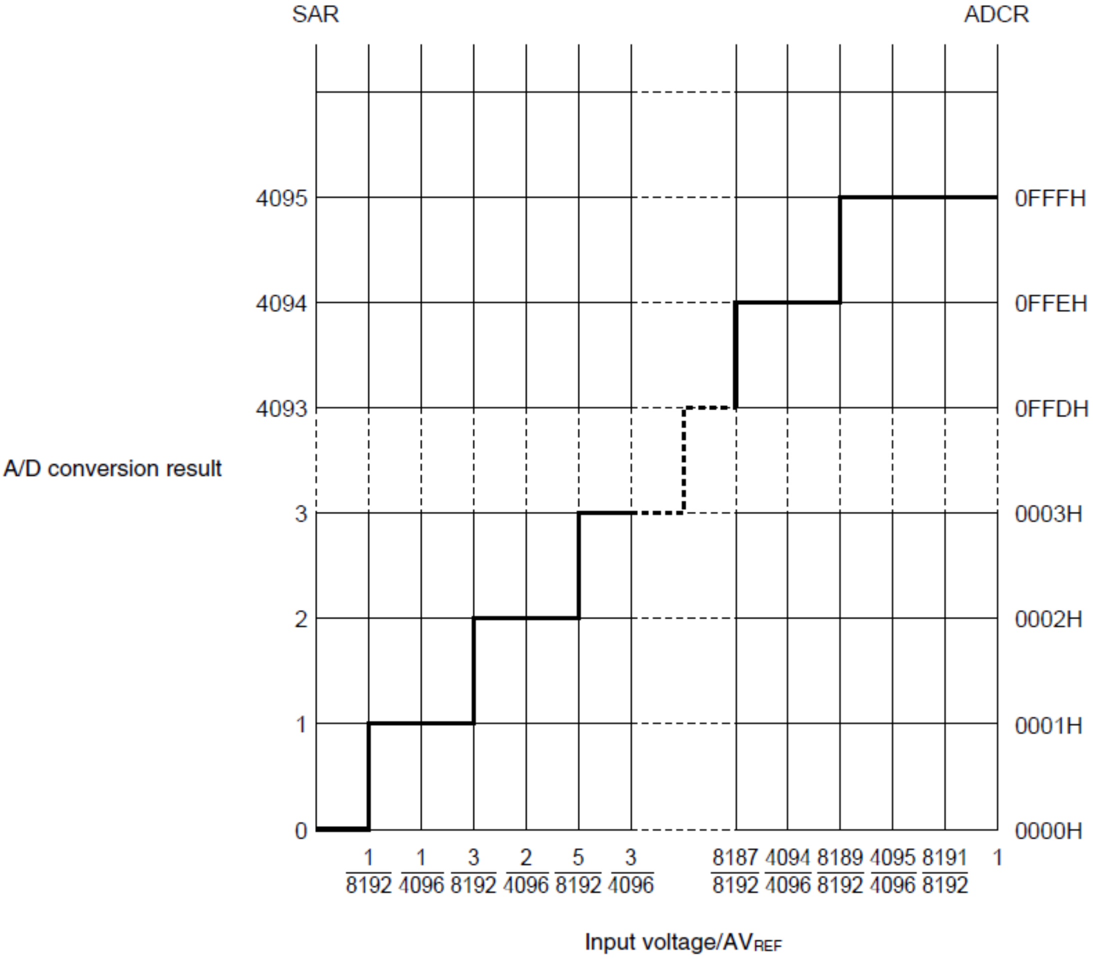
$_{AVREF}$:AVREF Pin Voltage

ADCR:The value of the A/D conversion result register (ADCR)

SAR: successive approximation register

The relationship between the simulated input voltage and the A/D conversion result is shown in the figure below.

Figure 14-21 Analog input voltage vs. A/D conversion result



Note: AV_{REF} is the positive (+) reference voltage of the A/D converter, either AV_{REFP} or V_{DD} can be selected.

14.4 Operation mode of A/D converter

The modes of the A/D converter are operated as follows. Refer to the "14.5 A/D Converter Set-up Flowchart" for set-up procedures.

14.4.1 Software trigger mode (select mode, continuous conversion mode)

① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to "1”.

② After the software counts up to the stabilization wait time (1 μs), the ADCS bit of the ADM0 register is set to 1 to perform the A/D conversion of the analog input specified by the analog input channel specification register (ADS).

③ If the A/D conversion ends, the conversion result is saved to the A/D conversion result register (ADCR, ADCRH) and the A/D conversion end interrupt request signal (INTAD). The next A/D conversion begins immediately after the A/D conversion.

④ If the ADCS bit is overridden "1" during the conversion, the current A/D conversion immediately aborts and starts again.

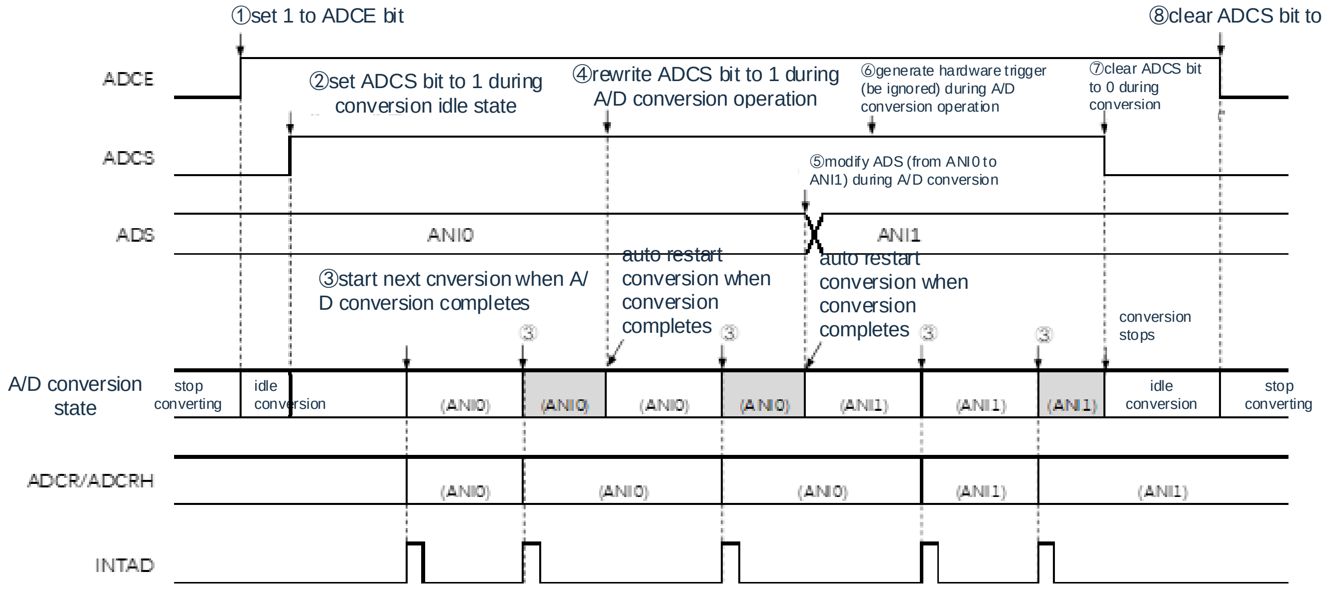
⑤ If the ADS register is rewritten or rewritten during the conversion, the current A/D conversion is immediately aborted and then A/D is converted.

⑥ An A/D conversion does not start even if a hardware trigger is entered during the conversion.

⑦ When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status.

⑧ When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start.

Figure 14-22 Example of software trigger mode (select mode, sequential conversion mode) operation timing


①set 1 to ADCE bit
⑧clear ADCS bit to 0
ADCE
②set ADCS bit to 1 during conversion idle state
④rewrite ADCS bit to 1 during A/D conversion operation
⑥generate hardware trigger (be ignored) during A/D conversion operation
⑦clear ADCS bit to 0 during conversion
ADCS
⑤modify ADS (from ANI0 to ANI1) during A/D conversion
ADS
ANI0
ANI1
③start next cnversion when A/D conversion completes
auto restart conversion when conversion completes
③
auto restart conversion when conversion completes
③
③
③
conversion stops
A/D conversion state
stop converting
idle conversion
(ANI0)
(ANI0)
(ANI0)
(ANI0)
(ANI1)
(ANI1)
(ANI1)
idle conversion
stop converting
ADCR/ADCRH
(ANI0)
(ANI0)
(ANI0)
(ANI1)
(ANI1)
INTAD

14.4.2 Software trigger mode (select mode, single conversion mode)

① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to "1” .

② After the software counts up to the stabilization wait time (1μs), the ADCS bit of the ADM0 register is set to 1 to perform the A/D conversion of the analog input specified by the analog input channel specification register (ADS).

③ If the A/D conversion ends, the conversion result is saved to the A/D conversion result register (ADCR, ADCRH) and the A/D conversion end interrupt request signal (INTAD).

④ After the A/D conversion, the ADCS bit automatically clears "0" and enters the A/D conversion standby state.

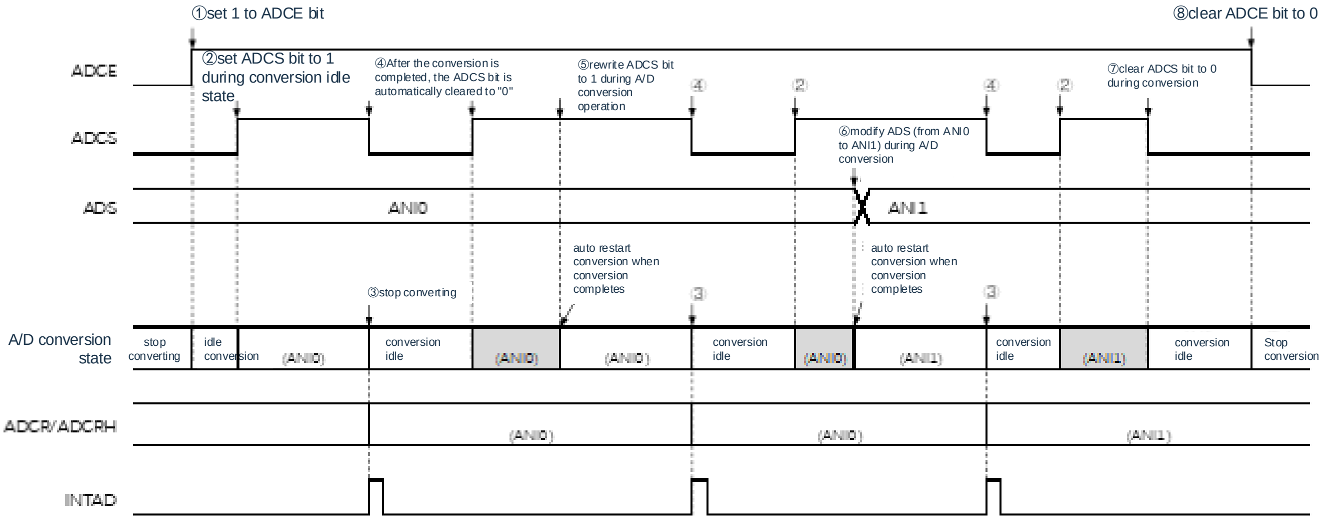
⑤ If the ADCS bit is overridden "1" during the conversion, the current A/D conversion immediately aborts and starts again.

⑥ If the ADS register is rewritten or rewritten during the conversion, the current A/D conversion is immediately aborted and then A/D is converted.

⑦ If the ADCS bit is set to "0" during the conversion process, the current A/D conversion is immediately suspended and the A/D conversion is put into standby mode.

⑧ When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start. In addition, A/D conversion does not start even if a hardware trigger is input while in the A/D conversion standby status.

Figure 14-23 Example of software trigger mode (select mode, single conversion mode) operation timing


①set 1 to ADCE bit
⑧clear ADCE bit to 0
ADCE
②set ADCS bit to 1 during conversion idle state
④After the conversion is completed, the ADCS bit is automatically cleared to "0"
⑤rewrite ADCS bit to 1 during A/D conversion operation
④
②
④
②
⑦clear ADCS bit to 0 during conversion
ADCS
⑥modify ADS (from ANI0 to ANI1) during A/D conversion
ADS
ANI0
ANI1
auto restart conversion when conversion completes
③stop converting
auto restart conversion when conversion completes
③
③
A/D conversion state
stop converting
idle conversion
(ANI0)
conversion idle
(ANI0)
(ANI0)
conversion idle
(ANI0)
(ANI1)
conversion idle
(ANI1)
conversion idle
Stop conversion
ADCR/ADCRH
(ANI0)
(ANI0)
(ANI1)
INTAD

14.4.3 Software trigger mode (scan mode, continuous conversion mode)

① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to "1" .

② After the stable waiting time (1μs) is counted by software, the ADCS bit of the ADM0 register is 1 to perform A/D conversion. The A/D conversion is performed from the analog input channel specified by the scan 0.

③ A/D conversion of 4 analog input channels is performed. Each time the A/D conversion ends, the conversion result is saved to the A/D conversion result register ADCR, ADCRH, and the A/D conversion end interrupt request signal is INTAD. The next A/D conversion (4 channels) automatically starts from the set channel immediately after the A/D conversion of the 4 channels is completed.

④ If the ADCS bit is overridden "1" during the conversion, the current A/D conversion immediately aborts and starts again.

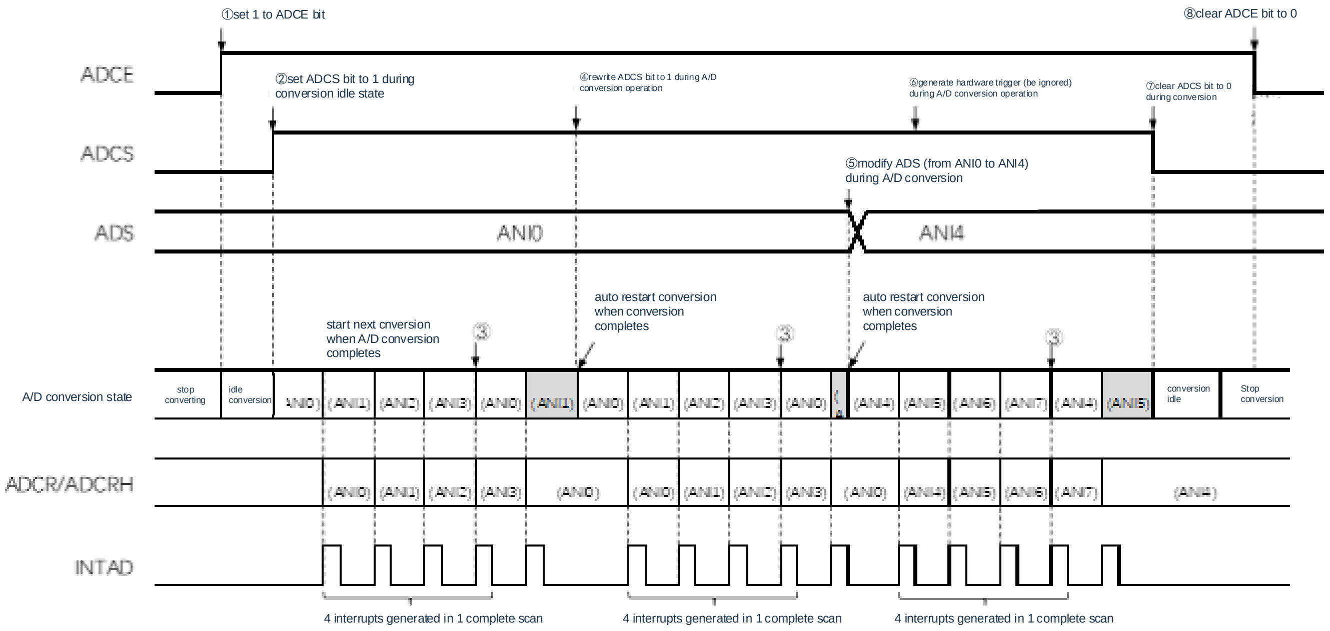
⑤ If the ADS register is rewritten or rewritten during the conversion, the current A/D conversion is immediately aborted and then A/D from the initial channel redesignated.

⑥ An A/D conversion does not start even if a hardware trigger is entered during the conversion.

⑦ When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status.

⑧ When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start.

Figure 14-24 Example of software trigger mode (scan mode, continuous conversion mode) operation timing


①set 1 to ADCE bit
⑧clear ADCE bit to 0
ADCE
②set ADCS bit to 1 during conversion idle state
④rewrite ADCS bit to 1 during A/D conversion operation
⑥generate hardware trigger (be ignored) during A/D conversion operation
⑦clear ADCS bit to 0 during conversion
ADCS
⑤modify ADS (from ANI0 to ANI4) during A/D conversion
ADS
ANI0
ANI4
start next cnversion when A/D conversion completes
③
auto restart conversion when conversion completes
③
auto restart conversion when conversion completes
③
A/D conversion state
stop converting
idle conversion
ANI0)
(ANI1)
(ANI2)
(ANI3)
(ANI0)
(ANI1)
(ANI0)
(ANI1)
(ANI2)
(ANI3)
(ANI0)
(A
(ANI4)
(ANI5)
(ANI6)
(ANI7)
(ANI4)
(ANI5)
conversion idle
Stop conversion
ADCR/ADCRH
(ANI0)
(ANI1)
(ANI2)
(ANI3)
(ANI0)
(ANI0)
(ANI1)
(ANI2)
(ANI3)
(ANI0)
(ANI4)
(ANI5)
(ANI6)
(ANI7)
(ANI4)
INTAD
4 interrupts generated in 1 complete scan
4 interrupts generated in 1 complete scan
4 interrupts generated in 1 complete scan

14.4.4 Software trigger mode (scan mode, single conversion mode)

① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to “1” .

② After the stabilization wait time (1μs) is counted by software, set the ADCS bit of the ADM0 register to "1" and perform A/D conversion for the four analog input channels from scan 0 to scan 3 specified by the analog input channel target register (ADS). A/D conversion is performed sequentially from the analog input channel designated by scan 0.

③ A/D conversion of 4 analog input channels is performed. Each time the A/D conversion ends, the conversion result is saved to the A/D conversion result register ADCR, ADCRH, and the A/D conversion end interrupt request signal is INTAD.

④ The ADCS bit automatically clears "0" after the A/D conversion of the 4 channels and enters the A/D conversion standby.

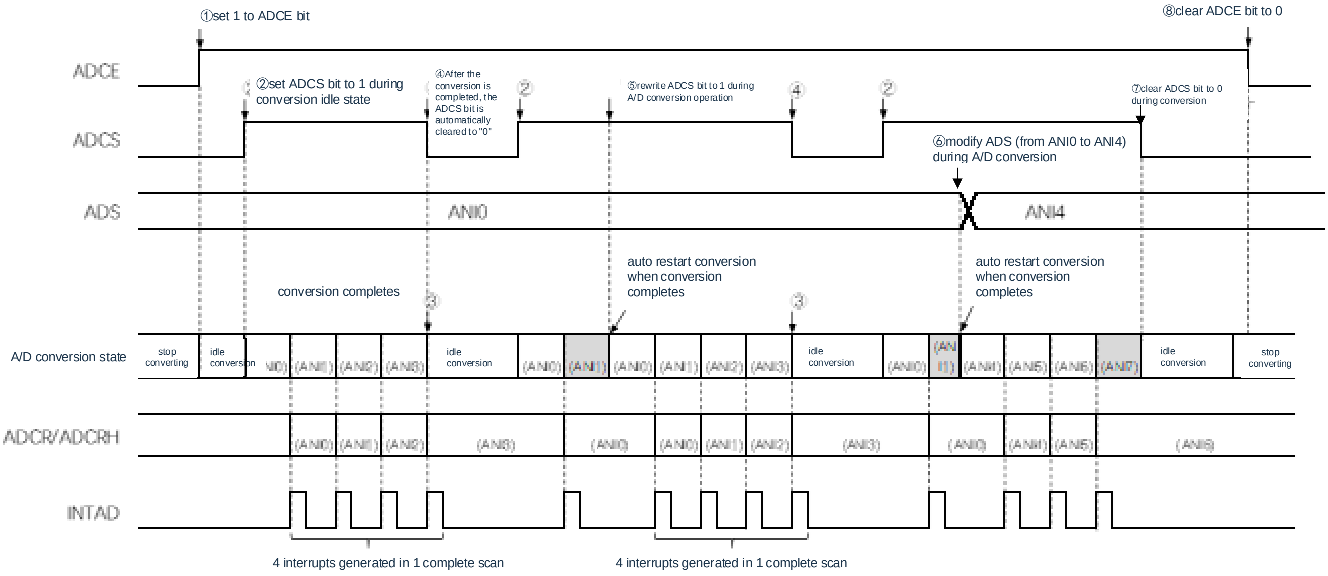
⑤ If the ADCS bit is overridden "1" during the conversion, the current A/D conversion immediately aborts and starts again.

⑥ If the ADS register is rewritten or rewritten during the conversion, the current A/D conversion is immediately aborted and then A/D from the initial channel redesignated.

⑦ When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status.

⑧ When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start. The A/D conversion does not start even if the hardware trigger is entered while the A/D conversion is standby.

Figure 14-25 Example of software trigger mode (scan mode, single conversion mode) operation timing


①set 1 to ADCE bit
⑧clear ADCE bit to 0
ADCE
ADCS
ADS
A/D conversion state
ADCR/ADCRH
INTAD
②set ADCS bit to 1 during conversion idle state
④After the conversion is completed, the ADCS bit is automatically cleared to "0"
②
⑤rewrite ADCS bit to 1 during A/D conversion operation
④
②
⑦clear ADCS bit to 0 during conversion
⑥modify ADS (from ANI0 to ANI4) during A/D conversion
ANI0
ANI4
conversion completes
③
auto restart conversion when conversion completes
③
auto restart conversion when conversion completes
stop converting
idle conversion
(ANI0)
(ANI1)
(ANI2)
(ANI3)
idle conversion
(ANI0)
(ANI1)
(ANI0)
(ANI1)
(ANI2)
(ANI3)
idle conversion
(ANI0)
(ANI1)
(ANI4)
(ANI5)
(ANI6)
(ANI7)
idle conversion
stop converting
(ANI0)
(ANI1)
(ANI2)
(ANI3)
(ANI0)
(ANI0)
(ANI1)
(ANI2)
(ANI3)
(ANI0)
(ANI4)
(ANI5)
(ANI6)
4 interrupts generated in 1 complete scan
4 interrupts generated in 1 complete scan

14.4.5 Hardware trigger no-wait mode (select mode, continuous conversion mode)

① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to “1” .

② After the stabilization wait time (1μs) is counted by software, set the ADCS bit of the ADM0 register to "1" to enter the hardware-triggered standby state (conversion does not start at this stage). In the hardware-triggered standby state, A/D conversion does not start even if ADCS is set to "1".

③ If a hardware trigger is entered with the ADCS bit '1', an A/D conversion is performed on analog inputs specified by analog input channel specification registers.

④ If the A/D conversion ends, the conversion result is saved to the A/D conversion result register (ADCR, ADCRH) and the A/D conversion end interrupt request signal (INTAD). The next A/D conversion begins immediately after the A/D conversion.

⑤ If you enter a hardware trigger during the conversion, the current A/D conversion is immediately aborted and then restarted.

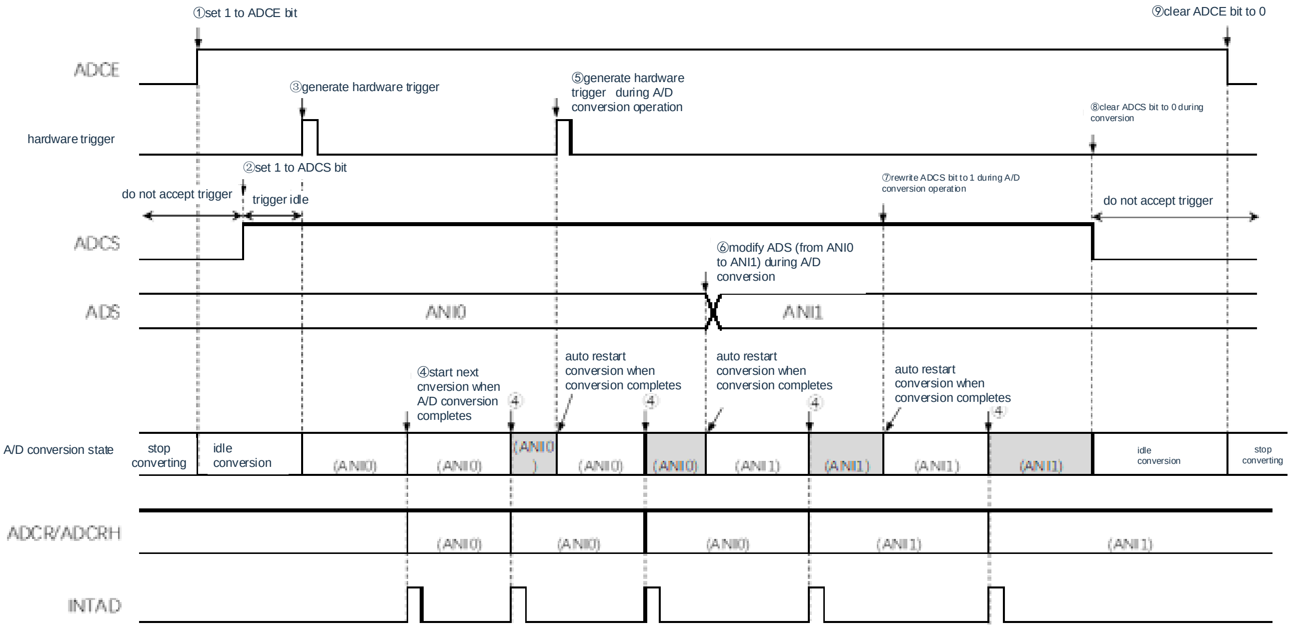
⑥ If the ADS register is rewritten or rewritten during the conversion, the current A/D conversion is immediately aborted and then A/D is converted.

⑦ If the ADCS bit is overridden "1" during the conversion, the current A/D conversion immediately aborts and starts again.

⑧ When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status. However, the A/D converter does not stop in this status.

⑨ When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCS = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

Figure 14-26 Example of hardware trigger no-wait mode (select mode, continuous conversion mode) operation timing


①set 1 to ADCE bit
⑨clear ADCE bit to 0
ADCE
③generate hardware trigger
⑤generate hardware trigger during A/D conversion operation
⑧clear ADCS bit to 0 during conversion
hardware trigger
②set 1 to ADCS bit
do not accept trigger
trigger idle
⑦rewrite ADCS bit to 1 during A/D conversion operation
do not accept trigger
ADCS
⑥modify ADS (from ANI0 to ANI1) during A/D conversion
ADS
ANI0
ANI1
④start next cnversion when A/D conversion completes
auto restart conversion when conversion completes
auto restart conversion when conversion completes
auto restart conversion when conversion completes
④
A/D conversion state
stop converting
idle conversion
(ANI0)
(ANI1)
idle conversion
stop converting
ADCR/ADCRH
(ANI0)
(ANI1)
INTAD

14.4.6 Hardware trigger no-wait mode (select mode, single conversion mode)

① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to “1” .

② After the stabilization wait time (1 μ s) is counted by software, set the ADCS bit of the ADM0 register to "1" to enter the hardware-triggered standby state (conversion does not start at this stage). In the hardware-triggered standby state, A/D conversion does not start even if the ADCS bit is set to "1".

③ If a hardware trigger is entered with the ADCS bit '1', an A/D conversion is performed on analog inputs specified by analog input channel specification registers.

④ If the A/D conversion ends, the conversion result is saved to the A/D conversion result register (ADCR, ADCRH) and the A/D conversion end interrupt request signal (INTAD).

⑤ After the A/D conversion is complete, the ADCS bit remains in the "1" state and enters A/D.

⑥ If you enter a hardware trigger during the conversion, the current A/D conversion is immediately aborted and then restarted.

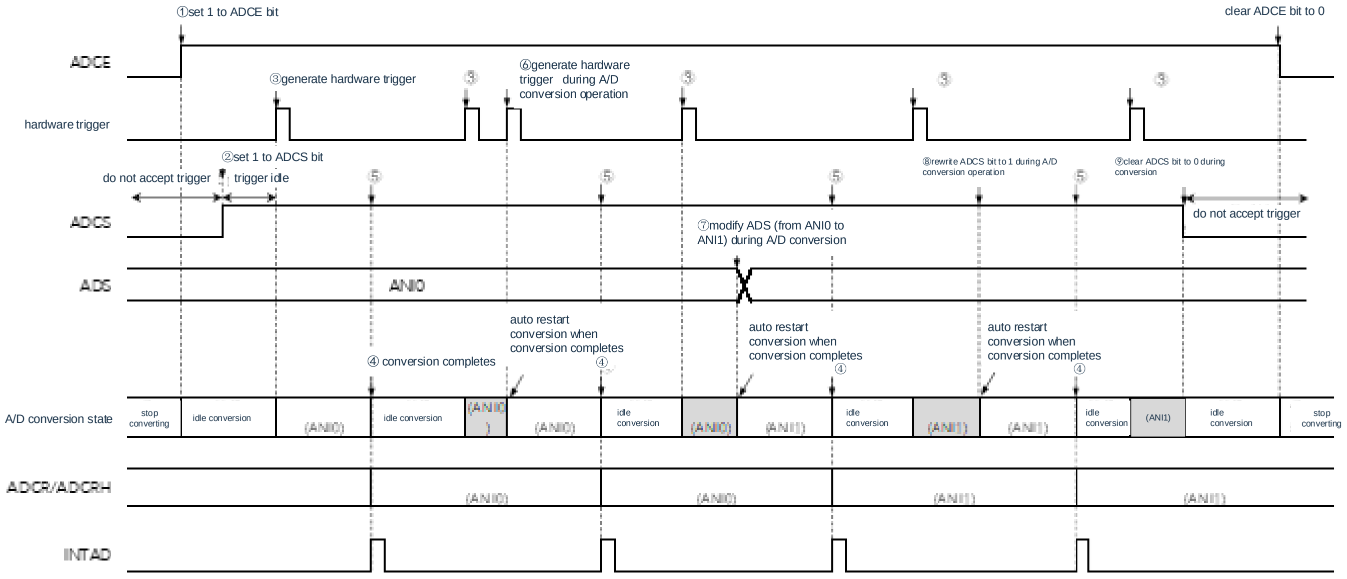
⑦ If the ADS register is rewritten or rewritten during the conversion, the current A/D conversion is immediately aborted and then A/D is converted.

⑧ If the ADCS bit is overridden "1" during the conversion, the current A/D conversion immediately aborts and starts again.

⑨ When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status. However, the A/D converter does not stop in this status.

⑩ When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCS = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

Figure 14-27 Example of hardware trigger no-wait mode (select mode, single conversion mode) operation timing



14.4.7 Hardware trigger no-wait mode (scan mode, continuous conversion mode)

① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to “1” .

② After the stabilization wait time (1 μ s) is counted by software, set the ADCS bit of the ADM0 register to "1" to enter the hardware-triggered standby state (conversion does not start at this stage). In the hardware-triggered standby state, A/D conversion does not start even if the ADCS bit is set to "1".

③ If a hardware trigger is entered in a state with ADCS bit '1', A/D conversion is performed on 4 analog input channels specified by analog input channel specification register (ADS). The A/D conversion is performed from the analog input channel specified by the scan 0.

④ A/D conversion of 4 analog input channels is performed. Each time the A/D conversion ends, the conversion result is saved to the A/D conversion result register ADCR, ADCRH, and the A/D conversion end interrupt request signal is INTAD. The next A/D conversion is automatically started from the set channel immediately after the A/D conversion of the 4 channels is completed.

⑤ If a hardware trigger is entered during the conversion, the current A/D conversion is aborted immediately and the conversion restarts from the original channel.

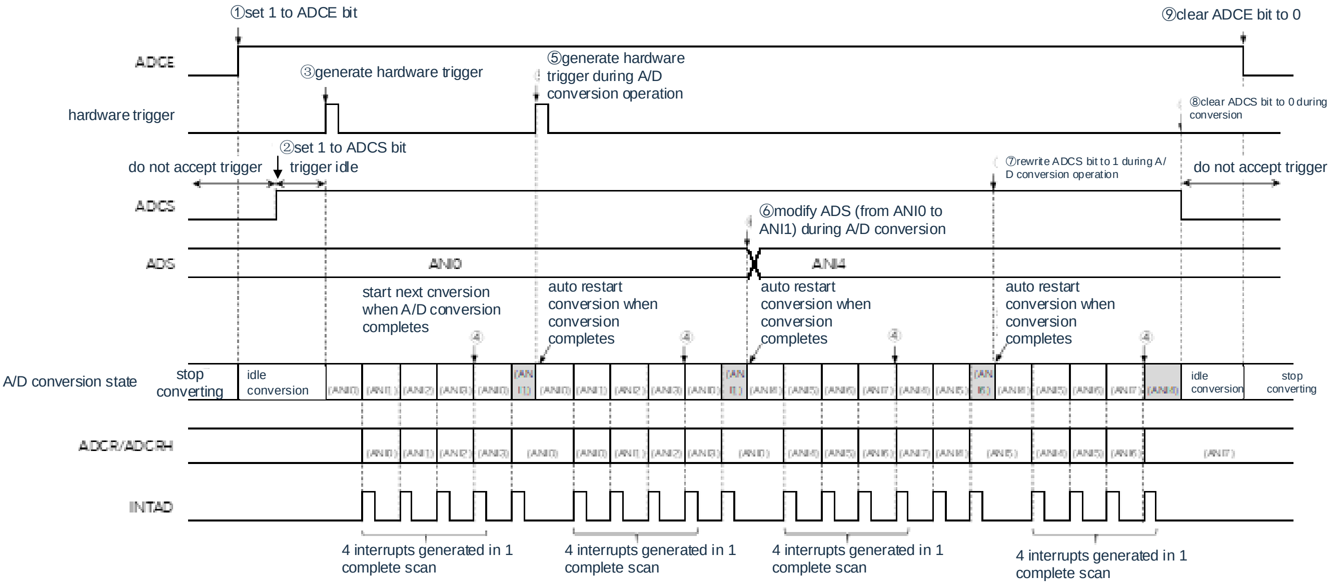
⑥ If the ADS register is rewritten or rewritten during the conversion, the current A/D conversion is immediately aborted and then A/D converted.

⑦ If the ADCS bit is overridden "1" during the conversion, the current A/D conversion immediately aborts and restarts from the original channel.

⑧ When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status. However, the A/D converter does not stop in this status.

⑨ When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start.

Figure 14-28 Example of hardware trigger no-wait mode (scan mode, continuous conversion mode) operation timing


①set 1 to ADCE bit
⑨clear ADCE bit to 0
ADCE
③generate hardware trigger
⑤generate hardware trigger during A/D conversion operation
hardware trigger
⑧clear ADCS bit to 0 during conversion
②set 1 to ADCS bit
do not accept trigger
trigger idle
⑦rewrite ADCS bit to 1 during A/D conversion operation
do not accept trigger
ADCS
⑥modify ADS (from ANI0 to ANI1) during A/D conversion
ADS
ANI0
ANI4
start next cnversion when A/D conversion completes
auto restart conversion when conversion completes
auto restart conversion when conversion completes
auto restart conversion when conversion completes
④
④
④
④
A/D conversion state
stop converting
idle conversion
idle conversion
stop converting
ADCR/ADCRH
INTAD
4 interrupts generated in 1 complete scan
4 interrupts generated in 1 complete scan
4 interrupts generated in 1 complete scan
4 interrupts generated in 1 complete scan

14.4.8 Hardware trigger no-wait mode (scan mode, single conversion mode)

① In the stop state, enter A/D conversion standby state by setting the ADCE bit of the A/D converter mode register 0 (ADM0) to “1” .

② After the stabilization wait time (1μs) is counted by software, set the ADCS bit of the ADM0 register to "1" to enter the hardware-triggered standby state (conversion does not start at this stage). In the hardware-triggered standby state, A/D conversion does not start even if the ADCS bit is set to "1".

③ If a hardware trigger is entered in a state with ADCS bit '1', A/D conversion is performed on 4 analog input channels specified by analog input channel specification register (ADS). The A/D conversion is performed from the analog input channel specified by the scan 0.

④ A/D conversion of 4 analog input channels is performed. Each time the A/D conversion ends, the conversion result is saved to the A/D conversion result register ADCR, ADCRH, and the A/D conversion end interrupt request signal is INTAD.

⑤ The ADCS bit remains in the "1" state after the A/D conversion of the 4 channels is completed and enters the A/D conversion standby state.

⑥ If a hardware trigger is entered during the conversion, the current A/D conversion is aborted immediately and the conversion restarts from the original channel.

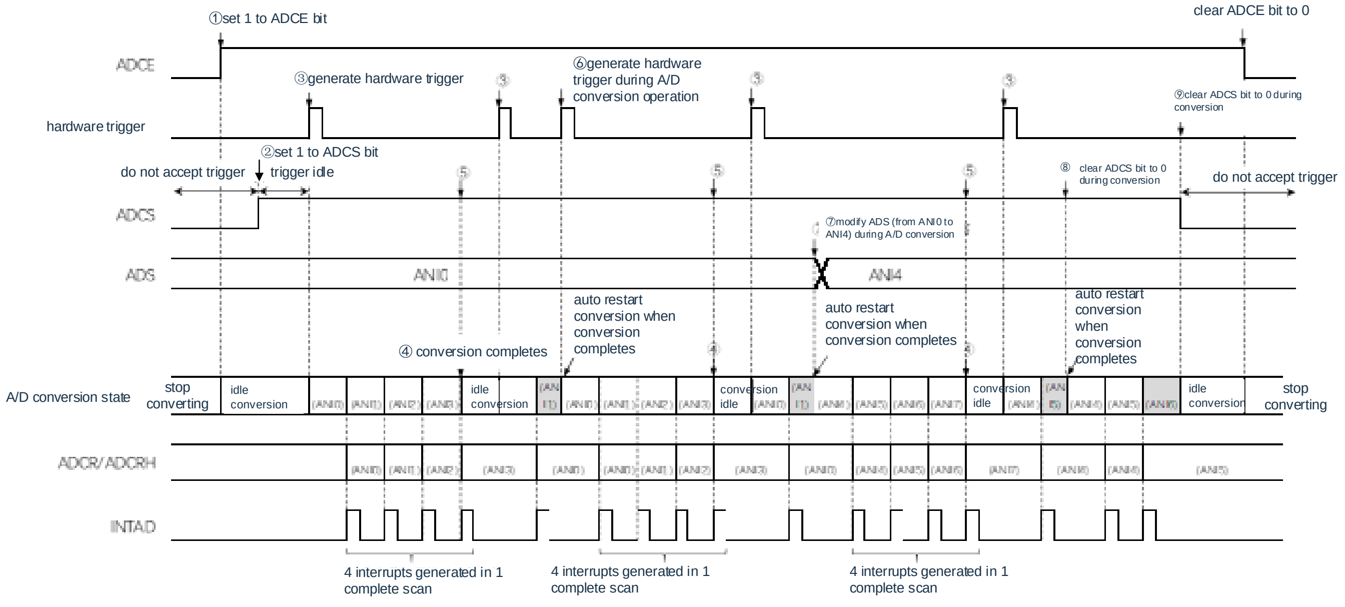
⑦ If the ADS register is rewritten or rewritten during the conversion, the current A/D conversion is immediately aborted and then A/D from the initial channel redesignated.

⑧ If the ADCS bit is overridden "1" during the conversion, the current A/D conversion is aborted immediately and then reconverted from the original channel.

⑨ When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status. However, the A/D converter does not stop in this status.

⑩ When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCS = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

Figure 14-29 Example of hardware trigger no-wait mode (scan mode, single conversion mode) operation timing


①set 1 to ADCE bit
clear ADCE bit to 0
ADCE
③generate hardware trigger
③
⑥generate hardware trigger during A/D conversion operation
③
③
⑨clear ADCS bit to 0 during conversion
hardware trigger
②set 1 to ADCS bit
do not accept trigger
trigger idle
⑤
⑤
⑤
⑧ clear ADCS bit to 0 during conversion
do not accept trigger
ADCS
⑦modify ADS (from ANI0 to ANI4) during A/D conversion
ADS
ANI0
ANI4
auto restart conversion when conversion completes
auto restart conversion when conversion completes
auto restart conversion when conversion completes
④ conversion completes
④
④
A/D conversion state
stop converting
idle conversion
(ANI0) (ANI1) (ANI2) (ANI3)
idle conversion
(ANI1)
(ANI0) (ANI1) (ANI2) (ANI3)
conversion idle
(ANI0) (ANI1)
(ANI4) (ANI5) (ANI6) (ANI7)
conversion idle
(ANI4) (ANI5)
(ANI4) (ANI5) (ANI6)
idle conversion
stop converting
ADCR/ADCRH
(ANI0) (ANI1) (ANI2) (ANI3)
(ANI0)
(ANI0) (ANI1) (ANI2) (ANI3)
(ANI0)
(ANI4) (ANI5) (ANI6) (ANI7)
(ANI4) (ANI4) (ANI5)
INTAD
4 interrupts generated in 1 complete scan
4 interrupts generated in 1 complete scan
4 interrupts generated in 1 complete scan

14.4.9 Hardware trigger wait mode (select mode, continuous conversion mode)

① In the stop state, the ADCE bit of mode register 0 (ADM0) of the A/D converter enters the hardware triggered standby state.

② If a hardware trigger is input in the hardware trigger standby state, A/D conversion is performed for the analog input specified by the analog input channel target register (ADS). The ADCS bit of the ADM0 register is automatically set to "1" while a hardware trigger is input.

③ If the A/D conversion ends, the conversion result is saved to the A/D conversion result register (ADCR, ADCRH) and the A/D conversion end interrupt request signal (INTAD). The next A/D conversion starts immediately after the A/D conversion (at which time no hardware trigger is required).

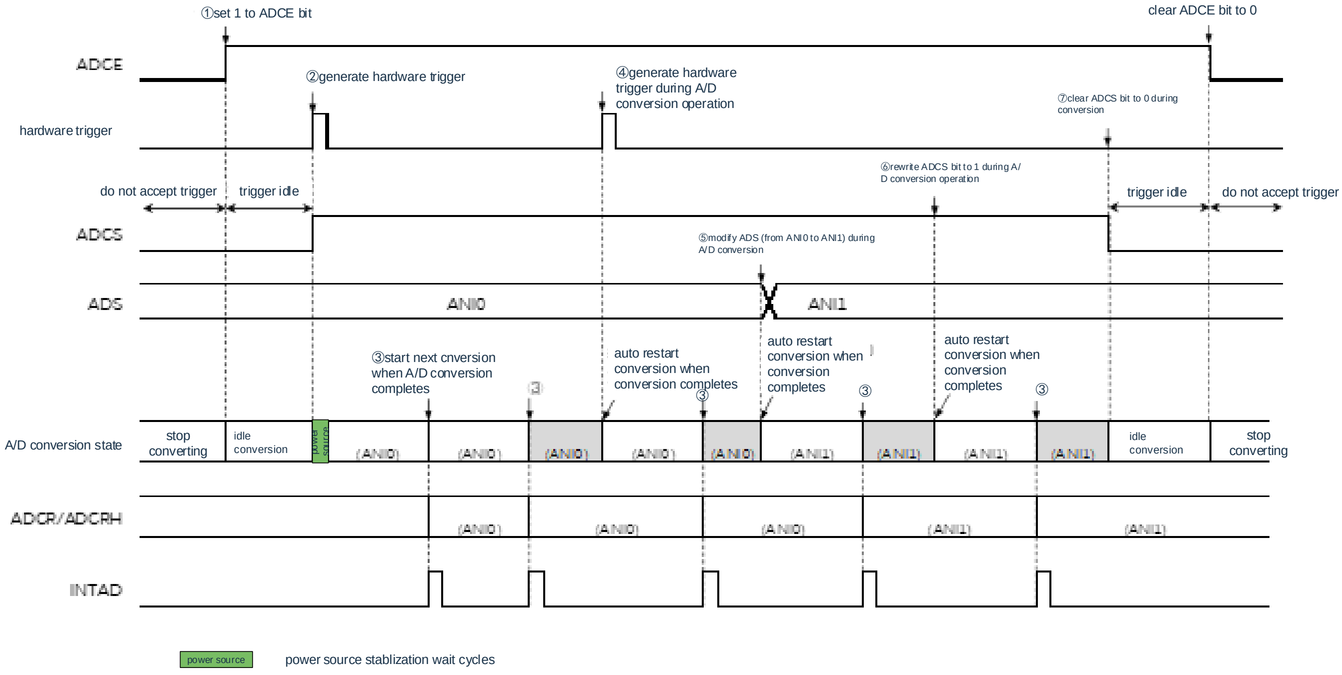
④ If you enter a hardware trigger during the conversion, the current A/D conversion is immediately aborted and then restarted.

⑤ If the ADS register is rewritten or rewritten during the conversion, the current A/D conversion is immediately aborted and then A/D is converted.

⑥ If the ADCS bit is overridden "1" during the conversion, the current A/D conversion immediately aborts and starts again.

⑦ When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, the system enters the hardware trigger standby status, and the A/D converter enters the stop status. When ADCE = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

Figure 14-30 Example of hardware trigger wait mode (select mode, sequential conversion mode) operation timing


①set 1 to ADCE bit
clear ADCE bit to 0
ADCE
②generate hardware trigger
④generate hardware trigger during A/D conversion operation
⑦clear ADCS bit to 0 during conversion
hardware trigger
⑥rewrite ADCS bit to 1 during A/D conversion operation
do not accept trigger
trigger idle
trigger idle
do not accept trigger
ADCS
⑤modify ADS (from ANI0 to ANI1) during A/D conversion
ADS
ANI0
ANI1
③start next cnversion when A/D conversion completes
③
auto restart conversion when conversion completes
③
auto restart conversion when conversion completes
③
auto restart conversion when conversion completes
③
A/D conversion state
stop converting
idle conversion
power source
(ANI0)
(ANI0)
(ANI0)
(ANI0)
(ANI0)
(ANI1)
(ANI1)
(ANI1)
(ANI1)
idle conversion
stop converting
ADCR/ADCRH
(ANI0)
(ANI0)
(ANI0)
(ANI1)
(ANI1)
INTAD
power source
power source stablization wait cycles

14.4.10 Hardware trigger wait mode (select mode, single conversion mode)

① In the stop state, the ADCE bit of mode register 0 (ADM0) of the A/D converter enters the hardware triggered standby state.

② If a hardware trigger is input in the hardware trigger standby state, A/D conversion is performed for the analog input specified by the analog input channel target register (ADS). The ADCS bit of the ADM0 register is automatically set to "1" while a hardware trigger is input.

③ If the A/D conversion ends, the conversion result is saved to the A/D conversion result register (ADCR, ADCRH) and the A/D conversion end interrupt request signal (INTAD).

④ After the A/D conversion is completed, the ADCS bit automatically clears "0", and the A/D converter goes to stop.

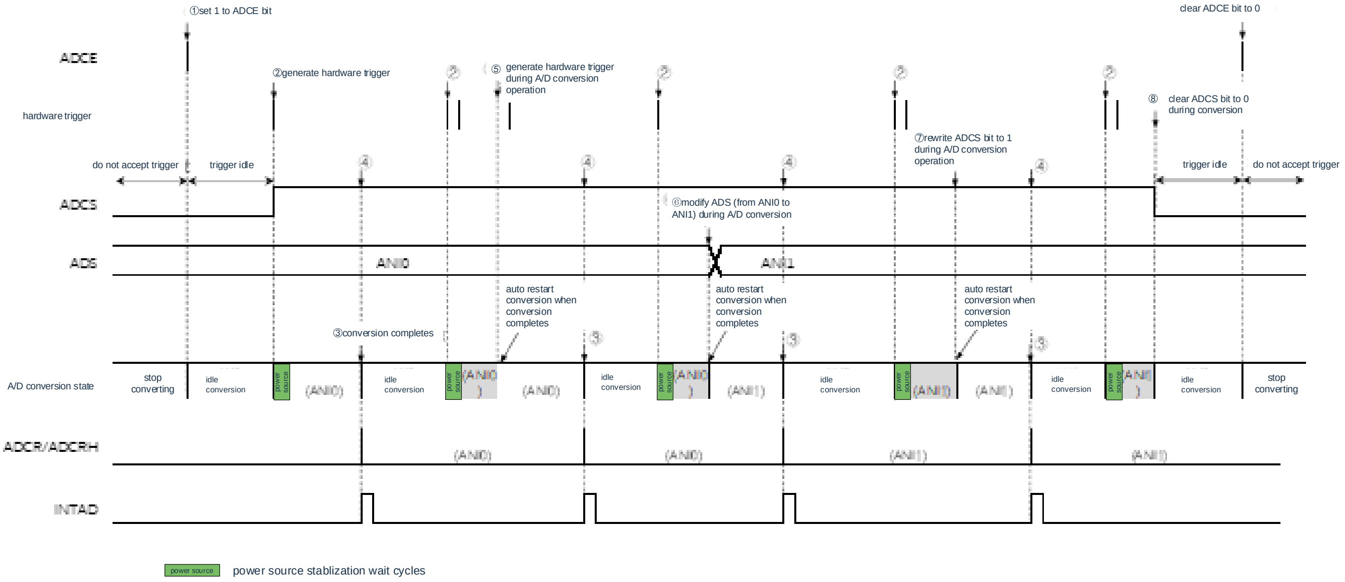
⑤ If you enter a hardware trigger during the conversion, the current A/D conversion is immediately aborted and then restarted.

⑥ If the ADS register is rewritten or rewritten during the conversion, the current A/D conversion is immediately aborted and then A/D is converted.

⑦ If the ADCS bit is overridden "1" during the conversion, the current A/D conversion immediately aborts and starts again.

⑧ When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, the system enters the hardware trigger standby status, and the A/D converter enters the stop status. When ADCE = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

Figure 14-31 Example of hardware trigger wait mode (select mode, single conversion mode) operation timing


①set 1 to ADCE bit
clear ADCE bit to 0
ADCE
②generate hardware trigger
⑤ generate hardware trigger during A/D conversion operation
hardware trigger
⑧ clear ADCS bit to 0 during conversion
⑦rewrite ADCS bit to 1 during A/D conversion operation
do not accept trigger
trigger idle
trigger idle
do not accept trigger
ADCS
⑥modify ADS (from ANI0 to ANI1) during A/D conversion
ADS
ANI0
ANI1
auto restart conversion when conversion completes
auto restart conversion when conversion completes
auto restart conversion when conversion completes
③conversion completes
A/D conversion state
stop converting
idle conversion
power source
(ANI0)
idle conversion
(ANI0)
idle conversion
(ANI1)
idle conversion
(ANI1)
idle conversion
stop converting
ADCR/ADCRH
(ANI0)
(ANI0)
(ANI1)
(ANI1)
INTAD
power source
power source stablization wait cycles

14.4.11 Hardware trigger wait mode (scan mode, continuous conversion mode)

① In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the hardware trigger standby status.

② If a hardware trigger is input while in the hardware trigger standby status, A/D conversion is performed on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 according to the hardware trigger input. A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.

③ A/D conversion of 4 analog input channels is performed. Each time the A/D conversion ends, the conversion result is saved to the A/D conversion result register ADCR, ADCRH, and the A/D conversion end interrupt request signal is INTAD. The next A/D conversion is automatically started from the set channel immediately after the A/D conversion of the 4 channels is completed.

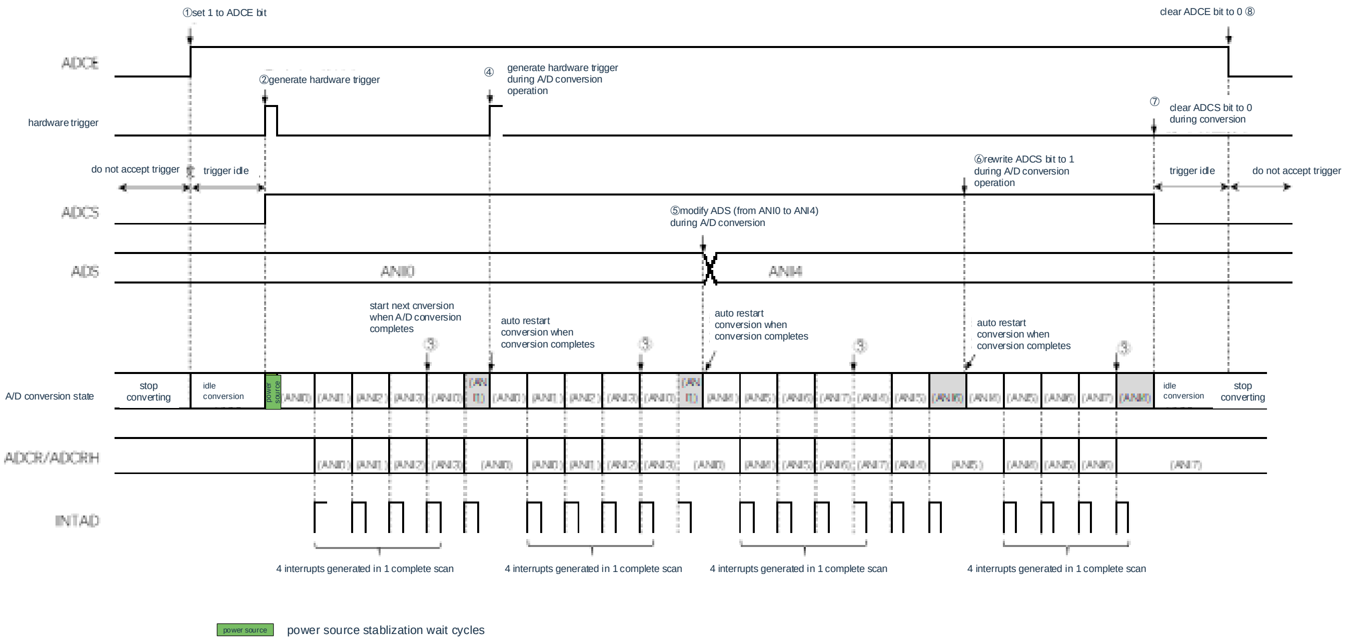
④ If a hardware trigger is entered during the conversion, the current A/D conversion is aborted immediately and the conversion restarts from the original channel.

⑤ If the ADS register is overridden or rewritten during the conversion, the current A/D conversion is immediately aborted and then scanned from the channel redesignated by the ADS register.

⑥ If the ADCS bit is overridden "1" during the conversion, the current A/D conversion immediately aborts and restarts from the original channel.

⑦ When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, the system enters the hardware trigger standby status, and the A/D converter enters the stop status. When ADCE = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

Figure 14-32 Example of hardware trigger wait mode (scan mode, continuous conversion mode) operation timing


①set 1 to ADCE bit
clear ADCE bit to 0 ⑧
ADCE
②generate hardware trigger
④ generate hardware trigger during A/D conversion operation
⑦ clear ADCS bit to 0 during conversion
hardware trigger
do not accept trigger
trigger idle
⑥rewrite ADCS bit to 1 during A/D conversion operation
trigger idle
do not accept trigger
ADCS
⑤modify ADS (from ANI0 to ANI4) during A/D conversion
ADS
ANI0
ANI4
start next cnversion when A/D conversion completes
auto restart conversion when conversion completes
auto restart conversion when conversion completes
auto restart conversion when conversion completes
③
A/D conversion state
stop converting
idle conversion
power source
(ANI0)
(ANI1)
(ANI2)
(ANI3)
(ANI4)
(ANI5)
(ANI6)
(ANI7)
idle conversion
stop converting
ADCR/ADCRH
INTAD
4 interrupts generated in 1 complete scan
4 interrupts generated in 1 complete scan
4 interrupts generated in 1 complete scan
4 interrupts generated in 1 complete scan
power source
power source stablization wait cycles

14.4.12 Hardware trigger wait mode (scan mode, single conversion mode)

① In the stop state, the ADCE bit of mode register 0 (ADM0) of the A/D converter enters the hardware triggered standby state.

② If a hardware trigger is input in a hardware trigger standby state, an A/D conversion is performed on 4 analog input channels. The ADCS bit of the ADM0 register is "1" automatically after the input hardware trigger. The A/D conversion is performed from the analog input channel specified by the scan 0.

③ A/D conversion of 4 analog input channels is performed. Each time the A/D conversion ends, the conversion result is saved to the A/D conversion result register ADCR, ADCRH, and the A/D conversion end interrupt request signal is INTAD.

④ After the A/D conversion is completed, the ADCS bit automatically clears "0", and the A/D converter goes to stop.

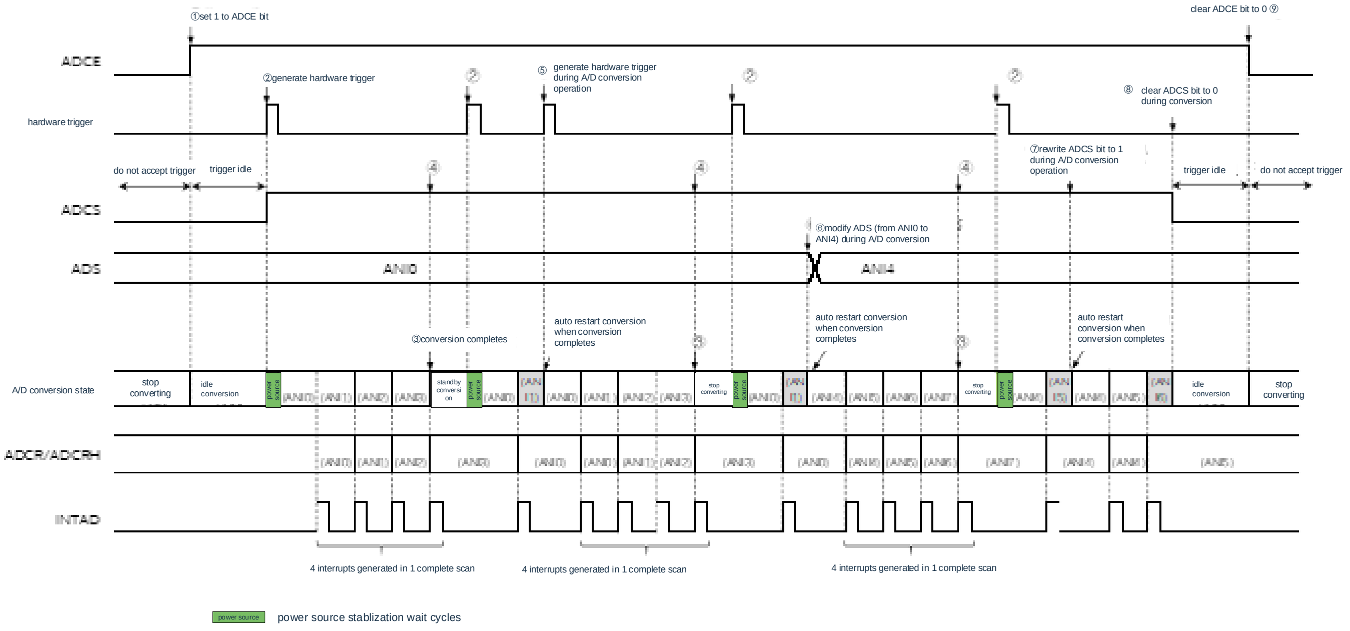
⑤ If a hardware trigger is entered during the conversion, the current A/D conversion is immediately aborted and the conversion is rescanned from the original channel.

⑥ If the ADS register is overridden or rewritten during the conversion, the current A/D conversion is immediately aborted and then scanned from the channel redesignated by the ADS register.

⑦ If the ADCS bit is rewritten "1" during the conversion, the current A/D conversion is aborted immediately and the conversion is scanned from the original channel.

⑧ When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, the system enters the hardware trigger standby status, and the A/D converter enters the stop status. When ADCE = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

Figure 14-33 Example of hardware trigger wait mode (scan mode, single conversion mode) operation timing

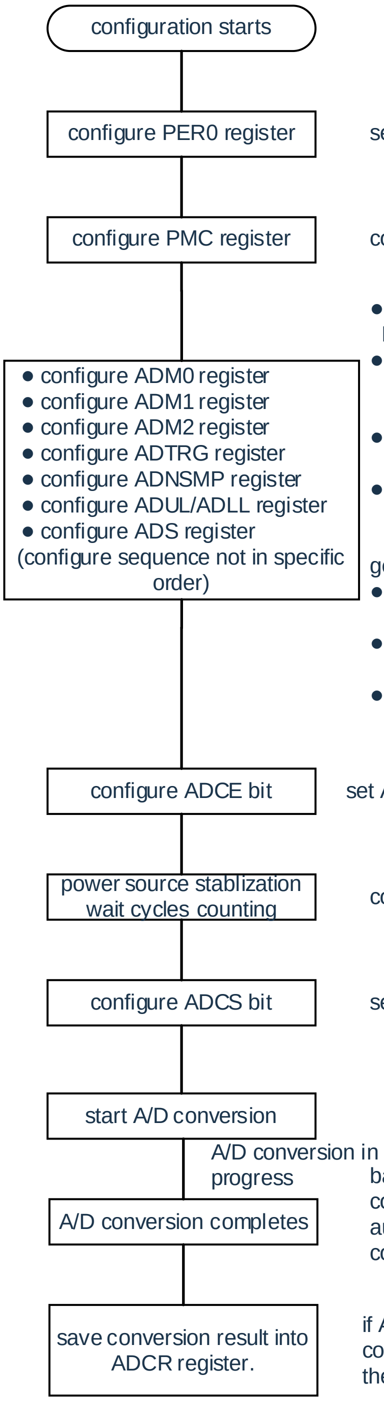

①set 1 to ADCE bit
clear ADCE bit to 0 ⑨
ADCE
②generate hardware trigger
⑤ generate hardware trigger during A/D conversion operation
②
⑧ clear ADCS bit to 0 during conversion
hardware trigger
do not accept trigger
trigger idle
④
⑦rewrite ADCS bit to 1 during A/D conversion operation
trigger idle
do not accept trigger
ADCS
⑥modify ADS (from ANI0 to ANI4) during A/D conversion
ADS
ANI0
ANI4
③conversion completes
auto restart conversion when conversion completes
auto restart conversion when conversion completes
auto restart conversion when conversion completes
③
A/D conversion state
stop converting
idle conversion
power source
(ANI0)
(ANI1)
(ANI2)
(ANI3)
standby conversion
stop converting
(ANI4)
(ANI5)
(ANI6)
(ANI7)
idle conversion
stop converting
ADCR/ADCRH
INTAD
4 interrupts generated in 1 complete scan
4 interrupts generated in 1 complete scan
4 interrupts generated in 1 complete scan
power source
power source stablization wait cycles

14.5 A/D converter set-up flowchart

The set-up flowchart of the A/D converters for each mode of operation is shown below.

14.5.1 Software trigger mode settings

Figure 14-34 Software trigger mode settings


configuration starts
configure PER0 register
configure PMC register
● configure ADM0 register
● configure ADM1 register
● configure ADM2 register
● configure ADTRG register
● configure ADNSMP register
● configure ADUL/ADLL register
● configure ADS register
(configure sequence not in specific order)
configure ADCE bit
power source stablization wait cycles counting
configure ADCS bit
start A/D conversion
A/D conversion in progress
A/D conversion completes
save conversion result into ADCR register.

set ADCEN bit of PER0 register to 1, start provide clock

configure port as analog input: configure via PMCx register

- ADM0 register
  FR2~FR0 bit: configure A/D conversion time.
- ADM1 register
  ADSCM bit: continous conversion / single conversion
  ADMD bit: selection mode/Scan mode
- ADTRG register
  ADTMD1 bit, ADTMD0 bit: configure as software trigger mode
- ADM2 register
  ADREFP bit, ADREFM bit: Select reference voltage
  ADRCK bit: select compare range of A/D conversion result which generates interrupt signal
- ADNSMP register
  configure sampling time
- ADUL/ADLL register
  configure A/D conversion result comparision upper and lower limit
- ADS register
  select analog input channel

set ADCE bit of ADM0 register to 1, enter A/D conversion standby state.

counting power source stablization wait cycles (1us) via software

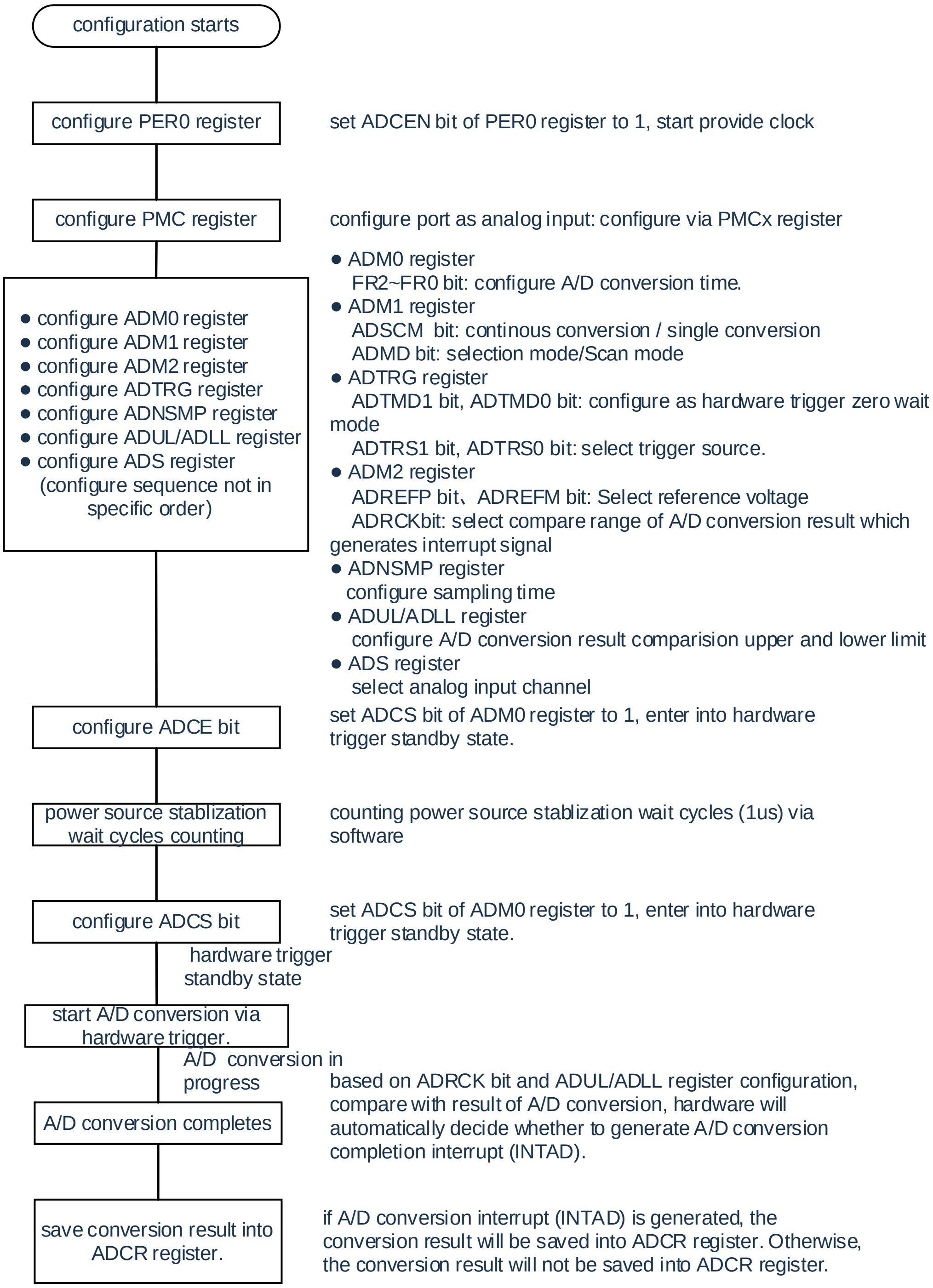
set ADCS bit of ADM0 register to 1m start A/D conversion.

based on ADRCK bit and ADUL/ADLL register configuration, compare with result of A/D conversion, hardware will automatically decide whether to generate A/D conversion completion interrupt (INTAD).

if A/D conversion interrupt (INTAD) is generated, the conversion result will be saved into ADCR register. Otherwise, the conversion result will not be saved into ADCR register.

14.5.2 Hardware trigger no-wait mode settings

Figure 14-35 Hardware trigger no-wait mode settings


configuration starts
configure PER0 register
configure PMC register
● configure ADM0 register
● configure ADM1 register
● configure ADM2 register
● configure ADTRG register
● configure ADNSMP register
● configure ADUL/ADLL register
● configure ADS register
(configure sequence not in specific order)
configure ADCE bit
power source stablization wait cycles counting
configure ADCS bit
hardware trigger standby state
start A/D conversion via hardware trigger.
A/D conversion in progress
A/D conversion completes
save conversion result into ADCR register.

set ADCEN bit of PER0 register to 1, start provide clock

configure port as analog input: configure via PMCx register

- ● ADM0 register
  FR2~FR0 bit: configure A/D conversion time.
- ● ADM1 register
  ADSCM bit: continous conversion / single conversion
  ADMD bit: selection mode/Scan mode
- ● ADTRG register
  ADTMD1 bit, ADTMD0 bit: configure as hardware trigger zero wait mode
  ADTRS1 bit, ADTRS0 bit: select trigger source.
- ● ADM2 register
  ADREFP bit、ADREFM bit: Select reference voltage
  ADRCKbit: select compare range of A/D conversion result which generates interrupt signal
- ● ADNSMP register
  configure sampling time
- ● ADUL/ADLL register
  configure A/D conversion result comparision upper and lower limit
- ● ADS register
  select analog input channel

set ADCS bit of ADM0 register to 1, enter into hardware trigger standby state.

counting power source stablization wait cycles (1us) via software

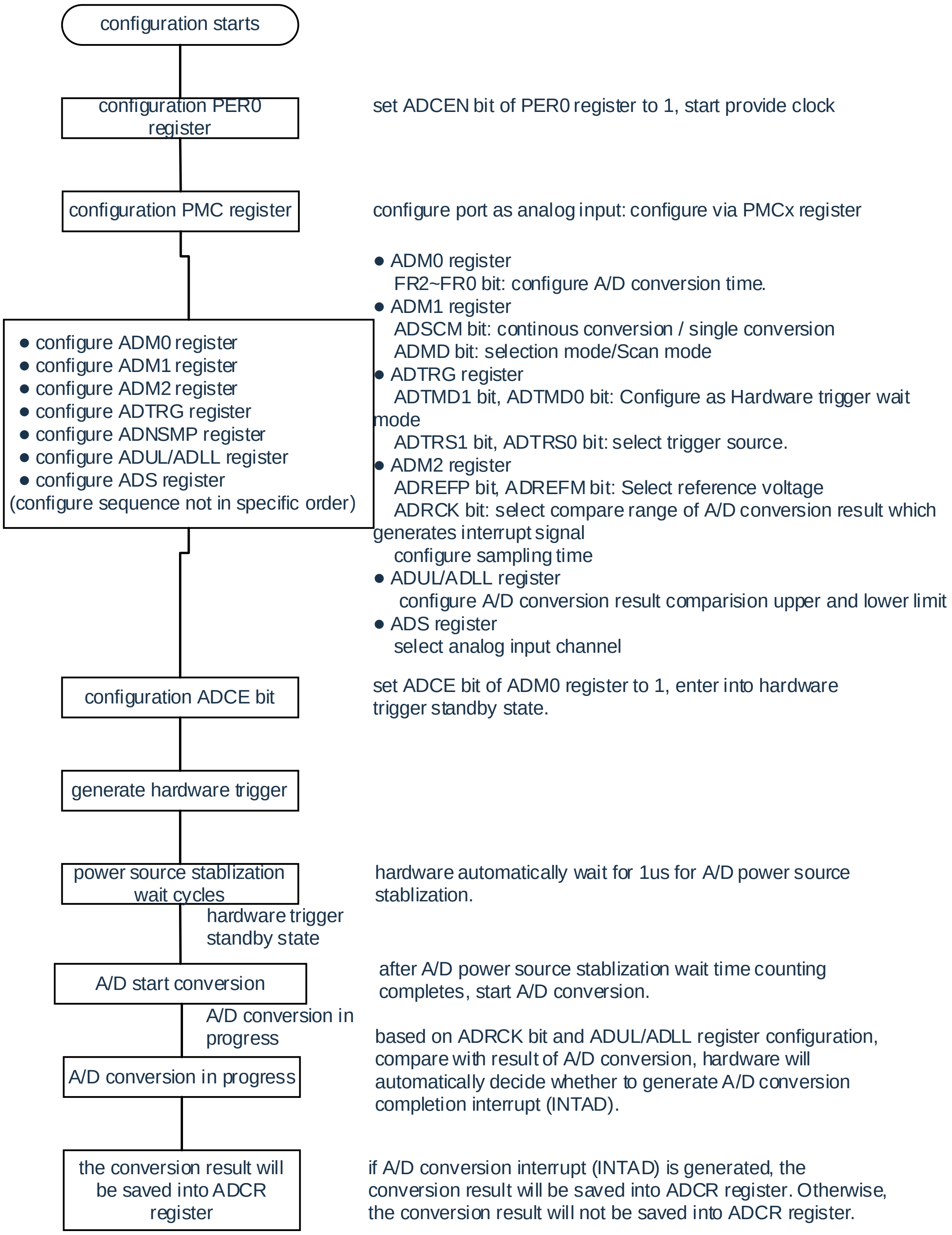
set ADCS bit of ADM0 register to 1, enter into hardware trigger standby state.

based on ADRCK bit and ADUL/ADLL register configuration, compare with result of A/D conversion, hardware will automatically decide whether to generate A/D conversion completion interrupt (INTAD).

if A/D conversion interrupt (INTAD) is generated, the conversion result will be saved into ADCR register. Otherwise, the conversion result will not be saved into ADCR register.

14.5.3 Hardware trigger wait mode settings

Figure 14-36 Hardware trigger wait mode settings


configuration starts
configuration PER0 register
configuration PMC register
● configure ADM0 register
● configure ADM1 register
● configure ADM2 register
● configure ADTRG register
● configure ADNSMP register
● configure ADUL/ADLL register
● configure ADS register
(configure sequence not in specific order)
configuration ADCE bit
generate hardware trigger
power source stablization wait cycles
hardware trigger standby state
A/D start conversion
A/D conversion in progress
A/D conversion in progress
the conversion result will be saved into ADCR register

set ADCEN bit of PER0 register to 1, start provide clock

configure port as analog input: configure via PMCx register

- ADM0 register
  FR2~FR0 bit: configure A/D conversion time.
- ADM1 register
  ADSCM bit: continous conversion / single conversion
  ADMD bit: selection mode/Scan mode
- ADTRG register
  ADTMD1 bit, ADTMD0 bit: Configure as Hardware trigger wait mode
  ADTRS1 bit, ADTRS0 bit: select trigger source.
- ADM2 register
  ADREFP bit, ADREFM bit: Select reference voltage
  ADRCK bit: select compare range of A/D conversion result which generates interrupt signal
  configure sampling time
- ADUL/ADLL register
  configure A/D conversion result comparision upper and lower limit
- ADS register
  select analog input channel

set ADCE bit of ADM0 register to 1, enter into hardware trigger standby state.

hardware automatically wait for 1us for A/D power source stablization.

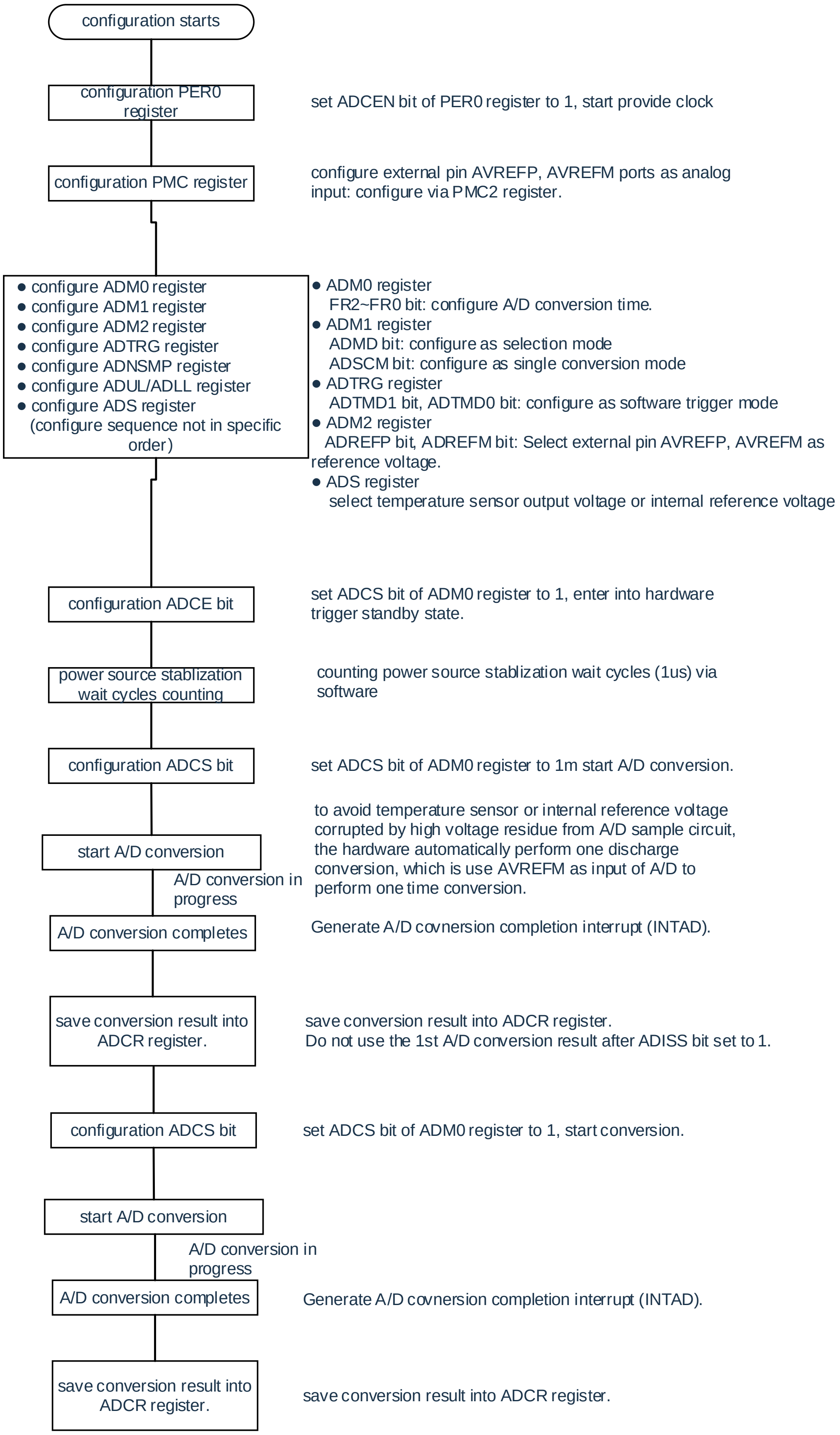
after A/D power source stablization wait time counting completes, start A/D conversion.

based on ADRCK bit and ADUL/ADLL register configuration, compare with result of A/D conversion, hardware will automatically decide whether to generate A/D conversion completion interrupt (INTAD).

if A/D conversion interrupt (INTAD) is generated, the conversion result will be saved into ADCR register. Otherwise, the conversion result will not be saved into ADCR register.

14.5.4 Settings when selecting the output voltage/internal reference voltage of temperaturesensor (Take software trigger mode, single conversion mode for example)

Figure 14-37 Settings when selecting the output voltage/internal reference voltage of the temperature sensor


configuration starts
configuration PER0 register
configuration PMC register
● configure ADM0 register
● configure ADM1 register
● configure ADM2 register
● configure ADTRG register
● configure ADNSMP register
● configure ADUL/ADLL register
● configure ADS register
(configure sequence not in specific order)
configuration ADCE bit
power source stablization wait cycles counting
configuration ADCS bit
start A/D conversion
A/D conversion in progress
A/D conversion completes
save conversion result into ADCR register.
configuration ADCS bit
start A/D conversion
A/D conversion in progress
A/D conversion completes
save conversion result into ADCR register.

set ADCEN bit of PER0 register to 1, start provide clock

configure external pin AVREFP, AVREFM ports as analog input: configure via PMC2 register.

- ADM0 register
  FR2~FR0 bit: configure A/D conversion time.
- ADM1 register
  ADMD bit: configure as selection mode
  ADSCM bit: configure as single conversion mode
- ADTRG register
  ADTMD1 bit, ADTMD0 bit: configure as software trigger mode
- ADM2 register
  ADREFP bit, ADREFM bit: Select external pin AVREFP, AVREFM as reference voltage.
- ADS register
  select temperature sensor output voltage or internal reference voltage

set ADCS bit of ADM0 register to 1, enter into hardware trigger standby state.

counting power source stablization wait cycles (1us) via software

set ADCS bit of ADM0 register to 1m start A/D conversion.

to avoid temperature sensor or internal reference voltage corrupted by high voltage residue from A/D sample circuit, the hardware automatically perform one discharge conversion, which is use AVREFM as input of A/D to perform one time conversion.

Generate A/D cvnersion completion interrupt (INTAD).

save conversion result into ADCR register.
Do not use the 1st A/D conversion result after ADISS bit set to 1.

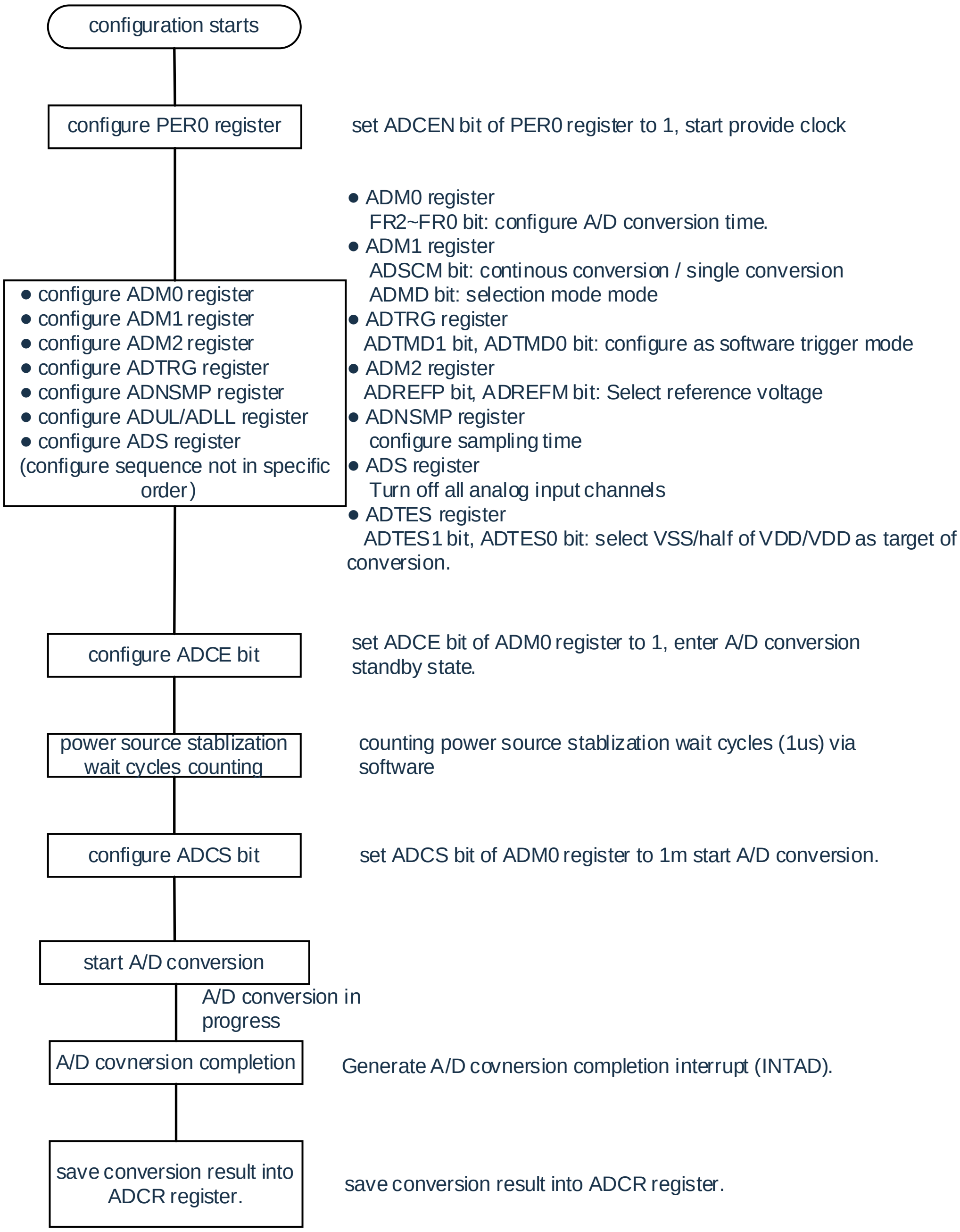
set ADCS bit of ADM0 register to 1, start conversion.

Generate A/D cvnersion completion interrupt (INTAD).

save conversion result into ADCR register.

14.5.5 Test mode settings

Figure 14-38 Test mode settings (VSS/half_VDD/VDD as the conversion object)


configuration starts
configure PER0 register
● configure ADM0 register
● configure ADM1 register
● configure ADM2 register
● configure ADTRG register
● configure ADNSMP register
● configure ADUL/ADLL register
● configure ADS register
(configure sequence not in specific order)
configure ADCE bit
power source stablization wait cycles counting
configure ADCS bit
start A/D conversion
A/D conversion in progress
A/D covnersion completion
save conversion result into ADCR register.

set ADCEN bit of PER0 register to 1, start provide clock

● ADM0 register
FR2~FR0 bit: configure A/D conversion time.
● ADM1 register
ADSCM bit: continous conversion / single conversion
ADMD bit: selection mode mode
● ADTRG register
ADTMD1 bit, ADTMD0 bit: configure as software trigger mode
● ADM2 register
ADREFP bit, ADREFM bit: Select reference voltage
● ADNSMP register
configure sampling time
● ADS register
Turn off all analog input channels
● ADTES register
ADTES1 bit, ADTES0 bit: select VSS/half of VDD/VDD as target of conversion.

set ADCE bit of ADM0 register to 1, enter A/D conversion standby state.

counting power source stablization wait cycles (1us) via software

set ADCS bit of ADM0 register to 1m start A/D conversion.

Generate A/D covnersion completion interrupt (INTAD).

save conversion result into ADCR register.

Chapter 15 Universal Serial Communication Unit

This product is equipped with 3 general-purpose serial communication units, each unit has 2 serial channels, each channel can realize 3-wire serial (SSPI), UART and simplified I^{2}C communication.

Function assignment of each channel supported by this product is as shown below.

○ 64-pin products

Unit	Channel	Used as SSPI	Used as UART	Used as simplified I^{2}C
0	0	-	UART0 (LIN-bus support)	-
	1	SSPI01		IIC01
1	0	-	UART1	-
	1	SSPI11		IIC11
2	0	-	UART2	-
	1	SSPI21		IIC21

Note: "-" indicates that it is not supported in this series of products

○ 48-pin products

Unit	Channel	Used as SSPI	Used as UART	Used as simplified I^{2}C
0	0	-	UART0 (LIN-bus support)	-
	1	-		-
1	0	-	UART1	-
	1	SSPI11		IIC11
2	0	-	UART2	-
	1	SSPI21		IIC21

Note: "-" indicates that it is not supported in this series of products

When UART0 is used for channels 0 and 1 of the unit 0, SSPI00, SSPI01, and IIC01 cannot be used.

When UART1 is used for channels 0 and 1 of the unit 1, SSPI10, SSPI11, and IIC11 cannot be used.

When UART2 is used for channels 0 and 1 of the unit 2, SSPI20, SSPI21, and IIC21 cannot be used.

15.1 Function of universal serial communication unit

The features of each serial interface supported by this product are shown below.

15.1.1 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20,SSPI21)

Data is transmitted and received synchronously with a serial clock (SCLK) output from the master control device.

This is a clock synchronization function communicating using 1 SCLK, 1 SDO and 1 SDI with 3 communication lines.

For specific set-up examples, refer to "15.5 3-wire serial I/O(SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) communication".

[Transmitting and Receiving Data]

- 7 bit ~16 bit data length
- Phase control for transmitting and receiving data
- MSB/LSB First Choice
- Level setting for transmit/receive data

[Clock Control]

- Master or subordinate selection
- Phase control of input/output clock
- A transmission period generated by a pre-divider and an intra-channel counter is set.
- maximum transfer rate Note
  Master Communications: Max.f_{CLK}/2
  Slave Communication: Max.f_{MCK}/6

[Interrupt Function]

- Interrupt transmission end, buffer null interrupt

[Error Detection Flag]

- overflow error

Note Use the clocks within a range satisfying the SCLK cycle time (T_{KCY}) characteristics. For details, please refer to the data sheet.

15.1.2 UART (UART0~UART2)

This is the ability to communicate asynchronously over a total of two lines, Serial Data Transmission (TxD) and Serial Data Reception (RxD). The two communication lines are used to transmit and receive data asynchronously (using internal baud rate) with other communication parties in data frames (consisting of start bit, data, parity bit and stop bit). Full duplex UART communication can be achieved by using two channels dedicated to transmit (even channel) and receive (odd channel), and LIN-bus can be supported by combining universal timer units and INTP0.

For specific set-up examples, refer to "15.7 Operation of UART(UART0~UART2) communication."

[Transmitting and Receiving Data]

- Data length notes for 7, 8, 9, or 16 bits$^{\text{Note}}$
- MSB/LSB First Choice
- Level setting for transmitting and receiving data, selection of inversion
- Additional, parity functions for parity bits
- Additional stop bit

[Interrupt Function]

- Interrupt transmission end, buffer null interrupt
- Error interrupt due to frame error, parity error, or overflow error

[Error Detection Flag]

- Frame error, parity error, overflow error

15.1.3 Simplified I2C (IIC00, IIC01, IIC10, IIC11, IIC20,IIC21)

This is the capability of clock synchronization with multiple devices through two lines of serial clock (SCL) and serial data (SDA). Since this simplified I^{2}C is designed for single communication with EEPROM, flash memory, A/D converter, etc., it is only used as master device.

For the start condition and stop condition, the AC specification must be observed, and the control register must be handled by software. For specific set-up examples, refer to "15.9 Operation of simplified I2C (IIC00, IIC01, IIC10, IIC11, IIC20,IIC21) communication".

[Transmitting and Receiving Data]

- Master Send, Master Receive (only for single master master master master functions)
- ACK output function Note, ACK detection function
- 8-bit data length (when sending an address, specifying the address with 7 bits high and R/W control with lowest bits)
- Manual generation of start and stop conditions

[Interrupt Function]

- End of transfer interrupt

[Error Detection Flag]

- ACK error, overflow error

※[Features not supported by Simplified I^{2}C]

- Slave send, slave receive
- Quorum failure detection
- Waiting for detection

Note When receiving the last data, the ACK is not output if writing "0" to the SOEmn bit (Serial Output Allowed Register m(SOEm). Refer to " 15.9.3 (2) Process Flow " for details.

Remark Refer to “Chapter 19 Serial Interface IICA” when using the full-function I^{2}C bus.

15.2 Structure of universal serial communication unit

The universal serial communication unit consists of the following hardware.

Table 15-1 Structure of universal serial communication unit

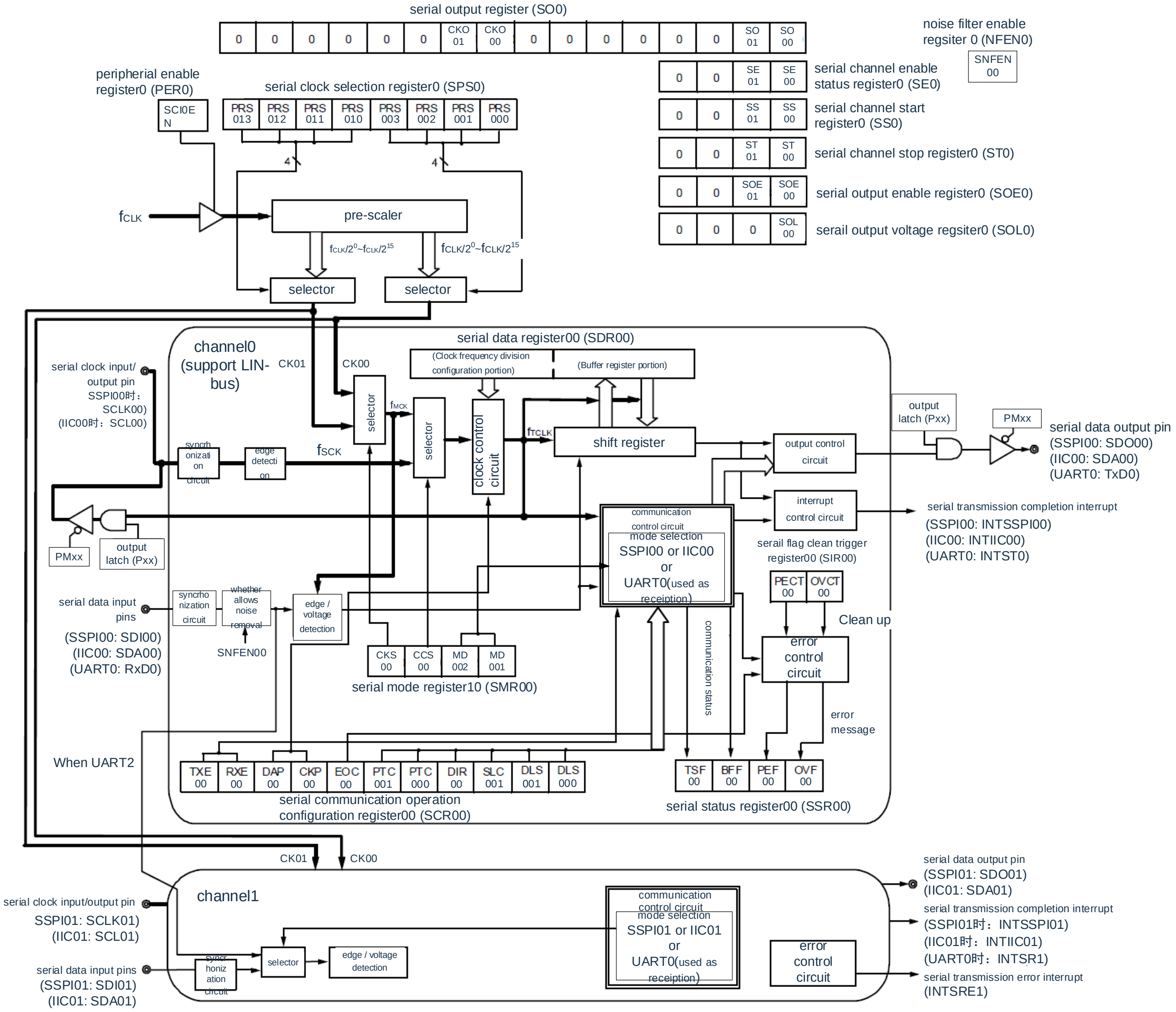
Item	Structure
Shift register	16-bit
Buffer register	Serial data register mn (SDRmn)$^{\text{Note}}$
Serial clock input/output	SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, SCLK21 pin (for 3-wire serial I/O), SCL00, SCL01, SCL10, SCL11, SCL20, SCL21 pin (for simplified I^{2}C)
Serial data input	SDI00, SDI01, SDI10, SDI11, SDI20, SDI21 pin (for 3-wire serial I/O), RxD0, RxD1, RxD2 pin (for UART)
Serial data output	SDO00, SDO01, SDO10, SDO11, SDO20, SDO21 pin (for 3-wire serial I/O), TxD0, TxD1, TxD2 pin (for UART)
Serial Data Input/Output	SDA00, SDA01, SDA10, SDA11, SDA20, SDA21 pin (for simplified I^{2}C)
Slave select input	SS00 Pin (for the slave selectioninput function)
Control register	<Register for unit set-up> •Peripheral enable register 0 (PER0) •Serial clock select register m (SPSm). •Serial channel enable status register m (SEm) •Serial channel start register m (SSm). •Serial channel stop register m (STm). •Serial output enable register m (SOEm). •Serial output register m (SOm). •Serial output level register m (SOLm). •Input switch control register (ISC) •Noise filter enable register 0 (NFEN0).
	<Registers of each channel> •Serial data register mn (SDRmn) •Serial mode register mn (SMRmn) •Serial communication run set-up register mn (SCRmn). •Serial state register mn (SSRmn). •Serial flag clear trigger register mn (SIRmn).
	•Port multiplexing function configuration register (PxxCFG) •Port output mode register (POMxx) •Port mode register (PMxx) •Port register (Pxx)

Note: SEmn=1 during operation.

Remark m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI number (p=00,01,10,11,20,21)
q:UART (q=0~2) r:IIC (r=00,01,10,11,20,21)

Block diagram of the universal serial communication unit 0 is shown in Figure 15-1. (Take unit 0 as an example)

Figure 15-1 Block diagram of the universal serial communication unit 0


serial output register (SO0)
0 0 0 0 0 0 CKO 01 CKO 00 0 0 0 0 0 0 SO 01 SO 00
noise filter enable regsiter 0 (NFEN0)
SNFEN 00
0 0 SE 01 SE 00
serial channel enable status register0 (SE0)
0 0 SS 01 SS 00
serial channel start register0 (SS0)
0 0 ST 01 ST 00
serial channel stop register0 (ST0)
0 0 SOE 01 SOE 00
serial output enable register0 (SOE0)
0 0 0 SOL 00
serail output voltage regsiter0 (SOL0)
peripherial enable register0 (PER0)
SCI0EN
serial clock selection register0 (SPS0)
PRS 013 PRS 012 PRS 011 PRS 010 PRS 003 PRS 002 PRS 001 PRS 000
4
4
fCLK
pre-scaler
fCLK/2⁰~fCLK/2¹⁵
fCLK/2⁰~fCLK/2¹⁵
selector
selector
channel0
(support LIN-bus)
CK01
CK00
serial data register00 (SDR00)
(Clock frequency division configuration portion)
(Buffer register portion)
serial clock input/output pin
SSPI00时：SCLK00)
(IIC00时：SCL00)
selector
fMCK
selector
clock control circuit
fTCLK
shift register
synchronization circuit
edge detection
fSCK
output control circuit
output latch (Pxx)
PMxx
serial data output pin
(SSPI00: SDO00)
(IIC00: SDA00)
(UART0: TxD0)
interrupt control circuit
serial transmission completion interrupt
(SSPI00: INTSSPI00)
(IIC00: INTIIC00)
(UART0: INTST0)
communication control circuit
mode selection
SSPI00 or IIC00 or UART0(used as receiption)
serail flag clean trigger register00 (SIR00)
PECT 00 OVCT 00
Clean up
PMxx
output latch (Pxx)
serial data input pins
(SSPI00: SDI00)
(IIC00: SDA00)
(UART0: RxD0)
synchronization circuit
whether allows noise removal
SNFEN00
edge / voltage detection
CKS 00 CCS 00 MD 002 MD 001
serial mode register10 (SMR00)
communication status
error control circuit
error message
When UART2
TXE 00 RXE 00 DAP 00 CKP 00 EOC 00 PTC 001 PTC 000 DIR 00 SLC 001 DLS 001 DLS 000
serial communication operation configuration register00 (SCR00)
TSF 00 BFF 00 PEF 00 OVF 00
serial status register00 (SSR00)
CK01
CK00
channel1
serial clock input/output pin
SSPI01: SCLK01)
(IIC01: SCL01)
serial data input pins
(SSPI01: SDI01)
(IIC01: SDA01)
synchonization circuit
selector
edge / voltage detection
communication control circuit
mode selection
SSPI01 or IIC01 or UART0(used as receiption)
error control circuit
serial data output pin
(SSPI01: SDO01)
(IIC01: SDA01)
serial transmission completion interrupt
(SSPI01时：INTSSPI01)
(IIC01时：INTIIC01)
(UART0时：INTSR1)
serial transmission error interrupt
(INTSRE1)

Note: Units 0, 1 and 2 have the same structure

15.2.1 Shift register

This is a 16-bit register that performs parallel and serial interconversion.

During reception, it converts data input to the serial pin into parallel data. When data is transmitted, the value set to this register is output as serial data from the serial output pin. The shift register cannot be directly manipulated by program.

To read or write the shift register, use the serial data register mn (SDRmn) when operation is in progress (SEmn = 1).

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

15.2.2 Serial data register mn (SDRmn)

The SDRmn register is a send and receive data register (16-bit) for channel n.

If operation is stopped (SEmn = 0), bits 15 to 9 are used as a register that sets the division ratio of the operation clock(F_{MCK}). If operation is in progress (SEmn = 1), bit15~9 selected as a transmit/receive buffer register.

When receiving data, the parallel data converted by the shift register is saved to the serial data register SDRmn; when sending data, the send data that is transferred to the shift register is set to the serial data register SDRmn.

Regardless of the output order of the data, the data saved to the SDRmn register according to the setting of bit3 to bit0 (DLSmn3 to DLSmn0) of the serial communication run setting register mn (SCRmn) is shown below:

- 7-bit data length (bit0~6 stored in the SDRmn register)
- 8-bit data length (bit0~7 stored in SDRmn register).

:

- 16-bit data length (bit0~15 stored in the SDRmn register) [Note 1]

The SDRmn register can be read and written in units of 16 bits.

When SEmn=1, the lower 8 bits of the SDRmn register can be read and written in 8-bit as SDRmnL[Note].

According to the communication mode, it can read and write SDRmnL registers with the following SFR names.

- SSPIp Communications......SDIOpL
- UARTq Receive......RXDq (UARTq Receive Data Register)
- UARTq Send......TXDq (UARTq send data register)
- IICr Communications......SDIOr (IICr data register)

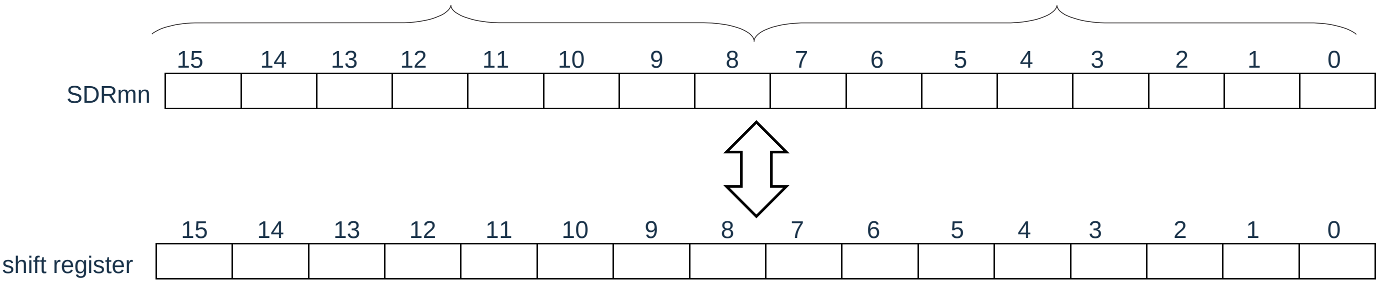
After the reset signal is generated, the value of the SDRmn register changes to "0000H".

Note: When running stop (SEmn=0), disable the SDRmn[7:0] rewrite via 8-bit memory operation instructions (otherwise SDRmn[15:9] is all cleared).

Note: m: Unit number (m=0,1,2) n: Channel number (n=0, 1) p: SSPI number (p=00,01,10,11,20,21)
q: UART (q=0~2) r: IIC (r=00,01,10,11,20,21)

Figure 15-2 Format of serial data register mn(SDRmn) (mn=00,01,10,11)

Reset: 0000H R/W



Note: For the function of the higher 7 bits of the SDRmn register, please refer to "15.3 Registers for controlling universal serial communication unit".

m: Unit number (m=0, 1, 2); n: Channel number (n=0, 1)

15.3 Registers for controlling universal serial communication unit

The registers that control the universal serial communication unit are as follows:

- Peripheral enable register 0 (PER0).
- Serial clock select register m (SPSm)
- Serial mode register mn (SMRmn)
- Serial communication run set-up register mn (SCRmn)
- Serial data register mn (SDRmn)
- Serial flag clear trigger register mn (SDIRmn)
- Serial state register mn (SSRmn)
- Serial channel start register m (SSm).
- Serial channel stop register m (STm).
- Serial channel enable state register m (SEm).
- Serial output enable register m (SOEm)
- Serial output level register m (SOLm)
- Serial output register m (SOm).
- Input switch control register (ISC)
- Noise filter enable register 0 (NFEN0).
- Port multiplexing function configuration register (PxxCFG)
- Port output mode register (POMx)
- Port mode register (PMx)
- Port register (Px)

Remark m: Unit number (m=0, 1, 2); n: Channel number (n=0, 1)

Serial Communication unit register list

Unit 0 register base address: 0x40041000

Unit 1 register base address: 0x40041200

Unit 2 register base address: 0x40041400

Offset address	Register name	R/W	Reset value
0x000	SSRm0	R	0000H
0x000	SSRm0L	R	00H
0x002	SSRm1	R	0000H
0x002	SSRm1L	R	00H
0x004	SIRm0	R	0000H
0x004	SIRm0L	R	00H
0x006	SIRm1	R	0000H
0x006	SIRm1L	R	00H
0x008	SMRm0	R/W	0000H
0x00A	SMRm1	R/W	0000H
0x00C	SCRm0	R/W	00H
0x00E	SCRm1	R/W	0000H
0x012	SSm	R/W	0020H
0x012	SSmL	R/W	0020H
0x014	STm	R/W	0020H
0x014	STmL	R/W	0020H
0x016	SPSm	R/W	0020H
0x016	SPSmL	R/W	0020H
0x018	SOm	R/W	0087H
0x01A	SOEm	R/W	0087H
0x01A	SOEmL	R/W	0087H
0x020	SOLm	R	0000H
0x020	SOLmL	R	00H
0x022	SSEm	R/W	0000H
0x022	SSEmL	R/W	0000H
0x110	SDRm0	R/W	0000H
0x110	SIOm0	R/W	00H
0x110	TXDm	R/W	00H
0x112	SDRm1	R/W	0000H
0x112	RXDm	R/W	00H
0x112	SIOm1	R/W	00H

Note: Unit number m=0, 1, 2

15.3.1 Peripheral enable register 0 (PER0)

The PER0 register is a register that sets a clock that is allowed or prohibited to supply to each peripheral hardware. Reduce power consumption and noise by stopping clock supply to unused hardware.

To use universal serial communication unit 0, bit2 (SCI0EN) must be set to "1".

To use universal serial communication unit 1, bit3 (SCI1EN) must be set to "1".

To use universal serial communication unit 2, bit4 (SCI2EN) must be set to "1".

The PER0 register is set by an 8-bit memory operation instruction.

After the reset signal is generated, the value of the PER0 register changes to '00H'.

Figure 15-3 Format of peripheral enable register 0 (PER0)

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

symbol	7	6	5	4	3	2	1	0
PER0	xx	xx	xx	SCI2EN	SCI1EN	SCI0EN	xx	xx

SCImEN	Control of an input clock of a universal serial communication unit m is provided
0	Stop provide an input clock. •Cannot write the SFR used by the universal serial communication unit m. •The universal serial communication unit m is in a reset state.
1	Allows providing an input clock. •SFR capable of reading and writing the UIC unit m.

Note 1. To set up the universal serial communication unit m, the following register must 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 all initial values (except input switch control register (ISC), noise filter permit register 0 (NFEN0), Port multiplexing function configuration register (PxxCFG), port output mode register (POMx), port mode register (PMx), port mode control register (PMCx) and port register (Px)).

- Serial clock select register m (SPSm).
- Serial mode register mn (SMRmn)
- Serial communication run setting register mn (SCRmn).
- Serial data register mn (SDRmn)
- Serial flag clear trigger register mn (SIRmn).
- Serial state 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).

15.3.2 Serial clock select register m (SPSm)

The SPSm register is a 16-bit register that selects two common run-time clocks (CKm0, CKm1). CKm1 is selected by bit7~4 of the SPSm register, and CKm0 is selected by bit3~0.

Prevents the SPSm register from being overridden during (SEmn=1).

The SPSm register is set by a 16-bit memory operation instruction.

A lower 8-bit of the SPSm register can be set with the SPSmL and through an 8-bit memory operation instruction.

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

Figure 15-4 Format of serial clock select 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	Selection for operation clock (CKmk) Note
0	0	0	0	f_{CLK}
0	0	0	1	$f_{CLK}/2$
0	0	1	0	$f_{CLK}/2^2$
0	0	1	1	$f_{CLK}/2^3$
0	1	0	0	$f_{CLK}/2^4$
0	1	0	1	$f_{CLK}/2^5$
0	1	1	0	$f_{CLK}/2^6$
0	1	1	1	$f_{CLK}/2^7$
1	0	0	0	$f_{CLK}/2^8$
1	0	0	1	$f_{CLK}/2^9$
1	0	1	0	$f_{CLK}/2^{10}$
1	0	1	1	$f_{CLK}/2^{11}$
1	1	0	0	$f_{CLK}/2^{12}$
1	1	0	1	$f_{CLK}/2^{13}$
1	1	1	0	$f_{CLK}/2^{14}$
1	1	1	1	$f_{CLK}/2^{15}$

Note 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 bit15~8 must be set to 0.

Note 1. f_{CLK}: Clock frequency of the CPU/peripheral hardware
2. m: Unit number (m=0,1,2)
3 .k=0,1

15.3.3 Serial mode register mn (SMRmn)

The SMRmn register is a register for setting the operation mode of channel n. It carries out the selection of the operation clock (f_{MCK}), the designation of whether the serial clock (f_{SCLK}) input can be used, the setting of the start trigger, the setting of the operation mode (SSPI, UART, simplified I^2C), and the selection of the interrupt source. In addition, the inversion level of received data is set in UART mode only.

Prevents the SMRmn register from being overwritten during the run (SEmn=1), but MDmn0 bits can be overwritten during the operation.

The SMRmn register is set by a 16-bit memory operation instruction.

After the reset signal is generated, the value of the SMRmn register changes to "0020H".

Figure 15-5 Format of serial mode register mn (SMRmn) (1/2)

After Reset:0020H R/W

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKS mn	CCS mn	0	0	0	0	0	STSm Note 1	0	SISm n0 Note 1	1	0	0	MD mn2	MD mn1	MD mn0

CKSmn	Selection of channel n operating clock (f_{MCK})
0	The SPSm register sets the operating clock CKm0
1	The SPSm register sets the operating clock CKm1
The runtime clock (f_{MCK}) is used for edge detection circuits. A transmission clock (f_{TCLK}) is generated by setting the CCSmn bit and the SDRmn register's high 7 bits.	

CCSmn	Selection of channel n transfer clock (f_{TCLK})
0	The CKSmn bit specifies the running clock F_{MCK} divider clock
1	Input clock F_{SCLK} from the SCLKp pin (slave transfer in SSPI mode).
The transmit clock F_{TCLK} is used for shift registers, communication control circuits, output controllers, interrupt control circuits, and error control circuits. When the CCSmn bit is at "0", the operating clock (F_{MCK}) is set the dividing ratio of the operating clock (F_{MCK}) by the higher 7 bits of the SDRmn register.	

STSmn Note1	Start triggering source selection
0	Only software triggers are valid (selected in SSPI, UART transmit, simplified I^2C).
1	The effective edge of the RxDq pin (selected when received by the UART).
When the above condition is satisfied after the SSm register is set "1", the transfer is started.	

Note1. Limited to SMR01, SMR11, SMR21 registers only.

Notice The bit13~9,7,4,3 (SMR00, SMR10, SMR20 registers must be bit13~6,4,3) set to "0" and the bit5 set to "1".

Remark m: Unit number (m=0,1,2)n: Channel number (n=0,1) p: SSPI Number (p=00,01,10,11,20,21)
q: UART (q=0~2) r: IIC (r=00,01,10,11,20,21)

Figure 15-5 Format of serial mode register mn (SMRmn) (2/2)

After reset:0020H R/W

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKS mn	CCS mn	0	0	0	0	0	STS mn Note 1	0	SISm n0 Note 1	1	0	0	MD mn2	MD mn1	MD mn0

SISmn0 Note 1	Level inversion control of channel n receiving data in UART mode
0	The descent edge is detected as the starting bit. The input communication data is not inverted.
1	Detect the rising edge as the starting bit. The input communication data is inverted.

MDmn2	MDmn1	Settings for channel n operation mode
0	0	SSPI mode
0	1	UART mode
1	0	Simplified I^{2}C mode
1	1	Settings are disabled.

MDmn0	Channel n interrupt source selection
0	End of Transfer Interrupt
1	Buffer null interrupt (Occurs when data is transferred from a SDRmn register to a shift register)
When sending continuously, if the MDmn0 bit is "1" and SDRmn's data is empty, write the next sending data.	

Note 1. Only SMR01, SMR11, SMR21 register.

Notice Bit13~9,7,4,3 (SMR00, SMR10, SMR20 registers must be bit13~6,4,3) set "0" and the bit5 set "1".

Remark m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI number (p=00,01,10,11,20,21)
q:UART (q=0~2) r:IIC (r=00,01,10,11,20,21)

15.3.4 Serial communication run setting register mn (SCRmn)

The SCRmn register is a communication operation setting register of channel n, setting data sending and receiving mode, data and clock phase, whether shielding error signal, parity check bit, start bit, stop bit and data length.

It is forbidden to overwrite the SCRmn register during operation (SEmn=1).

The SCRmn register is set by a 16-bit memory operation instruction.

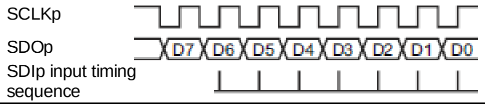
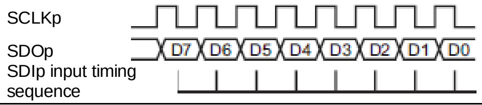
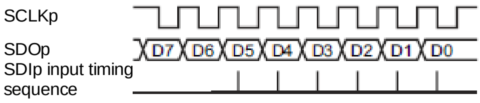
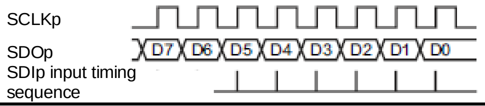
After the reset signal is generated, the value of the SCRmn register changes to "0087H".

Figure 15-6 Format of serial communication run setting register mn (SCRmn) (1/3)

After Reset: 0087H R/W

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn	RXEmn	DAPmn	CKPmn	0	0	PTCmn1	PTCmn0	DIRmn	0	SLCmn1[Note1]	SLCmn0	DLSmn3	DLSmn2	DLSmn1	DLSmn0

TXEmn	RXEmn	Settings for Channel n Running Mode
0	0	No communication.
0	1	Receive only.
1	0	Send only.
1	1	Enables sending and receiving.

DAPmn	CKPmn	data and clock phase selection in SSPI mode	Type
0	0	 SCLKp SDOp D7 D6 D5 D4 D3 D2 D1 D0 SDIp input timing sequence	1
0	1	 SCLKp SDOp D7 D6 D5 D4 D3 D2 D1 D0 SDIp input timing sequence	2
1	0	 SCLKp SDOp D7 D6 D5 D4 D3 D2 D1 D0 SDIp input timing sequence	3
1	1	 SCLKp SDOp D7 D6 D5 D4 D3 D2 D1 D0 SDIp input timing sequence	4
in UART mode and simple I2C mode, must set DAPmn bit and CKPmn bit both to 0.			

EOCmn	Masking control for error interrupt signals (INTSREx (x=0~2))
0	Prevents the generation of an error interrupt INTSREx (INTSRx generation).
1	Allow error-generated interrupt INTSREx (INTSRx is not generated when an error occurs).
The EOCmn bit must be set to "0" in SSPI mode and Simplified I^2C mode, or when transmitting from the UART [Note 2].	

Note 1. Limited to SCR00, SCR02, SCR10 registers only.

2. An error interrupt INTSREn may occur when the EOCmn bit is "0" and SSPImn is not used.

Note bit6, 10, 11 must be set to '0' (bit5 of the SCR01, SCR11, SCR21 register must also be set to '0').

Remark m: Unit number (m=0,1,2) n: Channel mumber (n=0,1) p: SSPI mumber (p=00,01,10,11,20,21)

Figure 15-6 Format of serial communication run setting register mn (SCRmn) (2/3)

After Reset:0087H R/W

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn	RXEmn	DAPmn	CKPmn	0	EOCmn	PTCmn1	PTCmn0	DIRmn	0	SLCmn1[Note1]	SLCmn0	DLSmn3	DLSmn2	DLSmn1	DLSmn0

PTCmn1	PTCmn0	Settings for parity bits in UART mode	
		Send	Receive
0	0	No parity bits are output.	No parity is received.
0	1	Output parity [note 3].	Parity is not judged.
1	0	Output even check.	Parity check.
1	1	Output odd check.	Judge odd check.
In SSPI mode and simplified I^2C mode, both PTCmn1 bits and PTCmn0 bits must be set to "0".			

DIRmn	Selection of data transfer order in SSPI and UART mode
0	MSB-first input/output.
1	LSB-preferred input/output.
In simplified I^2C mode, the DIRmn bit must be set to "0".	

SLCmn1 [Note 1]	SLCmn0	Settings for stop bits in UART mode
0	0	No stop bit
0	1	stop bit length=1 bit
1	0	Stop bit length=2 bits (mn=00, 10,20 only).
1	1	Disable from setting.
If an end-of-transfer interrupt is selecte, an interrupt is generated aft all stop bits have been transfer. Must be set to 1 stop bit (SLCmn1, SLCmn0=0, 11) at UART receive or in easy I2C mode. In SSPI mode, must be set to no stop bit (SLCmn1, SLCmn0=0,0). When UART is sent, it must be set to 1 bit (SLCmn1, SLCmn0=0, 1) or 2 bit (SLCmn1, SLCmn0=1, 0).		

Note 1. Limited to SCR00, SCR10, SCR20 registers only.
2. Always attach a "0" regardless of the content of the data.

Notice bit6, 10, 11 must be set to '0'.

Remark m: Unit number (m=0,1,2) n: Channel Number (n=0,1) p: SSPI Number (p=00,01,10,11,20,21)

Figure 15-6 Format of serial communication run setting register mn (SCRmn) (3/3)

After reset: 0087H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn	RXEmn	DAPmn	CKPmn	0	EOCmn	PTCmn1	PTCmn0	DIRmn	0	SLCmn1[Note 1]	SLCmn0	DLSmn3	DLSmn2	DLSmn1[Note 2]	DLSmn0

DLSmn3	DLSmn2	DLSmn1	DLSmn0	Setting of the data length	Serial function correspondence		
					SSPI	UART	IIC
0	1	1	0	7 bits of data length (bit0 to 6 stored in the SDRmn register).	○	○	×
0	1	1	1	8 bits of data length (bit0 to 7 saved in the SDRmn register).	○	○	○
1	0	0	0	9 bits of data length (bit0 to 8 saved in the SDRmn register).	○	○	×
1	0	0	1	10 bits of data length (bit0 to 9 saved in the SDRmn register).	○	×	×
1	0	1	0	11 bits of data length (bit0 to 10 stored in the SDRmn register).	○	×	×
1	0	1	1	12-bit data length (bit0 to 11 stored in the SDRmn register).	○	×	×
1	1	0	0	13 bits of data length (bit0 to 12 saved in the SDRmn register).	○	×	×
1	1	0	1	14-bit data length (bit0 to 13 stored in the SDRmn register).	○	×	×
1	1	1	0	15 bits of data length (bit0 to 14 saved in the SDRmn register).	○	×	×
1	1	1	1	16-bit data length (bit0 to 15 stored in the SDRmn register).	○	○	×
Others:				Settings are disabled.			
In the simplified I²C mode, DLSmn3~ DLSmn0=0111B must be set.							

Note 1: SCR00, SCR10, SCR20 registers only.

Notice: bits 6, 10 and 11 must be set to "0".

Remark: m: unit number(m=0, 1, 2) n: channel number(n=0, 1) p: SSPI number(p=00, 01, 10, 11, 20, 21)

15.3.5 Serial data register mn (SDRmn)

The SDRmn register is a data register (16-bit) sent and received by the 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.

If the CCSmn bit "0" of the serial mode register mn (SMRmn) set to 0, the frequency division clock of the running clock set by bit15~9 (High 7 bits) of the SDRmn register is used as the transfer clock.

The SIRmn register is set by means of a 16-bit memory manipulation instruction.

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

Figure 15-7 Format of serial data register mn(SDRmn)

After Reset:0000H R/W

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

SDRmn[15:9]							Transmit clock setting for operating 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	$f_{MCK}/128$

Notice: 1. When operation is stopped (SEmn=0), bit8~0 must be cleared to zero.

2. When using UART, it is prohibited to set SDRmn[15:9] to "0000000B" and "0000001B".

3. When using Simplified I^2C, it is prohibited to set SDRmn[15:9] to "0000000B", and the setting value of SDRmn[15:9] must be greater than or equal to "0000001B".

4. When operation is stopped (SEmn=0), it is prohibited to rewrite SDRmn[7:0] by 8-bit memory manipulation instruction (otherwise, all of SDRmn[15:9] is cleared to "0").

Remark: 1. For the function of the SDRmn register during operation, please refer to “15.2 Structure of universal serial communication unit”.

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

15.3.6 Serial flag clear trigger register mn (SIRmn)

This is the trigger register used to clear the error flags of channel n.

If each bit (FECTmn, PECTmn, OVCTmn) is set to "1", the corresponding bit (FEFmn, PEFmn, OVFmn) of the serial status register mn (SSRmn) is cleared to "0". Since the SDIRmn register is a trigger register, clearing the corresponding bit of the SSRmn register also clears the SDIRmn register immediately.

The SIRmn register is set by a 16-bit memory operation instruction.

A lower 8-bit of the SIRmn register can be set with the SIRmnL and through an 8-bit memory operation instruction.

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

Figure 15-8 Format of serial flag clear trigger register mn (SIRmn)

After Reset:0000H R/W

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIRmn	0	0	0	0	0	0	0	0	0	0	0	0	0	FECTmn Note1	PECTmn	OVCTmn

FECTmn [Note 1]	Clear trigger for channel n frame error flag
0	Do not clear.
1	Clear the FEFmn bit of the SSRmn register "0".

PECTmn	Clear trigger for channel n parity error flag
0	Do not clear.
1	Clear the PEFmn bit of the SSRmn register "0".

OVCTmn	Clear trigger for channel n overflow error flag
0	Do not clear.
1	Clear the OVFmn bit of the SSRmn register "0".

Note 1. Limited to SIR01, SIR11, SIR21 registers only.

Notice bit15~3 (SIR00, SIR10, SIR20 register bit15~2) must be set to 0.

Remark 1.m: Unit number (m=0,1,2) n: Channel number (n=0,1)

2. The read value of the SIRmn register is always "0000H".

15.3.7 Serial status register mn (SSRmn)

The SSRmn register indicates the communication state of the channel n and the occurrence of an error. The errors represented are frame errors, parity errors, and overflow errors. The SSRmn register is read by a 16-bit memory operation instruction.

The lower 8 bits of the SSRmn register can be read with the SSRmnL and through the 8-bit memory operation instruction.

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

Figure 15-9 Format of serial status register mn (SSRmn) (1/2)

After reset: 0000H R

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSRmn	0	0	0	0	0	0	0	0	0	TSF mn	BFF mn	0	0	FEF mn Note1	PEF mn	OVF mn

TSFmn	Flag for channel n communication state
0	Communication Stop State or Communication Standby State
1	Communication Operational Status
[Clear Criteria] · When the STmn bit of the STm register is set to "1" or the SSmn bit of the SSm register is set to "1" · When communication ends [Set Criteria] · When communication begins	

BFFmn	Status Indication Flag for Channel n Buffer Register
0	The SDRmn register does not hold valid data.
1	The SDRmn register store valid data.
[Clear Criteria] · Transfer the transmission data from the SDRmn register to the shift register during transmission · Read out received data from the SDRmn register during the receiving process ·When the STmn bit of the STm register is set to "1" (communication stop state) or the SSmn bit of the SSm register is set to "1" (communication enable state) [Set Criteria] · Write transmit data to the SCRmn register with the TXEmn bit of the SDRmn register "1" · When saving received data to the SCRmn register with the RXEmn bit of the SDRmn register "1" (Receive, Send and Receive modes in each communication mode) · When a receive error occurs	

Note1. Limited to SSR01, SSR03, SSR11 registers only.

Notice: If the SDRmn register is written while the BFFmn bit is "1", the saved transmit or receive data is discarded and an overflow error is detected (OVEmn=1).

Remark m: Unit number (m=0,1,2) n: Channel number (n=0,1)

Figure 15-11 Format of serial status register mn (SSRmn) (2/2)

After reset: 0000H R

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSRmn	0	0	0	0	0	0	0	0	0	TSF mn	BFF mn	0	0	FEF Mn Note1	PEF mn	OVF mn

FEFmn[Note1]	Detection flag for channel n frame errors
0	No error occurred.
1	An error occurred (when UART received).
[Clear Criteria] •When writing "1" to the FECTmn bit of the SIRmn register [Set Criteria] •When no stop bit is detected at the end of UART reception	

PEFmn	Detection flag for channel n parity error
0	No error occurred.
1	An error occurred (when UART received) or ACK was not detected (when I^{2}C sent).
[Clear Criteria] •When writing "1" to the PECTmn bit of the SIRmn register [Set Criteria] •When sending data with different parity and parity bits (parity errors) at the end of UART reception •At the time of I^{2}C transmission and at the time of ACK reception the slave did not return an ACK signal (no ACK was detected)	

OVFmn	Detection flag for channel n overflow error
0	No error occurred.
1	An error has occurred.
[Clear Criteria] •When writing "1" to the OVCTmn bit of the SIRmn register [Set Criteria] •In the SCRmn register, the RXEmn bit is "1" (reception mode, transmission and reception mode in each communication mode) •Data is not ready to be sent during a slave send or slave send and receive in SSPI mode	

Note 1. Limited to SSR01, SSR03, SSR11 registers only.

Notice m: Unit number (m=0,1,2) n: Channel number (n=0,1)

15.3.8 Serial channel start register m (SSm)

The SSm register is a trigger register that sets a communication/start count for each channel.

If you write "1" to each (SSmn), set "1" to the corresponding bit (SEmn) of the serial channel allowed register m (SEm). Because the SSmn bit is a trigger bit, if the SEmn bit is "1", clear the SSmn bit immediately.

The SSm register is set by a 16-bit memory operation instruction.

A lower 8-bit of the SSm register can be set with the SSmL and through an 8-bit memory operation instruction.

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

Figure 15-10 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	SSm1	SSm0

SSmn	Trigger at the beginning of channel n operation
0	No trigger.
1	Set the SEmn bit "1" and shift to communication standby state [Note].

Note If the SSmn bit is set to "1" during communication, communication will be stopped and enter standby state. At this time, the values of control register and shift register, SCLKmn pin and SDOmn pin, FEFmn flag, PEFmn flag and OVFmn flag remain in state.

Notice bit15~2 of SSm register must be set to '0'.

Remark 1. m: Unit number (m=0,1,2) n: Channel number (n=0,1)

2. SSm register always reads '0000H'.

15.3.9 Serial channel stop register m (STm)

The STm register is a trigger register that sets a communication/stop count that allows each channel.

If a "1" is written to each bit (STmn), the corresponding bit (SEmn) in the serial channel enable status register m (SEm) is cleared to "0" (stop status). Since the STmn bit is a trigger bit, if the SEmn bit is "0", the STmn bit is cleared immediately.

The STm register is set by a 16-bit memory operation instruction.

A lower 8-bit of the STm register can be set with the STmL and through an 8-bit memory operation instruction.

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

Figure 15-11 Format of serial channel stop register m (STm)

After reset: 0000H R/W

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ST01	ST00

STmn	Stop trigger for channel n operation
0	No trigger.
1	Clear the SEmn bit "0" to stop the communication run [Note].

Note The value of the control and shift registers, the SCLKmn and SDOmn pins, and the FEFmn, PEFmn, and OVFmn flags remain in state.

Notice Bit15~2 of STm register must be set to '0'.

Remark 1. m: Unit number (m=0,1,2) n: Channel number (n=0,1)

2. The read value of the STm register is always "0000H".

15.3.10 Serial channel enable state register m (SEm)

The SEm register is used to confirm the allowed or stopped states of serial transmission and reception of each channel.

If a "1" is written to each bit of the serial start enable register m (SSm), its corresponding bit is set to "1". If "1" is written to each bit of the serial channel stop register m (STm), the corresponding bit is cleared to "0".

For a channel n, the value of the CKOmn bit (serial clock output of channel n) of the subsequent serial output register m (SOm) cannot be overridden by software.

For a stopped channel n, the value of the CKOmn bit of the SOm register can be set by software and output from the serial clock pin. Thus, any waveform such as a start condition or a stop condition can be generated by software.

The SEm register is read by a 16-bit memory operation instruction.

The lower 8 bits of the SEm register can be read with the SEmL and through the 8-bit memory operation instruction.

The value of the SEm register changes to "0000H" after the reset signal is generated.

Figure 15-12 Format of serial channel enable state 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	0	SE01	SE00

SEm	Indication of the enable or stop state of channel n operation
0	Stop state
1	Operation enable state

Remark m: Unit number (m=0,1,2) n: Channel number (n=0,1)

15.3.11 Serial output enable register m (SOEm)

The SOEm register setting allows or stops the output of serial communication for each channel.

For channel n that allows serial output, the value of the SOmn bit of the serial output register m (SOm) cannot be overridden by software.

For a channel n that stops serial output, the value of the SOmn bit of the SOm register can be set by software and output from the serial data output pin. Thus, any waveform such as a start condition or a stop condition can be generated by software.

The SOEm register is set by a 16-bit memory operation instruction.

A lower 8-bit of the SOEm register can be set with the SOEmL and through an 8-bit memory operation instruction.

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

Figure 15-13 Format of serial output enable register m (SOEm)

After reset: 0000H R/W

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE 01	SOE 00

SOEmn	Channel n serial output enable or stop
0	Stop the output of serial communication.
1	Enable the output of serial communication.

Note Bit15~2 of SOEm register must be set to '0'.

Remark m: Unit number (m=0,1,2) n: Channel number (n=0,1)

15.3.12 Serial output register m (SOm)

The SOm register is a buffer register for serial output of each channel.

The value of the SOmn bit of this register is output from the serial data output pin of channel n.

The value of the CKOmn bit of this register is output from the serial clock output pin of channel n.

The SOmn bit of this register can be overwritten by software only if serial output is prohibited (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, ignoring the rewriting of the software.

The CKOmn bit of this register can be overridden by software only if the channel is stopped running (SEmn=0). When the channel is allowed to run (SEmn=1), the value of the CKOmn bit of this register can only be changed by serial communication, ignoring the rewriting of the software.

To use a serial interface pin as a non-serial interface function such as a port function, the corresponding CKOmn bit and SOmn bit must be set to "1".

The SOm register is set by a 16-bit memory operation instruction.

After the reset signal is generated, the value of the SOm register changes to "0303H".

Figure 15-14 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	0	0	CKOm1	CKOm0	0	0	0	0	0	0	SOm1	SOm0

CKOmn	Serial clock output of channel n
0	The output value of the serial clock is "0".
1	The output value of the serial clock is "1".

SOmn	Serial data output of channel n
0	The output value of serial data is "0".
1	The output value of serial data is "1".

Note Bit15~10 and bit7~2 of the SO1 register must be set to 0.

Remark m: Unit number (m=0,1,2) n: Channel number (n=0,1)

15.3.13 Serial output level register m (SOLm)

The SOLm register is a register that sets the reverse phase of the data output level of each channel.

This register can be set only in UART mode. In SSPI mode and Simplified I²C mode, the corresponding bit must be set to "0". Only when serial output is allowed (SOEmn=1), the inverted setting of each channel n of this register is reflected to the pin output. When serial output is prohibited (SOEmn=0), the value of the SOmn bit is output directly. Rewriting the SOLm register during operation (SEmn=1) is prohibited.

The SOLm register is set by a 16-bit memory operation instruction.

A lower 8-bit of the SDOLm register can be set with the SOLmL and through an 8-bit memory operation instruction.

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

Figure 15-15 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
SOLm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOLm0

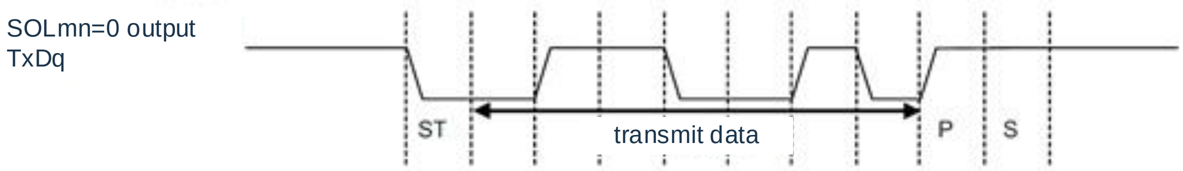
SOLmn	Selection of channel n transmission data level inversion in UART mode
0	The communication data is output directly.
1	Invert the communication data.

Note Bit15-1 of SOL0, SOL1 and SOL2 registers must be set to '0'.
Remark m: Unit number (m=0,1,2) n: Channel number (n=0,1)

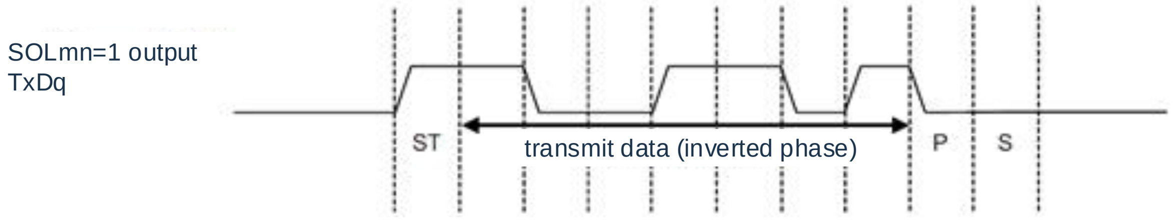
When UART transmission is performed, a level inversion example of the transmitted data is shown in Figure 15-16.

Figure 15-17 Example of level inversion for transmitting data

(a)positive phase output (SOLmn=0)



(b)inverted phase output (SOLmn=0)



Remark: m: Unit number (m=0,1,2) n: Channel number (n=0,1)

15.3.14 Input switch control register (ISC)

When LIN-bus communication is implemented by UART0, the ISC1 bit and ISC0 bit of the ISC register are used for external interrupts and coordination of timer array units. If bit0 is set to '1', the input signal of the serial data input (RxD0) pin is selected as the input of the external interrupt (INTP0), thereby detecting the wake-up signal by INTP0 interrupt.

If bit1 is set to "1", the input signal of the serial data input (RxD0) pin is selected as the input of the timer, so the timer can detect the wake-up signal and measure the low level width of the break field and the pulse width of the sync field.

The SS1E00 bit controls the SS00 pin input for channel 0 in the slave mode of SSPI00 communication. During the period of inputting a high level to the SS00 pin, no transmission and reception are performed even if a serial clock is inputted; during the period of inputting a low level to the SS00 pin, if a serial clock is input, transmission and reception are performed according to the settings of each mode.

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

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

Figure 15-18 Format of input switch control register (ISC)

Address: 40040473H After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
ISC	SIE00	0	0	0	0	0	ISC1	ISC0

SDIE00	Settings for SS00 input for channel 0 in slave mode of SSPI00 communication
0	Invalid SS00 pin input.
1	SS00 pin input is valid.

ISC1	Input switching of channel 3 of timer Timer4
0	Use the input signal of the TI03 pin as the input (usually running) of the timer.
1	The input signal of the RxD0 pin is used as the input of the timer (detecting the wake-up signal and measuring the low level width of the break field and the pulse width of the sync field).

ISC0	External Interrupt (INTP0) input switch
0	Use the input signal of the INTP0 pin as the input for the external interrupt (usually run).
1	Use the input signal of the RxD0 pin as the input of the external interrupt (detect wake-up signal).

Note Bit6~2 must be set to '0'.

15.3.15 Noise filter enable register 0 (NFEN0)

The NFEN0 register sets whether the noise filter is used for the input signal of the serial data input pin of each channel.

For pins used for SSPI or simplified I^{2}C communication, the corresponding bit must be set to "0" to disable the noise filter. For pins used for UART communication, the corresponding bit must be set to "1" to make the noise filter active.

When the noise filter is valid, the 2 clocks are checked for consistency after synchronization through the object channel's running clock (f_{MCK}); when the noise filter is invalid, synchronization is performed only through the object channel's running clock (f_{MCK}).

The NFEN0 register is set by an 8-bit memory operation instruction.

After the reset signal is generated, the value of the NFEN0 register changes to '00H'.

Figure 15-19 Format of noise filter enable register 0 (NFEN0).

Address: 40040470H After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
NFEN0	0	0	0	SNFEN20	0	SNFEN10	0	SNFEN00

SNFEN20	Whether the noise filter of the RxD2 pin is used or not
0	Noise filter is OFF
1	Noise filter is ON
When used as the RxD2 pin, the SNFEN20 bit must be set to "1". When used for functions other than the RxD2 pin, the SNFEN20 bit must be set to "0".	

SNFEN10	Whether the noise filter of the RxD1 pin is used or not
0	Noise filter is OFF
1	Noise filter is ON
When used as the RxD1 pin, the SNFEN10 bit must be set to "1". When used for functions other than the RxD1 pin, the SNFEN10 bit must be set to "0".	

SNFEN00	Whether the noise filter of the RxD0 pin is used or not
0	Noise filter is OFF
1	Noise filter is ON
When used as the RxD0 pin, the SNFEN00 bit must be set to "1". When used for functions other than the RxD0 pin, the SNFEN00 bit must be set to "0".	

Note Bit7~5,3,1 must be set to ‘0’.

15.3.16 Registers for controlling serial input/output pin port

When using the Universal Serial Communication Unit, the control registers for the multiplexed port function (Port Mode Register (PMxx), Port Multiplexed Function Configuration Register (PxxCFG), Port Output Mode Register (POMxx), and Port Mode Control Register (PMCxx)) must be set.

For details, please refer to "Chapter 2 Port Function".

When using the multiplexed port of serial data output pin or serial clock output pin as serial data output or serial clock output, the bit of Port Mode Control Register (PMCxx) and the bit of Port Mode Register (PMxx) corresponding to each port must be set to "0". In this case, the Port Register (Pxx) bit can be "0" or "1".

In addition, when used in N-channel open-drain output mode, the bit of the port output mode register (POMxx) corresponding to each port must be set to "1".

When using the multiplexed port of the serial data input pin or serial clock input pin as serial data input or serial clock input, you must set the bit of the Port Mode Register (PMxx) corresponding to each port to "1" and set the bit of the Port Mode Control Register (PMCxx) to "0". In this case, the Port Register (Pxx) bits can be "0" or "1".

15.4 Run stop mode

Each serial interface of the universal serial communication unit has a run stop mode. Serial communication is not possible in the run stop mode, so power consumption can be reduced. In addition, pin used for serial interface can be used as port function in idle mode.

15.4.1 Stop operation by unit

Sets the stop run in units by the Peripheral Enable Register 0 (PER0).

The PER0 register is a register that sets a clock that is allowed or prohibited to supply to each peripheral hardware. Reduce power consumption and noise by providing a clock to a hardware that is not in use.

To stop the Universal Serial Communication Unit 0, you must set the bit2(SCI0EN) to "0"; To stop Universal Serial Communication Unit 1, you must set bit3 (SCI1EN) to "0"; To stop Universal Serial Communication Unit 2, you must set bit4 (SCI2EN) to "0".

Figure 19-23 Setting of peripheral enable register 0 (PER0) when unit-based stop operation

(a) Peripheral enable register 0 (PER0)...... Only the corresponding bit of SCIm to be stopped is set to "0".

	7	6	5	4	3	2	1	0
PER0	XX	XX	XX	SCI2EN	SCI1EN	SCI0EN	XX	XX

Control of SCIm Input Clock
0: Stop providing input clock
1: Provide input clock

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. However, the following registers are excluded:

- Input switch control register (ISC)
- Noise filter allows register 0 (NFEN0).
- Port multiplexing function configuration register (PxxCFG)
- Port output mode register (POMx)
- Port mode register (PMx)
- Port register (Px)

Remark ×: This is the unused bit of the universal serial communication unit (depending on the setting of other peripheral functions).

0/1: The "0" or "1" is set according to the user.

15.4.2 Stopping the operation by channels

Stop operation by channel via each of the following register settings.

Figure 15-20 Each register setting when stopping the operation by channels

(a) Serial channel stop register m (STm)......This is a register that sets the 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	STm1 0/1	STm0 0/1

1: Clear SEmn bit '0' and stop communication running

※ Because the STmn bit is a trigger bit, clear the STmn bit immediately if the SEmn bit is "0".

(b) Serial channel allowed state register m (SEm)......The register indicates the running or stopping state of data transmission and reception of 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	SEm1 0/1	SEm0 0/1

0: Idle Status

The ※SEm register is a read-only state register, which stops running through the STm register. For a channel that has stopped running, the value of the CKOmn bit of the SOm register can be set by software.

(c) Serial output enable register m(SOEm)......This is a register that sets the permission or stop of serial communication output for each channel.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOEm1 0/1	SOEm0 0/1

0: Stop output by serial communication operation

※ For channels that have stopped serial output, the value of the SOmn bit of the SOm 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	0	CKOm1 0/1	CKOm0 0/1	0	0	0	0	0	0	SOm1 0/1	SOm0 0/1

1: Serial clock output value is "1"

1: The output value of serial data is "1"

※ When the pin corresponding to each channel is used as a port function, the corresponding CKOmn bit and SOmn bit must be set to "1".

Note It is limited to universal serial communication unit 0.

Note 1. m: Unit number (m=0,1,2) n: Channel number (n=0,1)

2. ▨ : Cannot set (set initial value). 0/1: The "0" or "1" is set according to the user.

15.5 3-wire serial I/O(SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) communication

This is clock synchronization communication through three lines of SCLK and SDO (SDI and SDO).

[Transmitting and Receiving Data]

- 7~16 bit data length
- Phase control for transmitting and receiving data
- MSB/LSB First

[Clock Control]

- Master or subordinate selection
- Phase control of input/output clock
- A transmission period generated by a pre-divider and an intra-channel counter is set.
- Maximum transfer rate note

  Master Communications: Max.f_{CLK}/2

  Slave Communication: Max.f_{MCK}/6

[Interrupt Function]

- Interrupt transmission end, buffer null interrupt

[Error Detection Flag]

- Overflow error

Note Must be used within a range that satisfies the SCLK Cycle Time (t_{KCY}) characteristic. Refer to the data guide for details.

Channels 0~1 for SCI0, 0~1 for SCI1 and 0~1 for SCI2 are channels that support 3-wire serial I/O (SSPI00,SSPI01,SSPI10,SSPI11,SSPI20,SSPI21).

The 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) has the following 6 communication operations:

- Master transmission (Refer to 15.5.1)
- Master reception (Refer to 15.5.2)
- Master transmission and reception (Refer to 15.5.3)
- Slave transmission (Refer to 15.5.4)
- Slave reception (Refer to 15.5.5)
- Slave transmission and reception (Refer to 15.5.6)

15.5.1 Master transmission

Master transmission refers to the operation of the product output transmission clock and sending data to other devices.

3-Wire Serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channel	Channel 0 for SCI0	Channel 1 for SCI0	Channel 2 for SCI0	Channel 3 for SCI0	Channel 0 for SCI1	Channel 1 for SCI1
Pin used	SLK00, SDO00	SLK01, SDO01	SLK10, SDO10	SLK11, SDO11	SLK20, SDO20	SLK21, SDO21
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21
	Interrupt at that end of the transfer may be selecte (single transfer mode) or buffer air-discontinuity (continuous transfer mode).					
Error detection flag	None					
Length of transmit data	7 ~ 16 bits					
Transfer Rate Note	Max.f_{CLK}/2[Hz] Min.$f_{CLK}/(2\times2^{11}\times128)$ [Hz] f_{CLK}: 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: forward • CKPmn=1: inverted					
Data orientation	MSB First or LSB First					

Note It must be used within the scope of peripheral functional characteristics (reference data manual) that meet this condition and electrical characteristics.

Remark m: Unit number (m=0,1,2) n: Channel number (n=0,1) mn=00～01, 10～11, 20～21

(1) Register settings

Figure 15-21 3-wire Serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20,SSPI21) Example of register setting content in master transmission

(a) serial mode register mn (SMRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn	CCSmn						STSmn		SISmn0				MDmn2	MDmn1	MDmn0
	0/1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0/1

channel n operational clock (fMCK)
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

interrupt source of channel n
0: Transmit completion interrupt
1: Buffer empty interrupt

(b) serial communication operation configuration registermn mn(SCRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn	RXEmn	DAPmn	CKPmn		EOCmn	PTCmn1	PTCmn0	DIRmn		SLCmn1	SLCmn0	DLSm n3	DLSm n2	DLSm n1	DLSm n0
	1	0	0/1	0/1	0	0	0	0	0/1	0	0	0	0/1	0/1	0/1	0/1

data and clock phase selection (details refer to " control universal serial communication unit registers)

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

data length configuration:
DLSmn3~0：7~16 bit data length selection

(c) serial data registermn mn(SDRmn)

(1) When operation stops (Semn=0)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration (operation clock (fmck) scaling configuration)							0	0	0	0	0	0	0	0	0

(2) During operation (SEmn=1) (lower 8 bits: SDRmnL)

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

SDRmnL

(d) serial output register m(SOm) ......Only configure bit of target channel

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm					CKOm3	CKOm2	CKOm1	CKOm0					SOm3	SOm2	SOm1	SOm0
	0	0	0	0	0/1	0/1	0/1	0/1	0	0	0	0	0/1	0/1	0/1	0/1

when clock phase is "positive phase" (CKPmn of SCRmn register as 0), "1" means starting communication; when clock phase is "inverted phase" (CKPmn=1), "0" means starting communication.

(e) serial output enable registerm (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	0/1	0/1

(f) serial channel start registerm (SSm)….only set bit of target channel to 1.

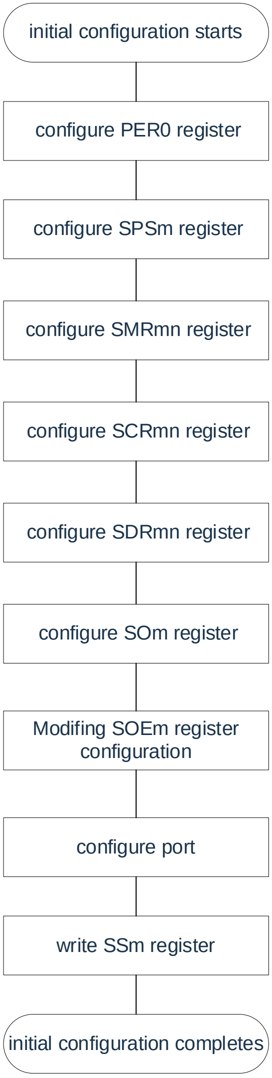
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm													SSm3	SSm2	SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	0/1	0/1	0/1	0/1

Note 1. m: Unit number (m=0,1,2) n: Channel number (n=0,1) mn=00～01, 10～11, 20～21.

2. ▨ : Cannot set (set initial value). 0/1: The "0" or "1" is set according to the user.

(2) Procedure

Figure 15-22 Initial set-up steps for master transmission



release universal serial communication unit from reset state, start providing clock.

configure operational clock

configure operational mode..etc.

configure communication format

configure transmit baud rate (configure operationl clock(fMCK) scaled transmission clock)

configure serial clock (CKOmn) and serial data(SOmn) initial output voltage

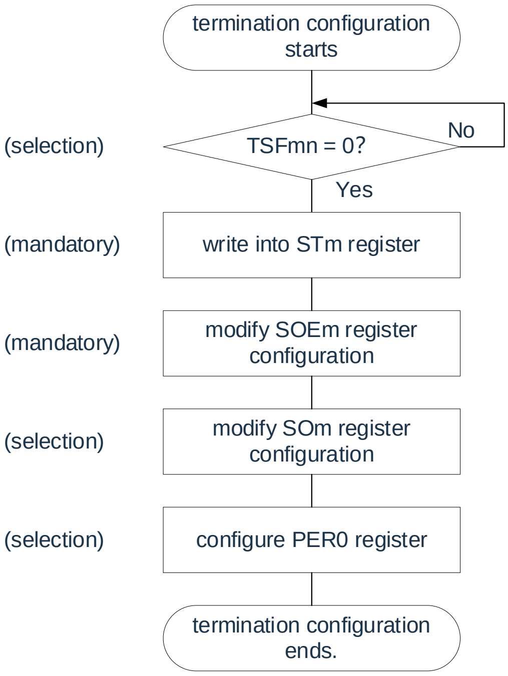
set SOEmn bit to 1, allow target channel data output

configure port register and port mode register (target channel data output and clock output are valid)

set SSmn bit of target channel to 1 (Semn=1: set as operation enable state)

Complete initial configuration.
If transmit data to SDRmn register, then communcation starts.

Figure 15-23 Stop steps for master transmission


termination configuration starts
(selection)
TSFmn = 0?
No
Yes
(mandatory)
write into STm register
(mandatory)
modify SOEm register configuration
(selection)
modify SOm register configuration
(selection)
configure PER0 register
termination configuration ends.

if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).

set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).

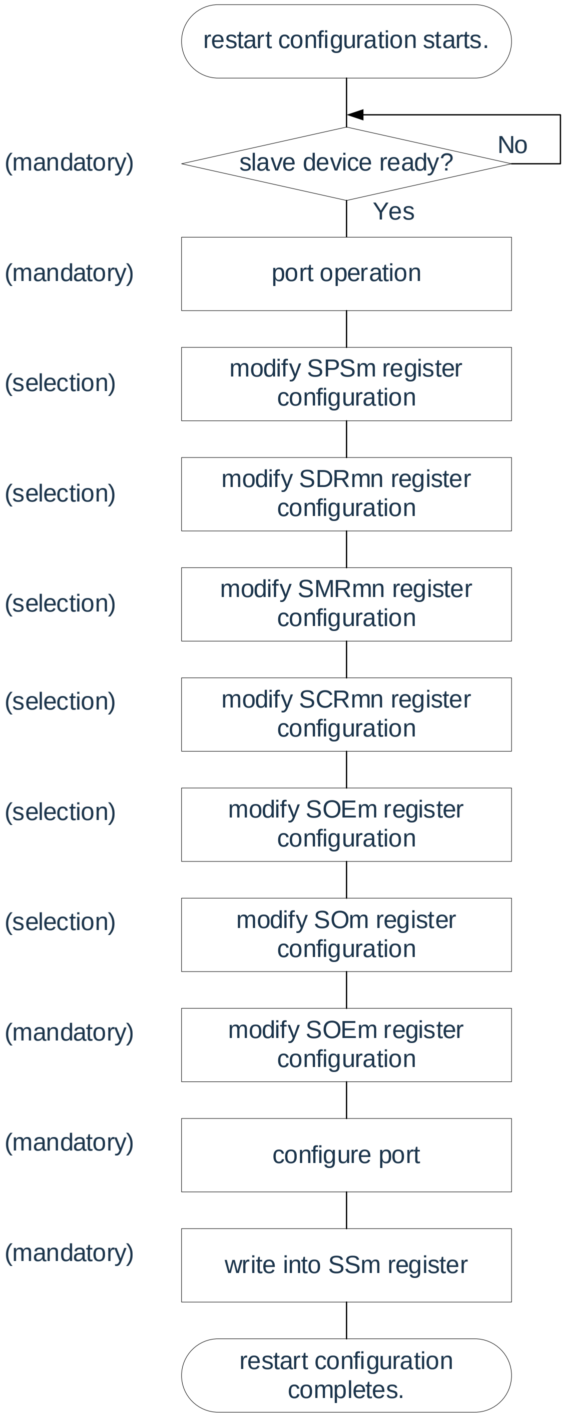
set SOEmn bit to 0, stop output of target channel

while emergency stop, based on needs, modify serial clock (CKOmn) and serial data(Somn) voltage of target channel.

when using deep sleep mode, stop clock of universal serial communication unit, configure to reset state.

finish termination configuration, enter into next processing.

Figure 15-24 Restart set-up steps for master transmission


restart configuration starts.
(mandatory)
slide device ready?
No
Yes
(mandatory)
port operation
(selection)
modify SPSm register configuration
(selection)
modify SDRmn register configuration
(selection)
modify SMRmn register configuration
(selection)
modify SCRmn register configuration
(selection)
modify SOEm register configuration
(selection)
modify SOm register configuration
(mandatory)
modify SOEm register configuration
(mandatory)
configure port
(mandatory)
write into SSm register
restart configuration completes.

wait till commuication target (slave device) stops or communication ends

via Configure port register and port mode register (data output and clock output of target channel set to invalid).

re-configure when modifying the operational clock configuration.

re-configure when modifying the transmit baud rate configuration.

re-configure when modifying serail mode register mn configuration.

re-configure when modifying serial communcation operation configuration register mn.

set SOEmn bit to 0, stop output of target channel

configure serial clock (CKOmn) and serial data(SOmn) initial output voltage

set SOEmn bit to 1, allow target channel data output

configure port register and port mode register, set target channel data output and clock output to be valid.

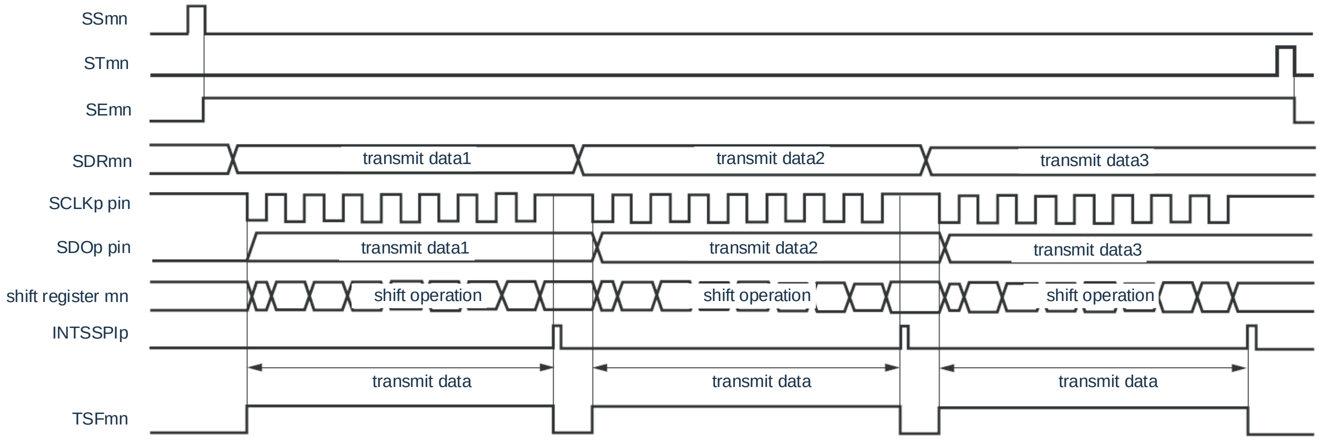
set SSmn bit of target channel to 1 (Semn=1: set as operation enable state)

configuration ends.
If configures transmission data into SDRmn register , then start communication.

Remark If you override PER0 in the abort setting to stop providing the clock, you must wait until the communication object (slave) is stopped or the communication is over for the initial set-up instead of restarting it.

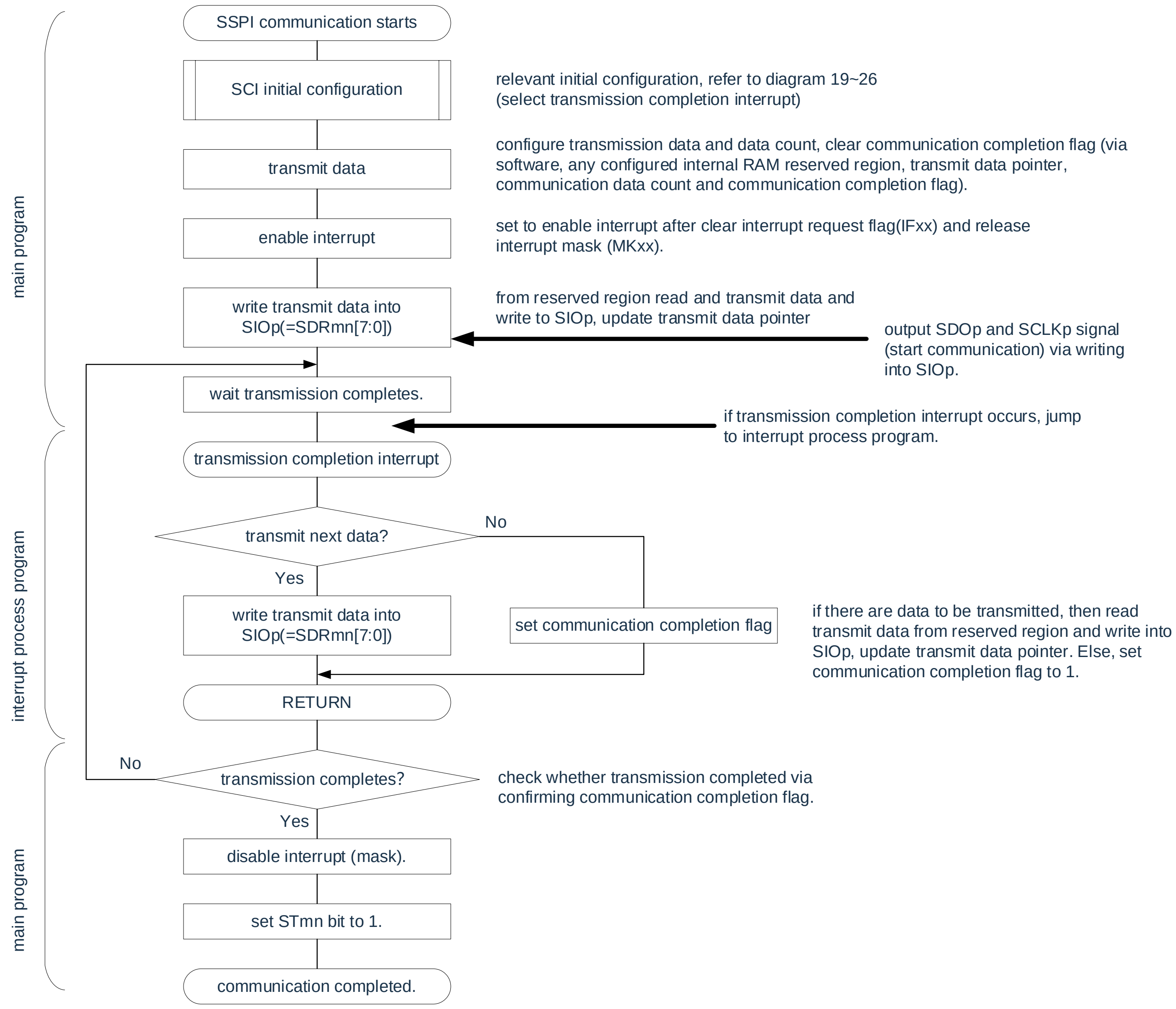
(3) Process flow (single transmit mode)

Figure 15-25 Timing diagram of master transmission (single transmit mode)
(Type 1:DAPmn=0, CKPmn=0)



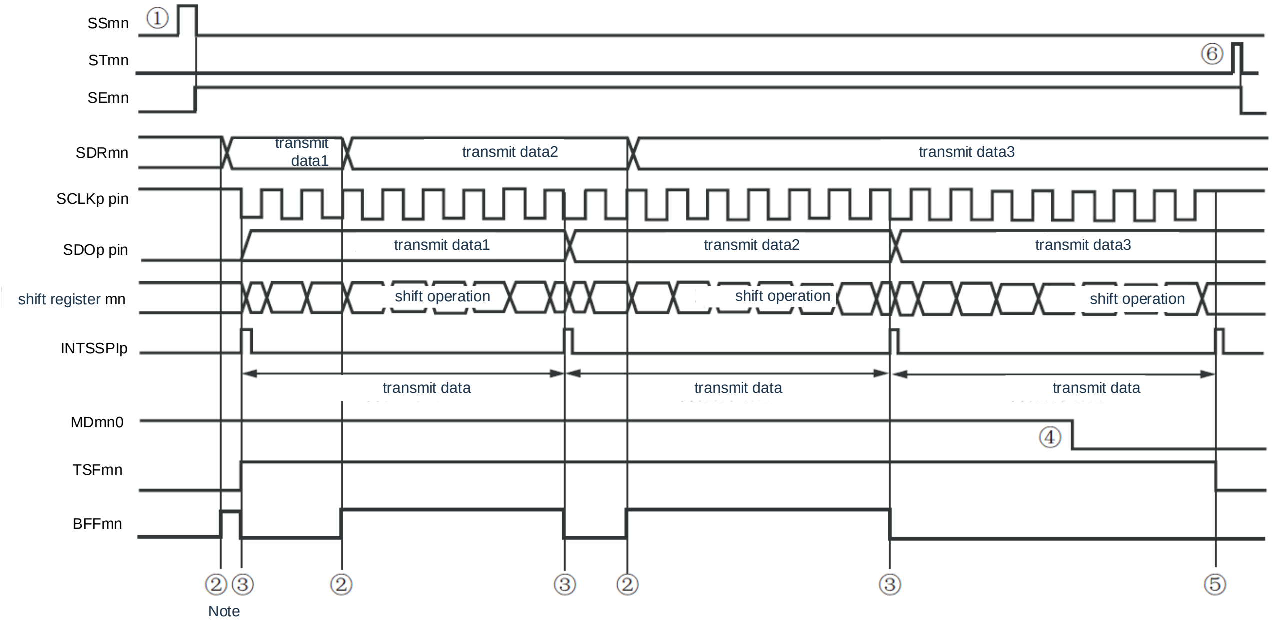
Remark m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI Number (p=00,01,10,11,20,21)

Figure 15-26 Flowchart of master transmission (single transmit mode)


main program
interrupt process program
main program
SSPI communication starts
SCI initial configuration
relevant initial configuration, refer to diagram 19~26
(select transmission completion interrupt)
transmit data
configure transmission data and data count, clear communication completion flag (via software, any configured internal RAM reserved region, transmit data pointer, communication data count and communication completion flag).
enable interrupt
set to enable interrupt after clear interrupt request flag(IFxx) and release interrupt mask (MKxx).
write transmit data into SIOp(=SDRmn[7:0])
from reserved region read and transmit data and write to SIOp, update transmit data pointer
output SDOp and SCLKp signal (start communication) via writing into SIOp.
wait transmission completes.
if transmission completion interrupt occurs, jump to interrupt process program.
transmission completion interrupt
transmit next data?
No
Yes
write transmit data into SIOp(=SDRmn[7:0])
set communication completion flag
if there are data to be transmitted, then read transmit data from reserved region and write into SIOp, update transmit data pointer. Else, set communication completion flag to 1.
RETURN
No
transmission completes?
check whether transmission completed via confirming communication completion flag.
Yes
disable interrupt (mask).
set STmn bit to 1.
communication completed.

(4) Process flow (continuous transmit mode)

Figure 15-27 Timing diagram of master transmission (continuous send mode)
(Type 1:DAPmn=0, CKPmn=0)

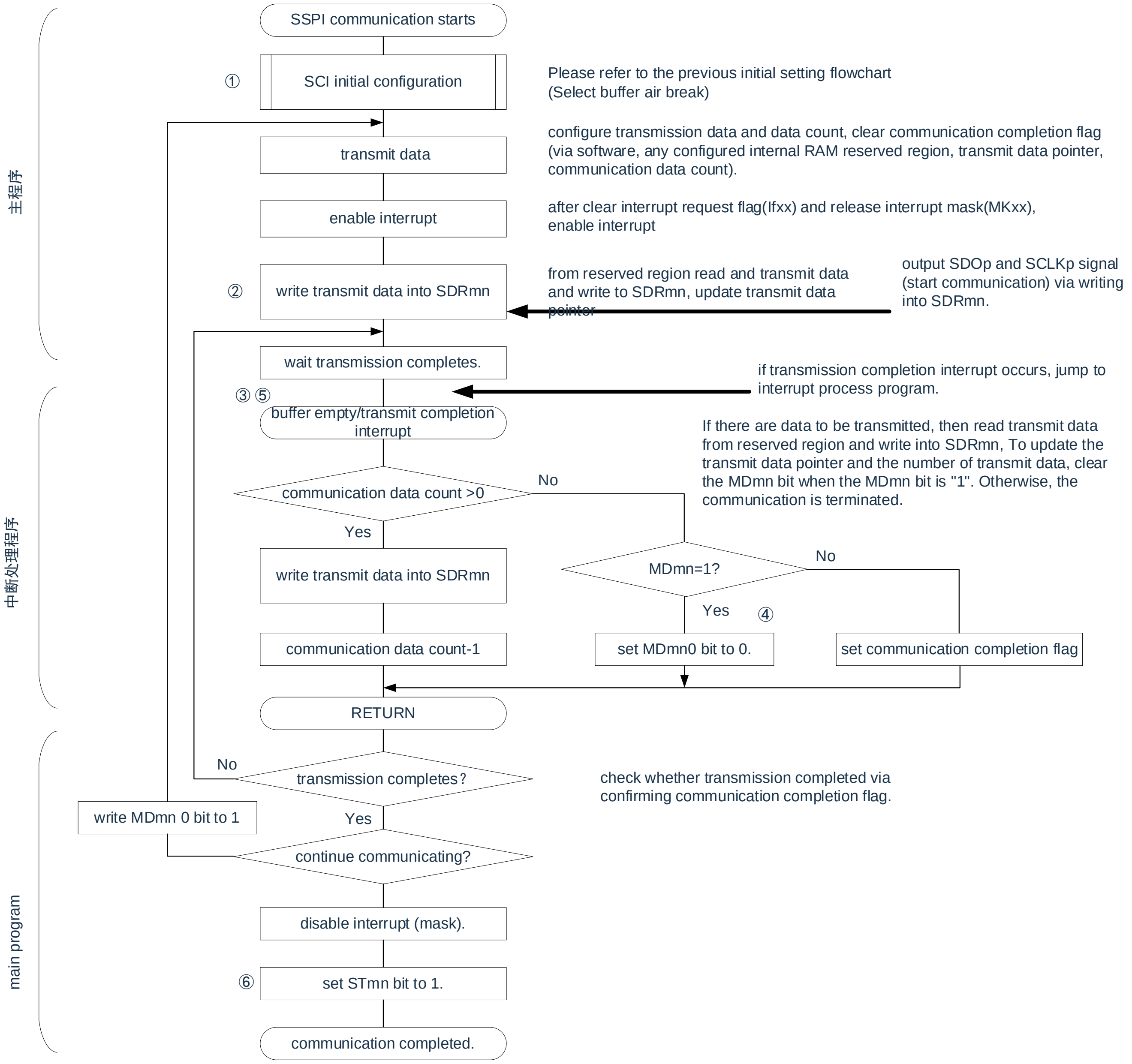


Note The transmission data is rewritten if the BFFmn bit of the serial status register mn (SSRmn) is "1" (SDRmn) when the valid data is stored in the serial data register mn (SDRmn).

Notice The MDmn0 bit of the serial mode register mn(SMRmn) can be rewritten even in operation. However, in order to be able to catch the end of the transmission of the last transmitted data interrupt, it is necessary to override before starting the last transmission.

Remark m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI Number (p=00,01,10,11,20,21)

Figure 15-28 Flow chart of master transmisison (continuous transmit mode)


主程序
中断处理程序
main program
SSPI communication starts
①
SCI initial configuration
transmit data
enable interrupt
②
write transmit data into SDRmn
wait transmission completes.
③ ⑤
buffer empty/transmit completion interrupt
communication data count >0
No
Yes
write transmit data into SDRmn
MDmn=1?
No
Yes
④
communication data count-1
set MDmn0 bit to 0.
set communication completion flag
RETURN
No
transmission completes?
write MDmn 0 bit to 1
Yes
continue communicating?
disable interrupt (mask).
⑥
set STmn bit to 1.
communication completed.

Please refer to the previous initial setting flowchart (Select buffer air break)

configure transmission data and data count, clear communication completion flag (via software, any configured internal RAM reserved region, transmit data pointer, communication data count).

after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt

from reserved region read and transmit data and write to SDRmn, update transmit data pointer

output SDOp and SCLKp signal (start communication) via writing into SDRmn.

if transmission completion interrupt occurs, jump to interrupt process program.

If there are data to be transmitted, then read transmit data from reserved region and write into SDRmn, To update the transmit data pointer and the number of transmit data, clear the MDmn bit when the MDmn bit is "1". Otherwise, the communication is terminated.

check whether transmission completed via confirming communication completion flag.

Remark: ①~⑥ in the figure correspond to ①~⑥ in "Figure 15-27 Timing diagram for master transmission (continuous transmit mode)".

15.5.2 Master reception

Master reception refers to the operation of this product outputting a transmit 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
	Interrupt at that end of the transfer may be selecte (single transfer mode) or buffer air-discontinuity (continuous transfer mode).					
Error detection flag	Only the overflow error detection flag (OVFmn).					
Length of transmit data	7 ~ 16 bits					
Transfer rate[Note]	Max.fCLK/2[Hz] Min.fCLK/($2\times2^{11}\times128$) [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: forward • CKPmn=1: inverted					
Data orientation	MSB First or LSB First					

Note: It must be used within the scope of peripheral functional characteristics (reference data manual) that meet this condition and electrical characteristics.

Remark: m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI Number (p=00,01,10,11,20,21)

(1) Register settings

Figure 15-29 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20,SSPI21)
Example of register setting content during master reception

(a) serial mode register mn (SMRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn 0/1	CCSmn 0	0	0	0	0	0	STSmn 0	0	SISmn0 0	1	0	0	MDmn2 0	MDmn1 0	MDmn0 0/1

channel n operational clock （fMCK）
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

interrupt source of channel n
0: Transmit completion interrupt
1: Buffer empty interrupt

(b) serial communication operation configuration register mn (SCRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn 0	RXEmn 1	DAPmn 0/1	CKPmn 0/1	0	EOCmn 0	PTCmn1 0	PTCmn0 0	DIRmn 0/1	0	SLCmn1 0	SLCmn0 0	DLSmn3 0/1	DLSmn2 0/1	DLSmn1 0/1	DLSmn0 0/1

data and clock phase selection (details refer to "Registers controlling universal serial communication unit)

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

Setting of data length
DLSmn3~0: 7-bit~16-bit data length selection

(c) serial data register mn (SDRmn)

(1) When operation stops (SEmn=0)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration (operation clock (fmck) scaling configuration)							0	0	0	0	0	0	0	0	0

(2) During operation（SEmn=1）（lower 8 bits: SDRmnL）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	receive data registers															

SDRmnL

(d) serial output register m(SOm) ......Only configure bit of target channel

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm	0	0	0	0	CKOm3 0/1	CKOm2 0/1	CKOm1 0/1	CKOm0 0/1	0	0	0	0	SOm3 ×	SOm2 ×	SOm1 ×	SOm0 ×

when clock phase is "positive phase" (CKPmn of SCRmn register as 0), "1" means starting communication; when clock phase is "inverted phase" (CKPmn=1), "0" means starting communication.

(e) serial output enable register m (SOEm)….not used in this mode.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	SOEm3 ×	SOEm2 ×	SOEm1 ×	SOEm0 ×

(f) serial channel start registerm (SSm)….only set bit of target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	SSm3 0/1	SSm2 0/1	SSm1 0/1	SSm0 0/1

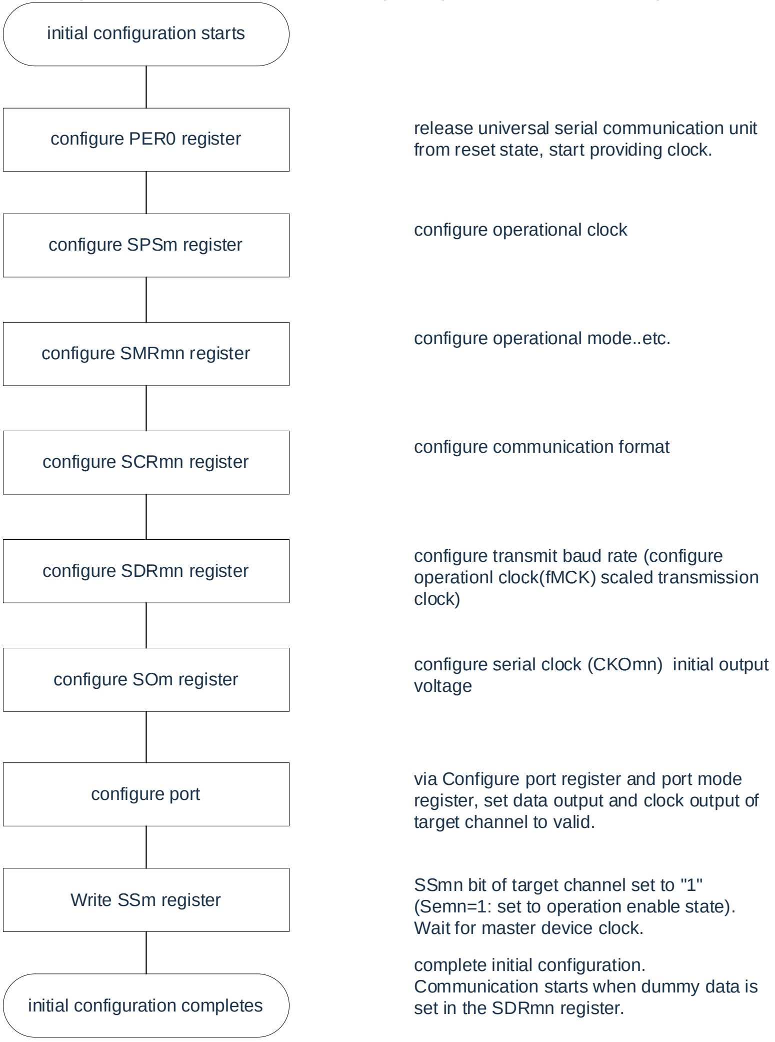
Note Limited to SCR00 and SCR01 registers, other fixed to "1".

Note 1. m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI Number (p=00,01,10,11,20,21)
mn=00~03,10~11

2. □: Fixed in SSPI Master Receive mode. ▨ : Cannot set (set initial value).
×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).
0/1: The "0" or "1" is set according to the user.

(2) Procedure

Figure 15-30 Initial set-up steps for master reception



release universal serial communication unit from reset state, start providing clock.

configure operational clock

configure operational mode..etc.

configure communication format

configure transmit baud rate (configure operationl clock(fMCK) scaled transmission clock)

configure serial clock (CKOmn) initial output voltage

via Configure port register and port mode register, set data output and clock output of target channel to valid.

SSmn bit of target channel set to "1" (Semn=1: set to operation enable state). Wait for master device clock.

complete initial configuration. Communication starts when dummy data is set in the SDRmn register.

Figure 15-31 Stop steps for master reception

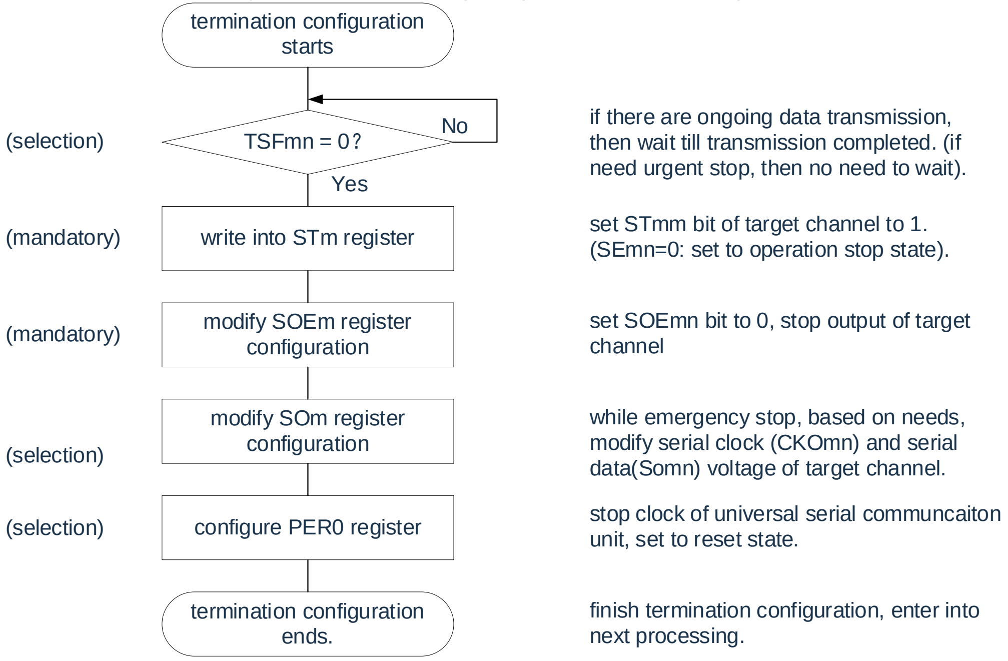
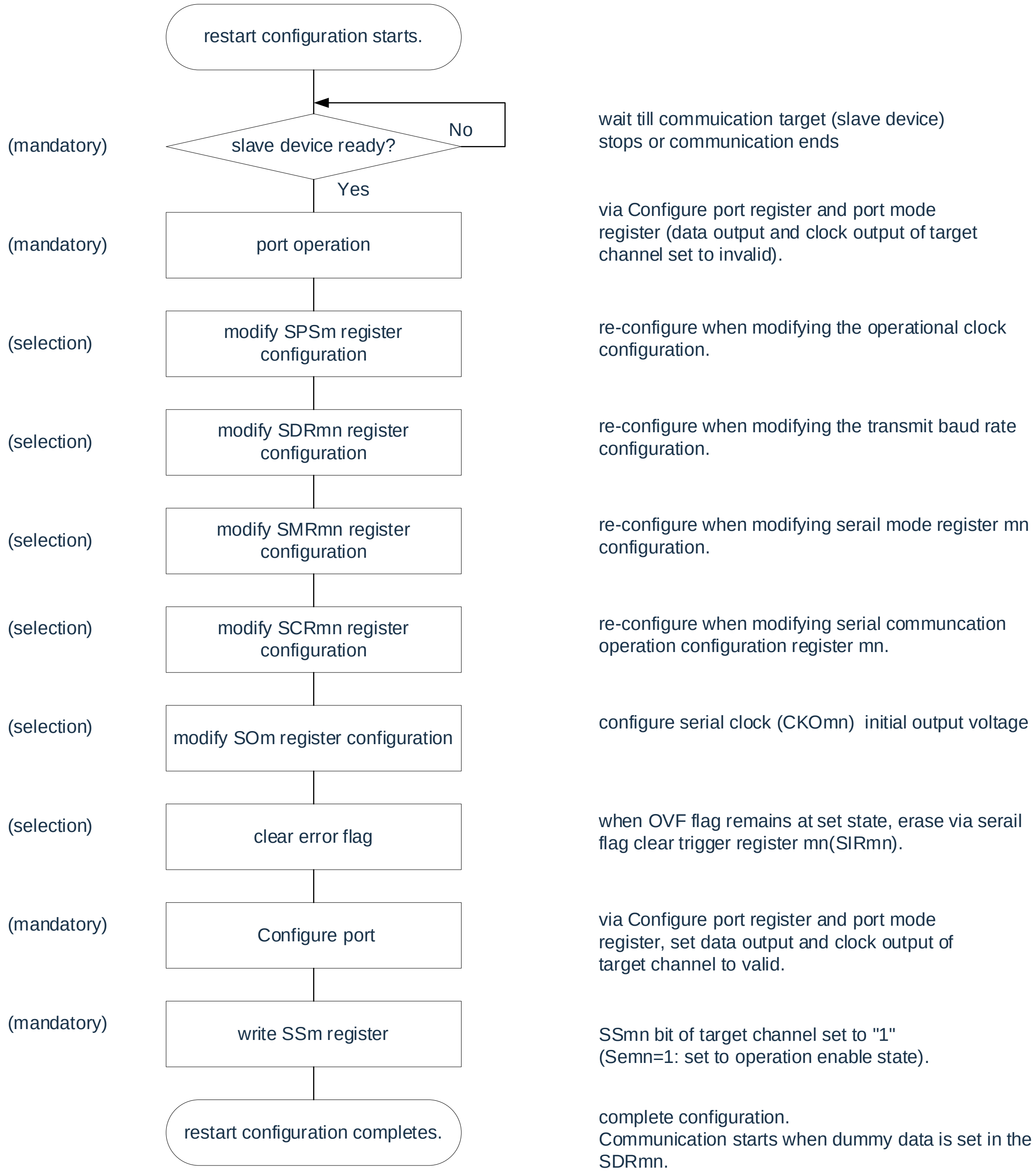

termination configuration starts
(selection)
TSFmn = 0?
No
Yes
if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).
(mandatory)
write into STm register
set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).
(mandatory)
modify SOEm register configuration
set SOEmn bit to 0, stop output of target channel
(selection)
modify SOm register configuration
while emergency stop, based on needs, modify serial clock (CKOmn) and serial data(Somn) voltage of target channel.
(selection)
configure PER0 register
stop clock of universal serial communcaiton unit, set to reset state.
termination configuration ends.
finish termination configuration, enter into next processing.

Figure 15-32 Restart set-up steps for master reception


restart configuration starts.
(mandatory)
slide device ready?
No
Yes
(mandatory)
port operation
(selection)
modify SPSm register configuration
(selection)
modify SDRmn register configuration
(selection)
modify SMRmn register configuration
(selection)
modify SCRmn register configuration
(selection)
modify SOm register configuration
(selection)
clear error flag
(mandatory)
Configure port
(mandatory)
write SSm register
restart configuration completes.

wait till commuication target (slave device) stops or communication ends

via Configure port register and port mode register (data output and clock output of target channel set to invalid).

re-configure when modifying the operational clock configuration.

re-configure when modifying the transmit baud rate configuration.

re-configure when modifying serail mode register mn configuration.

re-configure when modifying serial communcation operation configuration register mn.

configure serial clock (CKOmn) initial output voltage

when OVF flag remains at set state, erase via serail flag clear trigger register mn(SIRmn).

via Configure port register and port mode register, set data output and clock output of target channel to valid.

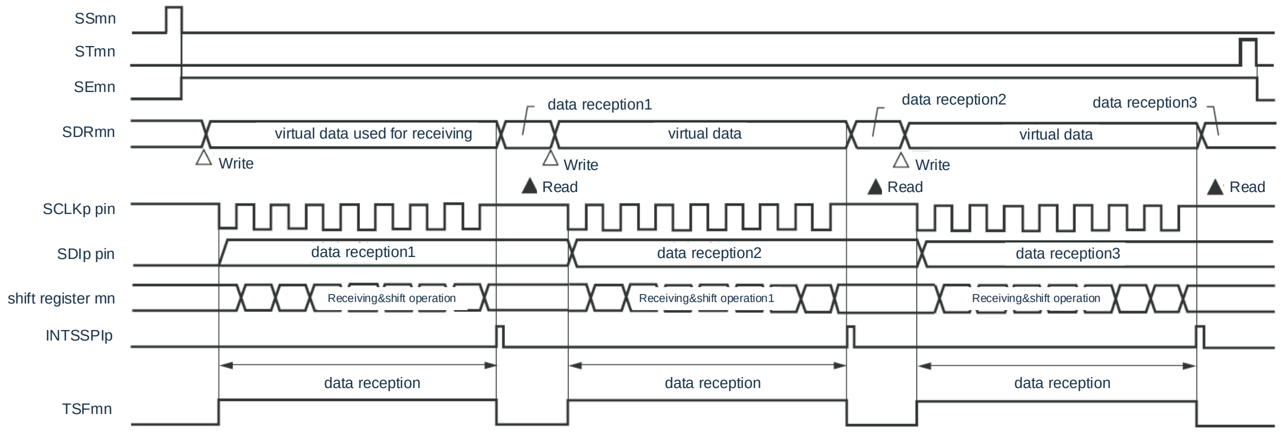
SSmn bit of target channel set to "1" (Semn=1: set to operation enable state).

complete configuration.
Communication starts when dummy data is set in the SDRmn.

Remark: If you override PER0 in the abort setting to stop providing the clock, you must wait until the communication object (slave) is stopped or the communication is over for the initial set-up instead of restarting it.

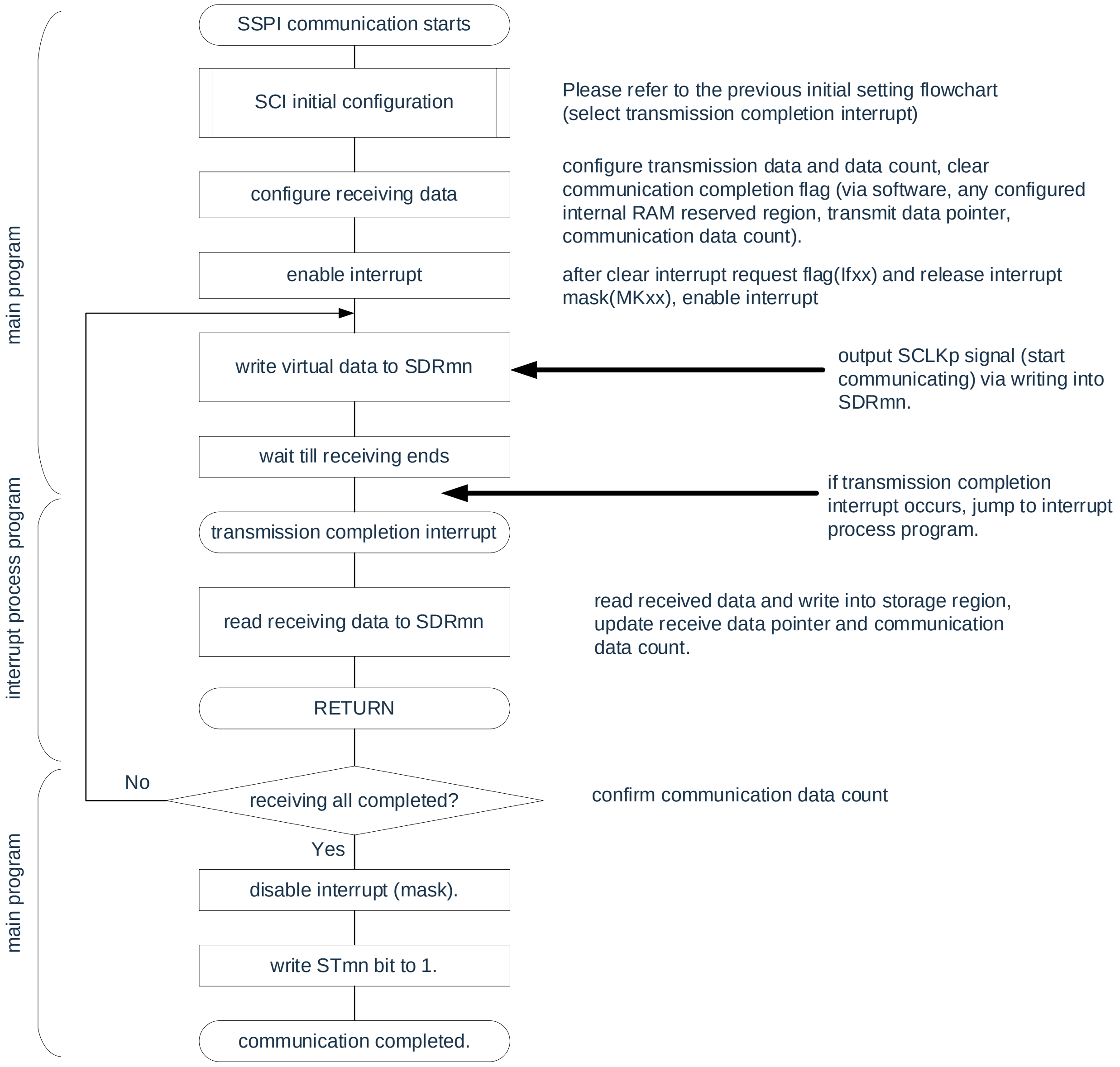
(3) Process flow (single receive mode)

Figure 15-33 Timing diagram of the master receive (single receive mode)
(Type 1:DAPmn=0, CKPmn=0)



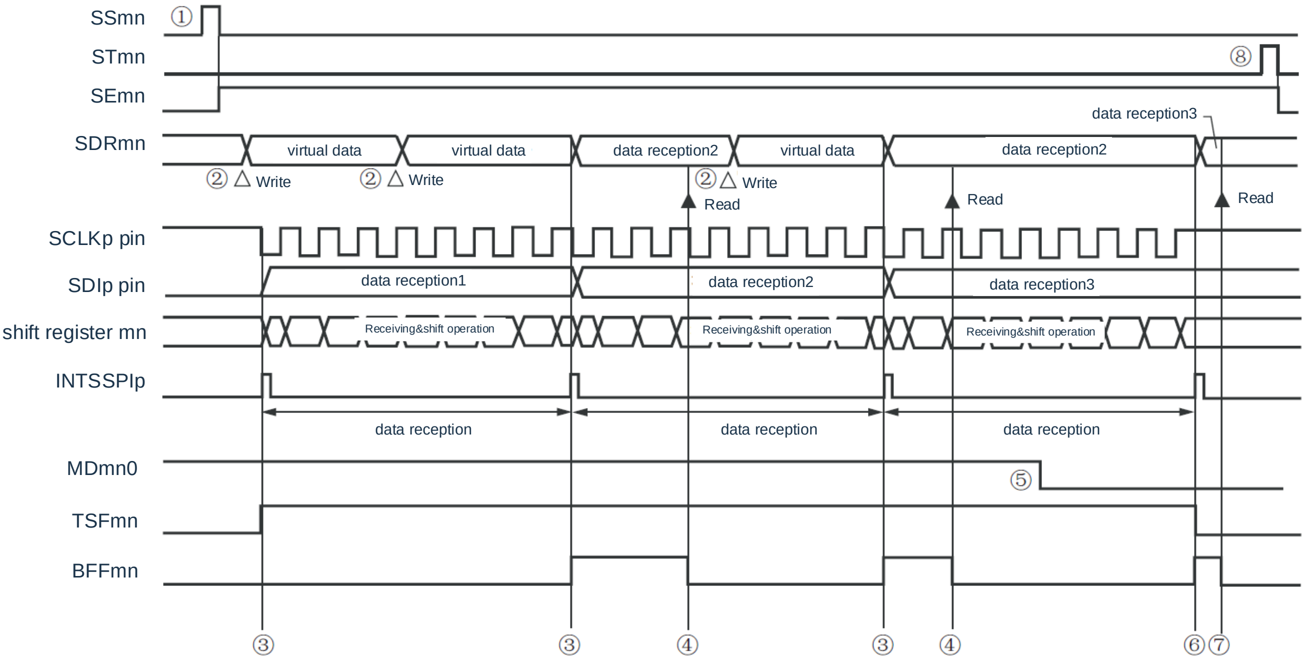
Remark m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI Number (p=00,01,10,11,20,21)

Figure 15-34 Flowchart for master reception (single receive mode)


main program
interrupt process program
main program
SSPI communication starts
SCI initial configuration
configure receiving data
enable interrupt
write virtual data to SDRmn
wait till receiving ends
transmission completion interrupt
read receiving data to SDRmn
RETURN
receiving all completed?
No
Yes
disable interrupt (mask).
write STmn bit to 1.
communication completed.
Please refer to the previous initial setting flowchart (select transmission completion interrupt)
configure transmission data and data count, clear communication completion flag (via software, any configured internal RAM reserved region, transmit data pointer, communication data count).
after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt
output SCLKp signal (start communicating) via writing into SDRmn.
if transmission completion interrupt occurs, jump to interrupt process program.
read received data and write into storage region, update receive data pointer and communication data count.
confirm communication data count

(4) Process flow (continuous receive mode)

Figure 15-35 Timing diagram of master reception (continuous receive mode)
(Type 1: DAPmn=0, CKPmn=0)

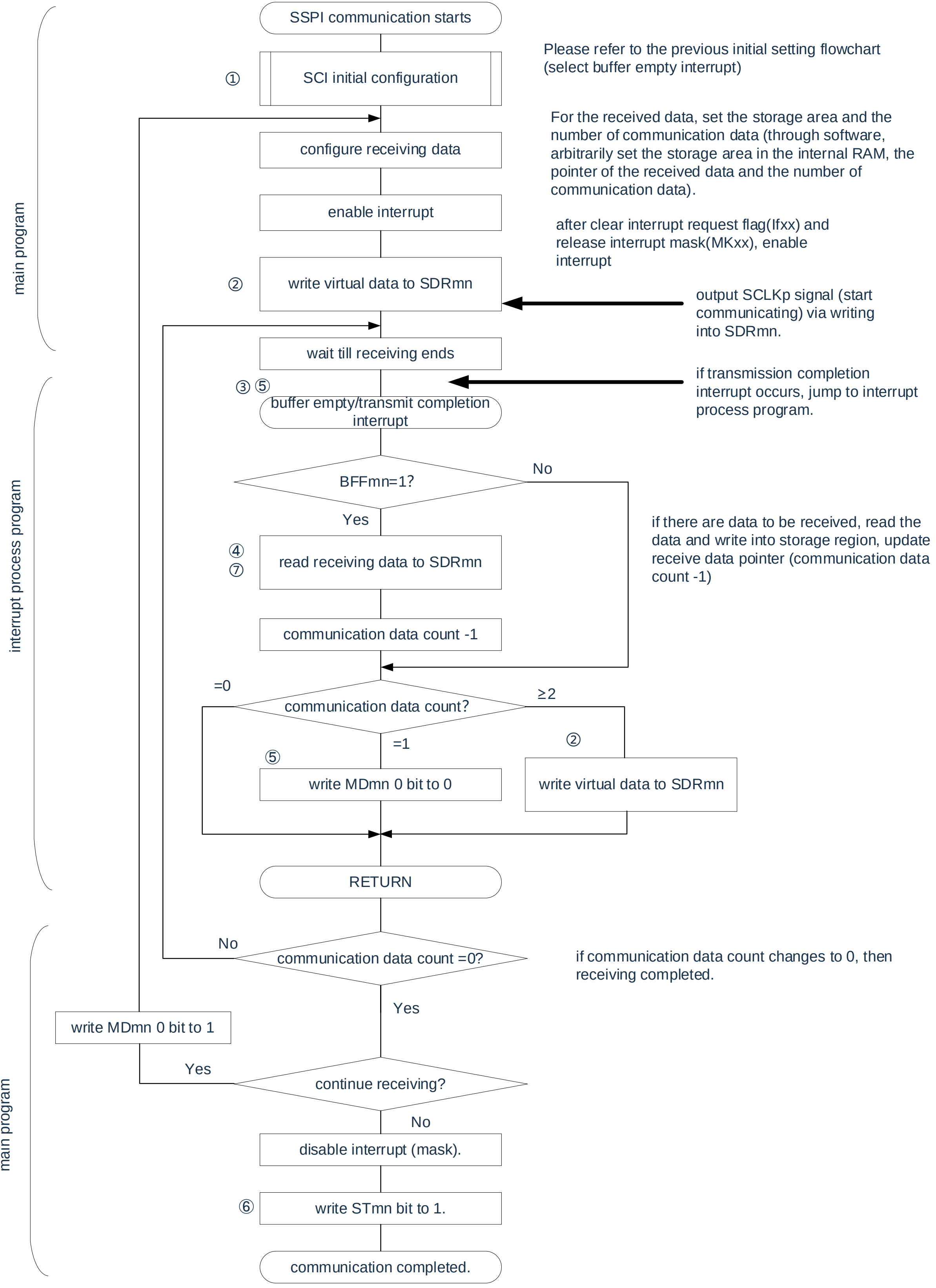


Notice: The MDmn0 bits can be overridden even during a run. However, in order to be able to catch that end interruption of the transmission of the last receive data, it is necessary to override before the last receive is started.

Remark: 1. ① to ⑧ in the figure corresponds to ① to ⑧ in Figure 15-36 Flow Chart.

2. m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI Number (p=00, 01, 10, 11, 20, 21)

Figure 15-36 Flowchart of master reception (continuous receive mode)


main program
SSPI communication starts
①
SCI initial configuration
Please refer to the previous initial setting flowchart (select buffer empty interrupt)
configure receiving data
For the received data, set the storage area and the number of communication data (through software, arbitrarily set the storage area in the internal RAM, the pointer of the received data and the number of communication data).
enable interrupt
after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt
②
write virtual data to SDRmn
output SCLKp signal (start communicating) via writing into SDRmn.
wait till receiving ends
if transmission completion interrupt occurs, jump to interrupt process program.
interrupt process program
③ ⑤
buffer empty/transmit completion interrupt
BFFmn=1?
No
Yes
④
⑦
read receiving data to SDRmn
if there are data to be received, read the data and write into storage region, update receive data pointer (communication data count -1)
communication data count -1
=0
communication data count?
≥2
=1
⑤
write MDmn 0 bit to 0
②
write virtual data to SDRmn
RETURN
main program
No
communication data count =0?
if communication data count changes to 0, then receiving completed.
Yes
write MDmn 0 bit to 1
Yes
continue receiving?
No
disable interrupt (mask).
⑥
write STmn bit to 1.
communication completed.

Remark: ① to ⑧ in the figure correspond to ① to ⑧ in “Figure 15-35 Timing diagram of master reception (continuous receive mode)” .

15.5.3 Master transmission and reception

The main control of sending and receiving refers to this product output transmission clock and other devices to send and receive data running.

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, SDI00, SDO00	SCLK01, SDI01, SDO01	SCLK10, SDI10, SDO10	SCLK11, SDI11, SDO11	SCLK20, SDI20, SDO20	SCLK21, SDI21, SDO21
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21
	Interrupt at that end of the transfer may be selecte (single transfer mode) or buffer air-discontinuity (continuous transfer mode).					
Error detection flag	Only the overflow error detection flag (OVFmn).					
Length of transmit data	7 ~ 16 bits					
Transfer rate [Note]	Max.f_{CLK}/2[Hz] Min.$f_{CLK}/(2\times2^{11}\times128)$ [Hz] f_{CLK}: 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: forward • CKPmn=1: inversion					
Data orientation	MSB First or LSB First					

Note: It must be used within the scope of the peripheral functional characteristics (refer to the data sheet) that meet this condition and satisfy the electrical characteristics.

Remark: m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI Number (p=00, 01, 10, 11, 20, 21)

(1) Register settings

Figure 15-37 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20,SSPI21)
Example of register setting content when master transmits and receives

(a) serial mode register mn (SMRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn 0/1	CCSmn 0	0	0	0	0	0	STSmn 0	0	SISmn0 0	1	0	0	MDmn2 0	MDmn1 0	MDmn0 0/1

channel n operational clock (fMCK)
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

interrupt source of channel n
0: Transmit completion interrupt
1: Buffer empty interrupt

(b) serial communication operation configuration registermn mn (SCRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn 1	RXEmn 1	DAPmn 0/1	CKPmn 0/1	0	EOCmn 0	PTCmn1 0	PTCmn0 0	DIRmn 0/1	0	SLCmn1 0	SLCmn0 0	DLSmn3 0/1	DLSmn2 0/1	DLSmn1 0/1	DLSmn0 0/1

data and clock phase selection (details refer to "Registers controlling universal serial communication unit)

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

Setting of data length
DLSmn3~0: 7-bit~16-bit
data length selection

(c) serial data registermn mn (SDRmn)

(1) When operation stops (SEmn=0)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration (operation clock (fmck) scaling configuration)							0	0	0	0	0	0	0	0	0

(2) During operation（SEmn=1）（lower 8 bits: SDRmnL）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	transmit/receive data registers															
									SDRmnL							

(d) serial output register m(SOm) ......Only configure bit of target channel

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm	0	0	0	0	CKOm3 0/1	CKOm2 0/1	CKOm1 0/1	CKOm0 0/1	0	0	0	0	SOm3 0/1	SOm2 0/1	SOm1 0/1	SOm0 0/1

when clock phase is "positive phase" (CKPmn of SCRmn register as 0), "1" means starting communication; when clock phase is "inverted phase" (CKPmn=1), "0" means starting communication.

(e) serial output enable register m (SOEm)….Only set bit of target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	SOEm3 0/1	SOEm2 0/1	SOEm1 0/1	SOEm0 0/1

(f) serial channel start registerm (SSm)….only set bit of target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	SSm3 0/1	SSm2 0/1	SSm1 0/1	SSm0 0/1

Note 1. m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI Number (p=00,01,10,11,20,21) mn=00~03,10~11

2. □ : Fixed in SSPI master transmit and receive mode. ▨ : Cannot set (set initial value).
0/1: The "0" or "1" is set according to the user.

(2) Procedure

Figure 15-38 Initial set-up steps for master transmission and reception

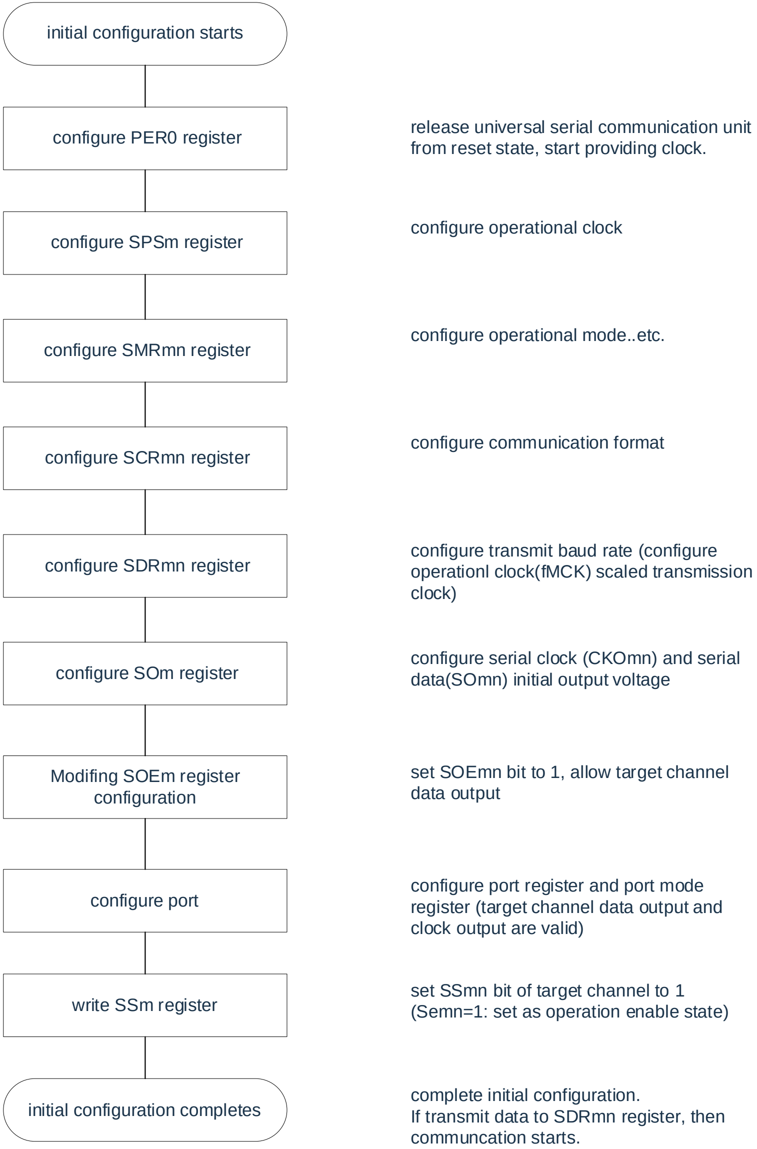
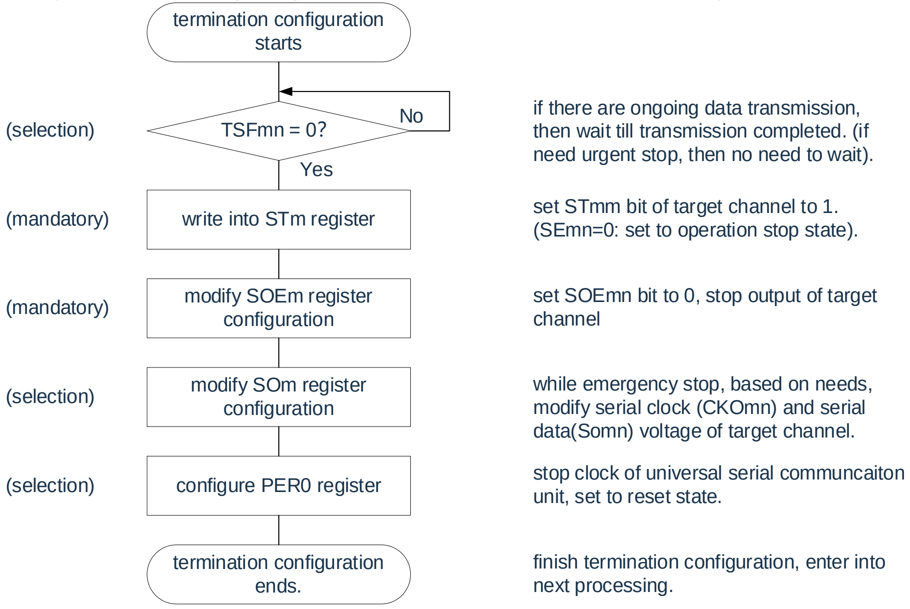

initial configuration starts
configure PER0 register
release universal serial communication unit from reset state, start providing clock.
configure SPSm register
configure operational clock
configure SMRmn register
configure operational mode..etc.
configure SCRmn register
configure communication format
configure SDRmn register
configure transmit baud rate (configure operationl clock(fMCK) scaled transmission clock)
configure SOm register
configure serial clock (CKOmn) and serial data(SOmn) initial output voltage
Modifing SOEm register configuration
set SOEmn bit to 1, allow target channel data output
configure port
configure port register and port mode register (target channel data output and clock output are valid)
write SSm register
set SSmn bit of target channel to 1 (Semn=1: set as operation enable state)
initial configuration completes
complete initial configuration.
If transmit data to SDRmn register, then communcation starts.

Figure 15-39 Stop steps for master transmission and reception


termination configuration starts
(selection)
TSFmn = 0?
No
Yes
(mandatory)
write into STm register
(mandatory)
modify SOEm register configuration
(selection)
modify SOm register configuration
(selection)
configure PER0 register
termination configuration ends.

if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).

set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).

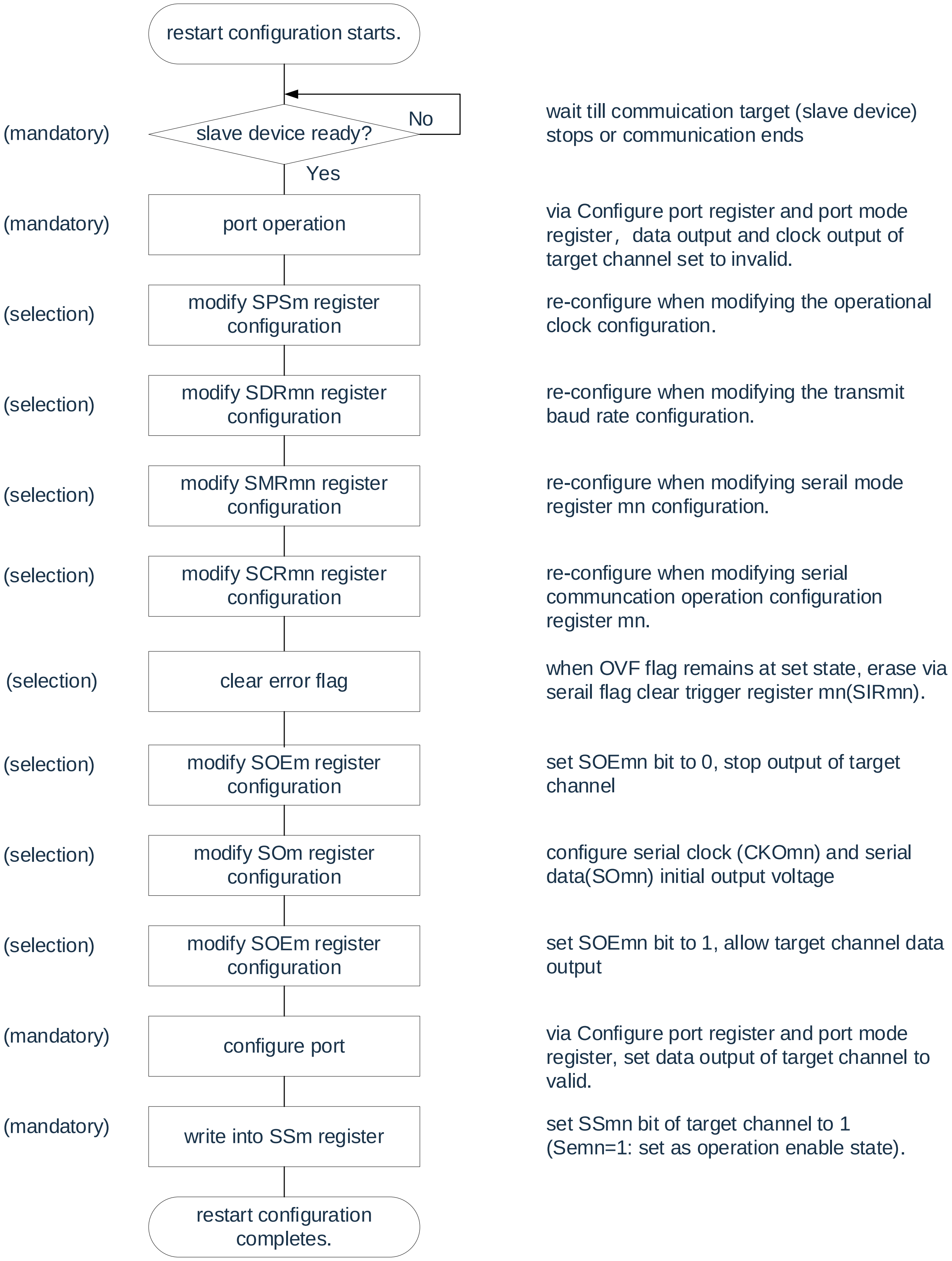
set SOEmn bit to 0, stop output of target channel

while emergency stop, based on needs, modify serial clock (CKOmn) and serial data(Somn) voltage of target channel.

stop clock of universal serial communcaiton unit, set to reset state.

finish termination configuration, enter into next processing.

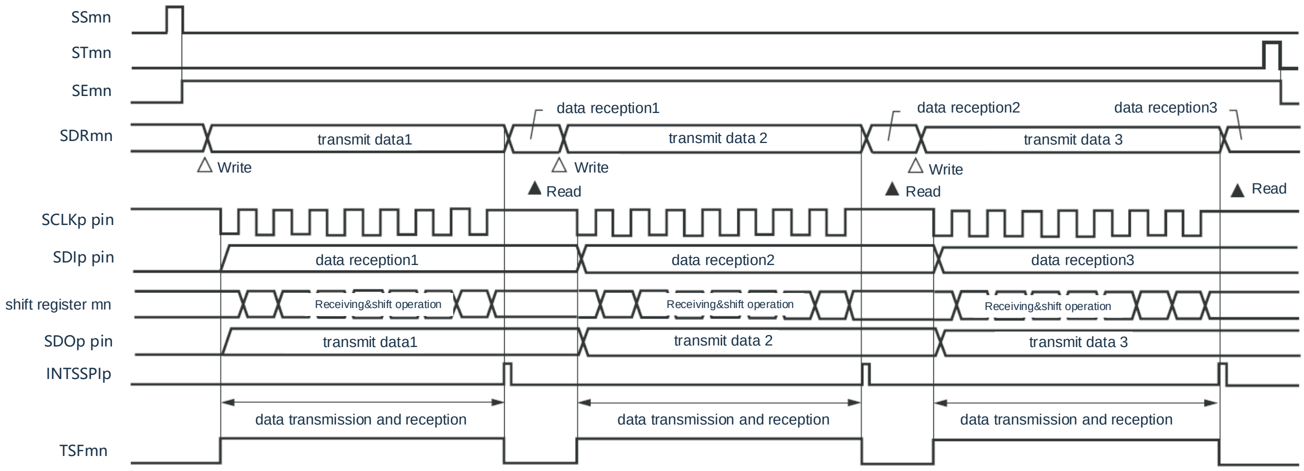
Figure 15-40 Restart set-up steps for master transmission and reception


restart configuration starts.
slave device ready?
No
Yes
port operation
modify SPSm register configuration
modify SDRmn register configuration
modify SMRmn register configuration
modify SCRmn register configuration
clear error flag
modify SOEm register configuration
modify SOm register configuration
modify SOEm register configuration
configure port
write into SSm register
restart configuration completes.

(mandatory)	slave device ready?	wait till commuication target (slave device) stops or communication ends
(mandatory)	port operation	via Configure port register and port mode register， data output and clock output of target channel set to invalid.
(selection)	modify SPSm register configuration	re-configure when modifying the operational clock configuration.
(selection)	modify SDRmn register configuration	re-configure when modifying the transmit baud rate configuration.
(selection)	modify SMRmn register configuration	re-configure when modifying serail mode register mn configuration.
(selection)	modify SCRmn register configuration	re-configure when modifying serial communcation operation configuration register mn.
(selection)	clear error flag	when OVF flag remains at set state, erase via serail flag clear trigger register mn(SIRmn).
(selection)	modify SOEm register configuration	set SOEmn bit to 0, stop output of target channel
(selection)	modify SOm register configuration	configure serial clock (CKOmn) and serial data(SOmn) initial output voltage
(selection)	modify SOEm register configuration	set SOEmn bit to 1, allow target channel data output
(mandatory)	configure port	via Configure port register and port mode register, set data output of target channel to valid.
(mandatory)	write into SSm register	set SSmn bit of target channel to 1 (Semn=1: set as operation enable state).

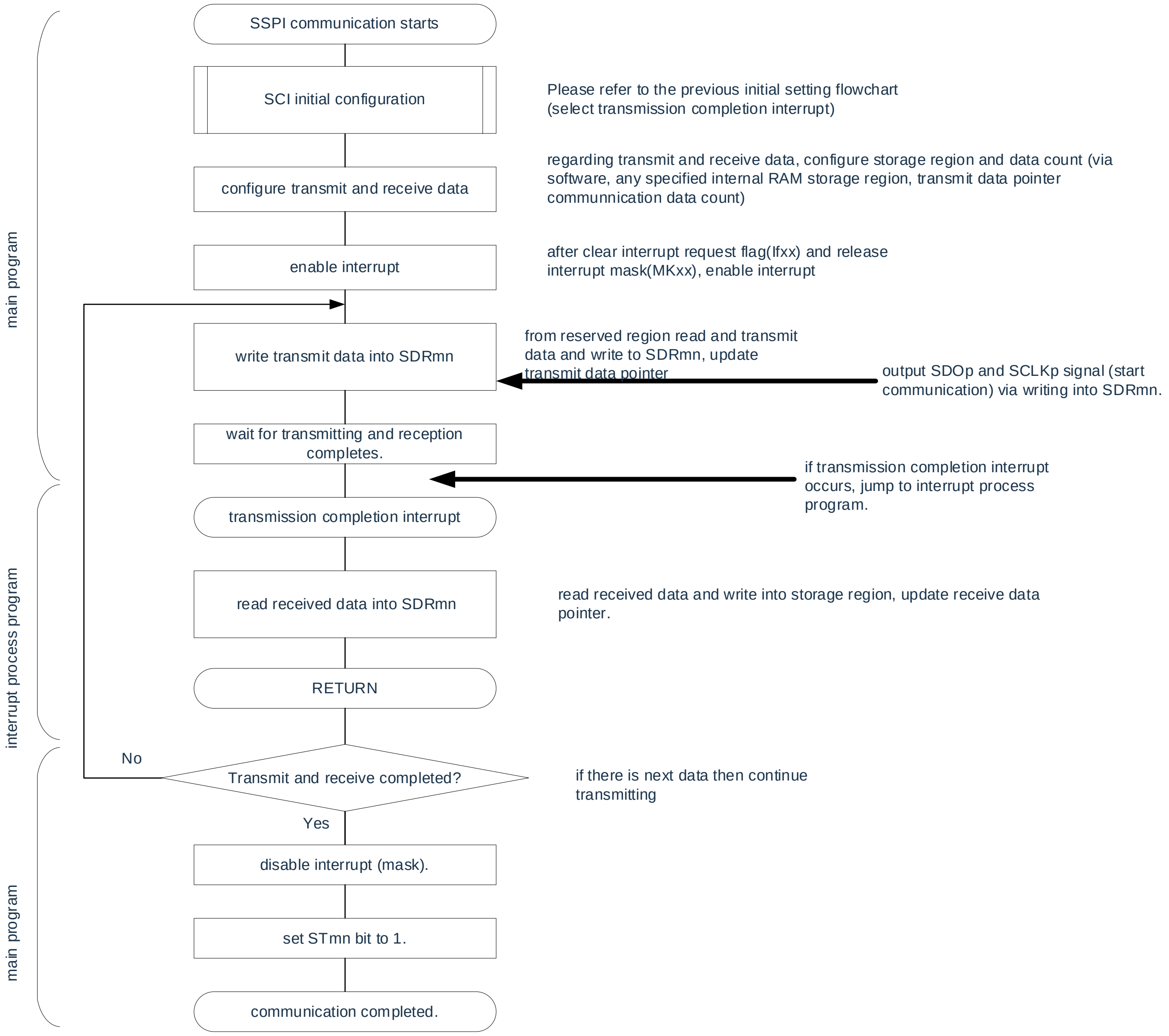
(3) Process flow (single transmit and receive mode)

Figure 15-41 Timing diagram for master send and receive (single send and receive mode)
(Type 1: DAPmn=0, CKPmn=0)



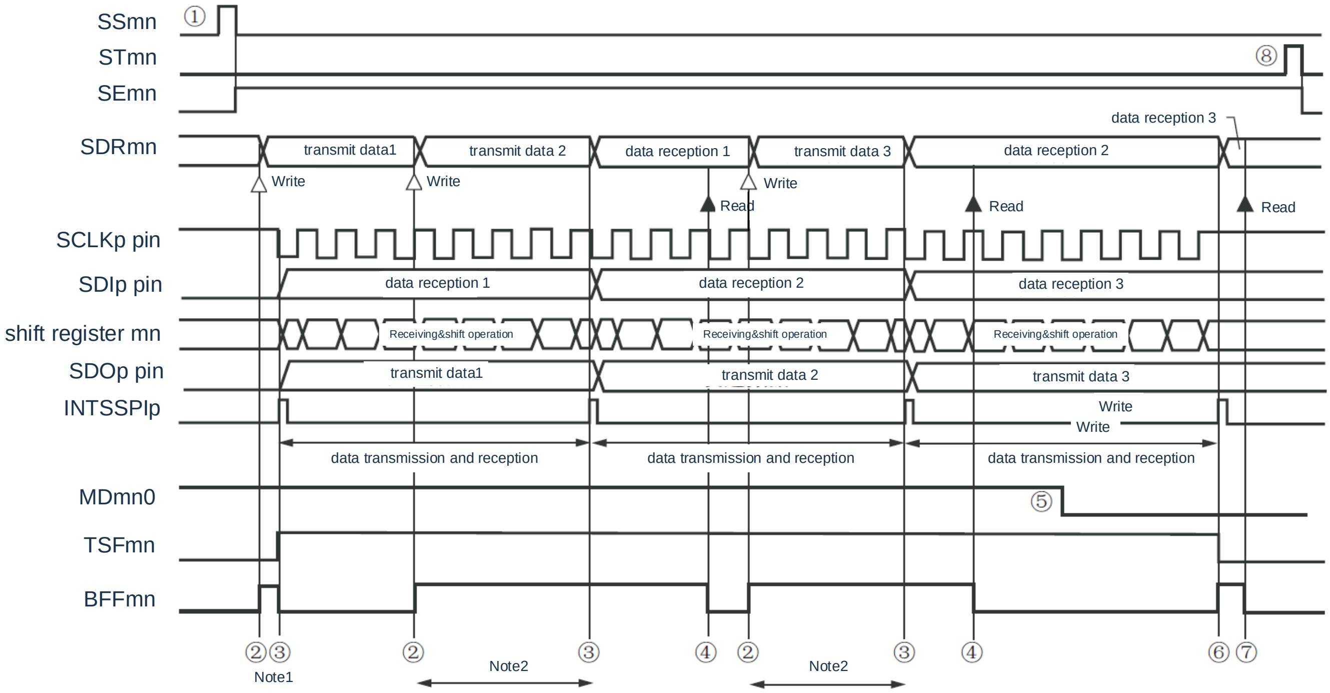
Remark: m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI Number (p=00,01,10,11,20,21)

Figure 15-42 Flow chart of master transmission and reception (single transmit and receive mode)


main program
SSPI communication starts
SCI initial configuration
Please refer to the previous initial setting flowchart (select transmission completion interrupt)
configure transmit and receive data
regarding transmit and receive data, configure storage region and data count (via software, any specified internal RAM storage region, transmit data pointer communnication data count)
enable interrupt
after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt
write transmit data into SDRmn
from reserved region read and transmit data and write to SDRmn, update transmit data pointer
output SDOp and SCLKp signal (start communication) via writing into SDRmn.
wait for transmitting and reception completes.
if transmission completion interrupt occurs, jump to interrupt process program.
interrupt process program
transmission completion interrupt
read received data into SDRmn
read received data and write into storage region, update receive data pointer.
RETURN
No
Transmit and receive completed?
if there is next data then continue transmitting
Yes
main program
disable interrupt (mask).
set STmn bit to 1.
communication completed.

(4) Processing flow (Continuous transmit and receive mode)

Figure 15-43 Timing diagram for master transmission and reception (continuous transmit and receive mode) (Type 1:DAPmn=0, CKPmn=0)



Note 1. Rewrite the transmission data 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)).

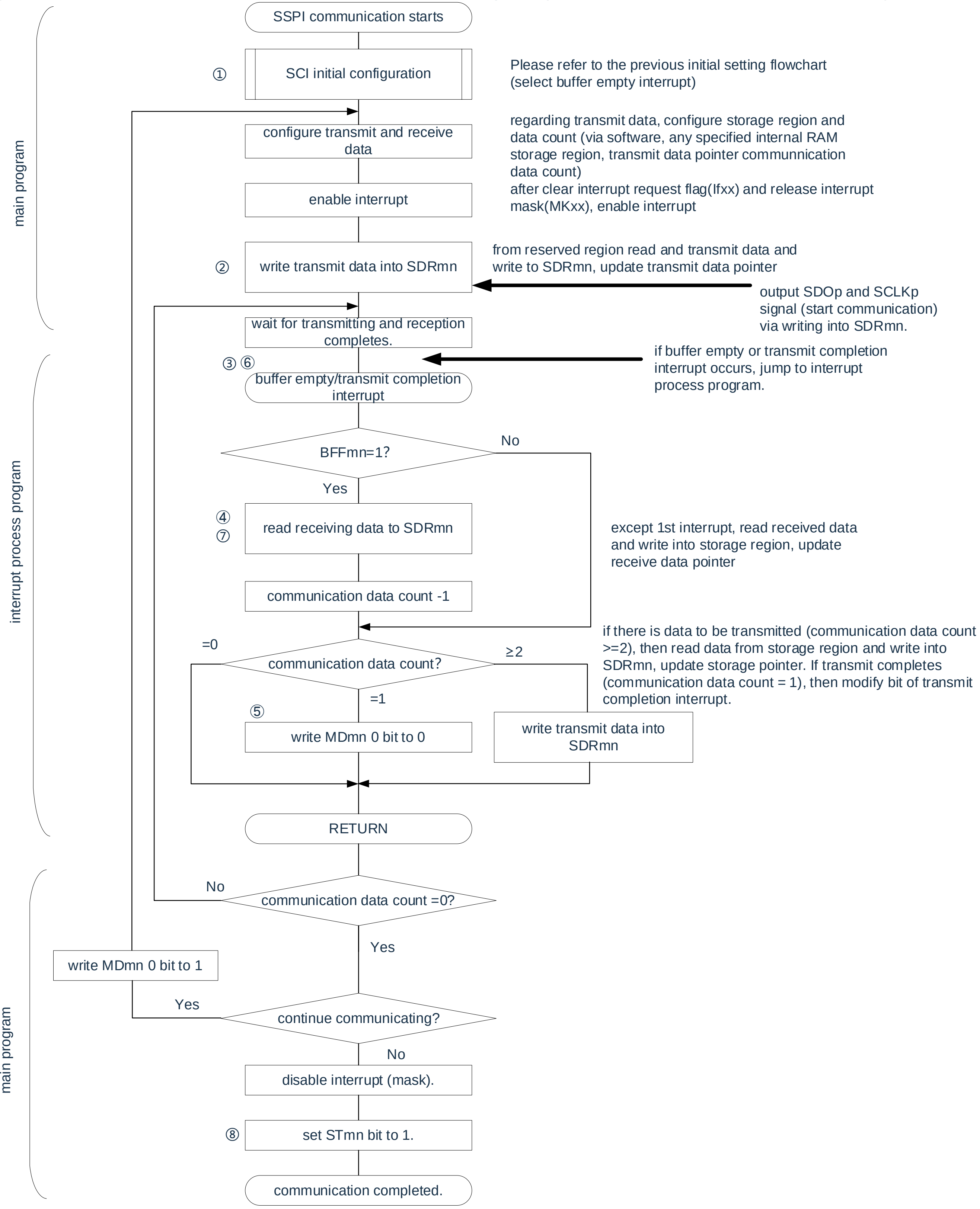
2. If the SDRmn register is read during this time, the transmit data can be read. At this time, the transmit operation is not affected.

Notice The MDmn0 bit of the serial mode register mn(SMRmn) can be rewritten even in operation. However, in order to be able to catch the end of the transmission of the last transmitted data interrupt, it is necessary to override before starting the last transmission.

Remark 1. ① to ⑧ in the figure corresponds to ① to ⑧ in Figure 15-44 Flow chart for master transmission and reception (continuous transmit and receive mode).

2. m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI Number (p=00, 01, 10, 11, 20, 21)

Figure 15-44 Flow chart for master transmission and reception (continuous transmit and receive mode)


main program
SSPI communication starts
①
SCI initial configuration
Please refer to the previous initial setting flowchart (select buffer empty interrupt)
configure transmit and receive data
regarding transmit data, configure storage region and data count (via software, any specified internal RAM storage region, transmit data pointer connmunication data count)
after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt
enable interrupt
②
write transmit data into SDRmn
from reserved region read and transmit data and write to SDRmn, update transmit data pointer
output SDOp and SCLKp signal (start communication) via writing into SDRmn.
wait for transmitting and reception completes.
interrupt process program
③ ⑥
buffer empty/transmit completion interrupt
if buffer empty or transmit completion interrupt occurs, jump to interrupt process program.
BFFmn=1?
No
Yes
④
⑦
read receiving data to SDRmn
except 1st interrupt, read received data and write into storage region, update receive data pointer
communication data count -1
=0
communication data count?
≥2
=1
if there is data to be transmitted (communication data count >=2), then read data from storage region and write into SDRmn, update storage pointer. If transmit completes (communication data count = 1), then modify bit of transmit completion interrupt.
⑤
write MDmn 0 bit to 0
write transmit data into SDRmn
RETURN
main program
No
communication data count =0?
Yes
write MDmn 0 bit to 1
Yes
continue communicating?
No
disable interrupt (mask).
⑧
set STmn bit to 1.
communication completed.

Remark ① to ⑧ in the figure correspond to ① to ⑧ in “Figure 15-43 Timing diagram for master transmission and reception (continuous transmit and receive mode)”.

15.5.4 Slave transmission

Slave sending is the operation of the BAT32G157 microcontroller sending data to other devices in a state where a transfer clock is input 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	SCLK01, SDO01	SCLK10, SDO10	SCLK11, SDO11	SCLK20, SDO20	SCLK21, SDO21
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21
	Interrupt at that end of the transfer may be selecte (single transfer mode) or buffer air-discontinuity (continuous transfer mode).					
Error detection flag	Only the overflow error detection flag (OVFmn).					
Length of transmit data	7 ~ 16 bits					
Transfer rate	Max.f_{MCK}/6[Hz][Note 1,2]					
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: forward • CKPmn=1: inverted					
Data orientation	MSB First or LSB First					

Note: 1. The maximum transfer rate is f_{MCK}/6[Hz]since the external serial clock input by SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, SCLK21 pins is used internally.

2. It must be used within the scope of the peripheral functional characteristics (refer to the data sheet) that meet this condition and satisfy the electrical characteristics.

Note 1.f_{MCK}: Operating clock frequency of the object channel

2.m: Unit number (m=0,1,2) n: Channel number (n=0,1)

(1) Register settings

Figure 15-45 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20,SSPI21)
Examples of register setting contents during slave transmission

(a) serial mode register mn (SMRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn	CCSmn						STSmn		SISmn0				MDmn2	MDmn1	MDmn0
	0/1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0/1

channel n operational clock (fMCK)
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

interrupt source of channel n
0: Transmit completion interrupt
1: Buffer empty interrupt

(b) serial communication operation configuration register mn (SCRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn	RXEmn	DAPmn	CKPmn		EOCmn	PTCmn1	PTCmn0	DIRmn		SLCmn1	SLCmn0	DLSmn3	DLSmn2	DLSmn1	DLSmn0
	1	0	0/1	0/1	0	0	0	0	0/1	0	0	0	0/1	0/1	0/1	0/1

data and clock phase selection (details refer to registers controlling universal serial communication unit)

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

Setting of data length
DLSmn3~0: 7-bit~16-bit data length selection

(c) serial data regsiter mn (SDRmn)

(1) When operation stops (SEmn=0)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration（0000000B)							0	0	0	0	0	0	0	0	0

(2) 运行期间（SEmn=1）（低8位：SDRmnL）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	transmit data register															

$\underbrace{\text{bits 7–0}}_{\text{SDRmnL}}$

SDRmnL

(d) serial output register m(SOm) ......Only configure bit of target channel

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm					CKOm3	CKOm2	CKOm1	CKOm0					SOm3	SOm2	SOm1	SOm0
	0	0	0	0	×	×	×	×	0	0	0	0	0/1	0/1	0/1	0/1

(e) serial output enable registerm (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	0/1	0/1

(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	0/1	0/1	0/1

Note 1.m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI Number (p=00,01,10,11,20,21)

2. □ : Fixed in SSPI slave send mode. ▨ : Cannot set (set initial value).

×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).

0/1: The "0" or "1" is set according to the user.

(2) Procedure

Figure 15-46 Initial set-up steps for slave transmission

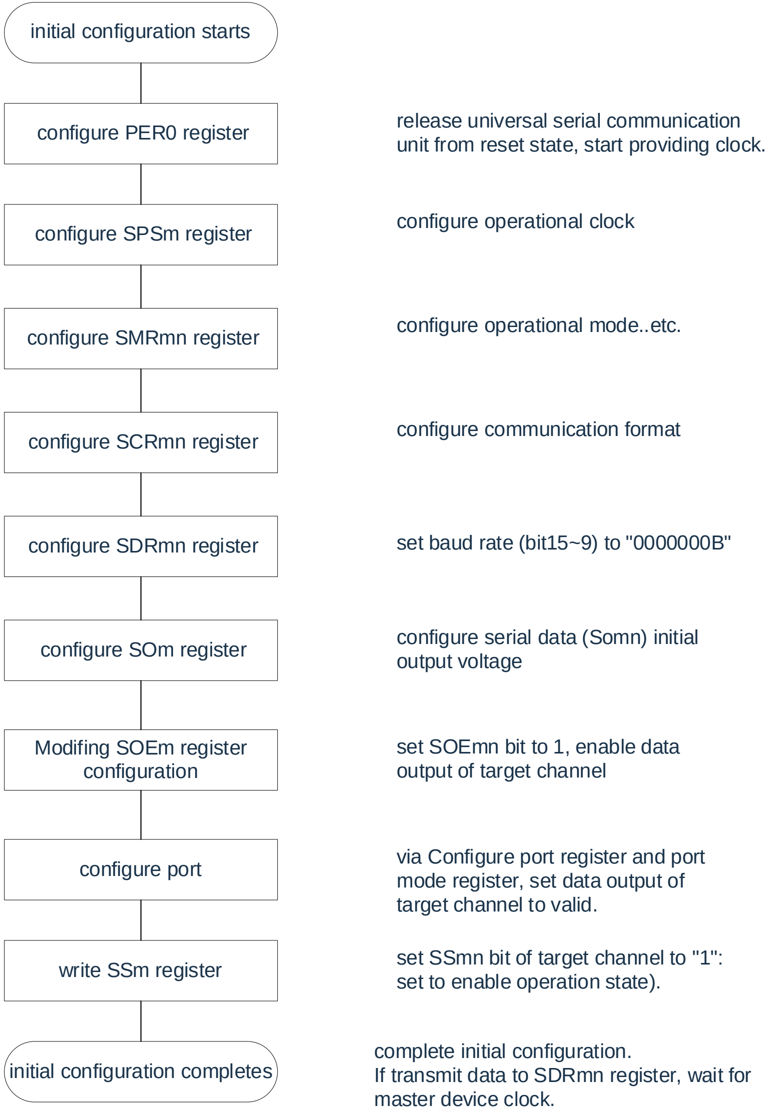

initial configuration starts
configure PER0 register
release universal serial communication unit from reset state, start providing clock.
configure SPSm register
configure operational clock
configure SMRmn register
configure operational mode..etc.
configure SCRmn register
configure communication format
configure SDRmn register
set baud rate (bit15~9) to "0000000B"
configure SOm register
configure serial data (Somn) initial output voltage
Modifing SOEm register configuration
set SOEmn bit to 1, enable data output of target channel
configure port
via Configure port register and port mode register, set data output of target channel to valid.
write SSm register
set SSmn bit of target channel to "1": set to enable operation state).
initial configuration completes
complete initial configuration.
If transmit data to SDRmn register, wait for master device clock.

Figure 15-47 Stop steps for slave transmission

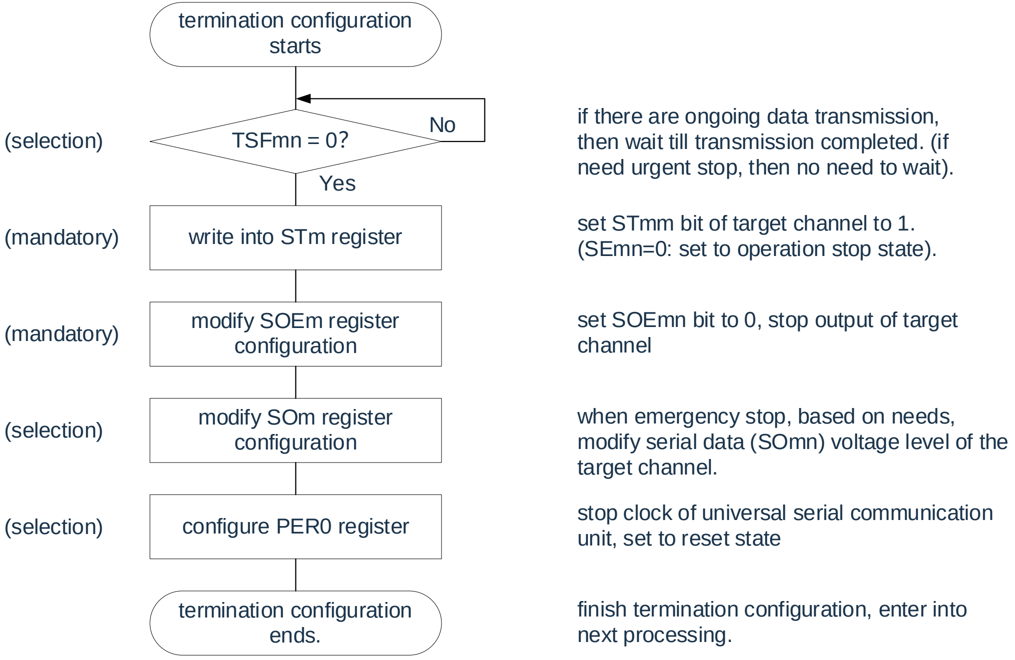
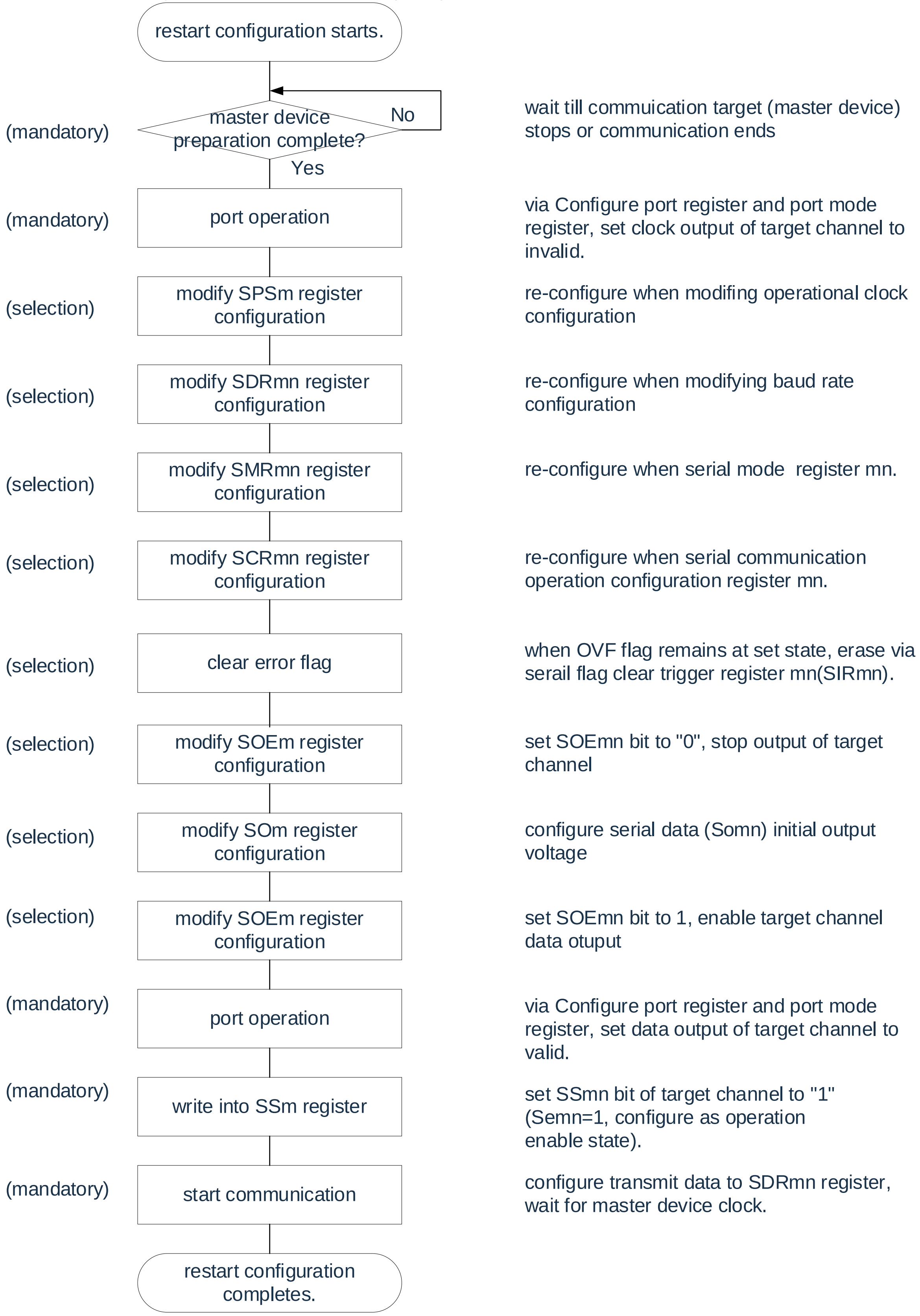

termination configuration starts
(selection)
TSFmn = 0?
No
Yes
if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).
(mandatory)
write into STm register
set STmm bit of target channel to 1.
(SEmn=0: set to operation stop state).
(mandatory)
modify SOEm register configuration
set SOEmn bit to 0, stop output of target channel
(selection)
modify SOm register configuration
when emergency stop, based on needs, modify serial data (SOmn) voltage level of the target channel.
(selection)
configure PER0 register
stop clock of universal serial communication unit, set to reset state
termination configuration ends.
finish termination configuration, enter into next processing.

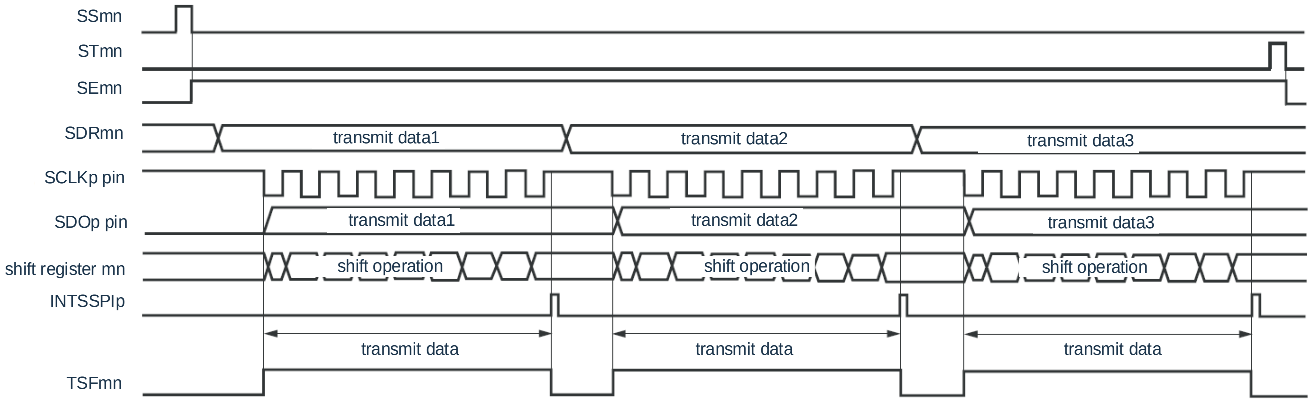
Figure 15-48 Restart set-up steps for slave transmission


restart configuration starts.
(mandatory)
master device preparation complete?
No
Yes
wait till commuication target (master device) stops or communication ends
(mandatory)
port operation
via Configure port register and port mode register, set clock output of target channel to invalid.
(selection)
modify SPSm register configuration
re-configure when modifing operational clock configuration
(selection)
modify SDRmn register configuration
re-configure when modifying baud rate configuration
(selection)
modify SMRmn register configuration
re-configure when serial mode register mn.
(selection)
modify SCRmn register configuration
re-configure when serial communication operation configuration register mn.
(selection)
clear error flag
when OVF flag remains at set state, erase via serail flag clear trigger register mn(SIRmn).
(selection)
modify SOEm register configuration
set SOEmn bit to "0", stop output of target channel
(selection)
modify SOm register configuration
configure serial data (Somn) initial output voltage
(selection)
modify SOEm register configuration
set SOEmn bit to 1, enable target channel data otuput
(mandatory)
port operation
via Configure port register and port mode register, set data output of target channel to valid.
(mandatory)
write into SSm register
set SSmn bit of target channel to "1" (Semn=1, configure as operation enable state).
(mandatory)
start communication
configure transmit data to SDRmn register, wait for master device clock.
restart configuration completes.

Note: If you override PER0 in the abort setting to stop providing the clock, you must wait until the communication object (the master device) is stopped or the communication is over for the initial set-up instead of restarting.

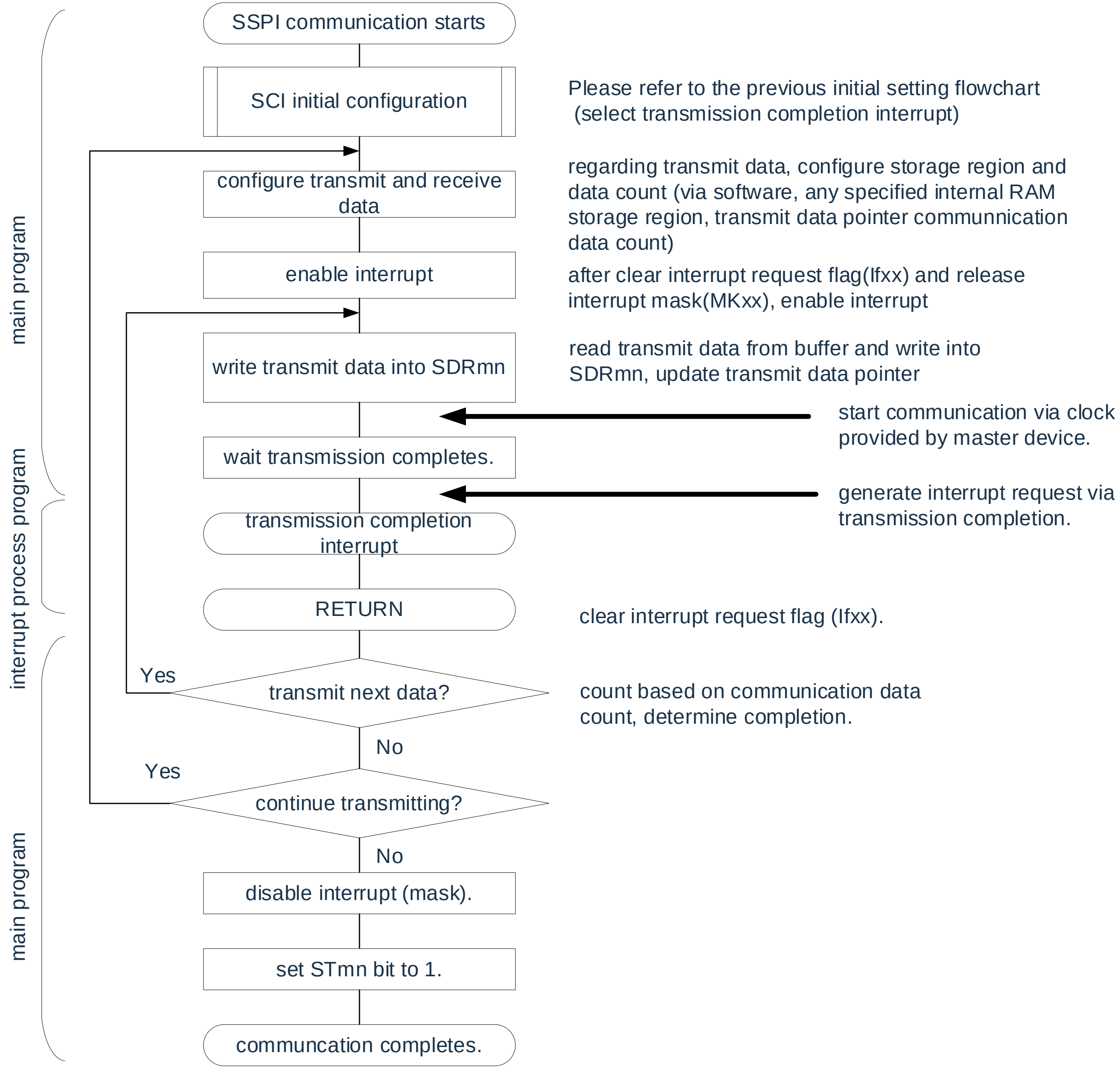
(3) Process flow (single transmit mode)

Figure 15-49 Timing of slave transmission (single transmit mode) (Type 1: DAPmn=0, CKPmn=0)



Remark: m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI Number (p=00, 01, 10, 11, 20, 21)

Figure 15-50 Flowchart of slave transmission (single transmit mode)


main program
interrupt process program
main program
SSPI communication starts
SCI initial configuration
configure transmit and receive data
enable interrupt
write transmit data into SDRmn
wait transmission completes.
transmission completion interrupt
RETURN
transmit next data?
Yes
No
continue transmitting?
Yes
No
disable interrupt (mask).
set STmn bit to 1.
communcation completes.

Please refer to the previous initial setting flowchart (select transmission completion interrupt)

regarding transmit data, configure storage region and data count (via software, any specified internal RAM storage region, transmit data pointer communnication data count)

after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt

read transmit data from buffer and write into SDRmn, update transmit data pointer

start communication via clock provided by master device.

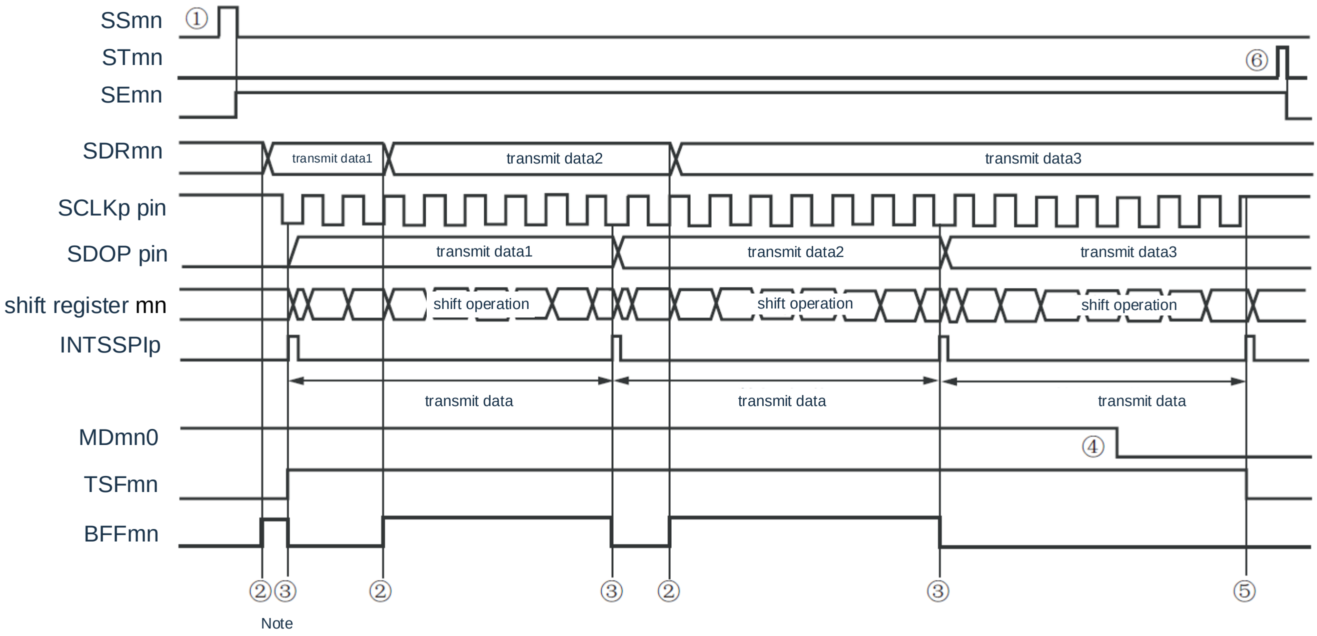
generate interrupt request via transmission completion.

clear interrupt request flag (Ifxx).

count based on communication data count, determine completion.

(4) Process flow (continuous transmit mode)

Figure 15-51 Timing of slave transmission (continuous transmit mode)
(Type 1:DAPmn=0, CKPmn=0)

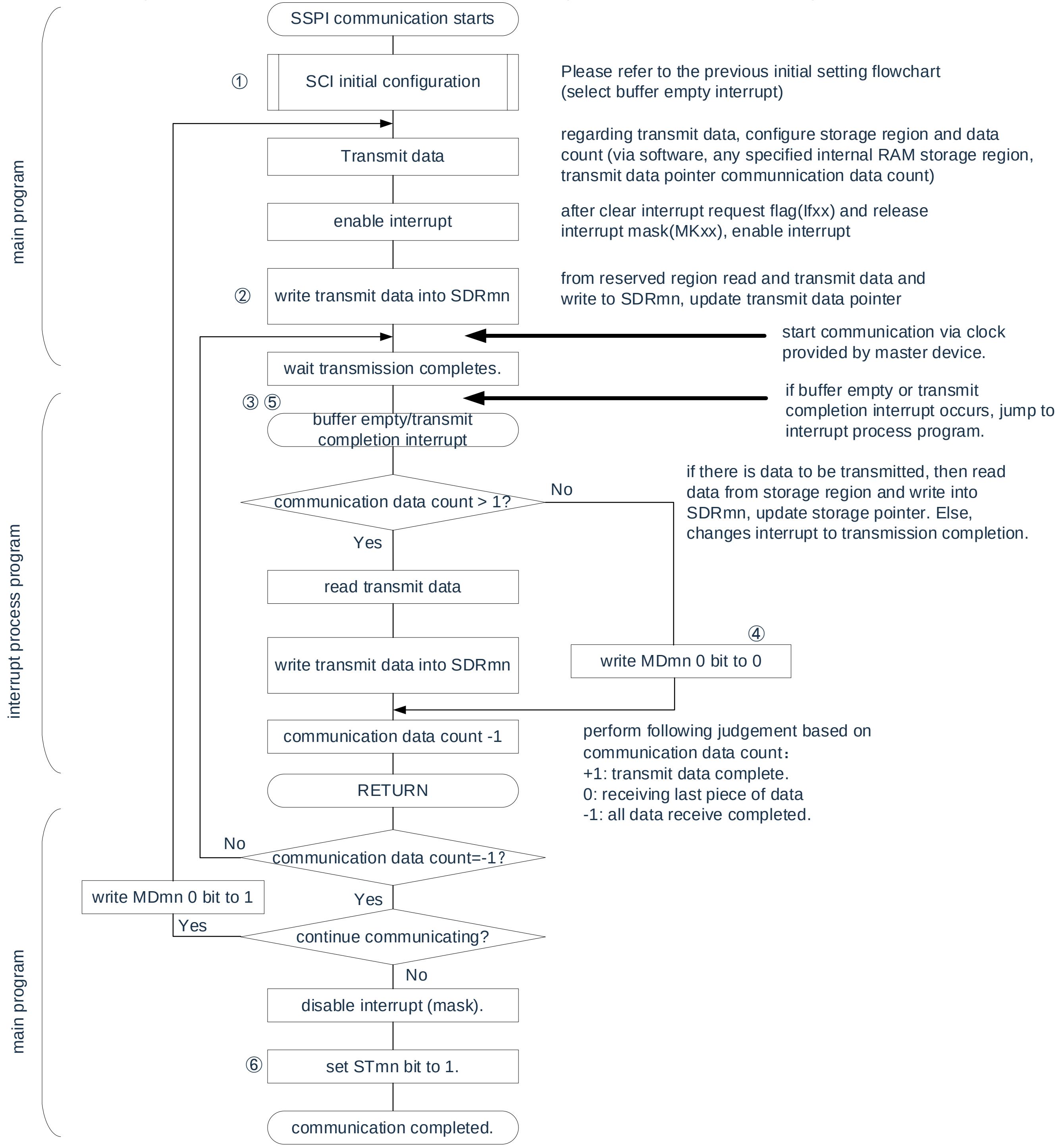


Note: The transmission data is rewritten if the BFFmn bit of the serial status register mn (SSRmn) is "1" (SDRmn) when the valid data is stored in the serial data register mn (SDRmn).

Notice: The MDmn0 bit of the serial mode register mn(SMRmn) can be rewritten even in operation. However, you must override before you start transferring the last bit.

Remark: m: Unit number (m=0, 1, 2) n: Channel number (n=0, 1) p: SSPI Number (p=00, 01, 10, 11, 20, 21)

Figure 15-52 Flowchart for slave transmission (continuous transmit mode)


main program
SSPI communication starts
①
SCI initial configuration
Please refer to the previous initial setting flowchart (select buffer empty interrupt)
Transmit data
regarding transmit data, configure storage region and data count (via software, any specified internal RAM storage region, transmit data pointer communnication data count)
enable interrupt
after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt
②
write transmit data into SDRmn
from reserved region read and transmit data and write to SDRmn, update transmit data pointer
start communication via clock provided by master device.
wait transmission completes.
if buffer empty or transmit completion interrupt occurs, jump to interrupt process program.
interrupt process program
③ ⑤
buffer empty/transmit completion interrupt
communication data count > 1?
No
if there is data to be transmitted, then read data from storage region and write into SDRmn, update storage pointer. Else, changes interrupt to transmission completion.
Yes
read transmit data
write transmit data into SDRmn
④
write MDmn 0 bit to 0
communication data count -1
perform following judgement based on communication data count：
+1: transmit data complete.
0: receiving last piece of data
-1: all data receive completed.
RETURN
main program
No
communication data count=-1?
write MDmn 0 bit to 1
Yes
Yes
continue communicating?
No
disable interrupt (mask).
⑥
set STmn bit to 1.
communication completed.

Remark: ① to ⑥ in the figure corresponds to ① to ⑥ in “Figure 15-51 Timing of slave transmission (continuous transmit mode)”.

15.5.5 Slave reception

A slave reception is the operation of this product to receive data from other devices in the state of inputting transmission clock 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, SDI00	SCLK01, SDI01	SCLK10, SDI10	SCLK11, SDI11	SCLK20, SDI20	SCLK21, SDI21
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21
	Interrupt at that end of the transfer only (Disable from setting buffer null interrupt).					
Error detection flag	Only the overflow error detection flag (OVFmn).					
Length of transmit data	7 ~ 16 bits					
Transfer rate	Max.f_{MCK}/6[Hz]$^{Note\ 1,2}$					
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: forward • CKPmn=1: inverted					
Data orientation	MSB First or LSB First					

Note: 1. The maximum transfer rate is f_{MCK}/6[Hz]since the external serial clock input by SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, SCLK21 pins is used internally.

2. It must be used within the scope of the peripheral functional characteristics (refer to the data sheet) that meet this condition and satisfy the electrical characteristics.

Remark: 1. f_{MCK}: Operating clock frequency of the object channel

2. m: Unit number (m=0,1,2) n: Channel number (n=0,1)

(1) Register settings

Figure 15-53 3-wire Serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20,SSPI21) Example of register setting content during slave reception

(a) serial mode register mn (SMRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn 0/1	CCSmn 1	0	0	0	0	0	STSmn 0	0	SISmn0 0	1	0	0	MDmn2 0	MDmn1 0	MDmn0 0

channel n operational clock (fMCK)
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

Interrupt source for channel n
0: End of transmission interrupt
1: Buffer air interrupt

(b) serial communication operation configuration register mn (SCRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn 0	RXEmn 1	DAPmn 0/1	CKPmn 0/1	0	EOCmn 0	PTCmn1 0	PTCmn0 0	DIRmn 0/1	0	SLCmn1 0	SLCmn0 0	DLSmn3 0/1	DLSmn2 0/1	DLSmn1 0/1	DLSmn0 0/1

data and clock phase selection (details refer to "Registers controlling universal communication unit)

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

Setting of data length
DLSmn3~0: 7-bit~16-bit data length selection

(c) serial data register mn (SDRmn)

(1) When operation stops (SEmn=0)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration (0000000B)							0	0	0	0	0	0	0	0	0

(2) During operation（SEmn=1）（lower 8 bits: SDRmnL）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	receive data registers															

SDRmnL

(d) 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 ×

(e) serial output enable register m (SOEm)….not used in this mode.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	SOEm3 ×	SOEm2 ×	SOEm1 ×	SOEm0 ×

(f) serial channel start registerm (SSm)….only set bit of target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	SSm3 0/1	SSm2 0/1	SSm1 0/1	SSm0 0/1

Note 1. m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI number (p=00,01,10,11,20,21)

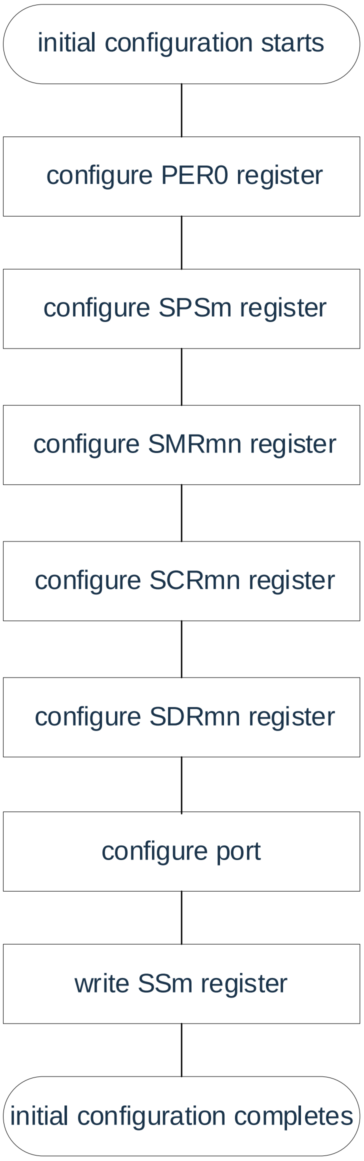
2. □ : Set in slave receive mode for fixed. : Cannot set (set initial value).

×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).

0/1: The "0" or "1" is set according to the user.

2) Procedure

Figure 15-54 Initial set-up steps for slave reception



release universal serial communication unit from reset state, start providing clock.

configure operational clock

configure operational mode..etc.

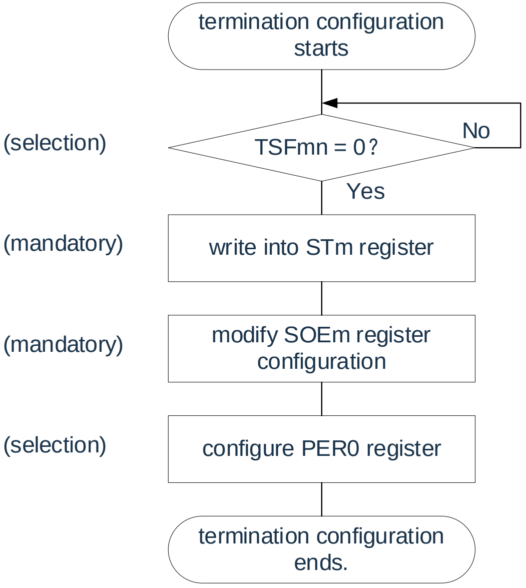
configure communication format

set baud rate (bit15~9) to "0000000B"

via Configure port register and port mode register, data input and clock input of target channel is set to valid.

SSmn bit of target channel set to "1" (Semn=1: set to operation enable state). Wait for master device clock.

Figure 15-55 Stop steps for slave reception



if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).

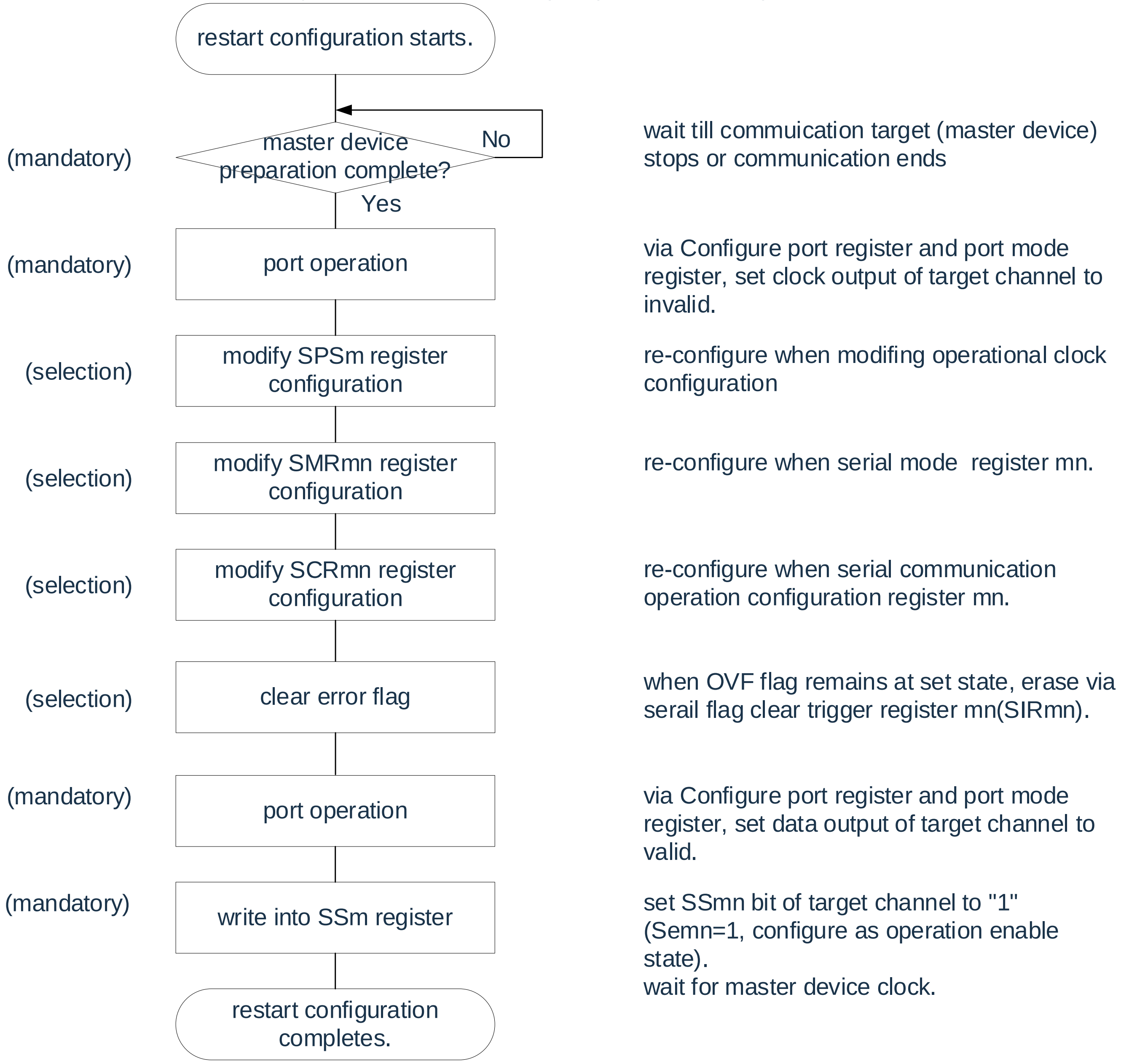
set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).

set SOEmn bit to 0, stop output of target channel

stop clock of universal serial communication unit, set to reset state

finish termination configuration, enter into next processing.

Figure 15-56 Restart set-up steps for slave reception


restart configuration starts.
(mandatory)
master device preparation complete?
No
Yes
(mandatory)
port operation
(selection)
modify SPSm register configuration
(selection)
modify SMRmn register configuration
(selection)
modify SCRmn register configuration
(selection)
clear error flag
(mandatory)
port operation
(mandatory)
write into SSm register
restart configuration completes.

wait till commuication target (master device) stops or communication ends

via Configure port register and port mode register, set clock output of target channel to invalid.

re-configure when modifing operational clock configuration

re-configure when serial mode register mn.

re-configure when serial communication operation configuration register mn.

when OVF flag remains at set state, erase via serail flag clear trigger register mn(SIRmn).

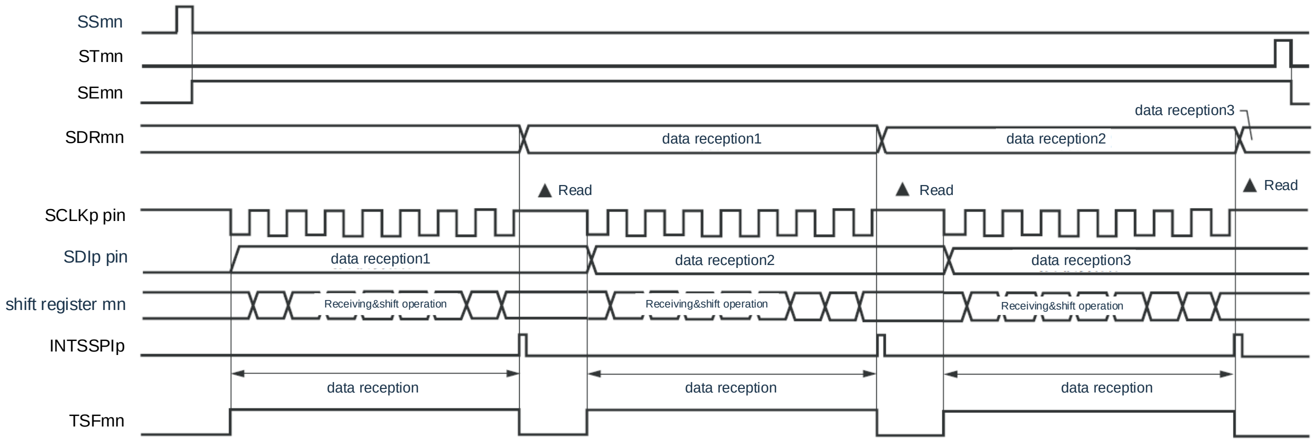
via Configure port register and port mode register, set data output of target channel to valid.

set SSmn bit of target channel to "1" (Semn=1, configure as operation enable state).
wait for master device clock.

Remark: If you override PER0 in the abort setting to stop providing the clock, you must wait until the communication object (the master device) is stopped or the communication is over for the initial set-up instead of restarting.

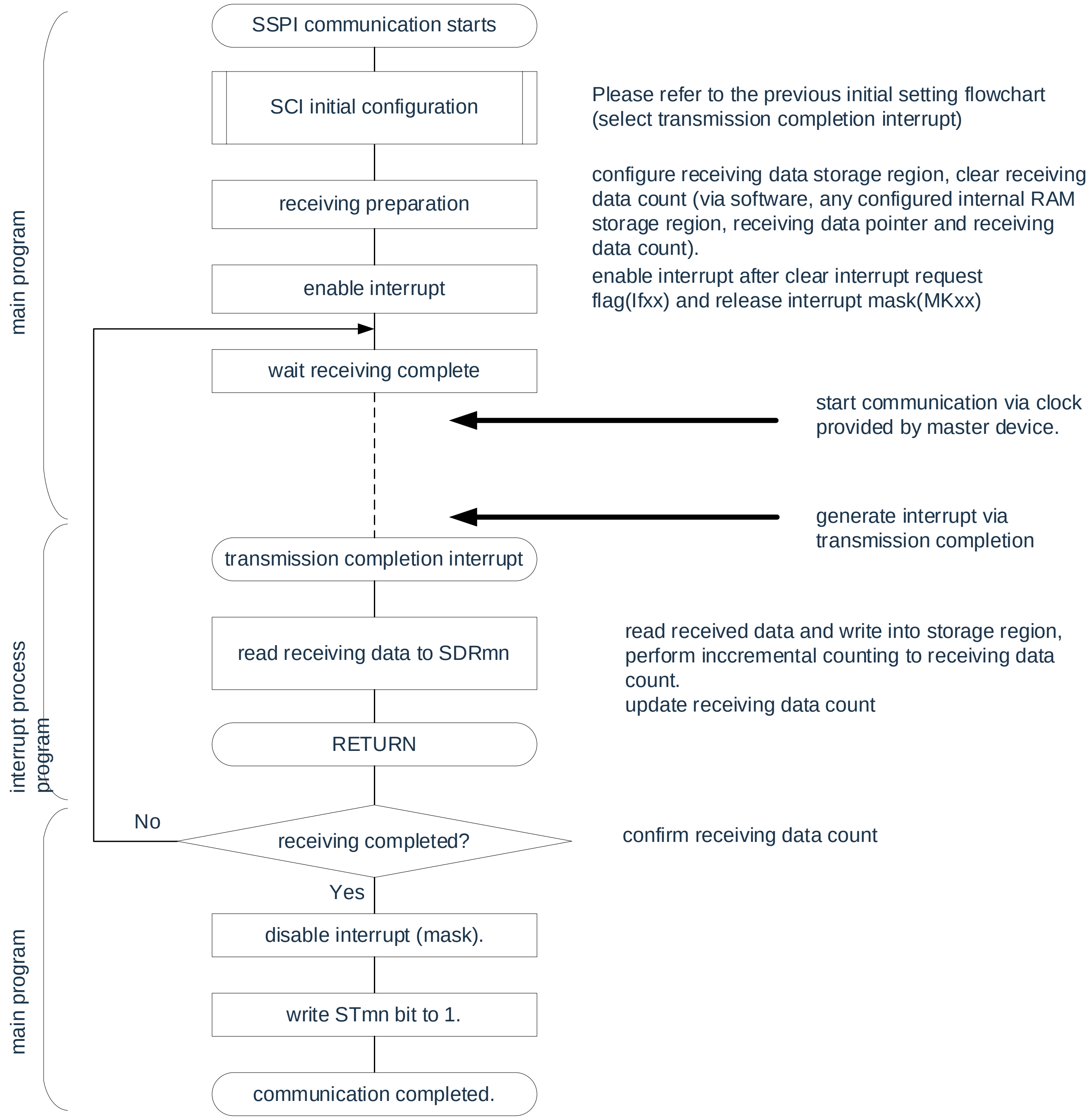
(3) Process flow (single receive mode)

Figure 15-57 Timing of slave reception (single receive mode)
(Type 1:DAPmn=0, CKPmn=0)



Remark: m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI number (p=00,01,10,11,20,21)

Figure 15-58 Flow chart of slave reception (single receive mode)


main program
interrupt process program
main program
SSPI communication starts
SCI initial configuration
Please refer to the previous initial setting flowchart (select transmission completion interrupt)
receiving preparation
configure receiving data storage region, clear receiving data count (via software, any configured internal RAM storage region, receiving data pointer and receiving data count).
enable interrupt
enable interrupt after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx)
wait receiving complete
start communication via clock provided by master device.
generate interrupt via transmission completion
transmission completion interrupt
read receiving data to SDRmn
read received data and write into storage region, perform inccremental counting to receiving data count.
update receiving data count
RETURN
No
receiving completed?
confirm receiving data count
Yes
disable interrupt (mask).
write STmn bit to 1.
communication completed.

15.5.6 Slave transmission and reception

Slave transmission and reception refers to the operation of this product microcontroller and other devices to send and receive data in the state of input transfer clock.

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, SDI00, SDO00	SCLK01, SDI01, SDO01	SCLK10, SDI10, SDO10	SCLK11, SDI11, SDO11	SCLK20, SDI20, SDO20	SCLK21, SDI21, SDO21
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21
	Interrupt at that end of the transfer may be selecte (single transfer mode) or buffer air-discontinuity (continuous transfer mode).					
Error detection flag	Only the overflow error detection flag (OVFmn).					
Length of transmit data	7 ~ 16 bits					
Transfer rate	Max.f_{MCK}/6[Hz]$^{Note\ 1,2}$					
Data phase	Can be selected by the DAPmn bit of the SCRmn register. ·DAPmn=0: The data input/output is started when the serial clock is started. ·DAPmn=1: The data input/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: forward • CKPmn=1: inverted					
Data orientation	MSB FiArst or LSB First					

Note 1. The maximum transfer rate f_{MCK}/6[Hz] is used because the external serial clock input by SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, SCLK21.

2. It must be used within the scope of the peripheral functional characteristics (refer to the data sheet) that meet this condition and satisfy the electrical characteristics.

Note 1.f_{MCK}: Operating clock frequency of the object channel

2.m: Unit number (m=0,1,2) n: Channel number (n=0,1)

(1) Register settings

Figure 15-59 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20,SSPI21)
Example of register setting sontents during slave transmission and reception

(a) serial mode register mn (SMRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn 0/1	CCSmn 1	0	0	0	0	0	STSmn 0	0	SISmn0 0	1	0	0	MDmn2 0	MDmn1 0	MDmn0 0/1

channel n operational clock (fMCK)
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

interrupt source of channel n
0: Transmit completion interrupt
1: Buffer empty interrupt

(b) serial communication operation configuration register mn (SCRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn 1	RXEmn 1	DAPmn 0/1	CKPmn 0/1	0	EOCmn 0	PTCmn1 0	PTCmn0 0	DIRmn 0/1	0	SLCmn1 0	SLCmn0 0	DLSmn3 0/1	DLSmn2 0/1	DLSmn1 0/1	DLSmn0 0/1

data and clock phase selection (details refer to registers controlling universal serial communication unit)

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

Setting of data length
DLSmn3~0: 7-bit~16-bit data length selection

(c) serial data regsiter mn (SDRmn)

(1) When operation stops (SEmn=0)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration（0000000B)							0	0	0	0	0	0	0	0	0

(2) During operation（SEmn=1）（lower 8 bits: SDRmnL）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	transmit/receive data register															
									SDRmnL							

(d) serial output register m(SOm) ......Only configure bit of target channel

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm	0	0	0	0	CKOm3 ×	CKOm2 ×	CKOm1 ×	CKOm0 ×	0	0	0	0	SOm3 0/1	SOm2 0/1	SOm1 0/1	SOm0 0/1

(e) serial output enable registerm (SOEm)….only set bit of target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	SOEm3 0/1	SOEm2 0/1	SOEm1 0/1	SOEm0 0/1

(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 0/1	SSm2 0/1	SSm1 0/1	SSm0 0/1

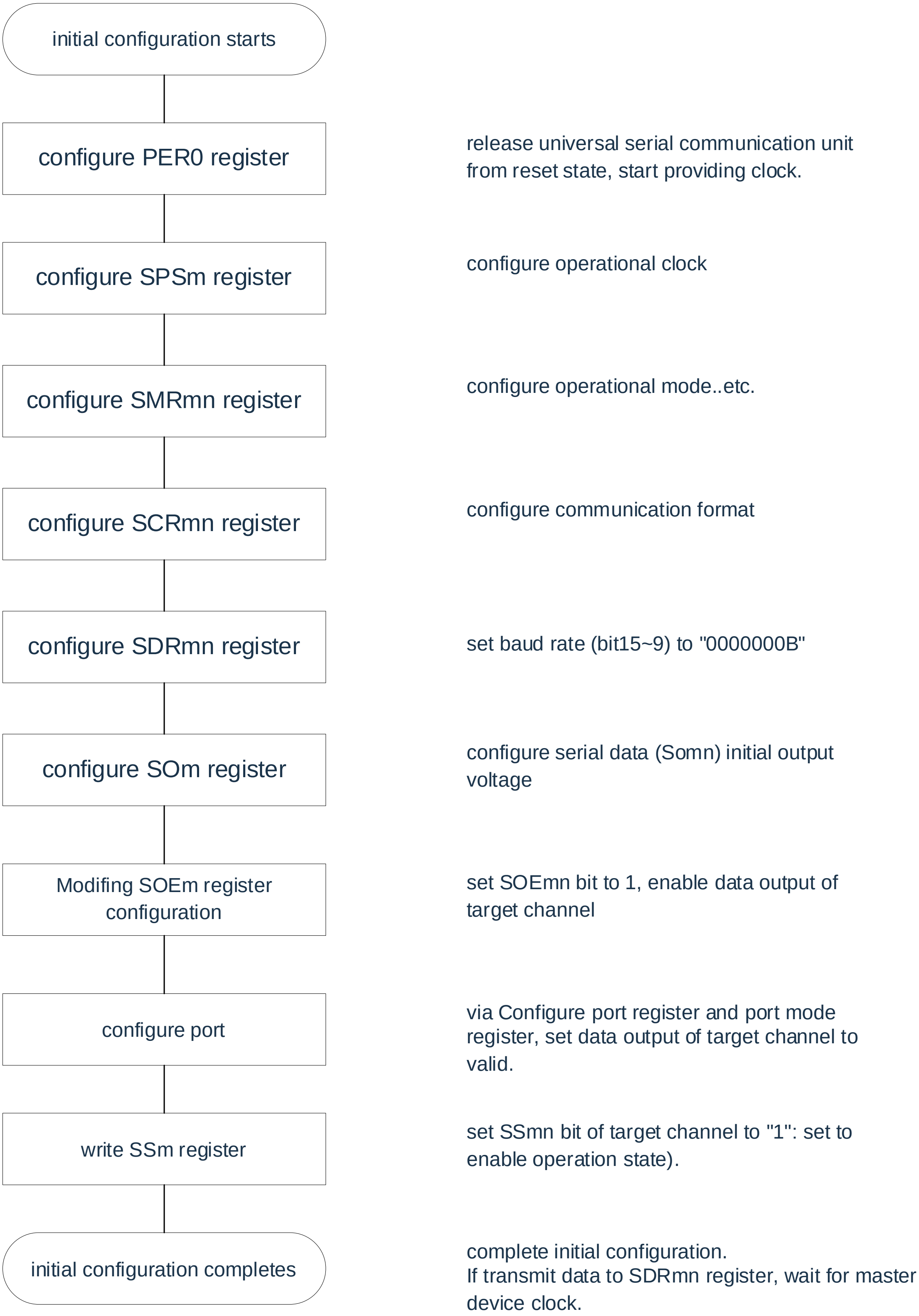
Note Before the master device starts outputting the clock, the SIOp register must be set to send data.

Note 1. m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI Number (p=00,01,10,11,20,21)

2. □ : Is fixed in the SSPI slave send and receive mode. ▨: Cannot set (set initial value).
×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).
0/1: The "0" or "1" is set according to the user.

(2) Procedure

Figure 15-60 Initial set-up steps for slave transmission and reception



release universal serial communication unit from reset state, start providing clock.

configure operational clock

configure operational mode..etc.

configure communication format

set baud rate (bit15~9) to "0000000B"

configure serial data (Somn) initial output voltage

set SOEmn bit to 1, enable data output of target channel

via Configure port register and port mode register, set data output of target channel to valid.

set SSmn bit of target channel to "1": set to enable operation state).

complete initial configuration.
If transmit data to SDRmn register, wait for master device clock.

Note Before the master device starts outputting the clock, the SDRmn register must be set to send data.

Figure 15-61 Stop steps for slave transmission and reception

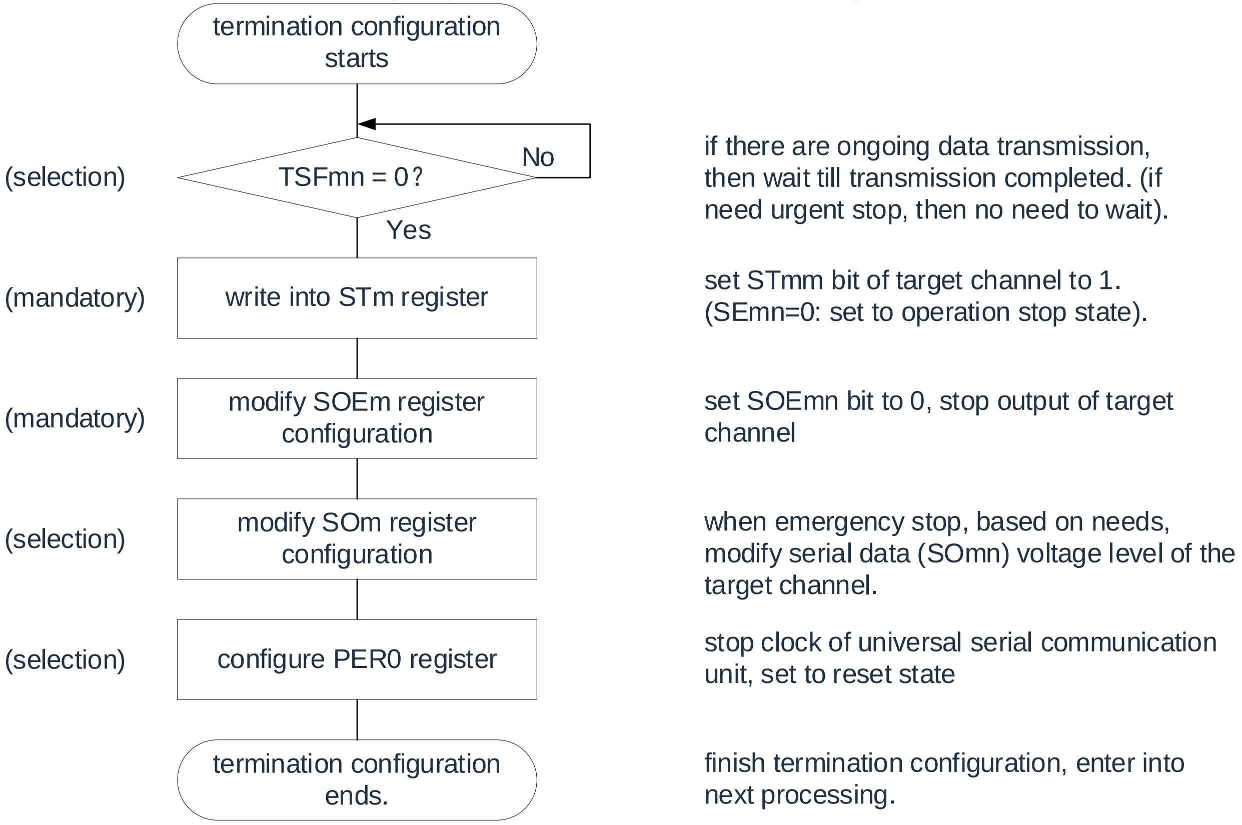
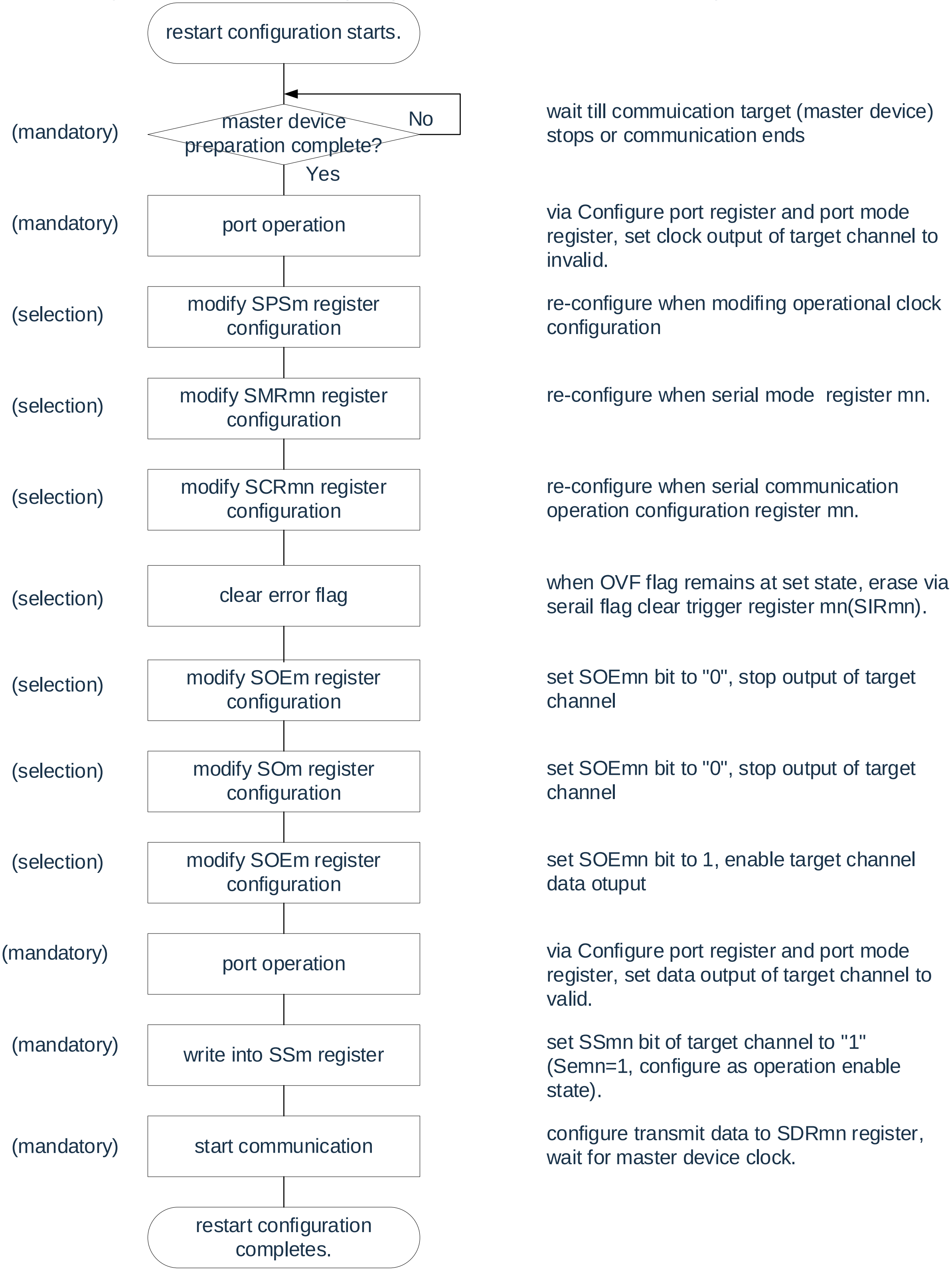

termination configuration starts
(selection)
TSFmn = 0 ?
No
Yes
if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).
(mandatory)
write into STm register
set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).
(mandatory)
modify SOEm register configuration
set SOEmn bit to 0, stop output of target channel
(selection)
modify SOm register configuration
when emergency stop, based on needs, modify serial data (SOmn) voltage level of the target channel.
(selection)
configure PER0 register
stop clock of universal serial communication unit, set to reset state
termination configuration ends.
finish termination configuration, enter into next processing.

Figure 15-62 Restart set-up steps for slave transmission and reception

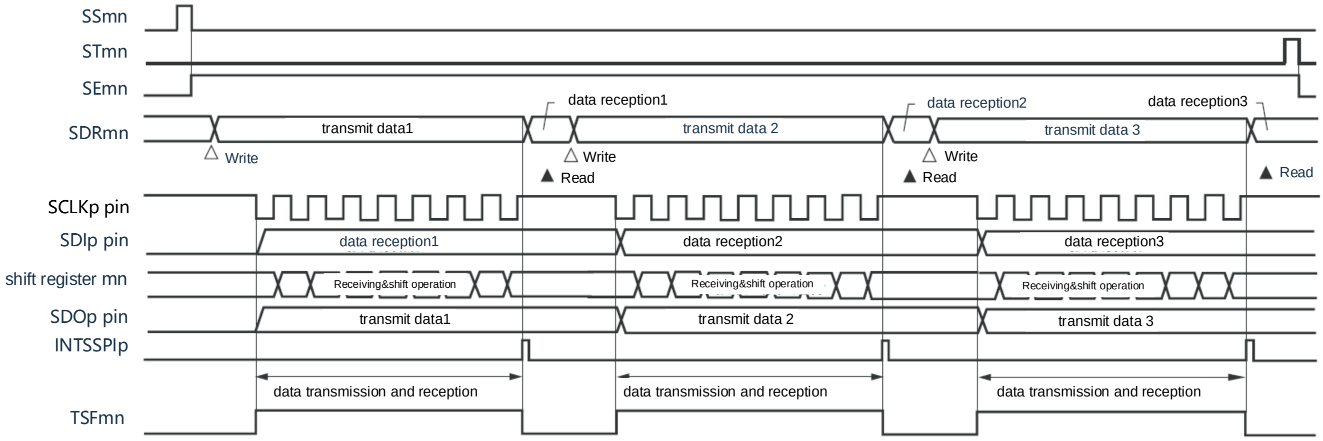

restart configuration starts.
(mandatory)
master device preparation complete?
No
Yes
wait till commuication target (master device) stops or communication ends
(mandatory)
port operation
via Configure port register and port mode register, set clock output of target channel to invalid.
(selection)
modify SPSm register configuration
re-configure when modifing operational clock configuration
(selection)
modify SMRmn register configuration
re-configure when serial mode register mn.
(selection)
modify SCRmn register configuration
re-configure when serial communication operation configuration register mn.
(selection)
clear error flag
when OVF flag remains at set state, erase via serail flag clear trigger register mn(SIRmn).
(selection)
modify SOEm register configuration
set SOEmn bit to "0", stop output of target channel
(selection)
modify SOm register configuration
set SOEmn bit to "0", stop output of target channel
(selection)
modify SOEm register configuration
set SOEmn bit to 1, enable target channel data otuput
(mandatory)
port operation
via Configure port register and port mode register, set data output of target channel to valid.
(mandatory)
write into SSm register
set SSmn bit of target channel to "1" (Semn=1, configure as operation enable state).
(mandatory)
start communication
configure transmit data to SDRmn register, wait for master device clock.
restart configuration completes.

Note 1. The SDRmn register setting must be sent before the master device starts outputting the clock.

2. If you override PER0 in the abort setting to stop providing the clock, you must wait until the communication object (master) stops or the communication is over for the initial set-up instead of restarting.

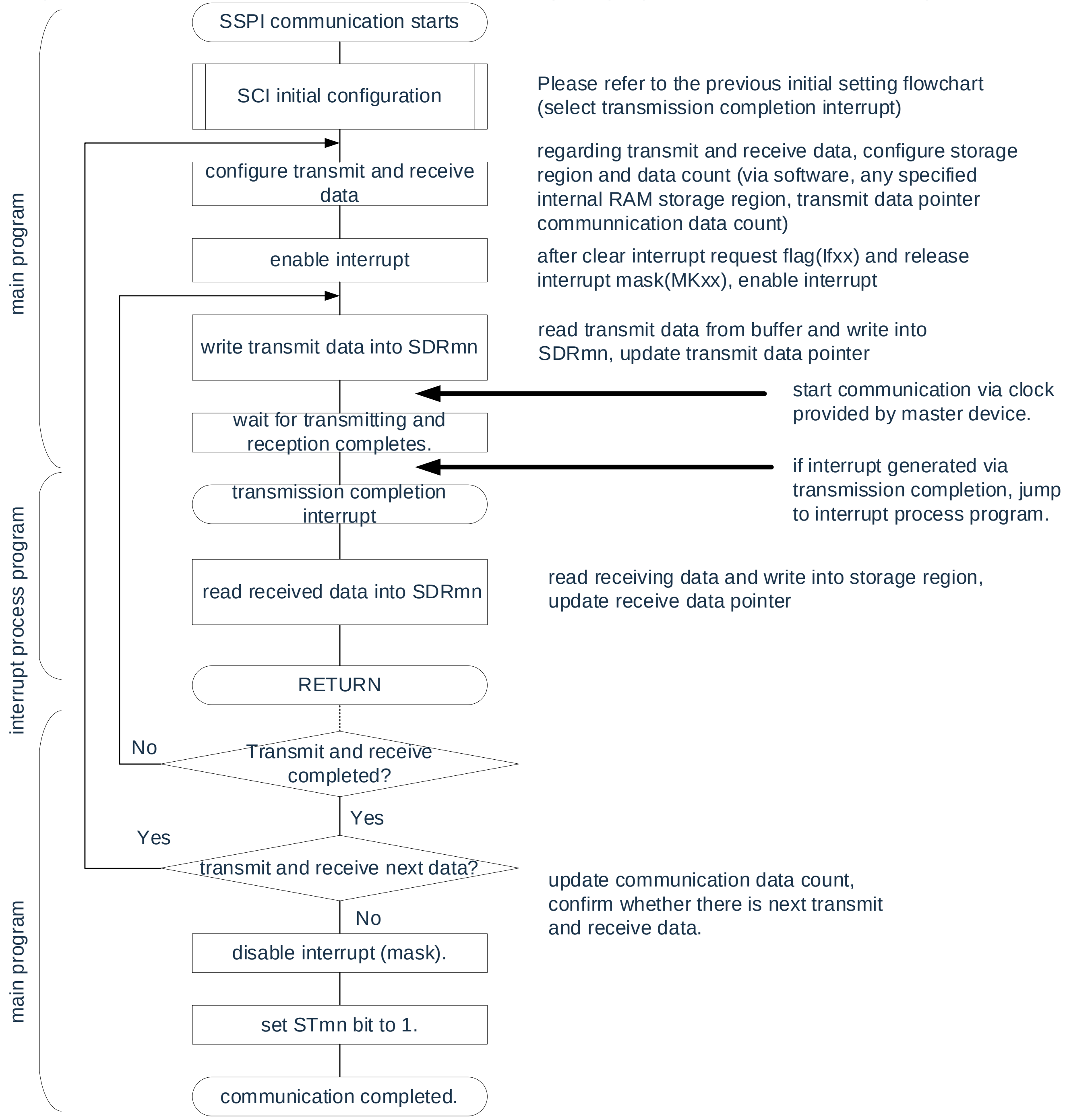
(3) Process flow (single transmit and receive mode)

Figure 15-63 Timing of slave transmission and reception (single transmit and receive mode)
(Type 1: DAPmn=0, CKPmn=0)



Remark: m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI Number (p=00,01,10,11,20,21)

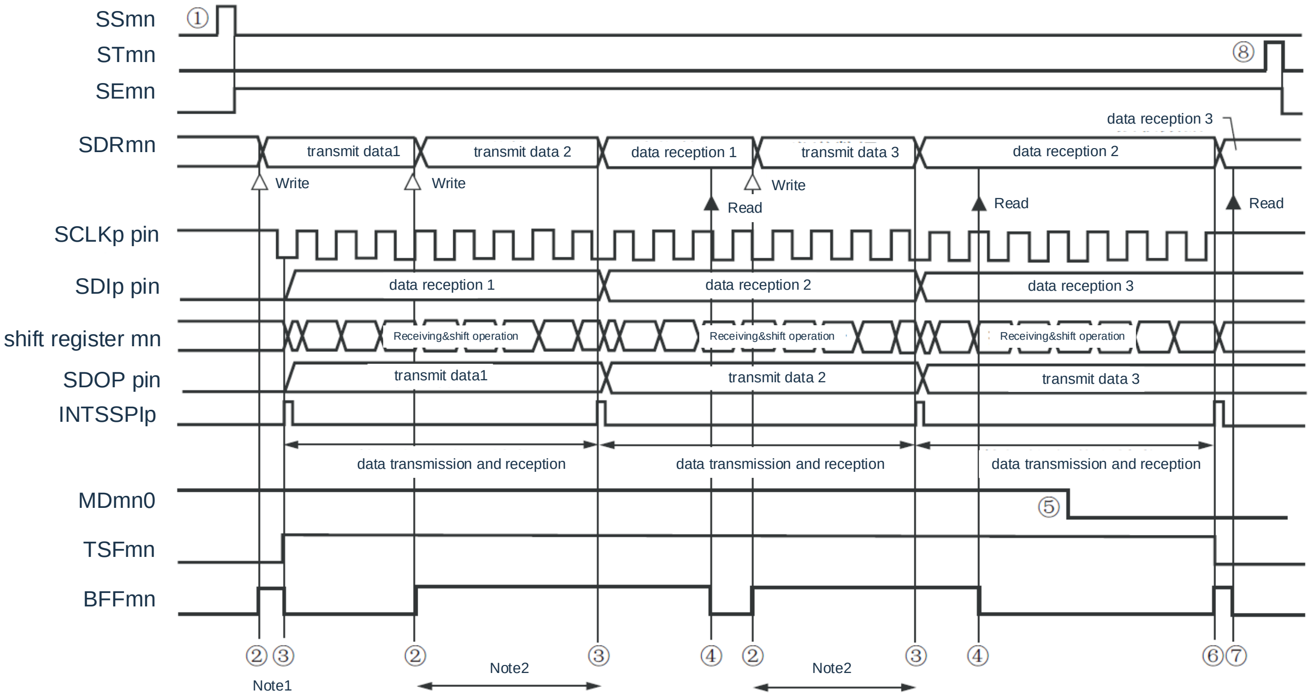
Figure 15-64 Flowchart for slave transmission and reception (single transmit and receive mode)


main program
interrupt process program
main program
SSPI communication starts
SCI initial configuration
Please refer to the previous initial setting flowchart (select transmission completion interrupt)
configure transmit and receive data
regarding transmit and receive data, configure storage region and data count (via software, any specified internal RAM storage region, transmit data pointer communnication data count)
enable interrupt
after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt
write transmit data into SDRmn
read transmit data from buffer and write into SDRmn, update transmit data pointer
start communication via clock provided by master device.
wait for transmitting and reception completes.
if interrupt generated via transmission completion, jump to interrupt process program.
transmission completion interrupt
read received data into SDRmn
read receiving data and write into storage region, update receive data pointer
RETURN
No
Transmit and receive completed?
Yes
Yes
transmit and receive next data?
update communication data count, confirm whether there is next transmit and receive data.
No
disable interrupt (mask).
set STmn bit to 1.
communication completed.

Remark: Before the master device starts outputting the clock, the SIOp register must be set to send data.

(4) Processing flow (continuous transmit and receive mode)

Figure 15-65 Timing of slave transmission and reception (continuous transmit and receive mode)
(Type 1: DAPmn=0, CKPmn=0)



Note 1. Rewrite the transmission data if the BFFmn bit of the serial status register mn (SSRmn) is "1" (when valid data is stored in SDRmn).

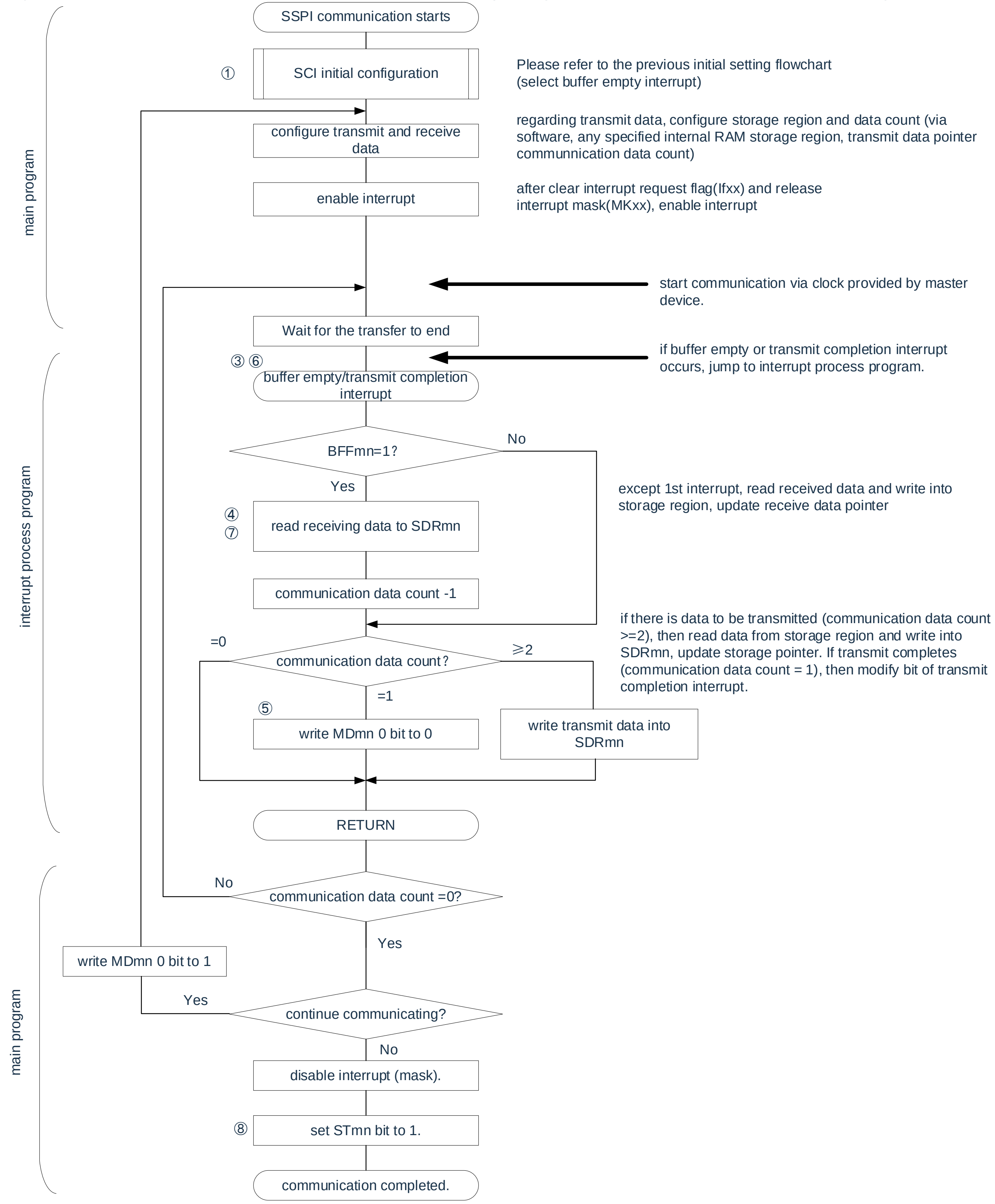
2. If the SDRmn register is read during this time, the transmit data can be read. At this time, the transmit operation is not affected.

Notice The MDmn0 bit of the serial mode register mn(SMRmn) can be rewritten even in operation. However, in order to be able to catch the end of the transmission of the last transmitted data interrupt, it is necessary to override before starting the last transmission.

Remark 1. ① to ⑧ in the Figure corresponds to ① to ⑧ in Figure 15-66 Flowchart of Slave transmission and reception (continuous transmit and receive mode).

2.m: Unit number (m=0,1,2) n: Channel number (n=0,1) p: SSPI Number (p=00, 01, 10, 11, 20, 21)

Figure 15-66 Flowchart of Slave transmission and reception (continuous transmit and receive mode)


main program
interrupt process program
main program
SSPI communication starts
①
SCI initial configuration
configure transmit and receive data
enable interrupt
Wait for the transfer to end
③ ⑥
buffer empty/transmit completion interrupt
BFFmn=1?
No
Yes
④
⑦
read receiving data to SDRmn
communication data count -1
=0
communication data count?
≥2
=1
⑤
write MDmn 0 bit to 0
write transmit data into SDRmn
RETURN
No
communication data count =0?
Yes
write MDmn 0 bit to 1
Yes
continue communicating?
No
disable interrupt (mask).
⑧
set STmn bit to 1.
communication completed.

Please refer to the previous initial setting flowchart (select buffer empty interrupt)

regarding transmit data, configure storage region and data count (via software, any specified internal RAM storage region, transmit data pointer communnication data count)

after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt

start communication via clock provided by master device.

if buffer empty or transmit completion interrupt occurs, jump to interrupt process program.

except 1st interrupt, read received data and write into storage region, update receive data pointer

if there is data to be transmitted (communication data count >=2), then read data from storage region and write into SDRmn, update storage pointer. If transmit completes (communication data count = 1), then modify bit of transmit completion interrupt.

Notice: Before the master device starts outputting the clock, the SDRmn register must be set to send data.

Remark: ① to ⑧ in the figure correspond to ① to ⑧ in “Figure 15-65 Timing of slave transmission and reception (continuous transmit and receive mode)”.

15.5.7 Calculation of transmission clock frequency

The transmitting clock frequency for 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI20,SSPI21) communication can be calculated using the following formula.

(1) Master

(Transfer clock frequency)={Operating clock frequency of the object channel (f_{MCK}) }÷(SDRmn[15:9]+1) ÷2[Hz]

(2) Slave

(Transfer clock frequency)={Serial clock (SCLK) frequency provided by the master device} Note [Hz]

Note: The maximum allowable transfer clock frequency is f_{MCK}/6.

Remark: The value of SDRmn[15:9] is 0~127 because it is bit15~9 of serial data register mn(SDRmn).

The operating clock (f_{MCK}) is determined by the bit15 (CKSmn) of the serial clock selection register m (SPSm) and the serial mode register mn (SMRmn).

Table 15-2 Selection of 3-wire serial I/O operating clock

SMRmn register	SPSm register								Operating clock (f_{MCK}) Note	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		f_{CLK}=32MHz in operation
0	X	X	X	X	0	0	0	0	f_{CLK}	32MHz
	X	X	X	X	0	0	0	1	$f_{CLK}/2$	16MHz
	X	X	X	X	0	0	1	0	$f_{CLK}/2^{2}$	8MHz
	X	X	X	X	0	0	1	1	$f_{CLK}/2^{3}$	4MHz
	X	X	X	X	0	1	0	0	$f_{CLK}/2^{4}$	2MHz
	X	X	X	X	0	1	0	1	$f_{CLK}/2^{5}$	1MHz
	X	X	X	X	0	1	1	0	$f_{CLK}/2^{6}$	500kHz
	X	X	X	X	0	1	1	1	$f_{CLK}/2^{7}$	250kHz
	X	X	X	X	1	0	0	0	$f_{CLK}/2^{8}$	125kHz
	X	X	X	X	1	0	0	1	$f_{CLK}/2^{9}$	62.5kHz
	X	X	X	X	1	0	1	0	$f_{CLK}/2^{10}$	31.25kHz
	X	X	X	X	1	0	1	1	$f_{CLK}/2^{11}$	15.63kHz
	X	X	X	X	1	1	0	0	$f_{CLK}/2^{12}$	7.81kHz
	X	X	X	X	1	1	0	1	$f_{CLK}/2^{13}$	3.91kHz
	X	X	X	X	1	1	1	0	$f_{CLK}/2^{14}$	1.95kHz
	X	X	X	X	1	1	1	1	$f_{CLK}/2^{15}$	977Hz
1	0	0	0	0	X	X	X	X	f_{CLK}	32MHz
	0	0	0	1	X	X	X	X	$f_{CLK}/2$	16MHz
	0	0	1	0	X	X	X	X	$f_{CLK}/2^{2}$	8MHz
	0	0	1	1	X	X	X	X	$f_{CLK}/2^{3}$	4MHz
	0	1	0	0	X	X	X	X	$f_{CLK}/2^{4}$	2MHz
	0	1	0	1	X	X	X	X	$f_{CLK}/2^{5}$	1MHz
	0	1	1	0	X	X	X	X	$f_{CLK}/2^{6}$	500kHz
	0	1	1	1	X	X	X	X	$f_{CLK}/2^{7}$	250kHz
	1	0	0	0	X	X	X	X	$f_{CLK}/2^{8}$	125kHz
	1	0	0	1	X	X	X	X	$f_{CLK}/2^{9}$	62.5kHz
	1	0	1	0	X	X	X	X	$f_{CLK}/2^{10}$	31.25kHz
	1	0	1	1	X	X	X	X	$f_{CLK}/2^{11}$	15.63kHz
	1	1	0	0	X	X	X	X	$f_{CLK}/2^{12}$	7.81kHz
	1	1	0	1	X	X	X	X	$f_{CLK}/2^{13}$	3.91kHz
	1	1	1	0	X	X	X	X	$f_{CLK}/2^{14}$	1.95kHz
	1	1	1	1	X	X	X	X	$f_{CLK}/2^{15}$	977Hz

Note: To change the clock selected as f_{CLK} (change the value of the System Clock Control Register (CKC), you must change after stopping Universal Serial Communication Unit (SCI) =000FH.

Remark: 1.X: Ignore

2. m: Unit number (m=0,1,2) n: Channel number (n=0,1)

15.5.8 Procedure for errors occurring during 3-wire serial I/O communication (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)

The steps for handling errors occurring during 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) communication are shown in Figure 15-67.

Figure 15-67 Handling steps when overflow errors occur

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.

Remark: m: Unit number (m=0,1,2) n: Channel number (n=0,1)

15.6 Operation of clock-synchronous serial communication with slave select input function

Channel 0 of the SCI0 is a channel for clock synchronous serial communication that supports the slave select input function.

[Transmitting and Receiving Data]

- 7-bit ~ 16-bit data length
- Phase control for transmitting and receiving data
- MSB/LSB first
- Level settings for sending and receiving data

[Clock Control]

- Phase control of input/output clock
- A transmission period generated by a pre-divider and an intra-channel counter is set.
- Maximum transfer rate Note

  Slave communication: Max.$f_{MCK}/6$

[Interrupt Function]

- Interrupt transmission end, buffer null interrupt

[Error Detection Flag]

- Overflow error

Note: It must be used within the range that satisfies the SCLK cycle time (T_{KCY}) characteristics. For details, please refer to the data sheet.

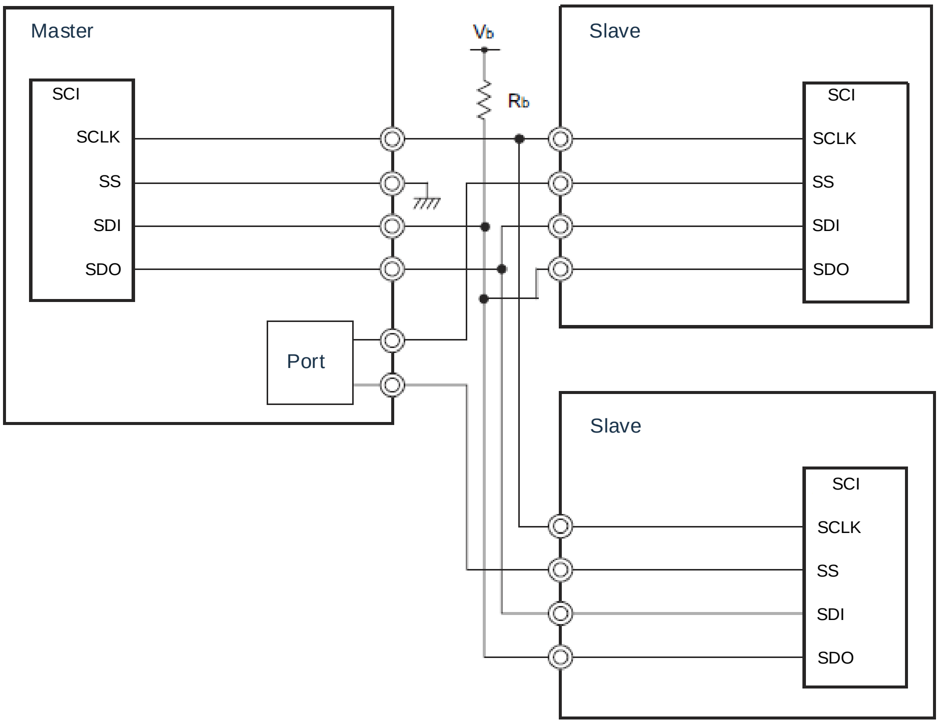
The Slave selectionInput function runs in three communications:

- Slave transmission (Refer to 15.6.1)
- Slave reception (Refer to 15.6.2)
- Slave transmission and reception (Refer to 15.6.3)

By using the slave selection input function, a master device can connect multiple slave devices for communication. The master device outputs the slave selection signal to the slave device (1) of the communication object, and each slave device determines whether it is selected as the communication object and controls the output of the SDO pin. When a slave device is selected as the communication object, the SDO pin can communicate to the master device by sending data; when a slave device is not selected as the communication object, the SDO pin is output high, so it is necessary to set the SDO pin to Nch-O.D and pull up the node when multiple slave devices are connected. In addition, the serial clock of the master device is not transmitted or received even if it is input.

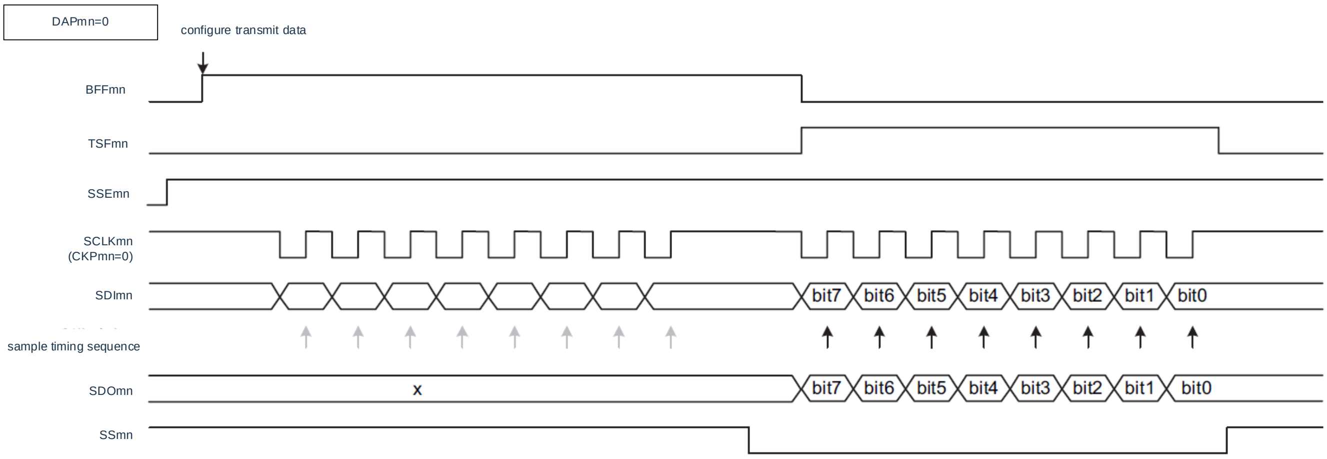
Note The slave selection signal must be output by the operation of the port.

Figure 15-68 Example of the structure of the slave selection input function



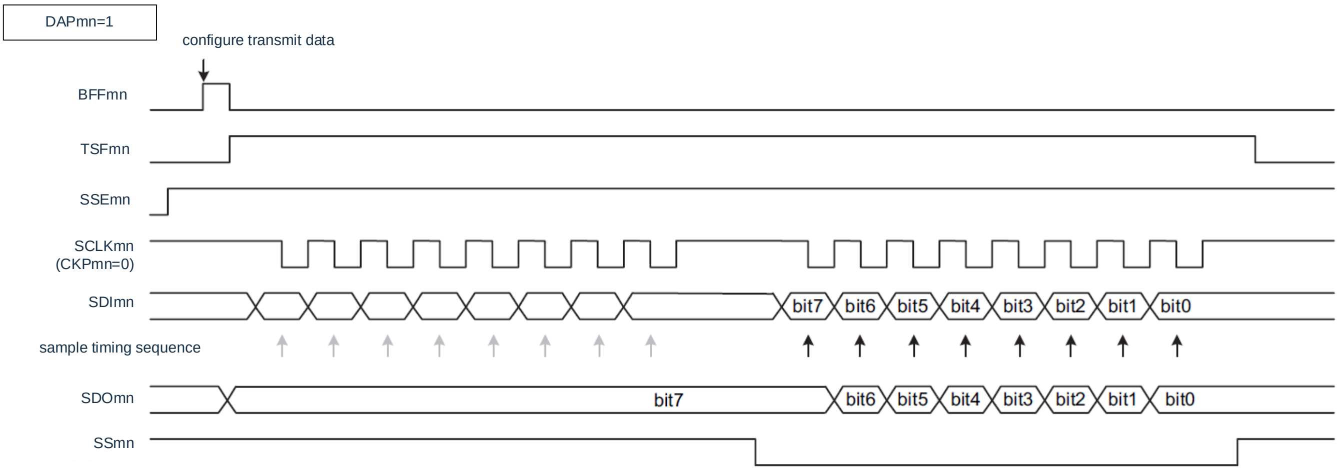
Notice: The SDO00 pin is selected as an N-channel drain open-circuit output mode.

Figure 15-69 Timing of slave select input function



During the period when SSmn is high, no transmission is performed even at the descending edge of SCKmn (serial clock) and no sampling of received data synchronized with the ascending edge is performed.

During a period when the SSmn is low, output data (shift) is synchronized with the falling edge of the serial clock and data is received synchronously with the rising edge.



When the DAPmn bit is '1', if the transmission data is set during the high level of the SSmn, the initial data (bit7) is supplied to the data output. However, even the rising edge of the SCLKmn (serial clock) does not shift, and does not sample the accepted data synchronized with the falling edge. If the SSmn becomes low, the output data (shift) is synchronized with the next rising edge and the data is received in synchronization with the falling edge.

Remark: m: Unit number (m=0) n: Channel number (n=0)

15.6.1 Slave transmission

Slave transmission refers to the operation of this product to send data to other devices in the state of transmitting clocks from other device inputs.

Slave selection input function	SSPI00
Object channel	Channel 0 for SCI0
Pin used	SCLK00, SDO00, SS00
Interrupt	INTSSPI00
	Interrupt at that end of the transfer may be selecte (single transfer mode) or buffer air-discontinuity (continuous transfer mode).
Error detection flag	Only the overflow error detection flag (OVFmn).
Length of transmit data	7 ~ 16 bits
Transfer rate	Max.f_{MCK}/6[Hz]Note 1,2
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: forward • CKPmn=1: inverted
Data orientation	MSB First or LSB First
Slave selectioninput function	You can select the operation of the slave selectionfunction.

Note 1. Since the external serial clock input to the SCLK00 pin is sampled internally and used, the maximum transfer rate is f_{MCK}/6[Hz].

2. It It must be used within the scope of the peripheral functional characteristics (refer to the data sheet) that meet this condition and satisfy the electrical characteristics.

Remark 1. f_{MCK}: Operating clock frequency of the object channel

2. m: Unit number (m=0)n: Channel number (n=0)

(1) Register settings

Figure 15-70 Example of register setting contents for slave select input function (SSPI00) during slave transmission (1/2)

(a) serial mode register mn (SMRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn 0/1	CCSmn 1	0	0	0	0	0	STSmn 0	0	SISmn0 0	1	0	0	MDmn2 0	MDmn1 0	MDmn0 0/1

channel n operational clock (fMCK)
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

interrupt source of channel n
0: Transmit completion interrupt
1: Buffer empty interrupt

(b) serial communication operation configuration registermn mn(SCRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn 1	RXEmn 0	DAPmn 0/1	CKPmn 0/1	0	EOCmn 0	PTCmn1 0	PTCmn0 0	DIRmn 0/1	0	SLCmn1 0	SLCmn0 0	DLSmn3 0/1	DLSmn2 0/1	DLSmn1 0/1	DLSmn0 0/1

data and clock phase selection (details refer to registers controlling universal serial communication unit)

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

Setting of data length
DLSmn3~0: 7-bit~16-bit data length selection

(c) serial data register mn (SDRmn)

(1) When operation stops (SEmn=0)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration (0000000B)							0	0	0	0	0	0	0	0	0

(2) During operation (SEmn=1) (Lower 8 bits: SDRmnL)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	transmit data register															
									SDRmnL							

(d) serial output register m (SOm)…. Only configure bit of target channel

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm	0	0	0	0	CKOm3 ×	CKOm2 ×	CKOm1 ×	CKOm0 ×	0	0	0	0	SOm3 ×	SOm2 ×	SOm1 ×	SOm0 0/1

(e) serial output enable register m (SOEm)···. Only set bit of target channel to "1".

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	SOEm3 ×	SOEm2 ×	SOEm1 ×	SOEm0 0/1

Note 1.m: Unit number (m=0)n: Channel Number (n=0)p:SSPI Number (p=00)

2. □ : Fixed in SSPI slave send mode. ▨ : Cannot set (set initial value).

×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).

0/1: The "0" or "1" is set according to the user.

Figure 15-72 Example of register setting contents for slave select input function (SSPI00) during slave transmission (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															SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	×	0/1

(g) Slave Select Function Enable Register (SSE) ...... This is the control of the SSImn pin of the SSPImn slave channel (channel n of unit m).

	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 is invalid
1: SS00 pin input is valid

Note 1.m: Unit number (m=0)n: Channel Number (n=0)p:SSPI Number (p=00)

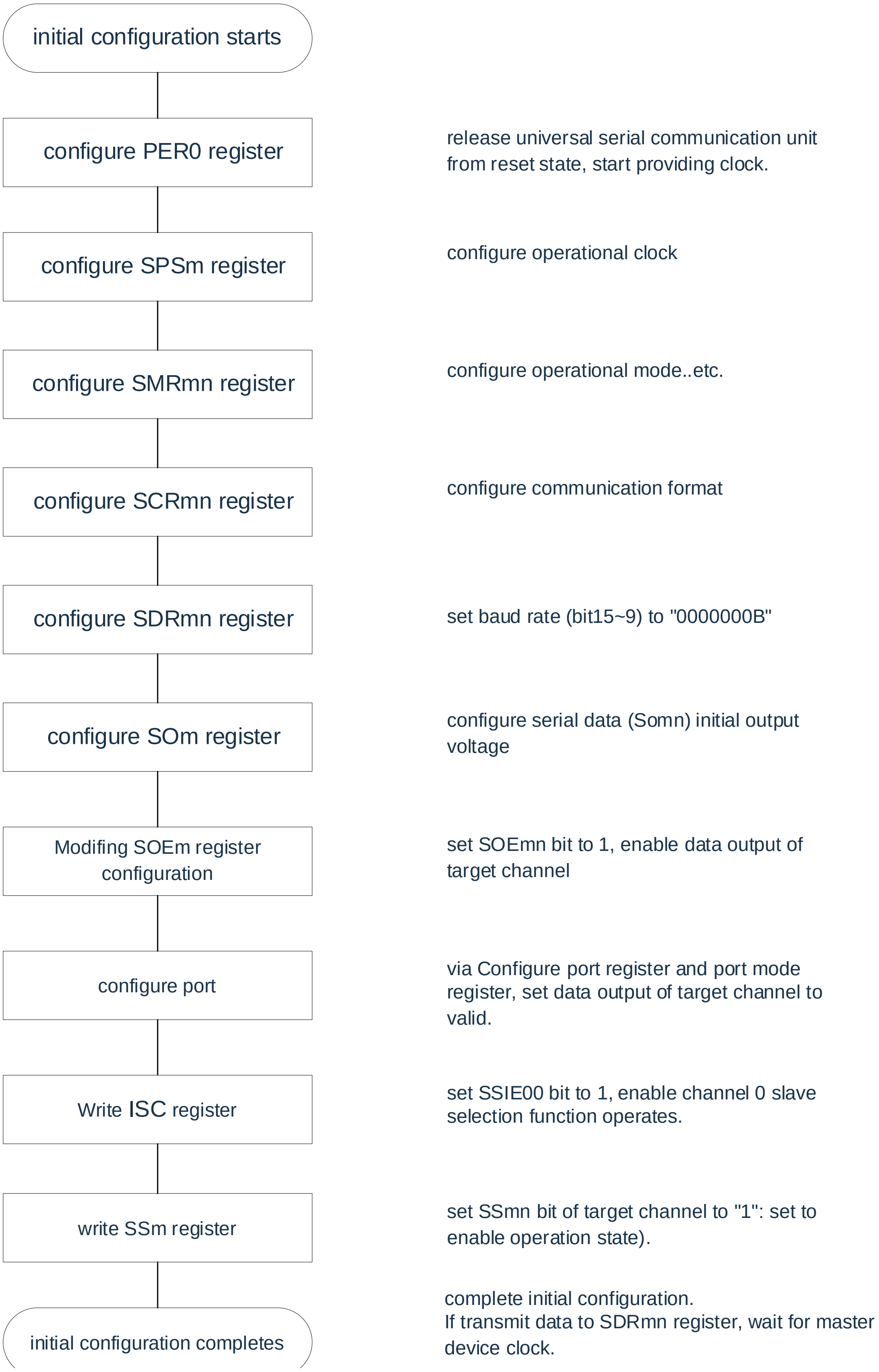
2. □ : Fixed in SSPI slave send mode. ▨ : Cannot set (set initial value).

×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).

0/1: The "0" or "1" is set according to the user.

(2) Operation procedure

Figure 15-71 Initial set-up steps for slave transmission



release universal serial communication unit from reset state, start providing clock.

configure operational clock

configure operational mode..etc.

configure communication format

set baud rate (bit15~9) to "0000000B"

configure serial data (Somn) initial output voltage

set SOEmn bit to 1, enable data output of target channel

via Configure port register and port mode register, set data output of target channel to valid.

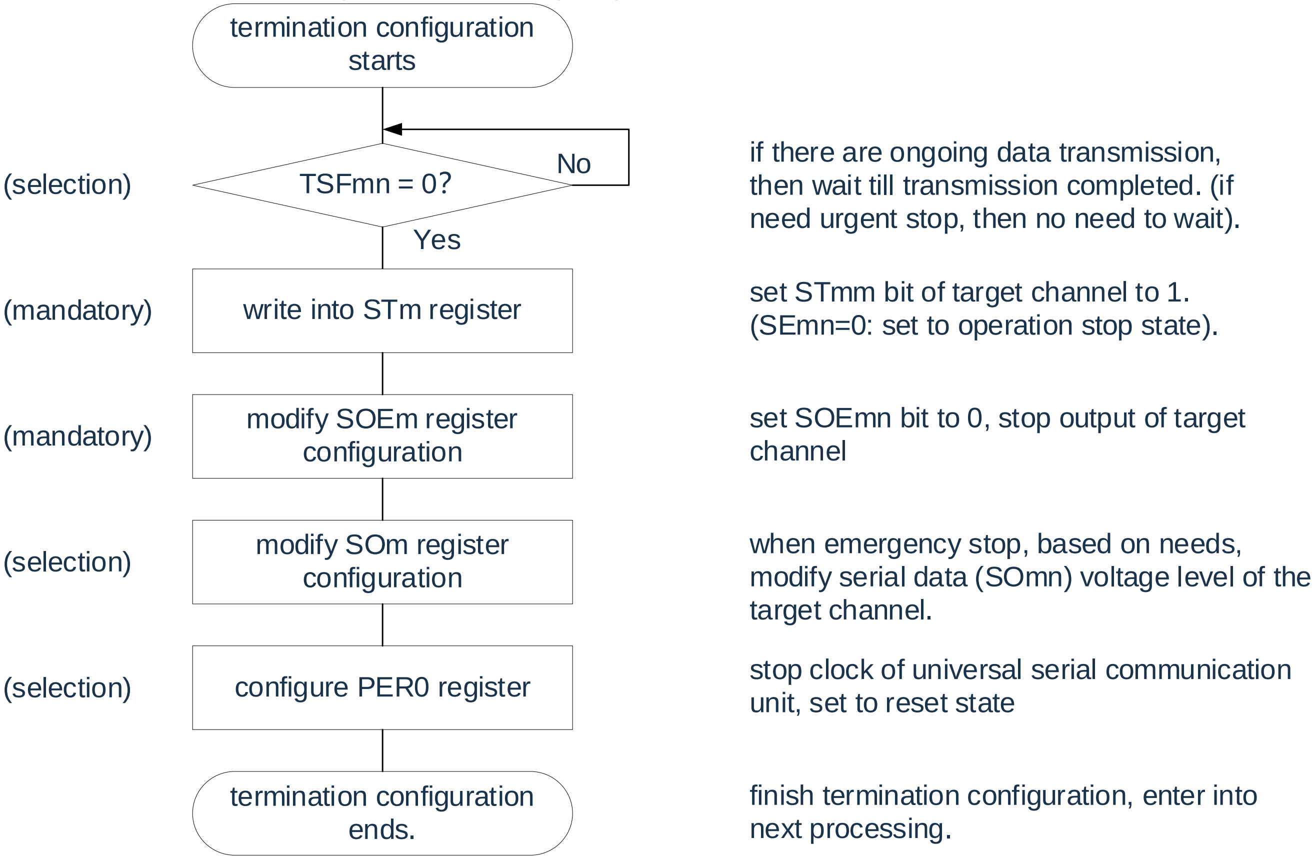
set SSIE00 bit to 1, enable channel 0 slave selection function operates.

set SSmn bit of target channel to "1": set to enable operation state).

complete initial configuration.
If transmit data to SDRmn register, wait for master device clock.

Remark m: Unit number (m=0)n: Channel Number (n=0)p:SSPI Number (p=00)

Figure 15-72 Stop steps for slave transmission

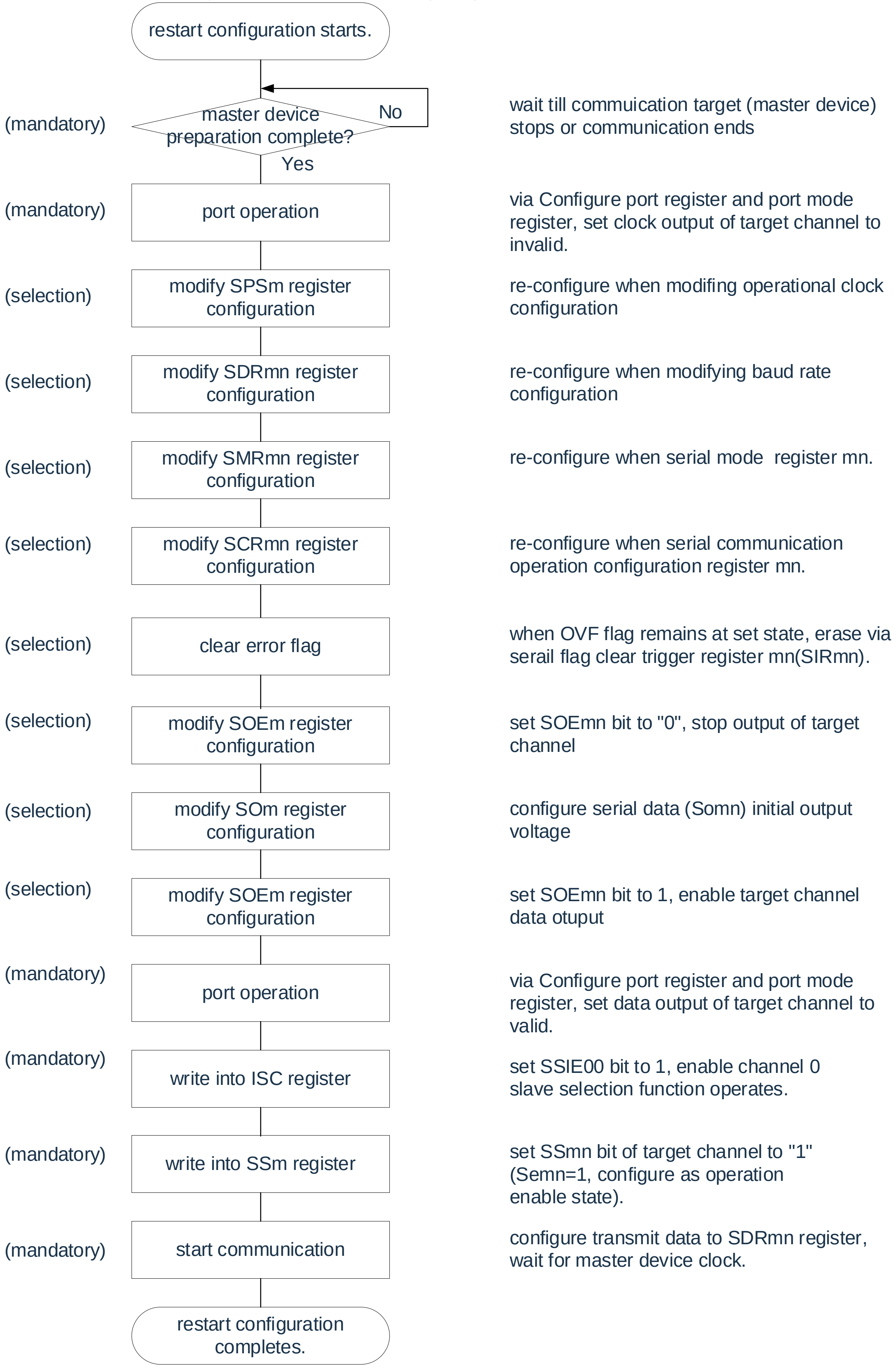

termination configuration starts
(selection)
TSFmn = 0?
No
Yes
if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).
(mandatory)
write into STm register
set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).
(mandatory)
modify SOEm register configuration
set SOEmn bit to 0, stop output of target channel
(selection)
modify SOm register configuration
when emergency stop, based on needs, modify serial data (SOmn) voltage level of the target channel.
(selection)
configure PER0 register
stop clock of universal serial communication unit, set to reset state
termination configuration ends.
finish termination configuration, enter into next processing.

Remark m: Unit number (m=0)n: Channel Number (n=0)p:SSPI Number (p=00)

Note 1. If you override PER0 in the abort setting to stop providing the clock, you must wait until the communication object (master) is stopped or the communication is over for the initial set-up instead of restarting the set-up.

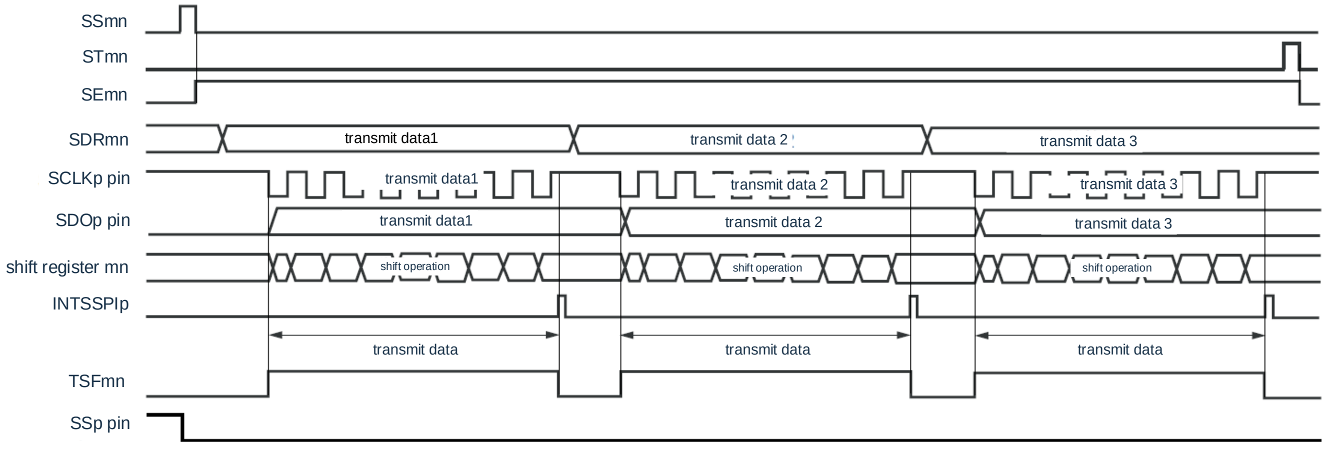
2.m: Unit number (m=0)n: Channel Number (n=0)p:SSPI Number (p=00)

Figure 15-73 Restart set-up steps for slave transmission


restart configuration starts.
(mandatory)
master device preparation complete?
No
Yes
wait till commuication target (master device) stops or communication ends
(mandatory)
port operation
via Configure port register and port mode register, set clock output of target channel to invalid.
(selection)
modify SPSm register configuration
re-configure when modifing operational clock configuration
(selection)
modify SDRmn register configuration
re-configure when modifying baud rate configuration
(selection)
modify SMRmn register configuration
re-configure when serial mode register mn.
(selection)
modify SCRmn register configuration
re-configure when serial communication operation configuration register mn.
(selection)
clear error flag
when OVF flag remains at set state, erase via serail flag clear trigger register mn(SIRmn).
(selection)
modify SOEm register configuration
set SOEmn bit to "0", stop output of target channel
(selection)
modify SOm register configuration
configure serial data (Somn) initial output voltage
(selection)
modify SOEm register configuration
set SOEmn bit to 1, enable target channel data otuput
(mandatory)
port operation
via Configure port register and port mode register, set data output of target channel to valid.
(mandatory)
write into ISC register
set SSIE00 bit to 1, enable channel 0 slave selection function operates.
(mandatory)
write into SSm register
set SSmn bit of target channel to "1" (Semn=1, configure as operation enable state).
(mandatory)
start communication
configure transmit data to SDRmn register, wait for master device clock.
restart configuration completes.

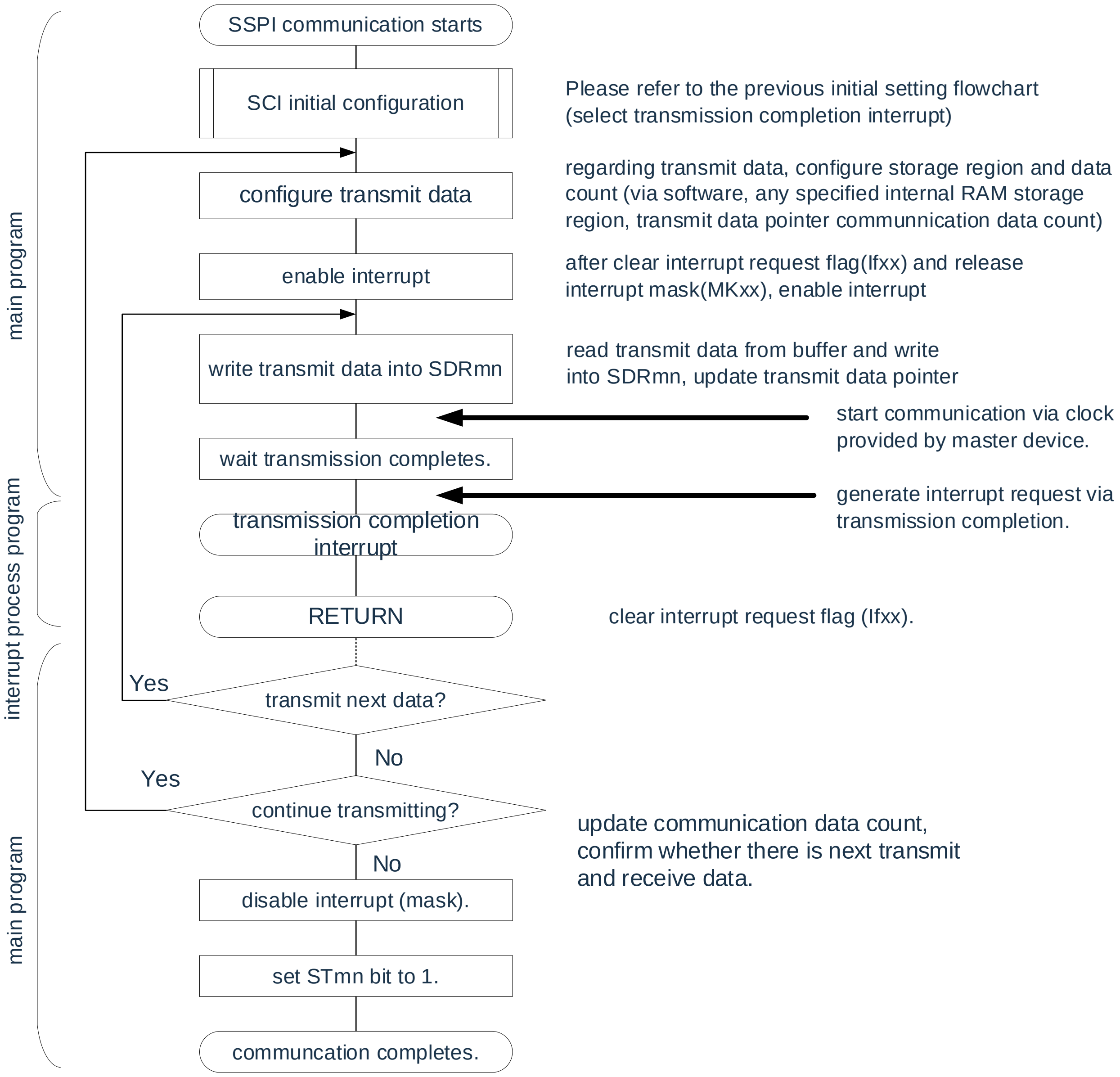
(2) Process flow (single transmit mode)

Figure 15-74 Timing of slave transmission (single transmit mode) (Type 1:DAPmn=0, CKPmn=0)



Remark: m: Unit number (m=0)n: Channel number (n=0) p:SSPI Number (p=00)

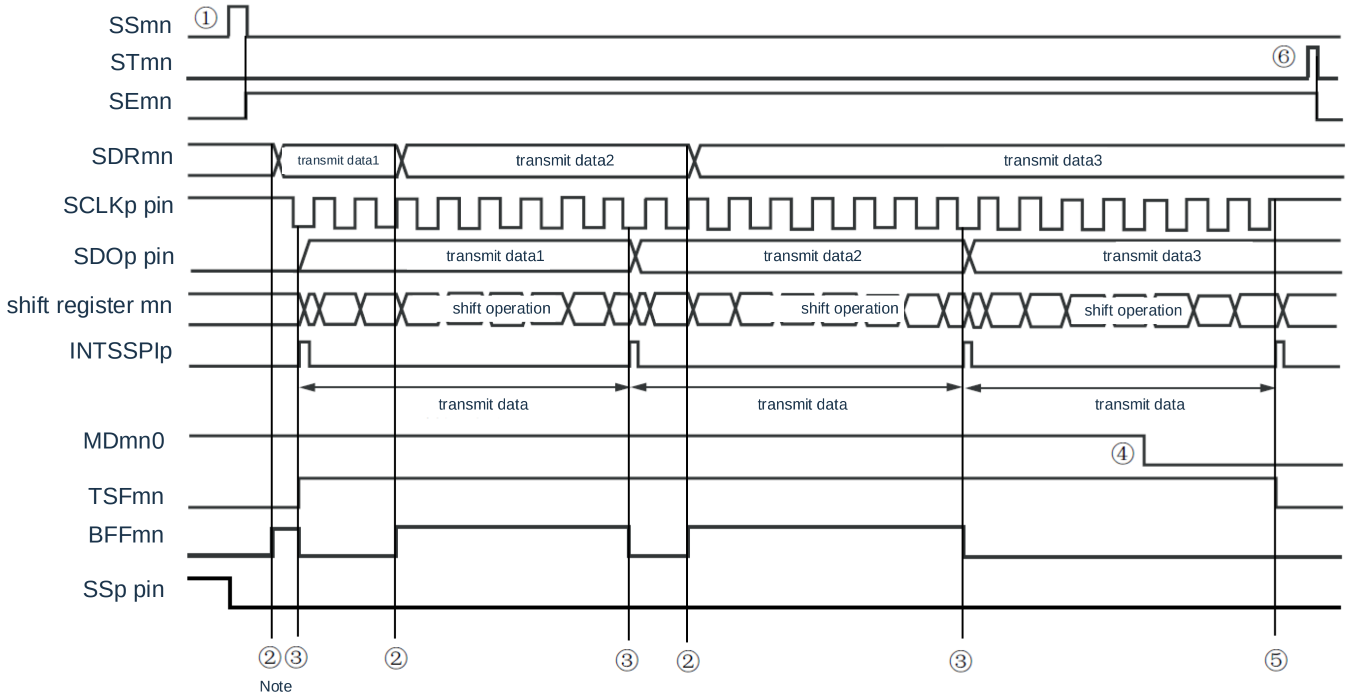
Figure 15-75 Flowchart for slave transmission (single transmit mode)


main program
interrupt process program
main program
SSPI communication starts
SCI initial configuration
Please refer to the previous initial setting flowchart (select transmission completion interrupt)
configure transmit data
regarding transmit data, configure storage region and data count (via software, any specified internal RAM storage region, transmit data pointer communnication data count)
enable interrupt
after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt
write transmit data into SDRmn
read transmit data from buffer and write into SDRmn, update transmit data pointer
start communication via clock provided by master device.
wait transmission completes.
generate interrupt request via transmission completion.
transmission completion interrupt
RETURN
clear interrupt request flag (Ifxx).
Yes
transmit next data?
No
Yes
continue transmitting?
No
update communication data count, confirm whether there is next transmit and receive data.
disable interrupt (mask).
set STmn bit to 1.
communcation completes.

Remark: m: Unit number (m=0)n: Channel number (n=0) p:SSPI Number (p=00)

(4) Process flow (continuous transmit mode)

Figure 15-76 Timing of slave transmission (continuous transmit mode)
(Type 1: DAPmn=0, CKPmn=0)

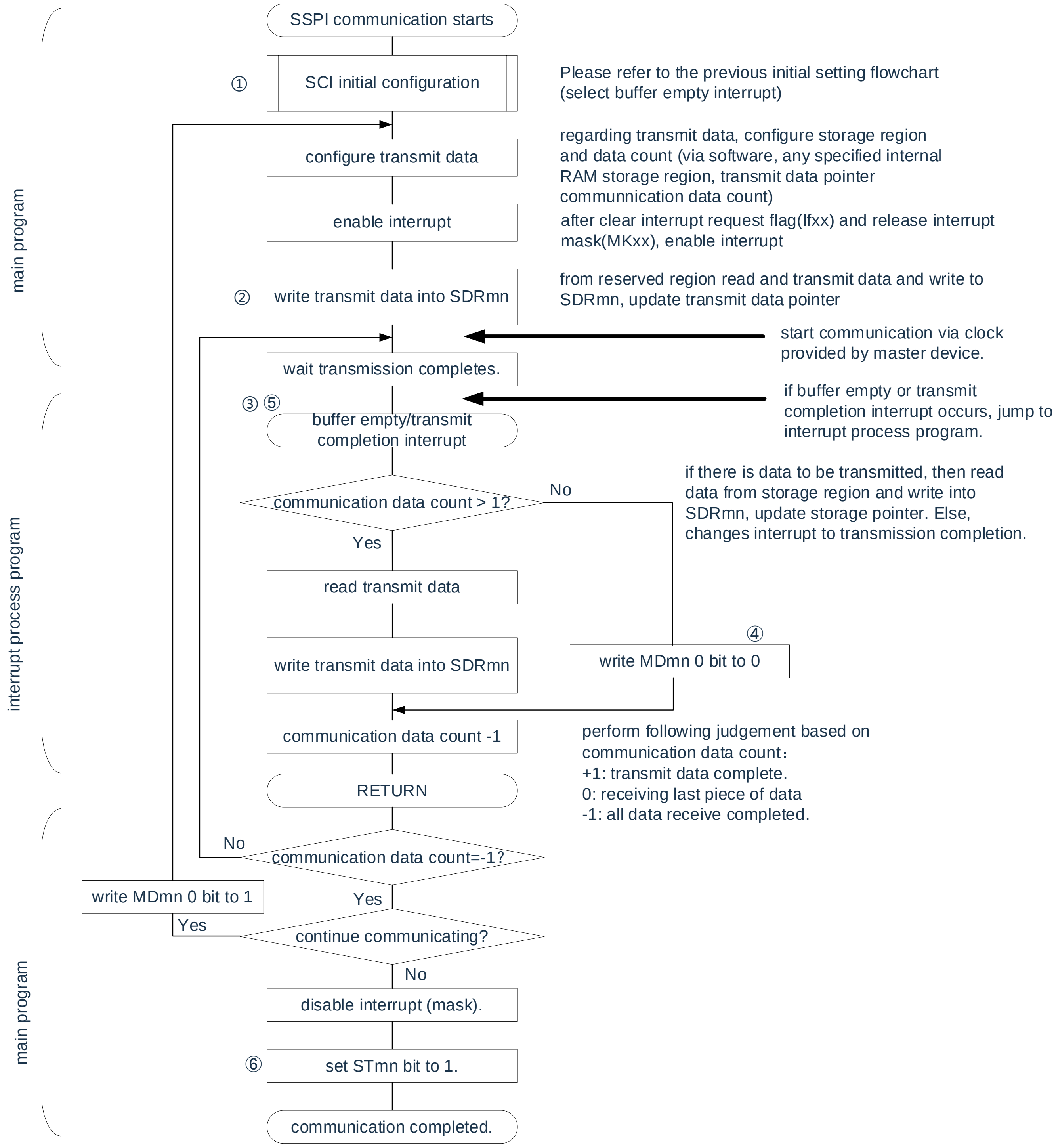


Note The transmission data is rewritten if the BFFmn bit of the serial status register mn (SSRmn) is "1" (SDRmn) when the valid data is stored in the serial data register mn (SDRmn).

Note The MDmn0 bit of the serial mode register mn(SMRmn) can be rewritten even in operation. However, you must override before you start transferring the last bit.

Remark m: Unit number (m=0)n: Channel Number (n=0)p:SSPI Number (p=00)

Figure 15-77 Flowchart for slave transmission (continuous transmit mode)


main program
SSPI communication starts
①
SCI initial configuration
Please refer to the previous initial setting flowchart
(select buffer empty interrupt)
configure transmit data
regarding transmit data, configure storage region
and data count (via software, any specified internal
RAM storage region, transmit data pointer
communnication data count)
enable interrupt
after clear interrupt request flag(Ifxx) and release interrupt
mask(MKxx), enable interrupt
②
write transmit data into SDRmn
from reserved region read and transmit data and write to
SDRmn, update transmit data pointer
start communication via clock
provided by master device.
wait transmission completes.
if buffer empty or transmit
completion interrupt occurs, jump to
interrupt process program.
interrupt process program
③ ⑤
buffer empty/transmit
completion interrupt
communication data count > 1?
No
Yes
if there is data to be transmitted, then read
data from storage region and write into
SDRmn, update storage pointer. Else,
changes interrupt to transmission completion.
read transmit data
write transmit data into SDRmn
④
write MDmn 0 bit to 0
communication data count -1
perform following judgement based on
communication data count：
+1: transmit data complete.
0: receiving last piece of data
-1: all data receive completed.
RETURN
main program
No
communication data count=-1?
write MDmn 0 bit to 1
Yes
Yes
continue communicating?
No
disable interrupt (mask).
⑥
set STmn bit to 1.
communication completed.

Note 1. ① to ⑥ in the figure corresponds to ① to ⑥ in Figure 15-76.

2.m: Unit number (m=0)n: Channel number (n=0) p:SSPI Number (p=00)

15.6.2 Slave reception

A slave reception is the operation of this product to receive data from other devices in the state of inputting transmission clock from other devices.

Slave select input function	SSPI00
Object channel	Channel 0 for SCI0
Pin used	SLK00, SDI00, SS00
Interrupt	INTSSPI00
	Interrupt at that end of the transfer only (Disable setting buffer null interrupt).
Error detection flag	Only the overflow error detection flag (OVFmn).
Length of transmit data	7 ~ 16 bits
Transfer rate	Maxf_{MCK}/6[Hz] Notes 1,2
Data phase	Can be selected by the DAPmn bit of the SCRmn register.
Clock phase	Can be selected by the CKPmn bit of the SCRmn register.
Data orientation	MSB First or LSB First
Slave select input function	You can choose to run the slave selectioninput function.

Note 1. Since the external serial clock input to the SCLK00 pin is sampled internally and used, the maximum transfer rate is f_{MCK}/6[Hz].

2. It must be used within the scope of the peripheral functional characteristics (refer to the data sheet) that meet this condition and satisfy the electrical characteristics.

Note 1.f_{MCK}: Operating clock frequency of the object channel

2.m: Unit number (m=0) n: Channel number (n=0)

(1) Register settings

Figure 15-78 Example of register setting contents for slave select input function (SSPI00) during slave reception (1/2)

(a) serial mode register mn (SMRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn 0/1	CCSmn 1	0	0	0	0	0	STSmn 0	0	SISmn0 0	1	0	0	MDmn2 0	MDmn1 0	MDmn0 0

channel n operational clock （fMCK）
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

interrupt source of channel n
0: Transmit completion interrupt
1: Buffer empty interrupt

(b) serial communication operation configuration register mn(SCRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn 0	RXEmn 1	DAPmn 0/1	CKPmn 0/1	0	EOCmn 0	PTCmn1 0	PTCmn0 0	DIRmn 0/1	0	SLCmn1 0	SLCmn0 0	DLSmn3 0/1	DLSmn2 0/1	DLSmn1 0/1	DLSmn0 0/1

data and clock phase selection (details refer to registers controlling universal serial communication unit)

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

Setting of data length
DLSmn3~0: 7-bit~16-bit data length selection

(c) serial data register mn (SDRmn)

(1) When operation stops (SEmn=0)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration （0000000B）							0	0	0	0	0	0	0	0	0

(2) During operation (SEmn=1) (Lower 8 bits: SDRmnL)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	transmit data register															
									SDRmnL							

(d) 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 ×

(e) serial output enable register m (SOEm)⋯. Not used in this mode.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	SOEm3 ×	SOEm2 ×	SOEm1 ×	SOEm0 ×

Note 1.m: Unit number (m=0)n: Channel number (n=0)p:SSPI Number (p=00)

2. □ : Set in slave receive mode for fixed. ▨: Cannot set (set initial value).

×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).

0/1: The "0" or "1" is set according to the user.

Figure 15-79 Example of register setting contents for slave select input function (SSPI00) during slave reception (2/2)

(f) serial channel start register m (SSm) …. Only set bit of target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm															SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	×	0/1

(g) Input switching control register (ISC) ...... This is the control of the SS00 pin of the SSPI00 slave channel (channel 0 of unit 0).

	7	6	5	4	3	2	1	0
ISC	SSIE00						ISC1	ISC0
	0/1	0	0	0	0	0	0/1	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. □ : Set in slave receive mode for fixed. ▨: Cannot set (set initial value).

×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).

0/1: The "0" or "1" is set according to the user.

(2) Procedure

Figure 15-80 Initial set-up steps for slave reception

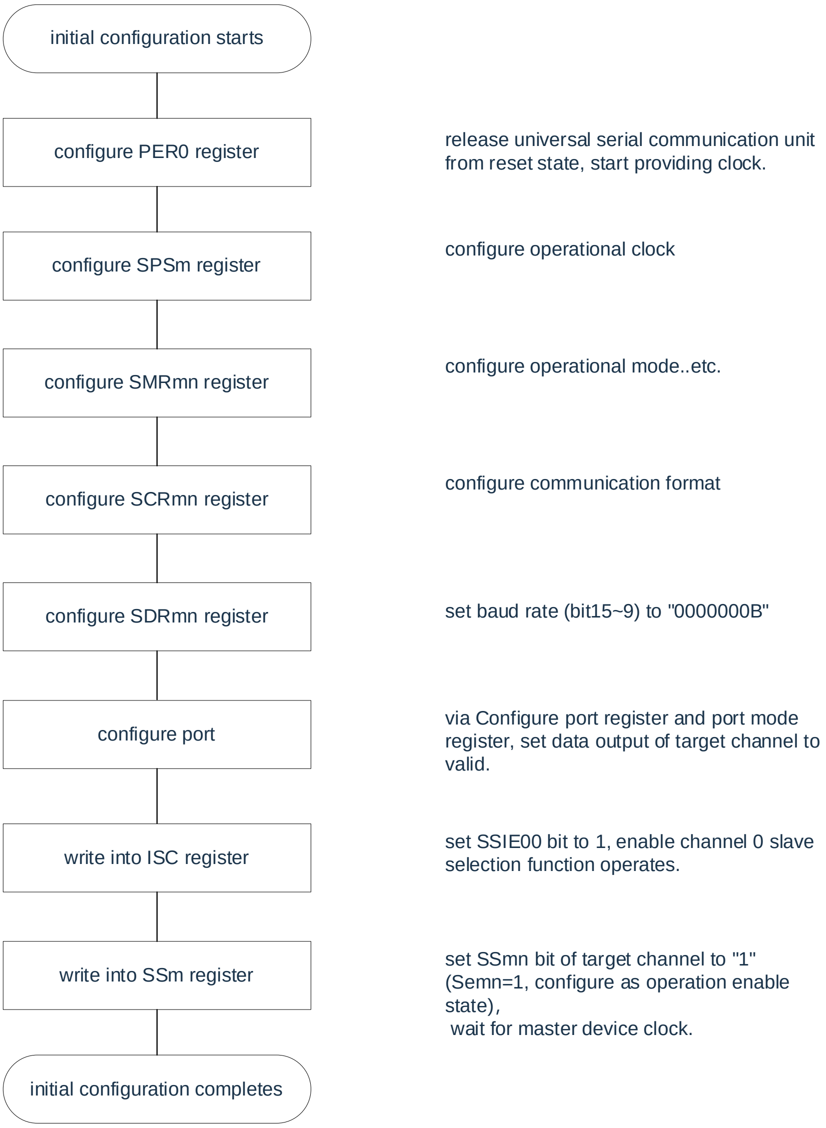
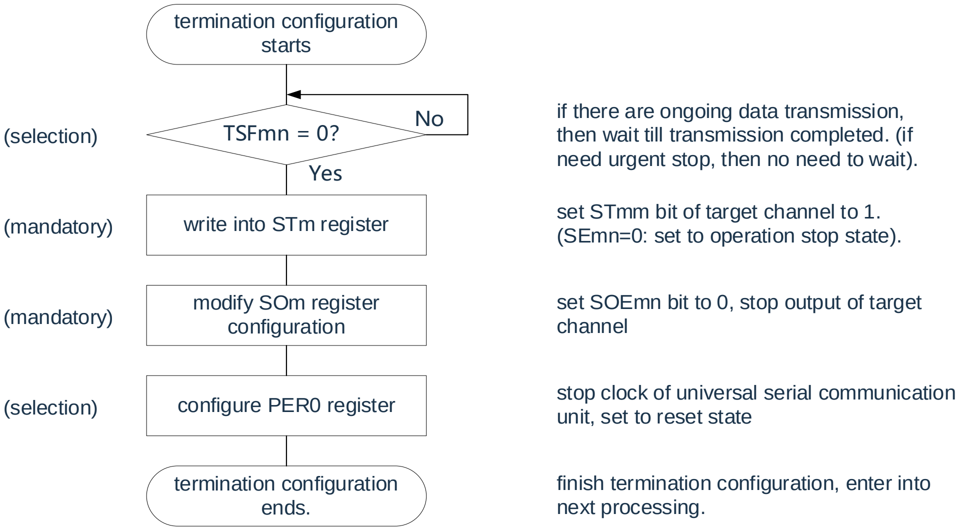
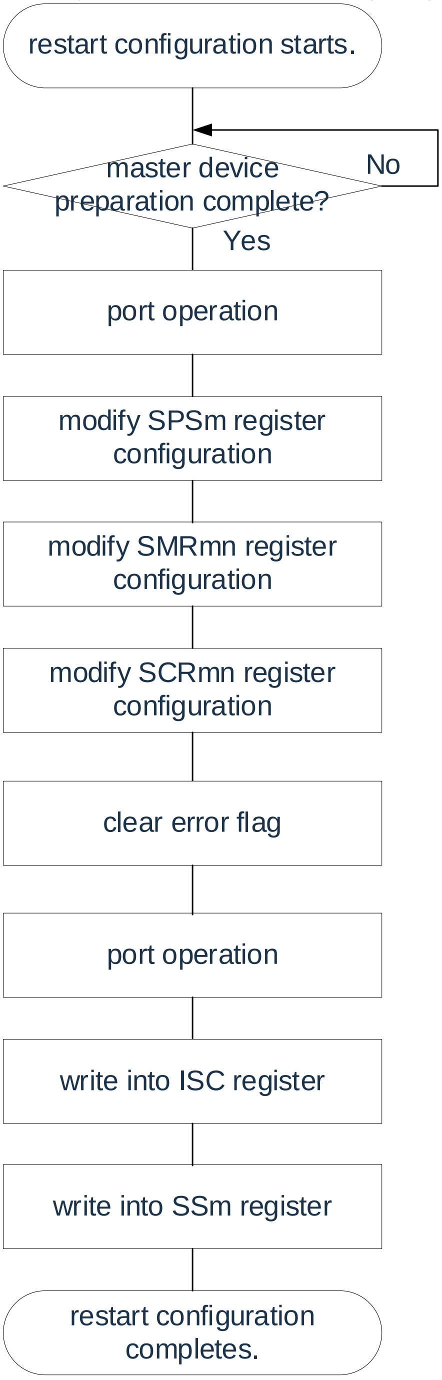

initial configuration starts
configure PER0 register
release universal serial communication unit from reset state, start providing clock.
configure SPSm register
configure operational clock
configure SMRmn register
configure operational mode..etc.
configure SCRmn register
configure communication format
configure SDRmn register
set baud rate (bit15~9) to "0000000B"
configure port
via Configure port register and port mode register, set data output of target channel to valid.
write into ISC register
set SSIE00 bit to 1, enable channel 0 slave selection function operates.
write into SSm register
set SSmn bit of target channel to "1" (Semn=1, configure as operation enable state),
wait for master device clock.
initial configuration completes

Figure 15-81 Stop steps for slave reception


termination configuration starts
(selection)
TSFmn = 0?
No
Yes
if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).
(mandatory)
write into STm register
set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).
(mandatory)
modify SOm register configuration
set SOEmn bit to 0, stop output of target channel
(selection)
configure PER0 register
stop clock of universal serial communication unit, set to reset state
termination configuration ends.
finish termination configuration, enter into next processing.

Remark m: Unit number (m=0) n: Channel number (n=0) p:SSPI Number (p=00)

Figure 15-82 Restart set-up steps for slave reception

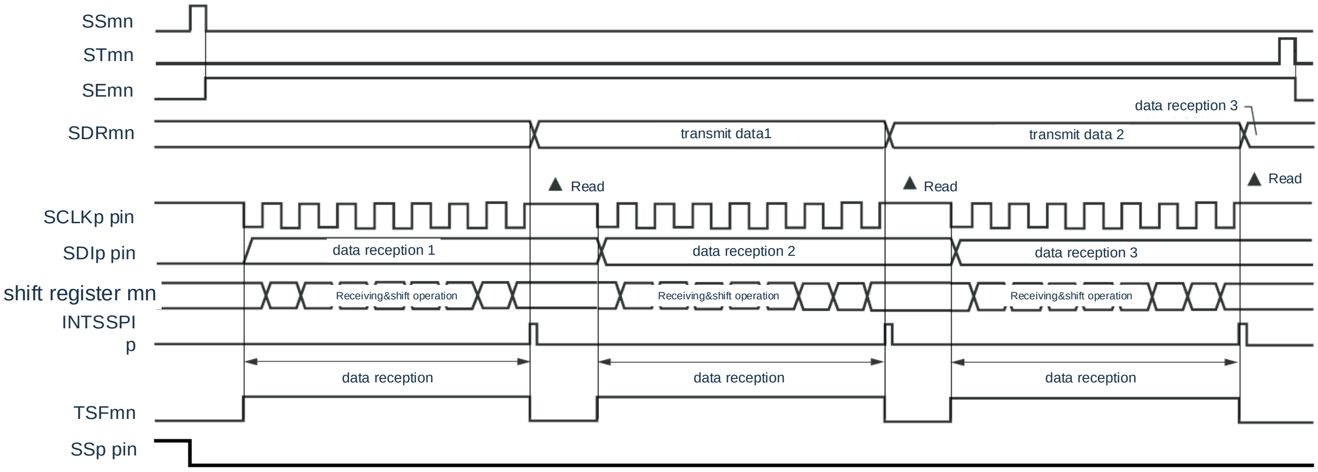

restart configuration starts.
master device preparation complete?
No
Yes
port operation
modify SPSm register configuration
modify SMRmn register configuration
modify SCRmn register configuration
clear error flag
port operation
write into ISC register
write into SSm register
restart configuration completes.

(mandatory)	master device preparation complete?	wait till commuication target (master device) stops or communication ends
(mandatory)	port operation	via Configure port register and port mode register, set clock output of target channel to invalid.
(selection)	modify SPSm register configuration	re-configure when modifing operational clock configuration
(selection)	modify SMRmn register configuration	re-configure when serial mode register mn.
(selection)	modify SCRmn register configuration	re-configure when serial communication operation configuration register mn.
(selection)	clear error flag	when OVF flag remains at set state, erase via serail flag clear trigger register mn(SIRmn).
(mandatory)	port operation	via Configure port register and port mode register, set data output of target channel to valid.
(mandatory)	write into ISC register	set SSIE00 bit to 1, enable channel 0 slave selection function operates.
(mandatory)	write into SSm register	set SSmn bit of target channel to "1" (Semn=1, configure as operation enable state), wait for master device clock.

Remark: m: Unit number (m=0) n: Channel number (n=0)p:SSPI Number (p=00)

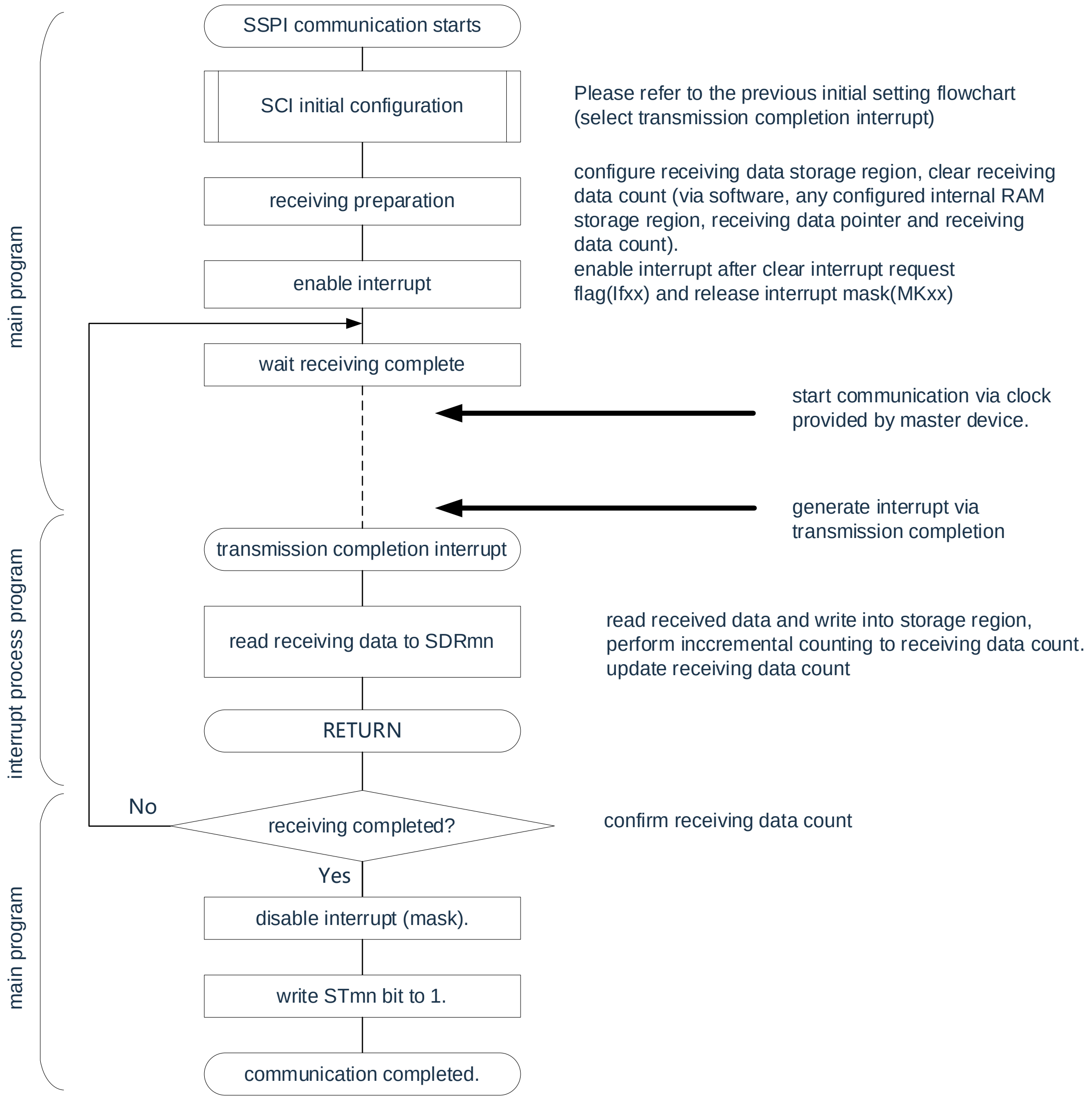
(3) Process flow (single receive mode)

Figure 15-83 Timing of slave reception (single receive mode) (Type 1: DAPmn=0, CKPmn=0)



Remark: m: Unit number (m=0)n: Channel number (n=0) p:SSPI Number (p=00)

Figure 15-84 Flow chart for slave reception (single receive mode)


main program
interrupt process program
main program
SSPI communication starts
SCI initial configuration
Please refer to the previous initial setting flowchart (select transmission completion interrupt)
receiving preparation
configure receiving data storage region, clear receiving data count (via software, any configured internal RAM storage region, receiving data pointer and receiving data count).
enable interrupt
enable interrupt after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx)
wait receiving complete
start communication via clock provided by master device.
generate interrupt via transmission completion
transmission completion interrupt
read receiving data to SDRmn
read received data and write into storage region, perform inccremental counting to receiving data count. update receiving data count
RETURN
No
receiving completed?
confirm receiving data count
Yes
disable interrupt (mask).
write STmn bit to 1.
communication completed.

15.6.3 Slave transmission and reception

Slave transmission and reception refers to the operation of sending and receiving data from this product and other devices in the state of input transfer clocks from other devices.

Slave select input function	SSPI00
Object channel	Channel 0 for SCI0
Pin used	SCLK00, SDI00, SDO00, SS00
Interrupt	INTSSPI00
	Interrupt at that end of the transfer may be selecte (single transfer mode) or buffer air-discontinuity (continuous transfer mode).
Error detection flag	Only the overflow error detection flag (OVFmn).
Length of transmit data	7 ~ 16 bits
Transfer rate	Max.f_{MCK}/6[Hz]Note 1,2
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: forward • CKPmn=1: inverted
Data orientation	MSB First or LSB First
Slave select input function	You can choose to run the slave selectioninput function.

Note1. Since the external serial clock input to the SCLK00 pin is sampled internally and used, the maximum transfer rate is f_{MCK}/6[Hz].

2. It must be used within the scope of the peripheral functional characteristics (refer to the data sheet) that meet this condition and satisfy the electrical characteristics.

Note 1.f_{MCK}: Operating clock frequency of the object channel

2.m: Unit number (m=0)n: Channel number (n=0)

(1) Register settings

Figure 15-85 Example of register setting contents for slave select input function (SSPI00) during slave transmission and reception (1/2)

(a) serial mode register mn (SMRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn 0/1	CCSmn 1	0	0	0	0	0	STSmn 0	0	SISmn0 0	1	0	0	MDmn2 0	MDmn1 0	MDmn0 0/1

channel n operational clock (fMCK)
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

interrupt source of channel n
0: Transmit completion interrupt
1: Buffer empty interrupt

(b) serial communication operation configuration registermn mn(SCRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn 1	RXEmn 1	DAPmn 0/1	CKPmn 0/1	0	EOCmn 0	PTCmn1 0	PTCmn0 0	DIRmn 0/1	0	SLCmn1 0	SLCmn0 0	DLSmn3 0/1	DLSmn2 0/1	DLSmn1 0/1	DLSmn0 0/1

data and clock phase selection (details refer to registers controlling universal serial communication unit)

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

Setting of data length DLSmn3~0: 7-bit~16-bit data length selection

(c) serial data register mn (SDRmn)

(1) When operation stops (SEmn=0)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	波特率的设定 （运行时钟（fMCK）的分频设定）							0	0	0	0	0	0	0	0	0

(2) During operation (SEmn=1) (Lower 8 bits: SDRmnL)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	transmit/receive data register															

SDRmnL

(d) serial output register m (SOm)…. Set the bits of the target channel only

	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 0/1

(e) serial output enable register m (SOEm)···. set the bit of the target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	SOEm3 ×	SOEm2 ×	SOEm1 ×	SOEm0 0/1

Note Before the master device starts outputting the clock, the SDRmn register must be set to send data.

Note 1.m: Unit number (m=0)n: Channel Number (n=0)p:SSPI Number (p=00)

2. □ : Set in slave receive mode for fixed. ▨: Cannot set (set initial value).
×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).
0/1: The "0" or "1" is set according to the user.

Figure 15-87 Example of register setting contents for slave select input function (SSPI00) during slave transmission and reception (2/2)

(f) serial channel start register m (SSm) …. Only set bit of target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm															SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	×	0/1

(g) Input switching control register (ISC) ...... This is the control of the SS00 pin of the SSPI00 slave channel (channel 0 of unit 0).

	7	6	5	4	3	2	1	0
ISC	SSIE00						ISC1	ISC0
	0/1	0	0	0	0	0	0/1	0/1

0: SS00 pin input invalid
1: SS00 pin input valid

Note Before the master device starts outputting the clock, the SDRmn register must be set to send data.

Note 1.m: Unit number (m=0)n: Channel number (n=0)p:SSPI Number (p=00)

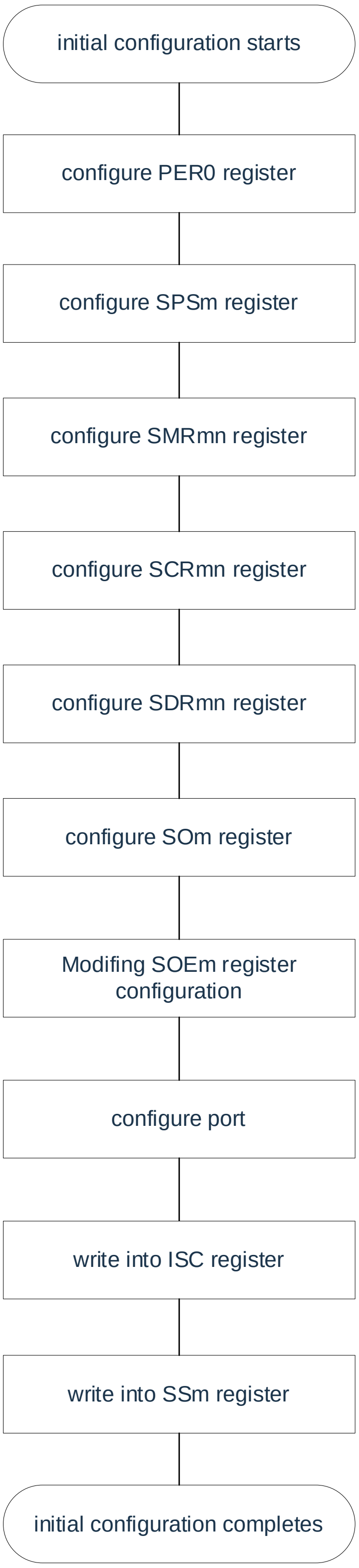
2. □ : Set in slave receive mode for fixed. ▨: Cannot set (set initial value).

×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).

0/1: The "0" or "1" is set according to the user.

(2) Procedure

Figure 15-86 Initial set-up steps for slave transmission and reception



release universal serial communication unit from reset state, start providing clock.

configure operational clock

configure operational mode..etc.

configure communication format

set baud rate (bit15~9) to "0000000B"

configure serial data (Somn) initial output voltage

set SOEmn bit to 1, enable data output of target channel

via Configure port register and port mode register, data output of target channel set to valid. When connecting to multiple slave devices, configure N-channel open drain before configure data output.

set SSIE00 bit to 1, enable channel 0 slave selection function operates.

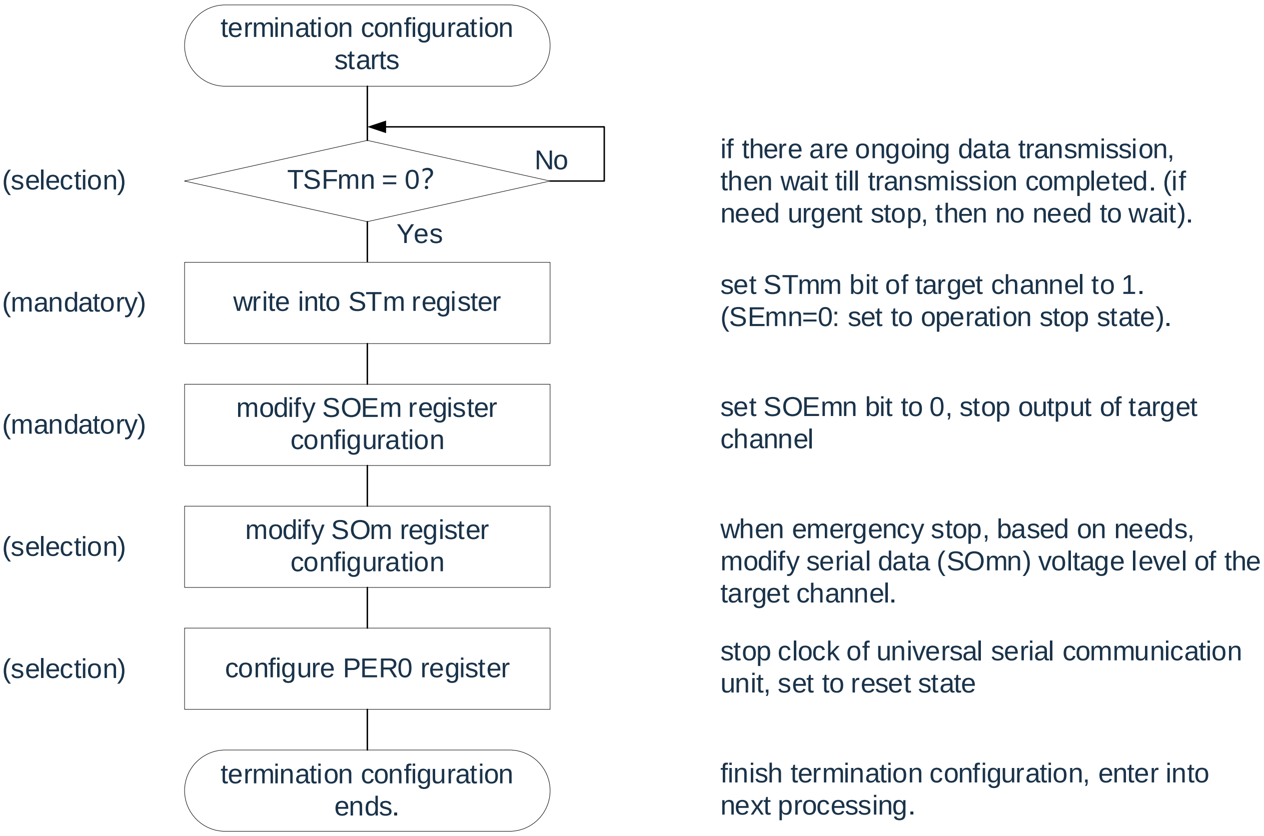
set SSmn bit of target channel to "1" (SEmn=1, configure to enable operation state).

initial configuration completes. Configure transmit data to SDRmn register, wait for master device clock.

Note Before the master device starts outputting the clock, the SDRmn register must be set to send data.

Remark m: Unit number (m=0)n: Channel Number (n=0)p:SSPI Number (p=00)

Figure 15-87 Stop steps for slave transmission and reception


termination configuration starts
(selection)
TSFmn = 0?
No
Yes
(mandatory)
write into STm register
(mandatory)
modify SOEm register configuration
(selection)
modify SOm register configuration
(selection)
configure PER0 register
termination configuration ends.

if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).

set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).

set SOEmn bit to 0, stop output of target channel

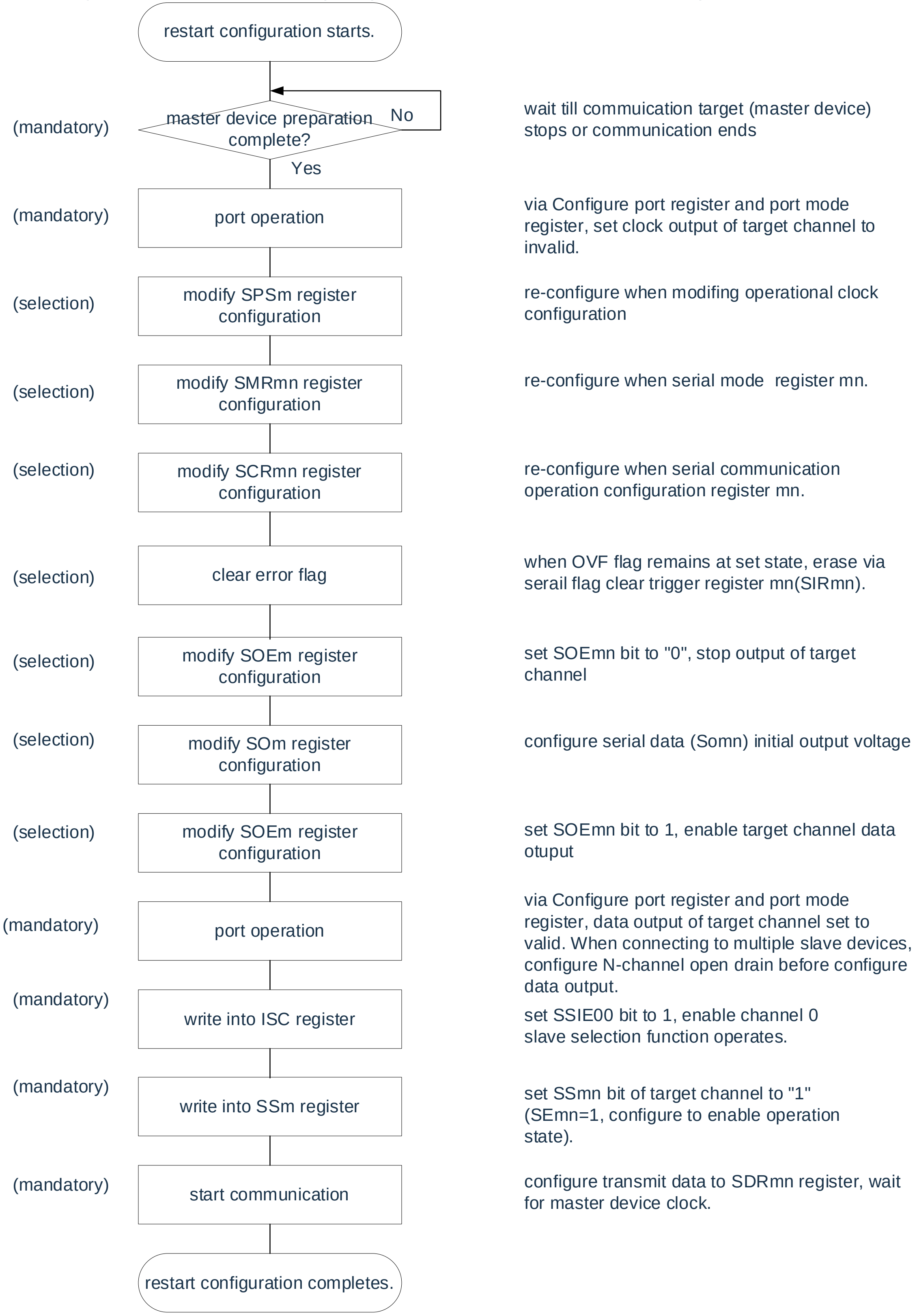
when emergency stop, based on needs, modify serial data (SOmn) voltage level of the target channel.

stop clock of universal serial communication unit, set to reset state

finish termination configuration, enter into next processing.

Note 1.m: Unit number (m=0) n: Channel number (n=0) p:SSPI Number (p=00)

Figure 15-88 Restart set-up steps for slave transmission and reception

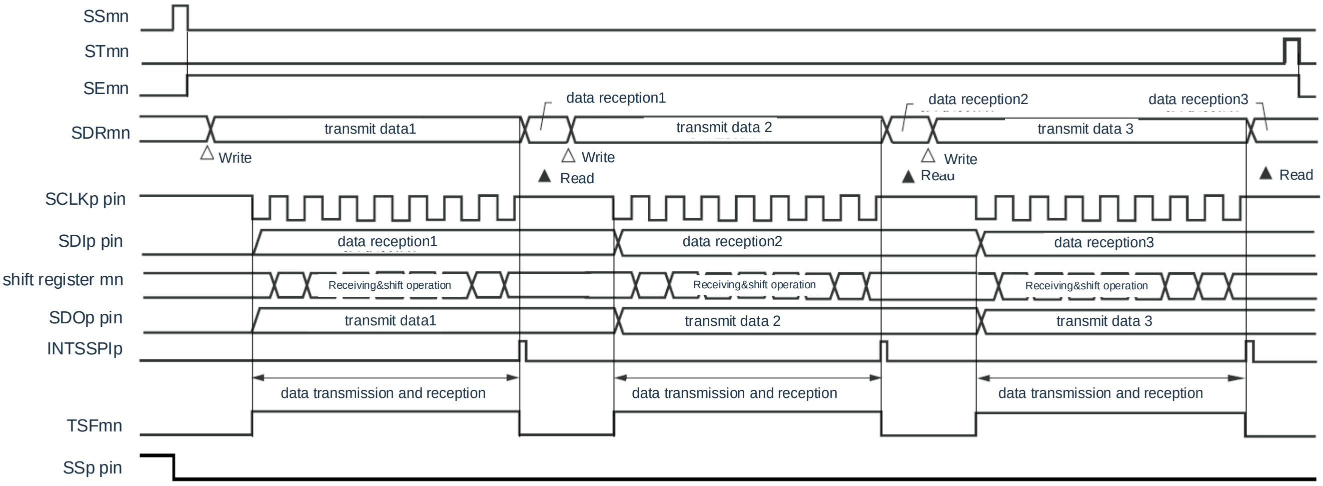

restart configuration starts.
(mandatory)
master device preparation complete?
No
Yes
wait till commuication target (master device) stops or communication ends
(mandatory)
port operation
via Configure port register and port mode register, set clock output of target channel to invalid.
(selection)
modify SPSm register configuration
re-configure when modifing operational clock configuration
(selection)
modify SMRmn register configuration
re-configure when serial mode register mn.
(selection)
modify SCRmn register configuration
re-configure when serial communication operation configuration register mn.
(selection)
clear error flag
when OVF flag remains at set state, erase via serail flag clear trigger register mn(SIRmn).
(selection)
modify SOEm register configuration
set SOEmn bit to "0", stop output of target channel
(selection)
modify SOm register configuration
configure serial data (Somn) initial output voltage
(selection)
modify SOEm register configuration
set SOEmn bit to 1, enable target channel data otuput
(mandatory)
port operation
via Configure port register and port mode register, data output of target channel set to valid. When connecting to multiple slave devices, configure N-channel open drain before configure data output.
(mandatory)
write into ISC register
set SSIE00 bit to 1, enable channel 0 slave selection function operates.
(mandatory)
write into SSm register
set SSmn bit of target channel to "1" (SEmn=1, configure to enable operation state).
(mandatory)
start communication
configure transmit data to SDRmn register, wait for master device clock.
restart configuration completes.

Note 1. The SDRmn register setting must be sent before the master device starts outputting the clock.

2. If you override PER0 in the abort setting to stop providing the clock, you must wait until the communication object (master) stops or the communication is over for the initial set-up instead of restarting the set-up.

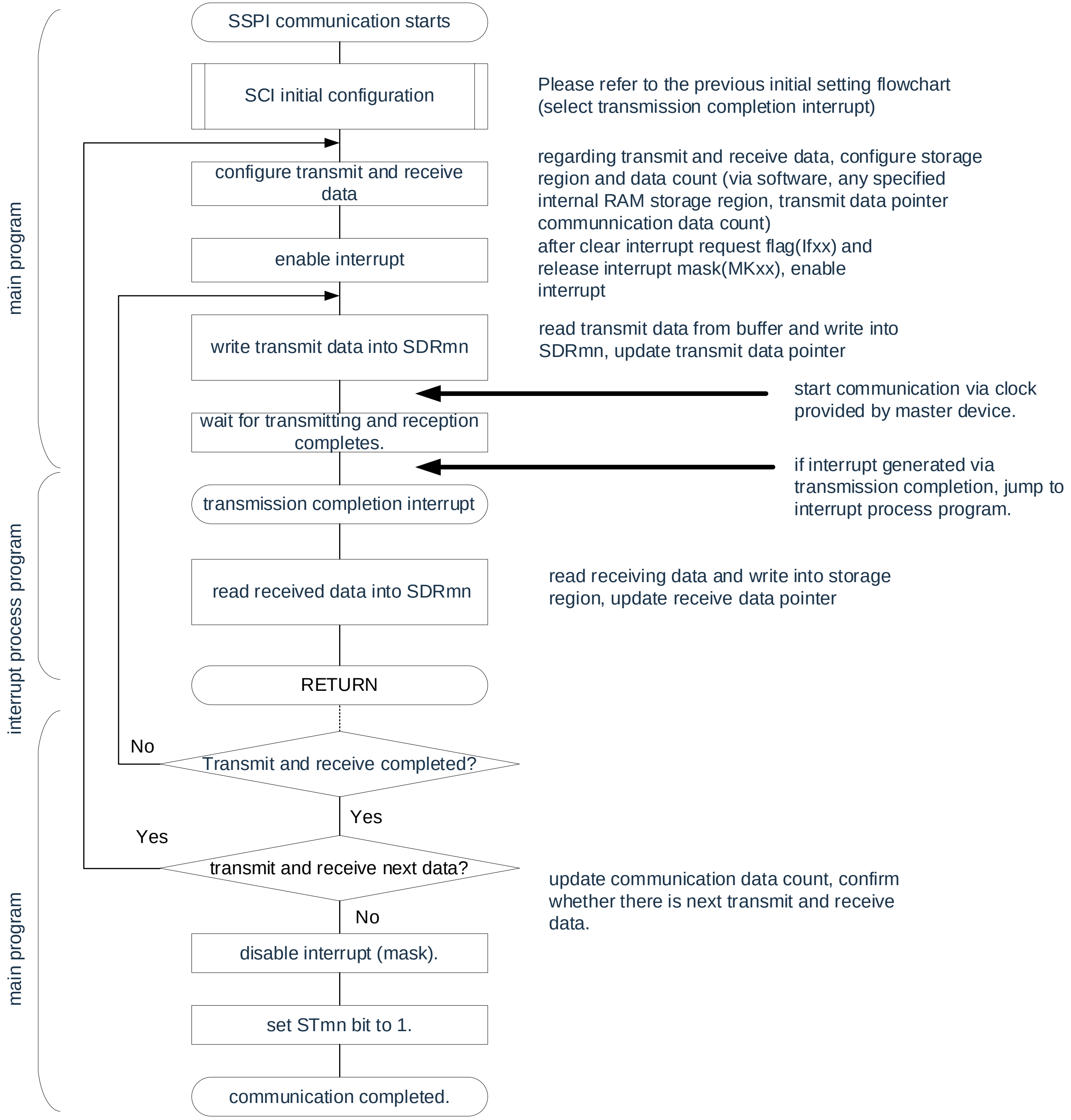
(3) Process flow (single send and receive mode)

Figure 15-89 Timing of slave transmission and reception (single transmit and receive mode)
(Type 1: DAPmn=0, CKPmn=0)



Remark: m: Unit number (m=0) n: Channel number (n=0) p:SSPI Number (p=00)

Figure 15-90 Flowchart of slave transmission and reception (single transmit and receive mode)

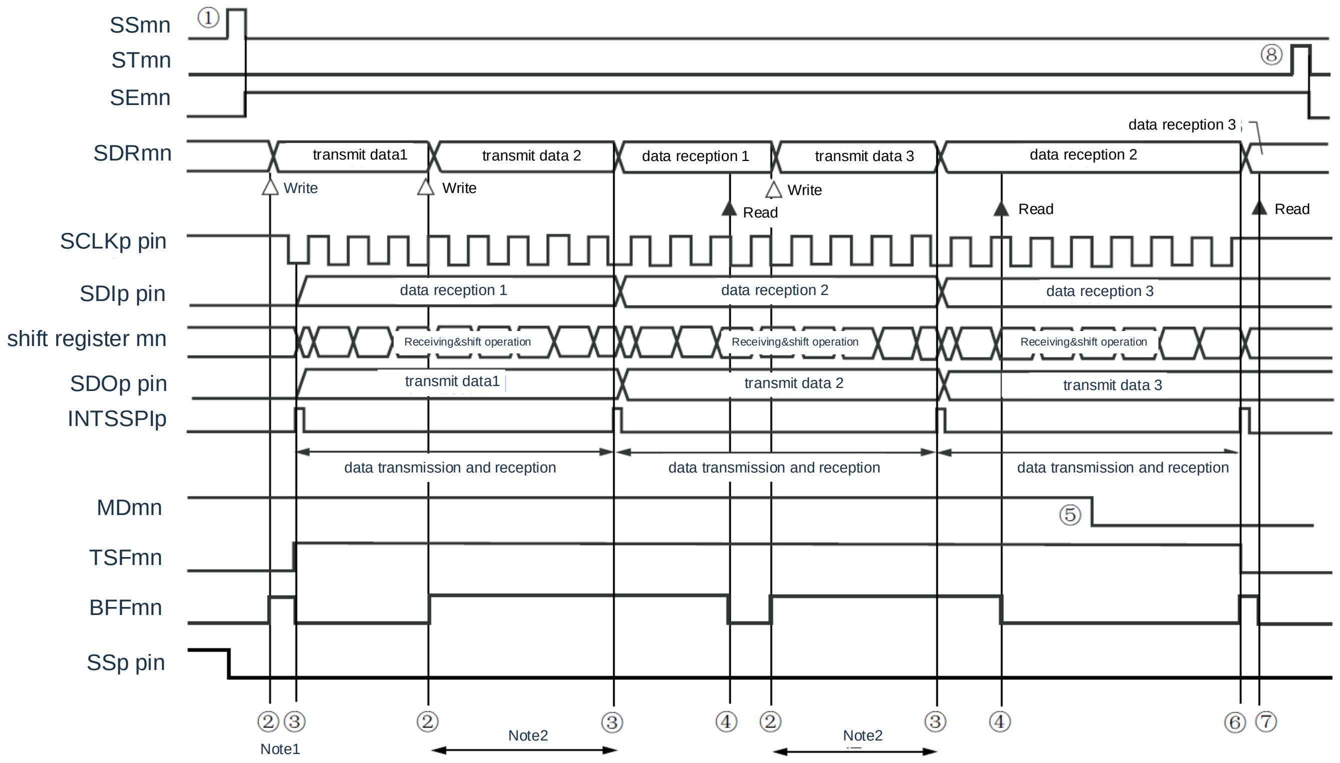

main program
interrupt process program
main program
SSPI communication starts
SCI initial configuration
Please refer to the previous initial setting flowchart (select transmission completion interrupt)
configure transmit and receive data
regarding transmit and receive data, configure storage region and data count (via software, any specified internal RAM storage region, transmit data pointer communnication data count)
enable interrupt
after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt
write transmit data into SDRmn
read transmit data from buffer and write into SDRmn, update transmit data pointer
wait for transmitting and reception completes.
start communication via clock provided by master device.
transmission completion interrupt
if interrupt generated via transmission completion, jump to interrupt process program.
read received data into SDRmn
read receiving data and write into storage region, update receive data pointer
RETURN
No
Transmit and receive completed?
Yes
Yes
transmit and receive next data?
update communication data count, confirm whether there is next transmit and receive data.
No
disable interrupt (mask).
set STmn bit to 1.
communication completed.

Notice: Before the master device starts outputting the clock, the SIOp register must be set to send data.

Remark: m: Unit number (m=0) n: Channel number (n=0) p:SSPI Number (p=00)

(4) Processing flow (continuous transmit and receive mode)

Figure 15-91 Timing of slave transmission and reception (continuous transmit and receive mode)
(Type 1: DAPmn=0, CKPmn=0)



Note: 1. Rewrite the transmission data 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)).

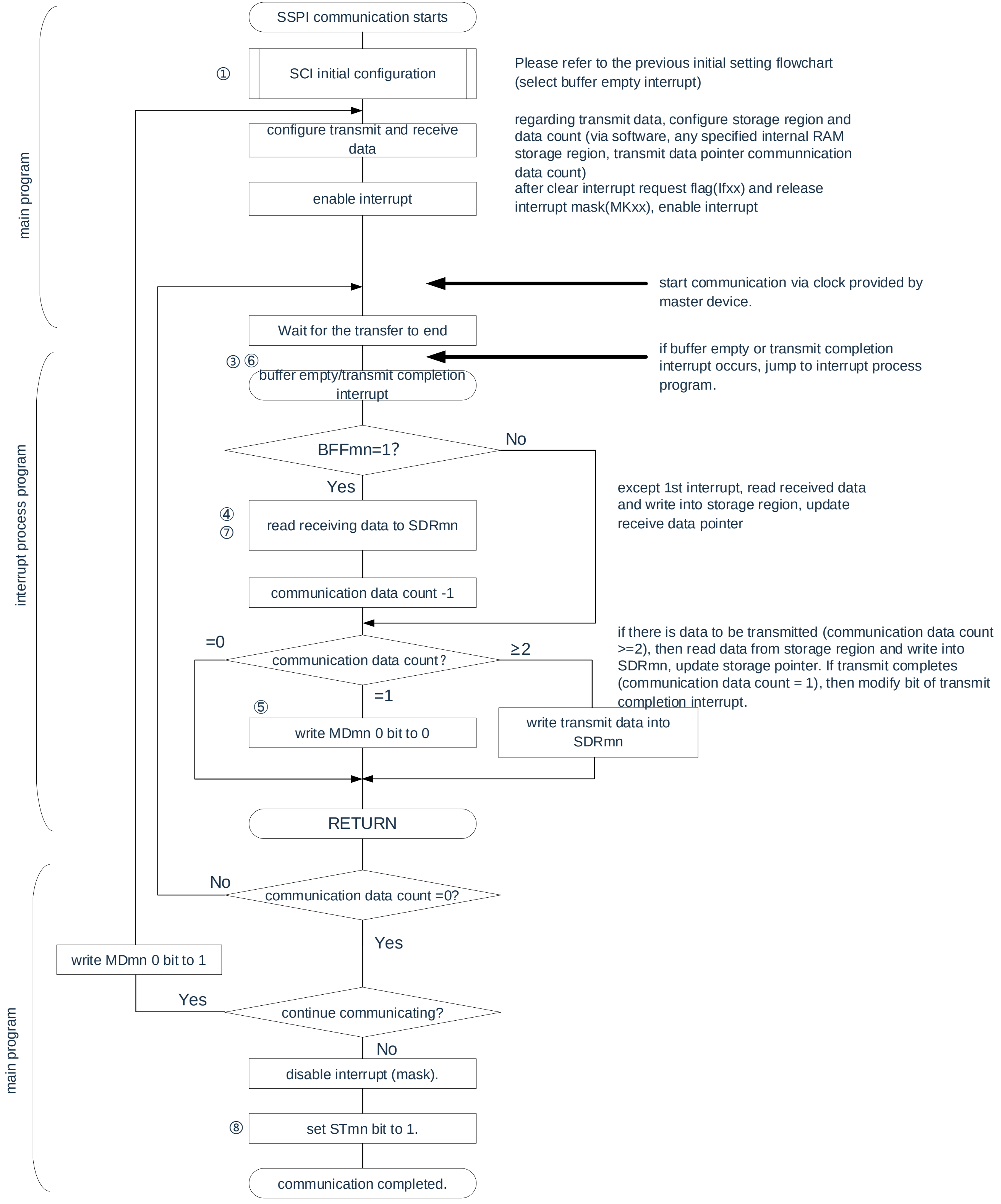
2. If the SDRmn register is read during this time, the transmit data can be read. At this time, the transmit operation is not affected.

Notice: The MDmn0 bit of the serial mode register mn(SMRmn) can be rewritten even in operation. However, in order to be able to catch the end of the transmission of the last transmitted data interrupt, it is necessary to override before starting the last transmission.

Remark: 1. ① to ⑧ in the figure corresponds to ① to ⑧ in “Figure 15-92 Flowchart of slave transmission and reception (continuous transmit and receive mode)”.

2. m: Unit number (m=0) n: Channel number (n=0) p:SSPI Number (p=00)

Figure 15-92 Flowchart of slave transmission and reception (continuous transmit and receive mode)


main program
interrupt process program
main program
SSPI communication starts
①
SCI initial configuration
configure transmit and receive data
enable interrupt
Wait for the transfer to end
③ ⑥
buffer empty/transmit completion interrupt
BFFmn=1?
No
Yes
④
⑦
read receiving data to SDRmn
communication data count -1
=0
communication data count?
≥2
=1
⑤
write MDmn 0 bit to 0
write transmit data into SDRmn
RETURN
No
communication data count =0?
Yes
write MDmn 0 bit to 1
Yes
continue communicating?
No
disable interrupt (mask).
⑧
set STmn bit to 1.
communication completed.

Please refer to the previous initial setting flowchart (select buffer empty interrupt)

regarding transmit data, configure storage region and data count (via software, any specified internal RAM storage region, transmit data pointer communnication data count)

after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt

start communication via clock provided by master device.

if buffer empty or transmit completion interrupt occurs, jump to interrupt process program.

except 1st interrupt, read received data and write into storage region, update receive data pointer

if there is data to be transmitted (communication data count >=2), then read data from storage region and write into SDRmn, update storage pointer. If transmit completes (communication data count = 1), then modify bit of transmit completion interrupt.

Notice: Before the master device starts outputting the clock, the SDRmn register must be set to send data.

Remark: 1. ① to ⑧ in the figure corresponds to ① to ⑧ in “Figure 15-91 Timing of slave transmission and reception (continuous transmit and receive mode)”.

2.m: Unit number (m=0) n: Channel number (n=0) p:SSPI Number (p=00)

15.6.4 Calculation of transmission clock frequency

The transmission clock frequency of the SSPI00 communication can be calculated using the following formula.

(1) Slave

(Transmit slock frequency)={Serial clock (SCLK) frequency provided by the master device} Note [Hz]

Note: The maximum allowable transfer clock frequency is f_{MCK}/6.

Remark: m: Unit number (m=0) n: Channel number (n=0) p:SSPI Number (p=00)

Table 15-3 Selection of the running clock of the slave select input function

SMRmn register	SPSm register								Operating Clock (f_{MCK}) Note	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		f_{CLK}=32MHz in operation
0	X	X	X	X	0	0	0	0	f_{CLK}	32MHz
	X	X	X	X	0	0	0	1	$f_{CLK}/2$	16MHz
	X	X	X	X	0	0	1	0	$f_{CLK}/2^{2}$	8MHz
	X	X	X	X	0	0	1	1	$f_{CLK}/2^{3}$	4MHz
	X	X	X	X	0	1	0	0	$f_{CLK}/2^{4}$	2MHz
	X	X	X	X	0	1	0	1	$f_{CLK}/2^{5}$	1kHz
	X	X	X	X	0	1	1	0	$f_{CLK}/2^{6}$	500kHz
	X	X	X	X	0	1	1	1	$f_{CLK}/2^{7}$	250kHz
	X	X	X	X	1	0	0	0	$f_{CLK}/2^{8}$	125kHz
	X	X	X	X	1	0	0	1	$f_{CLK}/2^{9}$	62.5kHz
	X	X	X	X	1	0	1	0	$f_{CLK}/2^{10}$	31.25kHz
	X	X	X	X	1	0	1	1	$f_{CLK}/2^{11}$	15.63kHz
	X	X	X	X	1	1	0	0	$f_{CLK}/2^{12}$	7.81kHz
	X	X	X	X	1	1	0	1	$f_{CLK}/2^{13}$	3.91kHz
	X	X	X	X	1	1	1	0	$f_{CLK}/2^{14}$	1.95kHz
	X	X	X	X	1	1	1	1	$f_{CLK}/2^{15}$	977Hz

Note: To change the clock selected as fCLK (change the value of the System Clock Control Register (CKC), you must change after stopping Universal Serial Communication Unit (SCI) =000FH.

Remark: 1.X: Ignore

2.m: Unit number (m=0) n: Channel number (n=0)

15.6.5 Processing steps when an error occurs during clock sync serial communication of slave selection input function

The processing steps when an error occurs during clock synchronization serial communication of a slave selection input function are shown in Figure 15-93.

Figure 15-93 Handling steps when overflow errors occur

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.

Remark: m: Unit number (m=0) n: Channel number (n=0)

15.7 Operation of UART(UART0~UART2) communication

This is the ability to communicate asynchronously over a total of two lines, Serial Data Send (TxD) and Serial Data Receive (RxD). The two communication lines are used to transmit and receive data asynchronously (using internal baud rate) with other communication parties in data frames (consisting of start bit, data, parity bit and stop bit). Full duplex asynchronous UART communication can be realized by using two channels dedicated for sending (even channel) and receiving (odd channel).

[Transmitting and Receiving Data]

- Data length for 7, 8, 9 or 16 bits notes
- MSB/LSB First Choice
- Level setting for transmitting and receiving data (select whether level is inverted)
- Additional, parity functions for parity bits
- Additional Stop Bit and Detection Function of Stop Bit

[Interrupt Function]

- Interrupt transmission end, buffer null interrupt
- Error interrupts due to frame errors, parity errors, and overflow errors

[Error Detection Flag]

- frame error, parity error, overflow error

UART0 uses channel 0 and channel 1 of SCI0.

UART1 uses channel 0 and channel 1 of SCI1.

UART2 uses channel 0 and channel 1 of SCI2.

Each channel selects one of the functions to use, and other functions cannot run except the selected function.

For example, you cannot use SSPI00 and IIC01 when UART0 is used in Channel 0 and Channel 1 of Cell.

Note When used as a UART, the sender (even channel) and receiver (odd channel) can only be used for UART.

UART has the following two communications operation methods:

- UART Transmission (Refer to 15.7.1)
- UART Reception (Refer to 15.7.2)

15.7.1 UART transmission

UART send is an operation where this product microcontroller asynchronously sends data to other devices.
The even of the 2 channels used by UART are for UART transmission.

UART	UART0	UART1	UART2
Object channel	Channel 0 for SCI0	Channel 2 for SCI0	Channel 0 for SCI1
Pin used	TxD0	TxD1	TxD2
Interrupt	INTST0	INTST1	INTST2
	Interrupt at that end of the transfer may be selecte (single transfer mode) or buffer air-discontinuity (continuous transfer mode).		
Error detection flag	None		
Length of transmit data	7-bit, 8-bit, 9-bit,or 16-bit Note 1		
Transfer rate	Max.f_{MCK}/6[bps] (SDRmn[15:9]≥3), Min.f_{CLK}/(2×2^{11}×128) [bps] Note		
Data phase	Forward output (default: High level). Inverted output (default: Low level).		
Parity bit	You can select the following: •No parity bits. •Additional zero check. •Additional even check. •Additional odd check.		
Stop bit	You can select the following: •One additional digits. •Two additional digits.		
Data orientation	MSB First or LSB First		

Note: It must be used within the scope of the peripheral functional characteristics (refer to the data sheet) that meet this condition and satisfy the electrical characteristics.

Note 1.f_{MCK}: Operating clock frequency of the object channel
f_{CLK}: system clock frequency
2.m: Unit number (m=0,1,2) n: Channel number (n=0)

(1) Register settings

Figure 15-94 Register setting contents when UART is transmitted by UART (UART0~UART2) (1/2)

(a) serial mode register mn (SMRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn 0/1	CCSmn 0	0	0	0	0	0	0	0	0	1	0	0	MDmn2 0	MDmn1 1	MDmn0 0/1

channel n operational clock (fMCK)
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

channel n interrupt source
0: Transmit completion interrupt
1: Buffer empty interrupt

(b) serial communication operation configuration register mn (SCRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn 1	RXEmn 0	DAPmn 0	CKPmn 0	0	EOCmn 0	PTCmn1 0/1	PTCmn0 0/1	DIRmn 0/1	0	SLCmn1 0/1	SLCmn0 0/1	DLSmn3 0/1	DLSmn2 0/1	DLSmn1 0/1	DLSmn0 0/1

parity check bit configuration
00B: no parity check
01B: add zero parity
10B: add even parity
11B: add odd parity

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

stop bit configuration
01B: appending 1 bit
10B: appending 2 bits

(c) Serial data register mn (SDRmn)

(1) When operation stops（SEmn=0）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration							0	0	0	0	0	0	0	0	0

(2) During operation（SEmn=1）（lower 8 bits：SDRmnL）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	transmit data register															

SDRmnL

(d) serial output voltage register m (SOLm) …. Only configure bit of target channel.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOLm	0	0	0	0	0	0	0	0	0	0	0	0	0	SOLm2 0/1	0	SOLm0 0/1

0: positive phase (normal) transmit
1: inverted phase transmit

Remark: 1.m: Unit number (m=0,1,2) n: Channel number (n=0) q: UART Number (q=0~2)

2.□ : Fixed in UART send mode. ▨ : Cannot set (set initial value).

×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).

0/1: The "0" or "1" is set according to the user.

Figure 15-94 Register setting contents when UART is transmitted by UART (UART0~UART2) (2/2)

(e) serial output register m (SOm)…. Only configure bit of target channel

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm	0	0	0	0	0	0	CKOm1 ×	CKOm0 ×	0	0	0	0	0	0	SOm1 ×	SOm0 0/1 注

0: serial data output value as "0"
1: serial data output value as "1"

(f) serial output enable register m (SOEm)…. Only set bit of target channel to "1".

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOEm1 ×	SOEm0 0/1

(g) serial channel start register m (SSm) …. Only set bit of target channel to "1".

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SSm1 ×	SSm0 0/1

Note: Before starting sending, '1' must be set when the SOLmn bit of the corresponding channel is '0'; When the SOLmn bit of the corresponding channel is '1', '0' must be set. During communication, the value changes due to communication data.

Remark: 1.m: Unit number (m=0,1,2) n: Channel number (n=0) q: UART Number (q=0~2)

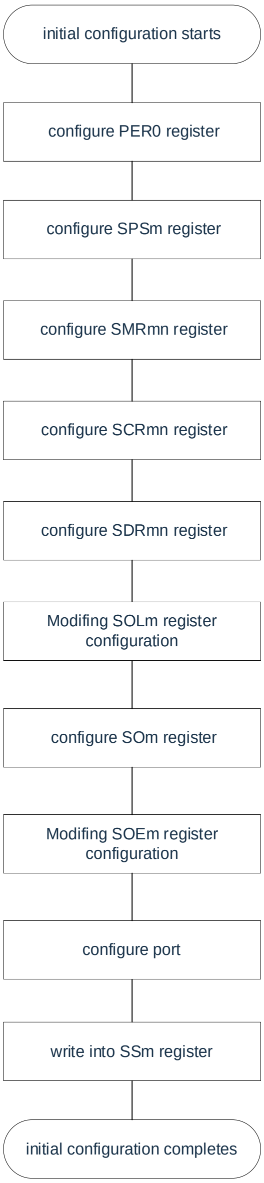
2.□ : Fixed in UART send mode. ▨ : Cannot set (set initial value).

×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).

0/1: The "0" or "1" is set according to the user.

(2) Procedure

Figure 15-95 Initial set-up steps for UART transmission



release universal serial communication unit from reset state, start providing clock.

configure operational clock

configure operational mode..etc.

configure communication format

configure transmit baud rate (configure operationl clock(fMCK) scaled transmission clock)

configure output data voltage level

configure serial data (SOmn) initial output voltage

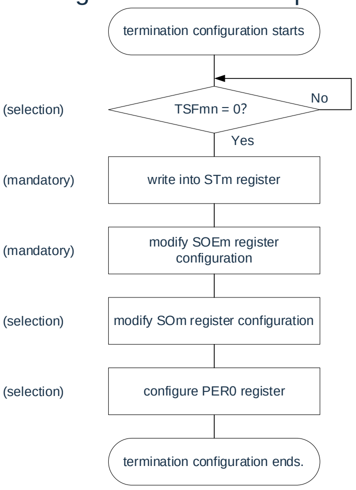
set SOEmn bit to "1", allow data output of target channel.

via Configure port register and port mode register, set data output of target channel set to valid.

set SSmn bit of target channel to "1" (Semn=1: set to operation enable state).

initial configuration completes.
Configure transmit data to SDRmn register, start communication.

Figure 15-96 Stop Steps for UART transmission


termination configuration starts
(selection)
TSFmn = 0?
No
Yes
(mandatory)
write into STm register
(mandatory)
modify SOEm register configuration
(selection)
modify SOm register configuration
(selection)
configure PER0 register
termination configuration ends.

if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).

set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).

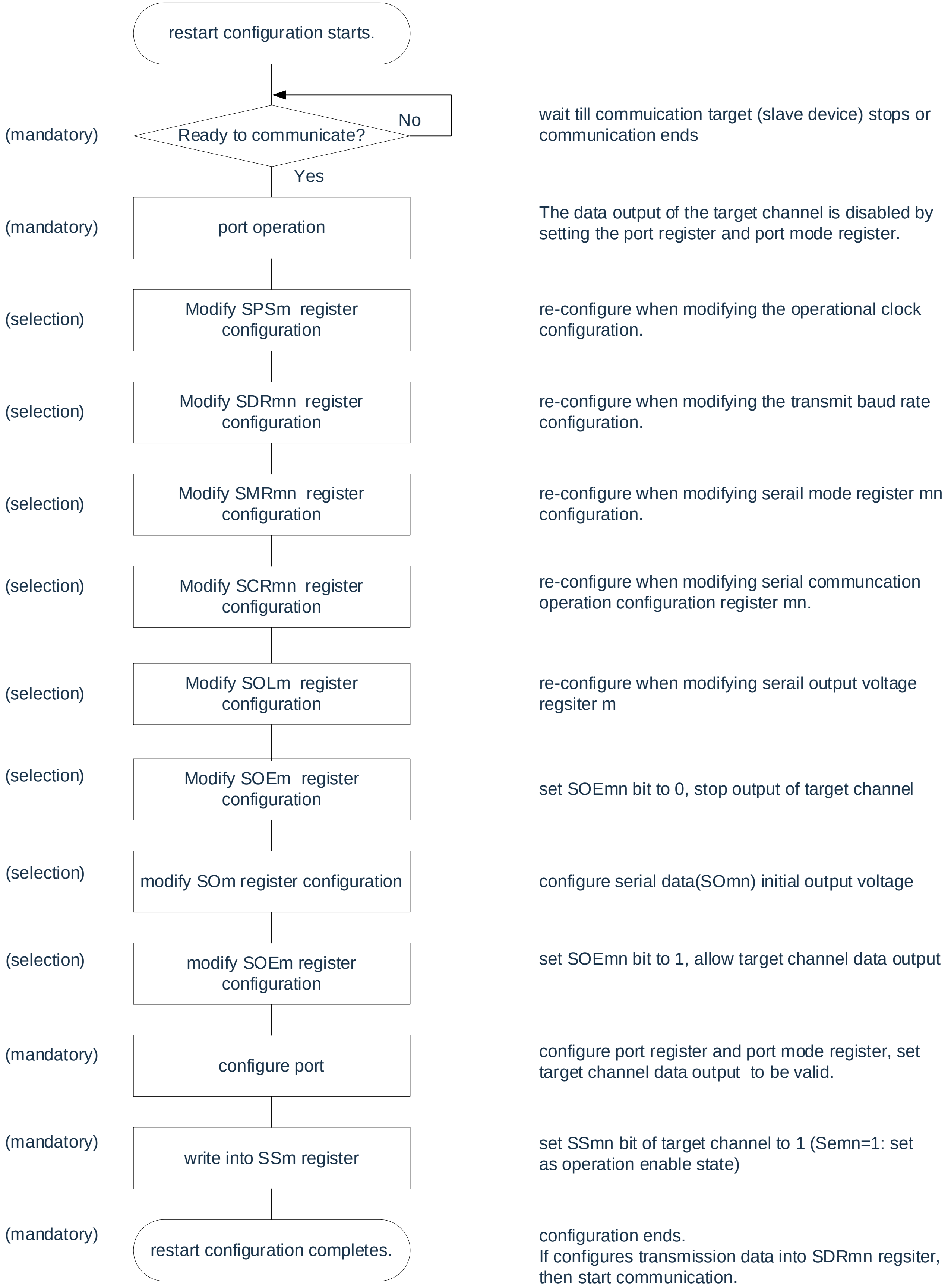
set SOEmn bit to 0, stop output of target channel

while emergency stop, based on needs, modifyserial data(Somn) voltage of target channel.

stop clock of univeral serial communication unit, set to reset state.

termination configuration completes, enter into next processing.

Figure 15-97 Restart set-up steps for UART transmission


restart configuration starts.
(mandatory)
Ready to communicate?
No
Yes
(mandatory)
port operation
(selection)
Modify SPSm register configuration
(selection)
Modify SDRmn register configuration
(selection)
Modify SMRmn register configuration
(selection)
Modify SCRmn register configuration
(selection)
Modify SOLm register configuration
(selection)
Modify SOEm register configuration
(selection)
modify SOm register configuration
(selection)
modify SOEm register configuration
(mandatory)
configure port
(mandatory)
write into SSm register
(mandatory)
restart configuration completes.

wait till commuication target (slave device) stops or communication ends

The data output of the target channel is disabled by setting the port register and port mode register.

re-configure when modifying the operational clock configuration.

re-configure when modifying the transmit baud rate configuration.

re-configure when modifying serail mode register mn configuration.

re-configure when modifying serial communcation operation configuration register mn.

re-configure when modifying serail output voltage regsiter m

set SOEmn bit to 0, stop output of target channel

configure serial data(SOmn) initial output voltage

set SOEmn bit to 1, allow target channel data output

configure port register and port mode register, set target channel data output to be valid.

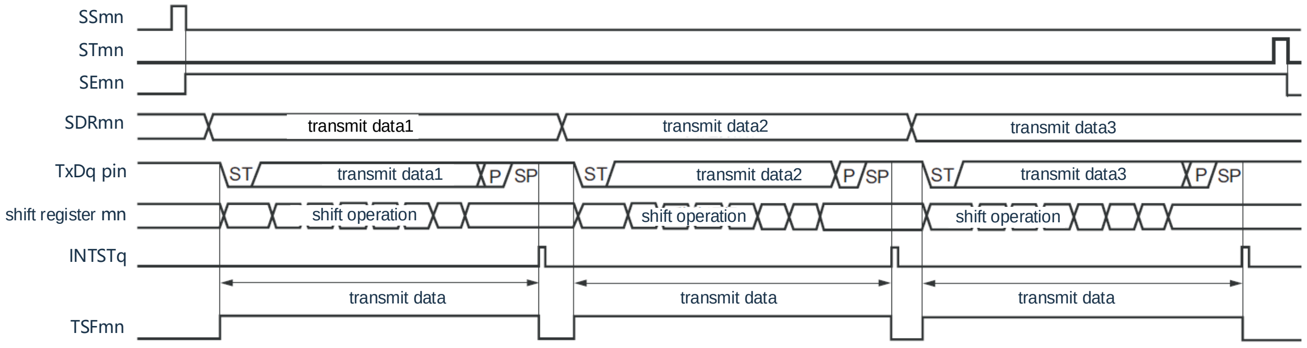
set SSmn bit of target channel to 1 (Semn=1: set as operation enable state)

configuration ends.
If configures transmission data into SDRmn regsiter, then start communication.

Remark: If that PER0 is override in the abort setting to stop the supply clock, the initial setting must wait until the communication object is stopped or communication is complete, rather than reset.

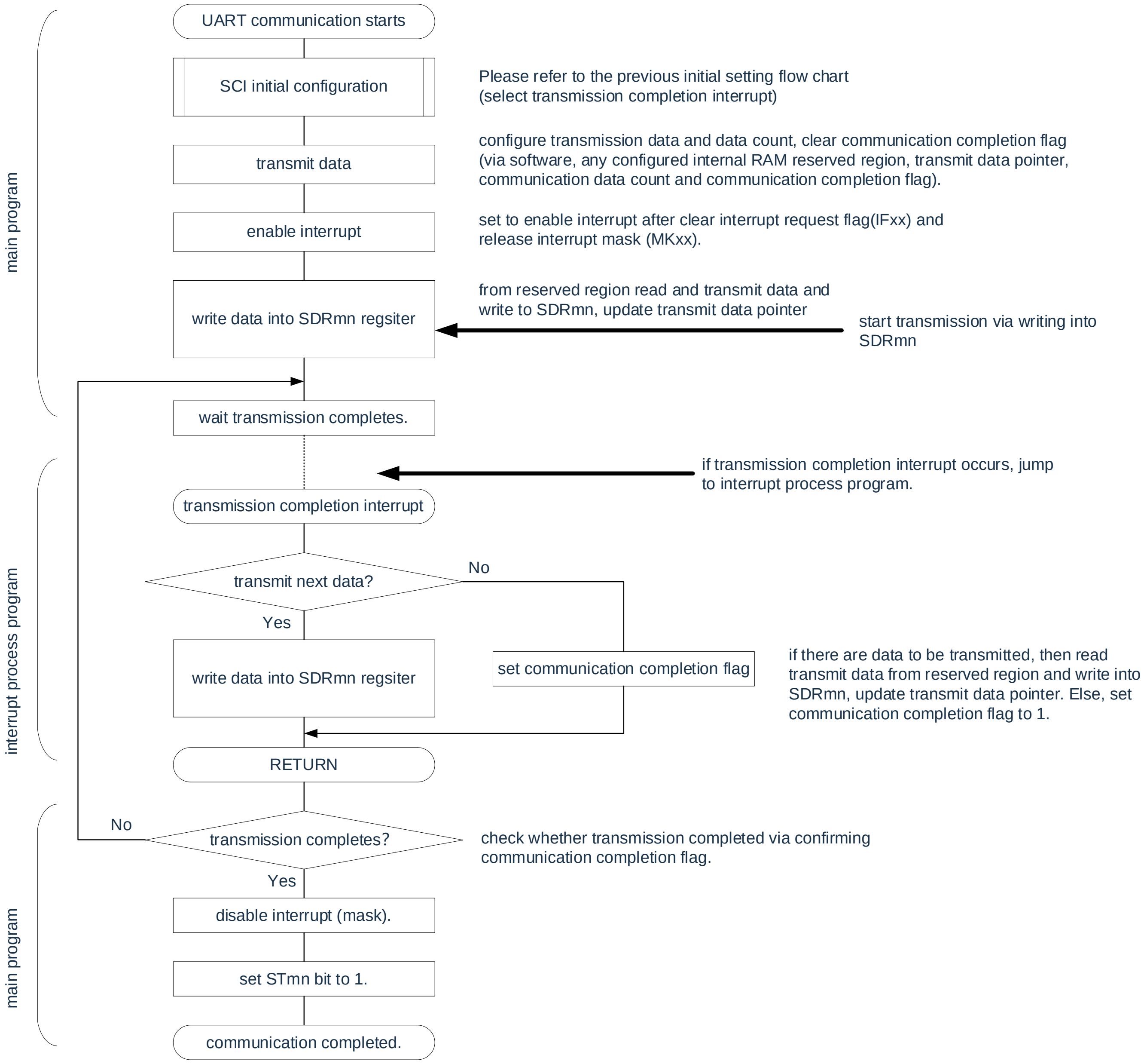
(3) Process flow (single transmit mode)

Figure 15-98 Timing of UART transmission (single transmit mode)



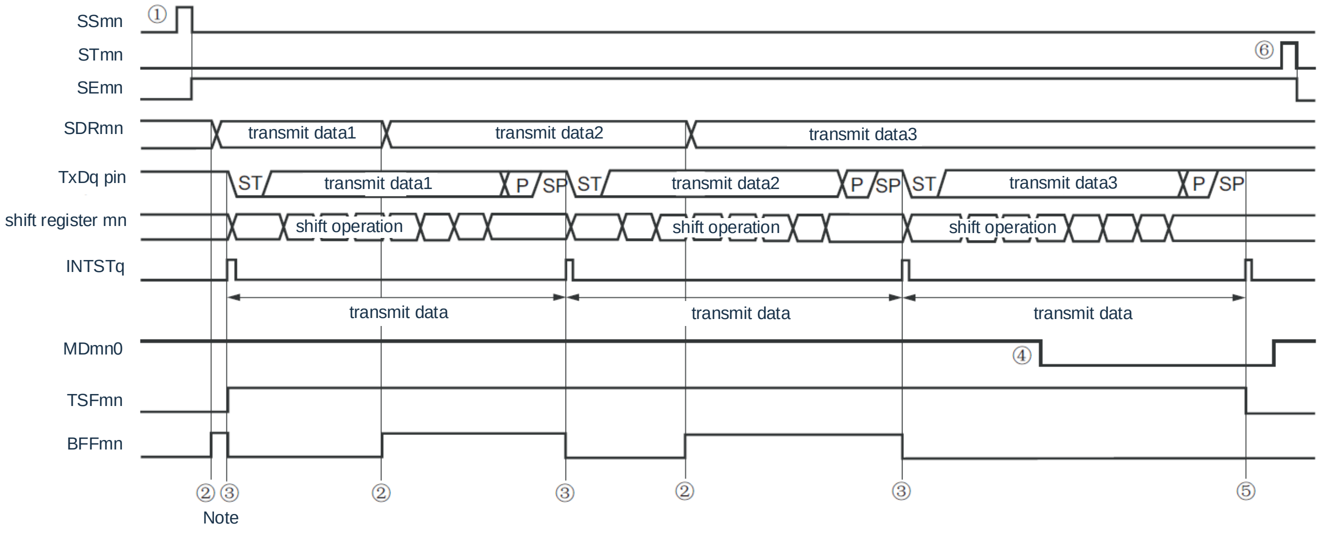
Remark: m: Unit number (m=0,1,2) n: Channel number (n=0) q: UART Number (q=0~2)

Figure 15-99 Flowchart of UART transmission (single transmit mode)


main program
UART communication starts
SCI initial configuration
Please refer to the previous initial setting flow chart (select transmission completion interrupt)
transmit data
configure transmission data and data count, clear communication completion flag (via software, any configured internal RAM reserved region, transmit data pointer, communication data count and communication completion flag).
enable interrupt
set to enable interrupt after clear interrupt request flag(IFxx) and release interrupt mask (MKxx).
write data into SDRmn regsiter
from reserved region read and transmit data and write to SDRmn, update transmit data pointer
start transmission via writing into SDRmn
wait transmission completes.
interrupt process program
if transmission completion interrupt occurs, jump to interrupt process program.
transmission completion interrupt
transmit next data?
No
Yes
write data into SDRmn regsiter
set communication completion flag
if there are data to be transmitted, then read transmit data from reserved region and write into SDRmn, update transmit data pointer. Else, set communication completion flag to 1.
RETURN
main program
No
transmission completes?
check whether transmission completed via confirming communication completion flag.
Yes
disable interrupt (mask).
set STmn bit to 1.
communication completed.

(4) Process flow (continuous transmit mode)

Figure 15-100 Timing of UART transmission (continuous transmit mode)

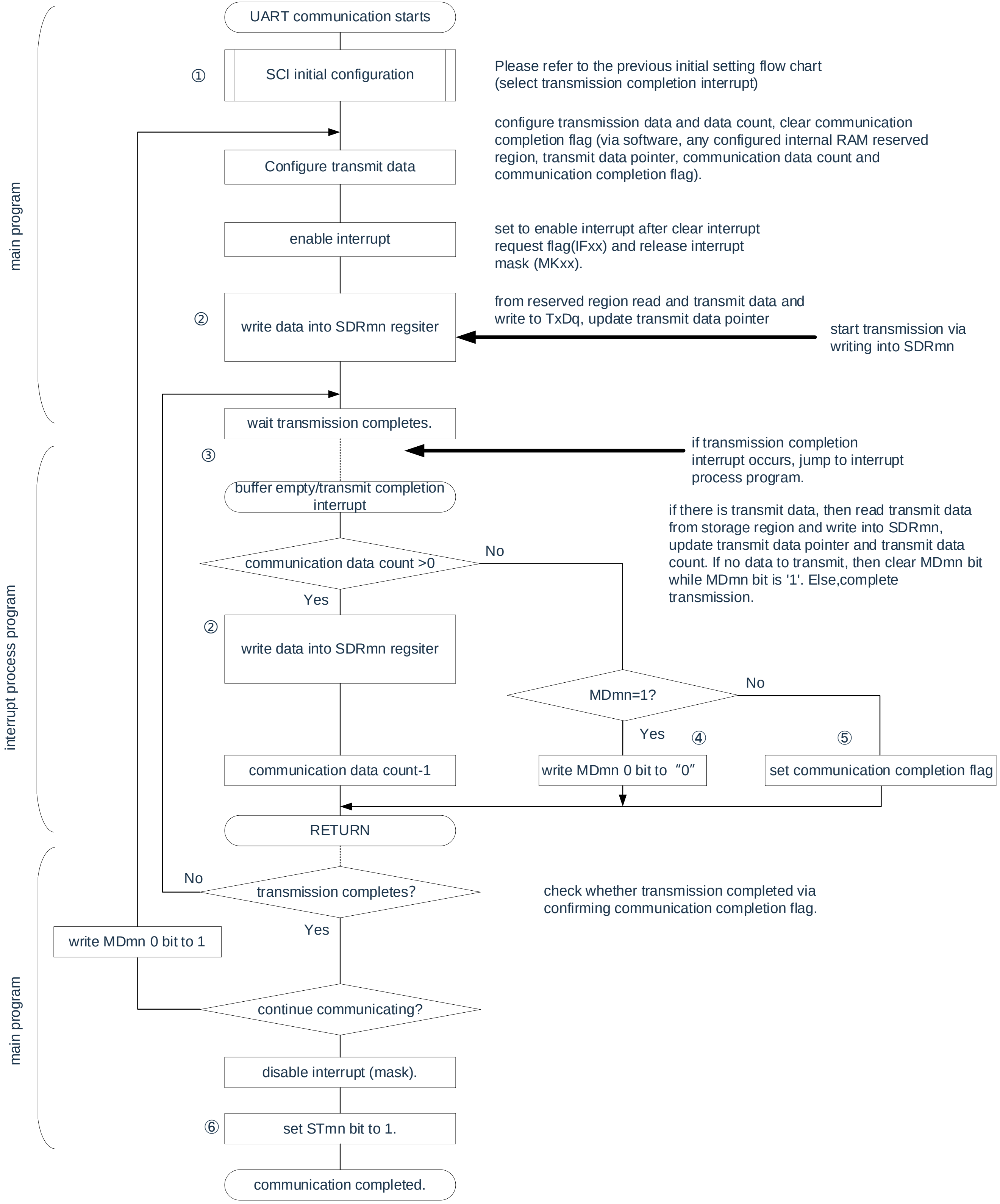


Note: The transmission data is rewritten if the BFFmn bit of the serial status register mn (SSRmn) is "1" (SDRmn) when the valid data is stored in the serial data register mn (SDRmn).

Notice: The MDmn0 bit of the serial mode register mn(SMRmn) can be rewritten even in operation. However, in order to be able to catch the end of the transmission of the last transmitted data interrupt, it is necessary to override before starting the last transmission.

Remark: m: Unit number (m=0,1,2) n: Channel Number (n=0) q:UART Number (q=0~2)

Figure 15-101 Flowchart for UART transmission (continuous transmit mode)


main program
UART communication starts
①
SCI initial configuration
Please refer to the previous initial setting flow chart (select transmission completion interrupt)
Configure transmit data
configure transmission data and data count, clear communication completion flag (via software, any configured internal RAM reserved region, transmit data pointer, communication data count and communication completion flag).
enable interrupt
set to enable interrupt after clear interrupt request flag(IFxx) and release interrupt mask (MKxx).
②
write data into SDRmn regsiter
from reserved region read and transmit data and write to TxDq, update transmit data pointer
start transmission via writing into SDRmn
wait transmission completes.
③
if transmission completion interrupt occurs, jump to interrupt process program.
interrupt process program
buffer empty/transmit completion interrupt
communication data count >0
No
Yes
if there is transmit data, then read transmit data from storage region and write into SDRmn, update transmit data pointer and transmit data count. If no data to transmit, then clear MDmn bit while MDmn bit is '1'. Else,complete transmission.
②
write data into SDRmn regsiter
MDmn=1?
No
Yes
④
⑤
communication data count-1
write MDmn 0 bit to “0”
set communication completion flag
RETURN
main program
No
transmission completes?
check whether transmission completed via confirming communication completion flag.
Yes
write MDmn 0 bit to 1
continue communicating?
disable interrupt (mask).
⑥
set STmn bit to 1.
communication completed.

Remark: ① to ⑥ in the figure corresponds to ① to ⑥ in “Figure 15-100 Timing of UART transmission (continuous transmit mode)”.

15.7.2 UART reception

UART reception is an operation in which other devices of this product microcontroller receive data asynchronously.

The odd channel of the 2 channels used by UART is used for UART reception. However, it is necessary to set the SMR registers for odd and even channels.

UART	UART0	UART1	UART2
Object channel	Channel 1 for SCI0	Channel 3 for SCI0	Channel 1 for SCI1
Pin used	RxD0	RxD1	RxD2
Interrupt	INTSR0	INTSR1	INTSR2
	Interrupt at that end of the transfer only (Disable from setting buffer null interrupt).		
Error interrupt	INTSRE0	INTSRE1	INTSRE2
Error detection flag	•Frame Error Detection Flag (FEFmn) •Parity error detection flag (PEFmn) •Overflow Error Detection Flag (OVFmn)		
Length of transmit data	7-bit, 8-bit, 9-bit or 16-bit		
Transfer rate	Max.$f_{MCK}/6$[bps] (SDRmn[15:9]≥2), Min.$f_{MCK}/(2\times2^{15}\times128)$ [bps]		
Data phase	Forward output (default: High level). Inverted output (default: Low level).		
Parity bit	You can select the following: •No parity bits (no parity). •Appending zero-check (no parity). •Even check •Odd check		
Stop bit	Appending 1-bit.		
Data orientation	MSB First or LSB First		

Note: It must be used within the scope of the peripheral functional characteristics (refer to the data sheet) that meet this condition and satisfy the electrical characteristics.

Note 1. .f_{MCK}: Operating clock frequency of the object channel
.f_{MCK}: System clock frequency
2.m: Unit number (m=0,1,2) n: Channel number (n=1)

(1) Register settings

Figure 15-102 Register setting contents when UART is received by UART (UART0~UART2) (1/2)

(a) serial mode register mn (SMRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn 0/1	CCSmn 0	0	0	0	0	0	STSmn 1	0	SISmn0 0/1	1	0	0	MDmn2 0	MDmn1 1	MDmn0 0

channel n operational clock (fMCK)
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

0: normal receiving
1: inverted phase receiving

channel N operational mode:
0: transmit completion interrupt

(b) serial mode register mr(SMRmr)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmr	CKSmr 0/1	CCSmr 0	0	0	0	0	0	0	0	0	1	0	0	MDmr2 0	MDmr1 1	MDmr0 0/1

same configuration as CKSmn bit

通道r的运行模式
0：传送结束中断
1：缓冲器空中断

(c) serial communication operation configuration register mn(SCRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn 0	RXEmn 1	DAPmn 0	CKPmn 0	0	EOCmn 1	PTCmn1 0/1	PTCmn0 0/1	DIRmn 0/1	0	SLCmn1 0	SLCmn0 0	DLSmn3 0/1	DLSmn3 0/1	DLSmn3 0/1	DLSmn3 0/1

parity check bit configuration
00B: no parity check
01B: add zero parity
10B: add even parity
11B: add odd parity

data transmit sequence selection
0: perform MSB first input/output
1: perform LSB first input/output

data length configuration

(d) serial data regsiter mn (SDRmn)

(1) When operation stops（SEmn=0）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration							0	0	0	0	0	0	0	0	0

(2) During operation（SEmn=1）（lower 8 bits：SDRmnL）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	received data register															
									SDRmnL							

Note When UART receives, you must also set the SMRmr register for channel r that is paired with channel n.

Note 1.m: Unit number (m=0,1,2) n: Channel Number (n=1)

r: Channel number (r=n~1)q:UART number (q=0~2)

2. ▣ : Fixed in UART receive mode. ▨ : Cannot set (set initial value).

×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).

0/1: The "0" or "1" is set according to the user.

Figure 15-102 Register setting contents when UART is received by UART (UART0~UART2) (2/2)

(e) serial output register m (SOm)…. Not used in this mode.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm	0	0	0	0	0	0	CKOm1 ×	CKOm0 ×	0	0	0	0	0	0	SOm1 ×	SOm0 ×

(f) serial output enable register m (SOEm)…. Not used in this mode.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOEm1 ×	SOEm0 ×

(g) serial channel start register m (SSm) …. Only set bit of target channel to "1".

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SSm1 0/1	SSm0 ×

Remark: 1.m: Unit number (m=0,1,2)

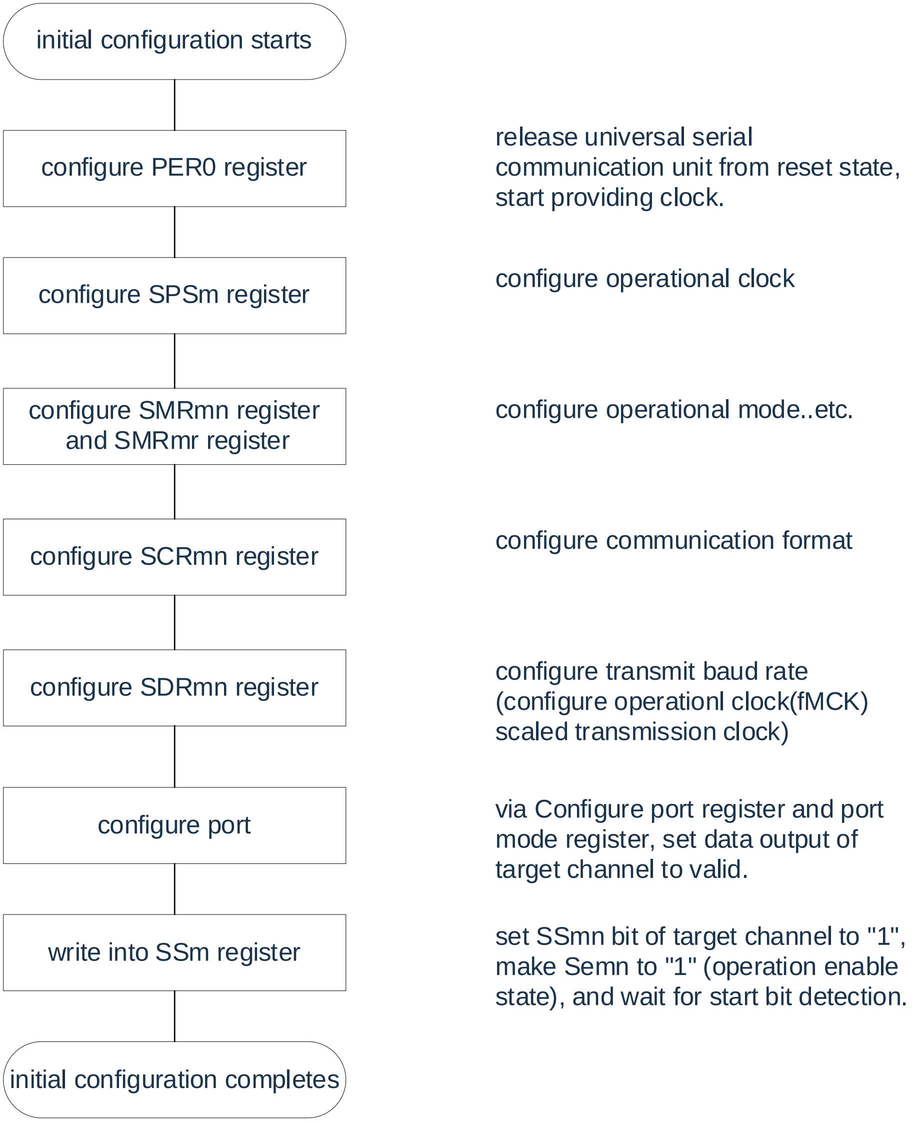
2. □ : Fixed in UART receive mode. ▨ : Cannot set (set initial value).

×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).

0/1: The "0" or "1" is set according to the user.

(2) Procedure

Figure 15-103 Initial set-up steps for UART reception


initial configuration starts
configure PER0 register
release universal serial communication unit from reset state, start providing clock.
configure SPSm register
configure operational clock
configure SMRmn register and SMRmr register
configure operational mode..etc.
configure SCRmn register
configure communication format
configure SDRmn register
configure transmit baud rate (configure operationl clock(fMCK) scaled transmission clock)
configure port
via Configure port register and port mode register, set data output of target channel to valid.
write into SSm register
set SSmn bit of target channel to "1", make Semn to "1" (operation enable state), and wait for start bit detection.
initial configuration completes

Note At least 4 f_{MCK} clocks must be set after setting the RXEmn bit of the SCRmn register to "1", and then setting the SSmn bit to "1".

Figure 15-104 Stop steps for UART reception

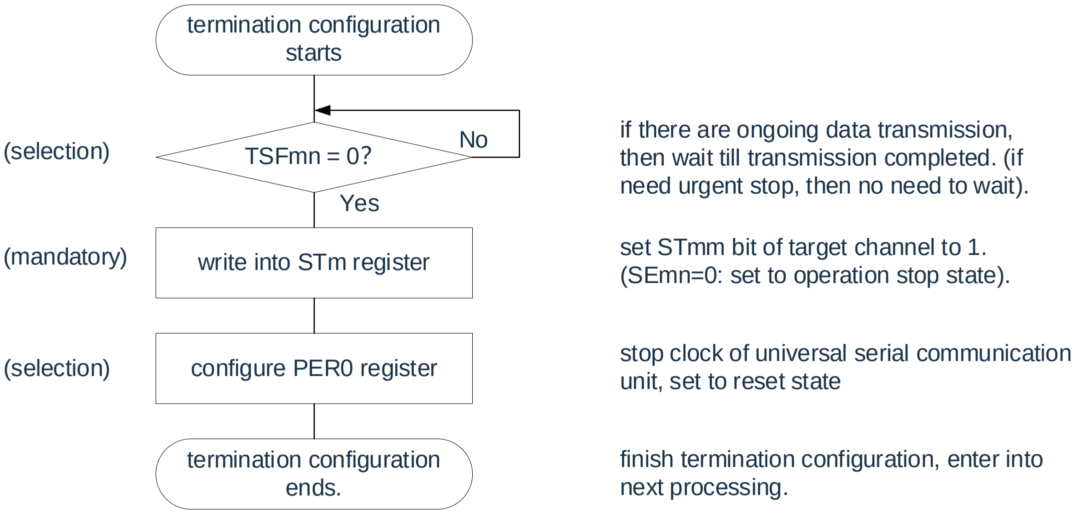
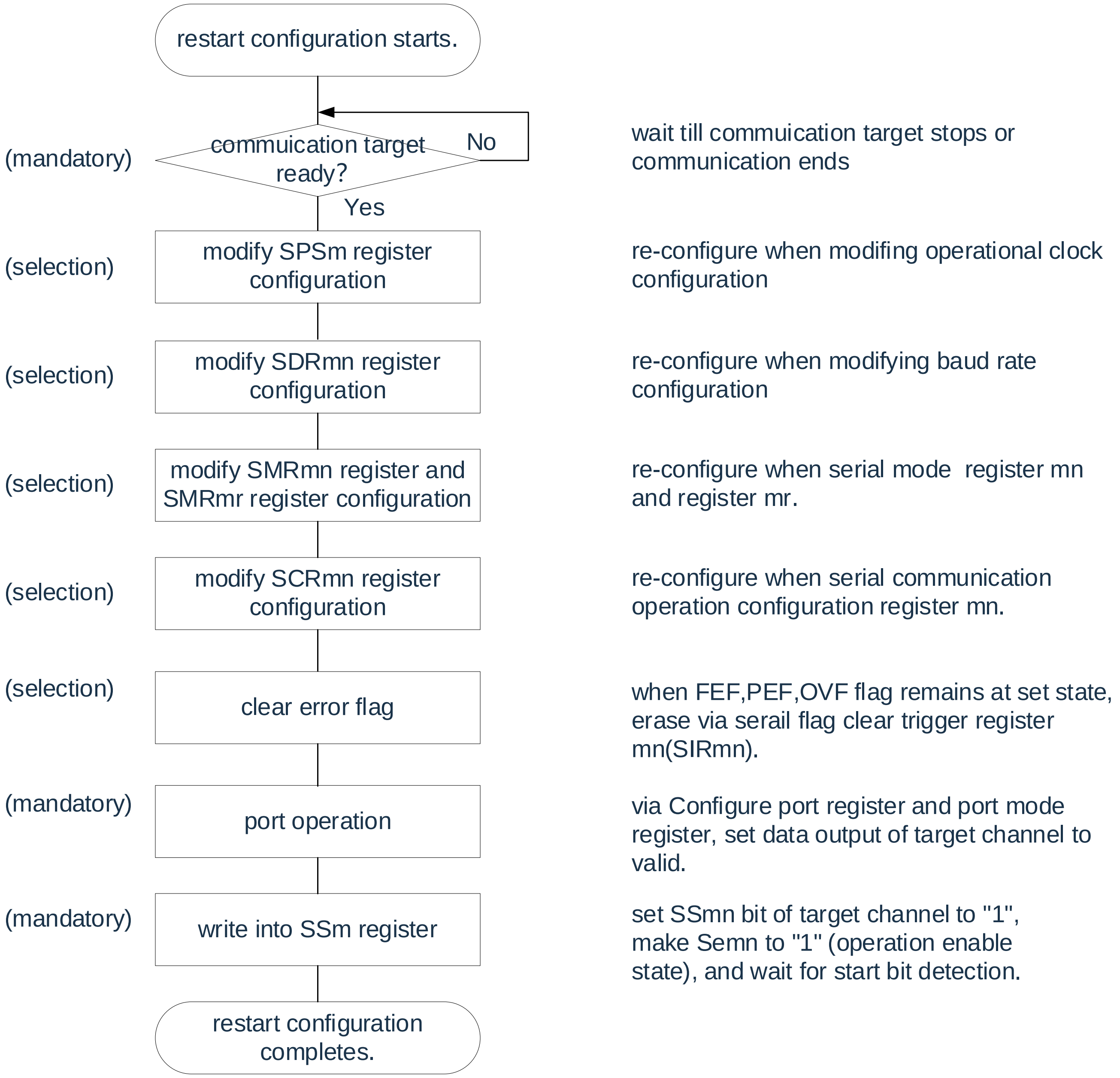

termination configuration starts
(selection)
TSFmn = 0?
No
Yes
if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).
(mandatory)
write into STm register
set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).
(selection)
configure PER0 register
stop clock of universal serial communication unit, set to reset state
termination configuration ends.
finish termination configuration, enter into next processing.

Figure 15-105 Restart set-up steps for UART reception

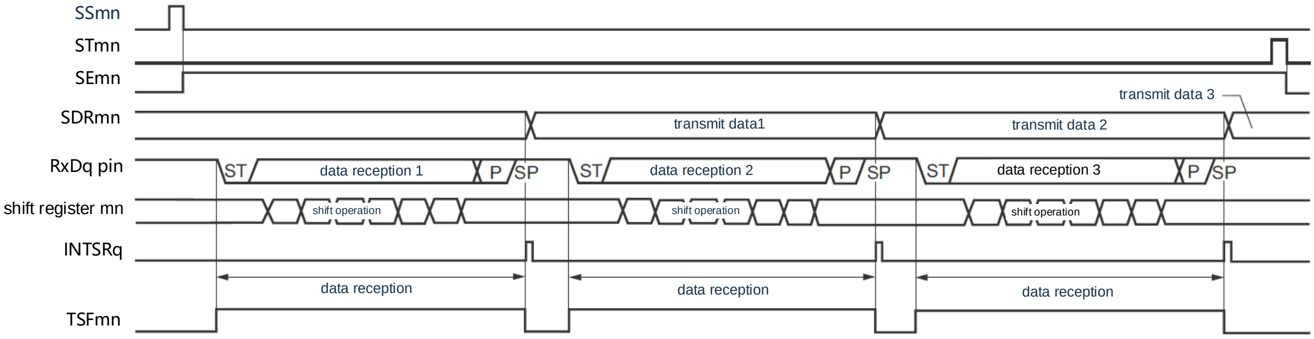

restart configuration starts.
(mandatory)
commuication target ready?
No
Yes
wait till commuication target stops or communication ends
(selection)
modify SPSm register configuration
re-configure when modifing operational clock configuration
(selection)
modify SDRmn register configuration
re-configure when modifying baud rate configuration
(selection)
modify SMRmn register and SMRmr register configuration
re-configure when serial mode register mn and register mr.
(selection)
modify SCRmn register configuration
re-configure when serial communication operation configuration register mn.
(selection)
clear error flag
when FEF,PEF,OVF flag remains at set state, erase via serail flag clear trigger register mn(SIRmn).
(mandatory)
port operation
via Configure port register and port mode register, set data output of target channel to valid.
(mandatory)
write into SSm register
set SSmn bit of target channel to "1", make Semn to "1" (operation enable state), and wait for start bit detection.
restart configuration completes.

Notice: At least 4 f_{MCK} clocks must be set after setting the RXEmn bit of the SCRmn register to "1", and then setting the SSmn bit to "1".

Remark: If that PER0 is override in the abort setting to stop the supply clock, the initial setting must wait until the communication object is stopped or communication is complete, rather than reset.

(3) process flow

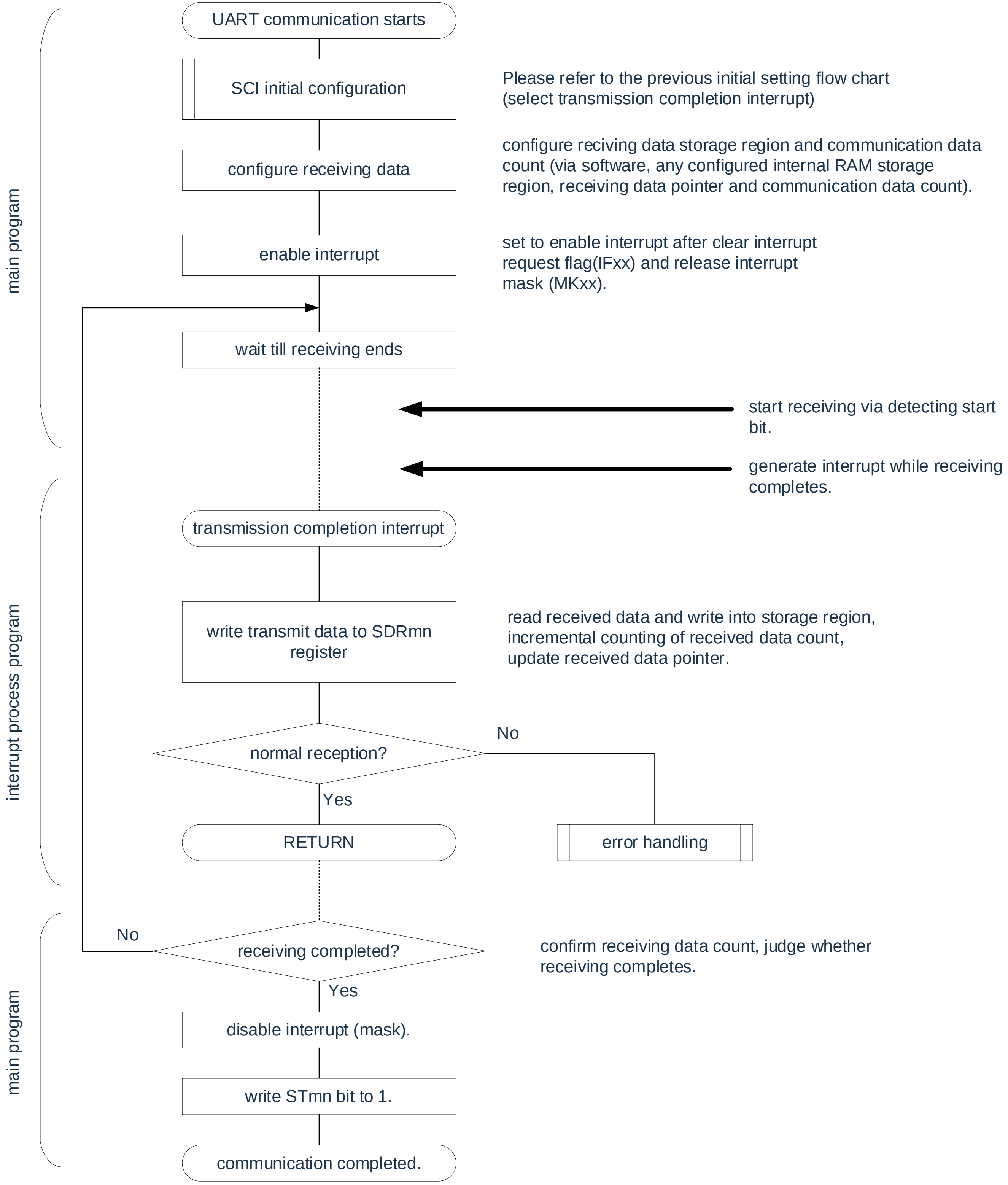
Figure 15-106 Timing of UART reception



Remark: m: Unit number (m=0,1,2) n: Channel number (n=1)

r: Channel number (r=n~1) q: UART number (q=0~2)

Figure 15-107 Flowchart of UART reception


main program
UART communication starts
SCI initial configuration
Please refer to the previous initial setting flow chart (select transmission completion interrupt)
configure receiving data
configure reciving data storage region and communication data count (via software, any configured internal RAM storage region, receiving data pointer and communication data count).
enable interrupt
set to enable interrupt after clear interrupt request flag(IFxx) and release interrupt mask (MKxx).
wait till receiving ends
start receiving via detecting start bit.
generate interrupt while receiving completes.
interrupt process program
transmission completion interrupt
write transmit data to SDRmn register
read received data and write into storage region, incremental counting of received data count, update received data pointer.
normal reception?
No
Yes
RETURN
error handling
main program
No
receiving completed?
confirm receiving data count, judge whether receiving completes.
Yes
disable interrupt (mask).
write STmn bit to 1.
communication completed.

15.7.3 Calculation of baud rate

(1) Formula for baud rate

The baud rate of UART(UART0~UART2) communication can be calculated by the following formula:

$$\boxed{\text{(Baud rate)=\{Operating clock (}f_{MCK}\text{) frequency\}} \div \text{(SDRmn [15:9]+1)} \div \text{2[bps]}}$$

Notice: The setting of SDRmn[15:9] of serial data register mn (SDRmn) to "0000000B" and "0000001B" is prohibited.

Remark: 1. Because the value of SDRmn[15:9] is bit15~9 of the SDRmn register (0000010B~1111111B) when using UART]. 2.m: Unit number (m=0,1,2) n: Channel Number (n=0,1)

The operating clock (f_{MCK}) is determined by the bit15 (CKSmn bit) of the serial clock selection register m (SPSm) and the serial mode register mn (SMRmn).

Table 15-4 Selection of UART operation clock

SMRmn register	SPSm register								Operating Clock (f_{MCK}) Note	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		f_{CLK}=32MHz in operation
0	X	X	X	X	0	0	0	0	f_{CLK}	32MHz
	X	X	X	X	0	0	0	1	$f_{CLK}/2$	16MHz
	X	X	X	X	0	0	1	0	$f_{CLK}/2^2$	8MHz
	X	X	X	X	0	0	1	1	$f_{CLK}/2^3$	4MHz
	X	X	X	X	0	1	0	0	$f_{CLK}/2^4$	2MHz
	X	X	X	X	0	1	0	1	$f_{CLK}/2^5$	1MHz
	X	X	X	X	0	1	1	0	$f_{CLK}/2^6$	500kHz
	X	X	X	X	0	1	1	1	$f_{CLK}/2^7$	250kHz
	X	X	X	X	1	0	0	0	$f_{CLK}/2^8$	125kHz
	X	X	X	X	1	0	0	1	$f_{CLK}/2^9$	62.5kHz
	X	X	X	X	1	0	1	0	$f_{CLK}/2^{10}$	31.25kHz
	X	X	X	X	1	0	1	1	$f_{CLK}/2^{11}$	15.63kHz
	X	X	X	X	1	1	0	0	$f_{CLK}/2^{12}$	7.81kHz
	X	X	X	X	1	1	0	1	$f_{CLK}/2^{13}$	3.91kHz
	X	X	X	X	1	1	1	0	$f_{CLK}/2^{14}$	1.95kHz
	X	X	X	X	1	1	1	1	$f_{CLK}/2^{15}$	977Hz
1	0	0	0	0	X	X	X	X	f_{CLK}	32MHz
	0	0	0	1	X	X	X	X	$f_{CLK}/2$	16MHz
	0	0	1	0	X	X	X	X	$f_{CLK}/2^2$	8MHz
	0	0	1	1	X	X	X	X	$f_{CLK}/2^3$	4MHz
	0	1	0	0	X	X	X	X	$f_{CLK}/2^4$	2MHz
	0	1	0	1	X	X	X	X	$f_{CLK}/2^5$	1MHz
	0	1	1	0	X	X	X	X	$f_{CLK}/2^6$	500kHz
	0	1	1	1	X	X	X	X	$f_{CLK}/2^7$	250kHz
	1	0	0	0	X	X	X	X	$f_{CLK}/2^8$	125kHz
	1	0	0	1	X	X	X	X	$f_{CLK}/2^9$	62.5kHz
	1	0	1	0	X	X	X	X	$f_{CLK}/2^{10}$	31.25kHz
	1	0	1	1	X	X	X	X	$f_{CLK}/2^{11}$	15.63kHz
	1	1	0	0	X	X	X	X	$f_{CLK}/2^{12}$	7.81kHz
	1	1	0	1	X	X	X	X	$f_{CLK}/2^{13}$	3.91kHz
	1	1	1	0	X	X	X	X	$f_{CLK}/2^{14}$	1.95kHz
	1	1	1	1	X	X	X	X	$f_{CLK}/2^{15}$	977Hz

Note To change the clock selected as f_{CLK} (change the value of the System Clock Control Register (CKC), you must change after stopping Universal Serial Communication Unit (SCI) =000FH.

Note 1.X: Ignore

2.m: Unit number (m=0,1,2) n: Channel number (n=0,1)

(2) Baud rate error during transmission

The baud rate error of UART (UART0~UART2) communication transmission can be calculated by the following formula, the baud rate of the sender must be set within the acceptable range of the baud rate of the receiver.

(Baud rate error)=(Calculated value of baud rate) ÷ (Value of target baud rate) ×100-100[%]

An example of setting the UART baud rate at f_{CLK}=32 MHz is shown below.

UART baud rate (target baud rate)	f_{CLK}=32MHz			
	Operating clock (f_{MCK})	SDRmn [15:9]	Calculated value of baud rate	Error with target baud rate
300bps	$f_{CLK}/2^9$	103	300.48bps	+0.16%
600bps	$f_{CLK}/2^8$	103	600.96bps	+0.16%
1200bps	$f_{CLK}/2^7$	103	1201.92bps	+0.16%
2400bps	$f_{CLK}/2^6$	103	2403.85bps	+0.16%
4800bps	$f_{CLK}/2^5$	103	4807.69bps	+0.16%
9600bps	$f_{CLK}/2^4$	103	9615.38bps	+0.16%
19200bps	$f_{CLK}/2^3$	103	19230.8bps	+0.16%
31250bps	$f_{CLK}/2^3$	63	31250.0bps	±0.0%
38400bps	$f_{CLK}/2^2$	103	38461.5bps	+0.16%
76800bps	$f_{CLK}/2$	103	76923.1bps	+0.16%
153600bps	f_{CLK}	103	153846bps	+0.16%
312500bps	f_{CLK}	50	313725bps	±0.39%

Remark: m: Unit number (m=0,1,2) n: Channel number (n=0)

(3) Acceptable range of baud rate at reception

The baud tolerance of UART (UART0~UART2) communication when receiving can be calculated by the following formula, the baud rate of the sender must be set within the baud tolerance of the receiver.

$$\text{(Maximum acceptable baud rate)} = \frac{2\times k\times Nfr}{2\times k\times Nfr-k+2}\times \text{Brate}$$

$$\text{(Minimum Acceptable Wavelet Rate)} = \frac{2\times k\times (Nfr-1)}{2\times k\times Nfr-k-2}\times \text{Brate}$$

Brate: The calculated value of the baud rate for the receiver (reference to "15.7.4(1) Baud rate formula")

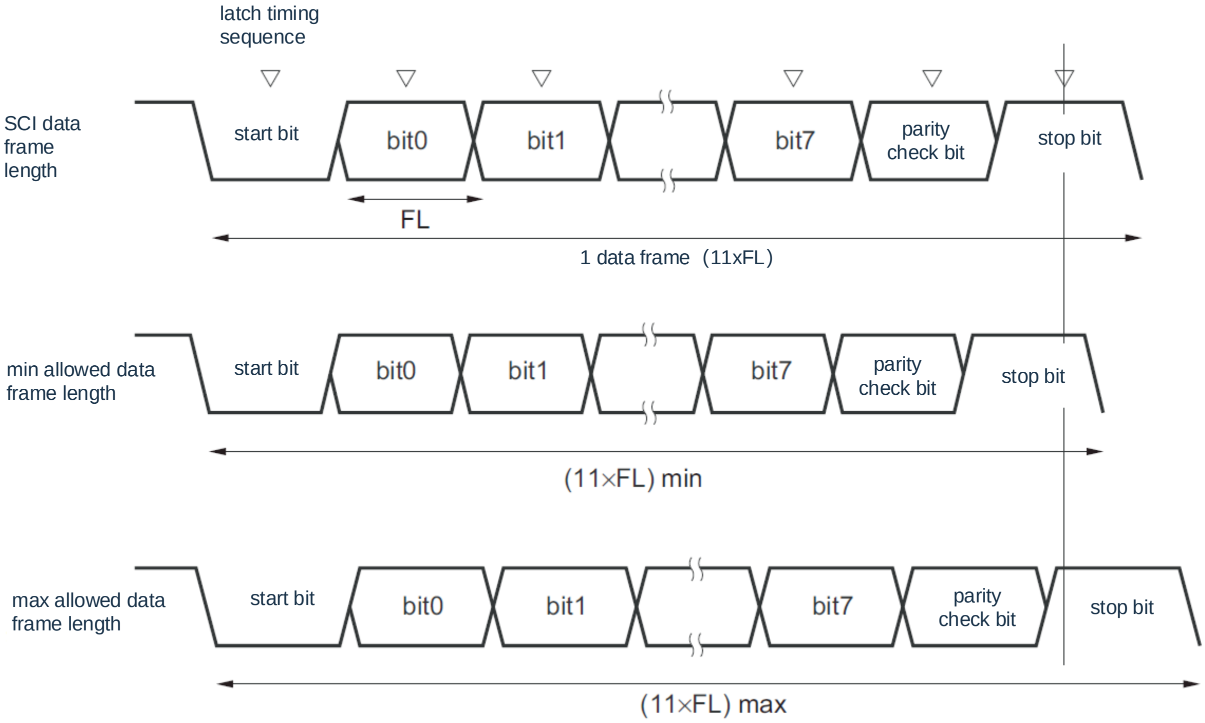
k: SDRmn [15:9]+1

Nfr: Frame length of 1 data [bit]

= (start bit)+(data length)+(parity bit)+(stop bit)

Remark m: Unit number (m=0, 1, 2) n: Channel number (n=1)

Figure 15-108 Acceptable range of baud rate at reception (in case of 1 data frame length=11 bits)



As shown in figure 15-108, after detecting the start bit, the latch timing of the received data depends on the bit15~9th set by serial data register mn (SDRmn). If the last data (the stop bit) can catch up with this latch sequence, it can be received normally.

15.7.4 Processing steps when an error occurs during UART (UART0~UART2) communication

The processing steps when an error occurs during UART (UART0~UART2) communication are shown in Figure 15-109 and Figure 15-110.

Figure 15-109 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 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).		Determine the type of error 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.

Figure 15-110 Processing steps when frame errors occur

software operation	Hardware Status	Comments
Read the serial data register mn	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).		Determine the type of error and the read value is used to clear the error flag.
write serial flag clear trigger register mn (SIRmn).	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.
Set the STmn bit of the serial channel stop register m (STm) to "1".	The serial channel allows the SEmn bit of the status register m (SEm) to be "0" and channel n to be running stopped.	A frame error can be considered to have occurred due to the start bit offset. It is therefore necessary to re-synchronize with that communicator and resume communication.
Synchronize with the communicating party.		
Set the SSmn bit of the serial channel start register m (SSm) to "1".	The serial channel allows the SEmn bit of the status register m (SEm) to be "1" and channel n to be operational.	

Remark m: Unit number (m=0,1,2) n: Channel number (n=0,1)

15.8 Operation of LIN communication

15.8.1 LIN transmission

UART0 supports LIN communication in UART transmission.
LIN transmits channel 0 of unit 0.

UART	UART0	UART1	UART2
LIN Communication Support	Yes	No	No
Object channel	Channel 0 for SCI0	—	—
Pin used	TxD0	—	—
Interrupt	INTST0	—	—
	Interrupt at that end of the transfer may be selecte (single transfer mode) or buffer air-discontinuity (continuous transfer mode).		
Error detection flag	None		
Length of transmit data	8-bit		
Transfer rate [Note]	Max.f_{MCK}/6[bps] (SDR.0[15:9]≥2), Min.f_{CLK}/(2×2^{15}×128) [bps]		
Data phase	Forward output (default: High level). Inverted output (default: Low level).		
Parity bit	No parity bits.		
Stop bit	Appending 1-bit.		
Data orientation	LSB first		

Note: It must be used within the context of peripheral functionality (reference data manual) that meets this condition and electrical characteristics, and 2.4/9.6/19.2kbps.

Remark: f_{MCK}: Operating clock frequency of the object channel
f_{CLK}: system clock frequency

LIN is the abbreviation of Local Interconnect Network, which is a low-speed (1~20kbps) serial communication protocol to reduce automobile network cost. LIN communications are single-master communications, with up to 15 slave devices connected to a single master device.

The LIN slave is used for the control of switches, transmission devices, sensors, etc., which are connected to the main control device through the LIN.

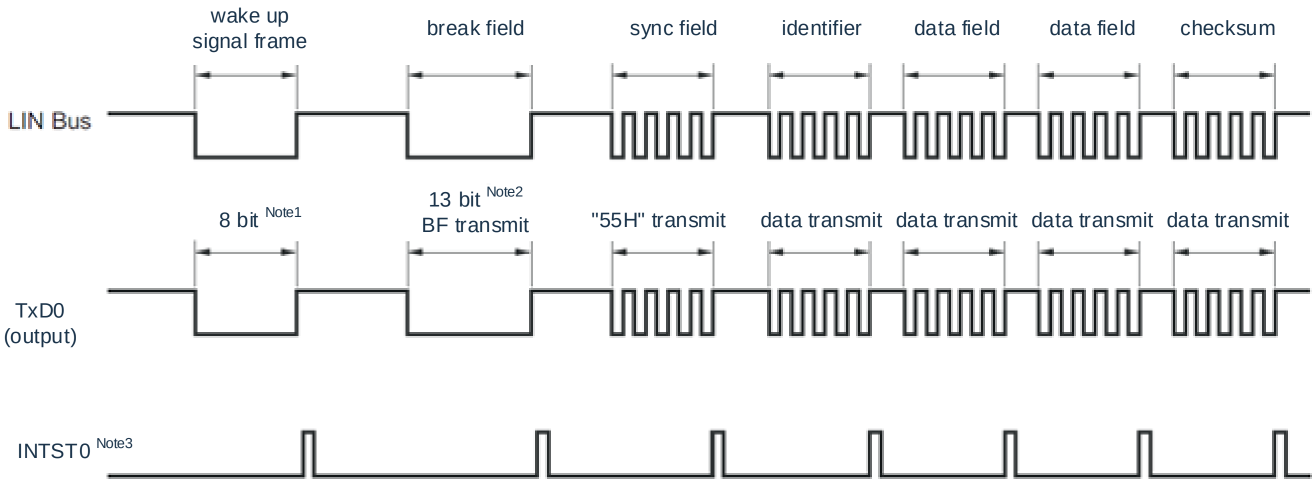
The LIN controls the network which connects CAN (Controller Area Network) and so on.

The LIN bus is a single-line bus, which connects nodes through ISDO9141-compliant transceivers.

According to the LIN protocol, the master device sends a frame with additional baud rate information, and the slave device receives the frame and corrects the baud rate error with the master device. Therefore, if the baud rate error of the slave device is not greater than ±15%, communication can be performed.

A summary of the LIN's send operations is shown in Figure 15-111.

Figure 15-111 Operation of LIN transmission



Note: 1. In order to meet the requirements of wake-up signal, set baud rate and send "80H" data to correspond.

2. The break field is specified as a 13-bit wide low-level output, so the baud rate used for the main transmission is N[bps]:

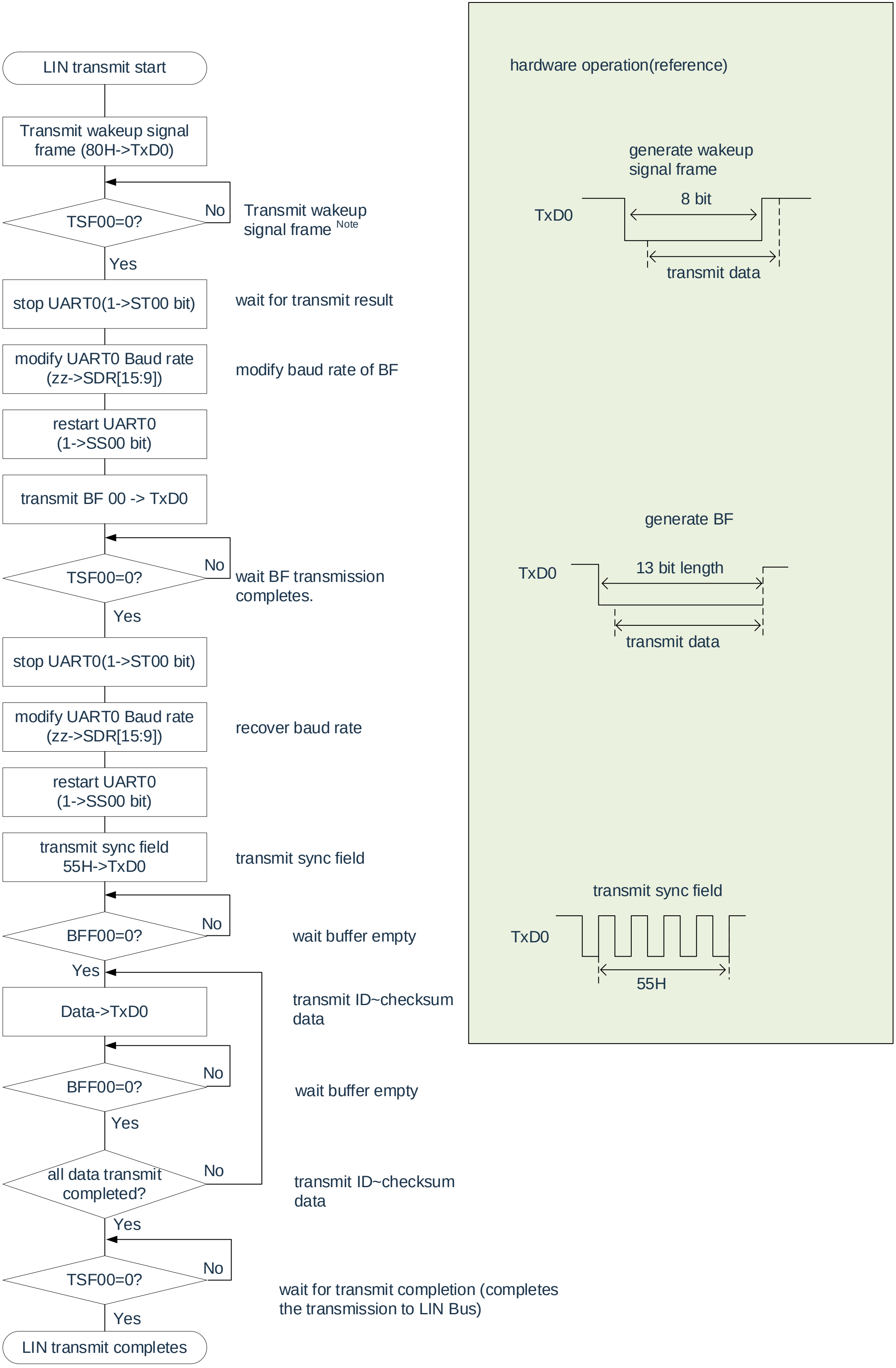
(Baud Rate for Interval) = 9/13×N

Transmitting the data of "00H" through this baud rate to generate a break field.

3. Output INTST0 at the end of each data transmission, and also output INTST0 at BF transmission.

Remark: The software controls each break field.

Figure 15-112 Flowchart of LIN transmission


LIN transmit start
Transmit wakeup signal frame (80H->TxD0)
TSF00=0?
No
Yes
Transmit wakeup signal frame Note
stop UART0(1->ST00 bit)
wait for transmit result
modify UART0 Baud rate (zz->SDR[15:9])
modify baud rate of BF
restart UART0 (1->SS00 bit)
transmit BF 00 -> TxD0
TSF00=0?
No
Yes
wait BF transmission completes.
stop UART0(1->ST00 bit)
modify UART0 Baud rate (zz->SDR[15:9])
recover baud rate
restart UART0 (1->SS00 bit)
transmit sync field 55H->TxD0
transmit sync field
BFF00=0?
No
Yes
wait buffer empty
Data->TxD0
transmit ID~checksum data
BFF00=0?
No
Yes
wait buffer empty
all data transmit completed?
No
Yes
transmit ID~checksum data
TSF00=0?
No
Yes
wait for transmit completion (completes the transmission to LIN Bus)
LIN transmit completes
hardware operation(reference)
generate wakeup signal frame
TxD0
8 bit
transmit data
generate BF
TxD0
13 bit length
transmit data
transmit sync field
TxD0
55H

Note: It is limit to situations starting from lin-bus sleep.

Remark: This is the process that starts by ending the initial set-up of the UART and allowing slave sending.

15.8.2 LIN reception

In UART reception, UART0 supports LIN communication.
LIN receives Channel 1 of unit 0.

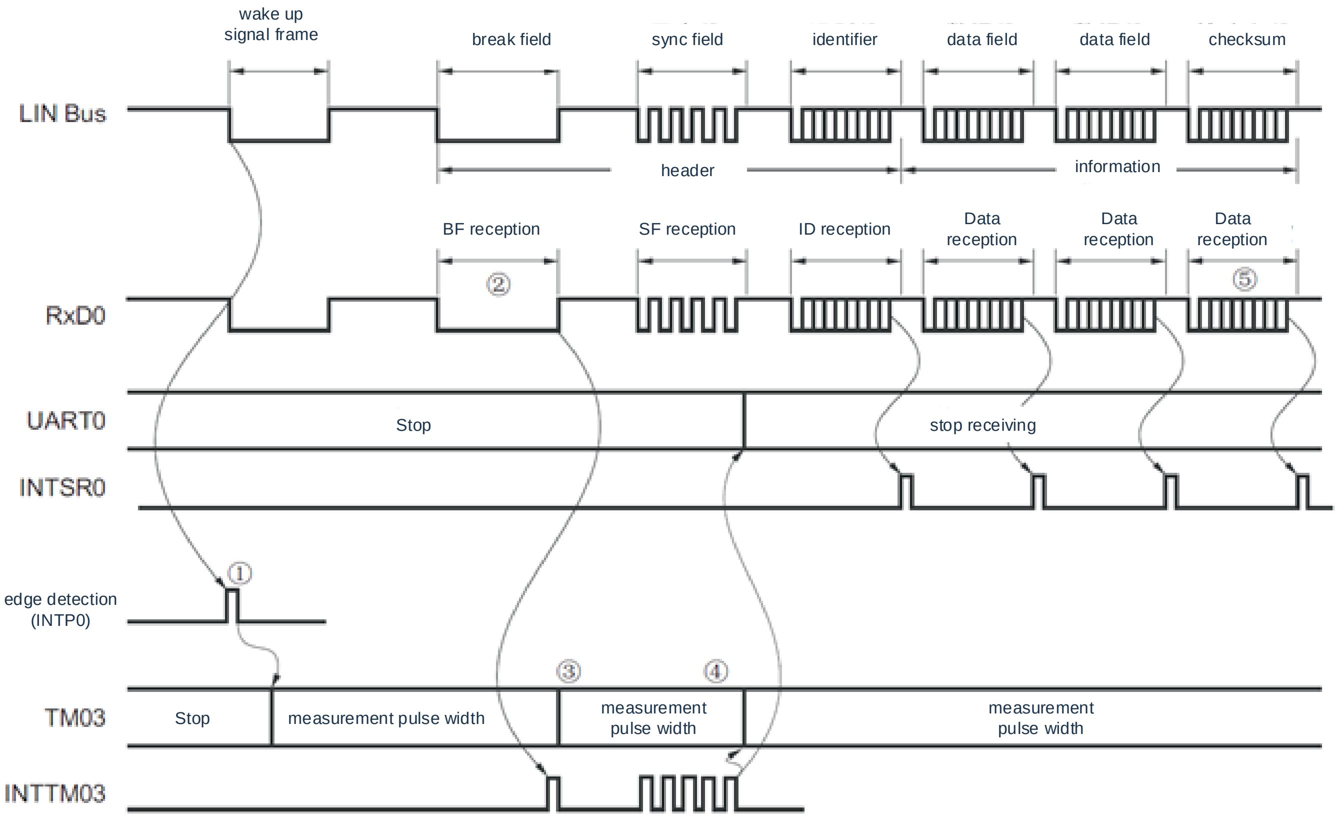
UART	UART0	UART1	UART2	UART3
LIN communication support	Yes	No	No	No
Object channel	Channel 1 for SCI0	—	—	—
Pin used	RxD0	—	—	—
Interrupt	INTSR0	—	—	—
	Interrupt at that end of the transfer only (Disable setting buffer null interrupt).			
Error interrupt	INTSRE0	—	—	—
Error detection flag	· Frame Error Detection Flag (FEF01) · Overflow Error Detection Flag (OVF01)			
Length of transmit data	8-bit			
Transfer rate [Note]	Max.f_{MCK}/6[bps](SDR01[15:9]≥2), Min.$f_{CLK}/(2\times2^{15}\times128)$[bps]			
Data phase	Forward output (default: High level). Inverted output (default: Low level).			
Parity bit	No parity bits (no parity).			
Stop bit	Appending 1-bit.			
Data orientation	LSB First			

Note: It must be used within the scope of peripheral functional characteristics (reference data manual) that meet this condition and electrical characteristics.

Remark: f_{MCK}: Operating clock frequency of the object channel
f_{CLK}: system clock frequency

A summary of the receive operations for the LIN is shown in Figure 15-113.

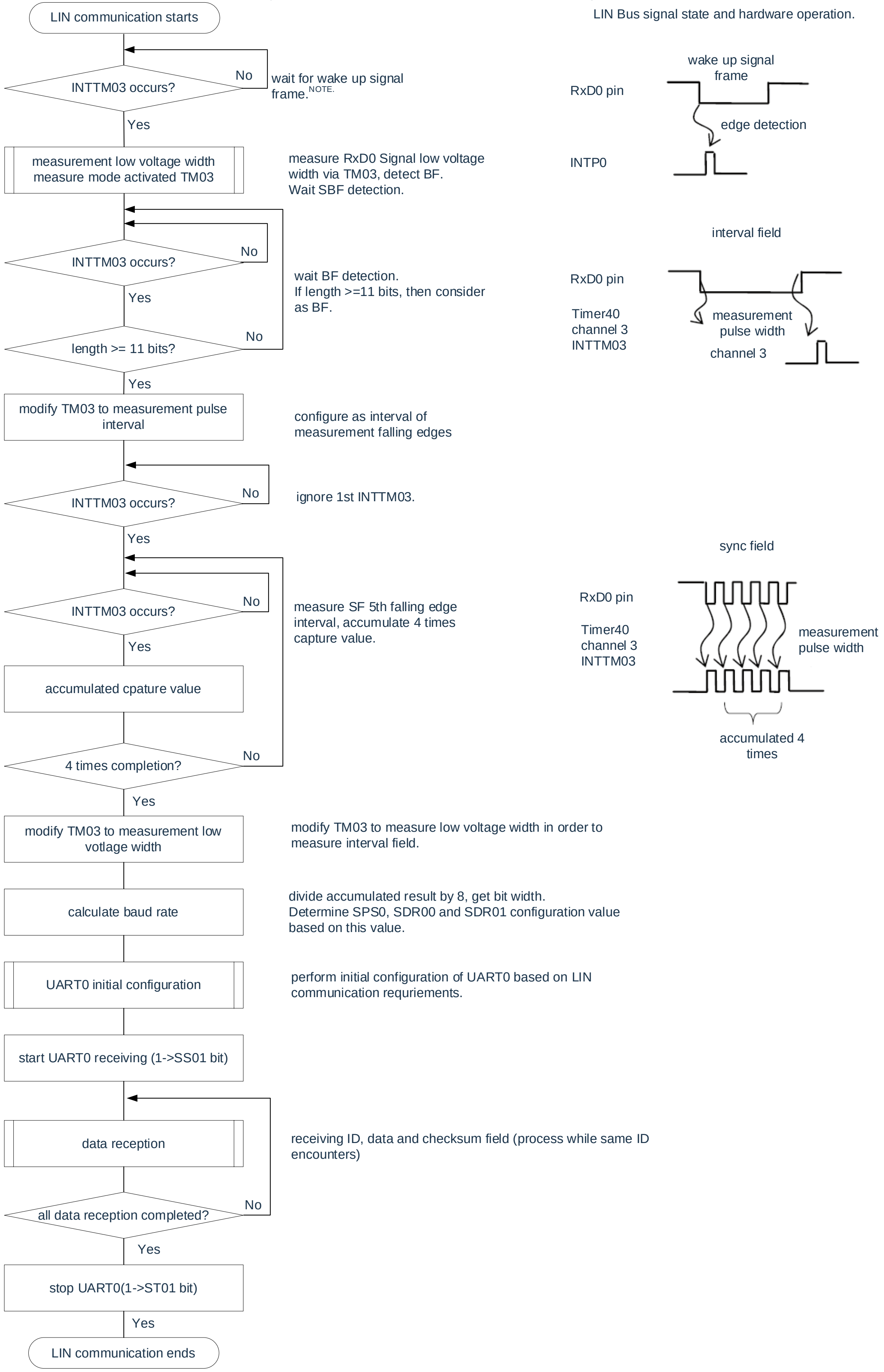
Figure 15-113 Operation of LIN reception



The signal processing flow is as follows:

1) The wake-up signal is detected by detecting the INTP0 of the pin. When the wake signal is detected, the TM03 is set to measure the pulse width in order to measure the low level width of BF.
2) If the falling edge of BF is detected, TM03 starts to measure the low level width and captures the rising edge of BF. The BF signal is judged according to the captured value.
3) When BF reception ends normally, TM03 must be set as the measurement pulse interval, and the interval of RxD0 signal falling edge of 4 synchronizations(Refer to "6.8.4 Operation as input pulse interval measurement").
4) Calculating the baud rate error according to the bit interval of the synchronization section (SF). The baud rate must then be adjusted (reset) after the UART0 run has been paused.
5) The checksum segment must be distinguished by software. You must also initialize the UART0 after receiving the checksum segment through the software and set it to the BF receive wait state again.

Figure 15-114 Flowchart of LIN reception


LIN Bus signal state and hardware operation.
LIN communication starts
INTTM03 occurs?
No
Yes
wait for wake up signal frame.NOTE.
wake up signal frame
RxD0 pin
edge detection
INTP0
measurement low voltage width measure mode activated TM03
measure RxD0 Signal low voltage width via TM03, detect BF.
Wait SBF detection.
interval field
INTTM03 occurs?
No
Yes
wait BF detection.
If length >=11 bits, then consider as BF.
RxD0 pin
Timer40 channel 3 INTTM03
measurement pulse width
channel 3
length >= 11 bits?
No
Yes
modify TM03 to measurement pulse interval
configure as interval of measurement falling edges
INTTM03 occurs?
No
Yes
ignore 1st INTTM03.
sync field
INTTM03 occurs?
No
Yes
measure SF 5th falling edge interval, accumulate 4 times capture value.
RxD0 pin
Timer40 channel 3 INTTM03
measurement pulse width
accumulated cpature value
accumulated 4 times
4 times completion?
No
Yes
modify TM03 to measurement low votlage width
modify TM03 to measure low voltage width in order to measure interval field.
calculate baud rate
divide accumulated result by 8, get bit width.
Determine SPS0, SDR00 and SDR01 configuration value based on this value.
UART0 initial configuration
perform initial configuration of UART0 based on LIN communication requriements.
start UART0 receiving (1->SS01 bit)
data reception
receiving ID, data and checksum field (process while same ID encounters)
all data reception completed?
No
Yes
stop UART0(1->ST01 bit)
Yes
LIN communication ends

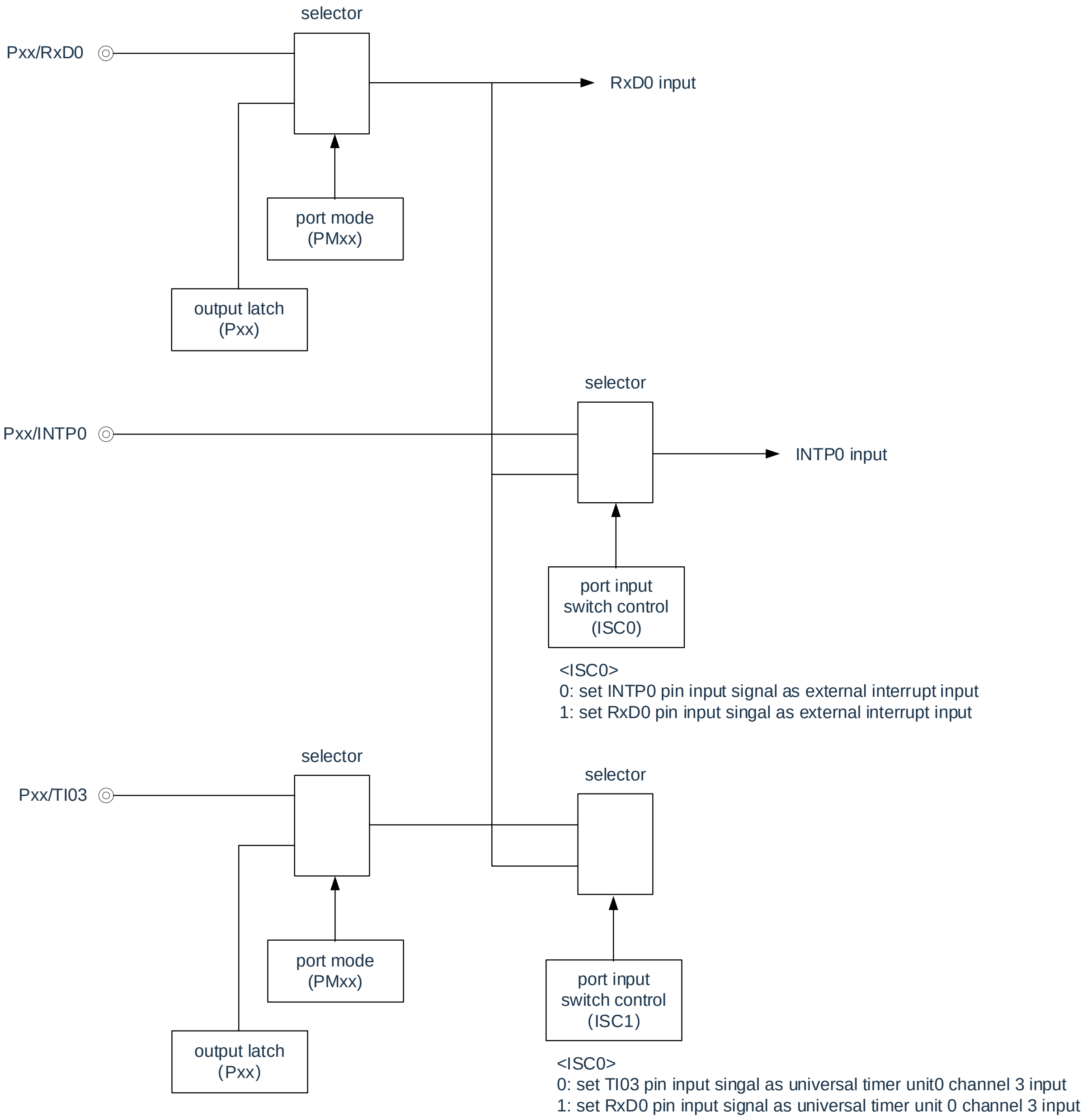
Note: Only needed for sleep status.

The port structure diagram for LIN receive operations is shown in Figure 15-115.

The wake-up signal sent by the LIN master is received through edge detection of the INTP0. The invention can measure the length of the sync field sent by the LIN master and calculate the baud rate error through external event capture operation.

The input source for the received port input (RxD0) can be input to the external interrupt (INTP0) and timer array unit without external connection by port input switching control (ISC0/ISC1).

Figure 15-115 Port map for LIN receive operations



<ISC0>
0: set INTP0 pin input signal as external interrupt input
1: set RxD0 pin input singal as external interrupt input

<ISC0>
0: set TI03 pin input singal as universal timer unit0 channel 3 input
1: set RxD0 pin input signal as universal timer unit 0 channel 3 input

Remark ISC0, ISC1: Enter the bit0 and bit1 for the Switch Control Register (ISC)

Peripheral features for LIN communication operations are summarized as follows:

<Peripheral Features Used>

- External interrupt (INTP0): Detection of wake-up signal

  Purposes of use: Detects edges of wake-up signals and the start of communication.
- Channel 3 of the universal timer unit: Detection of Baud Rate Error and Detection of Interval (BF)

  Purposes of use: The length of the synchronization section (SF) is detected and the baud rate error is detected by dividing its length by bits (the interval of the RxD0 input edge is measured by capture mode). A low level width is measured to determine whether it is a spacer (BF).
- Channel 0 and channel 1 (UART0) of universal serial communication unit 0 (SCI0)

15.9 Operation of simplified I^{2}C (IIC00, IIC01, IIC10, IIC11, IIC20,IIC21) communication

This is the capability of clock synchronization with multiple devices through two lines of serial clock (SCL) and serial data (SDA). Since this simplified I^{2}C is designed for single communication with EEPROM, flash memory, A/D converter, etc., it is only used as master device.

For the start condition and stop condition, the AC specification must be observed, and the control register must be handled by software.

[Transmitting and Receiving Data]

- Master Send, Master Receive (only for single master master master master functions)
- ACK output function Note, ACK detection function
- 8-bit data length (when sending an address, specifying the address with 7 bits high and R/W control with lowest bits)
- A start condition and a stop condition are generated by the software.

[Interrupt Function]

- End of Transfer Interrupt

[Error Detection Flag]

- ACK Error

※[Features not supported by Simplified I^{2}C]

- Slave send, slave receive
- Multi-Master (Quorum Failure Detection)
- Waiting for detection

Note: When the last data is received, if "0" is written to the SDOEmn bit (SDOEm register) to stop the serial communication data output. Refer to "15.9.3 (2) Process Flow".

Remark: m: Unit number (m=0,1,2) n: Channel number (n=0,1)

The channels 0~3 of SCI0 and 0~1 of SCI1 are the channels supporting simplified I^{2}C (IIC00, IIC01,IIC10,IIC11,IIC20,IIC21).

Simplified I^{2}C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21, IIC30, IIC31) has the following four kinds of communication running:

- Address field transmission (Refer to 15.9.1)
- Data transmission (Refer to 15.9.2)
- Data reception (Refer to 15.9.3)
- Generation of stop condition (Refer to 15.9.4)

15.9.1 Address field transmission

Address field transmission is a send run that is first performed when I²C communication, specifically specifying a transfer object (slave). After the start condition is generated, the address (7 bits) and the transmission direction (1 bits) are transmitted as 1 frames.

Simplified I²C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21
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	SCL00, SDA00 note 1	SCL01, SDA01 note 1	SCL10, SDA10 note 1	SCL11, SDA11 note 1	SCL20, SDA20 note 1	SCL21, SDA21 note 1
Interrupt	INTEGER00	INTEGER01	INTEGER10	INTEGER11	INTEGER20	INTEGER21
	Only interrupt at that end of the transfer (no buffer interrupt can be selected).					
Error detection flag	ACK Error Detection Flag (PEFmn)					
Length of transmit data	8-bit (High 7 bits as addresses and low 1 bits as R/W controls)					
Transfer Rate Note 2	Max.f_{MCK}/4[Hz](SDRmn[15:9]≥1)f_{MCK}: The operating clock frequency of the object channel, however, must meet the following conditions in each mode of I2C: · Max.1MHz · Max.400kHz (Quick Mode) · Max.100kHz (Standard Mode)					
Data level	Forward output (default: High level).					
Parity bit	No parity bits.					
Stop bit	Appending 1-bit (for ACK reception).					
Data orientation	MSB First					

Note: 1. To communicate through simplified I²C, the N-channel drain open output mode (POMxx=1) must be set through the port output mode register (POMxx). For details, refer to "Chapter 2 Port Function" .

2. It must be used within the scope of the peripheral functional characteristics (refer to the data sheet) that meet this condition and satisfy the electrical characteristics.

Remark: m: Unit number (m=0,1,2) n: Channel number (n=0,1).

(1) Register settings

Figure 15-116 Example of register setting contents when transmitting address field of Simplified I^{2}C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)

(a) serial mode register mn (SMRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn 0/1	CCSmn 0	0	0	0	0	0	STSmn 0 Note1	0	SISmn0 0 Note1	1	0	0	MDmn2 1	MDmn1 0	MDmn0 0

channel n operational clock (fMCK)
0: SPSm register configured pre-scaler output clock CKm0
1: SPSm register configured pre-scaler output clock CKm1

Interrupt source for channel n
0: End of transmission interrupt

(b) serial communication operation configuration register mn (SCRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn 1	RXEmn 0	DAPmn 0	CKPmn 0	0	EOCmn 0	PTCmn1 0	PTCmn0 0	DIRmn 0	0	SLCmn1 0 Note 2	SLCmn0 1	0	1	DLSmn1 1 Note 3	DLSmn0 1

parity check bit configuration
00B: no parity check

stop bit configuration
01B: append 1 bit (ACK)

(c) serial data regsiter mn (SDRmn) (low 8 bit: SIOr)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration							0	configuration of transmit data(Address+R/W)							

SIOr (bits 7–0)

(d) serial output register m (Som)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm	0	0	0	0	0	0	CKOm1 0/1	CKOm0 0/1	0	0	0	0	0	0	SOm1 0/1	SOm0 0/1

generate start condition via operating Somn bit.

(e)serial otuput enable register m (SOEm)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOEm1 0/1	SOEm0 0/1

Before generating start condition, SOEmn bit is '0'.
After generating start condition, SOEmn bit is '1'.

(f) serial channel start register m (SSm) .... Only set bit of target channel to 1.

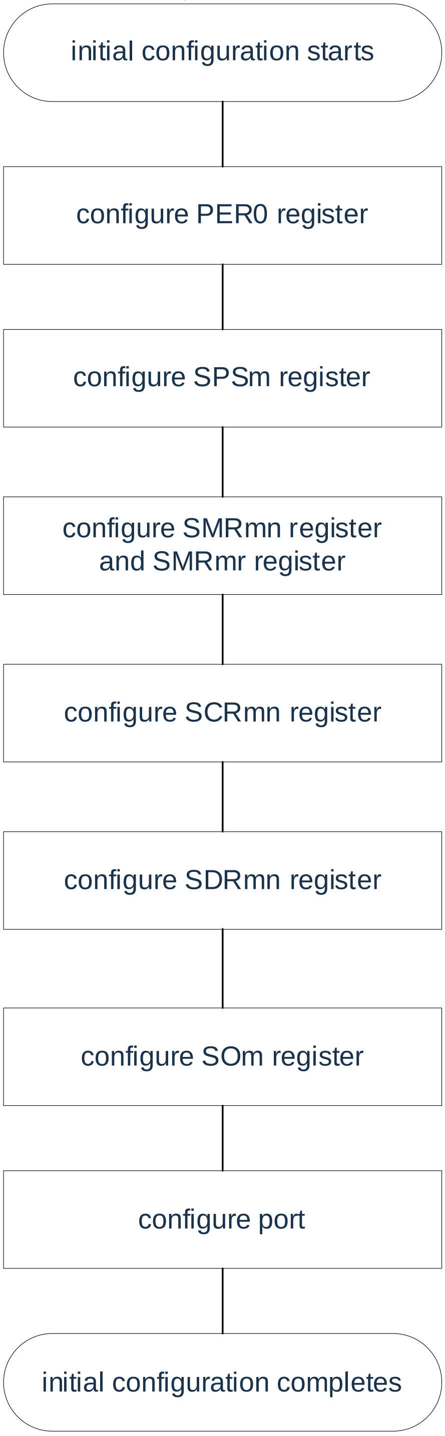
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SSm1 0/1	SSm0 0/1

Note 1. Limited to SMR00, SMR03, SMR11.
2. Limited to SCR00, SCR02, SCR10.
3. Limited to SCR00 and SCR01 registers, other fixed to "1".

Note 1.m: Unit number (m=0,1,2) n: Channel number (n=0,1)r: IIC number (r=00,01,10,11,20,21)
mn=00~03,10~11
2. □ : Set in IIC mode for Fixed. ▨ : Cannot set (set initial value).
×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).
0/1: The "0" or "1" is set according to the user.

(2) Procedure

Figure 15-117 Initial set-up steps for address field transmission



release universal serial communication unit from reset state, start providing clock.

configure operational clock

configure operational mode..etc.

configure communication format

configure transmit baud rate (configure operationl clock(fMCK) scaled transmission clock)

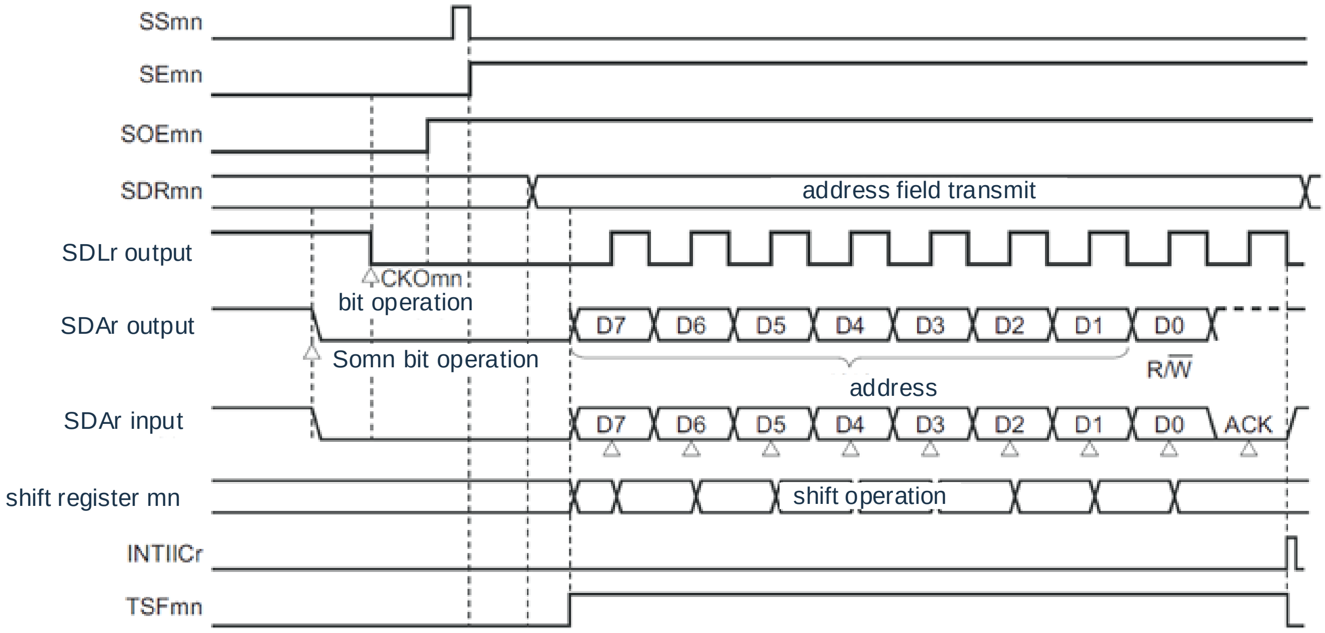
configure serial data(SOmn) and serial clock(CKOmn) initial output voltage (set "1")

via configure port register, port mode register and port output mode register, set data output, clock output and N-channel open-drain output of target channel to valid.

Remark: At the end of the initial set-up, Simplified I^{2}C (IIC00, IIC01, IIC10, IIC11, IIC20,IIC21) is output-disabled and in a run stop status.

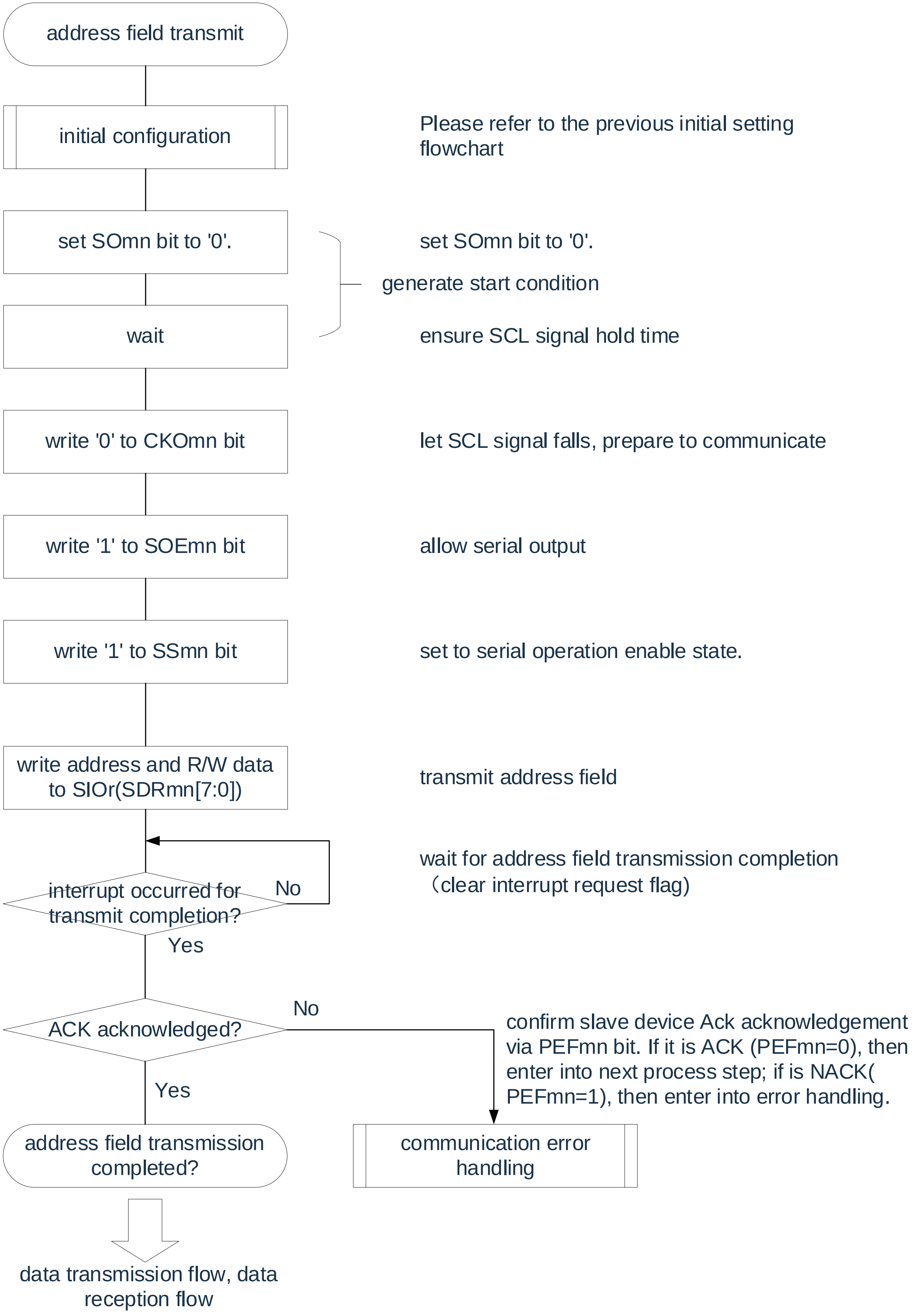
(3) Process flow

Figure 15-118 Timing diagram for address field transmission



Remark m: Unit number (m=0,1,2) n: Channel number (n=0,1) r: IIC number (r=00,01,10,11,20,21)

Figure 15-119 Flowchart of address field transmisson


address field transmit
initial configuration
Please refer to the previous initial setting flowchart
set SOmn bit to '0'.
set SOmn bit to '0'.
generate start condition
wait
ensure SCL signal hold time
write '0' to CKOmn bit
let SCL signal falls, prepare to communicate
write '1' to SOEmn bit
allow serial output
write '1' to SSmn bit
set to serial operation enable state.
write address and R/W data to SIOr(SDRmn[7:0])
transmit address field
interrupt occurred for transmit completion?
No
wait for address field transmission completion（clear interrupt request flag)
Yes
ACK acknowledged?
No
confirm slave device Ack acknowledgement via PEFmn bit. If it is ACK (PEFmn=0), then enter into next process step; if is NACK(PEFmn=1), then enter into error handling.
Yes
address field transmission completed?
communication error handling
data transmission flow, data reception flow

15.9.2 Data transmission

Data transmission is the operation of transmitting data to the transmission object (slave device) after the address segment is transmitted. A stop condition is generated after all data is sent to the object slave and the bus is released.

Simplified I^2C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21
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	SCL00, SDA00 note 1	SCL01, SDA01 note 1	SCL10, SDA10 note 1	SCL11, SDA11 note 1	SCL20, SDA20 note 1	SCL21, SDA21 note 1
Interrupt	INTEGER00	INTEGER01	INTEGER10	INTEGER11	INTEGER20	INTEGER21
	Only interrupt at that end of the transfer (no buffer interrupt can be selected).					
Error detection flag	ACK Error Flag (PEFmn)					
Length of transmit data	8-bit					
Transfer rate[Note2]	Max.f_{MCK}/4 [Hz] (SDRmn [15:9]≥1) f_{MCK}: The operating clock frequency of the object channel, however, must meet the following conditions in each mode of I^2C: · Max.1MHz · Max.400kHz (Quick Mode) · Max.100kHz (Standard Mode)					
Data level	Forward output (default: High level).					
Parity bit	No parity bits.					
Stop bit	Appending 1-bit (for ACK reception).					
Data orientation	MSB First					

Note: 1. To communicate through simplified I^2C, the N-channel drain open output mode (POMxx=1) must be set through the port output mode register (POMxx). Refer to "2.3 Registers for controlling port function" and 2.5 Register settings when using the multiplexing function".

2. It It must be used within the scope of the peripheral functional characteristics (refer to the data sheet) that meet this condition and satisfy the electrical characteristics.

Remark: m: Unit number (m=0,1,2) n: Channel number (n=0,1)

(1) Register settings

Figure 15-120 Example of register setting contents for simplified I²C data transmission (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)

(a) serial mode register mn (SMRmn)……do not operate this register wihle 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 Note1	0	0 Note1	1	0	0	1	0	0

(b) Serial communication operation setting register mn (SCRmn) ...... Bits other than the TXEmn bit and RXEmn bit are not operated during data transmission and reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn	RXEmn	DAPmn	CKPmn		EOCmn	PTCmn1	PTCmn0	DIRmn		SLCmn1	SLCmn0			DLSmn1	DLSmn0
	1	0	0	0	0	0	0	0	0	0	0 Note2	1	0	1	1 Note3	1

(c) Serial Data Register mn (SDRmn) (Low 8 bits: SIOr) ...... Only the low 8 bits are valid during data transmission and reception (SIOr).

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration Note4							0	configuration of transmit data							
									SIOr							

(d) Serial output register m (SOm) ...... This register is not operated during data sending and receiving.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm							CKOm1	CKOm0							SOm1	SOm0
	0	0	0	0	0	0	0/1 Note5	0/1 Note5	0	0	0	0	0	0	0/1 Note5	0/1 Note5

(e) Serial Output Enable Register m(SOEm) ...... This register is not operated during data sending and receiving.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm															SOEm1	SOEm0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

(f) Serial channel start register m(SSm) ...... This register is not operated during data transmission and reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm															SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0/1	0/1

Note 1. Limited to SMR01, SMR11, SMR21 registers only.
2. Limited to SCR00, SCR10, SCR20 registers only.
3. Limited to SCR00, SCR01, SCR10, SCR11, SCR20, SCR21 registers, other fixed to "1".
4. No set-up is required because the address segment is already set when it is sent.
5. During a communication run, the value changes due to communication data.

Note 1.m: Unit number (m=0,1,2) n: Channel number (n=0,1) r: IIC number (r=00,01,10,11,20,21)
2. □ : Set in IIC mode for Fixed. ▨ : Cannot set (set initial value).
×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).
0/1: The "0" or "1" is set according to the user.

(2) Process flow

Figure 15-121 Timing of data transmission

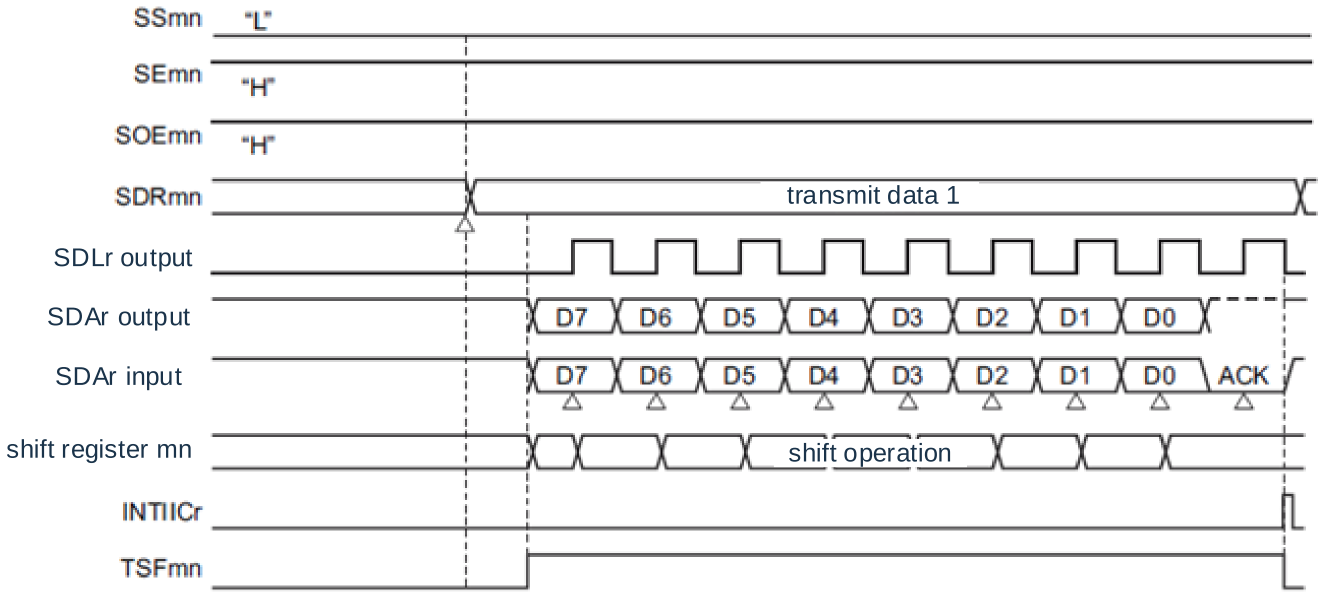
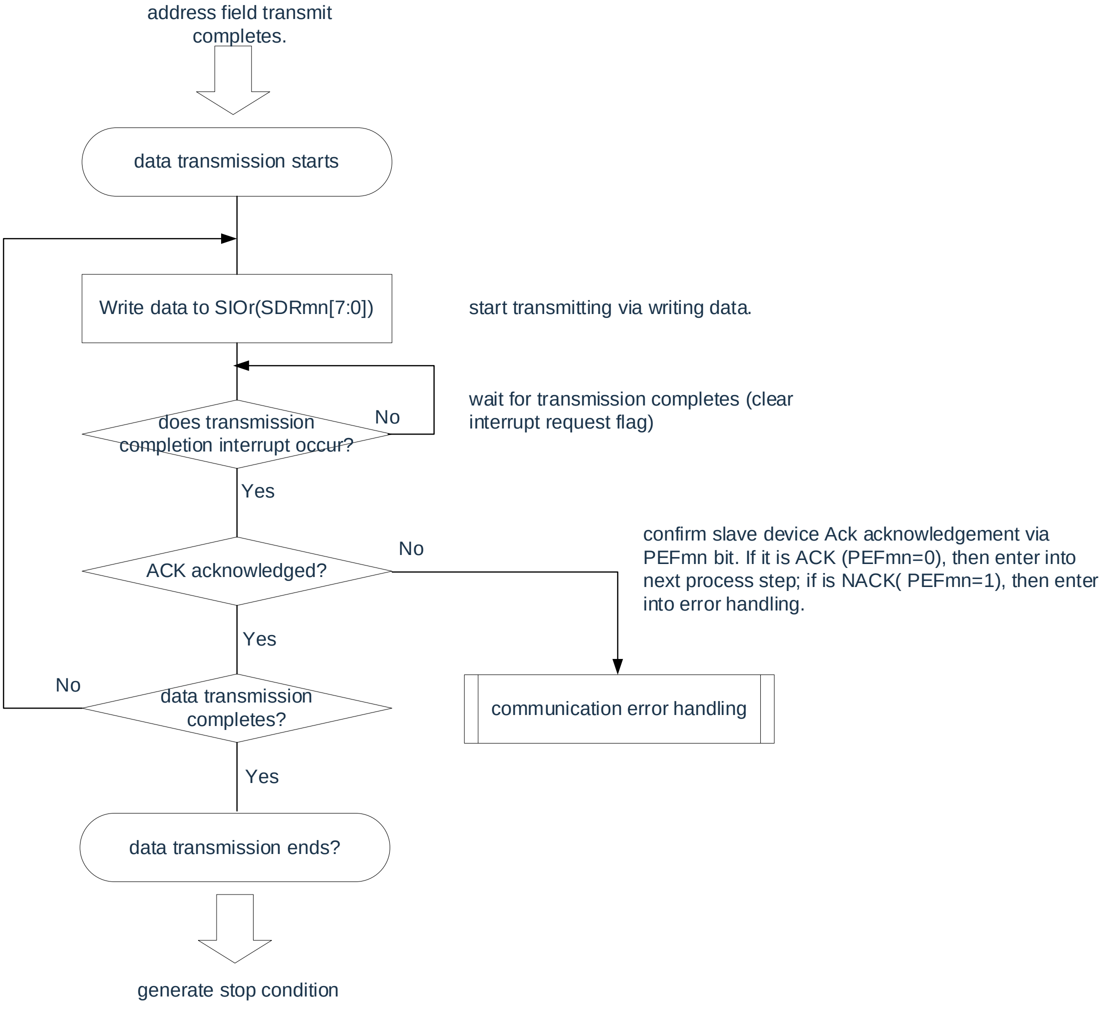


Figure 15-122 Flow chart for data transmssion


address field transmit completes.
data transmission starts
Write data to SIOr(SDRmn[7:0])
start transmitting via writing data.
does transmission completion interrupt occur?
No
Yes
wait for transmission completes (clear interrupt request flag)
ACK acknowledged?
No
Yes
confirm slave device Ack acknowledgement via PEFmn bit. If it is ACK (PEFmn=0), then enter into next process step; if is NACK(PEFmn=1), then enter into error handling.
communication error handling
data transmission completes?
No
Yes
data transmission ends?
generate stop condition

15.9.3 Data reception

Data reception is a run that receives data from a transfer object (slave) after sending an address segment. A stop condition is generated and the bus is released after receiving all the data from the object slave.

Simplified I²C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21
Object channel	SCI0's Channel 0	SCI0's Channel 1	SCI0's Channel 2	SCI0's Channel 3	SCI1's Channel 0	SCI1's Channel 1
Pin 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
Interrupt	INTEGER00	INTEGER01	INTEGER10	INTEGER11	INTEGER20	INTEGER21
	Only interrupt at that end of the transfer (no buffer interrupt can be selected).					
Error detection flag	Only the overflow error detection flag (OVFmn).					
Length of transmit data	8-bit					
Transfer Rate Note 2	Max.f_{MCK}/4 [Hz] (SDRmn [15:9]≥1) f_{MCK}: The operating clock frequency of the object channel, however, must meet the following conditions in each mode of I2C: · Max.1MHz · Max.400kHz (Quick Mode) · Max.100kHz (Standard Mode)					
Data level	Forward output (default: High level).					
Parity bit	No parity bits.					
Stop bit	Appending 1 bit (ACK send).					
Data orientation	MSB First					

Note: 1. To communicate through simplified I2C, the N-channel drain open output mode (POMxx=1) must be set through the port output mode register (POMxx). Refer to "2.3 Registers for controlling port function" and 2.5 Register settings when using the multiplexing function".

2. It must be used within the scope of the peripheral functional characteristics (refer to the data sheet) that meet this condition and satisfy the electrical characteristics.

Remark: m: Unit number (m=0,1,2) n: Channel number (n=0,1)

(1) Register settings

Figure 15-123 Example of register setting contents for simplified I²C data reception (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)

(a) Serial mode register mn(SMRmn) ...... This register is not operated during data transmission and reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn	CCSmn						STSmn		SISmn0				MDmn2	MDmn1	MDmn0
	0/1	0	0	0	0	0	0	0 Note1	0	0 Note1	1	0	0	1	0	0

(b) Serial communication operation setting register mn (SCRmn) ...... Bits other than the TXEmn bit and RXEmn bit are not operated during data transmission and reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn	RXEmn	DAPmn	CKPmn		EOCmn	PTCmn1	PTCmn0	DIRmn		SLCmn1	SLCmn0			DLSmn1	DLSmn0
	0	1	0	0	0	0	0	0	0	0	0 Note2	1	0	1	1 Note3	1

(c) Serial Data Register mn (SDRmn) (Low 8 bits: SIOr)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration Note4							0	virtual transmit data configuration (FFH)							

$\underbrace{\hspace{12em}}_{\text{SIOr}}$ (bits 7–0)

(d) Serial output register m (SOm) ...... This register is not operated during data sending and receiving.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm							CKOm1	CKOm0							SOm1	SOm0
	0	0	0	0	0	0	0/1 Note5	0/1 Note5	0	0	0	0	0	0	0/1 Note5	0/1 Note5

(e) Serial Output Enable Register m(SOEm) ...... This register is not operated during data sending and receiving.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm															SOEm1	SOEm0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0/1	0/1

(f) Serial channel start register m(SSm) ...... This register is not operated during data transmission and reception.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm															SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0/1	0/1

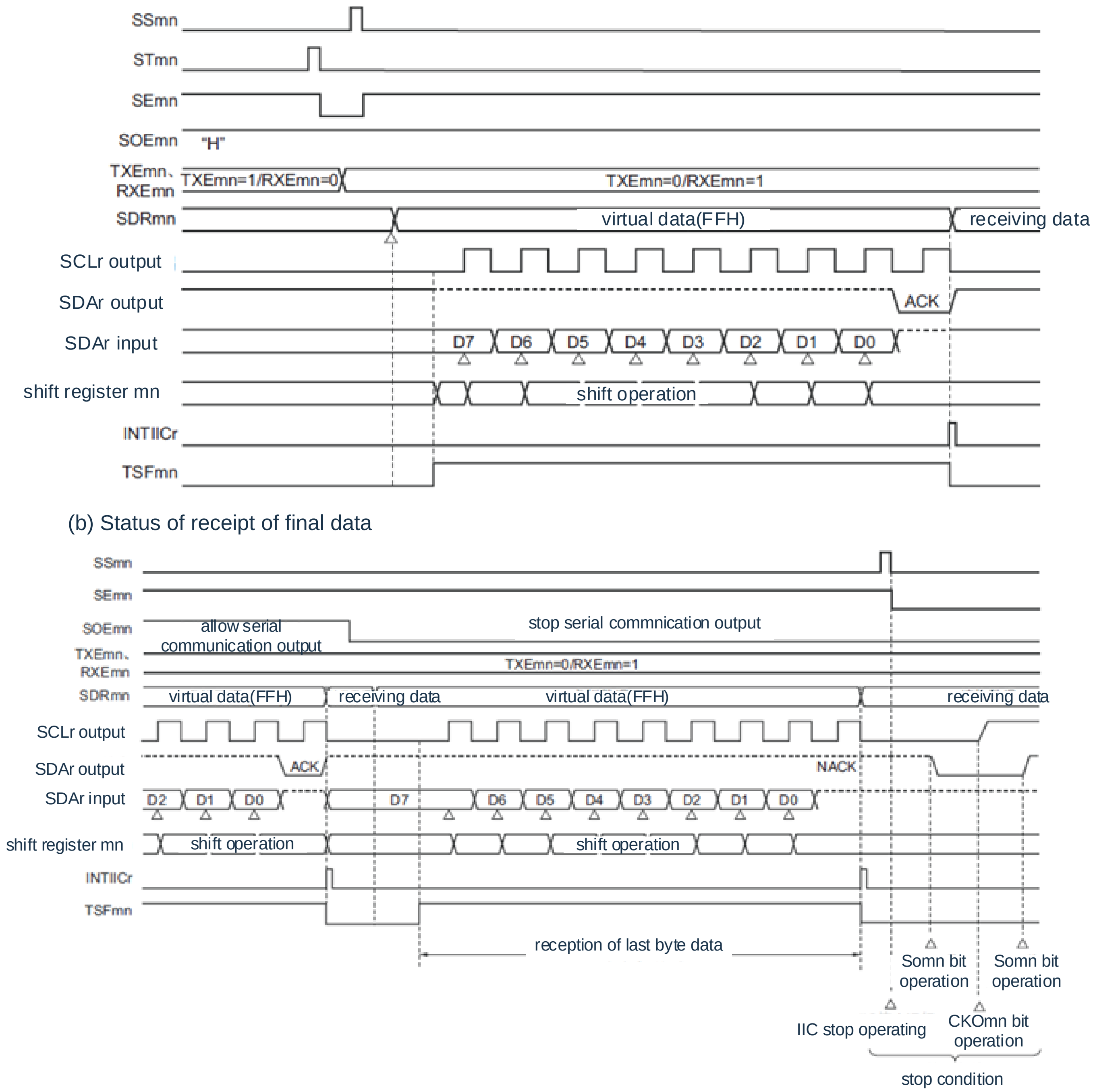
Note 1. Limited to SMR01, SMR11, SMR21 registers only.
2. Limited to SCR00, SCR10, SCR20 registers only.
3. Limited to SCR00, SCR01, SCR10, SCR11, SCR20, SCR21 registers, other fixed to "1".
4. No set-up is required because the address segment is already set when it is sent.
5. During a communication run, the value changes due to communication data.

Note 1.m: Unit number (m=0,1,2) n: Channel number (n=0,1) r: IIC number (r=00,01,10,11,20,21)
mn=00~03,10~11

2. □ : Set in IIC mode for Fixed. ▨ : Cannot set (set initial value).
×: This is a bit that cannot be used in this mode (set the initial value if not used in other modes).
0/1: The "0" or "1" is set according to the user.

(2) Process flow

Figure 15-124 Timing of data reception

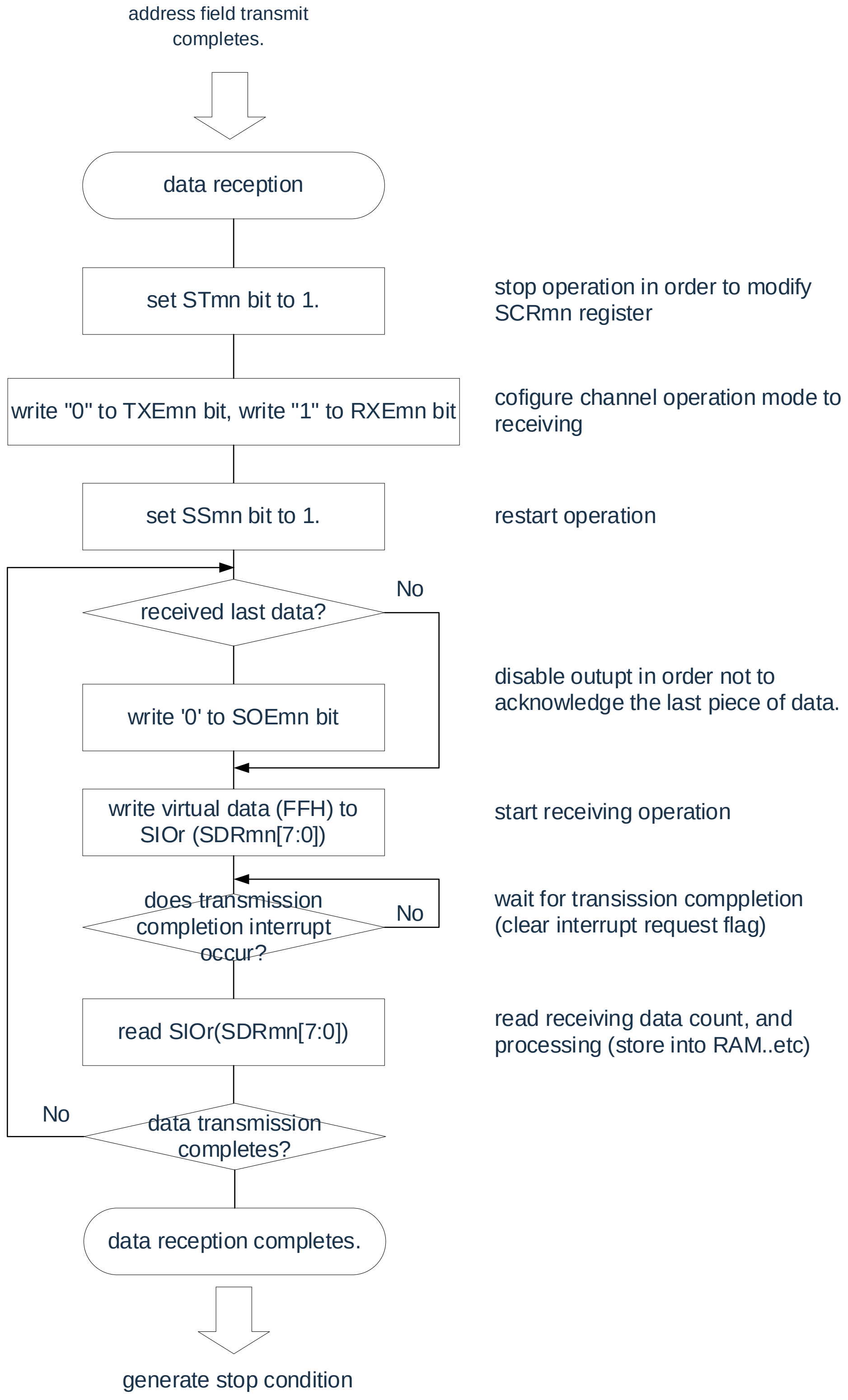


(a) Start of receiving data

(b) Status of receipt of final data

Remark: m: Unit number (m=0,1,2) n: Channel number (n=0,1) r: IIC number (r=00,01,10,11,20,21)

Figure 15-125 Flow chart of data reception


address field transmit completes.
data reception
set STmn bit to 1.
stop operation in order to modify SCRmn register
write "0" to TXEmn bit, write "1" to RXEmn bit
cofigure channel operation mode to receiving
set SSmn bit to 1.
restart operation
received last data?
No
write '0' to SOEmn bit
disable outuput in order not to acknowledge the last piece of data.
write virtual data (FFH) to SIOr (SDRmn[7:0])
start receiving operation
does transmission completion interrupt occur?
No
wait for transission comppletion (clear interrupt request flag)
read SIOr(SDRmn[7:0])
read receiving data count, and processing (store into RAM..etc)
No
data transmission completes?
data reception completes.
generate stop condition

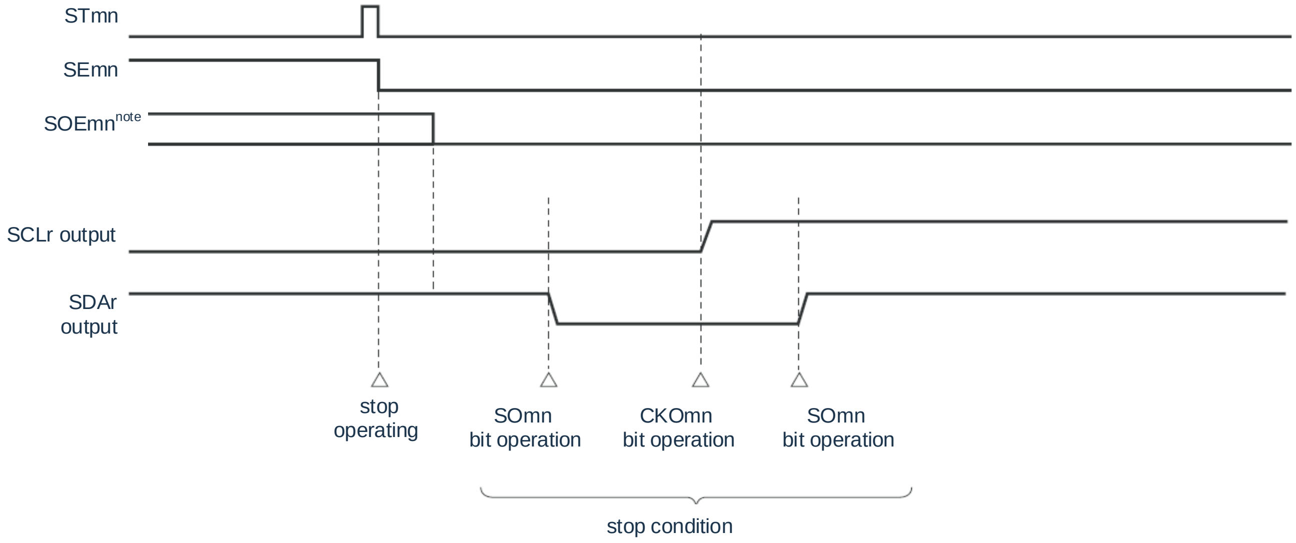
Note: ACK (NACK) is not output when the last data is received. Thereafter, operation is stopped first by setting the STmn bit of the serial channel stop register m (STm) to "1", and then a stop condition is generated to end communication.

15.9.4 Generation of stop condition

After sending and receiving all the data with the object slave, a stop condition is generated and the bus is released.

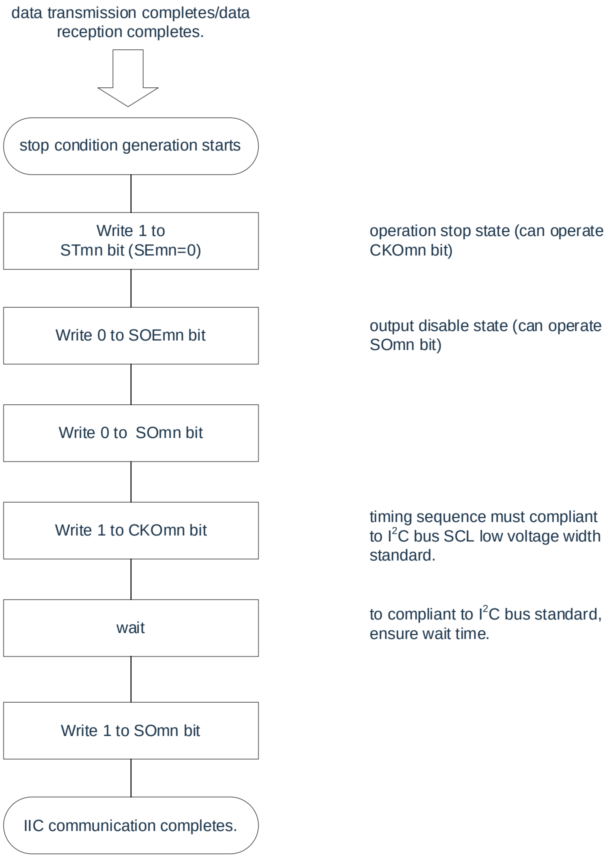
(1) Process flow

Figure 15-126 Timing diagram of generating stop condition



Note: The SOEmn bit of the serial output enable register m (SOEm) is set to "0" before the last data is received.

Figure 15-127 Flow chart of generating stop conditions


data transmission completes/data reception completes.
stop condition generation starts
Write 1 to STmn bit (SEmn=0)
operation stop state (can operate CKOmn bit)
Write 0 to SOEmn bit
output disable state (can operate SOmn bit)
Write 0 to SOmn bit
Write 1 to CKOmn bit
timing sequence must compliant to I2C bus SCL low voltage width standard.
wait
to compliant to I2C bus standard, ensure wait time.
Write 1 to SOmn bit
IIC communication completes.

15.9.5 Calculation of transfer rate

The transfer rate for simplified I2C (IIC00, IIC01, IIC10, IIC11, IIC20,IIC21) communication can be calculated using the following formula.

(Transfer rate) = {Operation clock frequency (f_{MCK}) } (SDRmn÷[15:9]+1) ÷2

Note Setting SDRmn[15:9] to '0000000B' is prohibited, and the setting value for SDRmn[15:9] must be greater than or equal to '0000001B. The duty ratio of the SCL signal output by the simplified I^{2}C is 50%. In I2C bus specification, the low level width of the SCL signal is greater than the high level width. Therefore, if 400kbps is set as a fast mode or 1Mbps is set as an enhanced fast mode, the low level width of the SCL signal output is less than the specification value of the I2C bus. You must set a value for SDRmn[15:9] that meets the I2C bus specification.

Note 1. Because the value of SDRmn[15:9] is the value of bit15~9 of the SDRmn (0000001B~1111111B), it is 1~127.
2.m: Unit number (m=0,1,2) n: Channel Number (n=0,1)

The operating clock (f_{MCK}) is determined by the bit15 (CKSmn bit) of the serial clock selection register m (SPSm) and the serial mode register mn (SMRmn).

Table 15-5 Simplfied I2C operating clock selection

SMRmn register	SPSm register								Runtime 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 above									Disable settings.	

Note To change the clock selected as fCLK (change the value of the System Clock Control Register (CKC), you must change after stopping Universal Serial Communication Unit (SCI) =000FH.

Note 1.X: Ignore

2.m: Unit number (m=0,1,2) n: Channel Number (n=0,1)

An example of setting the I²C transfer rate at f_{MCK}=f_{CLK}=32 MHz is shown below.

I²C transfer mode (Expected Transfer Rate)	f_{CLK}=32MHz			
	Runtime Clock (f_{MCK})	SDRmn [15:9]	Calculated transfer rate	Error with expected transfer rate
100kHz	$f_{CLK}/2$	79	100kHz	0.0%
400kHz	f_{CLK}	41	380kHz	5.0% Note
1MHz	f_{CLK}	18	0.84MHz	16.0% Note

Note The error cannot be set to '0'% because the SCL signal has a 50% duty cycle.

15.9.6 Procedure for handling errors during simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication

The processing steps when an error occurs during a simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication are shown in Figures 15-128 and 15-129.

Figure 15-128 Handling steps when overflow errors occur

software operation	Hardware Status	Comments
Read the serial data register mn	The BFFmn bit of the SSRmn register is "0" and the channel n is in a receiver state.	This is to prevent an overflow error from occurring to end the next receipt during error handling.
Read the serial status register mn (SSRmn).		The type of error is determined and the read value is used to clear the error flag.
Clear trigger register mn for serial flag	Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors in the read operation can only be cleared.

Figure 15-129 Processing steps when an ACK error occurs in a simplified I²C mode

software operation	Hardware Status	Comments
Read the serial status register mn (SSRmn).		Determine the type of error and the read value is used to clear the error flag.
write serial flag clear trigger register mn	Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors in the read operation can only be cleared.
Set the STmn bit of the serial channel stop register m (STm) to "1".	The serial channel allows the SEmn bit of the status register m (SEm) to be "0" and channel n to be running stopped.	The slave device is not ready to receive because no ACK is returned. Accordingly, a stop condition is generated and the bus is released, communication is started again from the start condition, or a restart condition can also be generated and restarted from the address transmission.
Generate a stop condition.		
Generate start condition.		
Set the SSmn bit of the serial channel start register m (SSm) to "1".	The serial channel allows the SEmn bit of the status register m (SEm) to be "1" and channel n to be operational.	

Remark: m: Unit number (m=0,1,2) n: Channel number (n=0,1) r: IIC number (r=00,01,10,11,20,21)

mn=00~03,10~11

Chapter 16 Serial Interface SPI

16.1 Function of SPI

This product is equipped with two serial interfaces SPI0, SPI1, with the following 2 modes.

(1) Run stop mode

This is a mode used when serial transfer is not in progress and reduces power consumption.

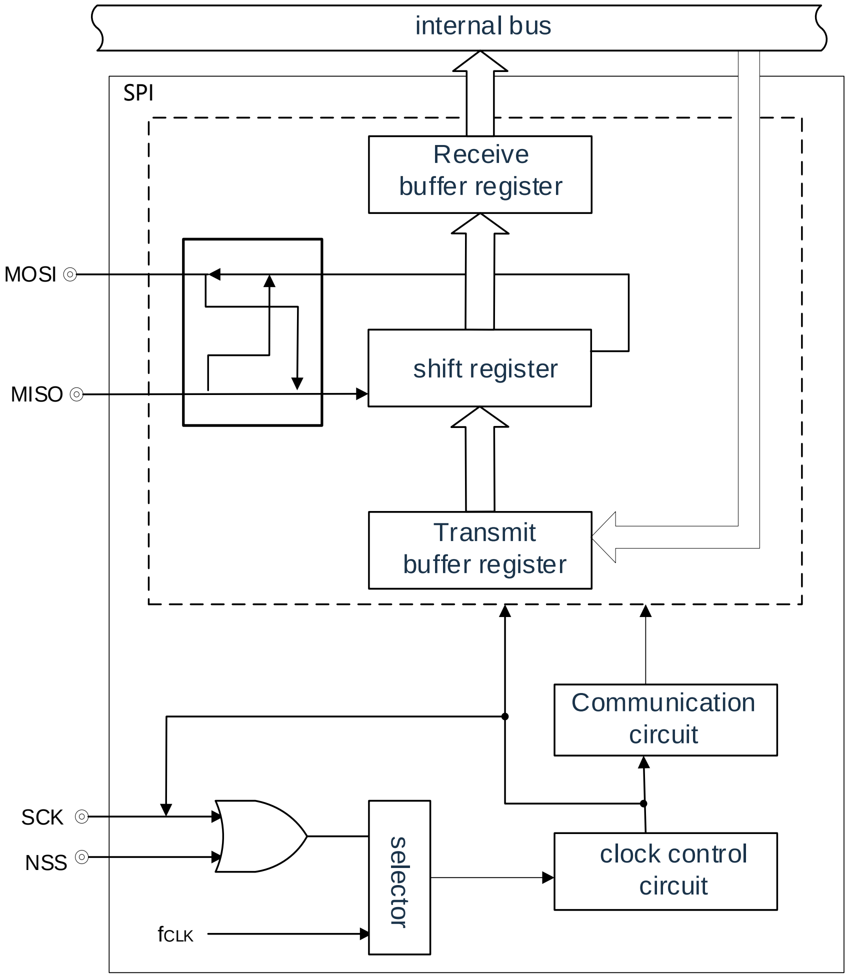
(2) 3-wire serial I/O mode

This mode performs 8-bit or 16-bit data transfer with multiple devices via 3 lines of the serial clock (SCKn) and serial data bus (MISOn and MOSIn).

Remark: n=0, 1

16.2 Structure of SPI

Figure 16-1 Diagram of SPI



16.3 Registers for controlling SPI

The SPI is controlled by the following registers.

- Peripheral enable register 1 (PER1)
- Serial operation mode register (SPIMn)
- Serial clock select register (SPICn)
- Transmit buffer register (SDROn)
- Receive buffer register (SDRIn)
- Port mode register (PMxx)
- Port mode control register (PMCxx)
- Port register (Pxx)

Remark: n=0, 1

16.3.1 Peripheral enable register 1 (PER1)

The PER1 register is a register that sets to enable or disable clocking to each peripheral hardware. It reduces power consumption and noise by stopping the clocking of unused hardware.

To use the SPI function, SPInEN must be set to "1".

For details, see "4.3.8 Peripheral Enable Registers 0, 1, 2 (PER0, PER1, PER2)".

Remark: n=0, 1

16.3.2 SPI operation mode register (SPIM)

SPIM is used to select the operation mode and control the operation enable or disable. SPIMn can be set by an 8-bit memory operation instruction.

A reset signal is generated to clear this register to 00H.

Figure 16-2 Format of SPI operation mode register(SPIM)

Address: SPI0:0x40042400 SPI1:0x40042800 After reset: 00H R/W Note1

Symbol	7	6	5	4	3	2	1	0
SPIMn	SPIEn	TRMDn	NSSEn	DIRn	INTMDn	DLSn	RECMDn	-

SPIEn	SPI running enable
0	Stop running.
1	Enable to run.

TRMD[Note3]	Transmit/Receive mode control
0	Receive mode
1	Transmit/Receive mode

NSSEn[Note4]	NSS pin usage 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	End-of-transmission interrupt
1	Transmit buffer null break

DLSn	Setting of the data length
0	8 bits of data length
1	16-bit of 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 bit to 1, fix the NSS pin input level to 0 or 1.

4. n=0, 1

16.3.3 SPI clock selection register (SPICn)

This register specifies the timing of data send/receive and sets the serial clock.

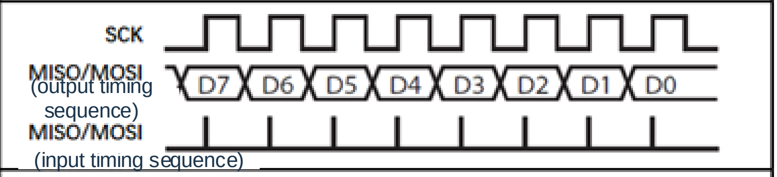
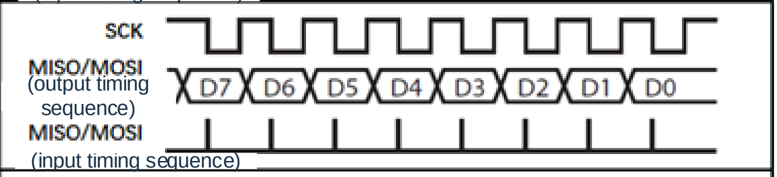
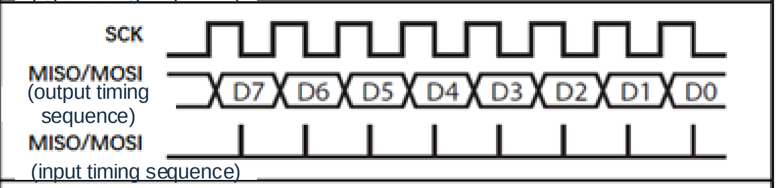
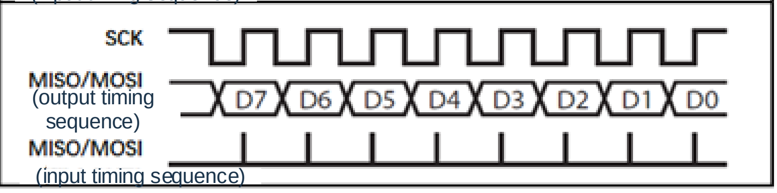
It can be set by an 8-bit storage operation instruction.

A reset signal is generated to clear the register to 01H.

Figure 16-3 Format of clock selection register (SPICn)

Address: SPI0:0x40042404 SPI1:0x40042804 After reset: 01H R/W

Symbol	7	6	5	4	3	2	1	0
SPICn	0	0	0	CPOLn	CPHAn	CKS2n	CKS1n	CKS0n

CPOL	CPHA	data transmit /Specify the receiving timing
0	0	 SCK MISO/MOSI (output timing sequence) MISO/MOSI (input timing sequence) D7 D6 D5 D4 D3 D2 D1 D0
0	1	 SCK MISO/MOSI (output timing sequence) MISO/MOSI (input timing sequence) D7 D6 D5 D4 D3 D2 D1 D0
1	0	 SCK MISO/MOSI (output timing sequence) MISO/MOSI (input timing sequence) D7 D6 D5 D4 D3 D2 D1 D0
1	1	 SCK MISO/MOSI (output timing sequence) MISO/MOSI (input timing sequence) D7 D6 D5 D4 D3 D2 D1 D0

CKS2n	CKS1n	CKS0n	SPI serial clock selection	Mode
0	0	0	F_{CLK}	Master mode
0	0	1	$F_{CLK}/2$	
0	1	0	$F_{CLK}/2^2$	
0	1	1	$F_{CLK}/2^3$	
1	0	0	$F_{CLK}/2^4$	
1	0	1	$F_{CLK}/2^5$	
1	1	0	$F_{CLK}/2^6$	
1	1	1	An external clock input from SCK	Slave mode

Notice: 1. Writing to SPICn is disabled when SPIEn=1 (operation enable).

2. The phase type of the data clock after reset is type 1.

Remark: n=0, 1

16.3.4 SPI status register (SPISn)

The SPIT register is used to acknowledge the communication status of the SPI. SPISn can be read by an 8-bit memory operation instruction.

A reset signal is generated to clear this register to 00H.

Figure 16-4 Format of SPI status register (SPISn)

Address: SPI0:0x40042410 SPI1:0x40042810 After reset: 00H R

Symbol	7	6	5	4	3	2	1	0
SPITn	-	-	-	-	-	-	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 SDRIF is read, the bit is cleared to 0

SPTFn[Note1]	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

16.3.5 Transmit buffer register (SDROn)

This register sets the transmit data.

When bit 7 (SPIEn) and bit 6 (TRMDn) of the serial operation mode register (SPIMn) are set to 1, transmit/receive is started by writing data to SDROn.

The serial I/O shift register converts the data in SDROn from parallel data to serial data and outputs it to the serial output pin.

SDROn can be written to or read from with 8-bit or 16-bit memory operation instructions.

A reset signal is generated to clear this register to 0000H.

Figure 16-5 Format of transmit buffer register (SDROn)

Address: SPI0:0x40042408 SPI1:0x40042808 After reset: 0000H R/W

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

16.3.6 Receive buffer register (SDRIn)

his register stores the received data.

If bit 6 (TRMDn) of the serial operation mode register (SPIMn) is set to 0, reception is started by reading data from SDRI.

During reception, the data is read from the serial input pin into SDRIn.

SDRIn can be read with 8-bit or 16-bit memory operation instructions.

A reset signal is generated to clear this register to 0000H.

Figure 16-6 Format of receive buffer register (SDRIn)

Address: SPI0:0x4004240C SPI1:0x4004280C After reset: 0000H R

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

Remark: n=0, 1

16.3.7 Registers for controlling SPI port functions

When using SPI, you must set the control register (Port Mode Register (PMxx, PMCxx)) for the port function that is multiplexed with the SPI input and output pins. For details, refer to "2.3.1 Port Mode Registers (PMxx)".

When using the SPI pin multiplexed port as an output of SCK/SO/MO, the corresponding Port Mode Register (PMxx, PMCxx) of each port must be set to bit "0". When the multiplexed port of SPI pin is used as the input of SCK/SI/MI, the bit "1" of the Port Mode Register (PMxx) and the bit "0" of PMCxx corresponding to each port must be set. In this case, the bit of the port register (Pxx) can be "0" or "1". For details, refer to "2.5 Register Settings when Using Multiplexing Functions".

16.4 Operation of SPI

In 3-wire serial I/O mode, data is sent or received by 8-bit or 16-bit. The data is sent or received synchronously with the serial clock.

After communication begins, bit 0 (SPTFn) of SPITn 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.

Notice

1. When SPTFn=1 (during serial communication), access to control registers and data registers is prohibited.

2. It must be used within the range that satisfies the SCLK Cycle Time (tKCY) characteristics. For details, please refer to the data sheet.

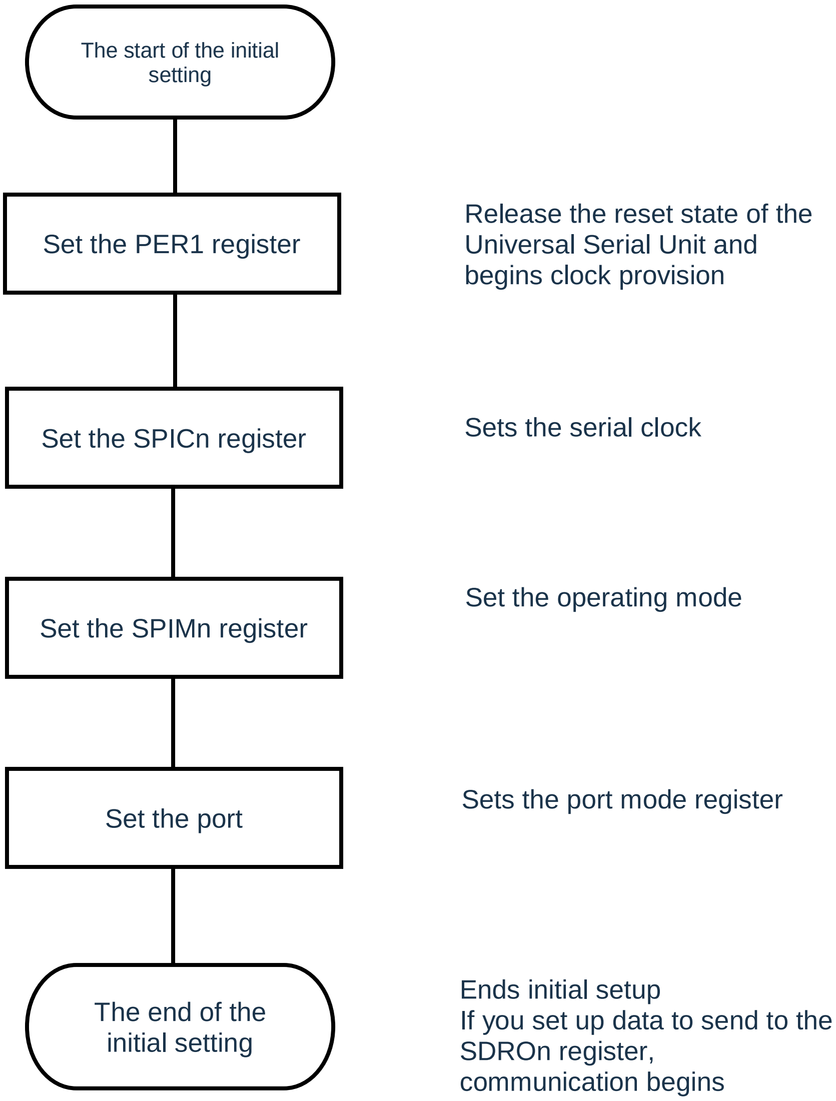
Remark: n=0, 1

16.4.1 Master tramission and reception

If the bit 6 (TRMDn) of the serial operation mode register (SPIMn) is 1, data can be sent or received. When a value is written to the transmit buffer register (SDROn), send/receive starts.

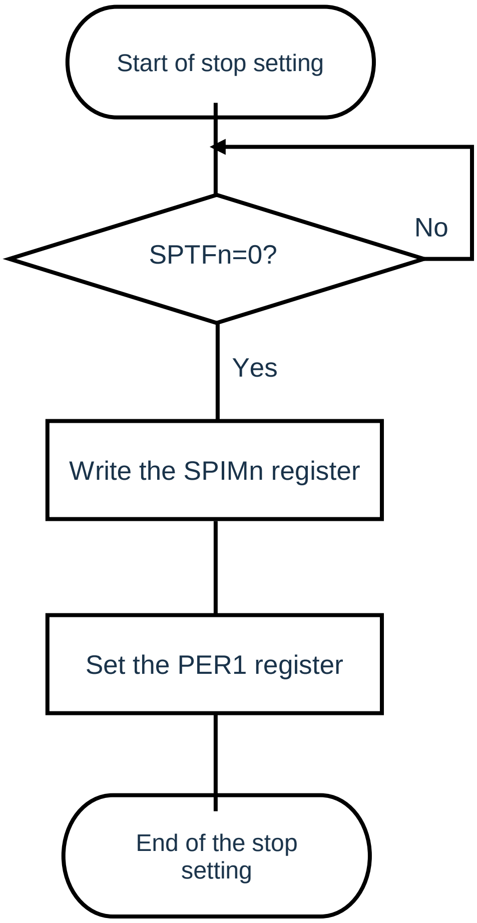
（1） Procedure

Figure 16-7 Initial setup steps for master transmission/reception


The start of the initial setting
Set the PER1 register
Release the reset state of the Universal Serial Unit and begins clock provision
Set the SPICn register
Sets the serial clock
Set the SPIMn register
Set the operating mode
Set the port
Sets the port mode register
The end of the initial setting
Ends initial setup
If you set up data to send to the SDROn register, communication begins

Remark: n=0, 1

Figure 16-8 Stop steps for transmission/reception



If there is data being transferred, wait for the transfer to end

Set the SPIEn bit to "0" and stop the operation of SPIn

To use deep sleep mode, stop the clock of the SPI unit and set the reset state

Remark: n=0, 1

(2) Process flow

Figure 16-9 Timing diagram of transmission/reception (single transmit mode) (INTMD=0,CPHA=1, CPOL=1)

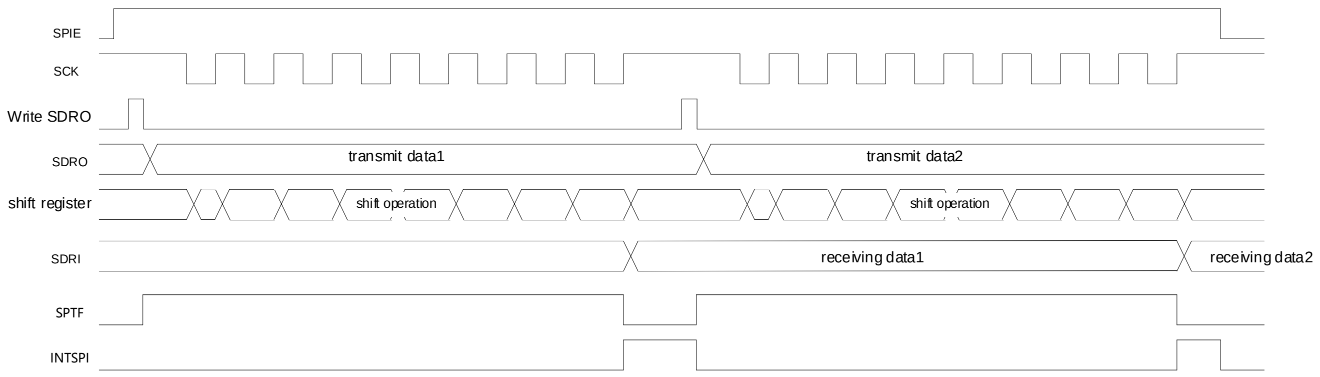
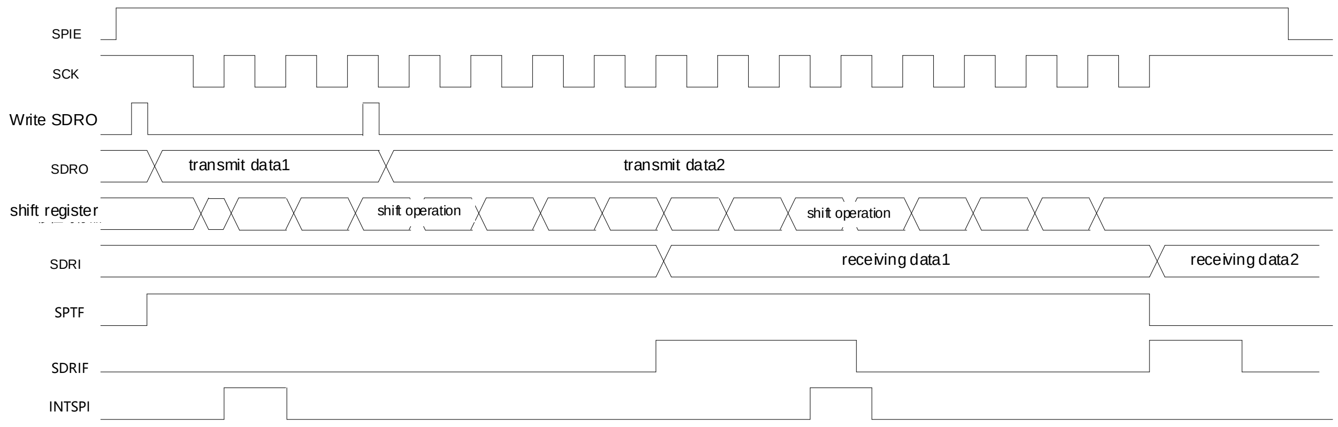


Figure 16-10 Timing diagram of transmission/reception (continuous transmit mode) (INTMD=1,CPHA=1, CPOL=1)

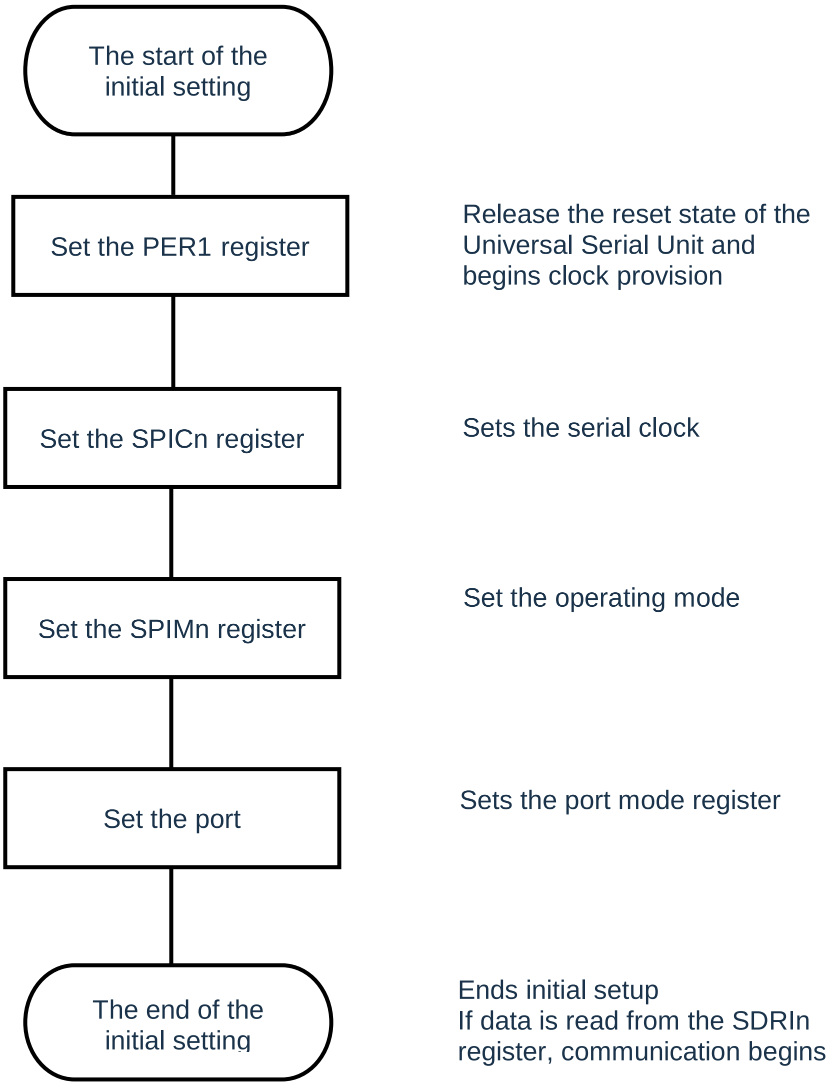


16.4.2 Master reception

If bit 6 (TRMDn) of the serial operation mode register (SPIMn) is 0, only data can be received. When data is read from the receive buffer register (SDRIn), reception starts.

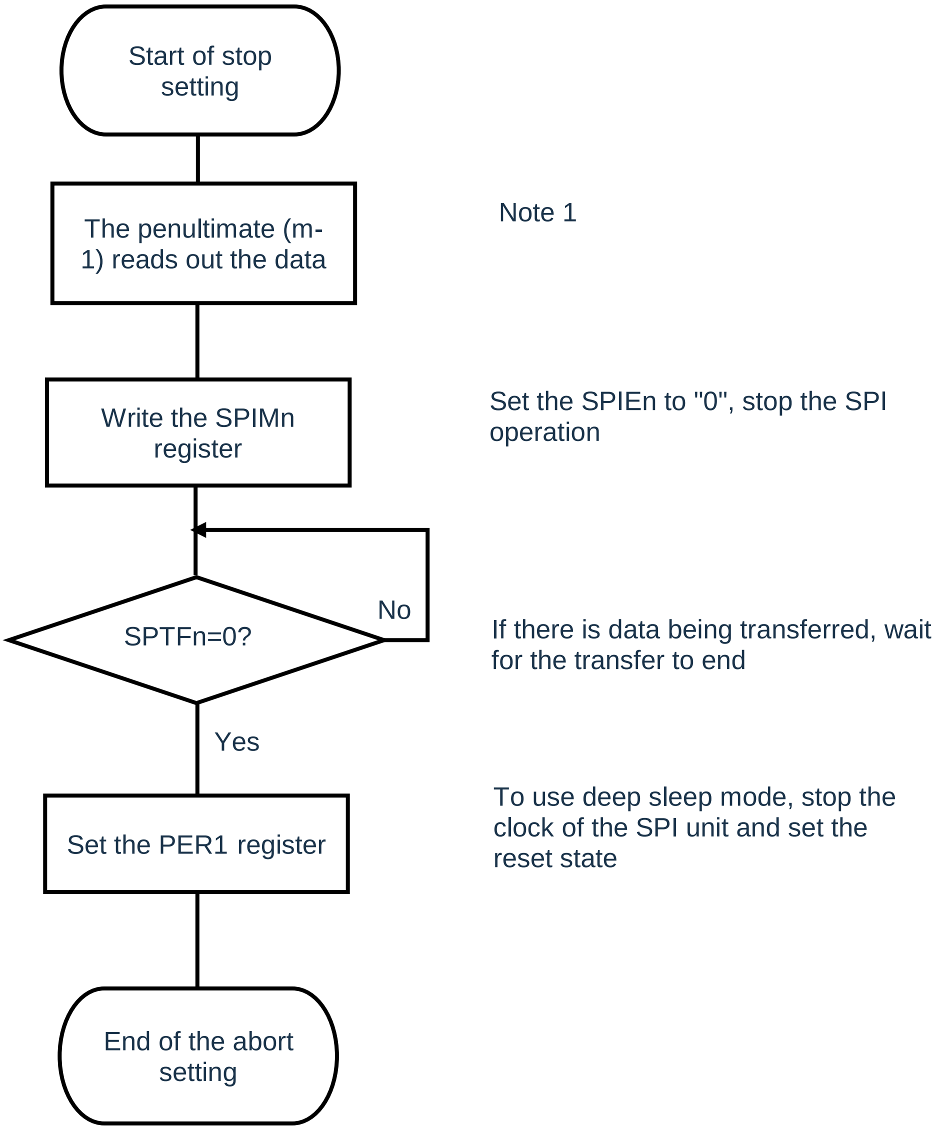
（1） Procedure

Figure 16-11 Initial setup steps for master reception


The start of the initial setting
Set the PER1 register
Release the reset state of the Universal Serial Unit and begins clock provision
Set the SPICn register
Sets the serial clock
Set the SPIMn register
Set the operating mode
Set the port
Sets the port mode register
The end of the initial setting
Ends initial setup
If data is read from the SDRIn register, communication begins

Remark: n=0, 1

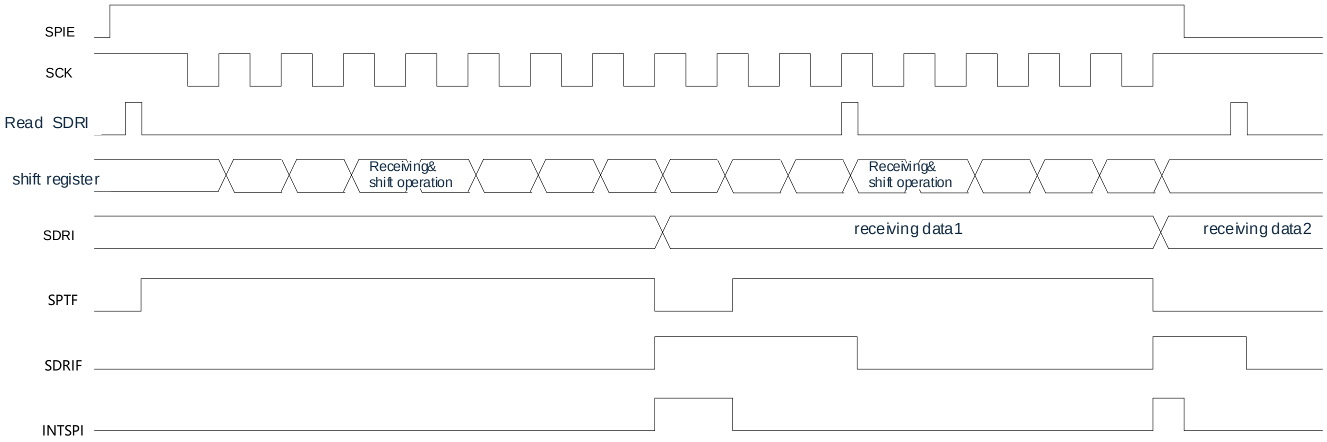
Figure 16-12 Stop steps for master reception


Start of stop setting
The penultimate (m-1) reads out the data
Note 1
Write the SPIMn register
Set the SPIEn to "0", stop the SPI operation
SPTFn=0?
No
If there is data being transferred, wait for the transfer to end
Yes
Set the PER1 register
To use deep sleep mode, stop the clock of the SPI unit and set the reset state
End of the abort setting

Remark: n=0, 1

(2) Process flow

Figure 16-13 Timing diagram of reception (CPHA=1, CPOL=1)

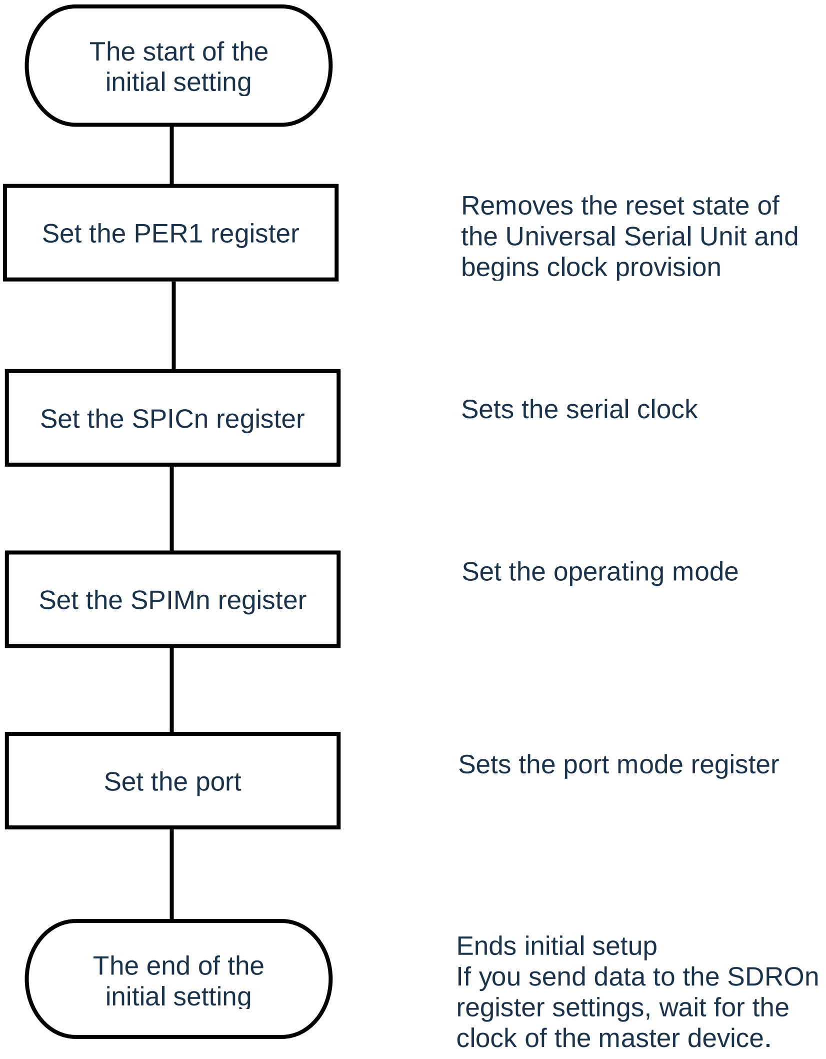


16.4.3 Slave transmission and reception

If bits CKS2-0 of the serial clock selection register (SPICn) select slave mode, bit 6 (TRMD) 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 (SDROn), wait for the clock of the master device to start sending/receiving.

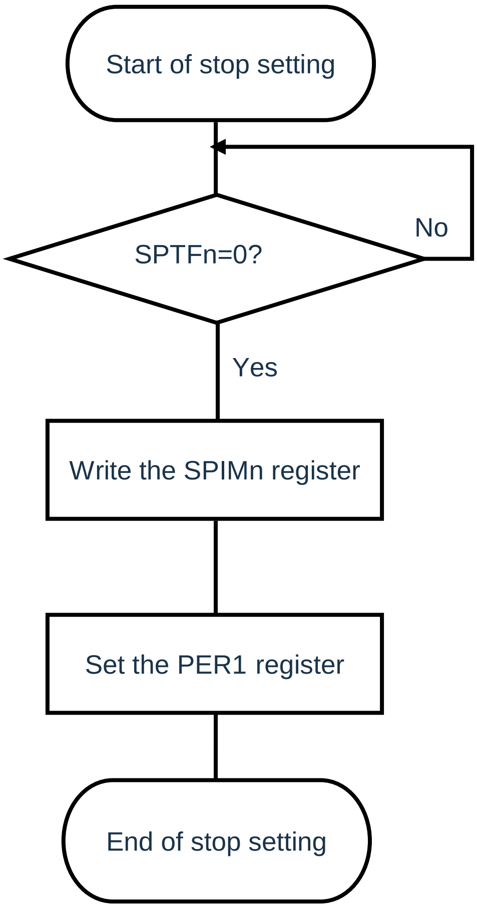
（1） Procedure

Figure 16-14 Initial setup steps for slave transmission/reception


The start of the initial setting
Set the PER1 register
Removes the reset state of the Universal Serial Unit and begins clock provision
Set the SPICn register
Sets the serial clock
Set the SPIMn register
Set the operating mode
Set the port
Sets the port mode register
The end of the initial setting
Ends initial setup
If you send data to the SDROn register settings, wait for the clock of the master device.

Remark: n=0, 1

Figure 16-15 Stop steps for slave transmission/reception



If there is data being transferred, wait for the transfer to end

Set the SPIEn bit to "0" to stop the operation of the SPI

To use deep sleep mode, stop the clock of the SPI unit and set the reset state

Remark: n=0, 1

(2) Process flow

Figure 16-16 Timing diagram of transmission/reception (single transmit mode) (INTMD=0,CPHA=1, CPOL=1)

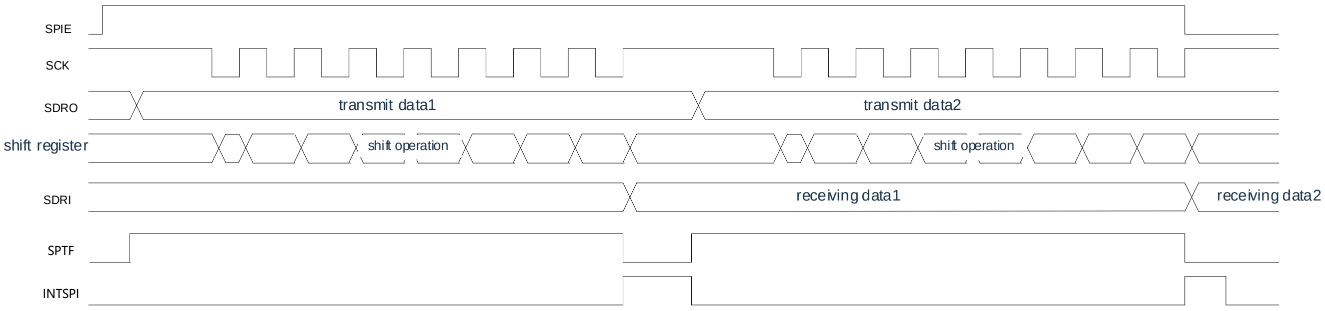
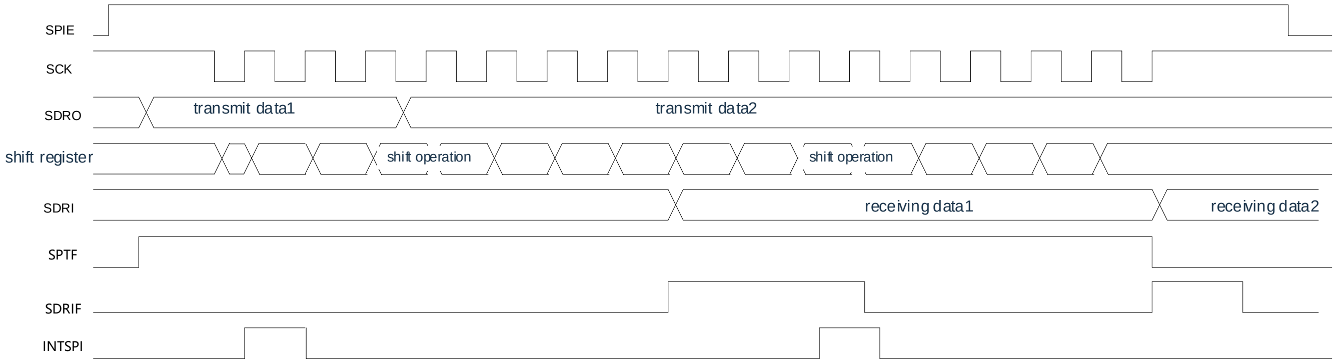


Figure 16-17 Timing diagram of transmission/reception (continuous transmit mode) (INTMD=1, CPHA=1, CPOL=1)

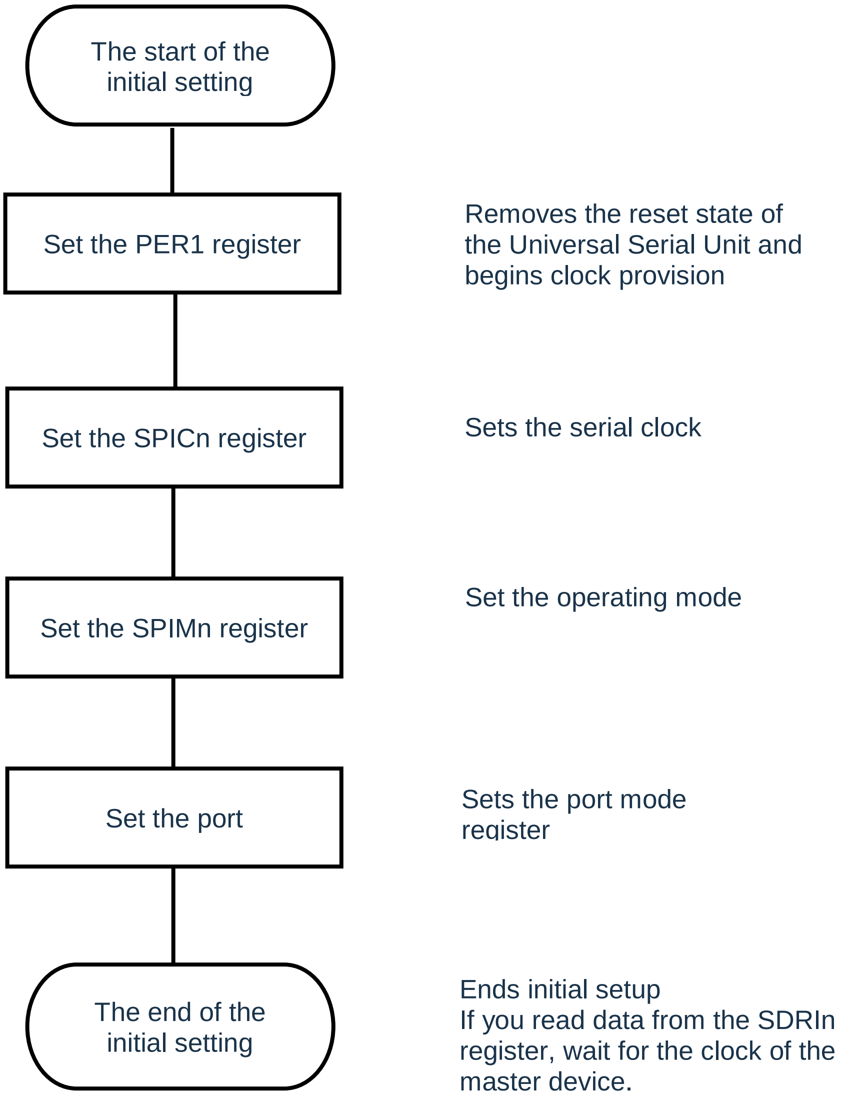


16.4.4 Slave reception

If bit CKS2-0n of the serial clock select register (SPICn) selects slave mode and bit 6 (TRMDn) of the serial operation mode register (SPIMn) is 0, the slave receive mode is entered. When data is read from the receive buffer register (SDRIn), wait for the clock of the master device and start receiving.

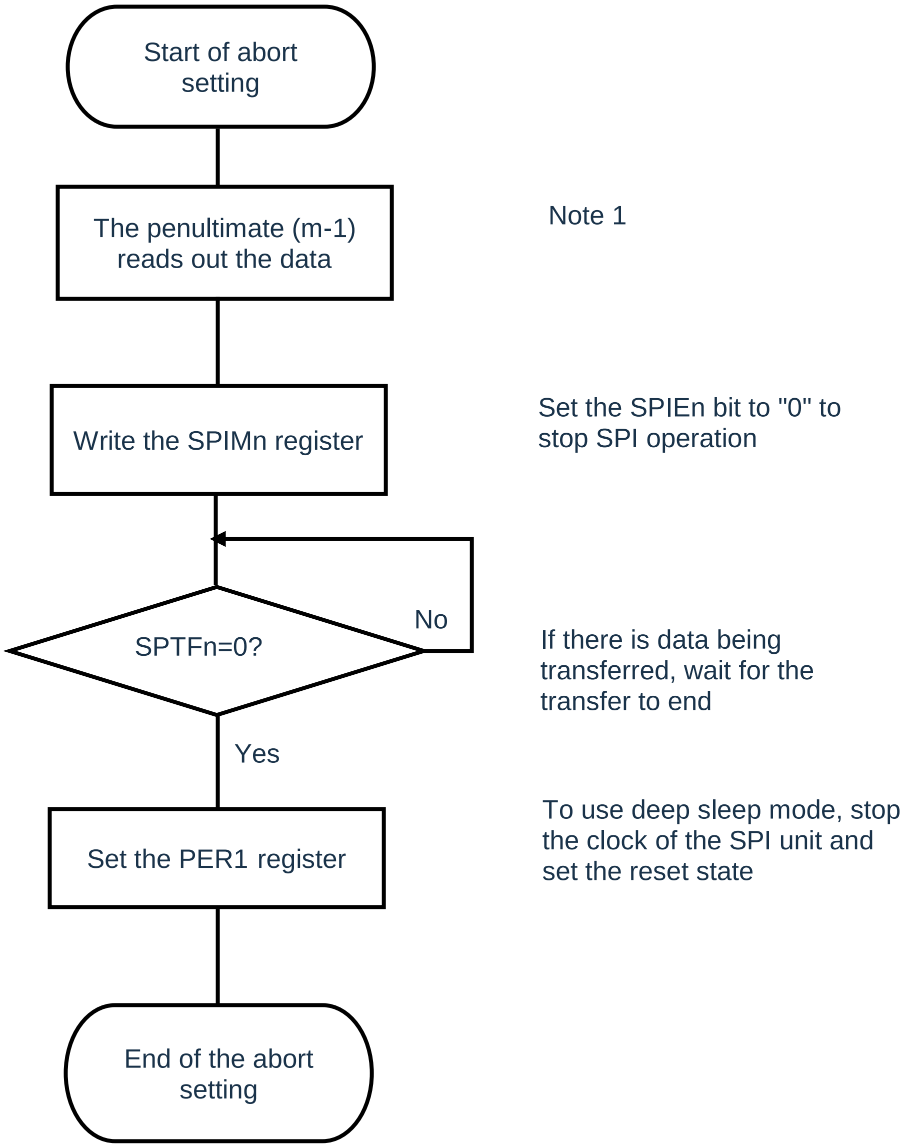
（1） Procedure

Figure 16-18 Initial setup steps for slave reception


The start of the initial setting
Set the PER1 register
Removes the reset state of the Universal Serial Unit and begins clock provision
Set the SPICn register
Sets the serial clock
Set the SPIMn register
Set the operating mode
Set the port
Sets the port mode register
The end of the initial setting
Ends initial setup
If you read data from the SDRIn register, wait for the clock of the master device.

Remark: n=0, 1

Figure 16-19 Stop steps for slave reception

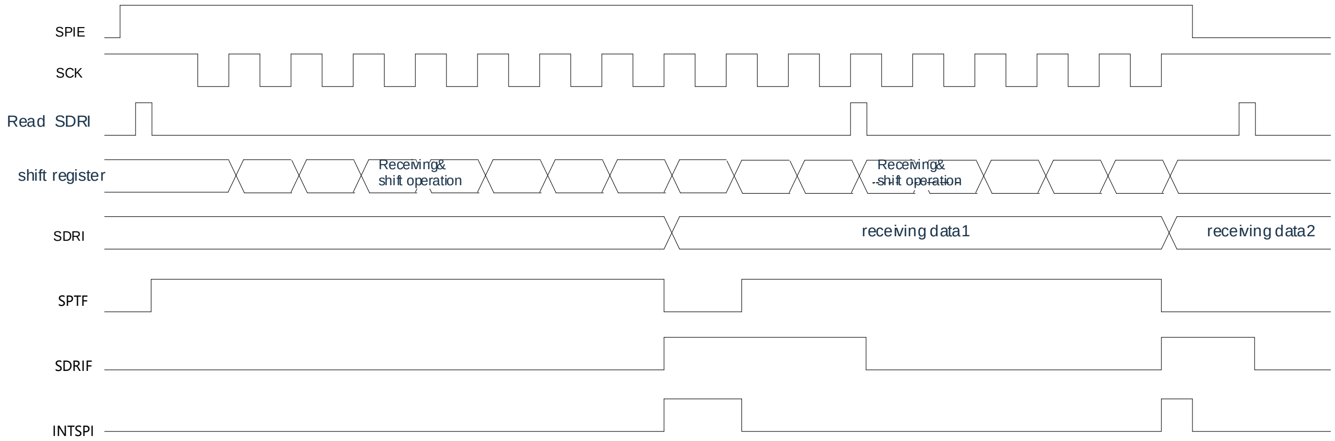

Start of abort setting
The penultimate (m-1) reads out the data
Note 1
Write the SPIMn register
Set the SPIEn bit to "0" to stop SPI operation
SPTFn=0?
No
If there is data being transferred, wait for the transfer to end
Yes
Set the PER1 register
To use deep sleep mode, stop the clock of the SPI unit and set the reset state
End of the abort setting

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 SPIEn after waiting for a SCK cycle after the penultimate data readout. The transfer of the SPI will be aborted after the last data transfer is complete.

Remark: n=0, 1

(2) Processing

Figure 16-20 Timing diagram of reception (CPHA=1, CPOL=1)



Chapter 17 QUAD SPI

17.1 Overview

The QUAD SPI Interface Module (QSPI for short) is a memory controller for connecting serial ROMs (non-volatile memories such as serial Flash, serial EEPROM or serial FeRAM) that have an SPI-compatible interface.

Table 17-1 QSPI Specification

Parameters	Specification
Number of channels	1 channel
	• SPI supports Extended SPI, DUAL SPI and QUAD SPI protocols • Can be configured for SPI mode 0 and SPI mode 3 • Address width can be 8, 16, 24, or 32 bits
Timing adjustment function	Supports multiple serial flash configurations
	• Flash read function It supports read, fast read, dual output fast read, dual I/O fast read, quad output fast read and quad I/O fast read instructions • Alternative instruction codes • Adjustable number of dummy cycles • Pre-read function • Polling processing • SPI bus period expansion
Direct communication function	It supports multiple serial flash instructions and functions through software control, including erase, write, ID read, and power-down control
Interrupt source	Error interruption
Module stop function	Module stop status can be set

Figure 17-1 Block diagram of QSPI

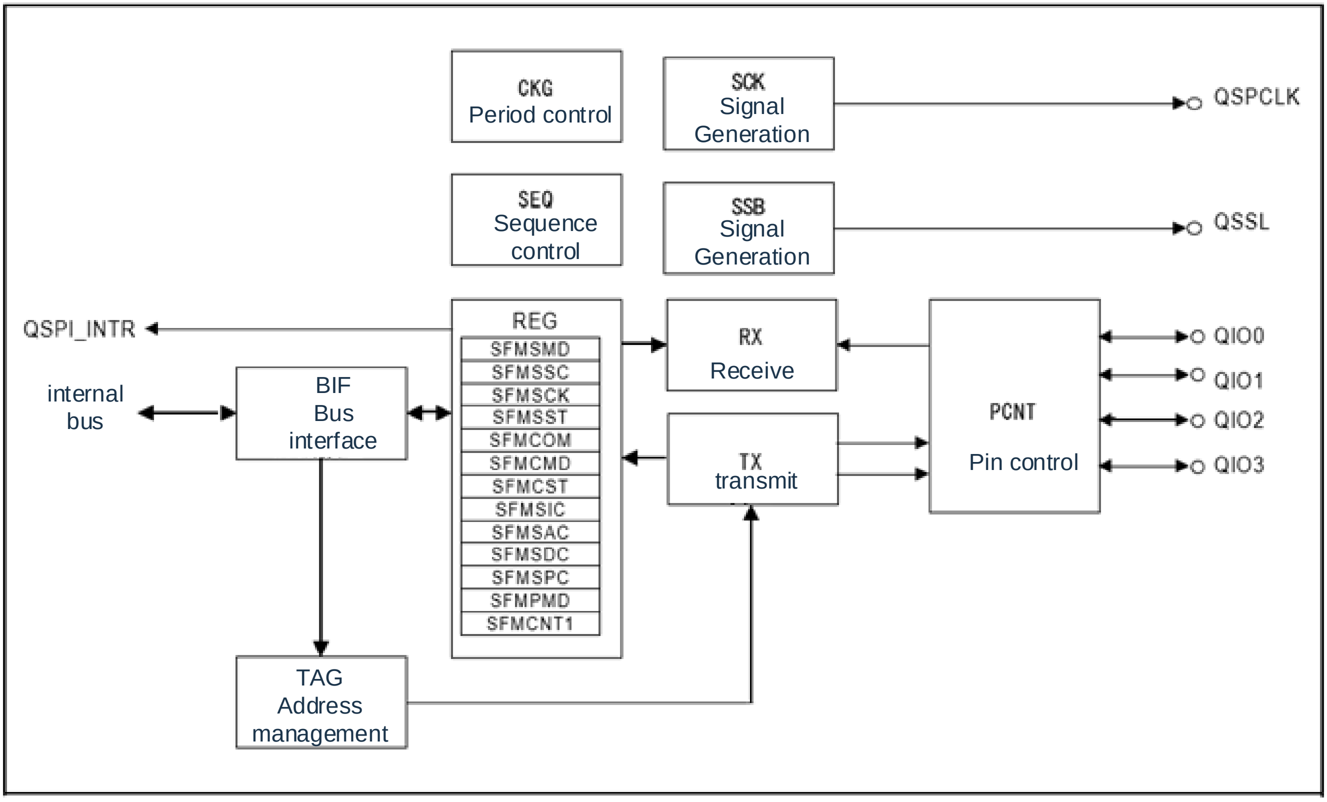


Table 17-2 QSPI I/O pin

Pin Name	I/O	Function
QSPCLK	Output	QSPI clock output pins
QSSL	Output	QSPI slave select pins
QIO0	I/O	Data 0 I/O
QIO1	I/O	Data 1 I/O
QIO2	I/O	Data 2 I/O
QIO3	I/O	Data 3 I/O

17.2 Register description

Register list:

Base Address	Offset Address	Register Name	R/W	Reset Value
0x64000000	0x000	SFMSMD	R/W	00000000h
	0x004	SFMSSC	R/W	00000037h
	0x008	SFMSKC	R/W	00000006h
	0x00C	SFMSST	R/W	00000080h
	0x010	SFMCOM	R/W	00000000h
	0x014	SFMCMD	R/W	00000000h
	0x018	SFMCST	R/W	00000000h
	0x020	SFMCIC	R/W	00000000h
	0x024	SFMSAC	R/W	00000002h
	0x028	SFMSDC	R/W	0000ff00h
	0x030	SFMSPC	R/W	00000010h
	0x034	SFMPMD	R/W	00000000h
	0x804	SFMCNT1	R/W	00000000h

17.2.1 Transmit mode control register (SFMSMD)

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SFMCC E	—	—	—	SFMOSW	SFMOHW	SFMOEX	SFMMD3	SFMPAE	SFMPFE	SFMSE[1:0]		—	SFMRM[2:0]		
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b2~b0	SFMRM[2:0]	Serial interface read mode selection	b2~b0 000:Standard read 001:Fast read 010:Fast read dual output 011:Fast read dual I/O 100:Fast read quad output 101:Fast read quad I/O 110:Prohibit setting 111:Prohibit setting	R/W
b3	-	Reserved	Write 0 only, read out value is 0	-
b5，b4	SFMSE[1:0]	After SPI bus access QSSL extension function selection	b5，b4 00: Do not extend QSSL 01: Extend QSSL by 33 QSPCLK 10: Extend QSSL by 129 QSPCLK 11: Extend QSSL indefinitely	R/W
b6	SFMPFE	Pre-read function selection	0: Pre-read is prohibited 1: Pre-read is allowed	R/W
b7	SFMPAE	Stop pre-reading at locations other than byte boundaries	0: Disable the function 1: Enable the function	R/W
b8	SFMMD3	SPI Mode Selection	0: SPI mode 0 1: SPI mode 3	R/W
b9	SFMOEX	Serial port pin-out permission Extension options for signals	0: Prohibit extension 1: Extend 1 QSPCLK	R/W
b10	SFMOHW	Serial communication hold time adjustment	0: Do not extend the high level width of QSPCLK during communication 1: Extend the high level width of QSPCLK by one PCLKA during communication	R/W
b11	SFMOSW	Serial communication establishment time adjustment	0: Do not extend the low level width of QSPCLK during communication 1: Extend the low level width of QSPCLK by one PCLKA during communication	R/W
b14~b12	-	Reserved	Write 0 only, read out value is 0	-
b15	SFMCCE	Read command code selection	0: Default instruction code for each instruction setting 1: Instruction code written to SFMSIC register	R/W
b31~b16	-	Reserved	Write 0 only, read out value is 0	-

17.2.2 Chip select control register (SFMSSC)

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	SFMSLD	SFMSHD	SFMSW			
Reset value:	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1

Bit	Symbol	Bit Name	Description	R/W
b3~b0	SFMSW	QSSL signal minimum high level width selection	b3~b0 0 0 0 0: 1 QSPCLK 0 0 0 1: 2 QSPCLK 0 0 1 0: 3 QSPCLK 0 0 1 1: 4 QSPCLK 0 1 0 0: 5 QSPCLK 0 1 0 1: 6 QSPCLK 0 1 1 0: 7 QSPCLK 0 1 1 1: 8 QSPCLK 1 0 0 0: 9 QSPCLK 1 0 0 1: 10 QSPCLK 1 0 1 0: 11 QSPCLK 1 0 1 1: 12 QSPCLK 1 1 0 0: 13 QSPCLK 1 1 0 1: 14 QSPCLK 1 1 1 0: 15 QSPCLK 1 1 1 1: 16 QSPCLK.	R/W
b4	SFMSHD	QSSL signal release time selection	0: Release QSSL 0.5 QSPCLK after the rising edge of the last QSPCLK 1: Release QSSL 1.5 QSPCLKs after the rising edge of the last QSPCLK	R/W
b5	SFMSLD	QSSL signal output time selection	0: 0.5 QSPCLK output QSSL before the rising edge of the first QSPCLK 1: 1.5 QSPCLK outputs QSSL before the rising edge of the first QSPCLK	R/W
b31~b6	-	Reserved	Write 0 only, read out value is 0	-

17.2.3 Clock control register (SFMSKC)

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	SFMDTY	SFMDV				
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Bit	Symbol	Bit name	Description	R/W
b4~b0	SFMDV	Serial interface reference period selection	b4~ b0 0 0 0 0 0: 2 PCLKA 0 0 0 0 1: 3 PCLKA (multiplied by an odd number)*1 0 0 0 1 0: 4 PCLKA 0 0 0 1 1: 5 PCLKA (multiplied by an odd number)*1 0 0 1 0 0: 6 PCLKA 0 0 1 0 1: 7 PCLKA (multiplied by an odd number)*1 0 0 1 1 0: 8 PCLKA 0 0 1 1 1: 9 PCLKA (multiplied by an odd number)*1 0 1 0 0 0: 10 PCLKA 0 1 0 0 1: 11 PCLKA (multiplied by an odd number)*1 0 1 0 1 0: 12 PCLKA 0 1 0 1 1: 13 PCLKA (multiplied by an odd number)*1 0 1 1 0 0: 14 PCLKA 0 1 1 0 1: 15 PCLKA (multiplied by an odd number)*1 0 1 1 1 0: 16 PCLKA 0 1 1 1 1: 17 PCLKA (multiplied by an odd number)*1 1 0 0 0 0: 18 PCLKA 1 0 0 0 1: 20 PCLKA 1 0 0 1 0: 22 PCLKA 1 0 0 1 1: 24 PCLKA 1 0 1 0 0: 26 PCLKA 1 0 1 0 1: 28 PCLKA 1 0 1 1 0: 30 PCLKA 1 0 1 1 1: 32 PCLKA 1 1 0 0 0: 34 PCLKA 1 1 0 0 1: 36 PCLKA 1 1 0 1 0: 38 PCLKA 1 1 0 1 1: 40 PCLKA 1 1 1 0 0: 42 PCLKA 1 1 1 0 1: 44 PCLKA 1 1 1 1 0: 46 PCLKA 1 1 1 1 1: 48 PCLKA.	R/W
b5	SFMDTY	QSPCLK signal duty cycle correction function selection	0: No correction 1: Delay the QSPCLK signal by 0.5 PCLKA (PCLKA multiplied by an odd number is valid)	R/W
b31~b6	-	Reserved	Write 0 only, read out value is 0	-

Note 1: When PCLKA is selected multiplied by an odd number, the high level width of the QSPCLK signal is 1 PCLKA longer than the low level width before duty cycle correction.

17.2.4 Status register (SFMSST)

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	PFOFF	PFFUL	—	PFCNT				
Reset value:	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Bit	Symbo	Bit name	Description	R/W
b4~b0	PFCNT	Number of pre-read data bytes	b4~ b0 0 0 0 0 0: 0 bytes 0 0 0 0 1: 1 byte 0 0 0 1 0: 2 bytes 0 0 0 1 1: 3 bytes 0 0 1 0 0: 4 bytes 0 0 1 0 1: 5 bytes 0 0 1 1 0: 6 bytes 0 0 1 1 1: 7 bytes 0 1 0 0 0: 8 bytes 0 1 0 0 1: 9 bytes 0 1 0 1 0: 10 bytes 0 1 0 1 1: 11 bytes 0 1 1 0 0: 12 bytes 0 1 1 0 1: 13 bytes 0 1 1 1 0: 14 bytes 0 1 1 1 1: 15 bytes 1 0 0 0 0: 16 bytes 1 0 0 0 1: 17 bytes 1 0 0 1 0: 18 bytes Other settings are prohibited	R
b5	-	Reserved	Write 0 only, read out value is 0	-
b6	PFFUL	Pre-read cache status	"0: Pre-read cache is not full 1: Pre-read cache is full"	R
b7	PFOFF	Pre-reading function operation status	0: Pre-reading in progress 1: Pre-read function is disabled or not in action	R
b31~b8	-	Reserved	Write 0 only, read out value is 0	-

17.2.5 Communication port register (SFMCOM)

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	SFMD							
Reset value:	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x

x: uncertain

Bit	Symbo	Bit name	Description	R/W
b7~b0	SFMD	Port for direct communication with SPI bus	Converts the input and output of this port to an SPI bus cycle When DCOM=1, this port can be accessed in direct communication mode. In ROM access mode, access to this port is ignored.	R/W
b31~b8	-	Reserved	Write 0 only, read out value is 0	-

17.2.6 Communication mode control register (SFMCMD)

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	DCOM
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	DCOM	Mode selection for communication with SPI bus	0: ROM access mode 1: Direct access mode	R/W
b31~b1	-	Reserved	Write 0 only, read out value is 0	-

17.2.7 Communication status register (SFMCST)

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	EROMR	—	—	—	—	—	—	COMBSY
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	COMBSY	SPI bus cycle completion status in d	0: No serial transmission	R
b6~b1	-		1: Serial transmission in progress	-
b7	EROMR	Reserved	Write 0 only, read out value is 0	R/(W)*1
b31~b8	-	ROM access detection status in dire	0:No ROM access detected	-

Note: 1. This bit can only be written to "0"

17.2.8 Instruction code register (SFMSIC)

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	SFMCIC							
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b7~b0	SFMCIC	Serial flash instruction codes to be replaced	Serial flash instruction codes to be replaced	R/W
b31~b8	-	Reserved	Write 0 only, read out value is 0	-

17.2.9 Address mode control register (SFMSAC)

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	SFM4BC	—	—	SFMAS	
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Bit	Symbol	Bit name	Description	R/W
b1,b0	SFMAS	Serial interface address byte number selection	b1，b0 00：1 byte 01：2 bytes 10：3 bytes 11：4 bytes	R/W
b3,b2	-	Reserved	Write 0 only, read out value is	-
b4	SFM4BC	Default command code selection (when serial interface address width is 4 bytes)	0: Do not use 4-byte address read instruction 1: Use 4-byte address read instruction	R/W
b31~b8	-	Reserved	Write 0 only, read out value is 0	-

17.2.10 Dummy cycle control register (SFMSDC)

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SFMXD								SFMXEN	SFMXST	—	—	SFMDN[3:0]			
Reset value:	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0

Bit	Symbol	Bit Name	Description	R/W
b3~b0	SFMDN[3:0]	Selection of the number of dummy cycles for fast read instructions	b3~b0 0 0 0 0:Default dummy cycle per instruction. - Fast read quad I/O: 6 QSPCLK - Fast read quad output: 8 QSPCLK - Fast read dual I/O: 4 QSPCLK - Fast read dual output: 8 QSPCLK - Fast read: 8 QSPCLK. 0 0 0 1: 3 QSPCLK*1 0 0 1 0: 4 QSPCLK 0 0 1 1: 5 QSPCLK 0 1 0 0: 6 QSPCLK 0 1 0 1: 7 QSPCLK 0 1 1 0: 8 QSPCLK 0 1 1 1: 9 QSPCLK 1 0 0 0: 10 QSPCLK 1 0 0 1: 11 QSPCLK 1 0 1 0: 12 QSPCLK 1 0 1 1: 13 QSPCLK 1 1 0 0: 14 QSPCLK 1 1 0 1: 15 QSPCLK 1 1 1 0: 16 QSPCLK 1 1 1 1: 17 QSPCLK.	R/W
b5,b4	-	Reserved	Write 0 only, read out value is 0	-
b6	SFMXST	XIP mode status	0: Non-XIP mode 1: XIP mode	R
b7	SFMXEN	XIP mode allows	0: XIP mode disabled 1: XIP mode enabled	R/W
b15~b8	SFMXD	Serial flash mode data (control XIP mode)	Serial flash mode data	R/W
b31~b16	-	Reserved	Write 0 only, read out value is 0	-

17.2.11 SPI protocol control register (SFMSPC)

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	SFMSDE	—	—	SFMSPI	
Reset value:	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b1，b0	SFMSPI	SPI Protocol Selection	b1, b0 00: Extended SPI protocol 01: Dual SPI Protocol 10: Quad SPI Protocol 11: Settings are prohibited	R/W
b3,b2	-	Reserved	Write 0 only, read out value is 0	-
b4	SFMSDE	Minimum time selection for input and output switching (when dual SPI or quad SPI protocol is selected and standard read mode)	0: Do not specify the minimum switching time 1: Specify the minimum switching time as 1 QSPCLK	R/W
b31~b5	-	Reserved	Write 0 only, read out value is 0	-

17.2.12 Port control register (SFMPMD)

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	SFMWPL	—	—
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b1，b0	-	Reserved	Write 0 only, read out value is 0	-
b2	SFMWPL	WP Pin Description	0: Low level 1: High level	R/W
b31~b3	-	Reserved	Write 0 only, read out value is 0	-

17.2.13 External QSPI address register (SFMCNT1)

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	QSPI_EXT[5:0]						—	—	—	—	—	—	—	—	—	—
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

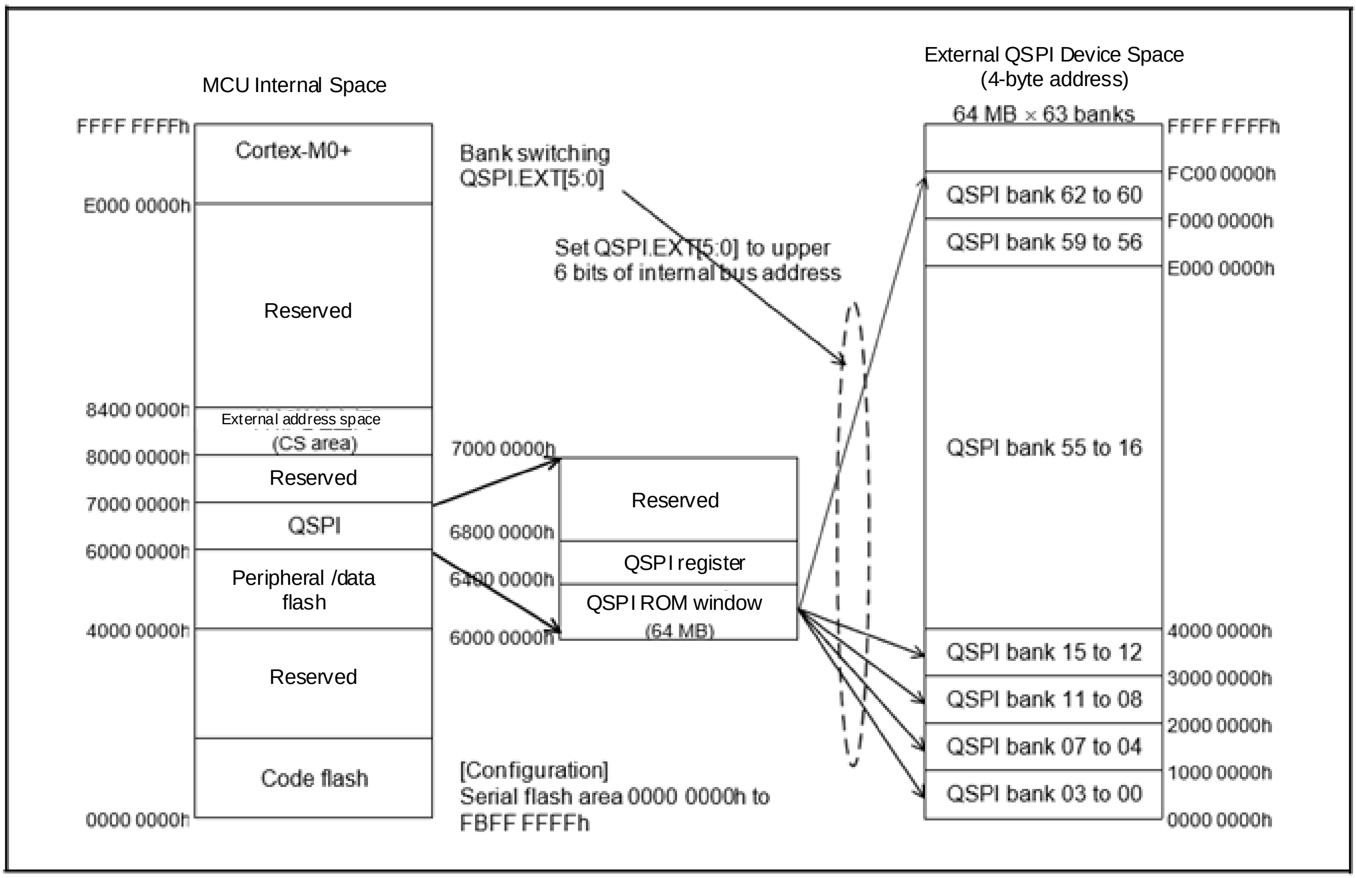
Bit	Symbol	Bit name	Description	R/W
b25~b0	-	Reserved	Write 0 only, read out value is 0	-
b31~b26	QSPI_EXT[5:0]	bank switch address	When accessing the interval 6000 0000h to 63FF FFFFh, the address bus is controlled by QSPI_EXT[5:0] is set to the high 6 bits of the internal bus address.	R/W

17.3 Memory mapping

17.3.1 Internal bus space

The location of the serial flash and control registers on the AHB space is determined by the address range set in the space configuration.

Figure 17-2 Default address settings and AHB space memory mapping



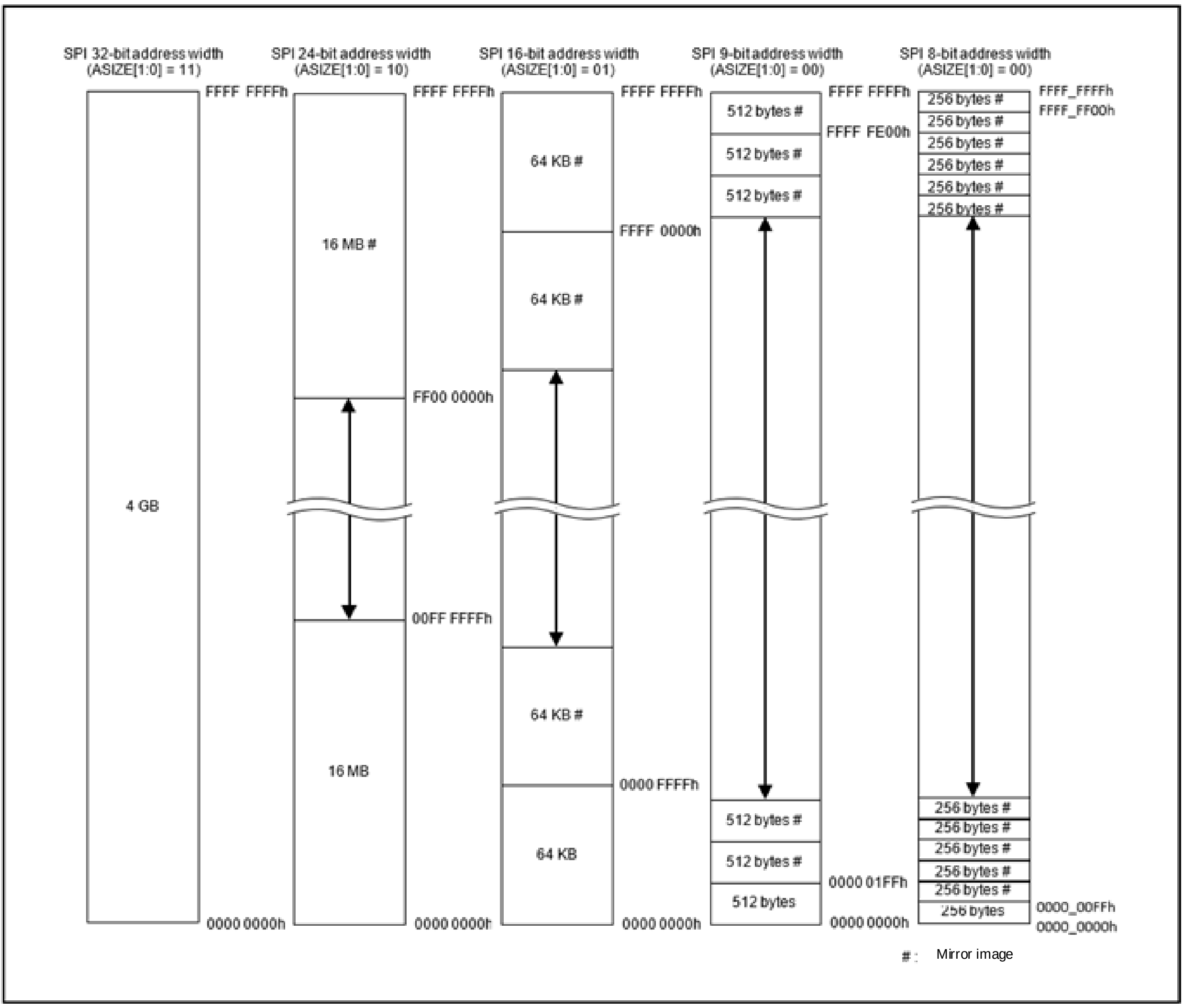
17.3.2 SPI space and address width of SPI bus

The SPI space has a 32-bit address width for serial flash memory. When a read access is made to the SPI space, the SPI bus cycle starts automatically and returns the data read from the serial flash.

The address width of SPI space is fixed at 32 bits. And the address width of SPI bus can be selected by SFMAS[1:0] of SFMSAC register with 8, 16, 24 or 32 bits.

If the address width of the SPI bus is selected to be 8, 16, or 24 bits, only the low address of the SPI space is sent to the serial flash via the SPI bus. Therefore, the mirror of the serial flash associated with the address width of the SPI bus is repeatedly present in the SPI space.

Figure 17-3 Memory mapping of SPI space



Notice: The SPI bus address width can be selected as 32-bit, 24-bit, 16-bit, and 8-bit by using the SFMAS[1:0] bits of the SFMSAC register. When 8-bit address width is selected, the address information in bit 9 can be embedded in the read instruction code. For more information on read instructions, refer to Section 17.6.2, "Standard Read Instructions".

17.4 SPI bus

17.4.1 SPI protocol

In addition to the SPI protocol for connecting serial flash memory, extended SPI, dual SPI, and quad SPI are also supported. the initial state of the SPI protocol is extended SPI, which can be changed using the SFMSPI bit in the SFMSPC register. The extended SPI protocol always outputs the instruction code from a single QIO0 pin. Depending on the instruction code format, it uses pins 1 ~ 4 to perform subsequent address and data I/O operations.

Figure 17-4 Example 1 of extended SPI protocol for fast read

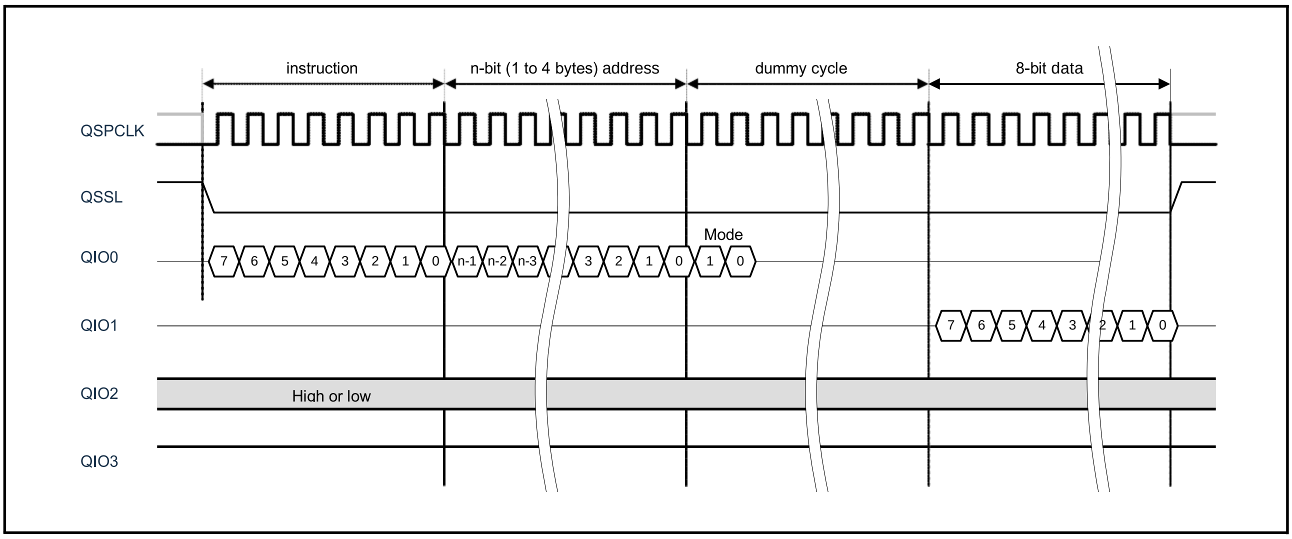
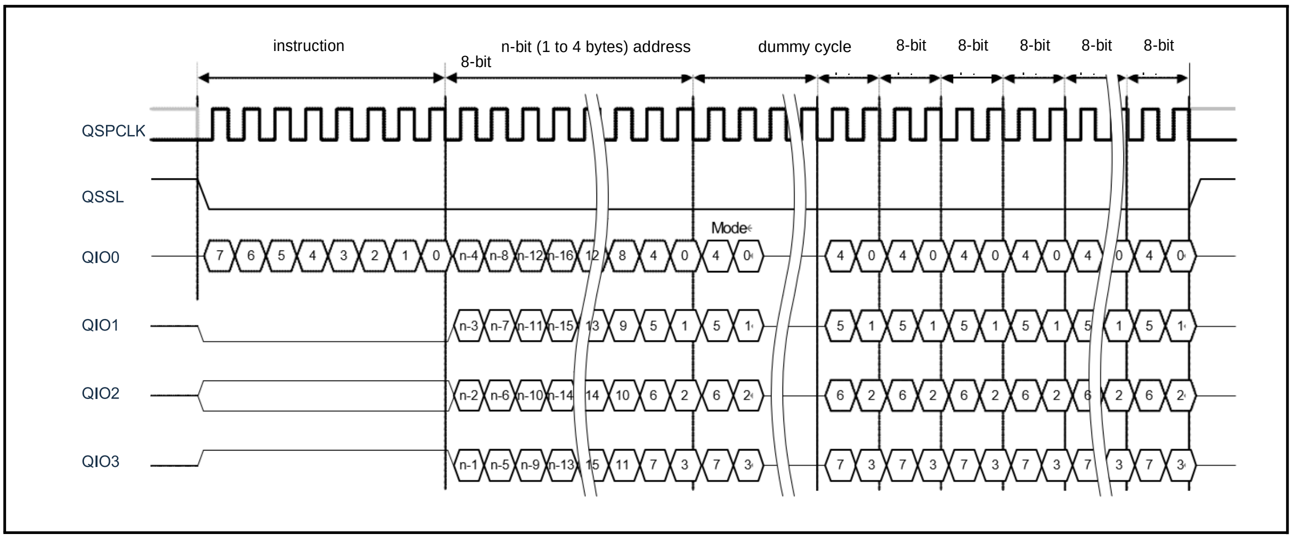
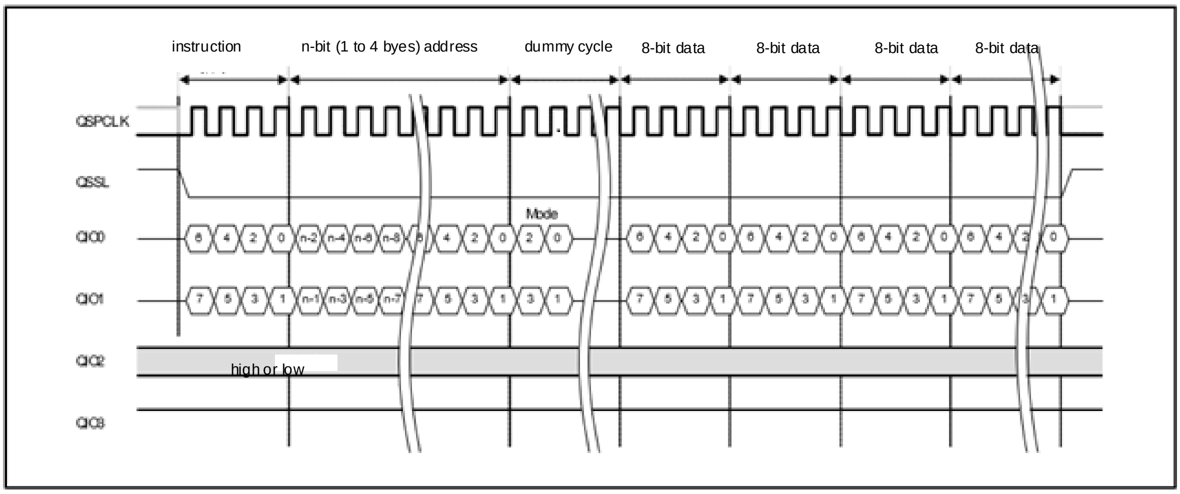


Figure 17-5 Example 2 of extended SPI protocol for quad I/O fast read



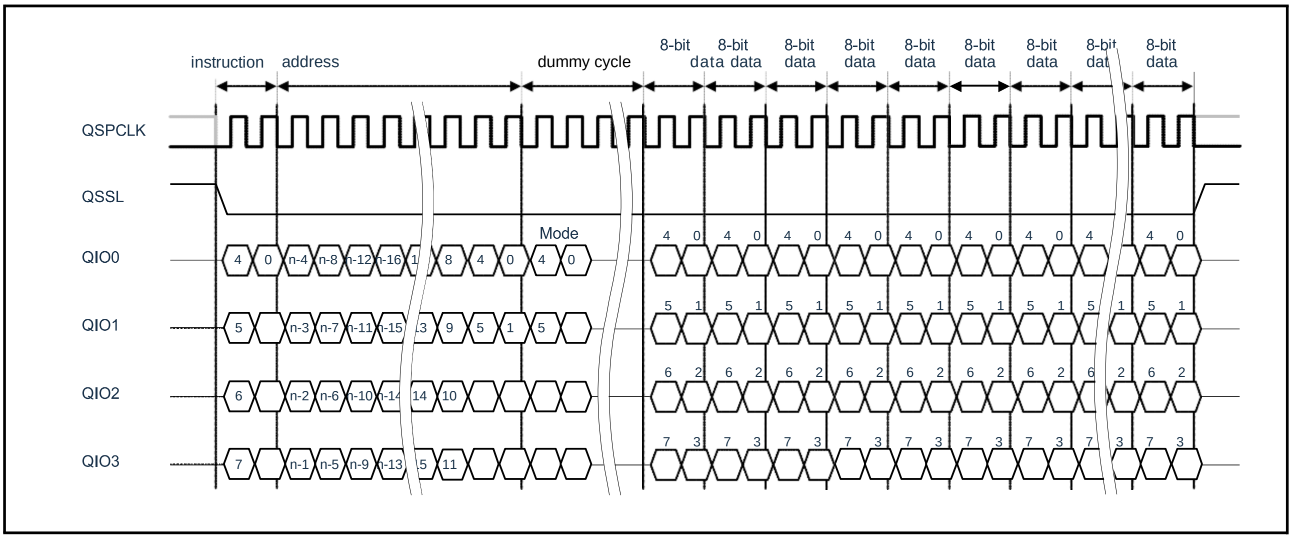
The dual SPI protocol uses two pins, QIO0 and QIO1, to perform I/O on all signals (such as instruction codes, addresses, and data)

Figure 17-6 Example of dual fast read SPI protocol



The Quad SPI protocol uses four pins (QIO0, QIO1, QIO2 and QIO3) to perform I/O operations on all signals such as instruction codes, addresses and data.

Figure 17-7 Example of quad fast read SPI protocol

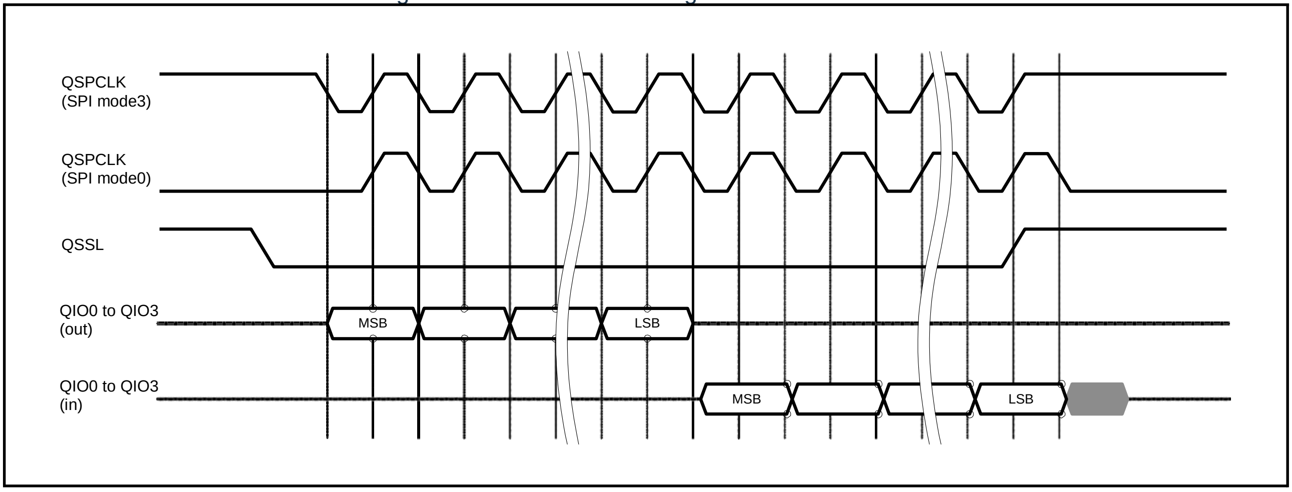


17.4.2 SPI mode

The SPI mode can be switched by changing the register settings during operation. The difference between SPI mode 0 and 3 is the standby level of the QSPCLK signal. The standby level of the QSPCLK signal is low in SPI mode 0 and high in SPI mode 3.

Serial data is output from the QSPI on the falling edge of the serial clock and is read into the external flash memory on the rising edge of the serial clock. Serial data is output from the external flash memory on the falling edge of the serial clock and is read into the QSPI on the next falling edge of the serial clock.

Figure 17-8 Basic timing of the serial interface



17.5 SPI bus timing adjustment

The timing of the SPI bus signals can be adjusted via registers. The configured timing is applied to all SPI bus accesses, including ROM accesses and direct communication.

17.5.1 SPI bus reference period

The reference period for SPI bus operation is adjustable and is obtained by multiplying PCLKA by some integer. By setting the SFMDV[4:0] bits in the SFMSKC register, the reference period can be selected in the range of PCLKA multiplied by 2 to PCLKA multiplied by 48.

Table 17-3 SFMDV[4:0] control for period multiplier and serial clock frequency

SFMDV[4:0]	Periodic multiplier	PCLKA Frequency (MHz)
		48
11111	48	1.00
11110	46	1.04
11101	44	1.09
11100	42	1.14
11011	40	1.20
11010	38	1.26
11001	36	1.33
11000	34	1.41
10111	32	1.50
10110	30	1.60
10101	28	1.71
10100	26	1.85
10011	24	2.00
10010	22	2.18
10001	20	2.40
10000	18	2.67
01111	17	2.82
01110	16	3.00
01101	15	3.20
01100	14	3.43
01011	13	3.69
01010	12	4.00
01001	11	4.36
01000	10	4.80
00111	9	5.33
00110	8	6.00
00101	7	6.86
00100	6	8.00
00011	5	9.60
00010	4	12.00
00001	3	16.00
00000	2	24.00

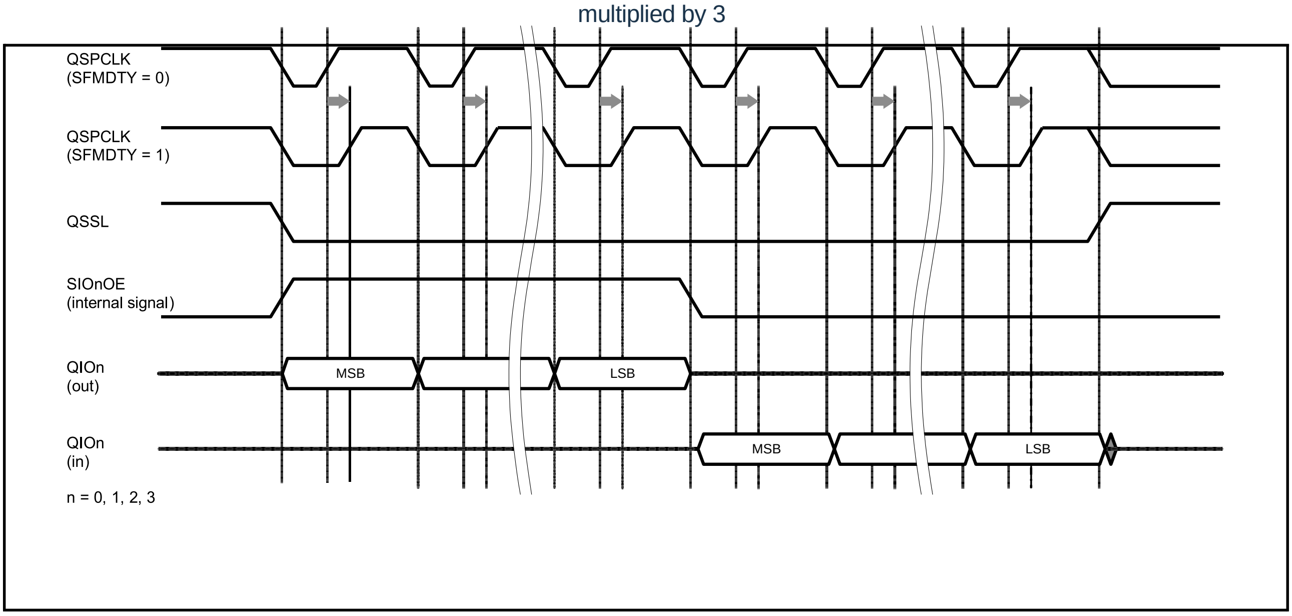
17.5.2 QSPCLK signal duty cycle

When the reference clock is configured with PCLKA multiplied by an even number, the high and low level widths of the QSPCLK signal match each other. When the PCLKA is multiplied by an odd number, the high level width of the QSPCLK signal is 1 PCLKA longer than the low level width.

When the reference clock is PCLKA multiplied by an odd number, to make the duty cycle of the QSPCLK signal close to 50%, the SFMDTY bit in the SFMSKC register needs to be set to 1. With this setting, the rising edge of the QSPCLK output signal is delayed by half a PCLKA period, which is equivalent to the 50% duty cycle interface operation.

When the reference clock is PCLKA multiplied by an even number, the SFMDTY setting in the SFMSKC register is ignored.

Figure 17-9 Example of using SFMDTY to adjust the duty cycle of the QSPCLK signal when the reference clock is PCLKA multiplied by 3



17.5.3 Minimum high-level width of QSSL signal

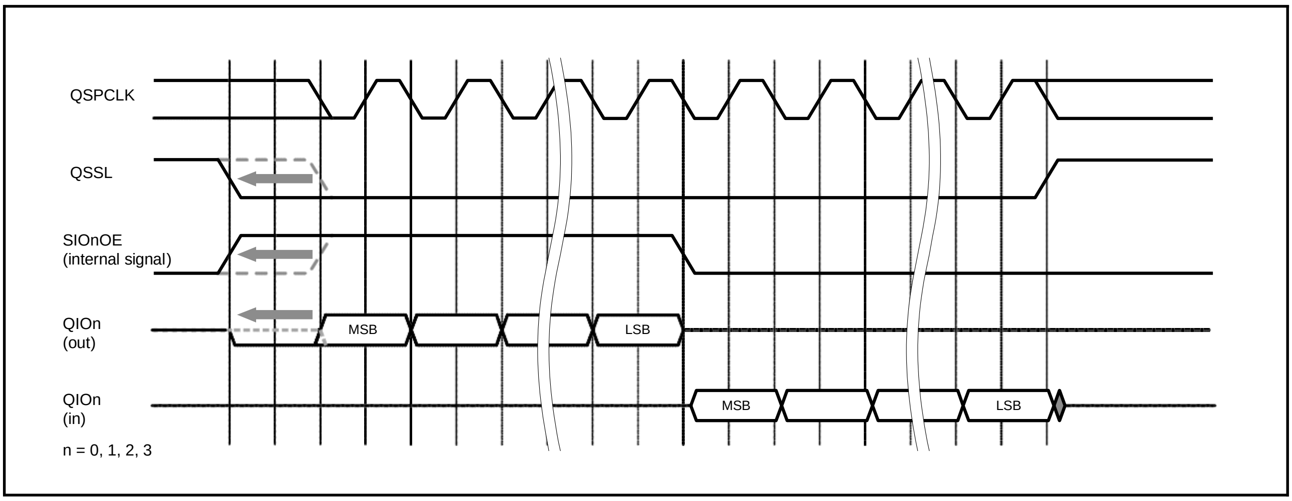
The QSSL signal must be held high (invalid) long enough between adjacent SPI bus cycles to meet the desired deselect time of the serial flash. The SFMSW[3:0] bits of the SFMSSC register are used to select a number from 1 to 16, multiplying this number by the reference cycle as the minimum high width of the QSSL output signal.

17.5.4 QSSL signal set-up time

The time from when the QSSL signal goes low to the first rise of the QSPCLK signal is the set-up time of the QSSL signal. The build time is selected by setting the SFMSLD bit in the SFMSSC register to 0.5 QSPCLK or 1.5 QSPCLK for the serial flash memory.

The SFMSLD setting in the SFMSSC register is also used to configure the build time from the serial data output allow signal (QIO0OE/QIO1OE/QIO2OE/QIO3OE) to the first rising edge of the QSPCLK signal. The set value satisfies the most strict timing constraints of the application.

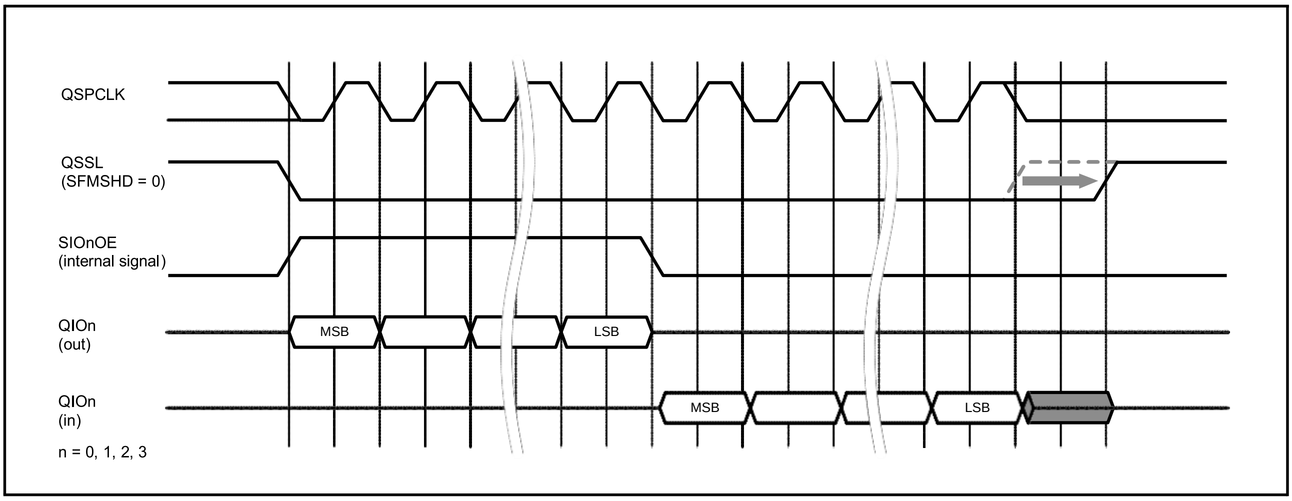
Figure 17-10 Adjusting the set-up time of QSSL signal with SFMSLD



17.5.5 QSSL signal hold time

The hold time for the QSSL signal is from the last rising edge of the QSPCLK signal until the QSSL signal is driven high. The hold time is selected by setting the SFMSHD bit of the SFMSSC register to 0.5 QSPCLK or 1.5 QSPCLK to meet the device requirements.

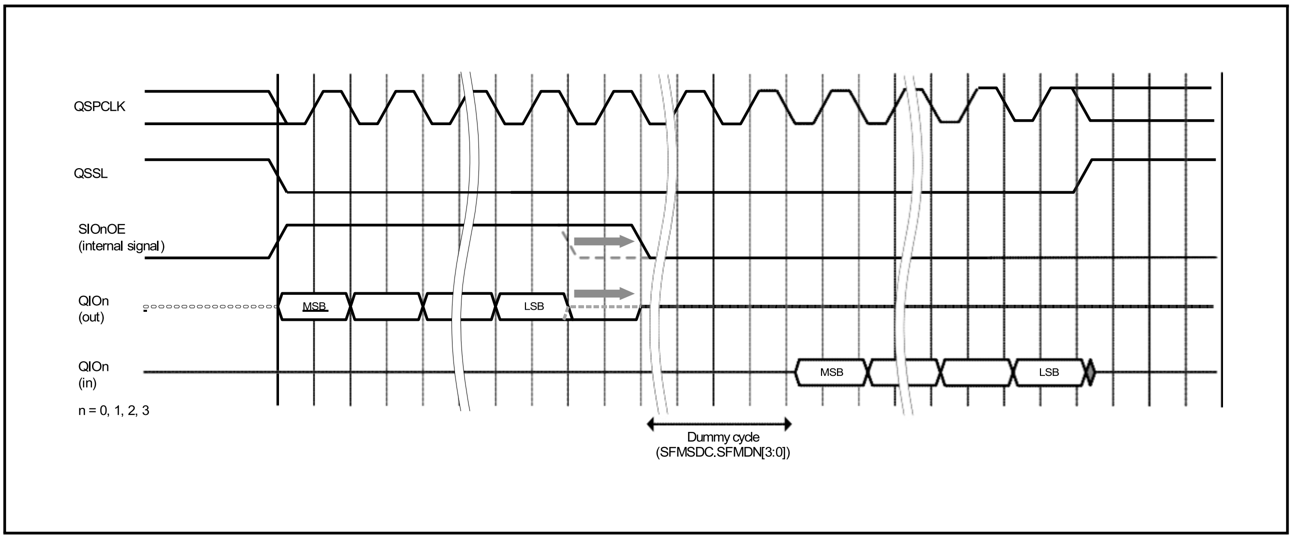
Figure 17-11 Adjusting the hold time of QSSL signal with SFMSHD



17.5.6 Serial data output enable hold time

The buffer output enable signals on the QIO0, QIO1, QIO2 or QIO3 pins can be extended by 1 QSPCLK by setting the SFMOEX bit in the SFMSMD register. The extended signals only include the output enable signals, i.e., the QIO0E, QIO1OE, QIO2OE and QIO3OE signals. The output data signals QIO0O, QIO1O, QIO2O, and QIO3O are not included.

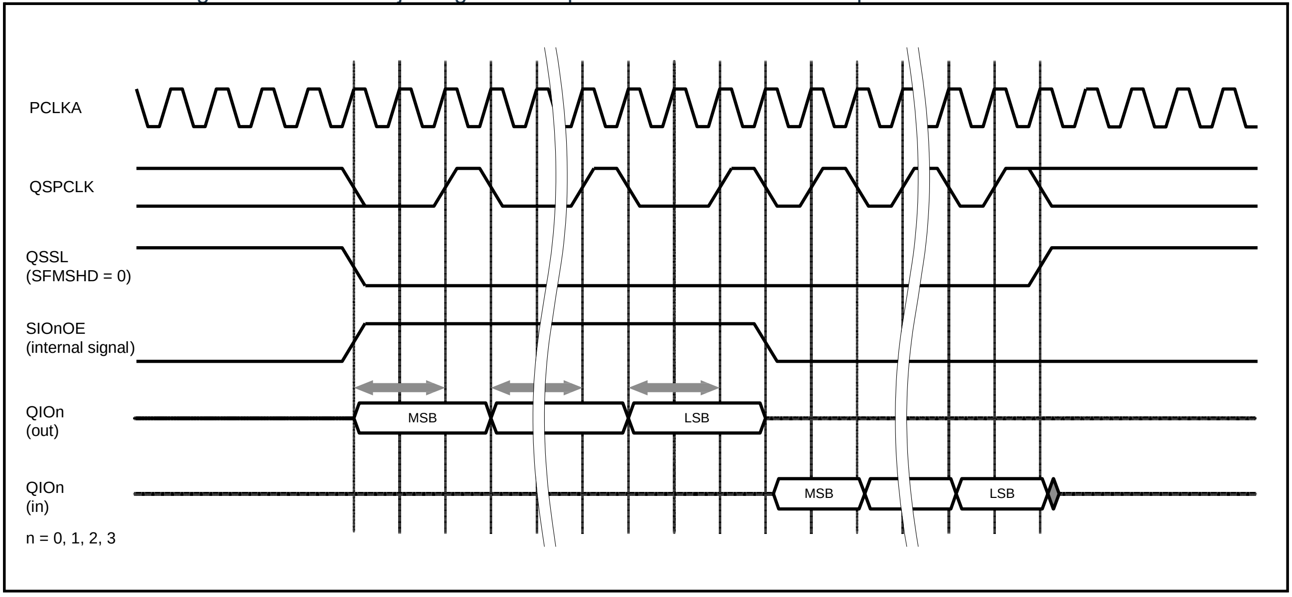
Figure 17-12 Adjusting the output enable hold time with the SFMOEX bit



17.5.7 Serial data output set-up time

When an instruction or address is transferred to the serial flash memory, the build-up time from the start of serial data output to the rising edge of the QSPCLK signal is the serial data output. If this build time is insufficient, the SFMOSW bit in the SFMSMD register can be set to extend the build time by 1 PCLKA. When the SFMOSW bit is set to 1 and data is output from the QSPI, the low width of QSPCLK during serial data transfer will be extended by 1 PCLKA. This function has no effect on serial data reception.

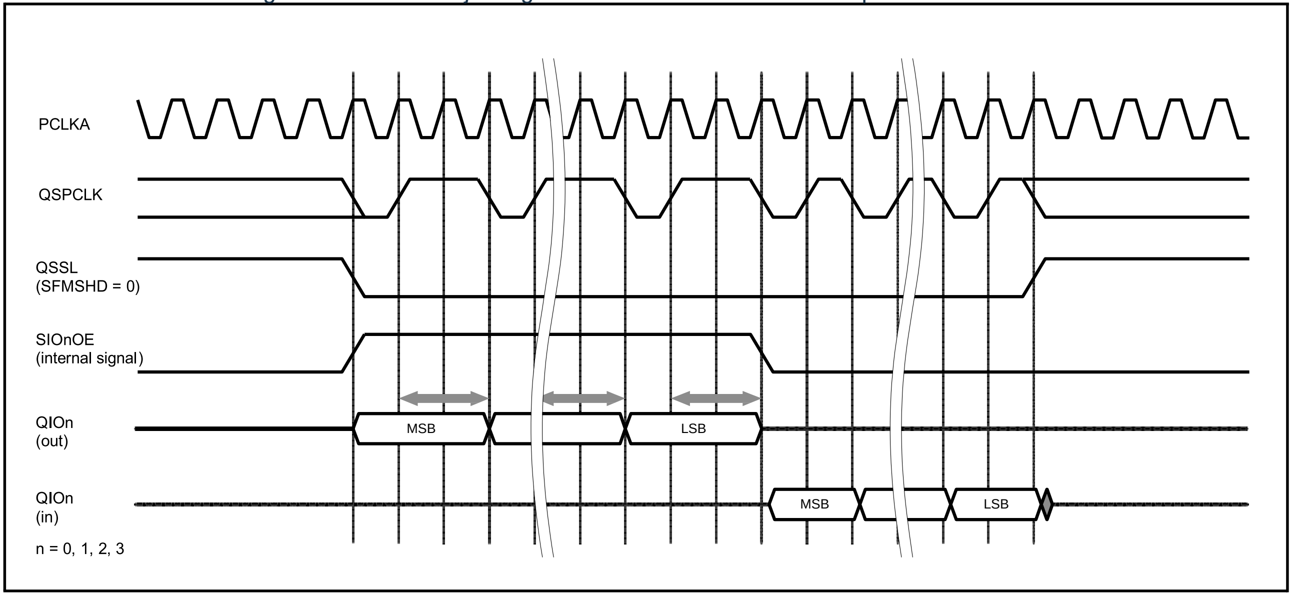
Figure 17-13 Adjusting the set-up time of the serial data output with the SFMOSW bit



17.5.8 Serial data output hold time

When an instruction or address is transferred to the serial flash memory, the hold time starts from the rising edge of QSPCLK and ends when the serial data makes another transfer. If the hold time is insufficient, the SFMOHW bit of the SFMSMD register can be set to extend the hold time by 1 PCLKA. When the SFMOHW bit is set to 1 and data is output from the QSPI, the high width of QSPCLK during serial data transfer will be extended by 1 PCLKA. This function has no effect on serial data reception.

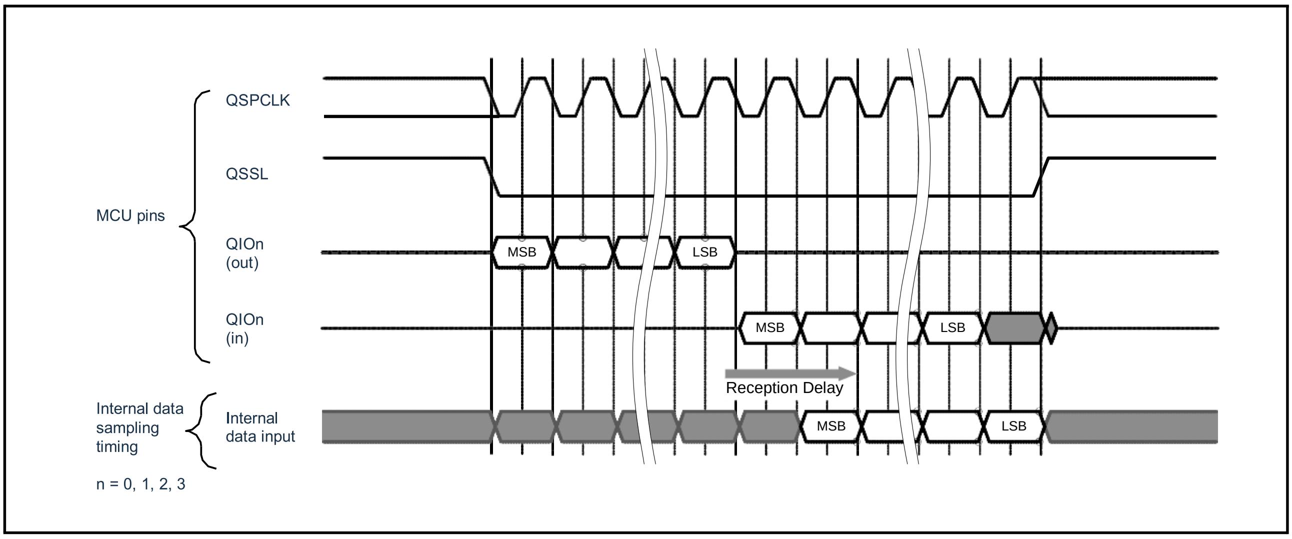
Figure 17-14 Adjusting the hold time of serial data output with SFMOHW



17.5.9 Serial data reception delay

The data output from the serial flash is synchronized with the falling edge of the QSPCLK signal. The QSPI receives this data synchronously on the next falling edge of the QSPCLK signal. The delay from the time the serial flash starts outputting data to the time the QSPI receives the data is called the receive delay. The QSPI adds a delay adjustment cycle before the first data receive cycle in the SPI bus cycle. From the serial flash side, this is seen as an increase in the number of data receive cycles. With no data reception, this increased delay adjustment cycle is not generated in the SPI bus cycle.

Figure 17-15 Reception delay



17.6 SPI instruction set for flash access

17.6.1 Types of automatically generated SPI instructions

When the serial flash memory is accessed, one of the following instructions for one SPI bus cycle is automatically generated based on the setting of the SFMAS[1:0] bits in the SFMSAC register and the setting of the SFMSMD register.

Table 17-4 SPI instruction set automatically generated when SFMAS[1:0]=00

Instruction format	Instruction code	Address bytes	Dummy cycle	Data bytes	Description
Read	03h[*1]	1	—	1 to ∞	Set (SFMRM[2:0] = 000), (A8 = 0)
	0Bh[*1]	1	—	1 to ∞	Set (SFMRM[2:0] = 000), (A8 = 1)

Note 1: If SFMSMD.SFMCCE bit is set to 1, the content stored in SFMSIC.SFMCIC[7:0] is considered as an instruction code.

Table 17-5 SPI instruction set automatically generated when SFMAS[1:0]=01

Instruction format	Instruction code	Address bytes	Dummy cycle	Data bytes	Description
Read	03h[*1]	2	—	1 to ∞	Set (SFMRM[2:0] = 000)

Note 1: If SFMSMD.SFMCCE bit is set to 1, the content stored in SFMSIC.SFMCIC[7:0] is considered as an instruction code.

Table 17-6 SPI instruction set automatically generated when SFMAS[1:0]=10

Instruction format	Instruction code	Address bytes	Dummy cycle	Data bytes	Description
Read	03h[*1]	3	—	1 to ∞	Set (SFMRM[2:0] = 000)
Quick Read	0Bh[*1]	3	8[*2]	1 to ∞	Set (SFMRM[2:0] = 001)
Dual output for fastreading	3Bh[*1]	3	8[*2]	1 to ∞	Set (SFMRM[2:0] = 010)
Dual I/O for fast reads	BBh[*1]	3	4[*2]	1 to ∞	Set (SFMRM[2:0] = 011)
Quad output for fastreading	6Bh[*1]	3	8[*2]	1 to ∞	Set (SFMRM[2:0] = 100)
Quad I/O for fastreads	EBh[*1]	3	6[*2]	1 to ∞	Set (SFMRM[2:0] = 101)
Write permission	06h	—	—	—	Set (ENEX4B[1:0] = 10)
Exit 4-byte mode	E9h	—	—	—	Set (ENEX4B[1:0] = 01,10)

Note 1: If the SFMSMD.SFMCCE bit is set to 1, the content stored in SFMSIC.SFMCIC[7:0] is considered as an instruction code.

Note 2: The number of dummy cycles can be set by the SFMDRC register.

Table 17-7 Automatically generated SPI instruction set when SFMAS[1:0]=11 and SFM4BC=0

Instruction format	Instruction code	Address bytes	Dummy cycle	Data bytes	Description
Read	03h[*1]	4	—	1 to ∞	Set (SFMRM[2:0]=000)
Quick read	0Bh[*1]	4	8[*2]	1 to ∞	Set (SFMRM[2:0]=001)
Dual output for fastreading	3Bh[*1]	4	8[*2]	1 to ∞	Set (SFMRM[2:0]=010)
Dual I/O for fast reads	BBh[*1]	4	4[*2]	1 to ∞	Set (SFMRM[2:0]=011)
Quad output for fastreading	6Bh[*1]	4	8[*2]	1 to ∞	Set (SFMRM[2:0]=100)
Quad I/O fast readout	EBh[*1]	4	6[*2]	1 to ∞	Set (SFMRM[2:0]=101)
Writeable	06h	—	—	—	Set (ENEX4B[1:0]=10)
Enter 4-byte mode	B7h	—	—	—	Set (ENEX4B[1:0]=01,10)

Note 1: If the SFMSMD.SFMCCE bit is set to 1, the content stored in SFMSIC.SFMCIC[7:0] is considered as an instruction code.

Note 2: The number of dummy cycles can be set by the SFMDRC register.

Table 17-8 Automatically generated SPI instruction set when SFMAS[1:0]=11 and SFM4BC=1

Instruction format	Instruction code	Address bytes	Dummy cycle	Data bytes	Description
Read	13h[*1]	4	—	1 to ∞	Set (SFMRM[2:0] = 000)
Quick read	0Ch[*1]	4	8[*2]	1 to ∞	Set (SFMRM[2:0] = 001)
Dual output for fastreading	3Ch[*1]	4	8[*2]	1 to ∞	Set (SFMRM[2:0] = 010)
Dual I/O for fast reads	BCh[*1]	4	4[*2]	1 to ∞	Set (SFMRM[2:0] = 011)
Quad output for fastreading	6Ch[*1]	4	8[*2]	1 to ∞	Set (SFMRM[2:0] = 100)
Quad I/O fast readout	ECh[*1]	4	6[*2]	1 to ∞	Set (SFMRM[2:0] = 101)
Writeable	06h	—	—	—	Set (ENEX4B[1:0] = 10)
Enter 4-byte mode	B7h	—	—	—	Set (ENEX4B[1:0] =01,10)

Note 1: If the SFMSMD.SFMCCE bit is set to 1, the content stored in SFMSIC.SFMCIC[7:0] is considered as an instruction code.

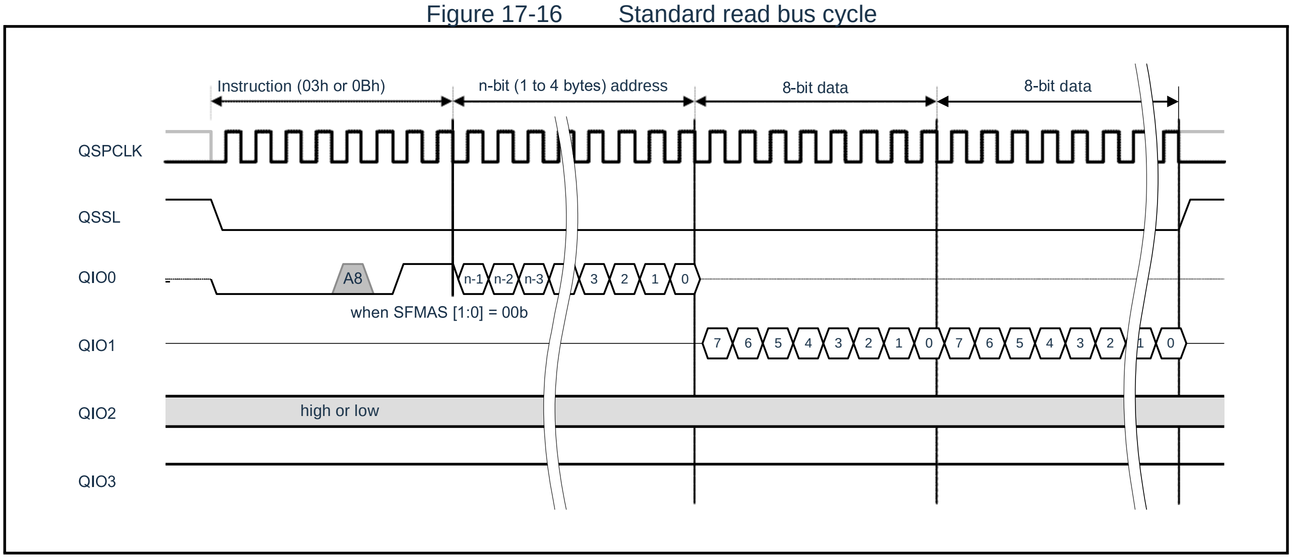
Note 2: The number of dummy cycles can be set by the SFMDRC register.

17.6.2 Standard read instruction

The standard read instruction is a common read instruction supported by most serial flash devices. This standard read instruction is selected in the initial setup of the QSPI. When the SPI bus cycle starts, the serial flash select signal is set and the instruction code (03h/13h)*1 is output. Next, the address is sent according to the width (1 to 4 bytes) specified in the SFMAS[1:0] bits of the SFMSAC register. Then the data is received.

Note 1. To reduce overhead, many 4kb serial flash devices have an address range no larger than 1 byte (A7 to A0) and receive A8 information from bit[3] of the read instruction code. To support these devices, QSPI outputs A8 (address bit 8) to bit[3] of the standard read instruction code when only a 1-byte address width is specified (SFMAS[1:0]=00). This means that 0Bh can be output instead of 03h as the standard read instruction code. This code is duplicated with the fast read instruction code. However, for most serial flash devices of 2kb or less with an address width of 1 byte, the bit[3] of the instruction is designed not to be decoded, so such a read instruction code is correctly recognized as a standard read instruction code. In rare cases, some serial flash devices allow bit[3] to be decoded. When connecting such a serial flash memory, configure the application to avoid the access results caused by A8=1.

Figure 17-16 Standard read bus cycle



17.6.3 Quick read instruction

Fast read instructions support higher speed communication clocks than standard read instructions. When the SPI bus cycle starts, the serial flash select signal is set and the instruction code (0Bh/0Ch) is output. Next, an address of width 1 to 4 bytes specified by the SFMAS[1:0] bits of the SFMSAC register is sent, and a specific number of dummy cycles specified by the SFMSDC register are sent, followed by data reception.

The first two dummy cycles are used to select or deselect the XIP mode. When XIP mode is selected, the same instruction is applied to the next SPI bus cycle, and the instruction code transfer for the next SPI bus cycle is skipped. For more information on XIP mode, refer to Section 17.8, XIP Control.

The SFMSMD register can be used to toggle fast read instructions.

Figure 17-17 Fast read bus cycle

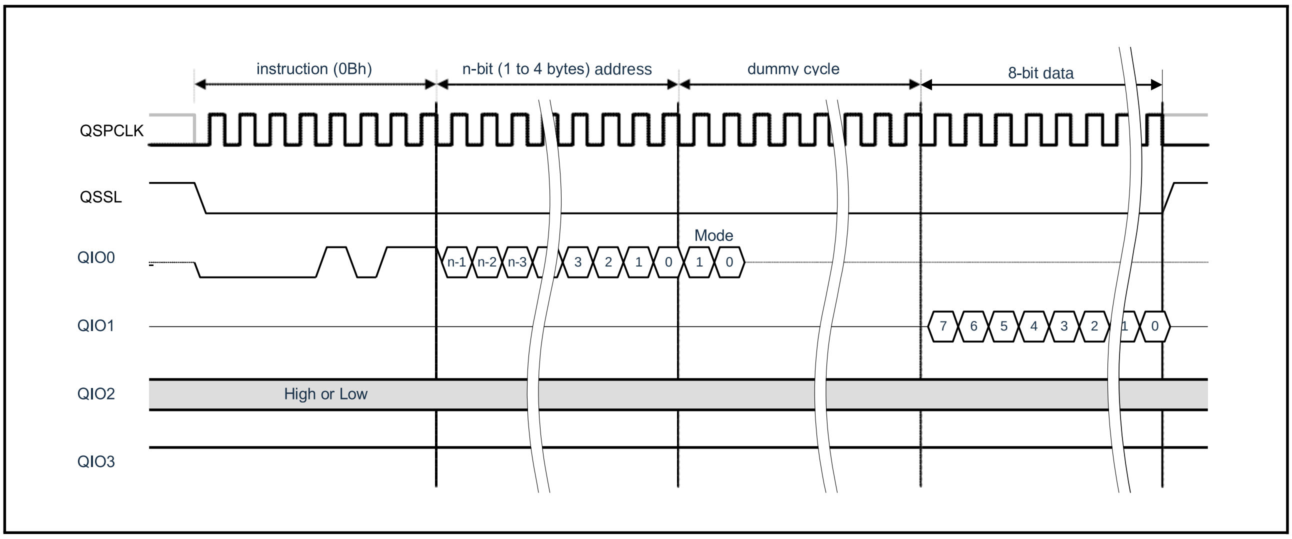
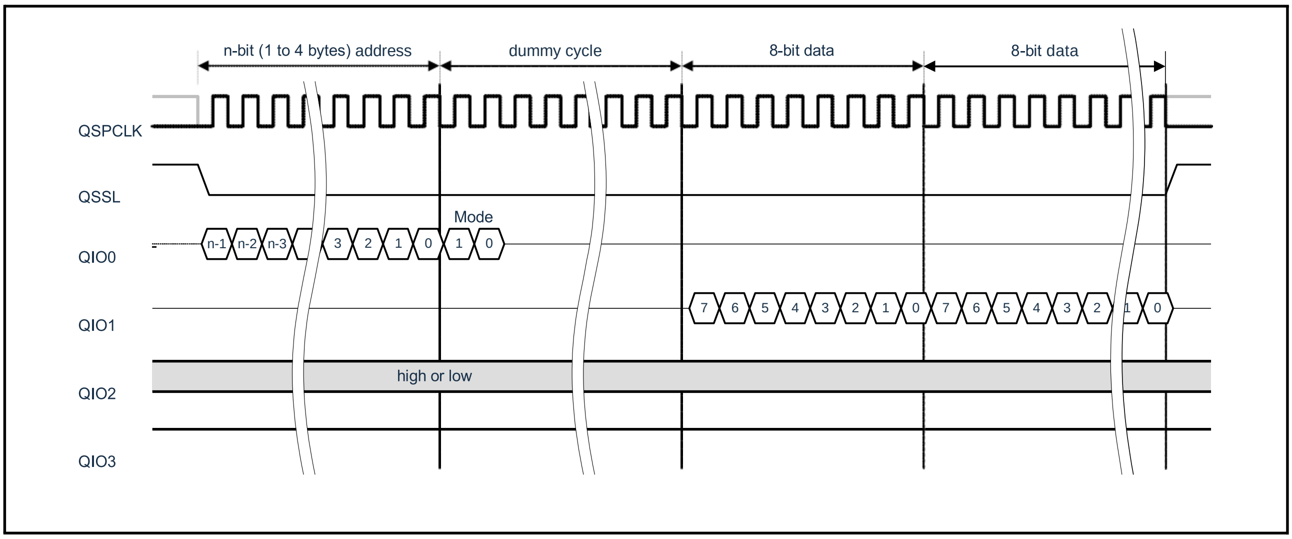


Figure 17-18 **XIP mode fast read bus cycle**



Notice: To use the fast read instruction, you must use a serial flash device that supports fast reads.

17.6.4 Dual output fast read instruction

The dual output fast read instruction is a read instruction that uses two signal lines to receive data. The serial flash select signal is set when the SPI bus cycle begins. The instruction code (3Bh/3Ch) and an address from 1 to 4 bytes wide, specified by the SFMAS[1:0] bits of the SFMSAC register, are transmitted from the QIO0 pin. Next, a specific number of dummy loops specified by the SFMSDC register are sent. The data is then received via the QIO0 and QIO1 pins. Even-bit data is received from the QIO0 pin, and odd-bit data is received from the QIO1 pin.

The first two dummy cycles are used to select the XIP mode. When XIP mode is selected, the same instruction used in the current cycle is applied to the next SPI bus cycle, and the instruction code transfer for the next SPI bus cycle is skipped. For more information on XIP mode, refer to Section 17.8, XIP Control.

The SFMSMD register can be used to switch to a dual output for fast read instructions.

Figure 17-19 Dual output fast read bus period

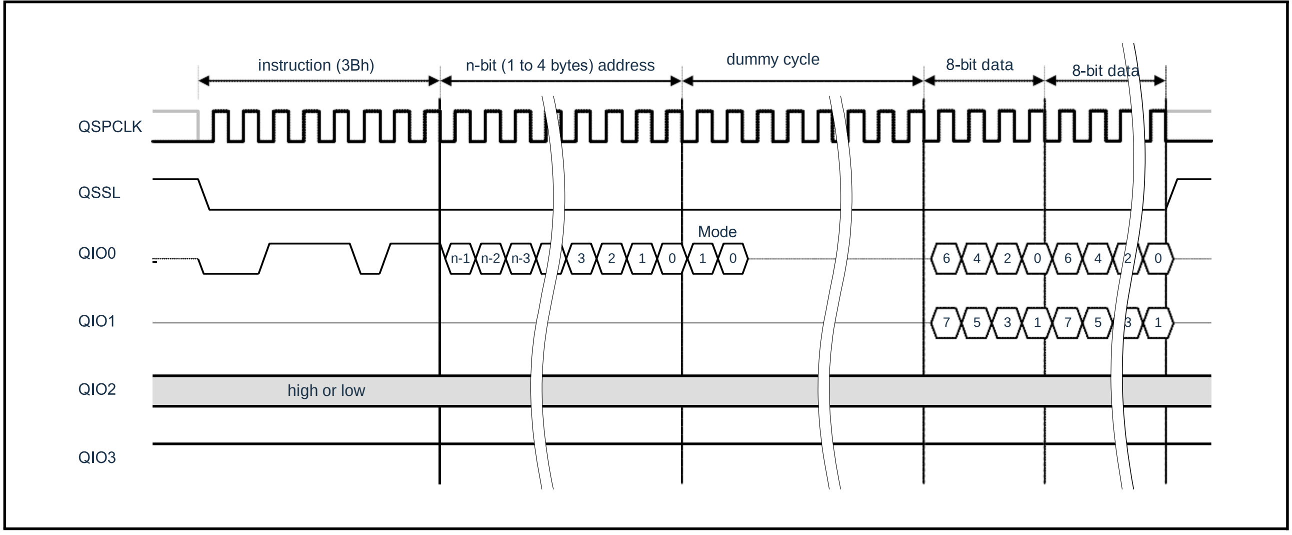
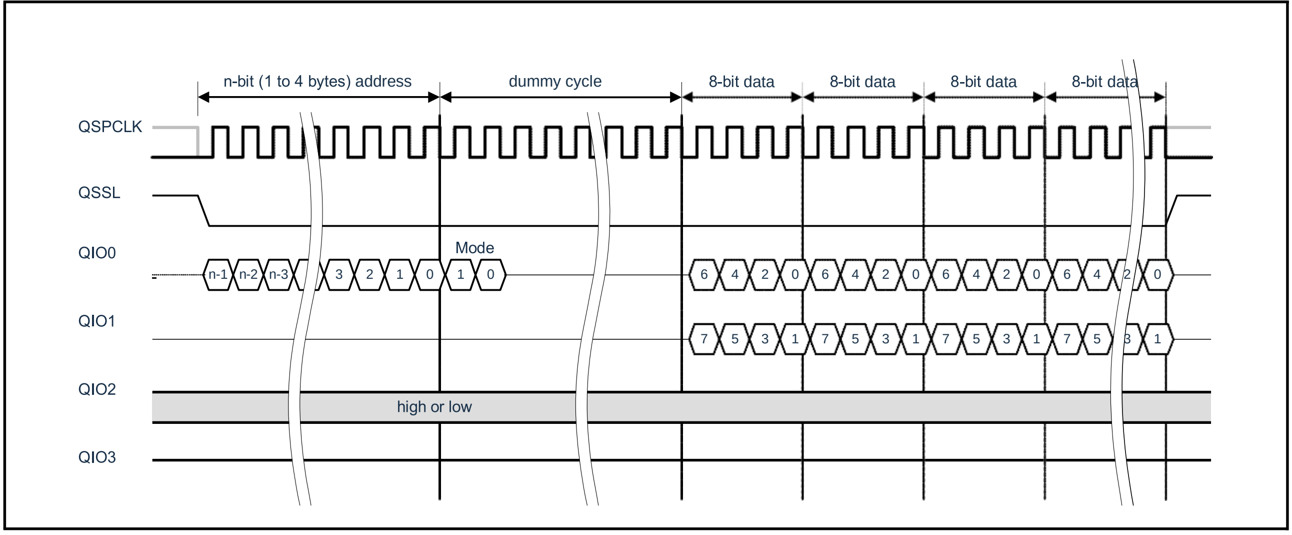


Figure 17-20 XIP mode dual output fast read bus cycle



Notice: To use the Dual Output Fast Read instruction, you must use a serial flash device that supports Dual Output Fast Read.

17.6.5 Dual I/O fast read instruction

The dual I/O fast read instruction is a read instruction that uses two signal lines to send addresses and receive data. When the SPI bus cycle begins, the serial flash select signal is set and the instruction code (BBh/BCh) is output from the QIO0 pin. Next, an address of width 1 to 4 bytes specified by the SFMAS[1:0] bits of the SFMSAC register is sent through the QIO0 and QIO1 pins and a specific number of dummy cycles specified by the SFMSDC register are sent. Data is then received via the QIO0 and QIO1 pins. The address and dummy cycles are sent and the data is received by transmitting even bits through the QIO0 pin and odd bits through the QIO1 pin.

The first two dummy cycles are used to select the XIP mode. When XIP mode is selected, the same instruction used in the current cycle is applied to the next SPI bus cycle, and the instruction code transfer for the next SPI bus cycle is skipped. For more information on XIP mode, refer to Section 17.8, XIP Control.

The SFMSMD register can be used to switch to dual I/O fast read.

Figure 17-21 Dual I/O fast read bus cycle

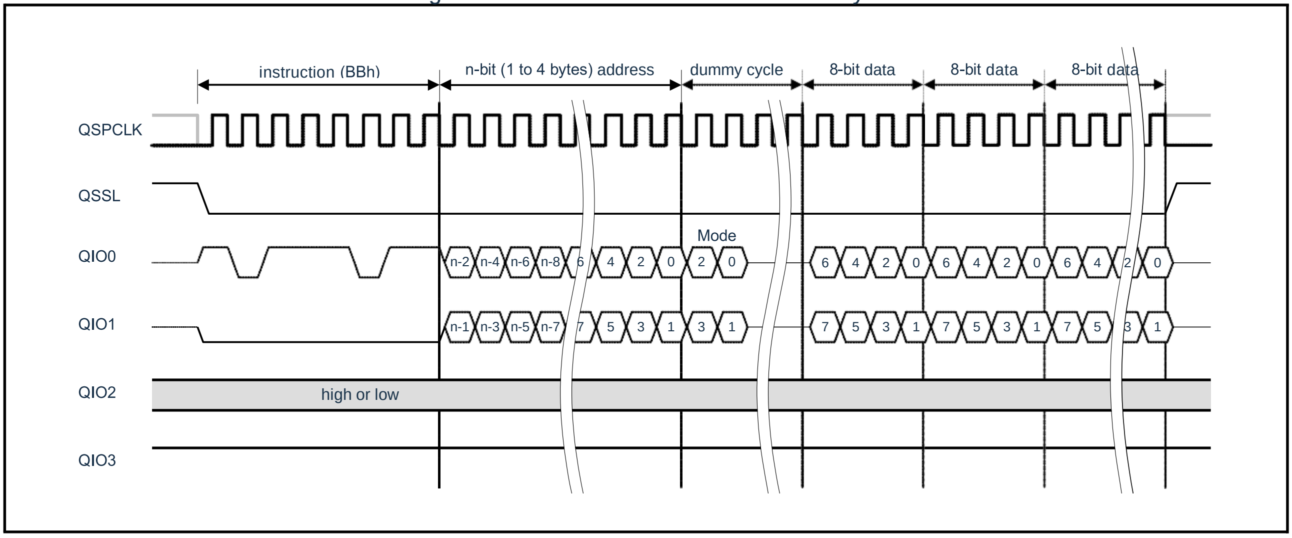
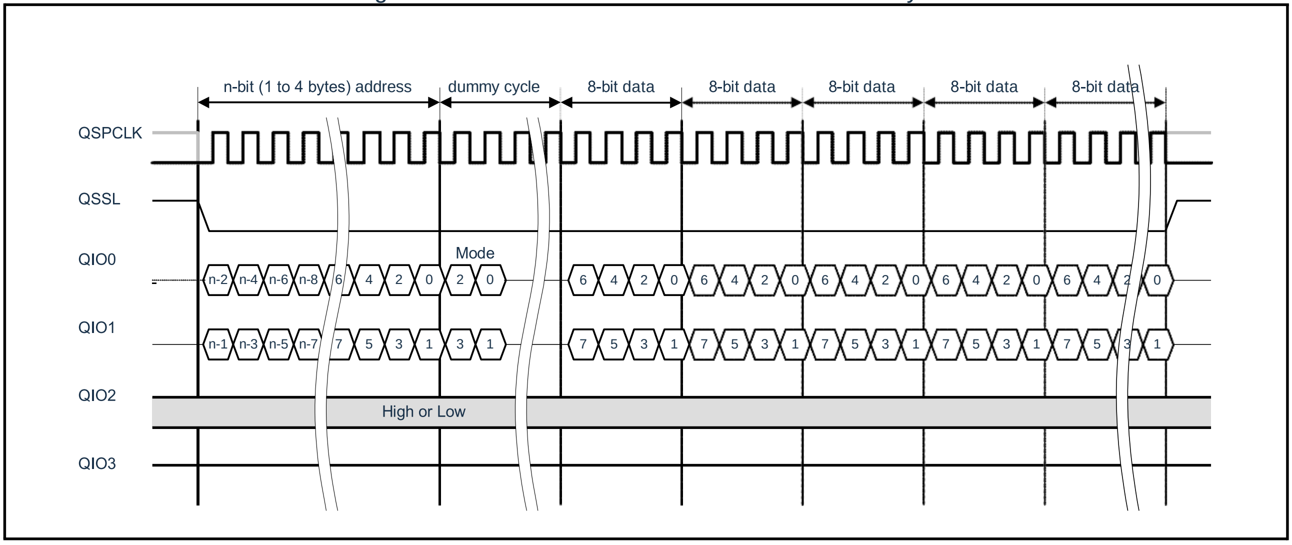


Figure 17-22 XIP mode dual I/O fast read bus cycle



To use the Dual I/O Fast Read isntruction, you must use a serial flash device that supports Dual I/O Fast Read.

17.6.6 Quad output fast read instruction

The quad output fast read instruction is a read instruction that uses four signal lines to receive data. The serial flash select signal is determined when the SPI bus cycle begins. The instruction code (6Bh/6Ch) and address of width 1 to 4 bytes specified by the SFMAS[1:0] bits of the SFMSAC register are sent from QIO0 pins and a specific number of dummy cycles specified by the SFMSDC register are sent. The data is then received on pins QIO0, QIO1, QIO2, and QIO3.

The first two dummy cycles are used to select the XIP mode. When XIP mode is selected, the same instruction used in the current cycle is applied to the next SPI bus cycle, and the instruction code transfer for the next SPI bus cycle is skipped. For more information on XIP mode, refer to Section 17.8, XIP Control.

The SFMSMD register can be used to switch to a quad output for fast reading.

Figure 17-23 Quad output fast read bus cycle

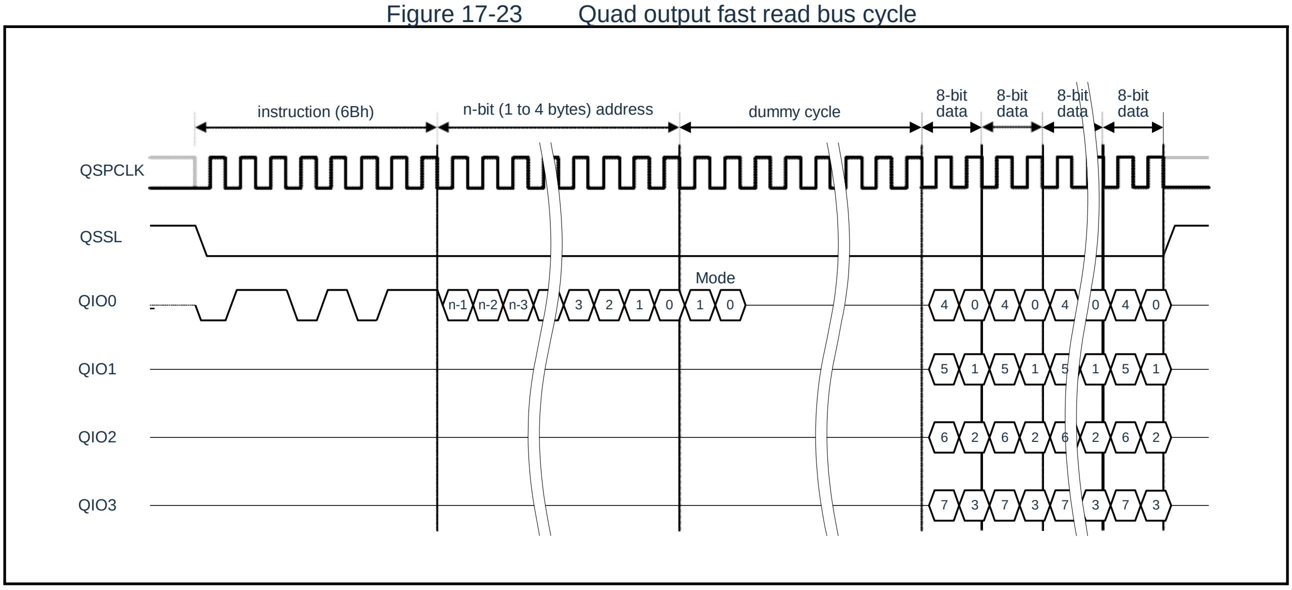
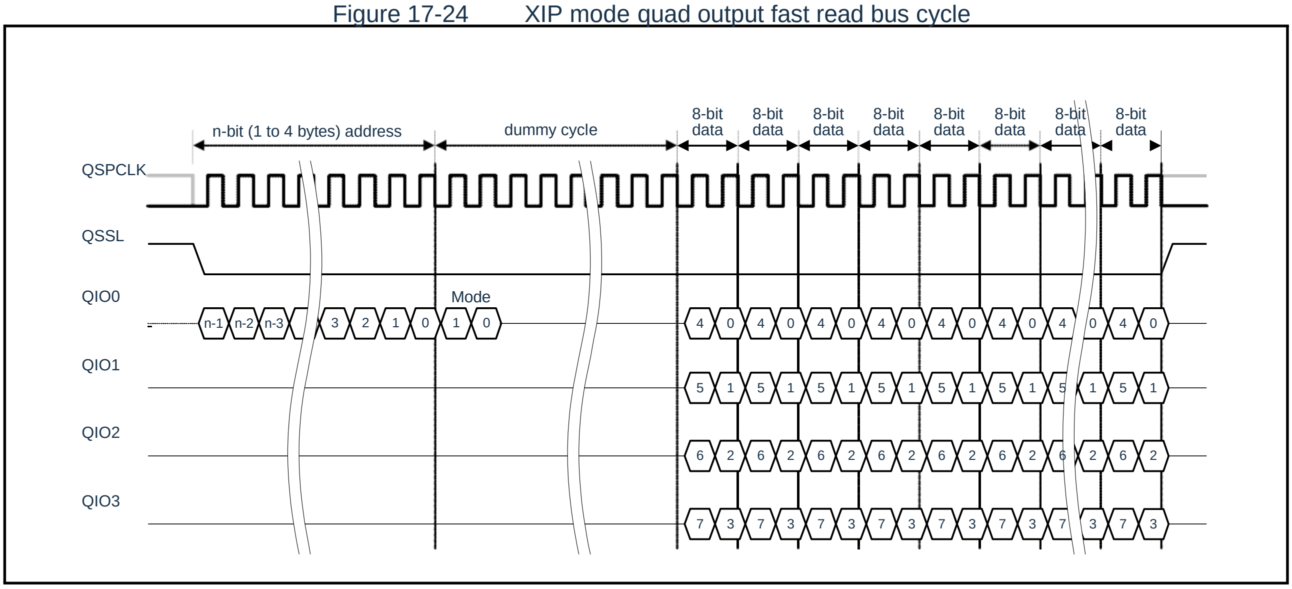


Figure 17-24 XIP mode quad output fast read bus cycle



Note: To use the quad output fast read, you must use a serial flash memory that supports the quad output fast read.

17.6.7 Quad I/O fast read instruction

The quad I/O fast read instruction is a read instruction that uses four signal lines to send addresses and receive data. When the SPI bus cycle begins, the serial flash select signal is set and the instruction code (EBh/ECh) is output. Next, an address of width 1 to 4 bytes specified by the SFMAS[1:0] bits of the SFMSAC register is sent via the QIO0, QIO1, QIO2, and QIO3 pins, and a certain number of dummy cycles specified by the SFMDN[3:0] bits of the SFMSMD register are sent. Data is then received via the QIO0, QIO1, QIO2, and QIO3 pins.

The first two dummy cycles are used to select the XIP mode. When XIP mode is selected, the same instruction used in the current cycle is applied to the next SPI bus cycle, and instruction code transfers for the next SPI bus cycle are skipped. For more information on XIP mode, refer to Section 17.8, XIP Control.

The SFMSMD register can be used to switch to quad I/O fast read.

Figure 17-25 Quad I/O fast read bus cycle

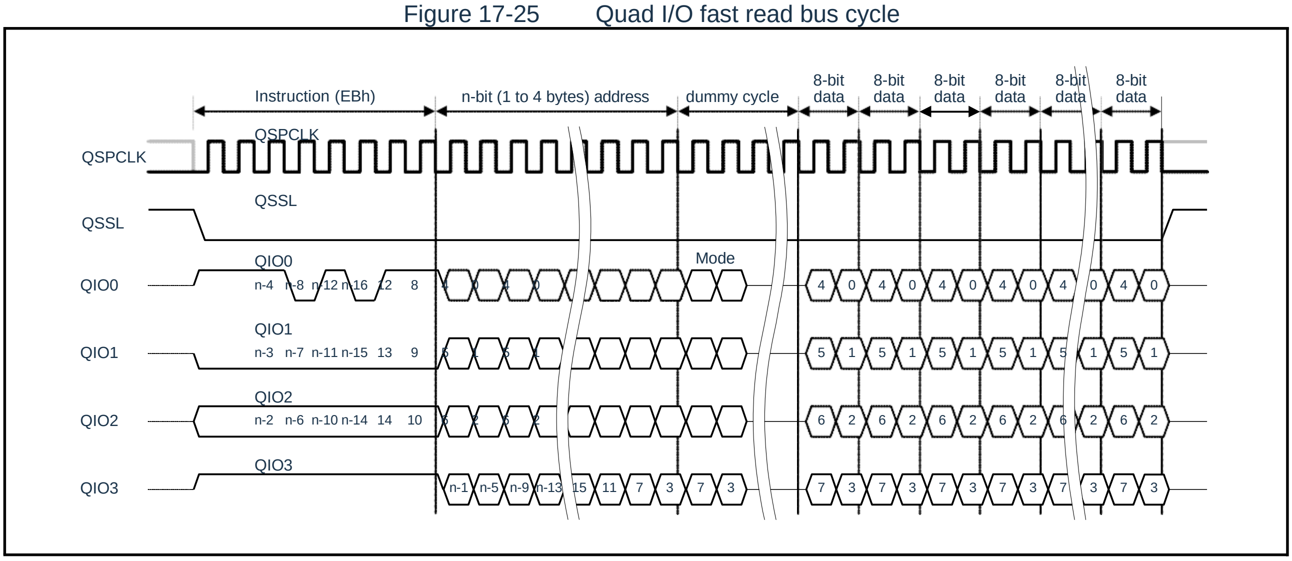
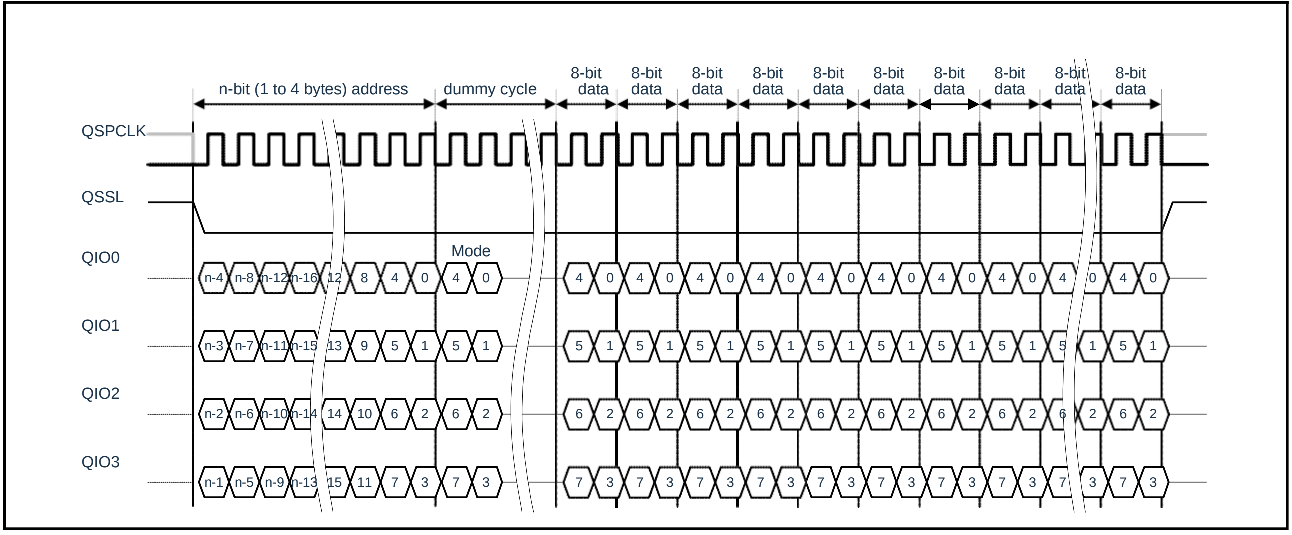


Figure 17-26 XIP mode quad I/O fast read bus cycle

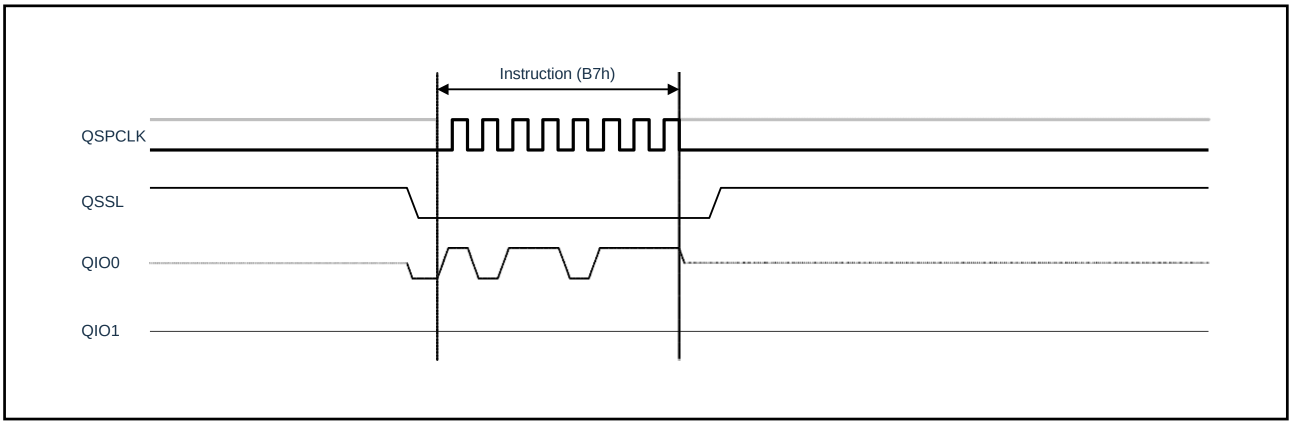


Note: To use the quad I/O fast read command, you must use a serial flash memory that supports quad I/O fast read.

17.6.8 Enter 4-byte mode instruction

The Enter 4-byte mode instruction sets the serial flash address width to 4 bytes. When the SPI bus cycle begins, the serial flash select signal is determined and the instruction code (B7h) is output.

Figure 17-27 Bus cycle for entering 4-byte mode instructions

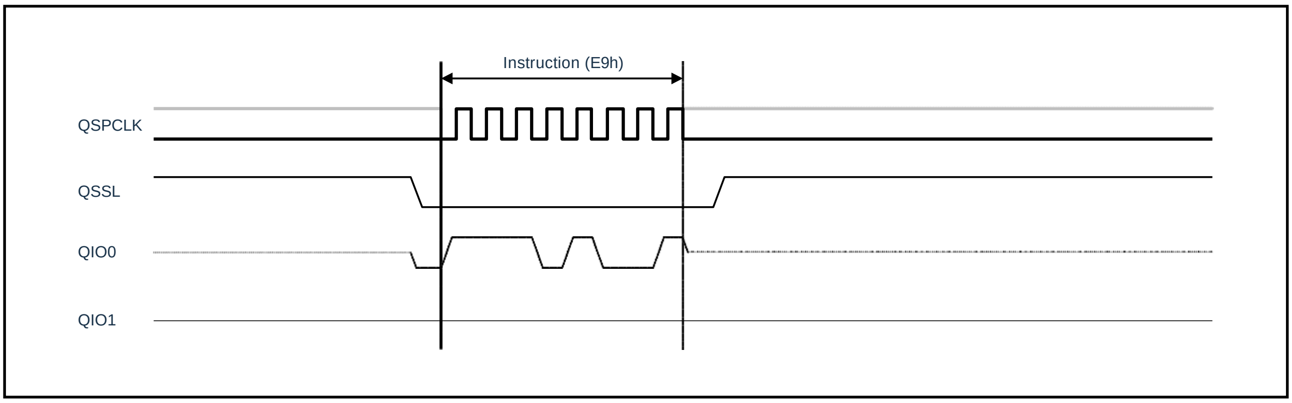


Note: The Enter 4-Byte Mode command is issued regardless of whether the serial flash is in 4-Byte Mode or 3-Byte Mode.

17.6.9 Exit 4-byte mode instruction

The exit 4-byte mode instruction sets the serial flash address width to 3 bytes. When the SPI bus cycle begins, the serial flash select signal is set and the instruction code (E9h) is output.

Figure 17-28 Exit 4-byte mode instruction bus cycle

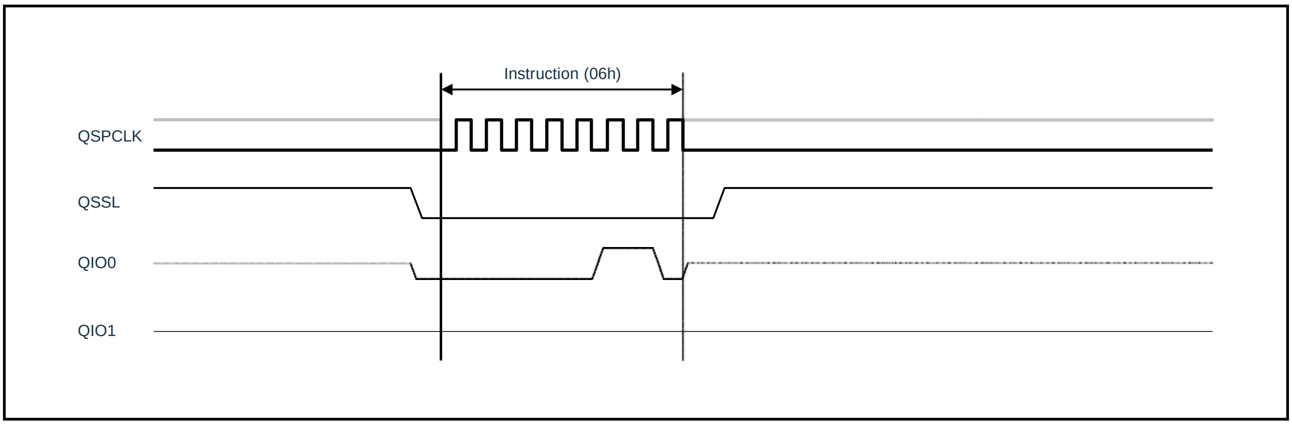


Note: The exit 4-byte mode command is issued whether the serial flash is in 4-byte mode or 3-byte mode.

17.6.10 Write enable instruction

The write enable instruction allows the serial flash address width to be changed. When the SPI bus cycle begins, the serial flash select signal is set and the instruction code (06h) is output.

Figure 17-29 Write enable bus cycle



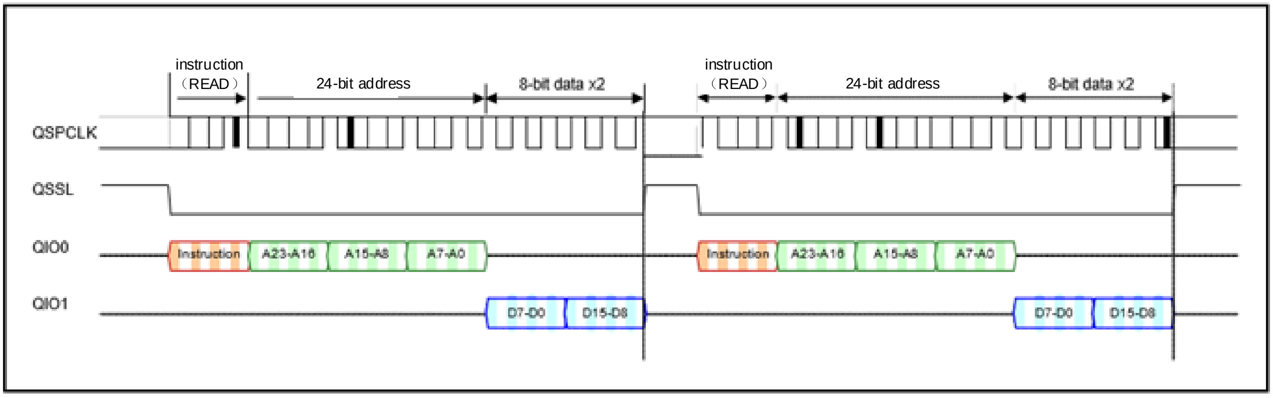
17.7 SPI bus period

17.7.1 Single conversion-based flash read

The ROM read internal bus cycle is converted to SPI bus cycle individually on a one-to-one basis. When a ROM read bus cycle is detected, the QSSL signal is set and the SPI bus cycle begins. When data is received from the serial flash, the QSSL signal is restored and the SPI bus cycle ends.

When another ROM read bus cycle is detected, the QSSL signal is set again and another SPI bus cycle begins after ensuring that the minimum high level width of the QSSL signal is reached.

Figure 17-30 Continuous data reading operations based on an individual conversion



17.7.2 Flash memory reading with pre-read function

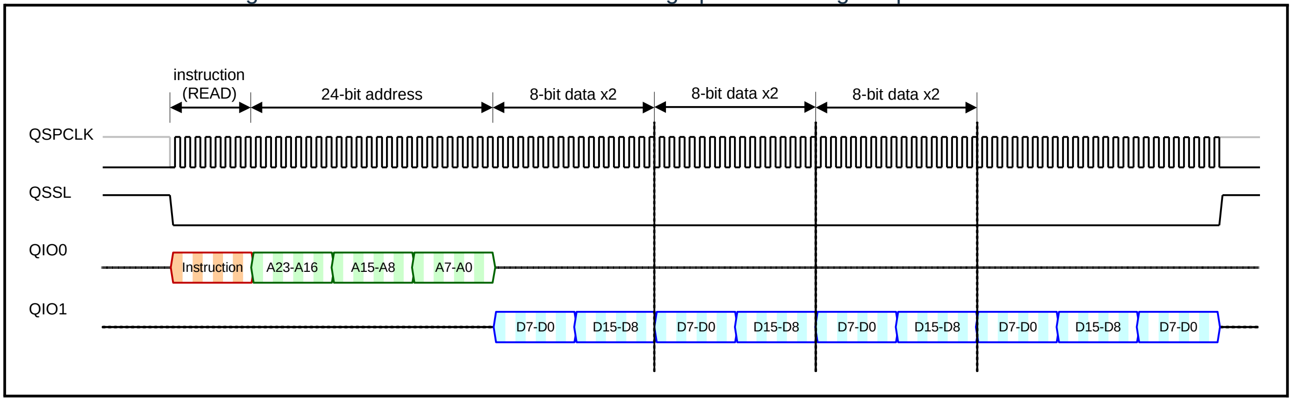
In operations such as executing CPU instructions and transferring block data, data is typically read from adjacent flash addresses in ascending order. Serial flash memory provides the ability to repeat data reception without reissuing instruction codes and addresses. However, if the bus cycles issued by the MCU are converted separately, the SPI bus cycles will be separated from each other, making it impossible to take advantage of this feature of serial flash. QSPI has a prefetch feature for sequential data reception.

To enable the pre-read feature, set the SFMPFE bit in the SFMSMD register to 1. When the pre-read feature is enabled, data is received continuously and stored in the buffer without waiting for another flash read request. When the MCU performs a flash read operation, an address check is performed. If an address match is confirmed, the data in the buffer is passed to the MCU. if an address mismatch is found, the data in the buffer is discarded and a new SPI bus cycle is issued.

The pre-read buffer size is 18 bytes. When the prefetch buffer is full, the SPI bus cycle ends. When the data in the prefetch buffer is read away to create free space, a new SPI bus cycle is automatically started to resume the preread.

When consecutive data is transferred in ascending order, efficient transfers from consecutive data are allowed.

Figure 17-31 Continuous data reading operation using the pre-read function



17.7.3 Pre-read stop

If a ROM read bus cycle from another address occurs during a serial transfer for a pre-read, the excess serial transfer will stop and a new SPI bus cycle will begin. Normally, this serial transfer stop occurs at the data receive byte boundary. However, if the SFMPAE bit in the SFMSMD register is set to 1, the stop can occur at a location other than the byte boundary. To use this feature, the serial flash device must support non-byte boundary stops.

17.7.4 Specify pre-reading destinations directly

When the SFMPFE bit is set up and the QSPI receives an internal bus write access to the QSPI window area. The system acquires it as the pre-reading address and starts pre-reading. Internal bus write accesses to the QSPI window area can only be used to fetch pre-read address data and cannot perform serial flash write operations.

Combining this feature with the Pre-Read Status Polling feature described in 17.7.5 "Pre-Read Status Polling", reduces the consumption on the internal bus when reading data from low- speed serial flash.

Note: When writing to the QSPI window area to indicate a pre-reading destination, write the first byte of the address where the pre-reading is to begin. Writing 2 bytes or more to the QSPI window area will return an error response.

17.7.5 Pre-read status polling

Reading data from low-speed serial flash memory increases system consumption because the internal bus is in a wait state until the SPI receive bus cycle is complete. Providing a pre-reading status polling function reduces this consumption.

The PFOFF bit in the SFMSST register indicate the status of the pre-reading function, and the PFCNT[4:0] bits in the SFMSST register indicate the number of data bytes that have been pre-read. This allows the pre-reading status to be determined by a single CPU operation.

```
//
// copy 1K byte (32bit x 256 word) data from Serial flash to external memory
//
unsigned long *sptr;                          // pointer for the Serial flash
unsigned long *dptr;                          // pointer for the external memory
int i;

SFMSMD |= 0x0040;                             // set SFMPFE bit to enable prefetch
*((volatile unsigned char *) sptr) = 0;       // make the TAG valid to start prefetch

for (i = 0; i < 256; i++){
while ((SFMSST & 0x00FF) < 0x04){};   // waiting for 4 byte data received
*(dptr++) = *(sptr++);
}
```

Note: When executing a polling program, set the program outside of the serial flash or enable the instruction buffer, otherwise the pre-reading target will switch to the instruction code frequently. This affects the effectiveness of polling and can lead to infinite loops because the pre-reading buffer is not filled.

17.7.6 Flash memory read using SPI bus cycle expansion

If the SFMSE[1:0] bits in the SFMSMD register are set to a value other than 00, the QSPI will wait for the next flash read, suspend the SPI bus cycle while stopping the QSPCLK signal, and hold the QSSL signal low after obtaining data from the serial flash.

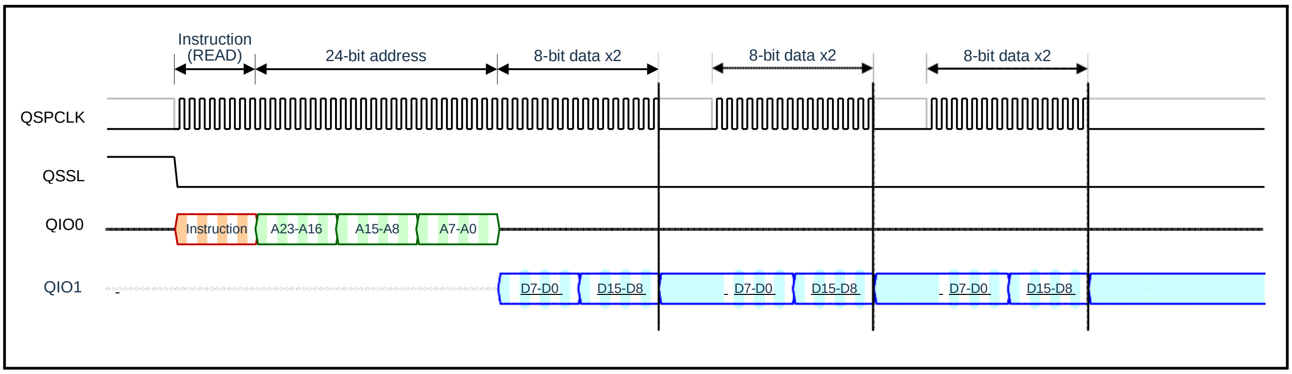
If the address of the next flash read is consecutive, the QSPCLK signal flip-flop is restarted to continue receiving subsequent data. If the address of the next flash read is not consecutive, the QSSL signal is driven high to end the suspended SPI bus cycle. A new SPI bus cycle then begins.

When reading data intermittently from ascending sequential addresses, this feature enables efficient transfer operations by reducing the overhead of instruction code and address transfers.

The SPI bus cycle expansion time can be selected by the SFMSE[1:0] bits of the SFMSMD register. When the specified expansion time has elapsed, the QSSL signal returns high to automatically end the suspended SPI bus cycle.

If the SFMSE[1:0] bits are set to 11, the QSSL expands indefinitely. This increases the overhead of the serial flash device.

Figure 17-32 Continuous data read operation using SPI bus cycle extension



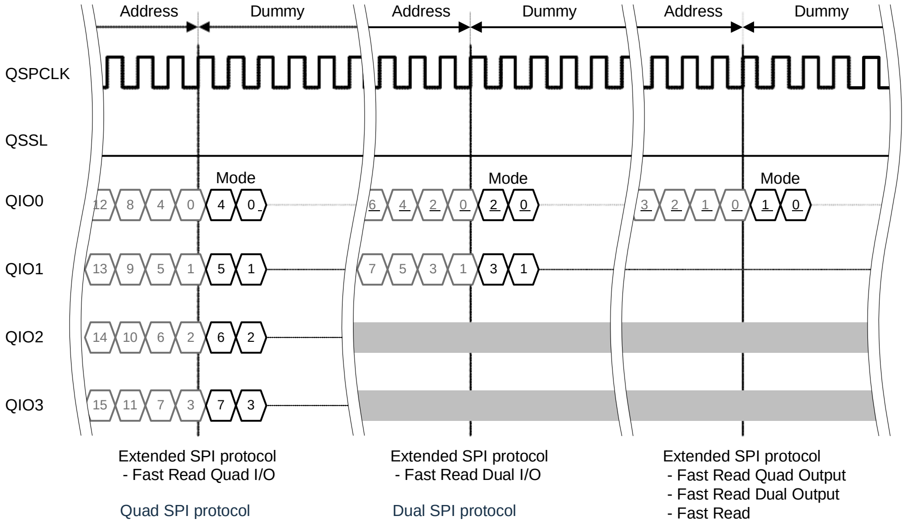
17.8 XIP control

Some serial flash devices allow skipping the instruction code for receiving flash reads to reduce latency. This instruction code skip function is selected when pattern data is received during the dummy cycle of the previous serial bus cycle.

During the dummy cycle of a fast read instruction, the QSPI controls the XIP mode of the serial flash by sending the mode data set in the SFMXD[7:0] bits of the SFMSDC register during the first 2 cycles, as shown in Figure 17-33.

The mode data to enable XIP mode varies from one serial flash device to another. Consider this when setting the mode data in the SFMXD[7:0] bits of the SFMSDC register.

Figure 17-33 XIP mode control data



17.8.1 Setting XIP mode

It is assumed that the specified XIP mode is set for the serial flash device via the SFMXD[7:0] bits of the SFMSDC register. After the SFMXEN bits are set to 1, the mode data specified by the SFMXD[7:0] bits of the SFMSDC register is transferred to the serial flash device during the next fast read dummy cycle. XIP mode is then enabled in the QSPI and the serial flash device. You can confirm that the actual XIP mode selection process is complete by reading the SFMXST bits in the SFMSDC register to see if they are 1.

Note: The SFMXD[7:0] bits in the SFMSDC register are the XIP mode setting data for the serial flash device. The enable of the XIP mode of the serial flash controller is controlled by the SFMXEN bits and is regardless of the SFMXD[7:0] setting in the SFMSDC register.

17.8.2 Releasing XIP mode

When the XIP mode release configuration is specified for the serial flash in SFMXD[7:0] of the SFMSDC register. After the SFMXEN bits are set to 0, the mode data specified in the SFMXD[7:0] bits of the SFMSDC register is transferred to the serial flash during the first two cycles of the next fast read dummy cycle. The XIP mode is then disabled in the QSPI and serial flash devices. You can confirm that the actual XIP mode release process is complete by reading the SFMXST bits in the SFMSDC register to see if they are 0.

Note: The SFMXD[7:0] bits in the SFMSDC register are the XIP mode setting data for the serial flash device. The disabling of the XIP mode of the serial flash controller is controlled by the SFMXEN bits and is regardless of the SFMXD[7:0] setting in the SFMSDC register.

17.9 QIO2 and QIO3 pin status

The state of the QIO2 and QIO3 pins depends on the serial interface read mode specified by the SFMRM[2:0] bits in the SFMSMD register.

Table 17-9 QIO2 and QIO3 pin status

SFMSMD.SFMRM[2:0] bits	QIO2 pin state[*1]	QIO3 pin state[*2]	Remark
111	Settings are prohibited		
110			
101	As input or output of serial data signal (Hi-Z in standby)	As input or output of serial data signal (Hi-Z in standby)	Quad I/O fast reading
100			Quad output fast reading
011	Output the SFMWPL bit of the SFMPMD register (output initial value is low)	Output high level	Dual I/O for fast reads
010			Dual output for fast reading
001			Quick Read
000			Read (initial state)

Note 1. Serial Flash can also be used with the QIO2 pin as a WP function.

Note 2. Serial Flash can also use the QIO3 pin as a hold function or reset function.

17.10 Direct communication mode

17.10.1 About direct communication

QSPI can read serial flash memory contents by automatically converting ROM read bus cycles to SPI bus cycles. However, serial flash devices have many different functions in addition to reading memory data, including ID information reading, erasing, programming, and status information reading. There is no standardized instruction set for the use of these functions, and different vendors are adding more and more functions to different devices. Therefore it is difficult to support these functions through hardware control.

QSPI provides this method of direct communication with the serial flash memory. The ability to set arbitrary SPI bus cycles by software, depending on the needs of the serial flash device, provides flexible support for these serial flash devices.

17.10.2 Direct communication mode

To communicate directly with a serial flash device, set the DCOM bit in the SFMCMD register to 1 to switch to direct communication mode. When direct communication mode is selected, standard flash memory read operations are disabled. Terminate the direct communication mode by setting the DCOM bit in the SFMCMD register to 0 before performing a standard flash access.

Note: If the QSPI is set to XIP mode, XIP mode must be terminated before direct communication mode can be initiated.

17.10.3 SPI bus cycles in direct communication

The SPI bus cycle in direct communication begins with the first access to the SFMCOM port and ends with a write to the SFMCMD register after a series of I/O operations are performed through the SFMCOM port. Writes to the SFMCOM port are converted to single-byte sends to the SPI bus, and reads from the SFMCOM port are converted to single-byte receives from the SPI bus.

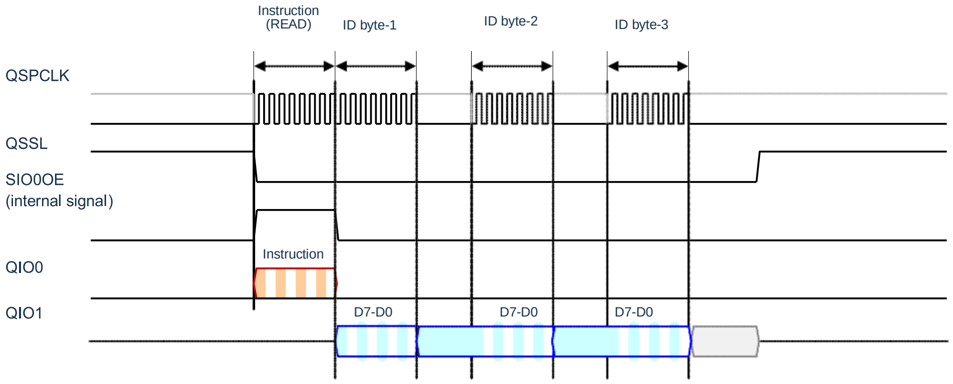
During the operation from the first access to the SFMCOM port to the last write to the SFMCMD register, the Serial Flash Select signal remains active to notify the Serial Flash that a series of SPI bus cycles are in progress.

Note: In direct communication mode, writes to registers other than SFMCMD (including SFMSMD, SFMSSC, SFMSKC, SFMSST, SFMCST, SFMSIC, SFMSAC, SFMSDC, SFMSPC, and SFMPMD) are disabled. In this configuration, writing to register areas other than the SFMCOM port terminates the SPI bus cycle. However, terminating the SPI bus cycle by writing to register areas other than SFMCMD does not guarantee proper operation.

The following is a sample program for direct communication.

```
//### CAUTION! ### This code must be outside the Serial flash that is going to be operated.

// Define specific instruction codes of the target Serial flash device.
#define Instruction_FREAD 0x0B  // Fast Read
#define Instruction_RDSR 0x05    // Read Status register
#define Instruction_RDID 0x9F     // Read Identification
#define Instruction_WREN 0x06    // Write Enable
#define Instruction_CERA 0xC7    // Chip Erase

unsigned char mfid, mtype, mcap, data, temp;

SFMCMD = 0x01; // Enable direct operation

// Get the device identification assigned by JEDEC.
SFMCOM = Instruction_RDID;          // put “Read Identification” instruction (open SPI bus cycle)
mfid = (unsigned char) SFMCOM;     // get “Manufacturer Identification”
mtype = (unsigned char) SFMCOM;  // get “Memory Type”
mcap = (unsigned char) SFMCOM;   // get “Memory Capacity”
SFMCMD = 0x01h;                         // close SPI bus cycle

// Get one byte from the address 0x012345h.
SFMCOM = Instruction_FREAD;      // put “Fast Read” instruction (open SPI bus cycle)
SFMCOM = 0x01;                         // put upper byte of the address 0x012345
SFMCOM = 0x23;                         // put middle byte of the target address 0x012345
SFMCOM = 0x45;                         // put lower byte of the target address 0x012345
temp = (unsigned char) SFMCOM;  // get one byte dummy code for FAST READ transaction
data = (unsigned char) SFMCOM;   // get the data
SFMCMD = 0x01;                         // close SPI bus cycle

// Erase All contents.
SFMCOM = Instruction_WREN;  // put “Write Enable” instruction (open SPI bus cycle)
SFMCMD = 0x01;                    // close SPI bus cycle
SFMCOM = Instruction_CERA;   // put “Chip Erase” instruction (open SPI bus cycle)
SFMCMD = 0x01;                    // close SPI bus cycle
SFMCOM = Instruction_RDSR;   // put “Read Status Register” instruction (open SPI bus cycle)
while (SFMCOM & 0x01){};        // Polling “Write Progress Bit” until completion
SFMCMD = 0x01;                    // close SPI bus cycle

SFMCMD = 0x00; // Disable direct operation
```

Figure 17-34 Example of direct communication timing for reading IDs



```
(1)   SFMCMD = 01h;                        // Enable direct operation
                                           // Get the device identification assigned by JEDEC.
(2)   SFMCOM = Instruction_RDID;           // Put “Read Identification” instruction (open the transaction)
(3)   mfid = (unsigned char) SFMCOM;       // Get ID byte-1 “Manufacturer Identification”
(4)   mtype = (unsigned char)              // Get ID byte-2 “Memory Type”
      SFMCOM;
(5)   mcap = (unsigned char) SFMCOM;      // Get ID byte-3 “Memory Capacity”
(6)   SFMCMD = 01h;                        // Close the transaction
(7)   SFMCMD = 00h;                        // Disable direct operation
```

Note: When using the extended SPI protocol in direct communication mode, you must use the standard read or fast read instructions to access the contents of the serial flash memory. In this configuration, QSPI does not support dual output fast read, dual I/O fast read, quad output fast read, or quad I/O fast read transfers. When these high-speed read operations are required, use standard flash access.

17.11 Operation

17.11.1 Procedure for modifying multiple control register settings

The QSPI control register settings can be dynamically modified during system operation. However, when multiple control register settings are modified sequentially, the SPI bus cycle may occur before all registers are updated. The order in which registers are set must be beware, and the SPI bus timing specification must be satisfied during all register setting modifications.

```
//
// Making QSPCLK faster
//
SFMSMD = 0x0041;  // SFMPAE: 0 SFMPFE: 1 SFMSE:00 SFMRM:01 (prefetch enable fast read)
SFMSSC = 0x04;    // SFMSLD: 0 SFMSHD: 0 SFMSW:4 (minimum QSSL high width = 5 sck)
SFMSKC = 0x00;    // SFMDTY: 0 SFMDV: 0 (1/2 mode) ### switch clock speed last ###

//
// Making QSPCLK slower
//
SFMSKC = 0x06;    // SFMDTY: 0 SFMDV:6 (1/8 mode) ### switch clock speed first ###
SFMSSC = 0x01;    // SFMSLD: 0 SFMSHD:0 SFMSW: 1 (minimum QSSL high width = 2 sck)
SFMSMD = 0x0040;  // SFMPAE: 0 SFMPFE:1 SFMSE: 00 SFMRM:00 (prefetch enable, standard read)
```

17.12 Interrupt

The QSPI requests an interrupt when the EROMR bit of the SFMCST register is 1. When a ROM read access is detected in direct communication mode, the EROMR bit changes to 1. The interrupt request is held until write 0 to clear the EROMR bit.

Chapter 18 Serial Sound Interface Enhanced (SSIE)

18.1 Overview

The Serial Sound Interface Enhanced (SSIE) can transmit and receive audio data to and from multiple devices that support different audio data formats, such as I2S and mono.

18.2 Specification of SSIE

Table 18-1 Specification of SSIE

Item		Description
Number of channels		One channel, SSIE0
Communication method		Master or Slave Transmit and receive (full duplex communication).
Communication format		I2S format Mono format
Serial Data		Left-aligned or right-aligned data Data delay from SSILRCK/SSIFS to SSITXD0/SSIRXD0 with optional delay of 1 clock cycle or no delay System word length: 8, 16, 24, 32, 48, 64, 128, or 256 bits Data word length: 8, 16, 18, 20, 22, 24, or 32 bits Filling polarity: low or high
Bit Clock （SSIBCK）	Master	Two clock sources: SSIMCLK, TO001A (Timer4 output) Clock source frequency division: 1/1, 1/2, 1/4, 1/6, 1/8, 1/12, 1/16, 1/24, 1/32, 1/48, 1/64, 1/ 96, and 1/128 Option to supply or stop the clock when communication is interrupted
	Master or slave	Selectable polarity (rising edge or falling edge)
LR clock/frame synchronization (SSILRCK/SSIFS)	Transmit	Selectable polarity (low or high) Option to supply or stop the clock when communication is interrupted
Transmit data (SSITXD0) and receive data (SSIRXD0)	Transmit	Selectable mute method (transmit FIFO data or transmit data fixed to 0)
FIFO	Capacity	Transmit or receive FIFO: 4 bytes × 8 segments
	Data Alignment	Optional data alignment between FIFO and shift registers (left-aligned or right-aligned)
Interrupt source	Interrupt output	Communication error/Communicati on idle receive data is full transmit data is empty
Low power consumption function		In master mode, you can select whether to provide audio clock
Module stop function		To reduce power consumption, the module can be set to stop

Table 18-2 Definition of Terms

Terminology	Description
Start Trigger	The first signal edge on the SSILRCK/SSIFS pin that meets the requirement initiates communication specified by LRCKP
Frame Boundary	The first data of the frame at the beginning of the SSIE transmission or the last data of the frame at the end of the SSIE transmission
Number of frame words	Number of sound channels per frame
System word length	Number of bits per channel
Data word length	Effective number of bits per channel
Control bits of the communication format	• SSICR register: DWL, SWL, LRCKP, SPDP, SDTA, PDTA and DEL bits • SSIFCR register: BSW bit • SSITDMR register: OMOD bit • SSISCR register: TDES and RDFS bits

Figure 18-1 SSIE communication format

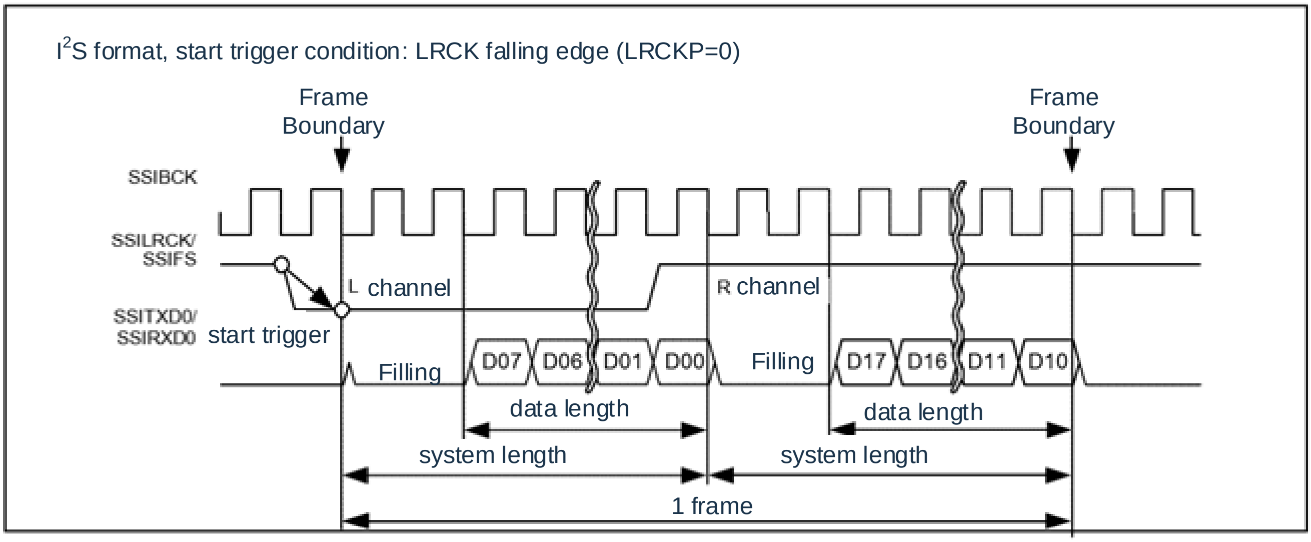
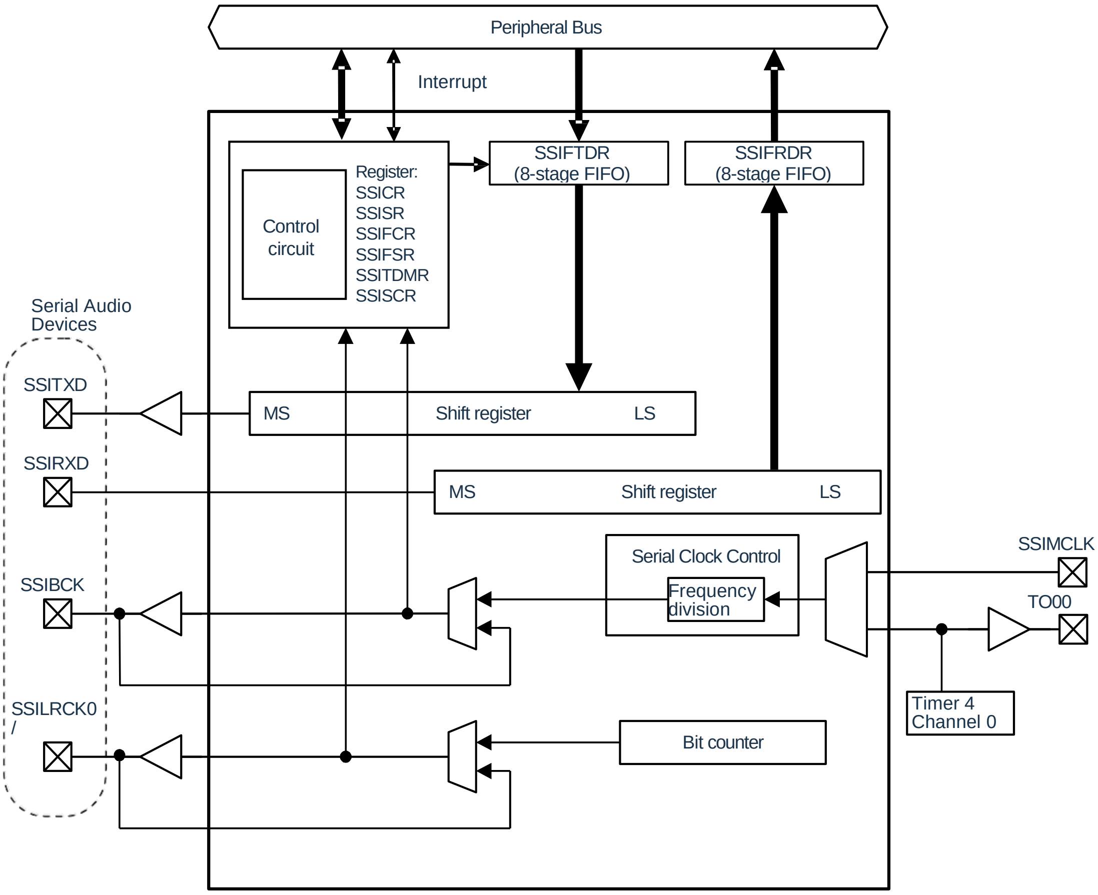


Figure 18-2 Block diagram of SSIE (SSIE0)



Note:

SSICR: Control register

SSISR: Status register

SSIFCR: FIFO control register

SSIFSR: FIFO status register

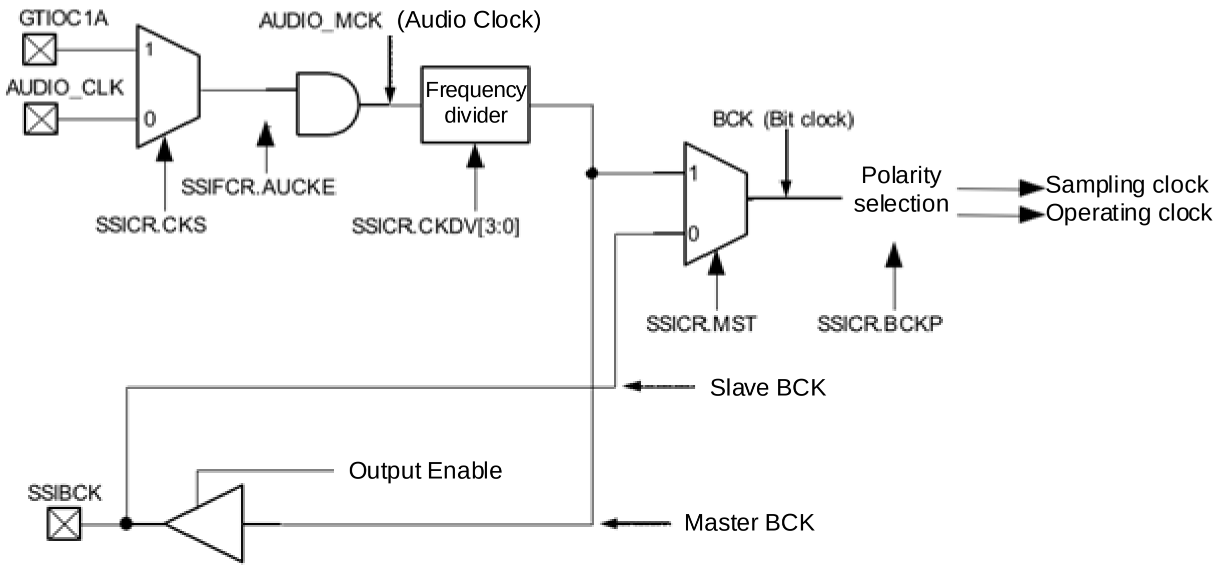
SSITDMR: Audio format register

SSIFTDR: Transmit FIFO data register

SSIFRDR: Receive FIFO data register

SSISCR: Status control register

Figure 18-3 SSIE clock settings



Note:

SSICR: Control register

SSISR: Status register

SSIFCR: FIFO control register

SSIFSR: FIFO status register

SSITDMR: Audio format register

SSIFTDR: Transmit FIFO data register

SSIFRDR: Receive FIFO data register

SSISCR: Status control register

18.3 Description of register

Register list:

Base Address	Offset Address	Register Name	R/W	Reset value
0x40090000	0x000	SSICR	R/W	0000h
	0x004	SSISR	R/W	2000h
	0x010	SSIFCR	R/W	0000h
	0x014	SSIFSR	R/W	0100h
	0x018	SSIFTDR	R/W	0000h
	0x01C	SSIFRDR	R/W	0000h
	0x020	SSITDMR	R/W	0000h
	0x024	SSISCR	R/W	0000h

18.3.1 Control register (SSICR)

This register allows you to select the audio clock, control interrupt requests, select the data format and set the operation mode

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	CKS	TUIEN	TOIEN	RUIEN	ROIEN	IIEN	—	—	—	DWL[2:0]			SWL[2:0]		
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	MST	BCKP	LRCKP	SPDP	SDTA	PDTA	DEL	CKDV[3:0]				MUEN	—	TEN	REN
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	REN	Transmit and receive enable Note 2	00: Sending and receiving are not allowed 01: Receiving allowed (start receiving) 10: Allow sending (start sending) 11: Sending and receiving allowed (start sending and receiving)	R/W
b1	TEN			R/W
b2	-	Reserved	Can only write 0 and read 0	-
b3	MUEN	Mute Enable	0: Mute is not allowed at the next frame boundary 1: Mute is allowed at the next frame boundary	R/W
b7~b4	CKDV[3:0]	Bit clock division ratio selection Note 1	0000：AUDIO_MCK 0001：AUDIO_MCK/2 0010：AUDIO_MCK/4 0011：AUDIO_MCK/8 0100：AUDIO_MCK/16 0101：AUDIO_MCK/32 0110：AUDIO_MCK/64 0111：AUDIO_MCK/128 1000：AUDIO_MCK/6 1001：AUDIO_MCK/12 1010：AUDIO_MCK/24 1011：AUDIO_MCK/48 1100：AUDIO_MCK/96 1101：setting is prohibited 1110：setting is prohibited 1111：setting is prohibited	R/W
b8	DEL	Serial data delay selection Note 1	0: One beat of SSIBCK clock delay between SSILRCK/SSIFS and SSITXD0/SSIRXD0 1: In mono format, there is no delay between SSILRCK/SSIFS and SSITXD0/SSIRXD0, and this bit controls the waveform of SSILRCK/SSIFS. For details, refer to 18.4.2 Mono format	R/W
b9	PDTA	Alignment options for data storage Note 1	0：左对齐存放(SSIFTDR, SSIFRDR) 1：右对齐存放(SSIFTDR, SSIFRDR)	R/W
b10	SDTA	Serial Data Alignment Selection Note 1	0: Send and receive serial data first and then transmit the padding bits 1: Send and receive padding bits first and then send and receive serial data	R/W
b11	SPDP	Fill polarity selection Note 1	0: Left-aligned storage (SSIFTDR, SSIFRDR) 1: Right-aligned storage (SSIFTDR, SSIFRDR)	R/W
b12	LRCKP	Initial value of LR Clock/frame synchronization signal and polarity selection Note 1	0:The initial value of SSILRCK/SSIFS is a high potential, and the falling edge triggers the start of a frame 1: The initial value of SSILRCK/SSIFS is low, and the rising edge triggers the start of a frame	R/W
b13	BCKP	Clock polarity selection Note 1	0: SSILRCK/SSIFS and SSITXD0/SSIRXD0 change on the falling edge (SSILRCK/SSIFS and SSIRXD0 are sampled on the rising edge of SSIBCK) 1: SSILRCK/SSIFS and SSITXD0/SSIRXD0 change at rising edge (SSILRCK/SSIFS and SSIRXD0 are sampled at the falling edge of SSIBCK)	R/W
b14	MST	Master enable Note 1	0: Slave mode communication 1: Master mode communication	R/W
b15	-	Reserved	Can only write 0 and read 0	-

Bit	Symbol	Bit name	Description	R/W
b18~b16	SWL[2:0]	System length selection Note 1	b18~b16 000 : 8 bits 001 : 16 bits 010 : 24 bits 011 : 32 bits 100 : 48 bits 101 : 64 bits 110 : 128 bits 111 : 256 bits	R/W
b21~b19	DWL[2:0]	Data length selection Note 1	b21~b19 000 : 8 bits 001 : 16 bits 010 : 18 bits 011 : 20 bits 100 : 22 bits 101 : 24 bits 110 : 32 bits 111 : Setting is prohibited	R/W
b24~b22	-	Reserved	Can only write 0 and read 0	-
b25	IIEN	Idle mode interrupt output enable	0: Output idle mode interrupt is not allowed 1: Allow output of idle mode interrupts	R/W
b26	ROIEN	Receive overflow interrupt output enable	0: Output receive overflow interrupt is not allowed 1: Output receive overflow interrupt is allowed	R/W
b27	RUIEN	Receive underflow interrupt output enable	0: Do not allow the output to receive underflow interrupts 1: Allow the output to receive underflow interrupts	R/W
b28	TOIEN	Send overflow interrupt output enable	0: Do not allow the output to send overflow interrupts 1: Allow the output to send overflow interrupts	R/W
b29	TUIEN	Send underflow interrupt output enable	0: Do not allow the output to send underflow interrupts 1: Allow the output to send underflow interrupts	R/W
b30	CKS	Audio clock selection during master mode communication Note 1	0: Select AUDIO_CLK input 1: Select GTIOC1A (GPT output).	R/W
b31	-	Reserved	Can only write 0 and read 0	-

Note 1. When SSIE is in communication state (SSISR.IIRQ=0), writing these bits is prohibited, otherwise the subsequent operation is not predictable.

Note 2. If the TEN bit or REN bit is overwritten, make sure that the SSISR.IIRQ bit is in the target state. Otherwise the subsequent operation is unpredictable. For example, check that the SSISR.IIRQ bit is 0 when transmitting or receiving is allowed and 1 when transmitting or receiving is disabled.

1) TEN and REN bits (transmit and receive enable)

The TEN and REN bits allow or disallow transmitting and receiving. When the TEN bit or REN bit is written to "1", the corresponding transmission operation starts while the SSILRCK/SSIFS signal is triggered to start. When the TEN bit or REN bit is written to "0", the current communication operation stops at the next frame boundary. When using SSIE for both transmitting and receiving, write "1" to both bits at the same time. When you want to stop the transmit and receive communication of SSIE, write "0" to the TEN bit and REN bit to disable the transmission and reception.

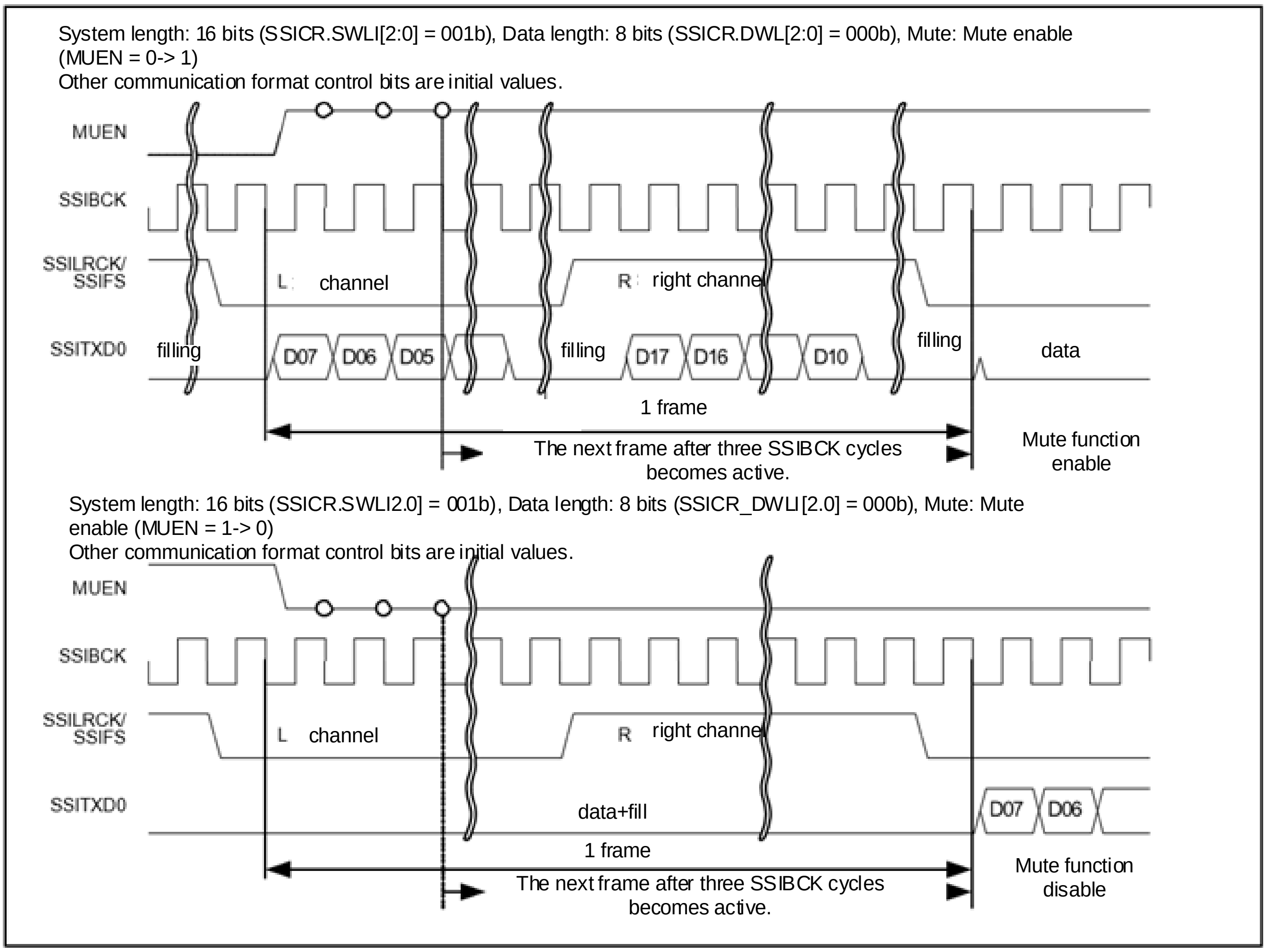
To stop SSIE before reaching the frame boundary, perform a software reset.

2) MUEN bit (mute enable)

The MUEN bit allows or disables the muting of the SSITXD0 pin data output. When this bit is set to 1 in the middle of a frame, the output of SSITXD0 becomes 0 at the next frame boundary. when this bit is set to 0 in the middle of a frame, the output of SSITXD0 becomes data in the transmit FIFO data register at the next frame boundary. This bit controls data only. It does not affect the generation of status flags and interrupt signals.

This bit value can only be changed after the communication format to be used has been set.

Figure 18-4 Transmit data to set the mute function



System length: 16 bits (SSICR.SWLI[2:0] = 001b), Data length: 8 bits (SSICR.DWL[2:0] = 000b), Mute: Mute enable (MUEN = 0-> 1)
Other communication format control bits are initial values.

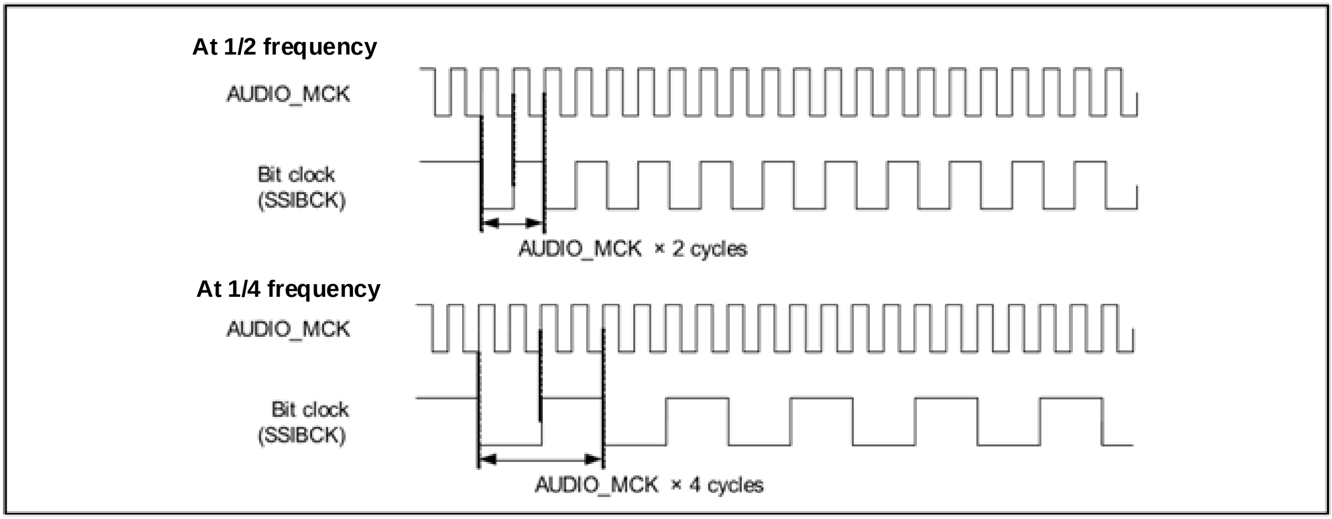
System length: 16 bits (SSICR.SWLI2.0] = 001b), Data length: 8 bits (SSICR_DWLI[2.0] = 000b), Mute: Mute enable (MUEN = 1-> 0)
Other communication format control bits are initial values.

3) CKDV[3:0] bits (Clock division selection)

In master mode (MST=1), set the division ratio of bit clock and AUDIO_MCK clock according to CKDV[3:0]. In slave mode (MST=0), the setting of CKDV[3:0] is invalid.

This bit is written when AUDIO_MCK is stopped. See the description of the AUCKE bit in the SSIFCR register for details.

Figure 18-5 Sampling frequency for master mode communication

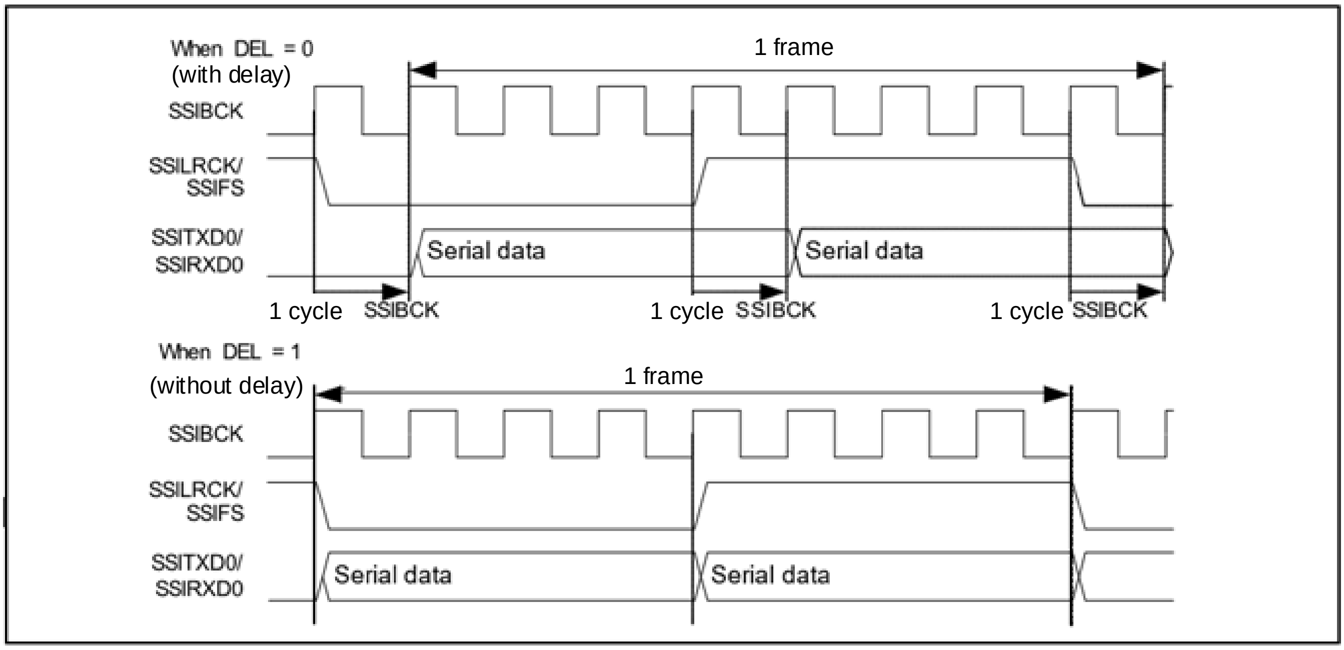


4) DEL bit (serial data delay selection)

The DEL bit selects whether there is a delay between SSILRCK/SSIFS and SSITXD0/SSIRXD0.

For the I2S format, set the DEL bit to 0. When using the mono format, this bit setting controls the high-level width of SSILRCK/SSIFS. When using a compatible communication format, specify the DEL bit setting that initiates communication.

Figure 18-6 Setting the serial data delay



5) PDTA bit (alignment selection for data storage)

The PDTA bit selects how to align the data. When 32-bit word length is set (SSICR.DWL[2:0]=110b), this bit is invalid.

Figure 18-7 Alignment of send data storage

First send data	Second send data	Third sends data	Fourth send data

DWL[2:0]	SSIFTDR		Transmit shift register
	PDTA = 0 (left-aligned)	PDTA = 1 (right-aligned)	
000 (8 bits)	7 0 Invalid 7 0 Invalid 7 0 Invalid 7 0 Invalid	Setting is disabled	7 0 Invalid 7 0 Invalid 7 0 Invalid 7 0 Invalid
001 (16 bits)	15 0 Invalid 15 0 Invalid 15 0 Invalid 15 0 Invalid	Setting is disabled	15 0 Invalid 15 0 Invalid 15 0 Invalid 15 0 Invalid
010 – 100 18-bit: X=17 20-bit: X=19 22-bit: X=21 24-bit: X=23	X 0 Invalid X 0 Invalid X 0 Invalid X 0 Invalid	Invalid X 0 Invalid X 0 Invalid X 0 Invalid X 0	X 0 Invalid X 0 Invalid X 0 Invalid X 0 Invalid
110 (32 bits)	31 0 31 0 31 0 31 0	Setting is disabled	31 0 31 0 31 0 31 0
111 (Setting is disabled)			

Figure 18-8 **Alignment of receive data storage**

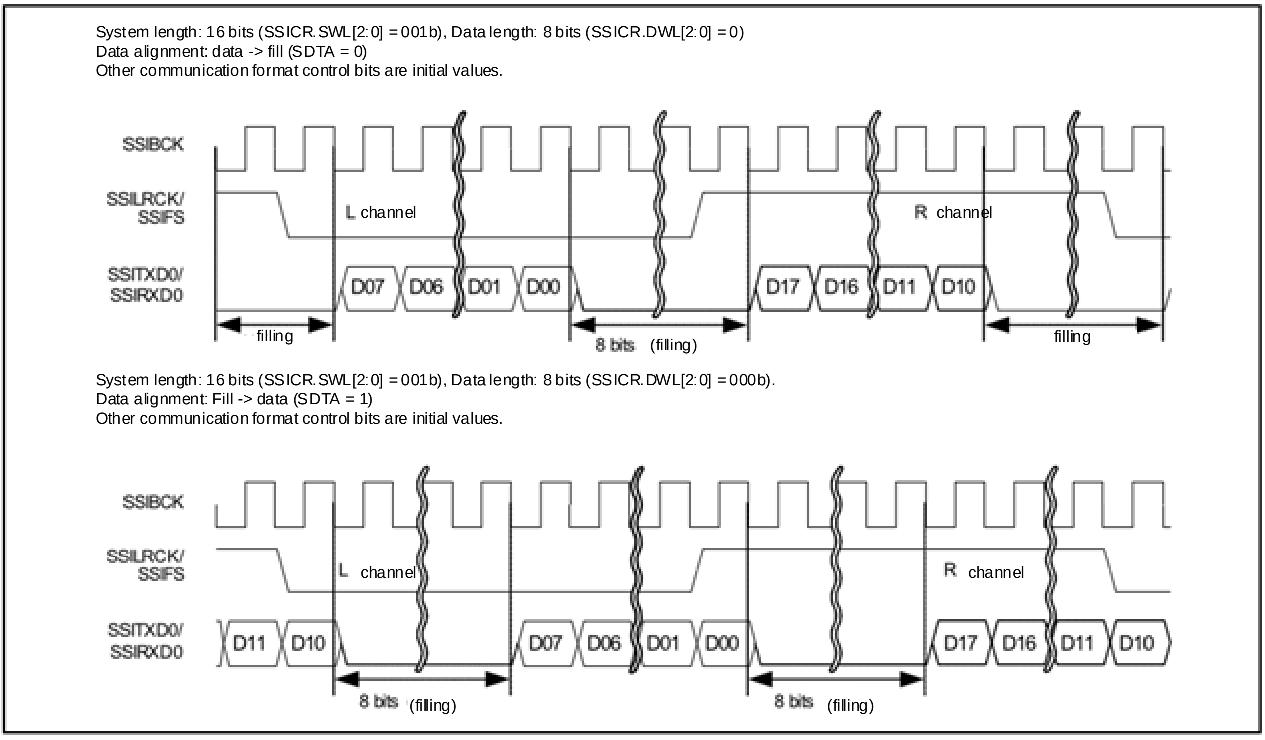
First send data	Second send data	Third sends data	Fourth send data

DWL[2:0]	Receive shift register	SSIFRDR	
		PDTA = 0 (left-aligned)	PDTA = 1 (right-aligned)
000 (8 bits)	Invalid 7 0 Invalid 7 0 Invalid 7 0 Invalid 7 0	7 0 Invalid 7 0 Invalid 7 0 Invalid 7 0 Invalid	Setting is disabled
001 (16 bits)	Invalid 15 0 Invalid 15 0 Invalid 15 0 Invalid 15 0	15 0 Invalid 15 0 Invalid 15 0 Invalid 15 0 Invalid	Setting is disabled
010 ~ 100 18-bit: X=17 20-bit: X=19 22-bit: X=21 24-bit: X=23	Invalid X 0 Invalid X 0 Invalid X 0 Invalid X 0	X 0 Invalid X 0 Invalid X 0 Invalid X 0 Invalid	Invalid X 0 Invalid X 0 Invalid X 0 Invalid X 0
110 (32 bits)	31 0 31 0 31 0 31 0	31 0 31 0 31 0 31 0	Setting is disabled
111 (Setting is disabled)			

6) SDTA bit (serial data alignment selection)

The SDTA bit selects how the serial data and padding bits are aligned. For communications without padding bits this bit is invalid.

Figure 18-9 Alignment of serial data and padding bits



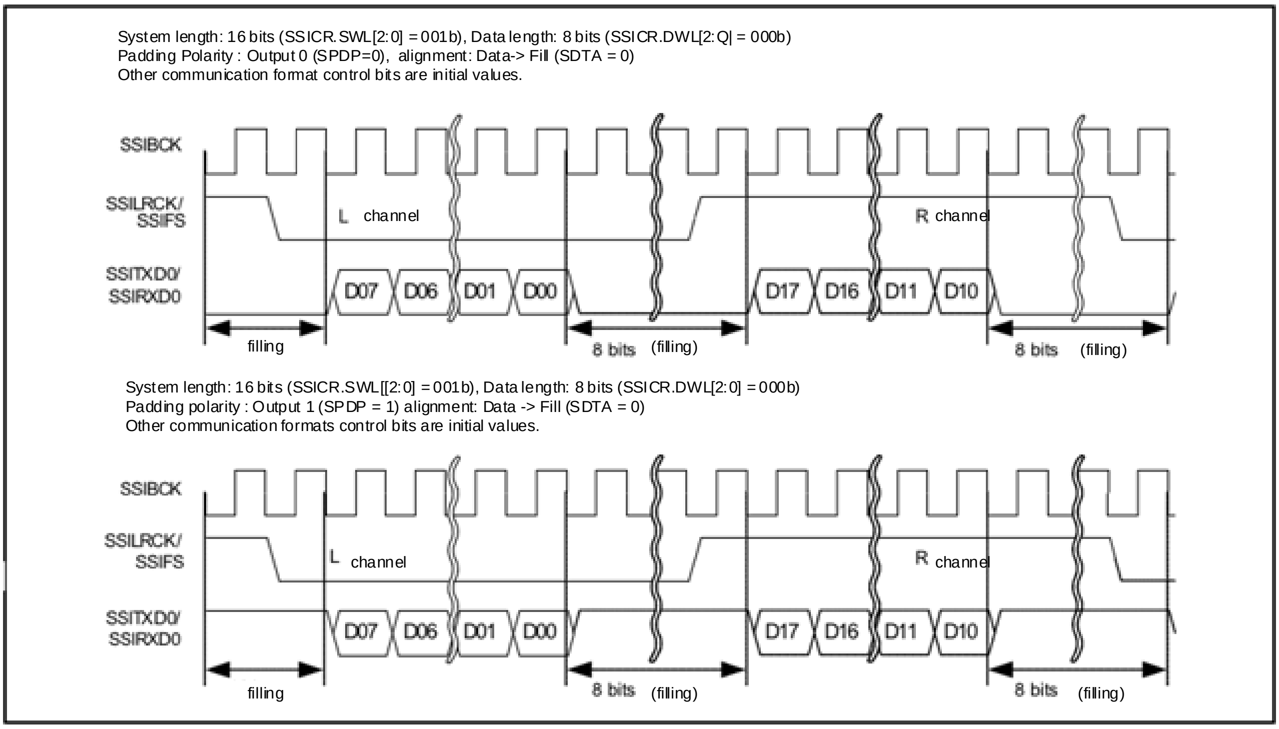
System length: 16 bits (SSICR.SWL[2:0] = 001b), Data length: 8 bits (SSICR.DWL[2:0] = 0)
Data alignment: data -> fill (SDTA = 0)
Other communication format control bits are initial values.

System length: 16 bits (SSICR.SWL[2:0] = 001b), Data length: 8 bits (SSICR.DWL[2:0] = 000b).
Data alignment: Fill -> data (SDTA = 1)
Other communication format control bits are initial values.

7) SPDP bit (selection of padding polarity)

The SPDP bit is used to select the polarity of the padding bit.

Figure 18-10 Padding bit polarity



System length: 16 bits (SSICR.SWL[2:0] = 001b), Data length: 8 bits (SSICR.DWL[2:Q| = 000b)
Padding Polarity : Output 0 (SPDP=0), alignment: Data-> Fill (SDTA = 0)
Other communication format control bits are initial values.

System length: 16 bits (SSICR.SWL[[2:0] = 001b), Data length: 8 bits (SSICR.DWL[2:0] = 000b)
Padding polarity : Output 1 (SPDP = 1) alignment: Data -> Fill (SDTA = 0)
Other communication formats control bits are initial values.

8) LRCKP bit (initial value and polarity selection of LR clock/frame synchronization signal)

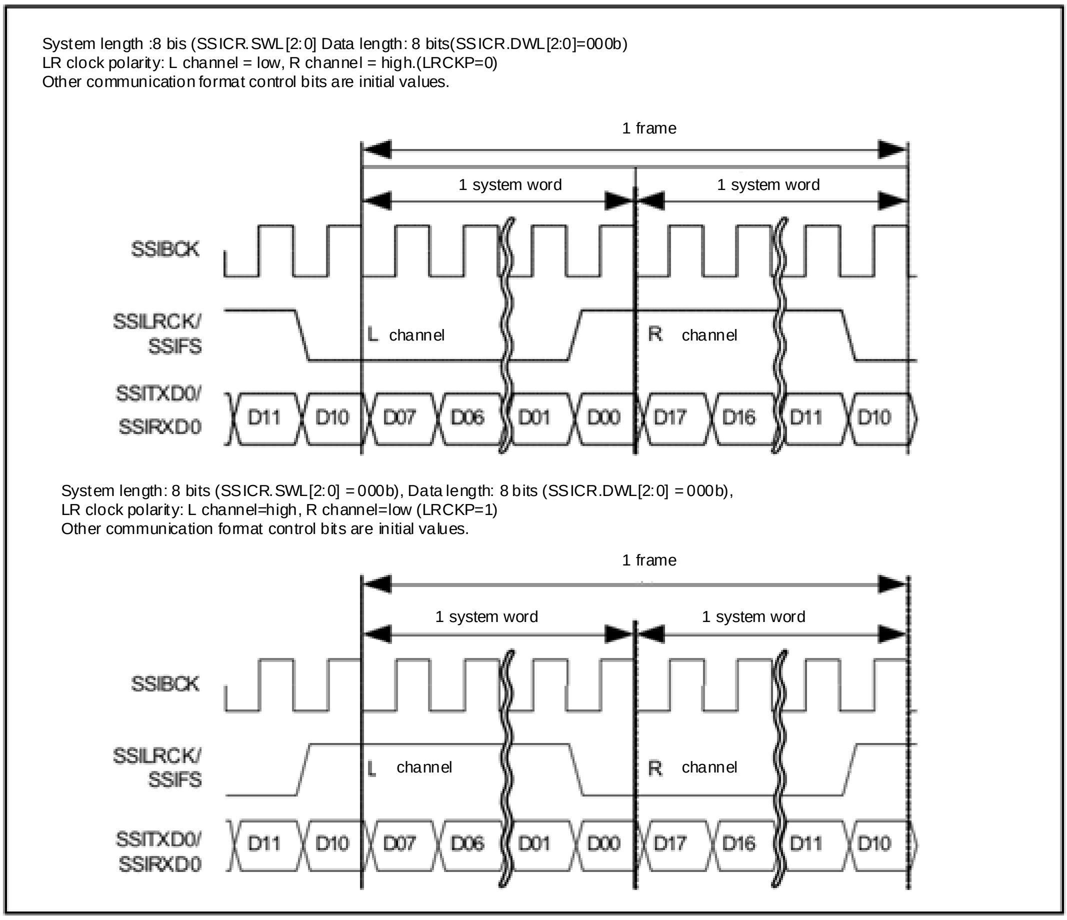
The LRCKP bit is used to select the initial value and polarity of SSILRCK/SSIFS. Set this bit according to the communication format to be used in SSIE. In slave mode communication (MST=0), it is used to trigger the start only. Writes are only available when the clock on the SSILRCK/SSIFS pin is stopped.

Table 18-3 Initial output value and polarity of SSILRCK/SSIFS pins

Communication format	Initial state expectation value	Setting value of LRCKP
I^{2}S	High	0
Mono	Low	1

Note: When the format to be used is compatible with I2S and mono formats, configure the appropriate settings to enable communication.

Figure 18-11 LR clock/frame synchronization polarity setting


System length :8 bis (SSICR.SWL[2:0] Data length: 8 bits(SSICR.DWL[2:0]=000b)
LR clock polarity: L channel = low, R channel = high.(LRCKP=0)
Other communication format control bits are initial values.
1 frame
1 system word
1 system word
SSIBCK
SSILRCK/
SSIFS
L channel
R channel
SSITXD0/
SSIRXD0
D11
D10
D07
D06
D01
D00
D17
D16
D11
D10
System length: 8 bits (SSICR.SWL[2:0] = 000b), Data length: 8 bits (SSICR.DWL[2:0] = 000b),
LR clock polarity: L channel=high, R channel=low (LRCKP=1)
Other communication format control bits are initial values.
1 frame
1 system word
1 system word
SSIBCK
SSILRCK/
SSIFS
L channel
R channel
SSITXD0/
SSIRXD0
D11
D10
D07
D06
D01
D00
D17
D16
D11
D10

9) BCKP bit (Bit clock polarity selection bit)

The BCKP bit is used to select the bit clock polarity. This bit is written when the audio clock AUDIO_MCK is stopped.

Table 18-4 Bit clock polarity

Communication	Master/Slave	Time sequence	BCKP = 0	BCKP = 1
Receive	Slave	SSILRCK/SSIFS sampling	SSIBCK Rising edge	SSIBCK falling edge
	Master/Slave	SSIRXD0 Sampling	SSIBCK Rising edge	SSIBCK falling edge
Transmit	Master	Changing SSILRCK/SSIFS output	SSIBCK falling edge	SSIBCK Rising edge
	Master/Slave	Changing SSITXD0 output	SSIBCK falling edge	SSIBCK Rising edge

10) MST bit (master mode enable)

The MST bit is used to select whether the communication mode is master or slave. This bit is written when the audio clock AUDIO_MCK is stopped.

11) SWL[2:0] bits (system word length selection)

SWL[2:0] is used to select how many bits are in a system word. It can be rewritten when the LR clock pin SSILRCK/SSIFS is stopped.

12) DWL[2:0] bits (data word length selection)

DWL[2:0] is used to select how many bits are in a data word. The data word length cannot exceed the system word length.

13) IIEN bit (Idle mode interrupt output enable)

The IIEN bit is used to enable or disable the output of an idle mode interrupt. By setting it to 1, an interrupt is output on the rising edge of SSISR.IIRQ=1. An interrupt is also output when the IIEN bit changes from 0 to 1 when SSISR.IIRQ=1.

14) ROIEN bit (Receive overflow interrupt output enable)

The ROIEN bit is used to enable or disable the output of the receive overflow interrupt. By setting it to 1, an interrupt is output on the rising edge when SSISR.ROIRQ = 1. An interrupt is also output when SSISR.ROIRQ = 1 and the ROIEN bit changes from 0 to 1.

15) RUIEN bit (Receive underflow interrupt output enable)

The RUIEN bit is used to enable or disable the output of the receive underflow interrupt. By setting it to 1, an interrupt is output on the rising edge when SSISR.RUIRQ = 1. An interrupt is also output when SSISR.RUIRQ = 1 and the RUIEN bit changes from 0 to 1.

16) TOIEN bit (transmit overflow interrupt output enable)

The TOIEN bit is used to enable or disable the output of the transmit overflow interrupt. By setting it to 1, an interrupt is output on the rising edge when SSISR.TOIRQ = 1. With SSISR.TOIRQ = 1, an interrupt is also output when the TOIEN bit changes from 0 to 1.

17) TUIEN bit (transmit underflow interrupt output enable)

The TUIEN bit is used to enable or disable the output of the transmit underflow interrupt. By setting it to 1, an interrupt is output on the rising edge when SSISR.TUIRQ = 1. An interrupt is also output when SSISR.TUIRQ = 1 and the TUIEN bit changes from 0 to 1.

18) CKS bit (selection of audio clock in master communication mode)

The CKS bit is used to set the audio clock in master communication mode (MST=1). This bit setting is disabled in slave mode communication (MST=0). This bit is written when the audio clock AUDIO_MCK is stopped.

18.3.2 Status register (SSISR)

This register is a flag register to indicate the operation status of SSIE.

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	TUIRQ	TOIRQ	RUIRQ	ROIRQ	IIRQ	—	—	—	—	—	—	—	—	—
Reset value:	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

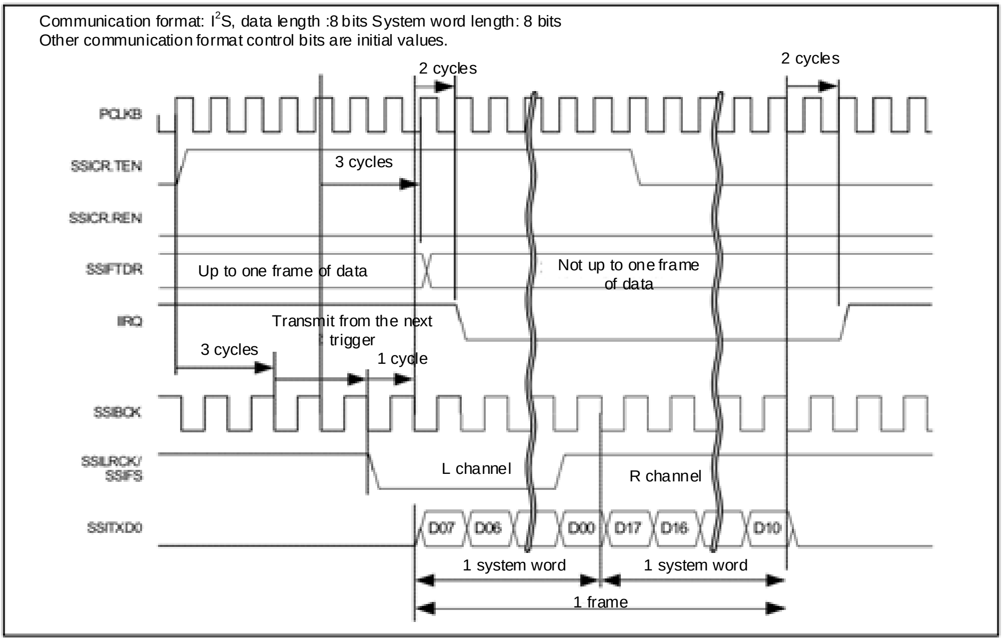
	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit Name	Description	R/W
b24 ~ b0	-	Reserved	Can only write 0 and read 0	-
b25	IIRQ	Idle mode status flag	0: In the communication status 1: In the idle state	R
b26	ROIRQ	Receive overflow error status flag	0: No receive overflow error generated 1: A receive overflow error generated	R/W
b27	RUIRQ	Receive underflow error status flag	0: No receive underflow error generated 1: A receive underflow error generated	R/W
b28	TOIRQ	Transmit overflow error status flag	0: No transmit overflow error generated 1: A transmit overflow error generated	R/W
b29	TUIRQ	Transmit underflow error status flag	0: No transmit down overflow error generated 1: A transmit down overflow error generated	R/W
b31，b30	-	Reserved	Can only write 0 and read 0	-

1) IIRQ bit (Idle mode status flag)

IIRQ is used to indicate whether the SSIE is in the idle state or the communication state.

Figure 18-12 IIRQ setup timing (transmit)



Communication format: I^{2}S, data length :8 bits System word length: 8 bits
Other communication format control bits are initial values.

Setting and clearing of IIRQ when transmitting only:

[Clear conditions for IIRQ]

- When transmission is enabled (SSICR.TEN = 1 , SSICR.REN = 0), the transmission data of the transmit frame is written to the SSIFTDR register and triggered by the SSILRCK/SSIFS signal to initiate transmission.

[Clear timing of IIRQ]

- After generating the trigger start, a period of 1 SSIBCK cycle + 2 PCLKB cycles elapses, which is the clear condition of IIRQ.

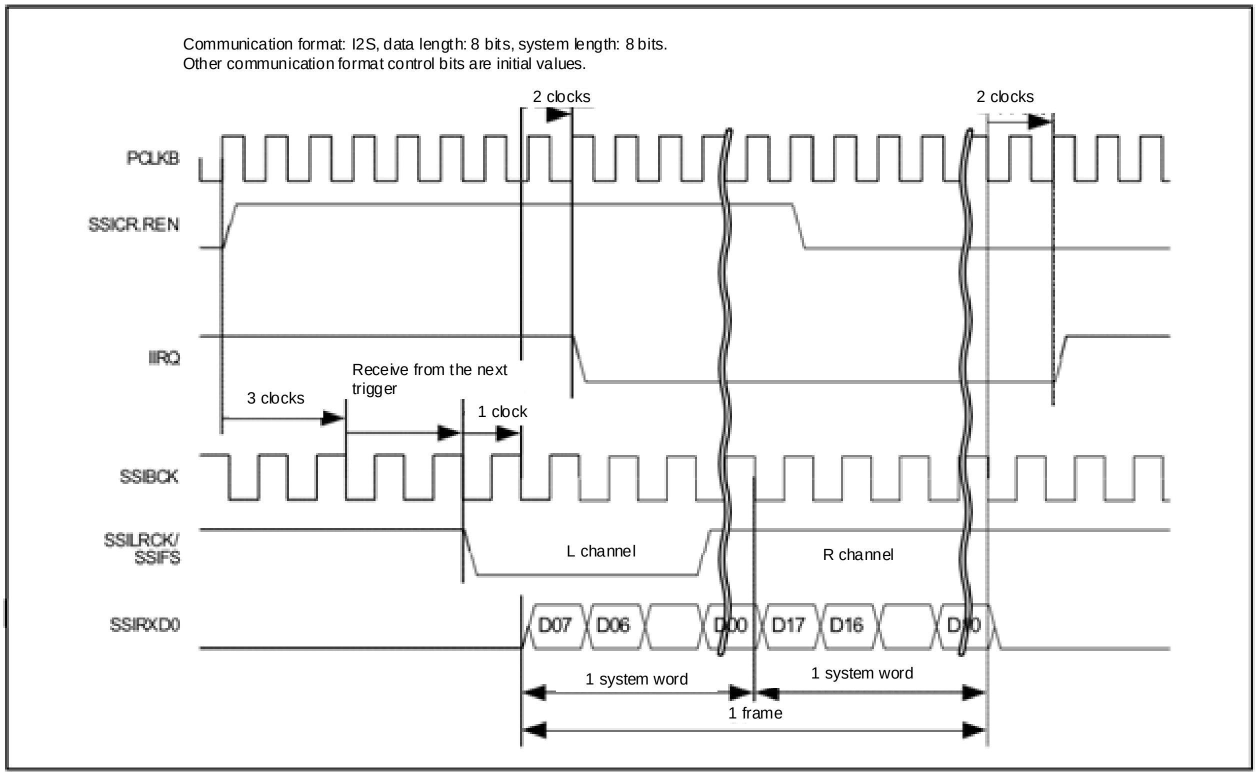
[Set conditions for IIRQ]

- Stop transmitting and receiving (SSICR.TEN = 0, SSICR.REN = 0), after completing the transmission of a frame.

[Clear timing of IIRQ]

- 2 PCLKB cycles after the end of the transmission (at the frame boundary), is the set condition.

Figure 18-13 IIRQ setup timing (receive)


Communication format: I2S, data length: 8 bits, system length: 8 bits.
Other communication format control bits are initial values.
2 clocks
2 clocks
PCLKB
SSICR.REN
IIRQ
Receive from the next trigger
3 clocks
1 clock
SSIBCK
SSILRCK/ SSIFS
L channel
R channel
SSIRXD0
D07
D06
D00
D17
D16
D10
1 system word
1 system word
1 frame

Setting and clearing of IIRQ on receive only:

[Clear conditions for IIRQ]

- When reception is enabled (SSICR.TEN = 0, SSICR.REN = 1), reception is triggered by the SSILRCK/SSIFS signal.

[Clear timing of IIRQ]

- After 1 SSIBCK cycle + 2 PCLKB cycles and generating a trigger start, which is the clear condition of IIRQ.

[Set condition of IIRQ]

- After stopping transmitting and receiving (SSICR.TEN = 0, SSICR.REN = 0) and after completing the reception of a frame.

[Set timing of IIRQ]

- 2 PCLKB cycles after the end of reception (at the frame boundary) is the set condition.

Simultaneously allow the IIRQ to be set and cleared when sending and receiving:

[Clear conditions for IIRQ]

- When transmit and receive are enabled (SSICR.TEN = 1, SSICR.REN = 1), the transmission data of the transmit frame is written to the SSIFTDR register and triggered by the SSILRCK/SSIFS signal to initiate transmission.

[Clear timing of IIRQ]

- After generating the trigger start, a period of 1 SSIBCK cycle + 2 PCLKB cycles elapses, which is the clear condition of IIRQ.

[Set conditions for IIRQ]

- After stopping transmitting and receiving, (SSICR.TEN = 0 , SSICR.REN = 0), after completing one frame of data transmission.

[Set timing of IIRQ]

- 2 PCLKB cycles after the end of the transmission (at the frame boundary) is the set condition.

2) ROIRQ bit (Receive overflow error status flag)

ROIRQ is the status flag for receive overflow errors. This flag bit is set automatically, but must be cleared by accessing the register. This flag indicates that the received data is out of request. When a receive overflow error is generated, data is not transferred from the receive shift register to SSIFRDR. A receive FIFO data register reset does not clear this flag (SSIFCR.RFRST).

[Priority order of setting and clearing]

Set first.*1

[Clear conditions]

One of the following actions will clear the flag bit:

1. Soft reset occurs (SSIFCR.SSIRST = 1)
2. Read 1 from this bit and write 0 to this bit.
3. Enable communication (Set SSICR.REN from 0 to 1).

[Clear timing]

Clear timing associated with the above clear conditions:

1. Read 1 from this bit and write 0 to this bit
2. 1 PCLKB cycle after SSICR.REN changes from 0 to 1. *2

Note 1. This bit is cleared by a soft reset (SSIFCR.SSIRST=1). Soft reset takes precedence over all the above-mentioned set and clear conditions.

Note 2. After communication enable (by changing the SSICR.REN bit from 0 to 1), the receive error flags RUIRQ and ROIRQ in the SSIR register are cleared. However, if the SSIR register is read immediately, the clear status of the receive error flags may not be read.

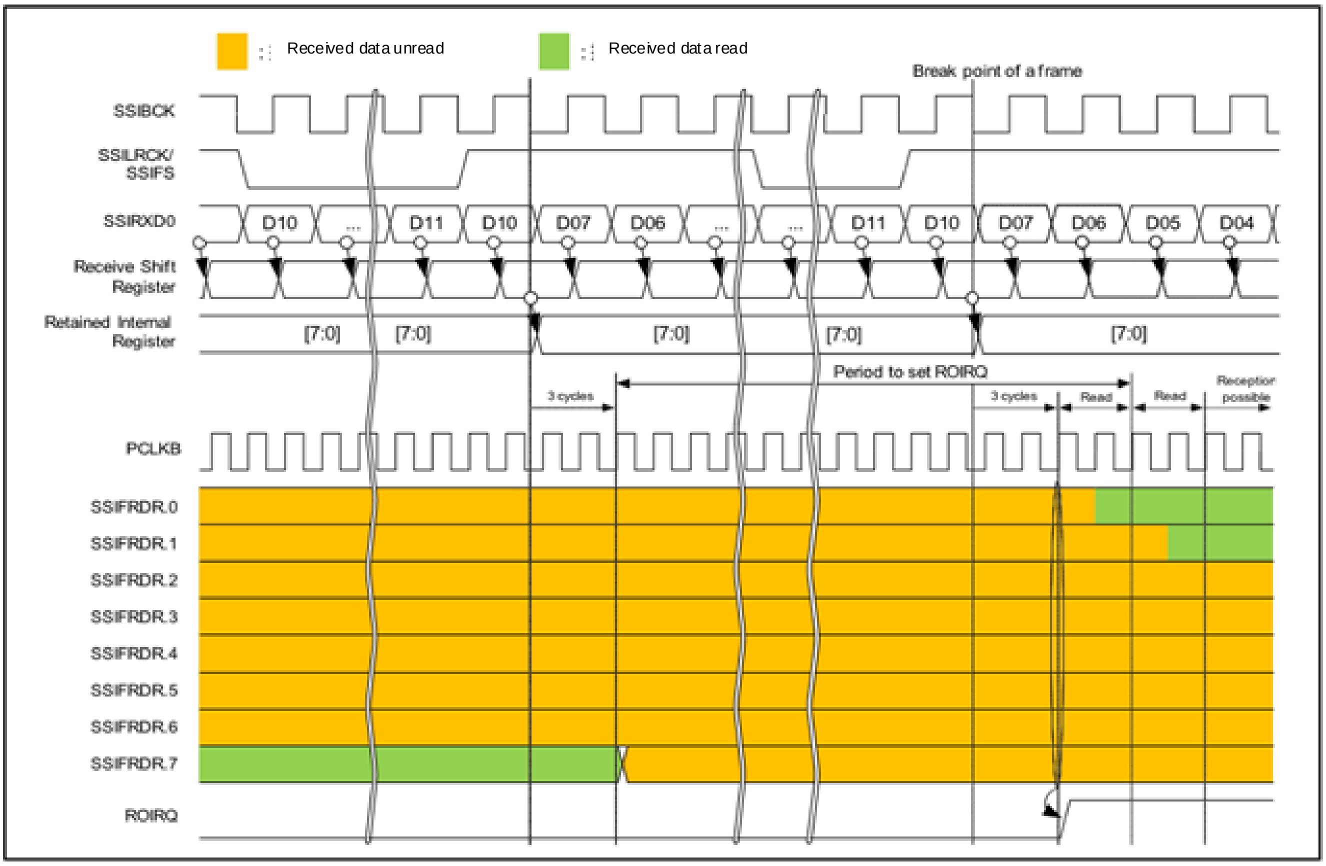
[Set conditions]

- When SSIFRDR is full and new data is received at the same time.

[Set timing]

- PCLKB cycles after reception completion.

Figure 18-14 ROIRQ set timing



3) RUIRQ bit (receive underflow error status flag)

RUIRQ is a status flag that indicates a receive underflow error. This flag is set automatically, but must be cleared by accessing the register. This flag indicates that SSIFRDR is read when it is empty. Data read from SSIFRDR is invalid when a receive underflow error is generated. A receive FIFO data register reset does not clear this flag (SSIFCR.RFRST). However, if the receive FIFO data register is reset (by setting SSIFCR.RFRST to 1) and the SSIFRDR register is read, this flag bit will not be set up.

[Priority order of setting and clearing]

Set first.*[1]

[Clear conditions]

One of the following actions will clear the flag bit:

1. Soft reset occurs (SSIFCR.SSIRST = 1)
2. Read 1 from this bit and then write 0 to this bit.
3. Enable communication (Set SSICR.REN from 0 to 1).

[Clear timing]

Clear timings associated with the above clearance conditions:

1. Read 1 from this bit and write 0 to this bit
2. 1 PCLKB cycle after SSICR.REN changes from 0 to 1.*[2]

Note 1. This bit is cleared by a soft reset (SSIFCR.SSIRST=1). Soft reset takes precedence over all the above-mentioned set and clear conditions.

Note 2. After communication enable (by changing the SSICR.REN bit from 0 to 1), the receive error flags RUIRQ and

ROIRQ in the SSIR register are cleared. however, if the SSIR register is read immediately, the clear status of the receive error flags may not be read.

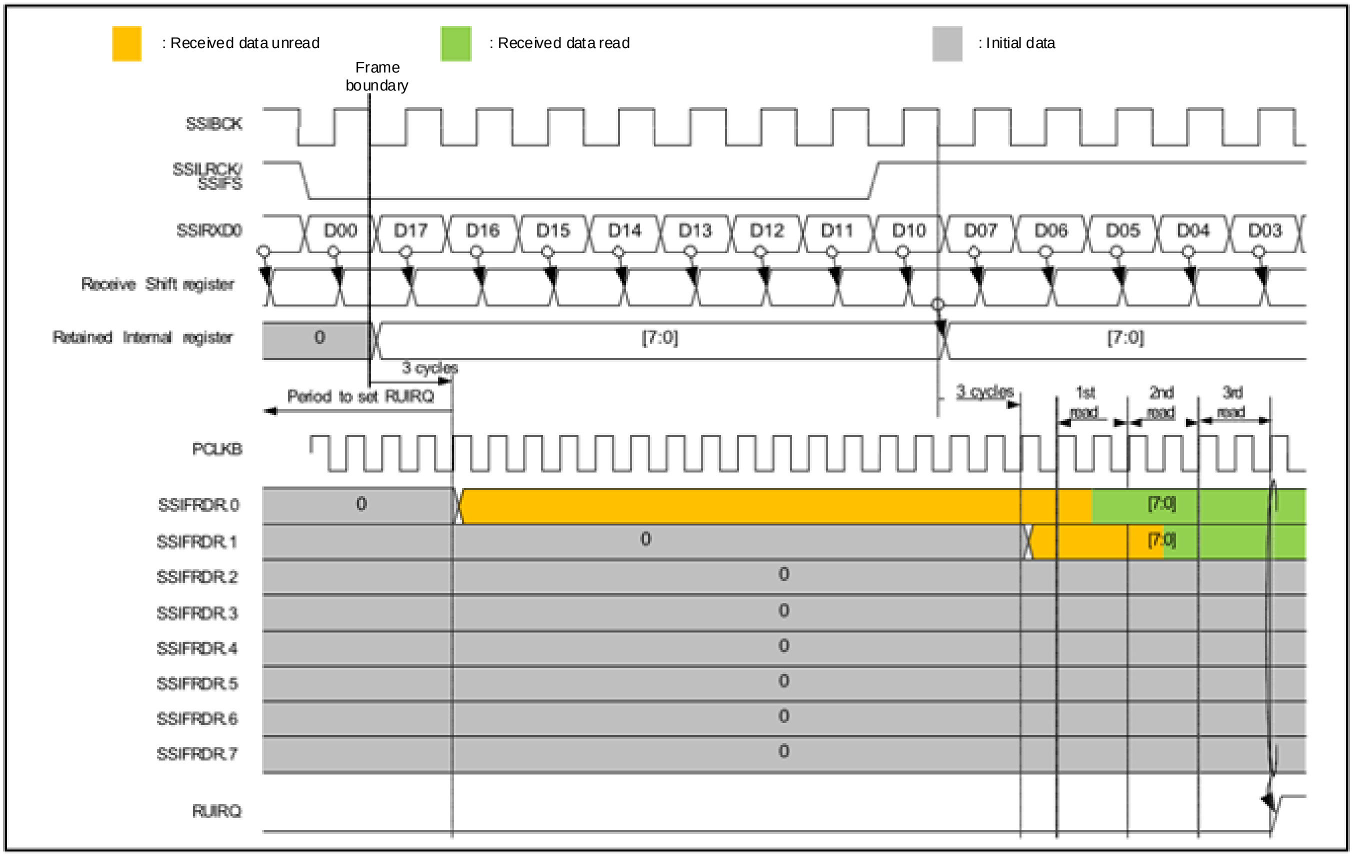
[Set conditions]

- This register is read when SSIFRDR is empty.

[Set timing]

- After reading FRSSISDR.

Figure 18-15 RUIRQ reset timing



4) TOIRQ bit (transmit overflow error status flag)

TOIRQ is the status flag for transmission overflow errors. This flag is set automatically, but must be cleared by accessing the register. This flag indicates an attempt to write new data to the SSIFTDR register when the register is full. The write data action is ignored at this point. This flag is not cleared by resetting the transmit FIFO data register (via SSIFCR.TFRST).

[Priority order of setting and clearing]

Set first.*1

[Clear condition]

One of the following actions will clear the flag bit:

1. Soft reset occurs (SSIFCR.SSIRST = 1)
2. Read 1 from this bit and then write 0 to this bit.
3. Enable communication (Set SSICR.TEN from 0 to 1).

[Clear timing]

Clear timings associated with the above clearance conditions:

1. Read 1 from this bit and write 0 to this bit

2. 1 PCLKB cycle after SSICR.TEN changes from 0 to 1. *2

Note 1. This bit is cleared by a soft reset (SSIFCR.SSIRST=1). Soft reset takes precedence over all the above-mentioned set and clear conditions.

Note 2. After communication enable (by changing the SSICR.TEN bit from 0 to 1), the receive error flags TUIRQ and TOIRQ in the SSIR register are cleared. however, if the SSIR register is read immediately, the clear status of the receive error flags may not be read.

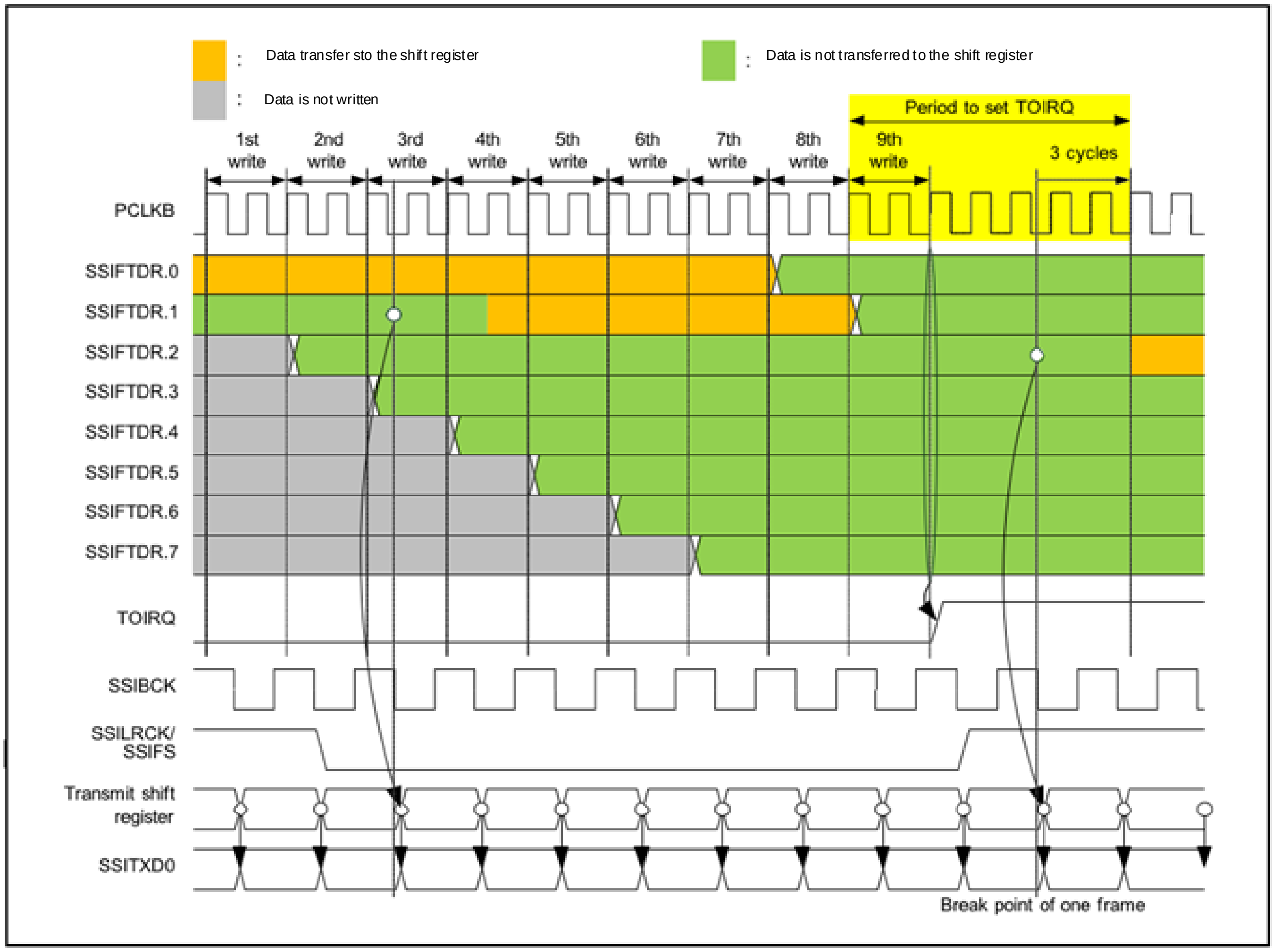
[Set conditions]

- Attempt to write new data when the SSIFTDR register is full.

[Set timing]

- After writing SSIFTDR register.

Figure 18-16 TOIRQ set timing



5) TUIRQ bit (Transmit underflow error status flag)

TUIRQ is the status flag for transmit underflow errors. This flag is set automatically and must be cleared by accessing the register. This flag indicates that the serial data required to write the frame to SSIFTDR was late and did not catch the transmission of the frame. Even if this flag is cleared after it is set, the SSITXD0 output remains at 0. To output data written to the Transmit FIFO Data Register (SSIFTDR) to the SSITXD0 pin, follow the communication stop procedure in Figure 18.52 and the error handling procedure in Figure 18.53. Refer to Section

18.7.6, "Error Handling" for the procedure to recover from an error. This flag will not be cleared by resetting the transmit FIFO data register (via SSIFCR.TFRST).

[Priority of set and clear]

Set first.*1

[Clear condition]

The flag bit can be cleared by one of the following actions:

1. Soft reset occurs (SSIFCR.SSIRST = 1)
2. Read 1 from this bit and then write 0 to this bit.
3. Enable communication (Set SSICR.TEN from 0 to 1).

[Clear Timing]

Clear timings associated with the above clearance conditions:

1. Read 1 from this bit and write 0 to this bit
2. 1 PCLKB cycle after SSICR.TEN changes from 0 to 1.*2

Note 1. This bit is cleared by a soft reset (SSIFCR.SSIRST=1). Soft reset takes precedence over all the above-mentioned set and clear conditions.

Note 2. After communication enable (by changing the SSICR.TEN bit from 0 to 1), the receive error flags TUIRQ and TOIRQ in the SSIR register are cleared. However, if the SSIR register is read immediately, the clear status of the receive error flags may not be read.

[Set conditions]

- When the continuous transmission has exceeded the frame boundary, the required next frame transmission data has not yet been written to the SSIFTDR register.

[Reset Timing]

- 3 PCLKB cycles after frame boundary is exceeded

Figure 18-17 TUIRQ clear timing

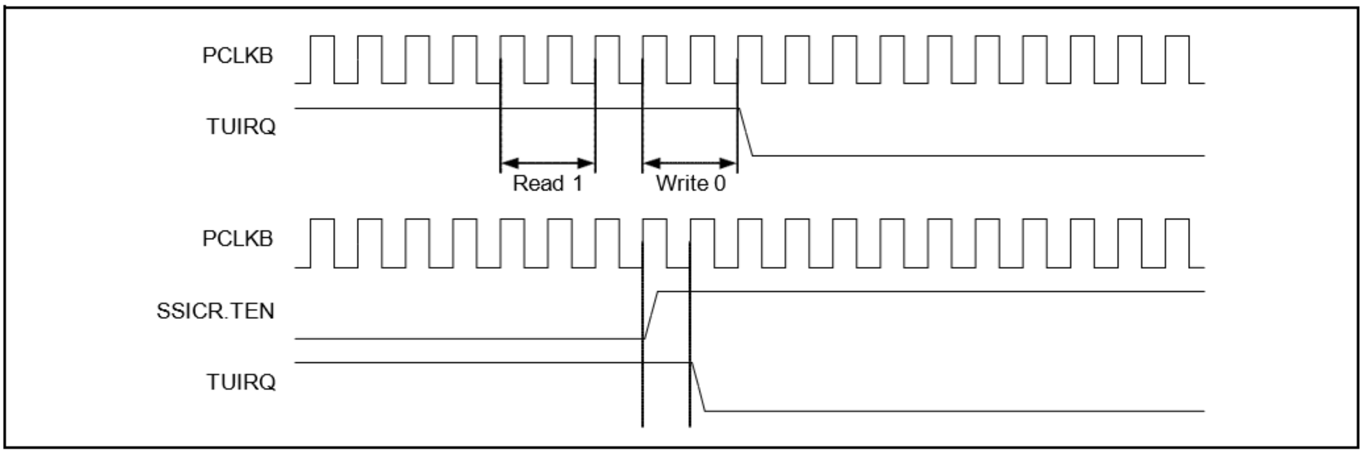


Figure 18-18 Timing of TUIRQ setting during continuous communication

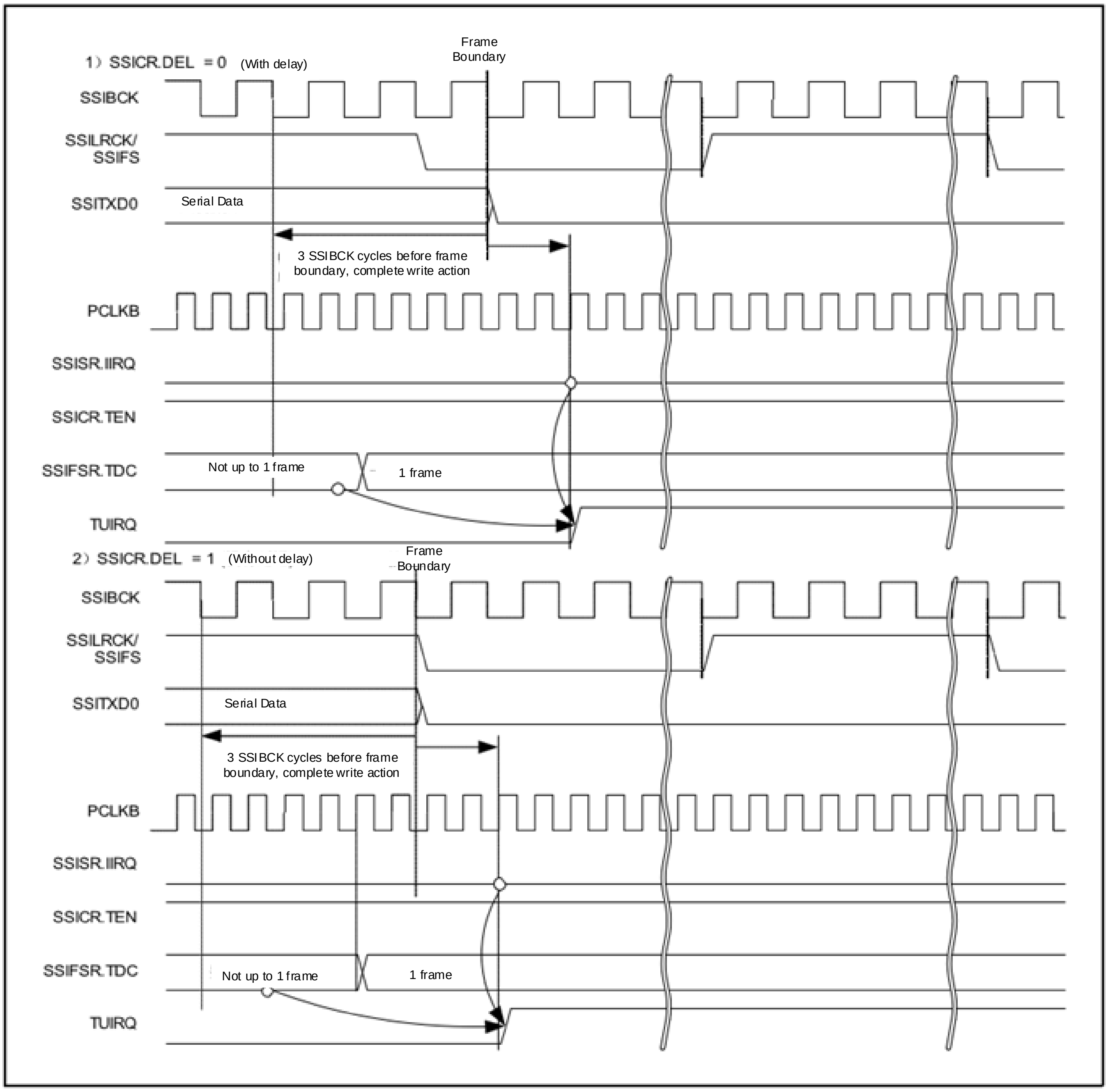
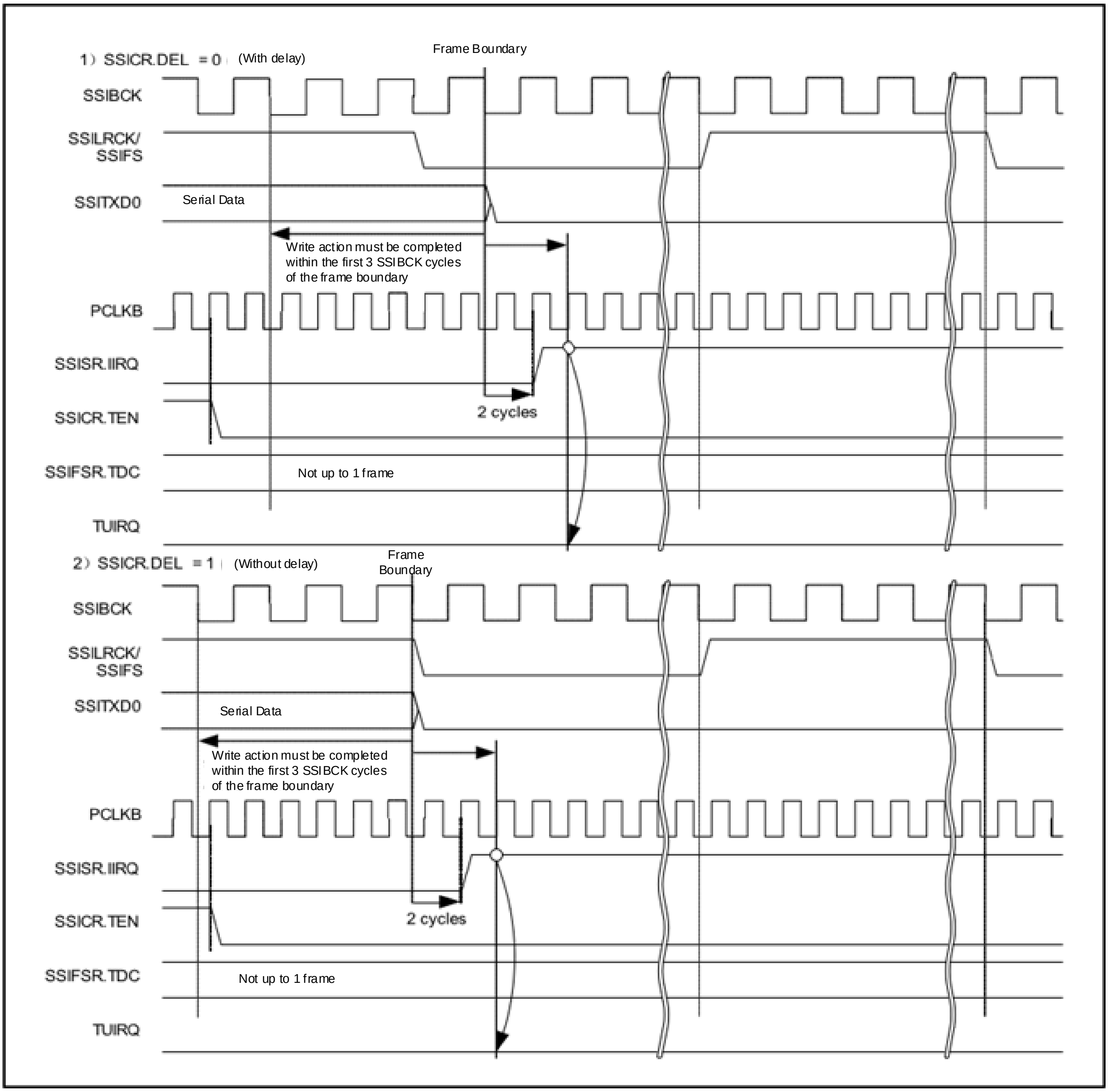


Figure 18-19 Timing of TUIRQ setting when communication is stopped



18.3.3 FIFO control register (SSIFCR)

This register is used to set software reset, byte swap and interrupt request enable or disable.

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	AUCKE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	SSIRST
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	BSW	—	—	—	—	—	—	—	TIE	RIE	TFRST	RFRST
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	RFRST	Reset receive FIFO data register*1	0: Clear reset of receive FIFO data 1: Reset the receive FIFO data	R/W
b1	TFRST	Reset transmit FIFO data register*1	0: Clear reset of receive FIFO data 1: Reset the receive FIFO data	R/W
b2	RIE	Allow output of receive data full interrupt	0: Do not allow receive data full interrupt 1: Receive data full interrupt allowed	R/W
b3	TIE	Allow output of transmit data empty interrupt	"0: Do not allow to transmit data null interrupt 1: Allow transmit data null interrupt"	R/W
b10~b4	-	Reserved	Can only write 0 and read 0	-
b11	BSW	Allow byte swap*1	0: Byte swapping is not allowed 1: Byte swapping is allowed	R/W
b15~b12	-	Reserved	Can only write 0 and read 0	-
b16	SSIRST	Software reset	0: Clear software reset 1: Set software reset	R/W
b30~b17	-	Reserved	Can only write 0 and read 0	-
b31	AUCKE	Allow AUDIO_MCK*1 supply in master communication mode	0: AUDIO_MCK is not allowed to be provided 1: Allow to provide AUDIO_MCK	R/W

Note 1. When SSIE is in communication state (SSISR.IIRQ = 0), it is forbidden to change these bits, otherwise the subsequent operations are unpredictable.

1) RFRST bit (Reset of receive FIFO data register)

The RFRST bit is used to set the software reset of the Receive FIFO Data Register (SSIFRDR). Writing a 1 to this bit will initialize the internal state associated with the SSIFRDR register. This bit is not automatically cleared after it is set up, and the reset can be cleared by writing a 0 to this bit. After writing a 0 to this bit, make sure to check that the bit is 0 before starting the next step.

This bit is controlled by the software reset caused by SSIRST. Since the software reset generated by SSIRST set-up takes precedence over the reset generated by the RFRST bit, the RFRST set-up is ignored when the SSIRST bit is set-up.

Table 18-5 Bit reset by the soft reset generated by RFRST

Symbol	Address (BASE+)		+0								+1							
			31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSICR	00h	+0	—	CKS	TUIEN	TOIEN	RUIEN	ROIEN	IIEN	—	—	—	DWL[2:0]			SWL[2:0]		
		+2	—	MST	BCKP	LRCKP	SPDP	SDTA	PDTA	DEL	CKDV[3:0]				MUEN	—	TEN	REN
SSISR	04h	+0	—	—	TUIRQ	TOIRQ	RUIRQ	ROIRQ	IIRQ	—	—	—	—	—	—	—	—	—
		+2	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
SSIFCR	10h	+0	AUCKE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	SSIRST
		+2	—	—	—	—	BSW	—	—	—	—	—	—	—	TIE	RIE	TFRST	RFRST
SSIFSR	14h	+0	—	—	—	—	TDC[3:0]				—	—	—	—	—	—	—	TDE
		+2	—	—	—	—	RDC[3:0]				—	—	—	—	—	—	—	RDF
SSIFTDR	18h	+0	FTDR[31:16]															
		+2	FTDR[15:0]															
SSIFRDR	1ch	+0	FRDR[31:16]															
		+2	FRDR[15:0]															
SSITDMR	20h	+0	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
		+2	—	—	—	—	—	—	BCKASTP	LRCONT	—	—	—	—	—	—	OMOD[1:0]	
SSISCR	24h	+0	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
		+2	—	—	—	—	—	TDES[2:0]			—	—	—	—	—	RDFS[2:0]		

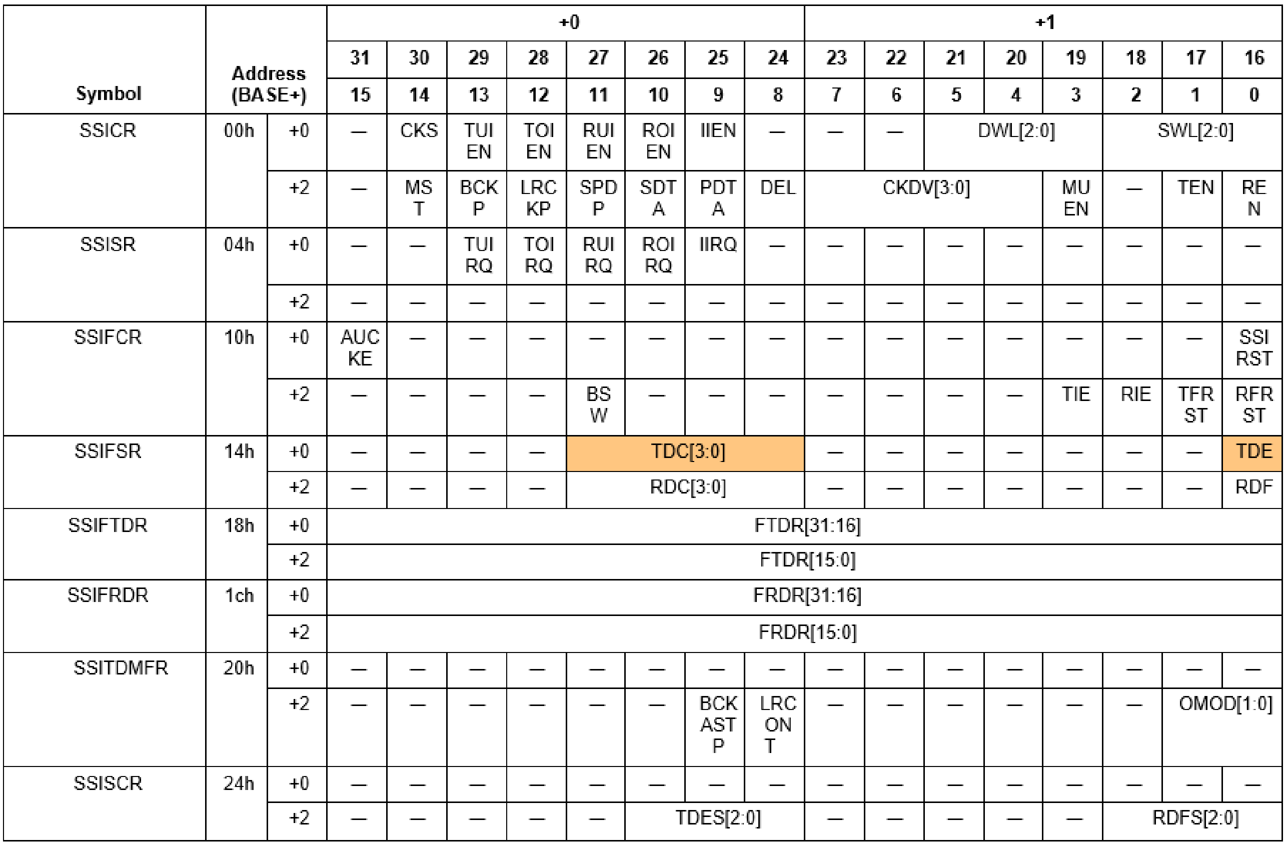
In the table above, the bits that fill the shadows can be reset by the RFRST software.

2) TFRST bit (reset of transmit FIFO data register)

The TFRST bit is used to set the software reset of the Transmit FIFO Data Register (SSIFTDR). Writing a 1 to this bit will initialize the internal state of the SSIFTDR register. This bit is not automatically cleared after it is set up, and the reset can be cleared by writing a 0 to this bit. After writing a 0 to this bit, make sure to check that the bit is 0 before starting the next step.

This bit is controlled for software resets caused by SSIRST. Since the software reset generated by the SSIRST setting takes precedence over the reset generated by the TFRST bit, the TFRST setting is ignored when the SSIRST bit is set.

Table 18-6 Bit reset by a soft reset generated by TFRST



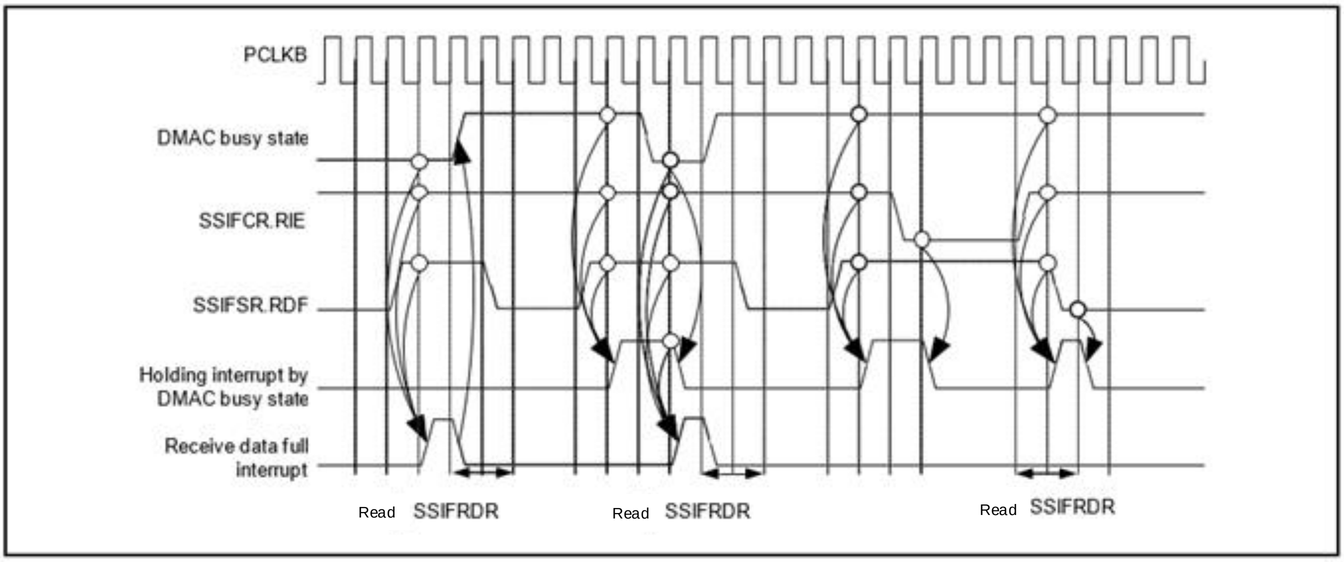
Symbol	Address (BASE+)		+0								+1							
			31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSICR	00h	+0	—	CKS	TUIEN	TOIEN	RUIEN	ROIEN	IIEN	—	—	—	DWL[2:0]			SWL[2:0]		
		+2	—	MST	BCKP	LRCKP	SPDP	SDTA	PDTA	DEL	CKDV[3:0]				MUEN	—	TEN	REN
SSISR	04h	+0	—	—	TUIRQ	TOIRQ	RUIRQ	ROIRQ	IIRQ	—	—	—	—	—	—	—	—	—
		+2	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
SSIFCR	10h	+0	AUCKE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	SSIRST
		+2	—	—	—	—	BSW	—	—	—	—	—	—	—	TIE	RIE	TFRST	RFRST
SSIFSR	14h	+0	—	—	—	—	TDC[3:0]				—	—	—	—	—	—	—	TDE
		+2	—	—	—	—	RDC[3:0]				—	—	—	—	—	—	—	RDF
SSIFTDR	18h	+0	FTDR[31:16]															
		+2	FTDR[15:0]															
SSIFRDR	1ch	+0	FRDR[31:16]															
		+2	FRDR[15:0]															
SSITDMFR	20h	+0	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
		+2	—	—	—	—	—	—	BCKASTP	LRCONT	—	—	—	—	—	—	OMOD[1:0]	
SSISCR	24h	+0	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
		+2	—	—	—	—	—	TDES[2:0]			—	—	—	—	—	RDFS[2:0]		

In the table above, the bits that fill the shadows can be reset by the TFRST software.

3) RIE bit (enable receive data full interrupt output)

The RIE bit is used to enable or disable the output of the receive data full interrupt. The receive data full interrupt triggers a read from the receive FIFO data register. Before setting this bit to "1", set the receive data full interrupt set condition via SSISCR.

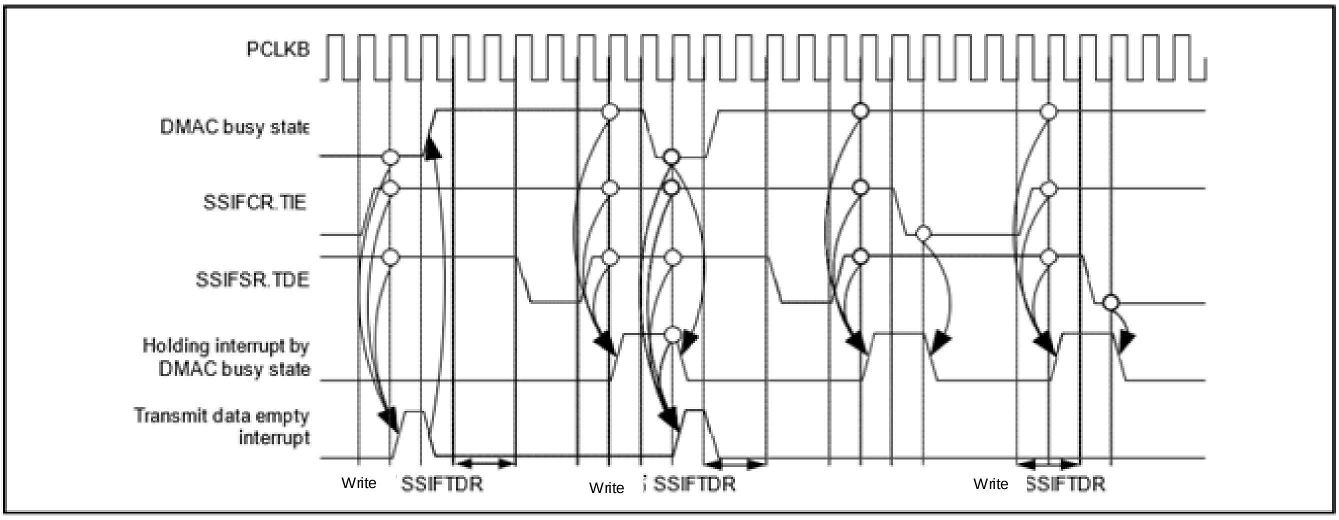
Figure 18-20 Timing of the receive data full interrupt



4) TIE bit (enable transmitting data with null interrupt output)

The TIE bit is used to enable or disable the output of the transmit data null interrupt. The transmit data null interrupt can trigger writing data to the transmit FIFO data register. Before setting this bit to "1", set the setting condition of the transmit data null interrupt via SSISCR.

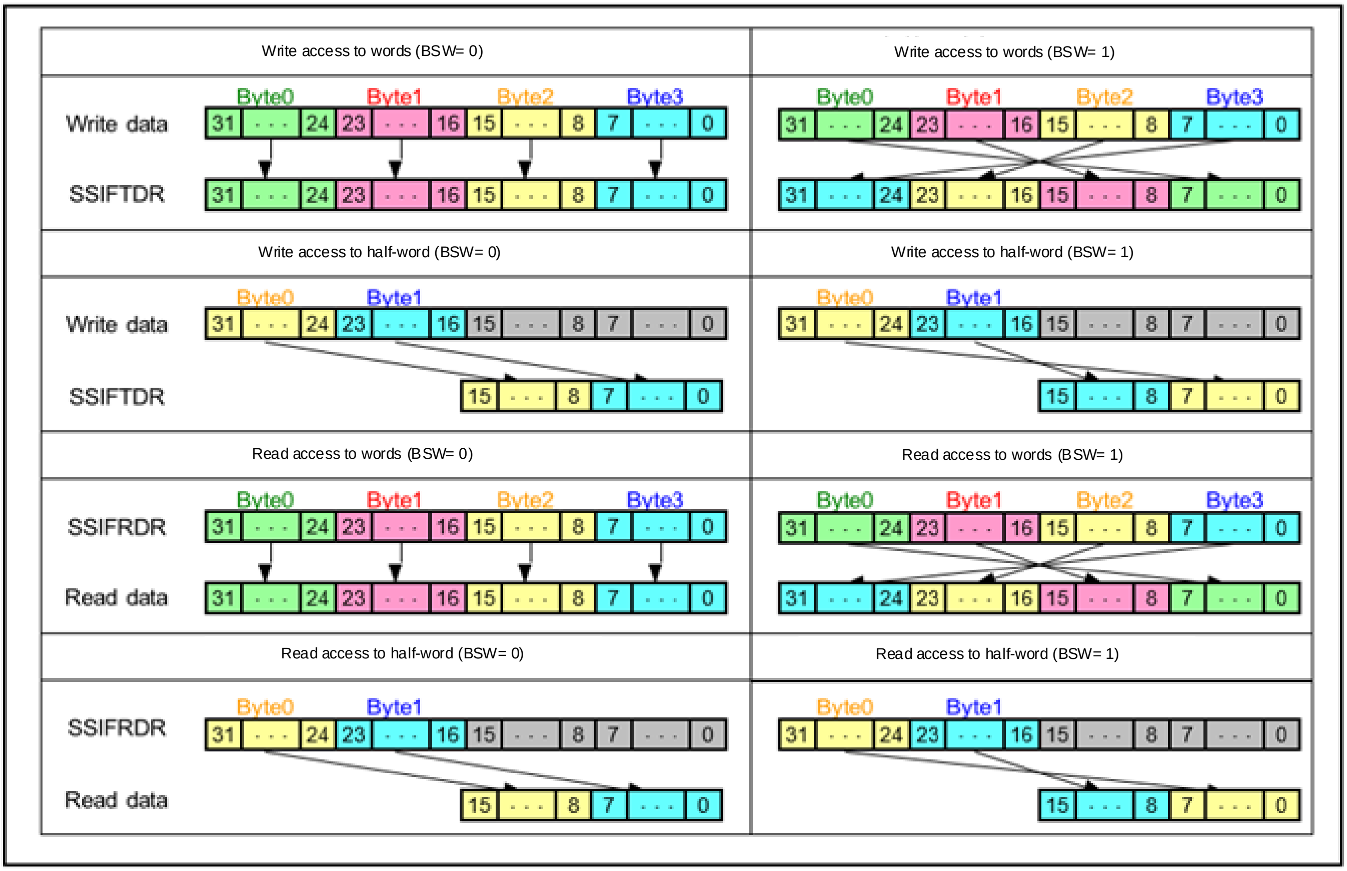
Figure 18-21 Timing of transmitting data null interrupt



5) BSW bit (byte swap enable)

The BSW bit is used to enable or disable byte swapping when accessing the Transmit FIFO Data Register (SSIFTDR) and the Receive FIFO Data Register (SSIFRDR). This bit is only valid for 16-bit access or 32-bit access to SSIFTDR and SSIFRDR.

Figure 18-22 Example of byte swap


Write access to words (BSW= 0)
Write access to words (BSW= 1)
Write data
SSIFTDR
Byte0
Byte1
Byte2
Byte3
31
24
23
16
15
8
7
0
Write access to half-word (BSW= 0)
Write access to half-word (BSW= 1)
Read access to words (BSW= 0)
Read access to words (BSW= 1)
SSIFRDR
Read data
Read access to half-word (BSW= 0)
Read access to half-word (BSW= 1)

6) SSIRST bit (software reset)

SSIRST is used to reset the SSIE software. Writing a 1 to this bit will initialize the internal state of the SSIE. This bit is not automatically cleared after it is set, and the reset can be cleared by writing 0 to this bit. After writing 0 to this bit, make sure to check that the bit is 0 before starting the next step.

To immediately stop SSIE communication, write 1 to this bit after turning off the peripheral functions and initialize with a software reset.

Table 18-7 Bits reset by soft reset generated by SSIRST

Symbol	Address (BASE+)		+0								+1							
			31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSICR	00h	+0	—	CKS	TUIEN	TOIEN	RUIEN	ROIEN	IIEN	—	—	—	DWL[2:0]			SWL[2:0]		
		+2	—	MST	BCKP	LRCKP	SPDP	SDTA	PDTA	DEL	CKDV[3:0]				MUEN	—	TEN	REN
SSISR	04h	+0	—	—	TUIRQ	TOIRQ	RUIRQ	ROIRQ	IIRQ	—	—	—	—	—	—	—	—	—
		+2	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
SSIFCR	10h	+0	AUCKE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	SSIRST
		+2	—	—	—	—	BSW	—	—	—	—	—	—	—	TIE	RIE	TFRST	RFRST
SSIFSR	14h	+0	—	—	—	—	TDC[3:0]				—	—	—	—	—	—	—	TDE
		+2	—	—	—	—	RDC[3:0]				—	—	—	—	—	—	—	RDF
SSIFTDR	18h	+0	FTDR[31:16]															
		+2	FTDR[15:0]															
SSIFRDR	1ch	+0	FRDR[31:16]															
		+2	FRDR[15:0]															
SSITDMR	20h	+0	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
		+2	—	—	—	—	—	—	BCKASTP	LRCONT	—	—	—	—	—	—	OMOD[1:0]	
SSISCR	24h	+0	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
		+2	—	—	—	—	—	TDES[2:0]			—	—	—	—	—	RDFS[2:0]		

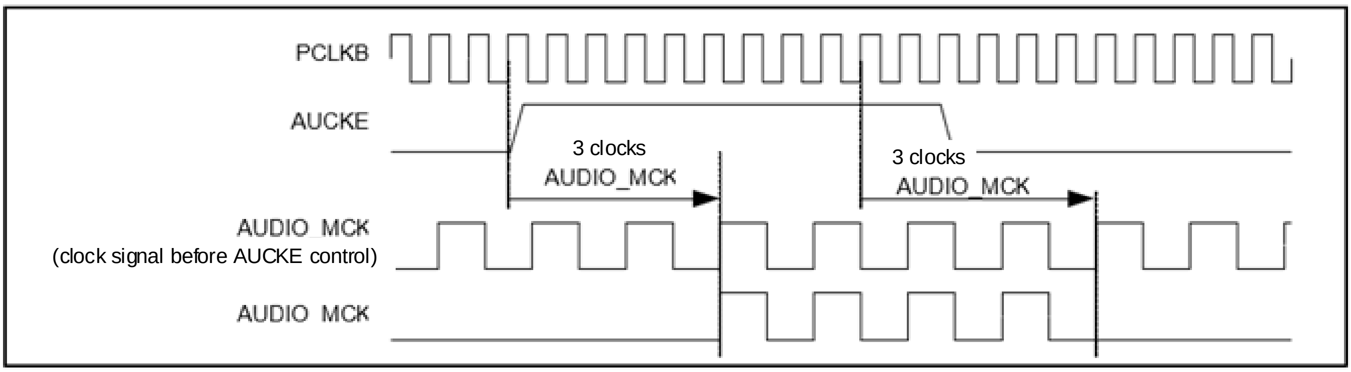
n the table above, the bits filled with shading can be reset by the SSIRST software.

7) AUCKE bit (AUDIO_MCK clock enable in master mode communication)

The AUCKE bit enables or disables the audio clock AUDIO_MCK in master mode communication (MST=1) .

This bit value can only be changed after the other AUDIO U MCK related settings (CKS, MST, BCKP and CKDV bits in the SSICR register) have been configured.

Figure 18-23 Stop/Start AUDIO_MCK



Note: When communicating in slave mode (SSICR.MST=0), SSIE requires SSIBCK from master to stop BCK. Please ensure that SSIE is idle (SSISR.IIRQ=1). If the BCK stops before the SSIE becomes idle, use the procedure shown in Figure 18.48 to initiate communication or use the procedure in Figure 18.54 to wait for the idle state to resume communication.

When communicating in master mode (SSICR.MST=1), SSIE operates with the audio clock (AUDIO_MCK). To stop SSIE completely, make sure SSIE is idle (SSISR.IIRQ=1) and then write 0 to SSIFCR.ADCKE. If you write 0 to SSIFCR.ADCKE before SSIE is idle, use the procedure shown in Figure 18.48 to start communication.

Figure 18-24 Timing diagram from module reset to start master communication mode

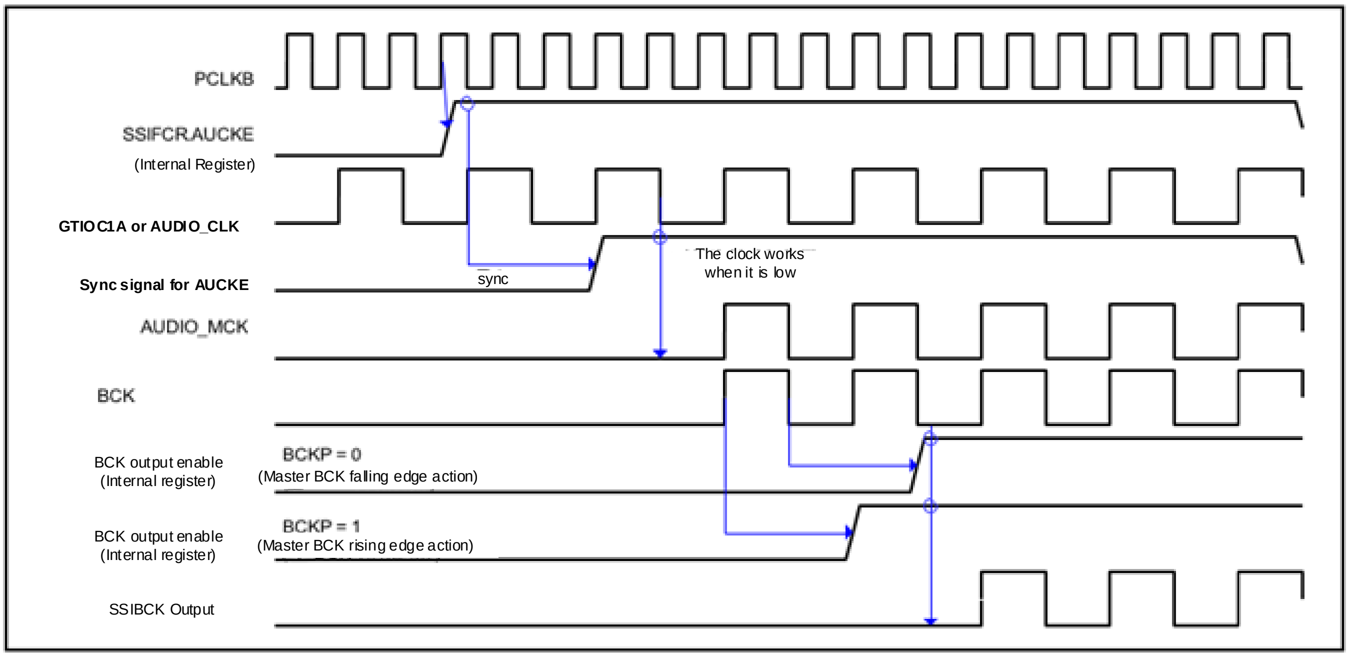
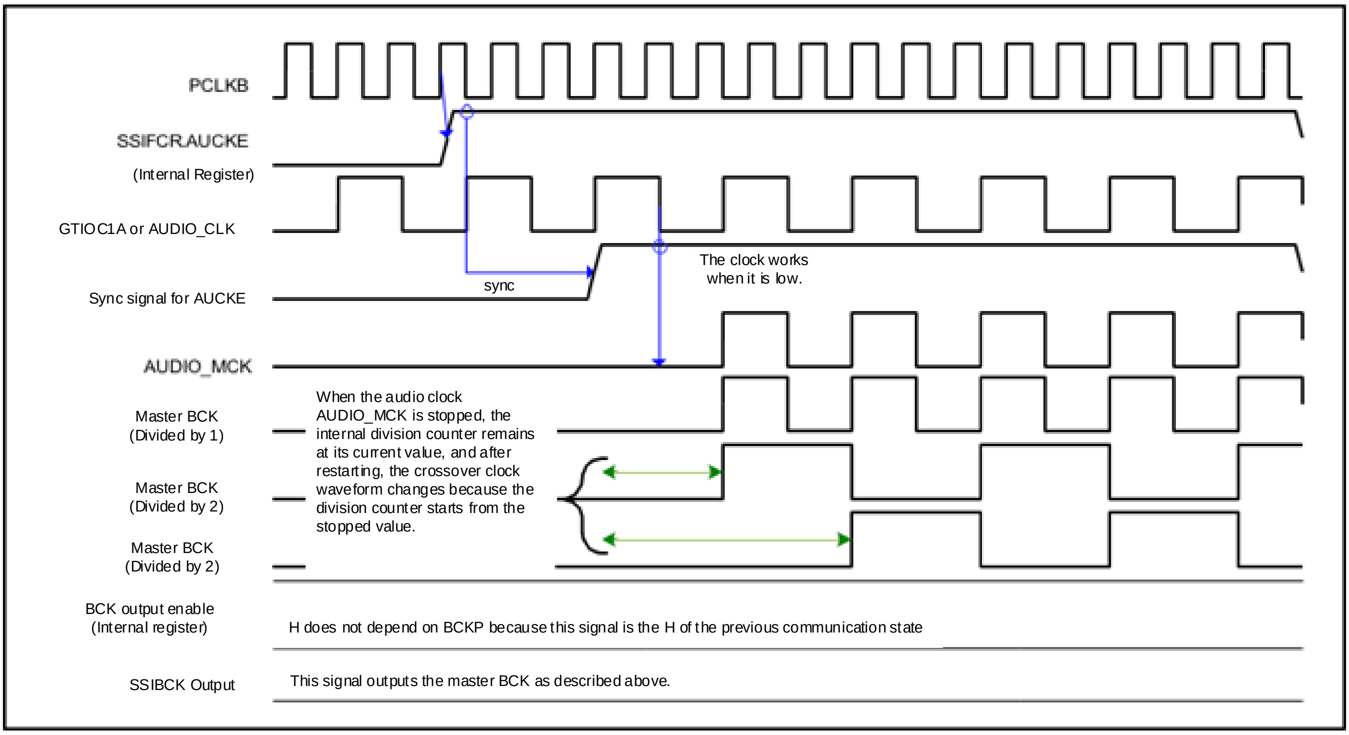


Figure 18-25 Timing diagram from stopping communication to starting master mode


PCLKB
SSIFCR.AUCKE
(Internal Register)
GTIOC1A or AUDIO_CLK
Sync signal for AUCKE
sync
The clock works when it is low.
AUDIO_MCK
Master BCK (Divided by 1)
Master BCK (Divided by 2)
Master BCK (Divided by 2)
When the audio clock AUDIO_MCK is stopped, the internal division counter remains at its current value, and after restarting, the crossover clock waveform changes because the division counter starts from the stopped value.
BCK output enable (Internal register)
H does not depend on BCKP because this signal is the H of the previous communication state
SSIBCK Output
This signal outputs the master BCK as described above.

Notice: If the audio clock AUDIO_MCK is stopped, the value of the SSIBCK pin remains unchanged. Therefore, the SSIBCK signal has to be stopped in the H (high) state.

18.3.4 FIFO status register (SSIFSR)

This register is used to indicate the status of the transmit FIFO data register and the receive FIFO data register.

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	TDC[3:0]				—	—	—	—	—	—	—	TDE
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	RDC[3:0]				—	—	—	—	—	—	—	RDF
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	RDF	Receive data full flag	0: The data received in SSIFRDR is not more than the value set in SSISCR.RDFS 1: The data received in SSIFRDR is not less than the value set in SSISCR.RDFS plus 1.	R/W
b7~b1	-	Reserved	Can only write 0 and read 0	-
b11~b8	RDC[3:0]	Number of receive FIFO data indication flags	Number of receive FIFO data indication flags	R
b15~b12	-	Reserved	Can only write 0 and read 0	-
b16	TDE	Transmit data empty flag	0: Blank space in SSIFTDR is not more than the set value in SSISCR.TDES 1: Blank space in SSIFTDR is not less than the value set in SSISCR.TDES plus 1.	R/W
b23~b17	-	Reserved	Can only write 0 and read 0	-
b27~b24	TDC[3:0]	Number of transmit FIFO data indication flags	Number of transmit FIFO data indication flags	R
b31~b28	-	Reserved	Can only write 0 and read 0	-

1) RDF bit (Receive data full flag)

The RDF bit indicates that the unread data in the Receive FIFO Data Register (SSIFRDR) is not less than the set value of SSISCR.RDFS plus one. This flag is set automatically and must be cleared by a register write access.

[Priority order of setting and clearing]

- Clear first.

[Clear conditions]

- The flag bit can be cleared in one of the following four cases:

1. This bit can be cleared by a software reset (SSIFCR.SSIRST = 1)
2. This bit can be cleared by a soft reset of the receive FIFO data register (SSIFCR.RFRST = 1)
3. Read 1 from this bit and write 0 to this bit (CPU access).
4. Issue the last read of the SSIFRDR register by using the interrupt routine of the DMA.

[Clear timing]

1. Read 1 from this bit and write 0 to this bit
2. Issue the last instruction to read the SSIFRDR register after 1 PCLKB cycle by using the interrupt routine of DMA.

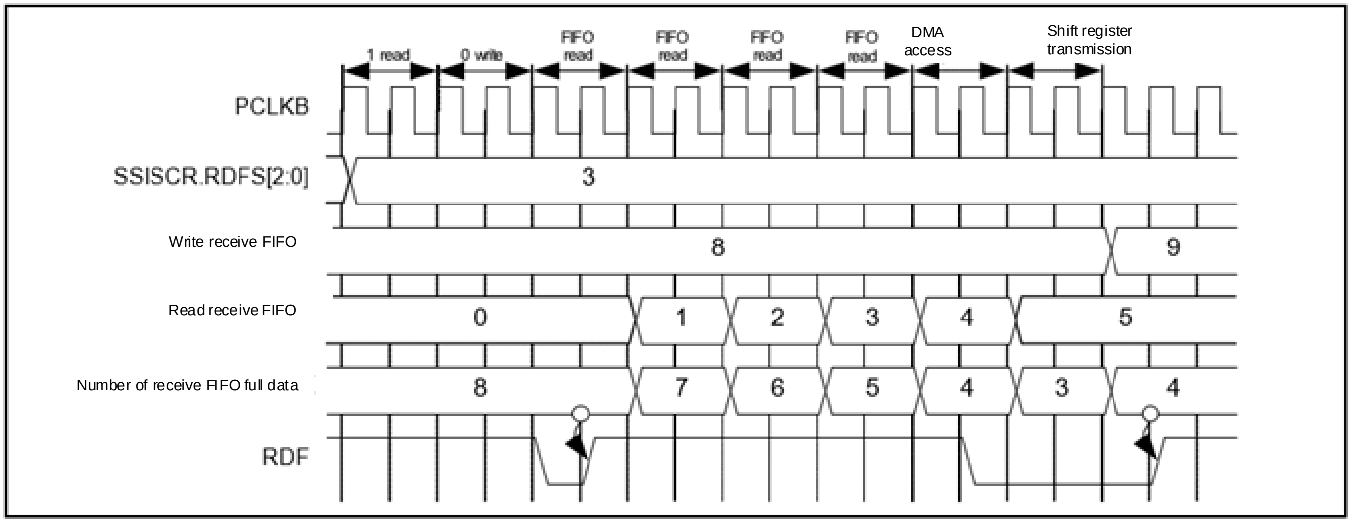
[Set conditions]

- The data in SSIFRDR is not less than the SSISCR.RDFS setting value plus 1.

[Set timing]

- The data transfer from the receive shift register is completed, and the data in SSIFRDR is not less than the value set in SSISCR.RDFS plus 1.

Figure 18-26 **Timing of setting and clearing the RDF**



2) RDC[3:0] bits (Number of receive FIFO data indication flags)

The RDC[3:0] bits indicate the amount of valid data stored in the Receive FIFO Data Register (SSIFRDR). When this flag is 0h, no data is received. When this flag is 8h, the register is filled with received data and there is no space available.

3) TDE bit (Transmit data empty flag)

The TDE bit indicates that the available space in the Transmit FIFO Data Register (SSIFTDR) is not less than the set value of SSISCR.TDES plus 1. This flag is set automatically and must be cleared by register write access.

[Priority of setting and clearing]

- Clear first.

[Clear conditions]

- - The flag bit can be cleared in one of the following four cases:

1. This bit can be cleared by software reset (SSIFCR.SSIRST = 1)
2. This bit can be cleared by a soft reset of the transmit FIFO data register (SSIFCR.TFRST = 1)
3. Read 1 from this bit and write 0 to this bit. (CPU access)
4. 4. Issue the last instruction to write the SSIFTDR register by using the interrupt routine of the DMA.

[Clear timing]

1. Read 1 from this bit and write 0 to this bit
2. Issue the last instruction to read the SSIFRDR register after 1 PCLKB cycle by using the interrupt routine of DMA.

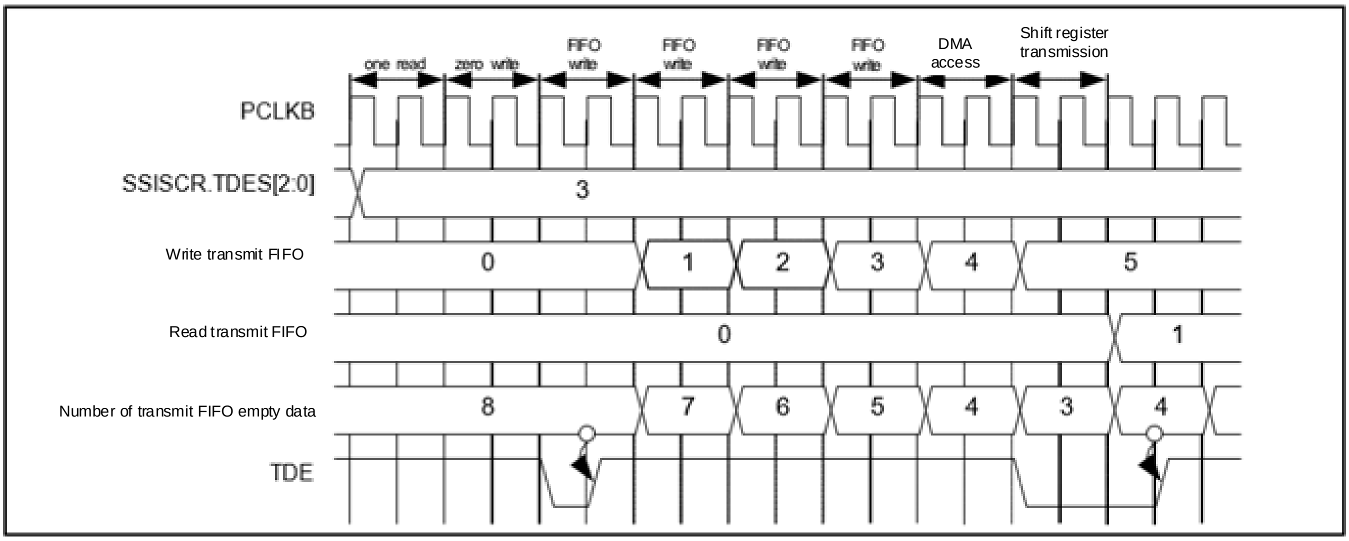
[Set conditions]

- The available space for SSIFTDR is not less than the SSISCR.TDES setting value plus 1.

[Set timing]

- When PCLKB is running, the available space for SSIFTDR is not less than the SSISCR.TDES setting value plus 1.

Figure 18-27 Timing of TDE setting and clearing



4) TDC[3:0] bits (Number of transmit FIFO data indication flags)

The TDC[3:0] bits indicate the amount of valid data stored in the Transmit FIFO Data Register (SSIFTDR). When this flag is 0h, there is no data to transmit. When this flag is 8h, there is no space to write data.

18.3.5 Transmit FIFO data register (SSIFTDR)

This register stores the data to be transmitted serially. Reading this register returns 0.

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	SSIFTDR [31:16]															
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SSIFTDR [15:0]															
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

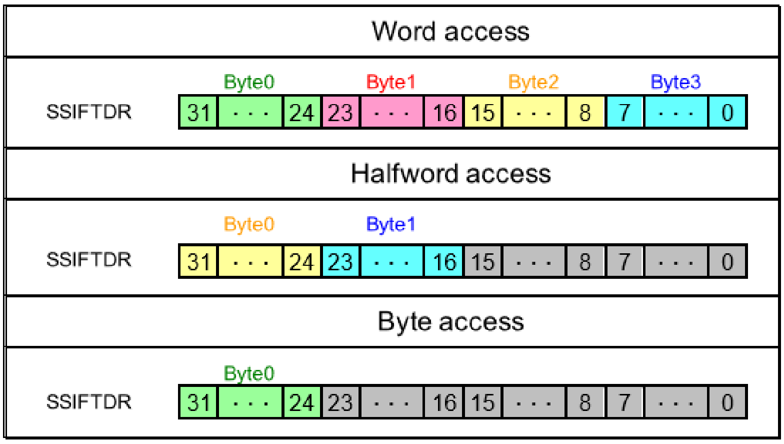
Bit	Symbol	Bit name	Description	R/W
b31~b0	SSIFTDR[31:0]	Transmit FIFO data	Transmit FIFO data	W

When using this register transmission, the transmit data null interrupt triggers the DMA to write the data to be transmitted to this register. Access to this register is determined according to the length of the data to be transmitted as shown in the table below.

Table 18-8 Access restrictions on FIFO registers

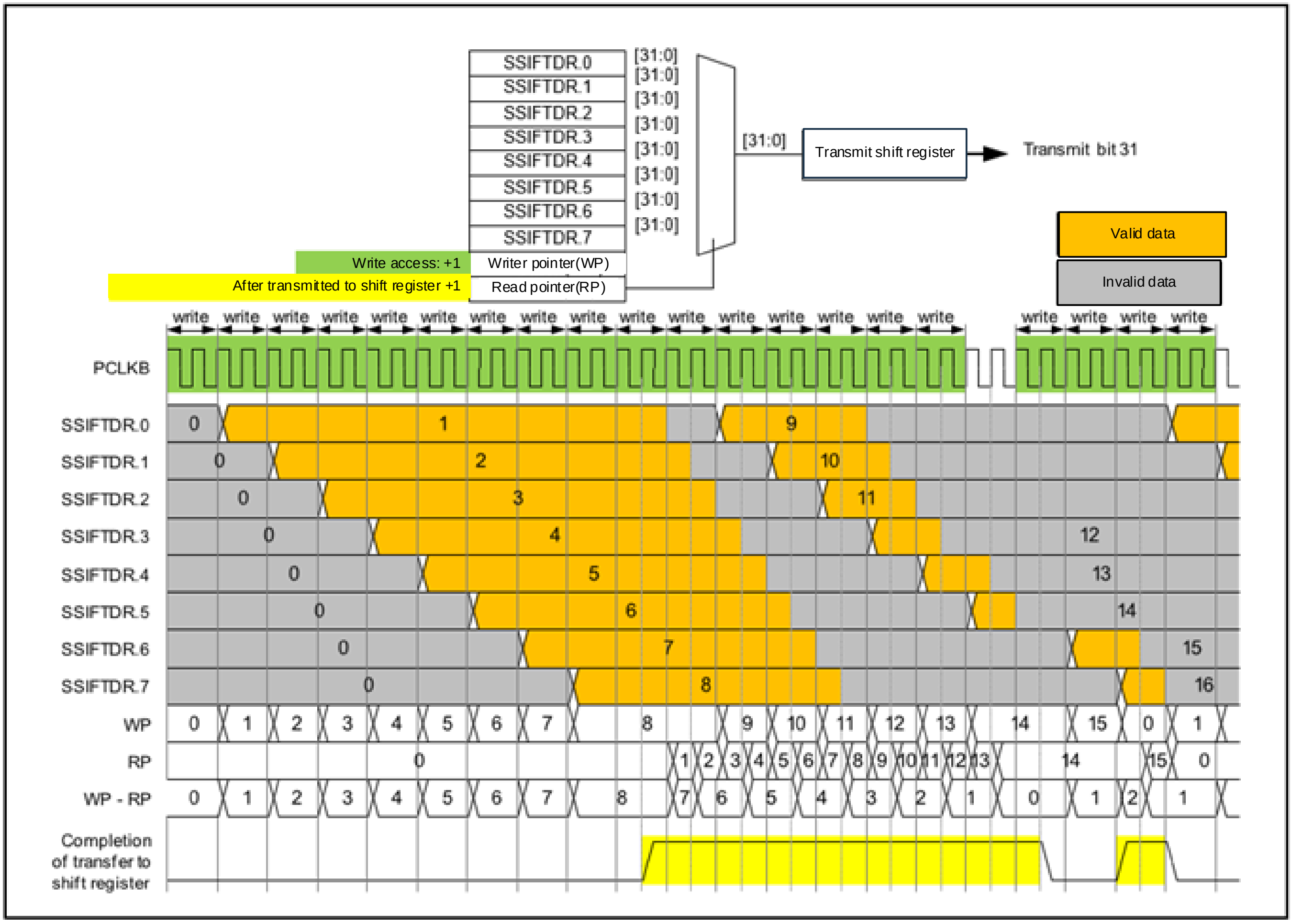
		Access Size		
SSICR.DWL[2:0]	Data length	Byte	Half-word	Word
000b	8	√	—	—
001b	16	—	√	—
010b	18	—	—	√
011b	20	—	—	√
100b	22	—	—	√
101b	24	—	—	√
110b	32	—	—	√
111b	Setting is disabled	—	—	—

Figure 18-28 Access to the transmit FIFO data register



The following figure shows an example of the configuration and operation of the transmit FIFO data register and the transmit shift register. These configurations are used to store data into the FIFO and do not involve communication.

Figure 18-29 Example of transmit FIFO data register, transmit shift register configuration, and FIFO action



18.3.6 Receive FIFO data register (SSIFRDR)

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	SSIFRDR [31:16]															
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

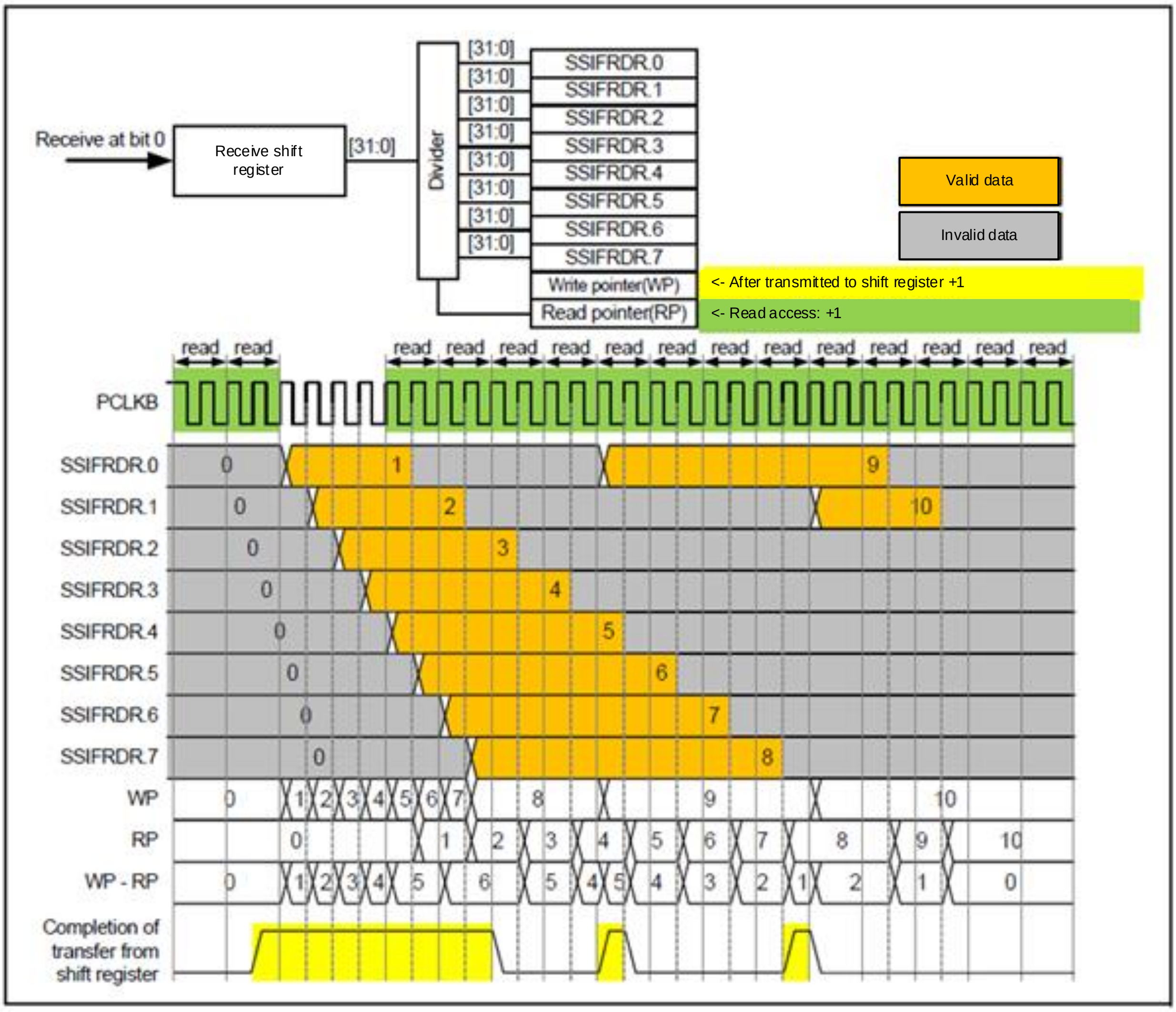
	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SSIFRDR [15:0]															
Reset value::	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b31~b0	SSIFRDR[31:0]	Receive FIFO data	Receive FIFO data	W

When using this register for reception, the receive data full interrupt triggers a DMA action to read data from this register. The access to this register is determined according to the length of the data to be transferred.

The access to the receive FIFO data register is the same as the access to the transmit FIFO data register.

Figure 18-30 Example of receive FIFO data register, receive shift register configuration and FIFO action



18.3.7 TDM mode register (SSITDMR)

This register is used to set the audio format, including the communication format, LR clock/frame sync continuous mode, and BCK output stop settings.

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reset value:	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	-	-	-	-	-	-	BCKAS TP	LRCON T	-	-	-	-	-	-	OMOD[1:0]	
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b1，b0	OMOD[1:0]	Audio format selection *3, *4	00:I2S format 01:No selection 10:Mono format 11:Selection disabled	R/W
b7~b2	-	Reserved	Can only write 0 and read 0	-
b8	LRCONT	Enable LRCK/FS continuous *1, *2	0:Disable LRCK/FS persistence 1:Enable LRCK/FS to persistence	R/W
b9	BCKASTP	Enable BCK to stop output when SSIE is idle *1, *2	0: BCK always output to SSIBCK pin 1: Automatically control whether BCK is output to SSIBCK pin	R/W
b31~b10	-	Reserved	Can only write 0 and read 0	-

Note 1. This bit is valid only for master mode communication (SSICR.MST=1) and is not valid for slave mode communication (SSICR.MST=0).

Note 2. The BCKASTP and LRCONT bits cannot be set to 1 at the same time.

Note 3. When SSIE communicates (SSISR.IIRQ=0), writing these bits is forbidden, and subsequent operation is unpredictable if these bits are overwritten.

Note 4. When communicating with another party device with SSIE compatible communication format, set the corresponding communication format before enabling communication.

1) OMOD[1:0] bits (audio format selection)

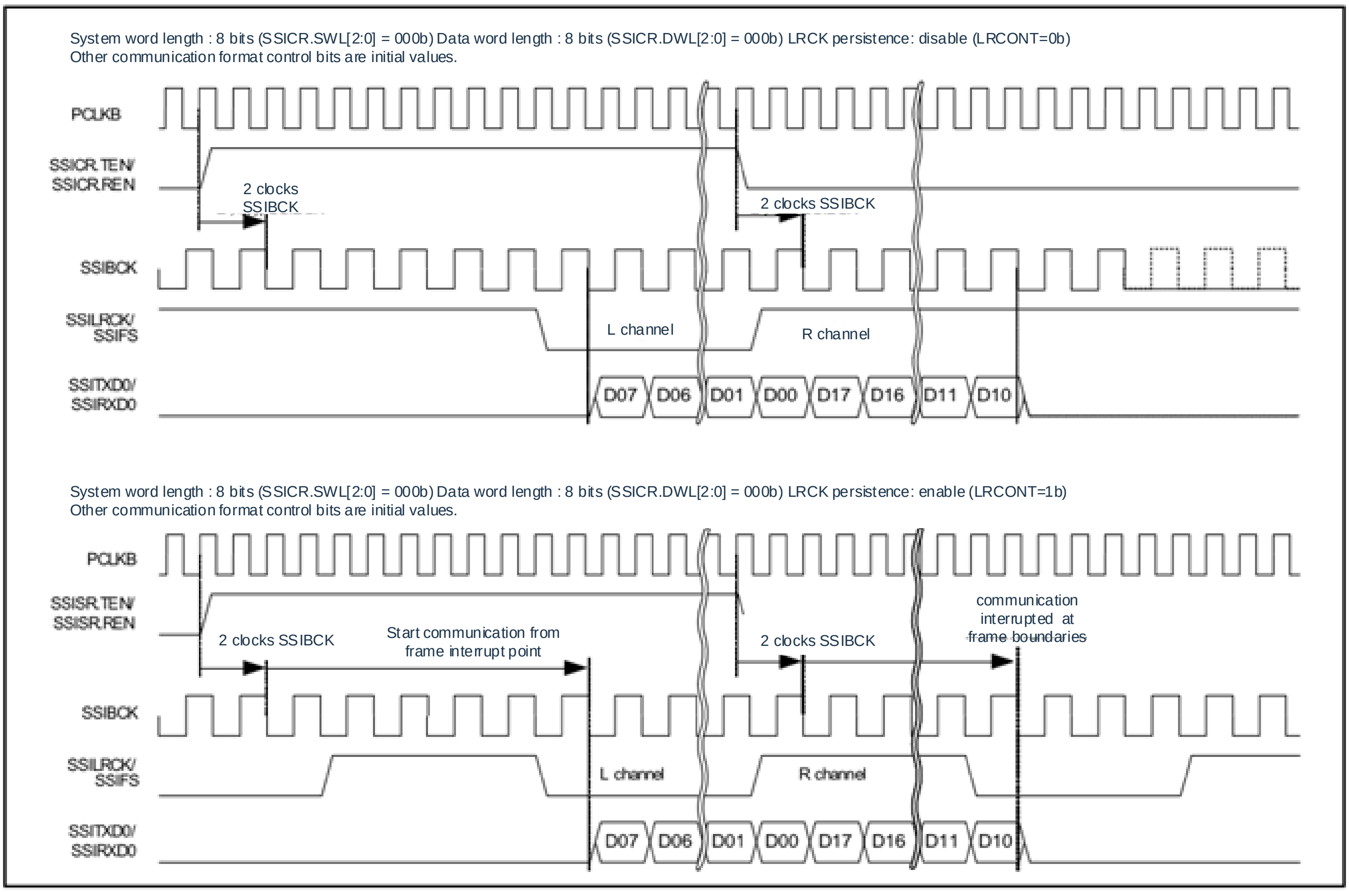
The OMOD[1:0] bits are used to select the audio format. These bits can only be written when the LR clock power at the SSILRCK/SSIFS pins is stopped. For more information on the LR clock output, refer to the detailed description of the LRCONT bits in the TDM Mode Register (SSITDMR).

2) LRCONT bit (Enable LRCK/FS persistence)

In master mode (SSICR.MST=1), the LRCONT bit is used to enable or disable the SSILRCK/SSIFS pin persistence output when the SSIE is idle (SSISR.IIRQ=1).

When this bit is set to 1 (enable LR clock/frame sync persistence) in master mode (SSICR.MST=1), the signal can be output from the SSILRCK/SSIFS pins even in the idle state.

Figure 18-31 Example of LR clock/frame synchronization running continuously


System word length : 8 bits (SSICR.SWL[2:0] = 000b) Data word length : 8 bits (SSICR.DWL[2:0] = 000b) LRCK persistence: disable (LRCONT=0b)
Other communication format control bits are initial values.
PCLKB
SSICR.TEN/
SSICR.REN
2 clocks
SSIBCK
2 clocks SSIBCK
SSIBCK
SSILRCK/
SSIFS
L channel
R channel
SSITXD0/
SSIRXD0
D07
D06
D01
D00
D17
D16
D11
D10
System word length : 8 bits (SSICR.SWL[2:0] = 000b) Data word length : 8 bits (SSICR.DWL[2:0] = 000b) LRCK persistence: enable (LRCONT=1b)
Other communication format control bits are initial values.
PCLKB
SSISR.TEN/
SSISR.REN
2 clocks SSIBCK
Start communication from
frame interrupt point
2 clocks SSIBCK
communication
interrupted at
frame boundaries
SSIBCK
SSILRCK/
SSIFS
L channel
R channel
SSITXD0/
SSIRXD0
D07
D06
D01
D00
D17
D16
D11
D10

3) BCKASTP bit (enable stopping BCK output when SSIE is idle)

In master mode communication (SSICR.MST=1), the BCKASTP bit is used to turn on or off the ability to output BCK to the SSIBCK pin. The value of this bit can be changed only after the communication format to be used has been set. To use this bit:

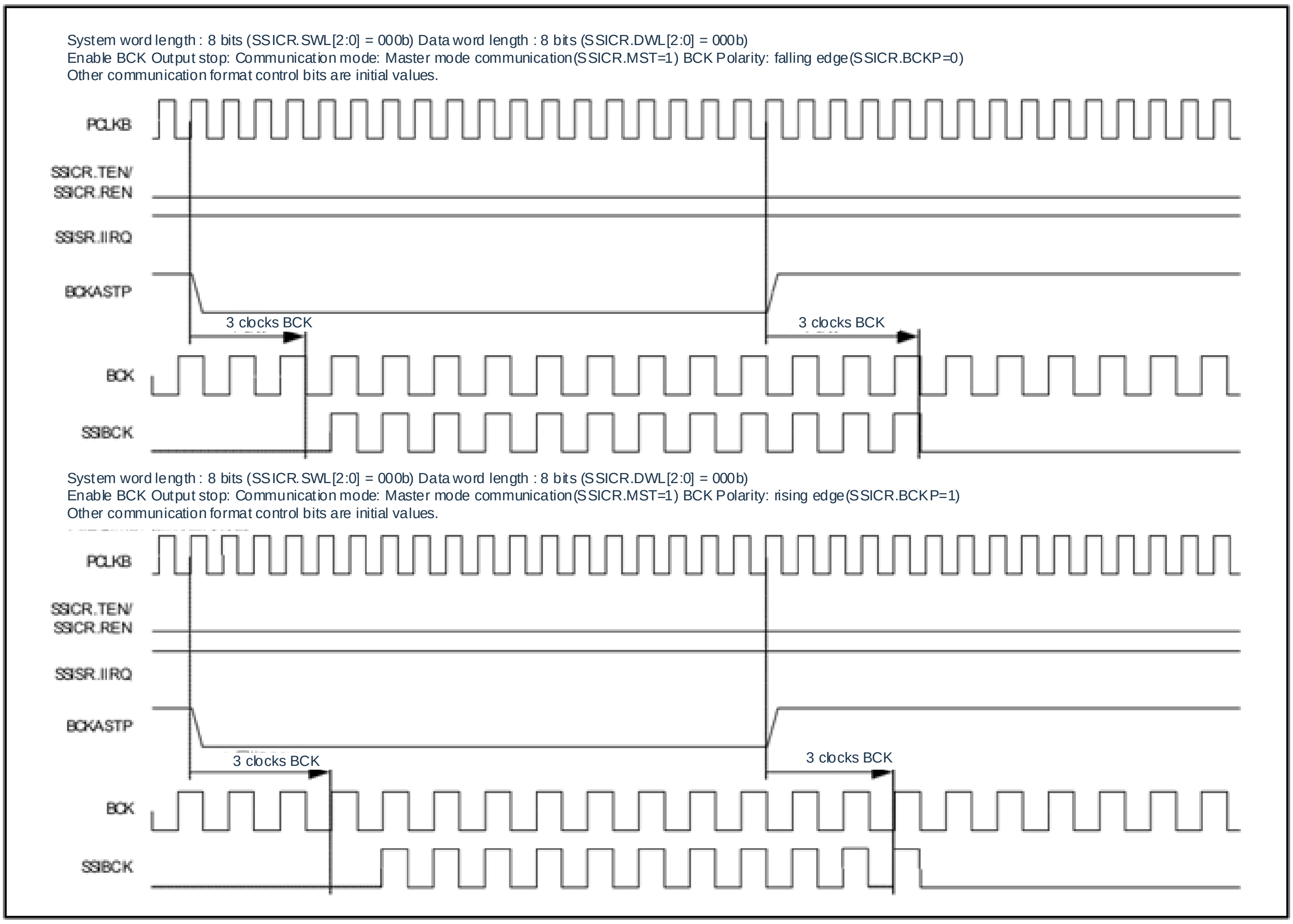
- BCKASTP bit is written to "0", then communication starts
- During communication, 1 is written to the BCKASTP bit. When communication is stopped, the bit clock output to the SSIBCK pin is automatically stopped.
- To resume communication, set SSIE to idle state (SSICR.IIRQ = 1), start AUDIO_MCK (SSIFCR.AUCKE = 1), and then write 0 to the BCKASTP bit. When the communication is in master mode (SSICR.MST = 1) and SSIE is idle (SSICR.IIRQ = 1), the BCKASTP bit status and SSIBCK pin output are shown in the following table

Table 18-9 BCKASTP bit status and SSIBCK pin output

BCKASTP bit	SSIBCK pin output status
0	Output
1	Stop

Note: The BCKASTP bit cannot be used when the other device (slave device) requests clock output from the SSIBCK pin before and during communication. In this case, the BCKASTP bit is used to stop the clock only after communication. See Figure 18.32 for the timing of enable clock stop function

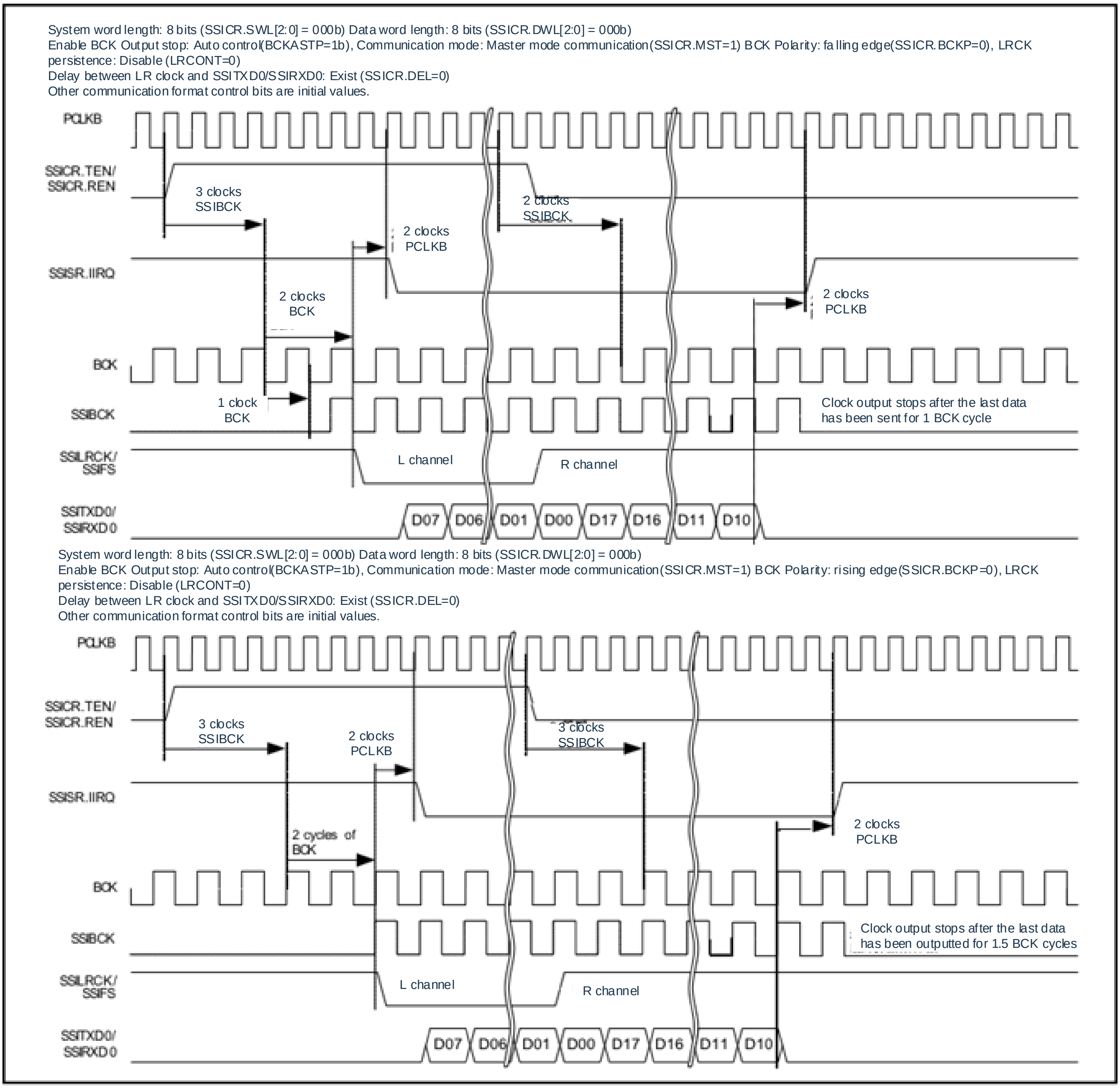
Figure 18-32 Example of the action of the BCKASTP bit in the idle state


System word length : 8 bits (SSICR.SWL[2:0] = 000b) Data word length : 8 bits (SSICR.DWL[2:0] = 000b)
Enable BCK Output stop: Communication mode: Master mode communication(SSICR.MST=1) BCK Polarity: falling edge(SSICR.BCKP=0)
Other communication format control bits are initial values.
PCLKB
SSICR.TEN/
SSICR.REN
SSISR.IIRQ
BCKASTP
3 clocks BCK
3 clocks BCK
BCK
SSIBCK
System word length : 8 bits (SSICR.SWL[2:0] = 000b) Data word length : 8 bits (SSICR.DWL[2:0] = 000b)
Enable BCK Output stop: Communication mode: Master mode communication(SSICR.MST=1) BCK Polarity: rising edge(SSICR.BCKP=1)
Other communication format control bits are initial values.
PCLKB
SSICR.TEN/
SSICR.REN
SSISR.IIRQ
BCKASTP
3 clocks BCK
3 clocks BCK
BCK
SSIBCK

In master mode (SSICR.MST=1) when the BCK output stop function is enabled (BCKASTP=1), the BCK output to the SSIBCK pin is as follows:

- Output start timing: BCK is output at the appropriate time to generate a valid edge when the LR clock/frame sync signal becomes valid.
- Output stop timing: 1 to 1.5 clock cycles after the frame boundary.

Figure 18-33 Example of communication operation with BCKASTP=1



System word length: 8 bits (SSICR.SWL[2:0] = 000b) Data word length: 8 bits (SSICR.DWL[2:0] = 000b)
Enable BCK Output stop: Auto control(BCKASTP=1b), Communication mode: Master mode communication(SSICR.MST=1) BCK Polarity: falling edge(SSICR.BCKP=0), LRCK persistence: Disable (LRCONT=0)
Delay between LR clock and SSITXD0/SSIRXD0: Exist (SSICR.DEL=0)
Other communication format control bits are initial values.

System word length: 8 bits (SSICR.SWL[2:0] = 000b) Data word length: 8 bits (SSICR.DWL[2:0] = 000b)
Enable BCK Output stop: Auto control(BCKASTP=1b), Communication mode: Master mode communication(SSICR.MST=1) BCK Polarity: rising edge(SSICR.BCKP=0), LRCK persistence: Disable (LRCONT=0)
Delay between LR clock and SSITXD0/SSIRXD0: Exist (SSICR.DEL=0)
Other communication format control bits are initial values.

18.3.8 Status control register (SSISCR)

	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Reset value:	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	-	-	-	-	-	TDES[2:0]			-	-	-	-	-	RDFS[2:0]		
Reset value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b2~b0	RDFS[2:0]	RDF setting condition selection*1	b2~b0 000: SSIFRDR has 1 or more segments of data 001: SSIFRDR has 2 or more segments of data 010: SSIFRDR has 3 or more segments of data 011: SSIFRDR has 4 or more segments of data 100: SSIFRDR has 5 or more segments of data 101: SSIFRDR has 6 or more segments of data 110: SSIFRDR has 7 or more segments of data 111: SSIFRDR has 8 or more segments of data	R/W
b7~b3	-	Reserved	Can only write 0 and read 0	-
b10~b8	TDES[2:0]	TDE setting condition selection*1	b10~b8 000: SSIFTDR has 1 or more blank segments 001: SSIFTDR has 2 or more blank segments 010：SSIFTDR has 3 or more blank segments 011: SSIFTDR has 4 or more blank segments 100: SSIFTDR has 5 or more blank segments 101: SSIFTDR has 6 or more blank segments 110: SSIFTDR has 7 or more blank segments 111: SSIFTDR has 8 or more blank segments	R/W
b31~b11	-	Reserved	Can only write 0 and read 0	-

Note 1.: Writing these bits while the SSIE is in communication (SSISR.IIRQ=0) is forbidden. If written, there is no guarantee that the subsequent actions are correct.

1) RDFS[2:0] bits (RDF Set condition selection)

RDFS[2:0] Conditions for selecting receive data full flag genenration (RDF).

2) TDES[2:0] bits (TDE Set condition selection)

TDES[2:0] Conditions for selecting transmit data empty flag genenration (TDE).

18.4 Communication format

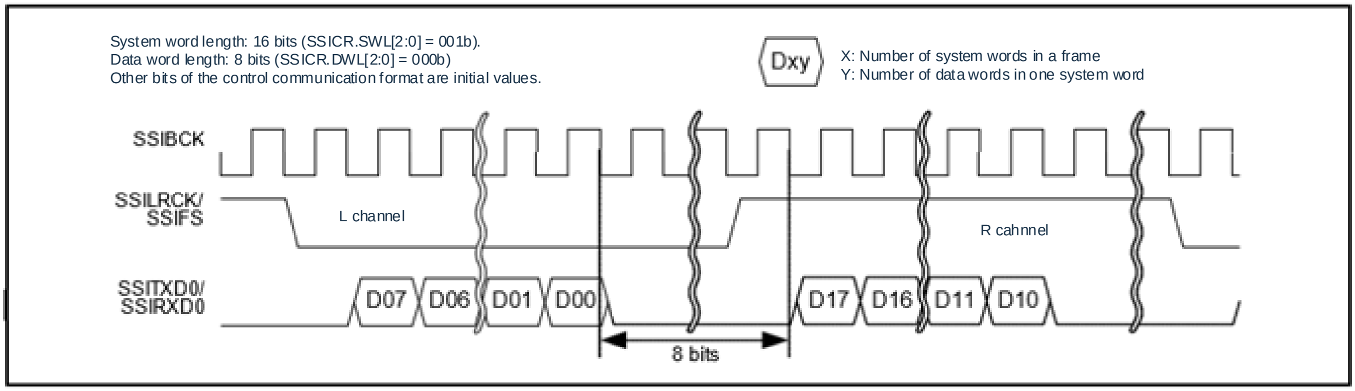
The communication formats supported by SSIE are listed in the table.

Table 18-10 Available communication formats

Communication format	SSITDMR.OMOD[1:0]
I²S format	00
Mono format	10

This section describes the serial data structure of the communication format. The serial data structure consists of system word length (defined by SSICR.SWL[2:0]) and data word length (defined by SSICR.DWL[2:0]). If the data word length is less than the system word length, the padding bits are transmitted in the serial data. For more information, see the following figure.

Figure 18-34 Example of I²S format padded bit transmission when system word length > data word length


System word length: 16 bits (SSICR.SWL[2:0] = 001b).
Data word length: 8 bits (SSICR.DWL[2:0] = 000b)
Other bits of the control communication format are initial values.
Dxy
X: Number of system words in a frame
Y: Number of data words in one system word
SSIBCK
SSILRCK/
SSIFS
L channel
R cahnnel
SSITXD0/
SSIRXD0
D07
D06
D01
D00
D17
D16
D11
D10
8 bits

The following table lists the number of padding bits to be transmitted for each combination of system word length (SSICR.SWL[2:0]) and data word length (SSICR.DWL[2:0]). "-" indicates that the setting is disabled.

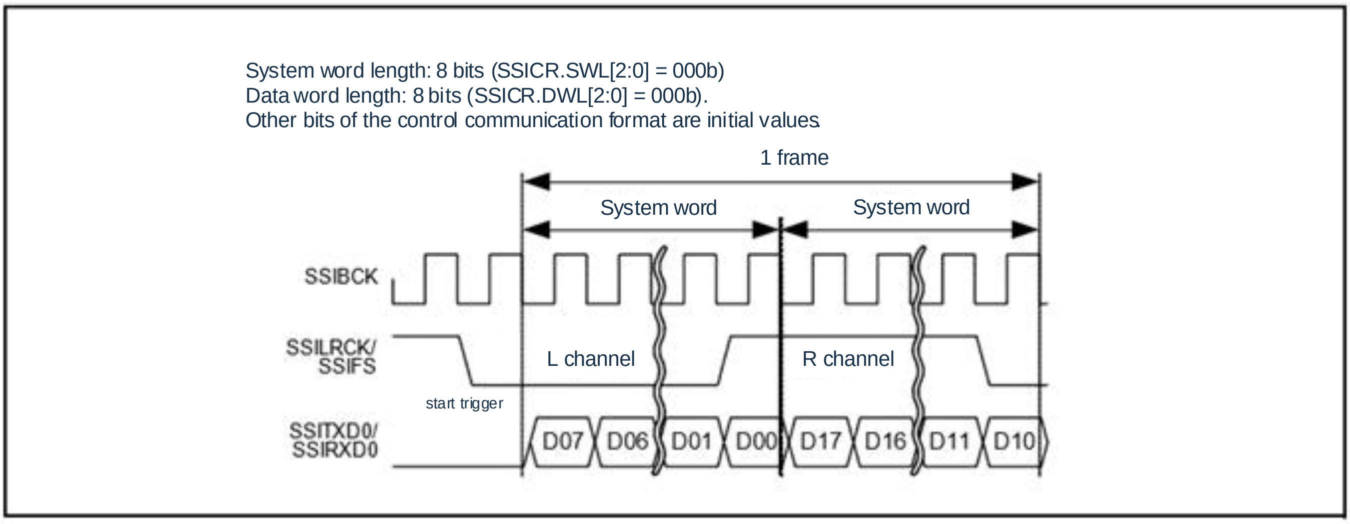
Table 18-11 Number of padding bits

	SSICR.DWL[2:0]	000b	001b	010b	011b	100b	101b	110b	111b
SSICR.SWL[2:0]	Data word length System word length	8	16	18	20	22	24	32	Prohibit settings
000b	8	0	—	—	—	—	—	—	—
001b	16	8	0	—	—	—	—	—	—
010b	24	16	8	6	4	2	0	—	—
011b	32	24	16	14	12	10	8	0	—
100b	48	40	32	30	28	26	24	16	—
101b	64	56	48	46	44	42	40	32	—
110b	128	120	112	110	108	106	104	96	—
111b	256	248	240	238	236	234	232	224	—

18.4.1 I^{2}S format

The I2S format is used to connect I2S-compatible serial devices. In this format setting (SSITDMR.OMOD [1:0]=00b), a frame consists of two system words, one for the L channel and the other for the R channel. the SSILRCK/SSIFS signal is low for the L channel and high for the R channel. The polarity of the signal is set by the SSICR.LRCKP bit.

Figure 18-35 I2S format is not padded when system word length = data word length


System word length: 8 bits (SSICR.SWL[2:0] = 000b)
Data word length: 8 bits (SSICR.DWL[2:0] = 000b).
Other bits of the control communication format are initial values.

Please refer to 18.6.1 Idle status for the external pin state when SSIE is in idle state.

Notice: The SSILRCK/SSIFS pins in the SSIE are used for synchronization of communication. When the SSIE is in slave mode (SSICR.MST=0), the SSIE communication format must match the communication format of the other party's device. The SSIE uses the signal input from the SSILRCK/SSIFS pin as the trigger signal to initiate communication.

18.4.2 Mono format

The mono format is used to connect mono-compatible serial devices. When mono format is specified (SSITDMR.OMOD [1:0]=10b), a frame consists of one system word. And, the rising edge of the SSILRCK/SSIFS signal triggers communication initiation. Figure 18-36 and Figure 18-37 illustrate the unpadded and padded mono formats, respectively.

Figure 18-36 Short frame in mono format without padding when system word length = data word length

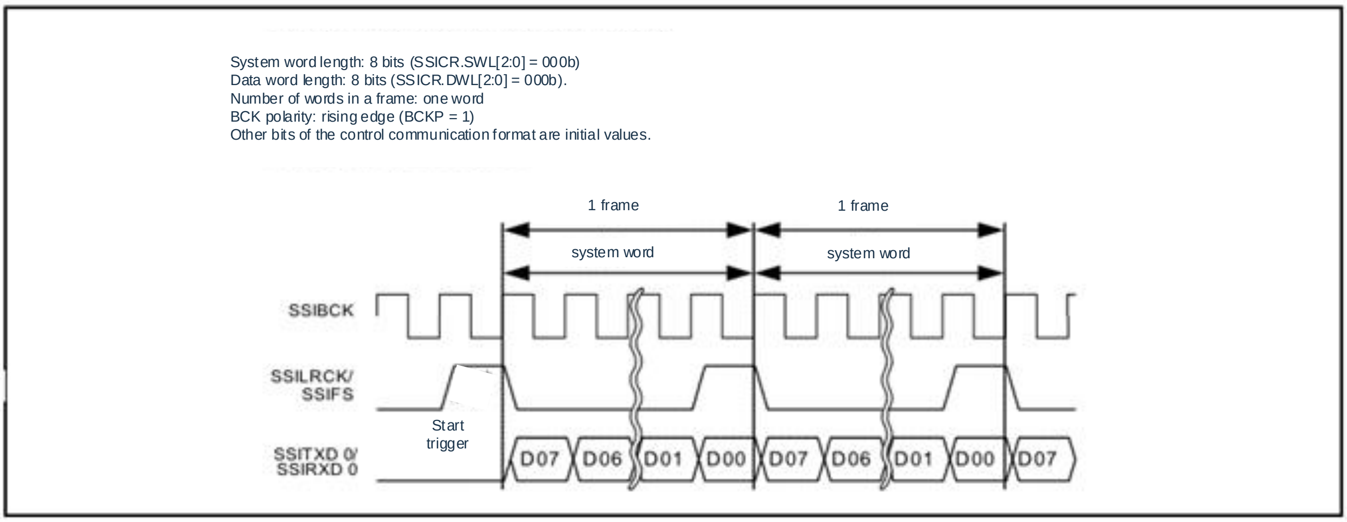
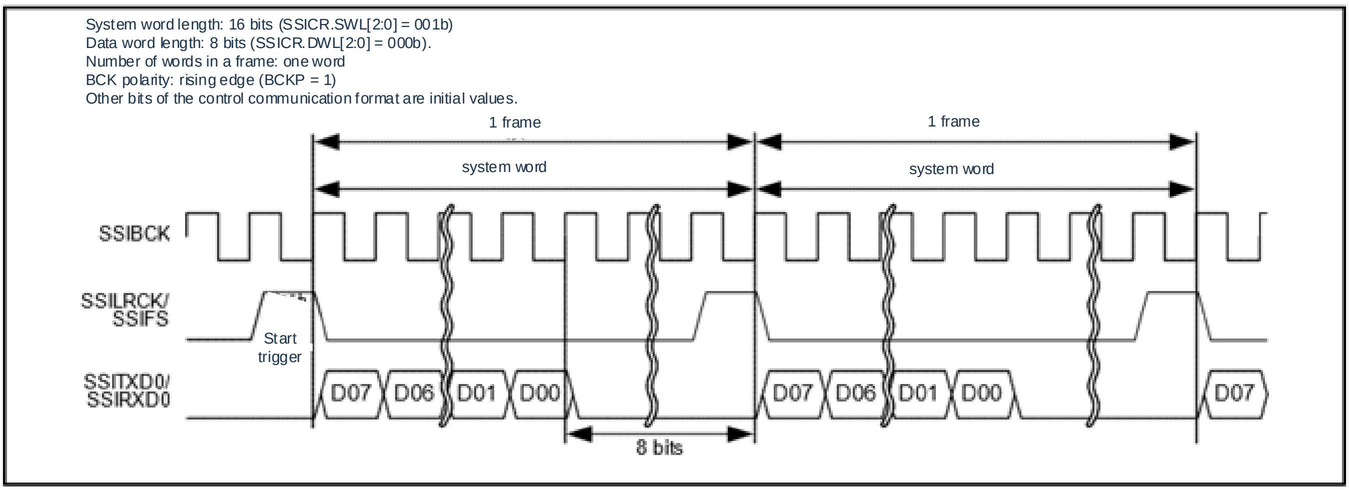

System word length: 8 bits (SSICR.SWL[2:0] = 000b)
Data word length: 8 bits (SSICR.DWL[2:0] = 000b).
Number of words in a frame: one word
BCK polarity: rising edge (BCKP = 1)
Other bits of the control communication format are initial values.

Figure 18-37 Short frame with padded mono format when system word length > data word length



System word length: 16 bits (SSICR.SWL[2:0] = 001b)
Data word length: 8 bits (SSICR.DWL[2:0] = 000b).
Number of words in a frame: one word
BCK polarity: rising edge (BCKP = 1)
Other bits of the control communication format are initial values.

The mono formats supported by SSIE include short frame and long frame. The difference between these two formats is described in 18.4.2.1 Short frame and 18.4.2.2 Long frame.

Refer to Section 18.6.1 Idle status for the state of the external pins when the SSIE is in the idle state.

Notice: The SSILRCK/SSIFS pins in the SSIE are used for synchronization of communication. When the SSIE is in slave mode (SSICR.MST=0), the SSIE communication format must match the communication format of the other party's device. The SSIE uses the signal input from the SSILRCK/SSIFS pin as the trigger signal to initiate communication.

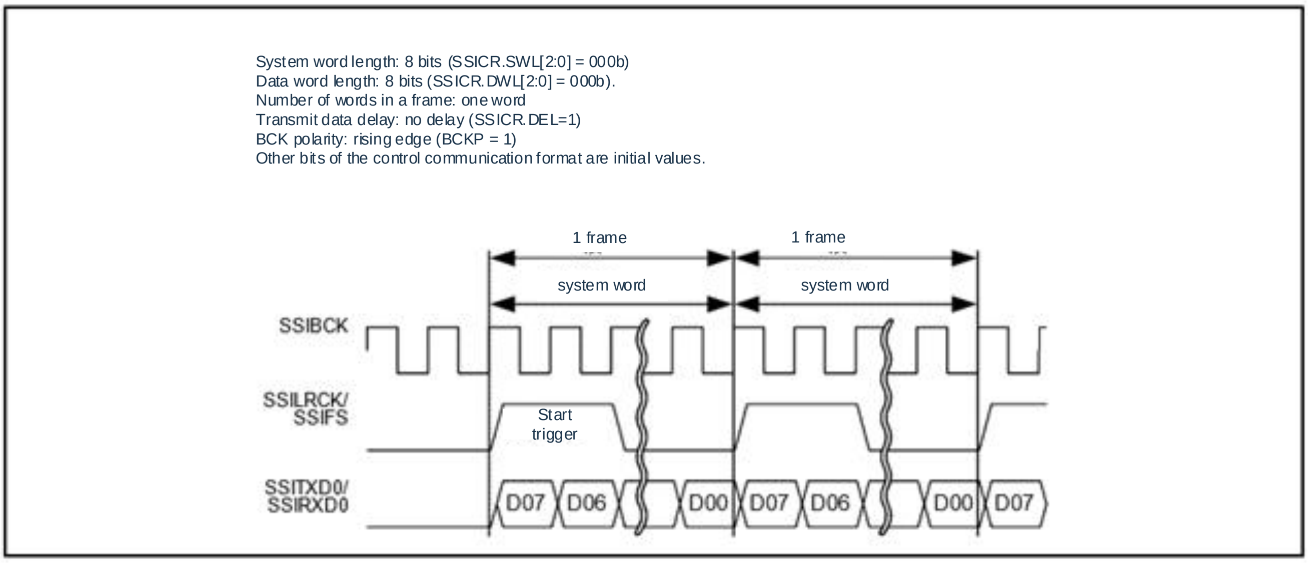
18.4.2.1 Short frame

When a short frame is used (SSICR.DEL=0), the SSILRCK/SSIFS signal is set high for 1 SSIBCK cycle, indicating the start of serial data transmission. Data transmission starts from the falling edge of the signal.

18.4.2.2 Long frame

When a long frame is used (SSICR.DEL=1), the SSILRCK/SSIFS signal indicating the start of serial data transfer is set high for 2 SSIBCK cycles. Figure 18-38 shows a long frame in mono format without padding. The data transfer starts from the rising edge of the signal.

Figure 18-38 A long frame in mono format without padding



System word length: 8 bits (SSICR.SWL[2:0] = 000b)
Data word length: 8 bits (SSICR.DWL[2:0] = 000b).
Number of words in a frame: one word
Transmit data delay: no delay (SSICR.DEL=1)
BCK polarity: rising edge (BCKP = 1)
Other bits of the control communication format are initial values.

18.5 Communication mode

Table 18-12 lists the communication modes supported by SSIE. Table 18-13 lists the control bits that are not available for each communication mode.

Table 18-12 Communication mode

Communication mode	SSICR.MST bit	SSICR.REN bit	SSICR.TEN bit
Slave mode transmission	0	0	1
Slave mode reception	0	1	0
Slave mode transmission and reception	0	1	1
Master mode transmission	1	0	1
Master mode reception	1	1	0
Master mode transmission and reception	1	1	1

Table 18-13 Unavailable control bits in each communication mode

Control bit	Communication Mode					
	Slave mode reception	Slave mode transmission	Slave mode transmission and reception	Master mode reception	Master mode transmission	Master mode transmission and reception
SSICR.CKS	Invalid	Invalid	Invalid	Valid	Valid	Valid
SSICR.CKDV	Invalid	Invalid	Invalid	Valid	Valid	Valid
SSICR.MUEN	Invalid	Valid	Valid	Invalid	Valid	Valid
SSICR.TEN	Invalid	Valid	Valid	Invalid	Valid	Valid
SSICR.REN	Valid	Invalid	Valid	Valid	Invalid	Valid
SSIFCR.AUCKEN	Invalid	Invalid	Invalid	Valid	Valid	Valid
SSIFCR.TIE	Invalid	Valid	Valid	Invalid	Valid	Valid
SSIFCR.RIE	Valid	Invalid	Valid	Valid	Invalid	Valid
SSIFCR.TFRST	Invalid	Valid	Valid	Invalid	Valid	Valid
SSIFCR.RFRST	Valid	Invalid	Valid	Valid	Invalid	Valid
SSITDMR.BCKASTP	Invalid	Invalid	Invalid	Valid	Valid	Valid
SSITDMR.LRCONT	Invalid	Invalid	Invalid	Valid	Valid	Valid
SSITDMR.OMOD	Valid	Valid	Valid	Valid	Valid	Valid
SSISCR.TDES	Invalid	Valid	Valid	Invalid	Valid	Valid
SSISCR.RDFS	Valid	Invalid	Valid	Valid	Invalid	Valid

Notice: Invalid means that the bit can be written, but has no effect on the action.

18.5.1 Slave mode communication

When SSICR.MST=0, SSIE operates in slave mode. The SSIBCK and SSILRCK/SSIFS signals for serial data communication must be provided by an external device. If the communication format of these signals does not match the communication format of SSIE, the operation is unpredictable.

18.5.2 Master mode communication

When SSICR.MST=1, the SSIE operates in master mode. The SSIBCK and SSILRCK/SSIFS signals for serial data communication must be generated from the internal audio clock. The signal format is specified by SSIE. If the communication format of the slave does not match the SSIE communication format, the operation is unpredictable.

18.5.3 Transmission

When SSICR.TEN is 1 and SSICR.REN is 0, the SSIE sends serial data to the other party's device. If the communication format of the other party's device does not match the SSIE communication format, the operation is unpredictable.

18.5.4 Reception

When SSICR.TEN is 0 and SSICR.REN is 1, the SSIE receives serial data from the other party's device. If the communication format of the other party's device does not match the SSIE communication format, the operation is unpredictable.

18.5.5 Transmission and reception

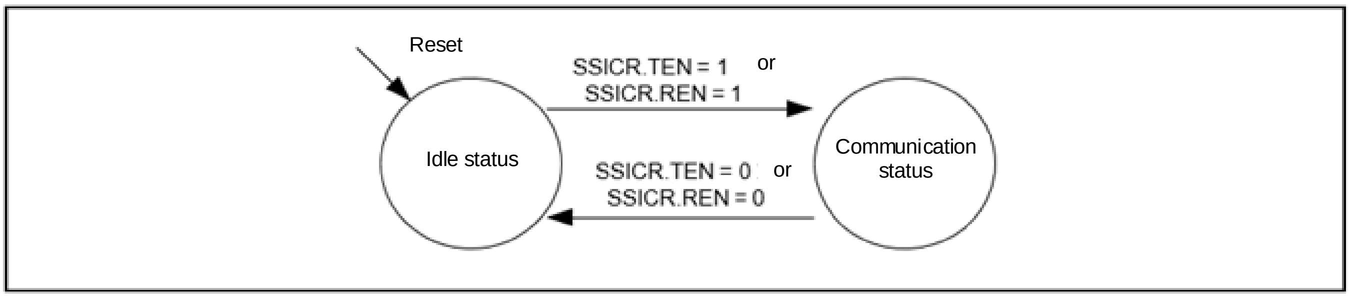
When SSICR.TEN is 1 and SSICR.REN is 1, serial data is sent and received between the SSIE and the other party's device. If the communication format of the other party's device does not match the SSIE communication format, the operation is unpredictable.

18.6 Operation

SSIE has the following two operating states:

- Idle status (SSISR.IIRQ = 1)
- Communication status (SSISR.IIRQ = 0)

Figure 18-39 **SSIE Status Transition**



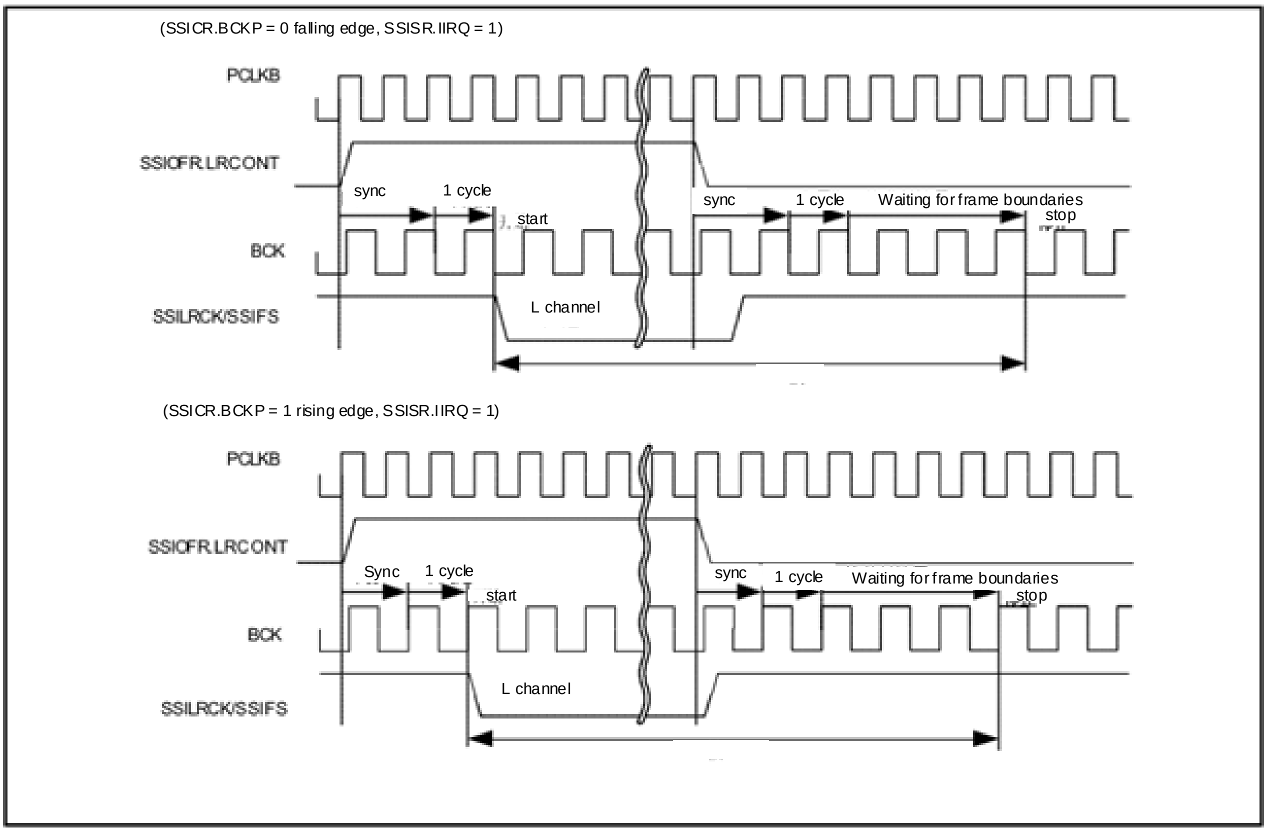
18.6.1 Idle status

In this state, the SSIE communication is stopped. However, if the SSICR.MST bit is 1, the BCK and LR clock/frame sync signals output to the external pins can be controlled by the SSITDMR.BCKASTP and SSITDMR.LRCONT bits. This function is common to all formats. See the following table for details.

Table 18-14 External Pin Outputs in Idle Status

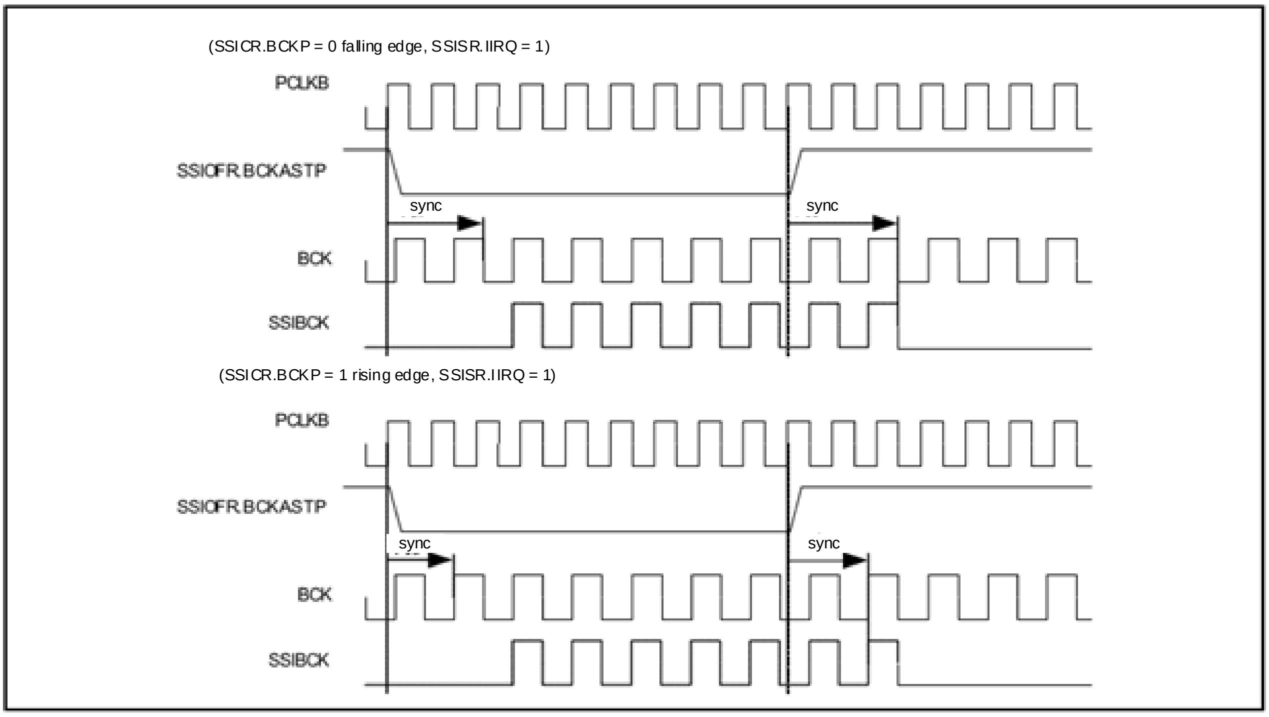
SSICR.MST	SSITDMR.BCKASTP	SSITDMR.LRCONT	Pin output		
			SSIBCK	SSILRCK/SSIFS	SSITXD0
0	—	—	Stop	Stop	Stop
1	0	0	Supply	Stop	Stop
1	0	1	Supply	Supply	Stop
1	1	0	Stop	Stop	Stop
1	1	1	Stop	Supply	Stop

Figure 18-40 Example of disabling LR clock/frame synchronization persistence by setting SSITDMR.LRCONT



Notice: When communicating in master mode (SSICR.MST=1), the output of SSILRCK/ SSIFS pin can be stopped by changing the SSITDMR.LRCONT bit from 1 to 0 for SSIE in idle status, while ensuring that the other party's device is not affected.

Figure 18-41 Example of stopping SSIBCK by setting SSITDMR.BCKASTP


(SSICR.BCKP = 0 falling edge, SSISR.IIRQ = 1)
PCLKB
SSIOFR.BCKASTP
sync
sync
BCK
SSIBCK
(SSICR.BCKP = 1 rising edge, SSISR.IIRQ = 1)
PCLKB
SSIOFR.BCKASTP
sync
sync
BCK
SSIBCK

Notice: During master mode communication (SSICR.MST=1), for SSIE in idle state, the SSIBCK pin output stops when the SSITDMR.BCKASTP bit is changed from 0 to 1, while ensuring that the other party's device is not affected.

18.6.2 Communication status

Figure 18-42 shows the transition of the communication state, and Table 18-15 lists the conditions for the transition. If the transition conditions are not met, the state is not transferred.

Figure 18-42 Communication state transition

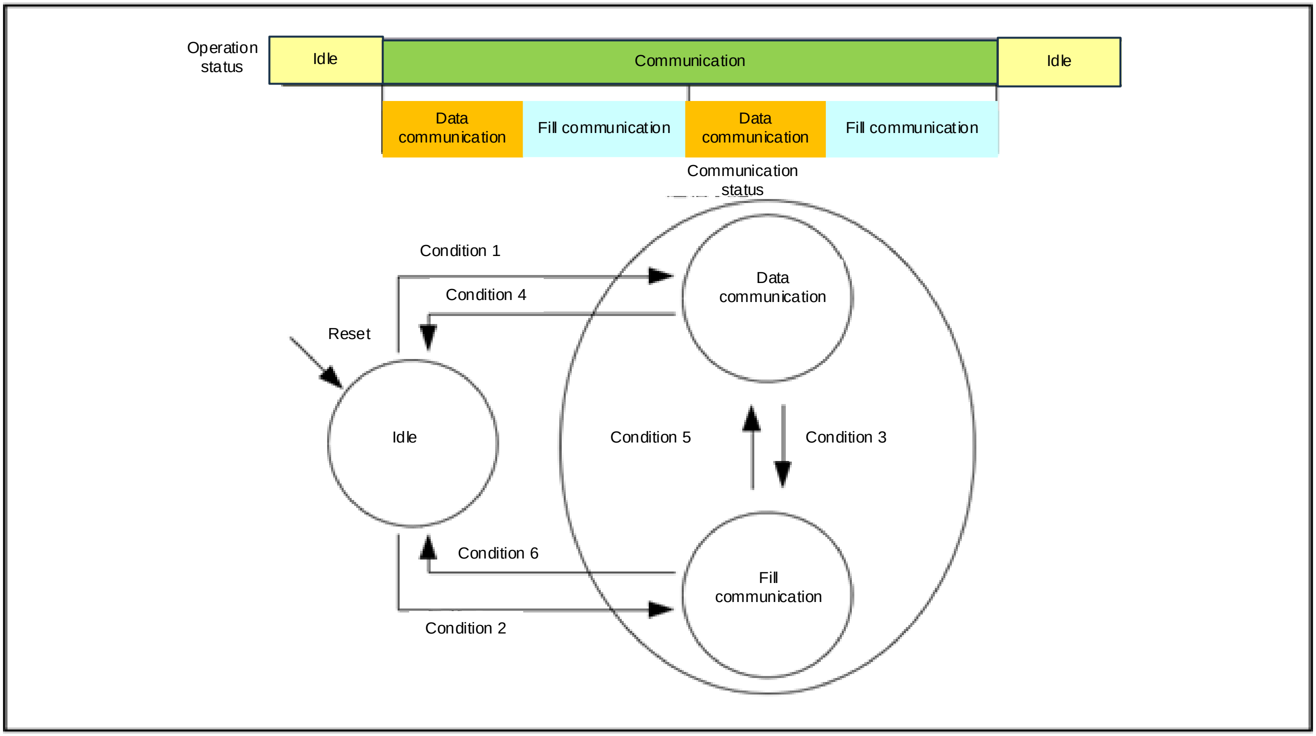


Table 18-15 Conditions for communication state transition

Condition No.	Transition condition
1	Write SSICR.TEN = 1 or SSICR.REN = 1 when SSICR.SDTA = 0 or if there is no padding bit
2	When SSICR.SDTA = 1 and there is a padding bit, write SSICR.TEN = 1 or SSICR.REN = 1
3	The following three conditions are met: • SSICR.TEN = 1 or SSICR.REN = 1 • Have padding bits • The last bit of the data word has been transferred
4	The following two conditions are met: • SSICR.SDTA = 1 or no padding bit • When SSICR.TEN = 0 and SSICR.REN = 0, the last bit of the data word in a frame has been transmitted
5	When SSICR.TEN = 1 or SSICR.REN = 1, the last padding bit has been transferred
6	The following two conditions are met: • SSICR.SDTA = 0 and has padding bits • When SSICR.TEN = 0 and SSICR.REN = 0, the last fill bit has been transferred

18.6.2.1 Data communication status

The data communication status is the status that the SSIE is sending, receiving, or sending and receiving data according to the data word length set in SSICR.DWL[2:0].

- State transition without padding bits

During communication (SSISR.IIRQ = 0), the SSIE is always in the data communication state. By disabling transmit and receive (SSICR.TEN = 0, SSICR.REN = 0), the SSIE shifts to the idle state. See Figure 18-43 and Figure 18-44 for details.

Figure 18-43 Continuous data communication

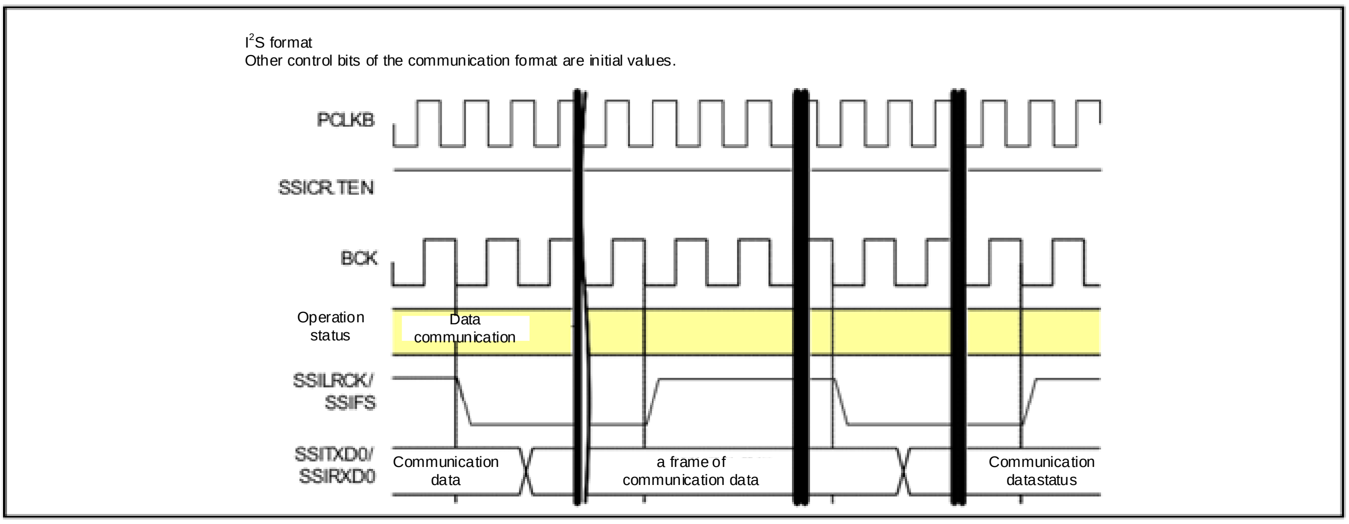
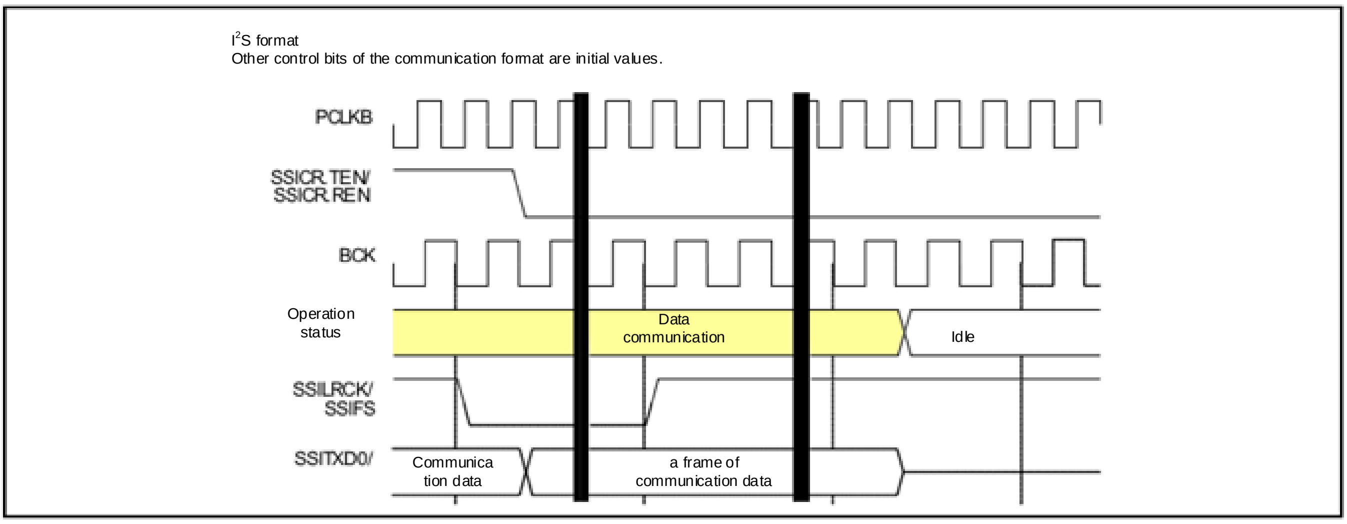


Figure 18-44 Data communication stop when there is no padding bit



- State transition with padding bits

When the SSIE ends the transmission of the last bit of the data word during communication (SSISR.IIRQ = 0), the SSIE transitions from the data communication state to the padding bit communication state, as shown in Figure 18-45. When SSICR.SDTA = 1 and transmission and reception are stopped (SSICR.TEN = 0 and SSICR.REN = 0), the SSIE stops communication and transitions from the fill communication state to the idle state, as shown in Figure 18-47.

Figure 18-45 Transition from data communication state to fill communication state

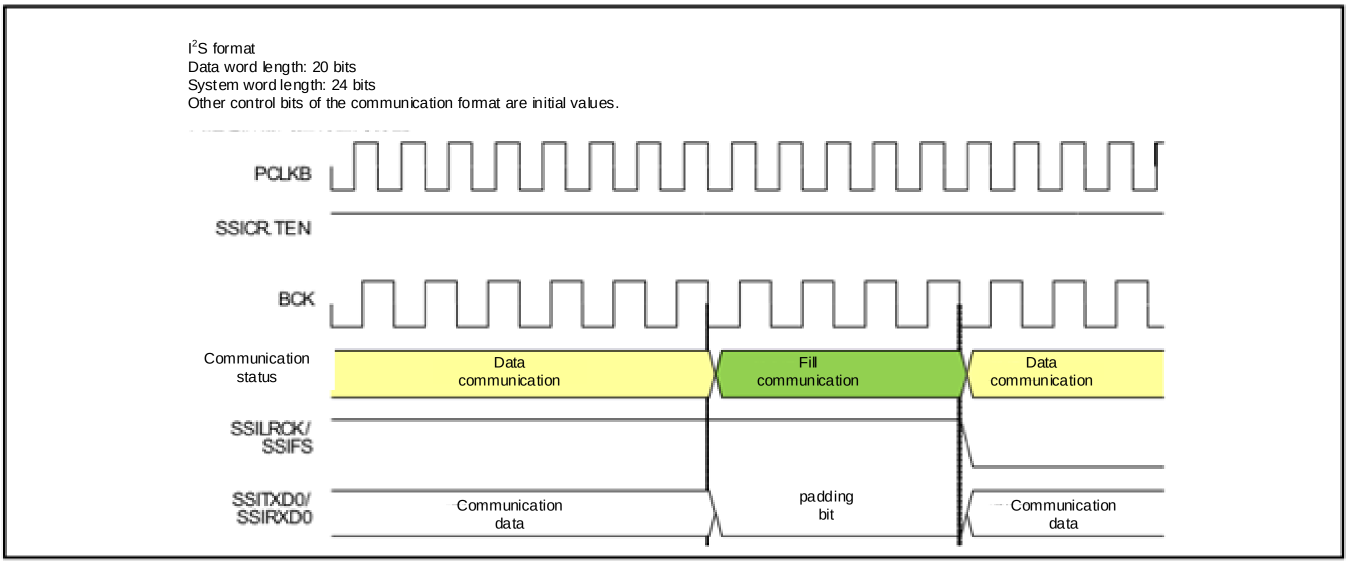
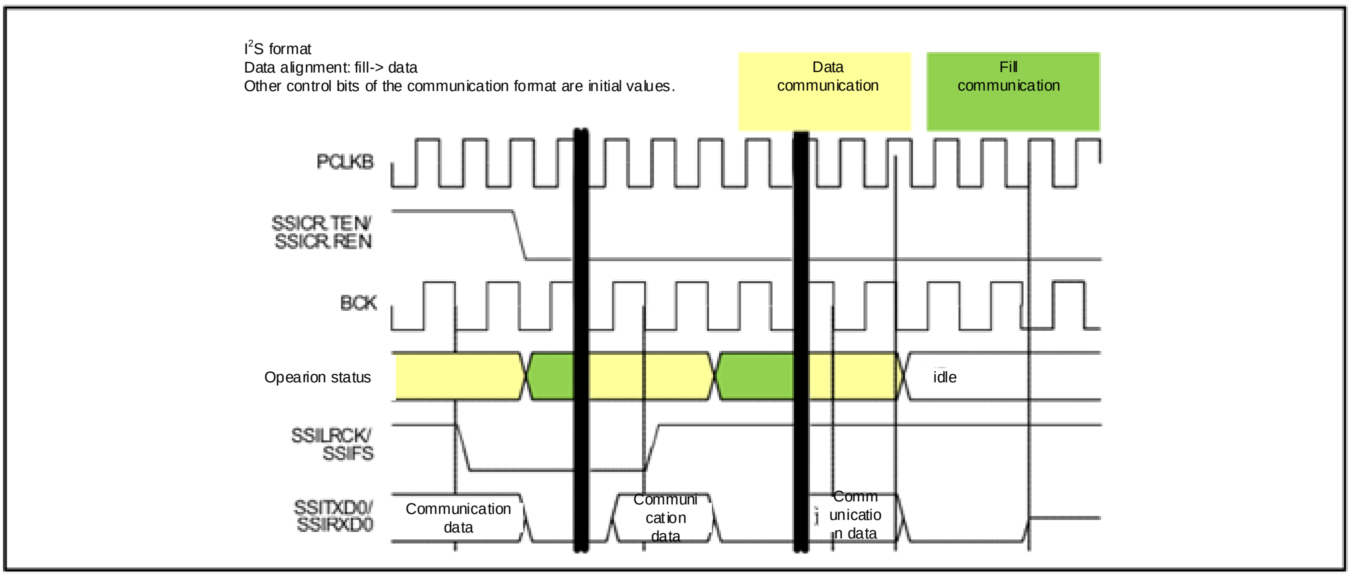


Figure 18-46 Stopping data communication with padding bits



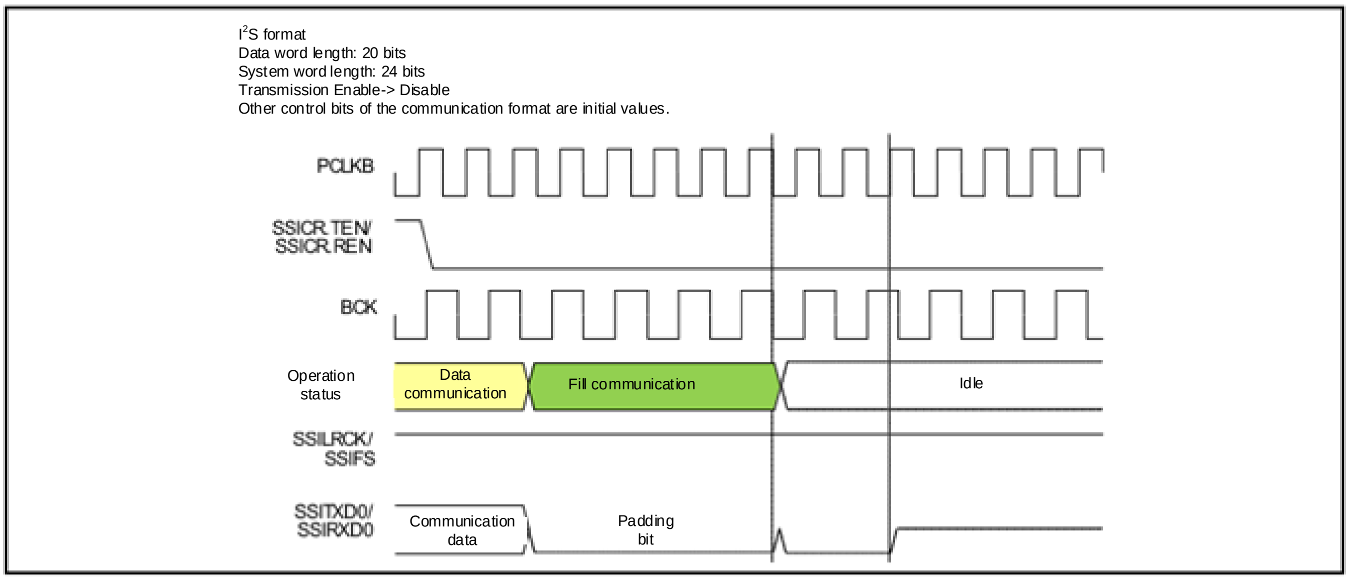
18.6.2.2 Fill communication status

The fill communication state is when the SSIE is sending, receiving, or sending and receiving filling data according to the padding bits set by SSICR.SWL[2:0] and SSICR.DWL[2:0].

- State migration with padding bits

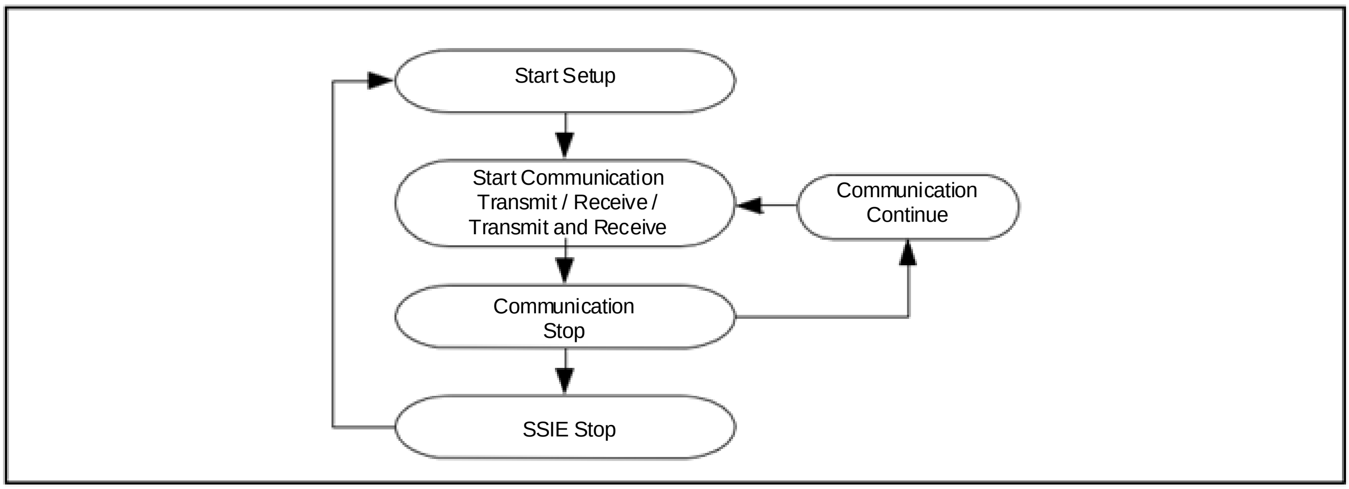
When the SSIE ends the transmission of the last fill bit during communication (SSISR.IIRQ = 0), the SSIE shifts to the data communication state. If SSICR.SDTA = 0 and transmit and receive are stopped (SSICR.TEN = 0 and SSICR.REN = 0), the SSIE shifts from the fill communication state to the idle state when it stops communicating.

Figure 18-47 Stopping from filling communication status



18.7 Communication operation

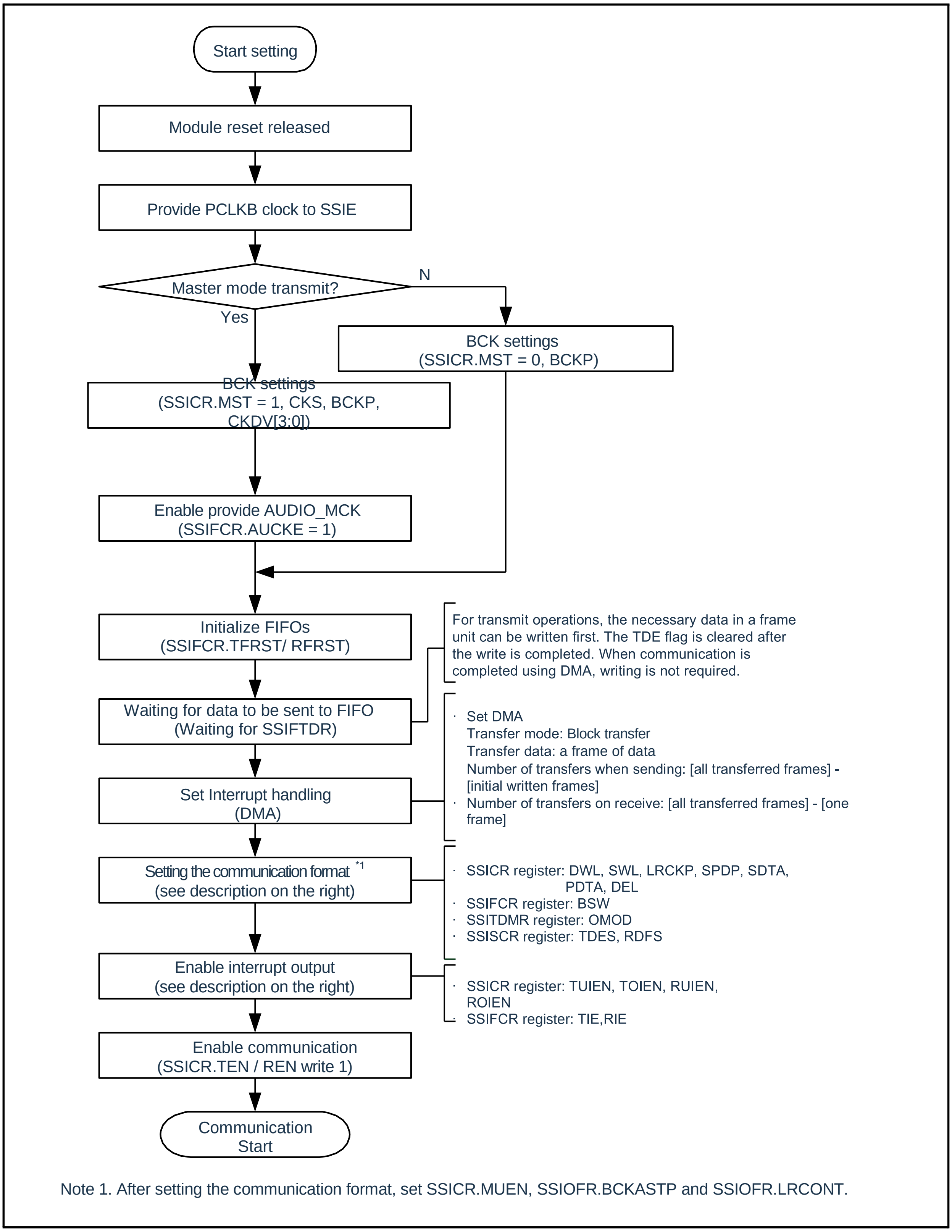
Figure 18-48 SSIE communication flow chart



18.7.1 Starting communication

This section describes how to start SSE communication. To start communication, be sure to follow the process as shown in Figure 18 49 Process of starting communication (CPU operation process).

Figure 18-49 Process of starting communication (CPU operation process)



For transmit operations, the necessary data in a frame unit can be written first. The TDE flag is cleared after the write is completed. When communication is completed using DMA, writing is not required.

- Set DMA
  Transfer mode: Block transfer
  Transfer data: a frame of data
  Number of transfers when sending: [all transferred frames] - [initial written frames]
- Number of transfers on receive: [all transferred frames] - [one frame]

- SSICR register: DWL, SWL, LRCKP, SPDP, SDTA, PDTA, DEL
- SSIFCR register: BSW
- SSITDMR register: OMOD
- SSISCR register: TDES, RDFS

- SSICR register: TUIEN, TOIEN, RUIEN, ROIEN
- SSIFCR register: TIE,RIE

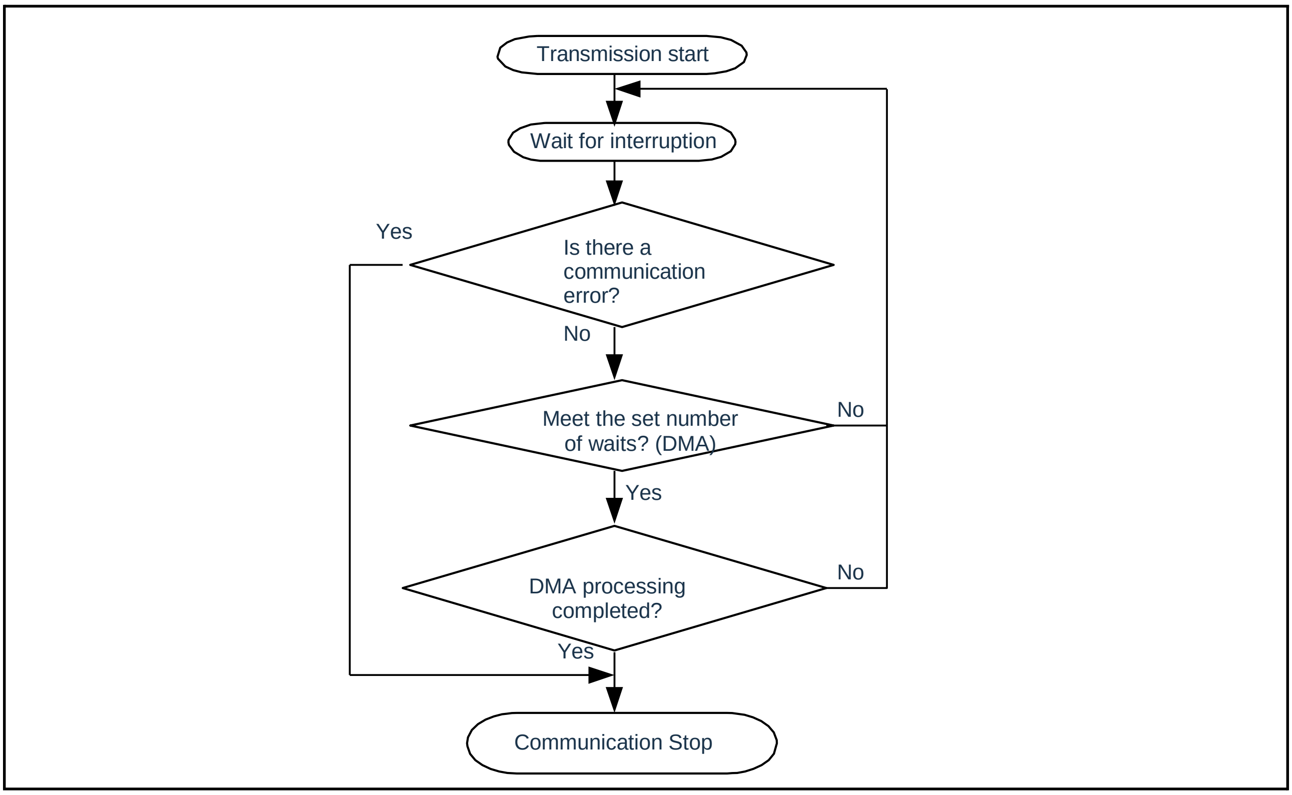
Note 1. After setting the communication format, set SSICR.MUEN, SSIOFR.BCKASTP and SSIOFR.LRCONT.

SSIE can perform continuous communication using interrupts of DMA. For sending, write "1" to SSIFCR.TIE, SSICR.TUIEN and SSICR.TOIEN. For receive, write "1" to SSIFCR.RIE, SSICR.RUIEN and SSICR.ROIEN.

18.7.2 Transmission

After transmitting is allowed (SSICR.TEN = 1 , SSICR.REN = 0), when the transmit FIFO data register (SSIFTDR) contains serial data for at least one frame, SSILRCK/SSIFS generates a start trigger signal and the SSIE starts transmitting. According to the condition set by the TDE (SSISCR.TDES) and the status of the transmit data null interrupt enable bit set in the communication start program (SSIFCR.TIE), the SSIE generates a transmit data null interrupt for requesting a write to the transmit FIFO data register (SSIFTDR). In the communication start program, the transmit data null interrupt is set to trigger the DMA to perform the write to the transmit FIFO data register (SSIFTDR). With this setting, SSIE can send data continuously without the CPU. When the available space size of the transmit FIFO data register reaches the value set in SSISCR.TDES, a transmit data null interrupt is generated. The number of data writes must be specified according to the free space size of the transmit FIFO data register set in the transmit data null interrupt. If an error occurs, perform the error handling procedure as described in the communication stop procedure. The flow in Figure 18-50 must be executed during the transmit action.

Figure 18-50 Transmission process

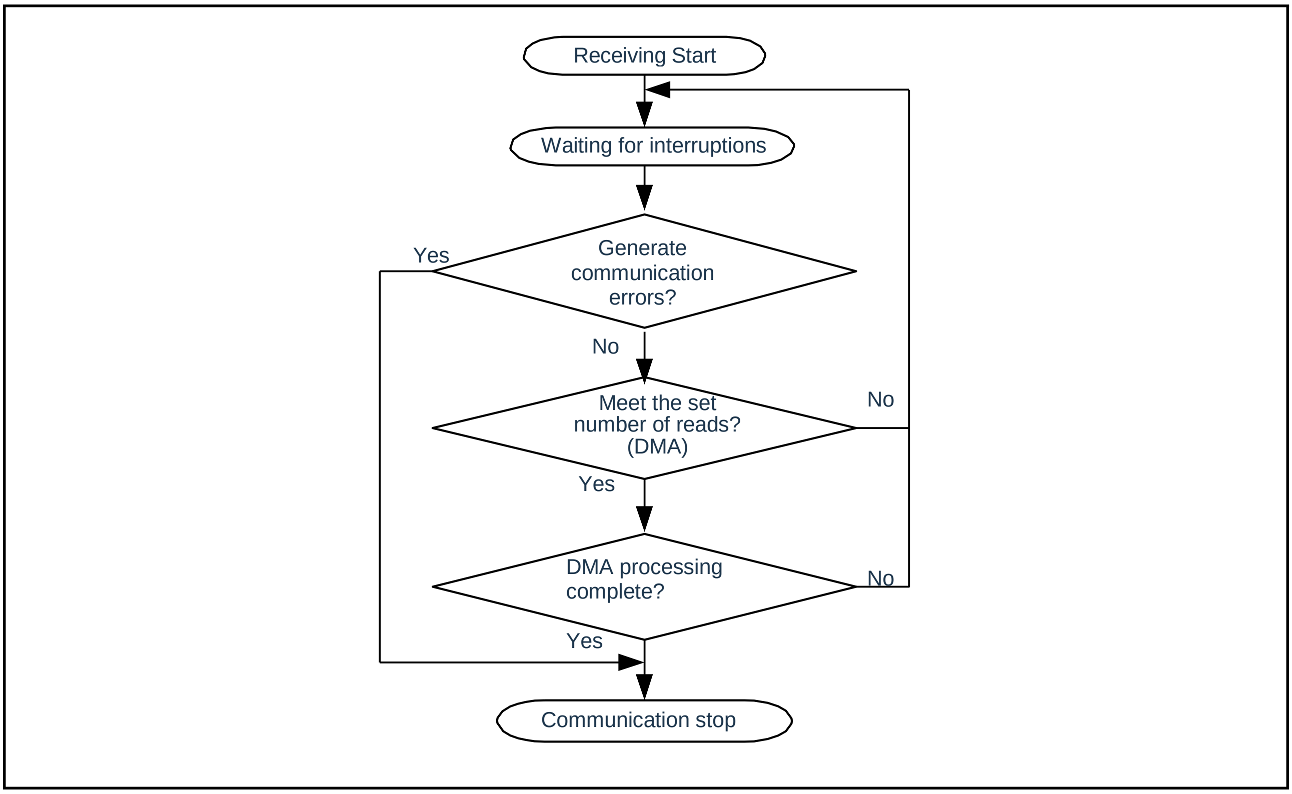

Transmission start
Wait for interruption
Yes
Is there a communication error?
No
Meet the set number of waits? (DMA)
No
Yes
DMA processing completed?
No
Yes
Communication Stop

Notice: This process uses DMA to complete the communication of SSIE. If DMA is not used, poll SSIFSR.TDE to write data to SSIFTDR when SSIFSR.TDE is 1. The number of times to write data to SSIFTDR when SSIFSR.TDE is detected as 1 must be set according to the available space size of the transmit FIFO data register specified in SSISCR.TDES. The SSIFSR.TDE flag must be cleared after writing the send data that matches the available space size to SSIFTDR. Continuous sending can be achieved by repeatedly writing data. If the SSIFSR.TDE flag is not cleared, the flag will not be cleared automatically.

18.7.3 Reception

After reception is allowed (SSICR.TEN = 0, SSICR.REN = 1), SSIE starts receiving when SSILRCK/SSIFS generates a start trigger. The SSIE outputs the receive data full interrupt to the DMA according to the condition of the RDF setting specified in the communication initiator (SSISCR.RDFS) and the state allowed by the receive data full interrupt (SSIFCR.RIE). This interrupt requests to read data from the Receive FIFO Data Register (SSIFRDR). In the communication initiator, specify that the receive data full interrupt triggers the DMA to read data from the receive FIFO data register (SSIFRDR). With this setting, the SSIE can read data continuously without the CPU. When data equal to the capacity of the receive FIFO data register is stored, a receive data full interrupt is generated. The number of data reads must be specified according to the size of the receive FIFO data register indicated by the receive data full interrupt. If an error occurs, the error handling procedure described in the communication stop program is executed. The flow in Figure 18-51 must be executed during the receive action.

Figure 18-51 Reception process


Receiving Start
Waiting for interruptions
Generate communication errors?
Yes
No
Meet the set number of reads? (DMA)
No
Yes
DMA processing complete?
No
Yes
Communication stop

Notice: This process uses DMA to complete the communication of SSIE. If DMA is not used, poll SSIFSR.RDE to read data from SSIFRDR when SSIFSR.RDE is 1. The number of times data is read from SSIFRDR when SSIFSR.RDF is detected as 1 must match the storage capacity of the receive FIFO data register specified in SSISCR.RDFS. After reading data from SSIFRDR, the SSIFSR.RDF flag bit must be cleared to zero, and continuous reception can be achieved by repeatedly reading data. If the SSIFSR.RDF flag is not cleared, the flag will not be cleared automatically.

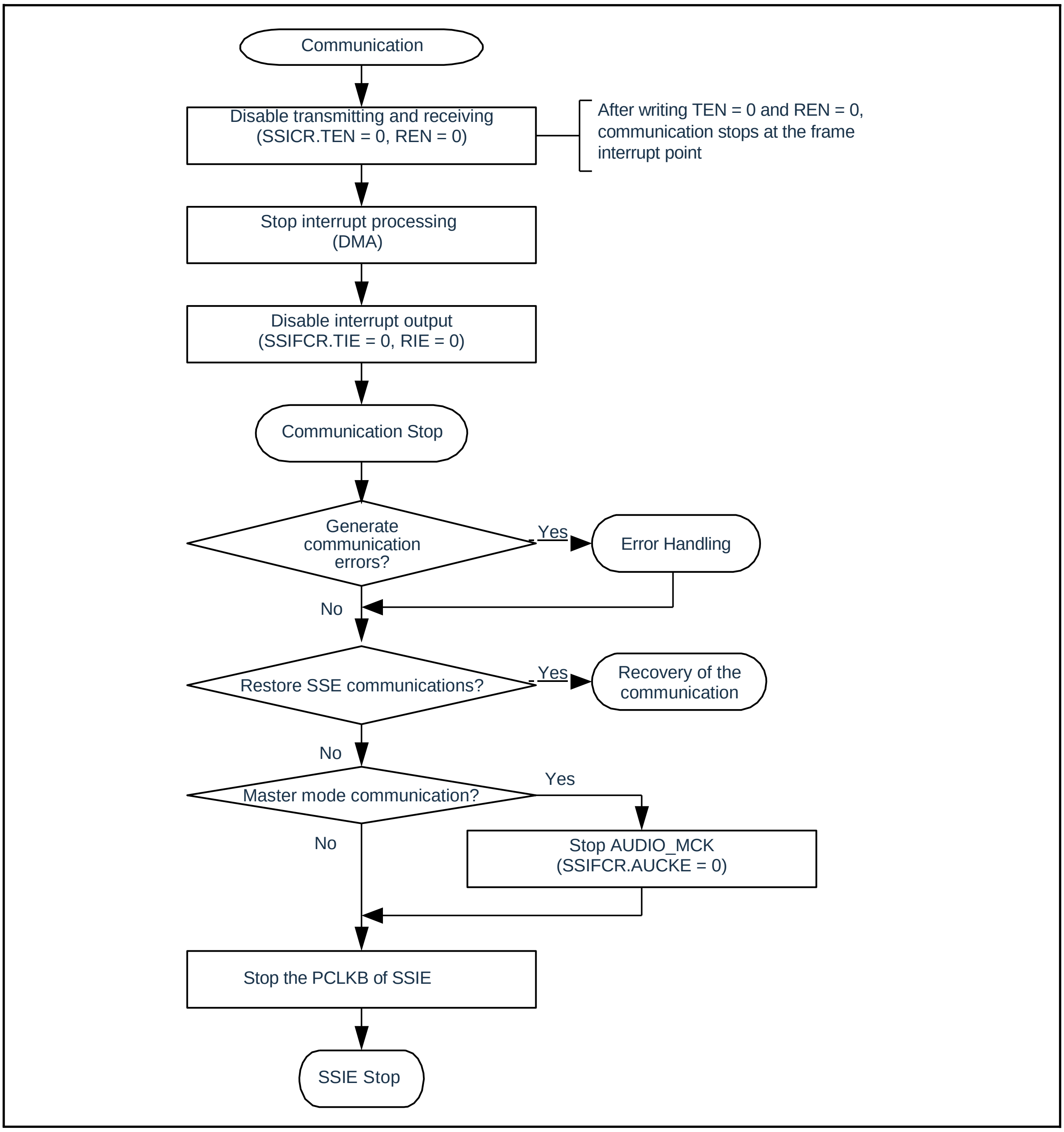
18.7.4 Transmission and reception

After transmit and receive are allowed (SSICR.TEN = 1, SSICR.REN = 1), the SSIE starts transmitting and receiving when the transmit FIFO data register (SSIFTDR) contains serial data for at least one frame, and SSILRCK/SSIFS generates a start trigger signal. The SSIE can continuously transmit and receive data by performing the processes described in Section 18.7.2, "Transmission" and Section 18.7.3, "Reception", respectively. See Section 18.7.5, "Stopping Communication" for information on how to stop transmitting and receiving.

18.7.5 Stopping communication

This section describes how to stop the SSE communication. Follow the flow shown below to stop communication.

Figure 18-52 Process of stopping communication (CPU operation)


Communication
Disable transmitting and receiving
(SSICR.TEN = 0, REN = 0)
After writing TEN = 0 and REN = 0, communication stops at the frame interrupt point
Stop interrupt processing
(DMA)
Disable interrupt output
(SSIFCR.TIE = 0, RIE = 0)
Communication Stop
Generate communication errors?
Yes
Error Handling
No
Restore SSE communications?
Yes
Recovery of the communication
No
Master mode communication?
Yes
Stop AUDIO_MCK
(SSIFCR.AUCKE = 0)
No
Stop the PCLKB of SSIE
SSIE Stop

To stop the SSE communication, the following clocks need to be provided until SSISR.IIRQ becomes idle:

- When SSICR.MST = 0, the clock input from the SSIBCK pin
- When SSICR.MST = 1, the AUDIO_MCK clock

To restore SSE communication from previous settings, see Section 18.7.7, "Restoring Communication".

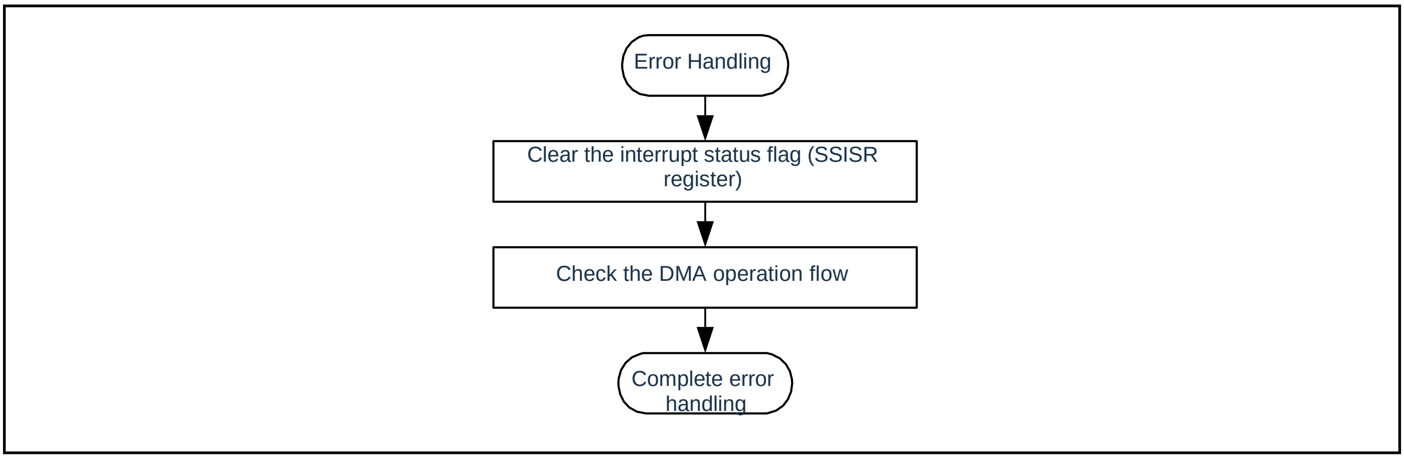
18.7.6 Error handling

SSIE has the following 4 types of errors.

- Transmit underflow error
- Transmit overflow error
- Receive underflow error
- Receive overflow error

When an underflow error or an overflow error is generated, the SSIE needs to be restarted. Follow the stop communication flow in and the error handling flow in Figure 18-53.

Figure 18-53 Error handling process


Error Handling
Clear the interrupt status flag (SSISR register)
Check the DMA operation flow
Complete error handling

The four error operations are described below. An error interrupt is generated when the SSICR register is set to allow interrupt output and an error flag is generated. Refer to the flag description in 18.3.2 for the conditions of error flag generation.

1) Transmit underflow error:

If a transmit underflow error occurs, check the number of times data is written to the transmit FIFO data register (SSIFTDR) triggered according to the transmit data null interrupt. After a transmit underflow error occurs, SSIE outputs 0 as data. To output serial data written to the Transmit FIFO Data Register (SSIFTDR) to the SSITXD0 pin normally, follow the Stop Communication procedure in and the error handler in Figure 18-53. Serial data is normally consumed after this error occurs. If communication is resumed, start writing serial data from the beginning.

2) Transmit overflow error:

If a transmit overflow error occurs, check the number of times data was written to the transmit FIFO data register (SSIFTDR) triggered by the transmit data null interrupt. The serial data written to the transmit FIFO data register (SSIFTDR) that caused the transmit overflow error is invalid. This error can occur whether or not a transmit operation is being performed. To recover from the error, follow the Stop Communication procedure in and the error handling procedure in Figure 18-53. When resuming communication, pay attention to handle the invalid serial data.

3) Receive underflow error:

If a receive underflow error occurs, check the number of reads from the receive FIFO data register (SSIFRDR) triggered by the receive data full interrupt. The value read from the receive FIFO data register (SSIFRDR) that causes the receive underflow error to be generated is uncertain. This error can occur whether or not a receive operation is being performed. To recover from the error, follow the Stop Communication procedure in and the error handling procedure in Figure 18-53.

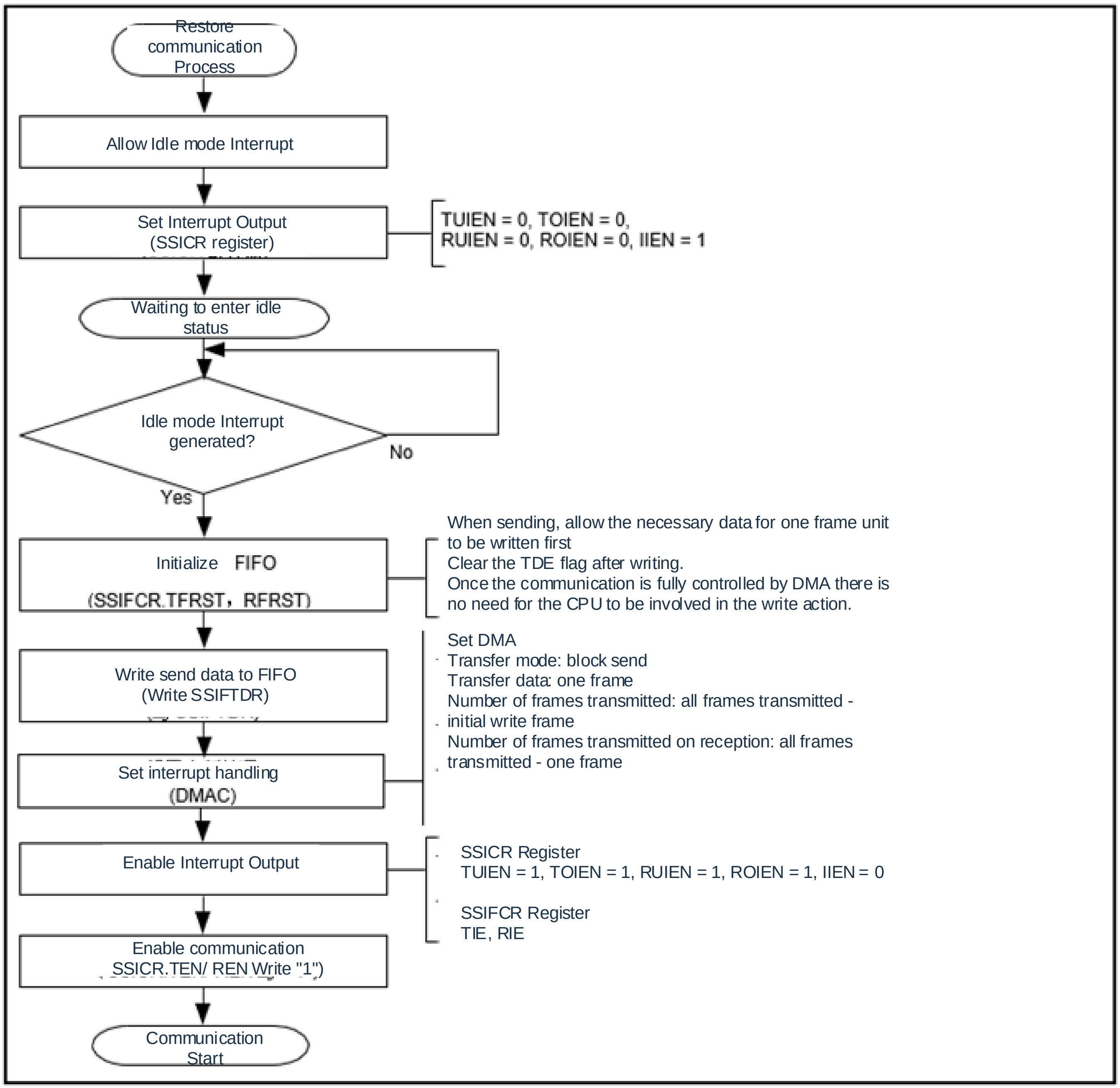
4) Receive overflow error

If a receive overflow error occurs, check the number of data reads from the receive FIFO data register (SSIFRDR) triggered by the receive data full interrupt. The receive data that caused the receive overflow error to occur cannot be stored in the receive FIFO data register (SSIFRDR). To recover from the error, follow the Stop Communication procedure in and the error handler in Figure 18-53.

18.7.7 Restoring communication

To resume SSE communications, follow the communications recovery process in Figure 18-54. This procedure assumes that communication that was stopped by the communication stop process is restored without changing any settings. To change the clock and slave/master settings, follow the communication start procedure in Figure 18-49. See Section 18.7.2 "Transmission" and Section 18.7.3 "Reception" for more information on the transmit and receive operations after communication has started, respectively.

Figure 18-54 Flow of restoring communication (CPU operation)


Restore communication Process
Allow Idle mode Interrupt
Set Interrupt Output (SSICR register)
TUIEN = 0, TOIEN = 0,
RUIEN = 0, ROIEN = 0, IIEN = 1
Waiting to enter idle status
Idle mode Interrupt generated?
No
Yes
Initialize FIFO (SSIFCR.TFRST, RFRST)
When sending, allow the necessary data for one frame unit to be written first
Clear the TDE flag after writing.
Once the communication is fully controlled by DMA there is no need for the CPU to be involved in the write action.
Write send data to FIFO (Write SSIFTDR)
Set DMA
Transfer mode: block send
Transfer data: one frame
Number of frames transmitted: all frames transmitted - initial write frame
Number of frames transmitted on reception: all frames transmitted - one frame
Set interrupt handling (DMAC)
Enable Interrupt Output
SSICR Register
TUIEN = 1, TOIEN = 1, RUIEN = 1, ROIEN = 1, IIEN = 0
SSIFCR Register
TIE, RIE
Enable communication SSICR.TEN/ REN Write "1")
Communication Start

18.8 Interrupt

The following table lists the interrupt sources. the TUIEN, TOIEN, RUIEN, ROIEN, and IIEN bits in the SSICR register and the TIE and RIE bits in the SSIFCR register are used to enable or disable the interrupt output for each source.

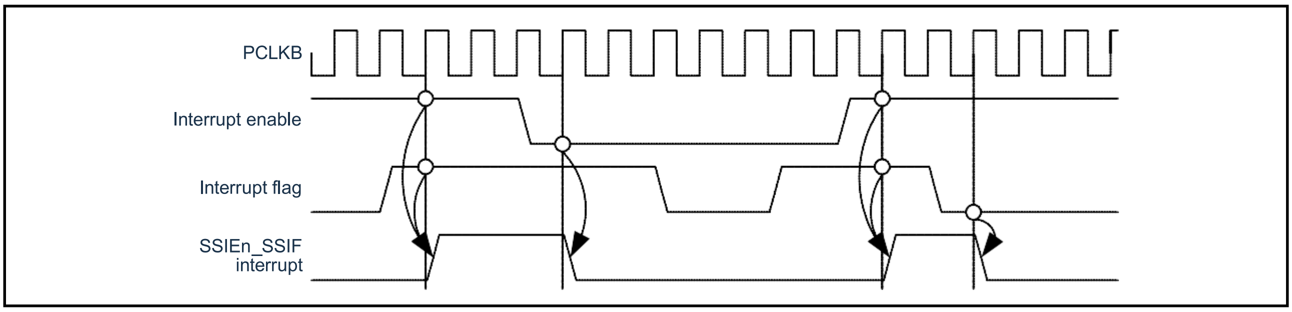
Table 18-16 SSIE interrupt source

Channel	Interrupt source	Description	Interruption flags	DMA Trigger
SSIE0	SSIE0_SSIF	• Transmit underflow interrupt • Transmit overflow interrupt • Receive underflow interrupt • Receive overflow interrupt • Idle interrupt	SSISR.TUIRQ SSISR.TOIRQ SSISR.RUIRQ SSISR.ROIRQ SSISR.IIRQ	No
	SSIE0_SSIRXI	Receive data full interrupt	SSIFSR.RDF	Yes
	SSIE0_SSITXI	Transmit data null interrupt	SSIFSR.TDE	Yes

18.8.1 SSIE0_SSIF interrupt

This interrupt source combines five interrupts. These five interrupts are controlled and acted upon by their respective output enable bits and flag bits. Set the output enable for the desired interrupt before using SSIE. To clear the interrupts, write 0 to the interrupt enable bit or write 0 to the interrupt flag bit.

Figure 18-55 Timing diagram of SSIEn_SSIF interrupt



- Transmit underflow interrupt

  SSISR.TUIRQ is used as the transmit underflow interrupt, and the output of this interrupt is allowed when SSICR.TUIEN = 1.

- Transmit overflow interrupt

  SSISR.TOIRQ is used as the transmit overflow interrupt, and the output of this interrupt is allowed when SSICR.TOIRQ = 1.

- Receive underflow interrupt

  SSISR.RUIRQ is used as the receive underflow interrupt, and the output of this interrupt is allowed when SSICR.RUIRQ = 1.

- Receive overflow interrupt

  SSISR.ROIRQ is used as the receive overflow interrupt and the output of this interrupt is allowed when SSICR.ROIRQ = 1.

- Idle mode interrupt

  SSISR.IIRQ is used as an idle mode interrupt and this interrupt output is allowed when SSICR.IIEN = 1.

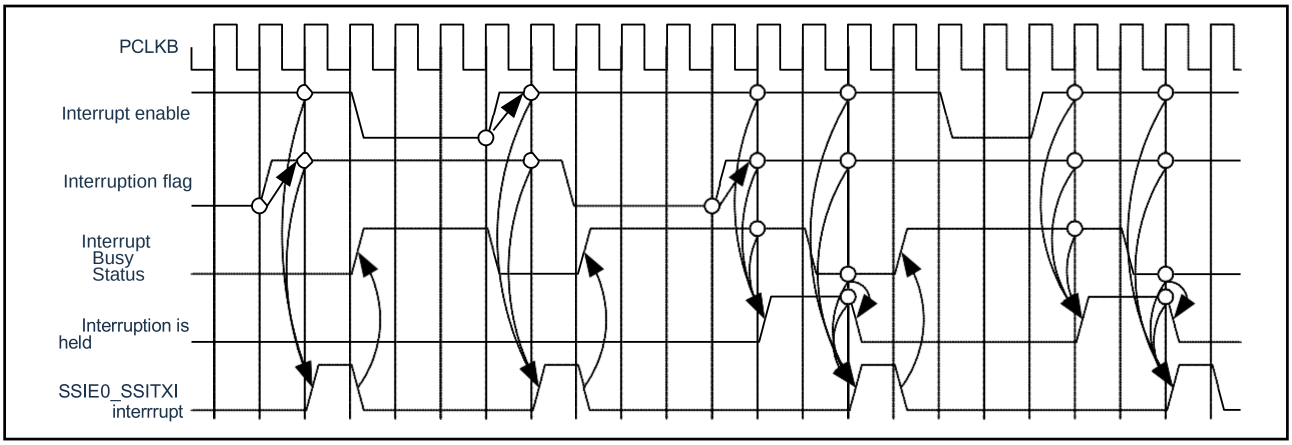
18.8.2 SSIE0_SSITXI interrupt (full duplex communication)

A transmit data null interrupt is an interrupt pulse that is output when the following conditions are met:

- SSIFCR.TIE = 1, SSIFSR.TDE = 1
- SSIE Action: When the value of SSIFCR.TIE is 1 and the value of SSIFSR.TDE changes from 0 to 1
- CPU instruction: When the value of SSIFSR.TDE is 1 and the value of SSIFCR.TIE changes from 0 to 1

This interrupt is controlled by the interrupt pending function. If this interrupt is generated when the DMA is busy and cannot handle a new interrupt, the interrupt pending function will maintain the interrupt output and output it to the DMA for processing when the DMA can receive the interrupt.

Figure 18-56 Interrupt timing of SSIE0_SSITXI



18.8.3 SSIE0_SSIRXI interrupt (full duplex communication)

The receive data full interrupt is an interrupt pulse that is output when the following conditions are met:

- SSIFCR.RIE = 1, SSIFSR.RDF = 1.
- SSIE action: When SSIFCR.RIE is "1", the value of SSIFSR.RDF changes from 0 to 1
- CPU instruction: When SSIFSR.RDE is "1", the value of SSIFCR.RIE changes from 0 to 1.

This interrupt is controlled by the interrupt pending function. If this interrupt is generated when the DMA is busy and cannot handle a new interrupt, the interrupt pending function will maintain the interrupt output and output it to the DMA for processing when the DMA can receive the interrupt.

18.9 Soft reset

SSIE has 3 soft reset control bits:

- SSIE Soft Reset (SSIFCR.SSIRST)
- Transmit FIFO data register reset (SSIFCR.TFRST)
- Receive FIFO data register reset (SSIFCR.RFRST).

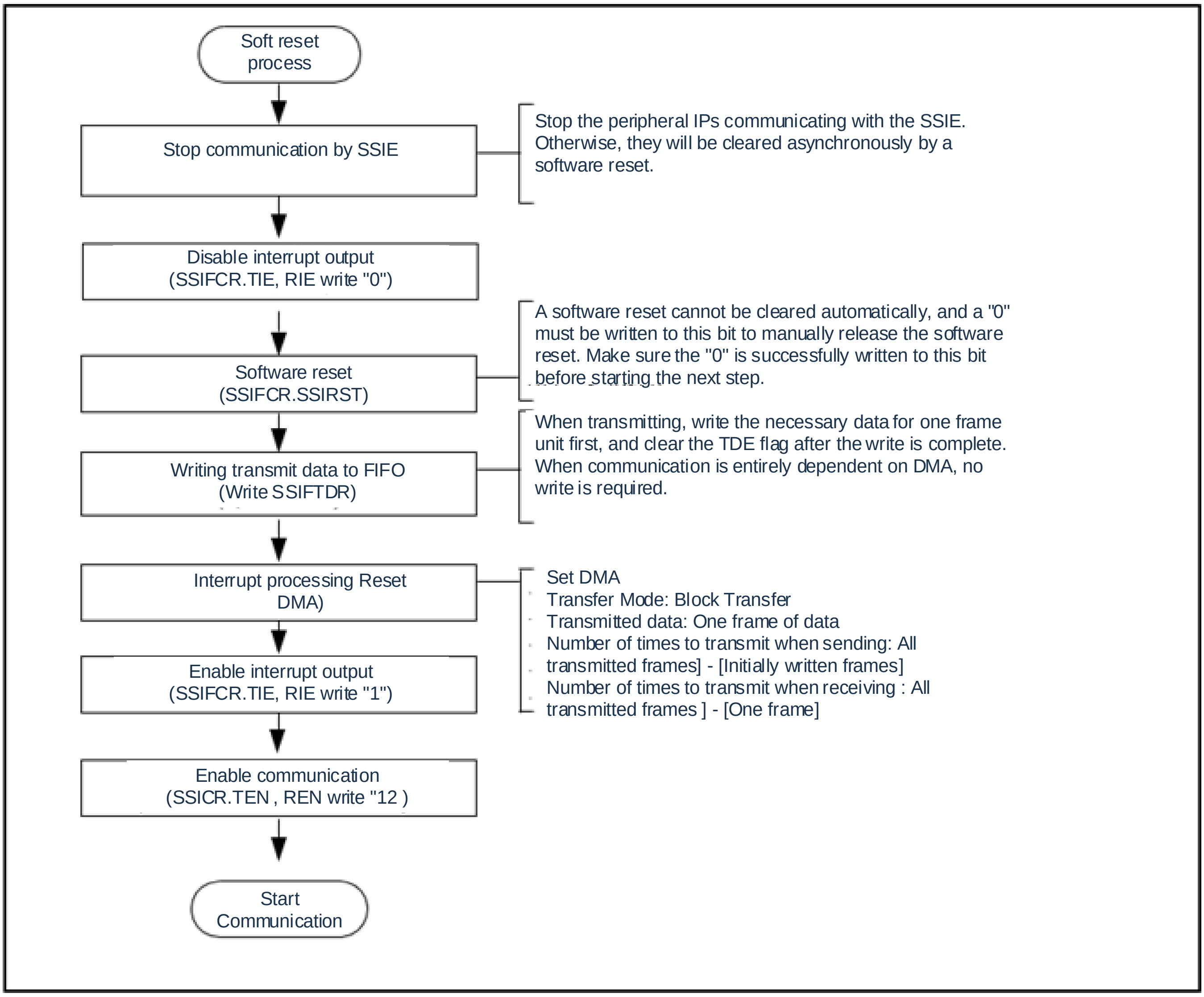
This section describes the flow of these three types of soft resets.

18.9.1 Soft reset process

1) SSIE soft reset

SSIFCR.SSIRST serves as the software reset control bit for the SSIE and follows the flow shown in Figure 18-57. To change the clock and slave/master mode settings, initiate communication according to the procedure in Figure 18-49. When communication is resumed, see Section 18.7.2 "Transmission" and Section 18.7.3 "Reception" for transmitting and receiving, respectively.

Figure 18-57 Software reset process (CPU operation)


Soft reset process
Stop communication by SSIE
Stop the peripheral IPs communicating with the SSIE. Otherwise, they will be cleared asynchronously by a software reset.
Disable interrupt output (SSIFCR.TIE, RIE write "0")
A software reset cannot be cleared automatically, and a "0" must be written to this bit to manually release the software reset. Make sure the "0" is successfully written to this bit before starting the next step.
Software reset (SSIFCR.SSIRST)
When transmitting, write the necessary data for one frame unit first, and clear the TDE flag after the write is complete. When communication is entirely dependent on DMA, no write is required.
Writing transmit data to FIFO (Write SSIFTDR)
Interrupt processing Reset DMA)
Set DMA
Transfer Mode: Block Transfer
Transmitted data: One frame of data
Number of times to transmit when sending: All transmitted frames] - [Initially written frames]
Number of times to transmit when receiving : All transmitted frames] - [One frame]
Enable interrupt output (SSIFCR.TIE, RIE write "1")
Enable communication (SSICR.TEN , REN write "12)
Start Communication

2) Transmit FIFO data register soft reset

To perform a soft reset of the transmit FIFO data register, follow the process in Figure 18-49 to initiate communication and follow the process in Figure 18-54 to resume communication.

3) Receive FIFO data register soft reset

To perform a soft reset of the Receive FIFO Data Register, follow the procedure in Figure 18-49 to initiate communication and follow the procedure in Figure 18-54 to resume communication.

18.10 Cautions

18.10.1 Cautions for slave mode communication

18.10.1.1 ADCKE control

SSIE requires SSIBCK when communicating in slave mode (SSICR.MST=0.) To stop BCK on the master, ensure that SSIE is idle (SSISR.IIRQ=1). If you stop BCK before the SSIE becomes idle, start communication using the procedure shown in Figure 18-49, or wait for the idle state using the procedure shown in Figure 18-54 to resume communication.

18.10.1.2 SSILRCK/SSIFS pin

The SSILRCK/SSIFS pins are used for synchronization of communication. When the SSIE is in slave mode (SSICR.MST=0), the SSIE communication format must match the communication format of the other party's device. The SSIE uses the signal input from the SSILRCK/SSIFS pin as the trigger signal to initiate communication.

18.10.2 Cautions for master mode communication

18.10.2.1 ADCKE control

When communicating in master mode (SSICR.MST=1), the SSIE operates with the audio clock (AUDIO_MCK). To stop SSIE completely, make sure SSIE is idle (SSISR.IIRQ=1), then write 0 to SSIFCR.ADCKE.

18.10.2.2 LRCONT control

When SSIE is in master idle mode (SSICR.MST=1), the SSITDMR.LRCONT bit is changed from 1 to 0 while ensuring that the other side of the device is not affected, at which point the SSILRCK/SSIFS signal output to the pin is stopped. For more information, see Figure 18-40.

18.10.2.3 BCKASTP control

When the SSIE is in master idle mode (SSICR.MST=1), the SSITDMR.BCKASTP bit is changed from 0 to 1 while ensuring that the other side of the device is not affected, at which point the SSIBCK signal output to the pin is stopped. For more information, see Figure 18-41.

The BCKASTP bit cannot be used to stop the clock when the other party device (slave) requests a clock from the SSIBCK pin both before and during communication.

18.10.3 Cautions for communication process

18.10.3.1 When an error interrupt is generated

SSIE has four types of errors.

- Transmit underflow error
- Transmit overflow error
- Receive underflow error
- Receive overflow error

SSIE needs to be restarted when a underflow error or an overflow error occurs. Follow the flow to stop communication in Figure 18-52 and the error handling flow in Figure 18-53.

18.10.3.2 Transmit data null interrupt

DMA is recommended for the transmitting process of SSIE. If DMA is not used, poll SSIFSR.TDE to "1" and write data to SSIFTDR after "1" is detected. The number of times the data is written to SSIFTDR must match the size of the available space in the transmit FIFO data register set by SSIFSR.TDES. The SSIFSR.TDE flag needs to be cleared manually after writing data that matches the available space size to SSIFTDR, and the SSIFSR.TDE flag will not be cleared automatically. Continuous transmission is achieved by repeatedly writing data.

18.10.3.3 Receive data full interrupt

DMA is recommended for the reception process of SSIE. If DMA is not used, the SSIFSR.RDF is polled to "1" and data is read from SSIFRDR after "1" is detected. The number of reads from SSIFRDR must match the data storage capacity of the received FIFO data register set by SSIFSR.RDFS. After reading the received data from SSIFRDR, you need to clear the SSIFSR.RDF flag manually, and the SSIFSR.RDF flag will not be cleared automatically. Continuous reception is achieved by repeatedly reading the data.

18.10.3.4 Transmission mode conversion

The conversion mode can be switched as follows.

1. For the state transition between transmit/receive/transmit and receive, first disable transmit and receive (SSICR.TEN = 0, SSICR.REN = 0)
2. Verify that SSIE is idle (SSISR.IIRQ = 1)
3. In the idle state, set the SSICR.TEN bit and SSICR.REN bit again to resume communication

18.10.3.5 Resuming communication after SSIE is stopped

When the SSE communication is stopped according to the process shown in Figure 18-52, the communication is resumed according to the process shown in Figure 18-54.

18.10.4 Write access restrictions

18.10.4.1 SSICR register

If you rewrite the TEN bit or REN bit, make sure that the SSISR.IIRQ bit is in the desired state. Otherwise, if the value of the TEN bit or REN bit is changed by rewriting, the subsequent operation is unpredictable. For example, check that SSISR.IIRQ is 0 when transmitting or receiving is enabled and 1 when transmitting or receiving is disabled.

The TEN and REN bits enable or disable transmitting and receiving. When "1" is written to one of these bits, the corresponding communication operation is initiated with the SSILRCK/SSIFS signal trigger. For more information, see Section 18.7.2, "Transmission", Section 18.7.3, "Reception", and Section 18.7.4, " Transmission and Reception ".

When 0 is written to one of the bits, the current communication operation stops at the next frame boundary. To use SSIE for both transmit and receive, write 1 to both bits, and to stop all SSIE transmit and receive communications, write a 0 to the TEN and REN bits

18.10.4.2 SSISR register

Clear TUIRQ and TOIRQ

After communication is enabled (by changing the SSICR.TEN bit from 0 to 1), the transmit error flags TOIRQ and TUIRQ in the SSISR register are cleared.

Clear RUIRQ and ROIRQ

After communication is enabled (by changing the SSICR.REN bit from 0 to 1), the receive error flags RUIRQ and ROIRQ in the SSISR register are cleared.

18.10.4.3 Communication status

Bits with shaded areas in Table 18-17 are not allowed to be written. If these bits are written, subsequent operations performed after writing are not guaranteed.

Table 18-17 Non-rewritable bits during communication

Symbol	Address (BASE+)		+0								+1							
			31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSICR	00h	+0	—	CKS	TUI EN	TOI EN	RUI EN	ROI EN	IIEN	—	—	—	DWL[2:0]			SWL[2:0]		
		+2	—	MS T	BCK P	LRC KP	SPD P	SDT A	PDT A	DEL	CKDV[3:0]				MU EN	—	TEN	RE N
SSISR	04h	+0	—	—	TUI RQ	TOI RQ	RUI RQ	ROI RQ	IIRQ	—	—	—	—	—	—	—	—	—
		+2	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
SSIFCR	10h	+0	AUC KE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	SSI RST
		+2	—	—	—	—	BS W	—	—	—	—	—	—	—	TIE	RIE	TFR ST	RFR ST
SSIFSR	14h	+0	—	—	—	—	TDC[3:0]				—	—	—	—	—	—	—	TDE
		+2	—	—	—	—	RDC[3:0]				—	—	—	—	—	—	—	RDF
SSIFTDR	18h	+0	FTDR[31:16]															
		+2	FTDR[15:0]															
SSIFRDR	1ch	+0	FRDR[31:16]															
		+2	FRDR[15:0]															
SSITDMR	20h	+0	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
		+2	—	—	—	—	—	—	BCK AST P	LRC ON T	—	—	—	—	—	—	OMOD[1:0]	
SSISCR	24h	+0	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
		+2	—	—	—	—	—	TDES[2:0]			—	—	—	—	—	RDFS[2:0]		

Chapter 19 Serial Interface IICA

19.1 Function of serial interface IICA

This product is equipped with two serial interfaces IICA0 and IICA1, and has the following three modes.

19.1.1 Idle mode

This is a mode for non-serial transfer, which reduces power consumption.

19.1.2 I^{2}C bus mode (multi-master capable)

This mode communicates 8-bit data with multiple devices through two lines of the serial clock (SCLAn) and the serial data bus (SDAAn). The master device can generate start condition, address, indication of transmission direction, data and stop condition for slave devices. The slave device automatically detects the received status and data through the hardware. This feature simplifies the I2C bus control section of the application.

Because the SCLAn and SDAAn pins of the serial interface IICA are used as open-drain outputs, pull-up resistors are required for the serial clock line and serial data bus. The deep sleep mode can be released by generating an interrupt request signal (INTIICAn) when an extension code or local station address is received from the master device in the sleep mode. It is set by the WUPn bit of the IICA control register n1 (IICCTLn1).

The block diagram of the serial interface IICA is shown in Figure 19-1.

Remark: n=0,1

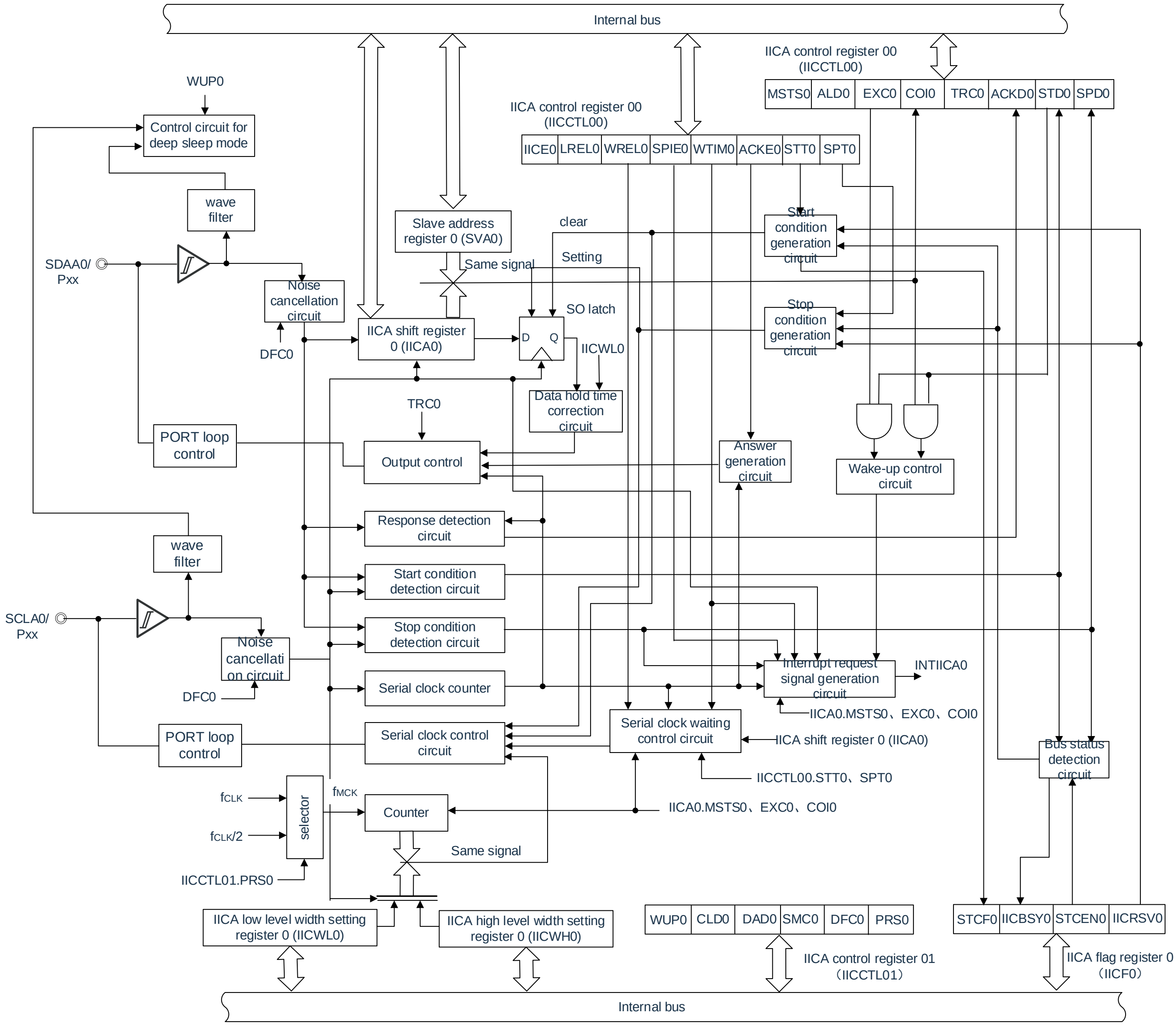
19.1.3 Wake-up mode

In a deep sleep mode, when an extension code or a local station address from a master control device is received, the deep sleep mode can be canceled by generating an interrupt request signal (INTIICAn). Set by the WUPn bit of the IICA control register n1 (IICCTLn1).

The block diagram of the serial interface IICA is shown in Figure 19-1.

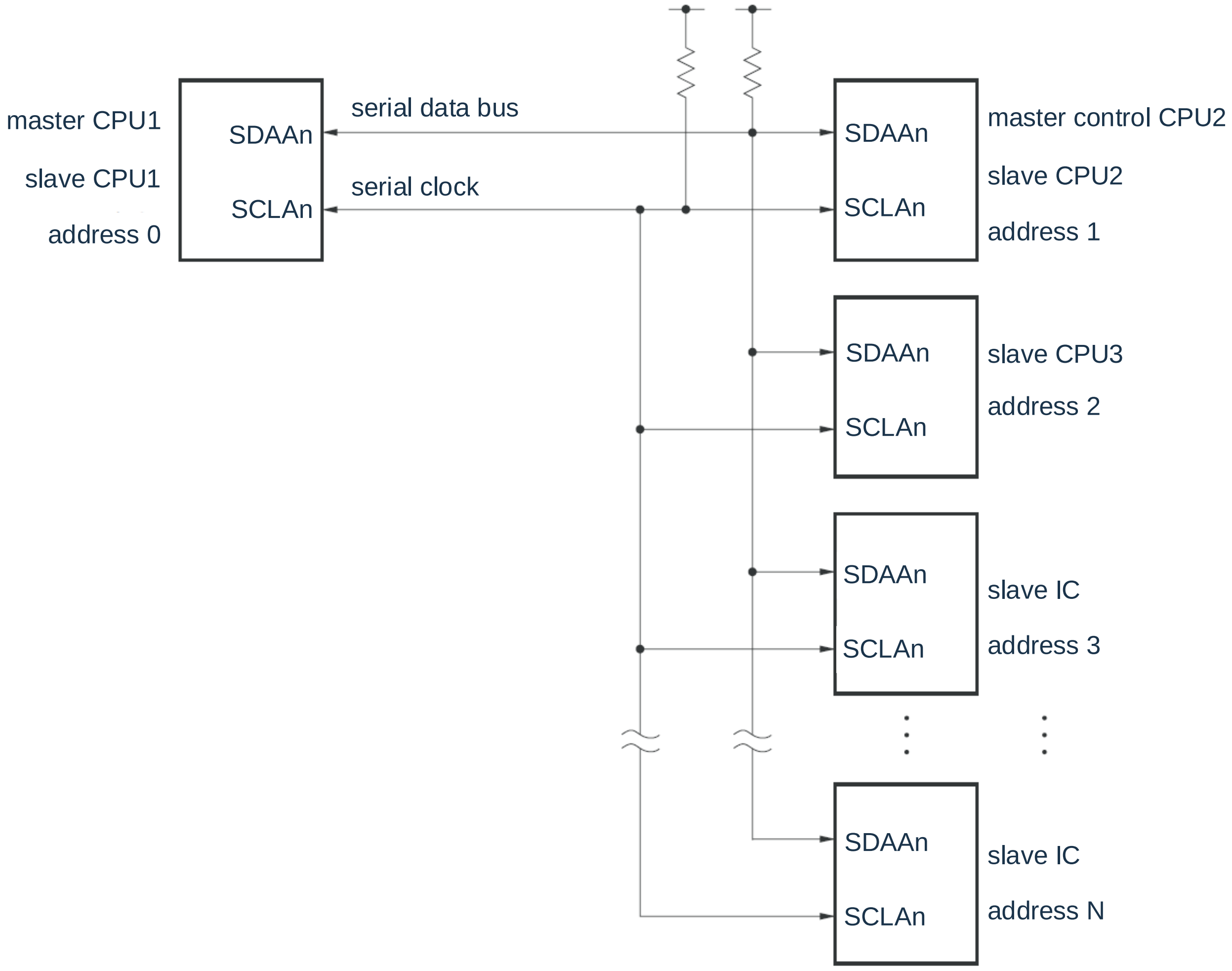
Remark: n=0,1

Figure 19-1 Block diagram of serial interface IICA


Internal bus
IICA control register 00
(IICCTL00)
MSTS0
ALD0
EXC0
COI0
TRC0
ACKD0
STD0
SPD0
WUP0
Control circuit for deep sleep mode
IICA control register 00
(IICCTL00)
IICE0
LREL0
WREL0
SPIE0
WTIM0
ACKE0
STT0
SPT0
wave filter
Slave address register 0 (SVA0)
clear
Start condition generation circuit
SDAA0/ Pxx
Same signal
Setting
Noise cancellation circuit
SO latch
Stop condition generation circuit
IICA shift register 0 (IICA0)
D
Q
DFC0
IICWL0
Data hold time correction circuit
TRC0
PORT loop control
Output control
Answer generation circuit
Wake-up control circuit
Response detection circuit
wave filter
Start condition detection circuit
SCLA0/ Pxx
Stop condition detection circuit
Noise cancellation circuit
Interrupt request signal generation circuit
INTIICA0
DFC0
Serial clock counter
IICA0.MSTS0、EXC0、COI0
Serial clock waiting control circuit
PORT loop control
Serial clock control circuit
IICA shift register 0 (IICA0)
Bus status detection circuit
IICCTL00.STT0、SPT0
fMCK
fCLK
selector
Counter
IICA0.MSTS0、EXC0、COI0
fCLK/2
Same signal
IICCTL01.PRS0
IICA low level width setting register 0 (IICWL0)
IICA high level width setting register 0 (IICWH0)
WUP0
CLD0
DAD0
SMC0
DFC0
PRS0
STCF0
IICBSY0
STCEN0
IICRSV0
IICA control register 01
（IICCTL01）
IICA flag register 0
（IICF0）
Internal bus

Examples of serial bus structures are shown in Figure 19-2.

Figure 19-2 Example of serial bus structure for I²C bus



Remark: n=0,1

19.2 Structure of serial interface IICA

The serial interface IICA consists of the following hardware.

Table 19-1 Structure of Serial Interface IICA

Item	Structure
Register	IICA shift register n (IICAn) slave address register n (SVAn).
Control register	Peripheral Enable Register 0 (PER0) IICA control register n0 (IICCTLn0) IICA status register n (IICSn) IICA flag register n (IICFn) IICA control register n1 (IICCTLn1) IICA low level width set register n (IICWLn) IICA high level width set register n (IICWHn) Port mode register (PMxx) Port mode control register (PMCxx) Port multiplexing function configuration register (PxxCFG)

Remark: 1. n=0,1

2. This product can reuse IICA input/output pin functions to multiple ports. When a port is configured as the multiplexing function of IICA pin, the N-channel drain open circuit output (V_{DD}/EV_{DD} withstand voltage) mode of the port is automatically opened by the design guarantee, that is, the POMxx register does not need to be set by the user.

Register list:

Base address	Offset address	Register name	R/W	Reset value
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

19.2.1 IICA shift register n (IICAn)

The IICAn register is a register for transmitting and receiving 8-bit serial data and 8-bit parallel data interconversion synchronously with the serial clock. The actual transmission and reception can be controlled by reading and writing the IICAn register.

During the waiting period, the waiting is lifted by writing the IICAn register to start transferring data. The IICAn register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "00H".

Figure 19-3 Format of 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, no data can be written to IICAn registers.

2. IICAn registers can only be read and written during the wait period. Access to the IICAn register is prohibited in the communication state except for the waiting period. However, in the case of a master device, the IICAn register can be written once after setting the communication trigger bit (STTn) to "1".

3. When making a reservation for communication, data must be written to the IICAn register after detecting an interrupt caused by a stop condition.

Remark: n=0, 1

19.2.2 Slave address register n (SVAn)

This is a register that holds the 7-bit local station address {A6, A5, A4, A3, A2, A1, A0} when used as a slave.

The SVAn register is set by an 8-bit memory operation instruction. However, this register is not allowed to be overwritten when the STDn bit is "1".

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

Figure 19-4 Format of the slave address register n(SVAn)

After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
SVAn	A6	A5	A4	A3	A2	A1	A0	0 Notes

Note: Bit0 is fixed "0".

19.2.3 SO latch

The SO latch holds the output level of the SDAAn pin.

19.2.4 Wake-up control circuit

The circuit generates an interrupt request (INTIICAn) when the address value set in the slave address register n (SVAn) is the same as the received address.

19.2.5 Serial clock counter

During the sending and receiving process, the counter counts the output or input serial clock to check whether the 8-bit data is sent or received.

19.2.6 Interrupt request signal generate circuit

This circuit controls the generation of an interrupt request signal (INTIICAn).

The I^{2}C interrupt request is generate by that following two trigger.

- Drop of the 8th or 9 th serial clock (set by the WTIMn bit)
- Interrupt request (set by SPIEn bit) due to detection of stop condition.

Remark: WTIMn bit: bit3 for IICA control register n0(IICCTLn0)
SPIEn bit :bit4 for IICA control register n0(IICCTLn0)

19.2.7 Serial clock control circuit

In the master mode, the circuit generates a clock output to the SCLAn pin from the sampling clock.

19.2.8 Serial clock waiting control circuit

This circuit controls the waiting sequence.

19.2.9 ACK generation circuit, stop condition detection circuit, start condition detection circuit, ACK detection circuit

These circuits generate and detect various states.

19.2.10 Data hold time correction circuit

This circuit generates a data hold time for the serial clock drop.

19.2.11 Start condition generation circuit

If the STTn bit is set to "1", this circuit generates a start condition.

However, in a state where scheduled communication is disabled (IICRSVn bit=1) and the bus (IICBSYn bit=1) is not released, STCFn.

19.2.12 Stop condition generation circuit

If the SPTn bit is set to "1", this circuit generates a stop condition.

19.2.13 Bus state detection circuit

The circuit detects whether the bus is released by detecting a start condition and a stop condition. However, the bus state cannot be detected immediately after operation, so the initial state of the bus state detection circuit must be set through STCENn bits.

Note 1.STTn bit: bit1 for IICA control register n0 (IICCTLn0)
SPTn bit: bit0 for 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

19.3 Registers for controlling serial interface IICA

The serial interface, IICA, is controlled through 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)

Remark: n=0,1

19.3.1 Peripheral enable register 0 (PER0)

The PER0 register is a register that sets a clock that is allowed or prohibited to supply to each peripheral hardware.Reduce power consumption and noise by stopping clock supply to unused hardware.

To use the serial interface IICA0, you must set bit5 (IICA0EN) to "1".

To use the serial interface IICA1, you must set bit6 (IICA1EN) to "1".

The PER0 register is set by an 8-bit memory operation instruction.

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

Figure 19-5 Format of peripheral enable register 0 (PER0)

After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
PER0	XX	IICA1EN	IICA0EN	XX	XX	XX	XX	XX

IICAnEN	Provides control of input clock of serial interface IICA
0	Stop providing an input clock. •SFR used by serial interface IICA cannot be written. •The serial interface IICA is in a reset state.
1	Enable providing an input clock. •SFR that can read and write to the serial interface IICA.

Note 1: To set the serial interface IICA, you must first set the following register in the state with the IICAnEN bit "1". When the IICAnEN bit is "0", the control register value for serial interface IICAn is the initial value, ignoring writes (except Port multiplexing function configuration register (PxxCFG), port mode register (PMxx) and port mode control register (PMCxx).

- IICA control register n0 (IICCTLn0)
- IICA flag register n (IICFn)
- IICA status register n (IICSn)
- IICA control register n1 (IICCTLn1)
- IICA Low level width setting register n (IICWLn)
- IICA High Level Width Setting Register n (IICWHn)

Remark: n=0,1

19.3.2 IICA control register n0 (IICCTLn0)

This is a register that allows or stops I2C running, sets wait times, and sets other I2C runs.

The IICCTLn0 register is set by an 8-bit memory operation instruction. However, SPIEn bits, WTIMn bits and ACKEn bits must be set when the IICEn bit is '0' or during waiting and IICEn bits can be set simultaneously.

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

Remark: n=0,1

Figure 19-6 Format of IICA control register n0 (IICCTLn0) (1/4)

After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
IICCTLn0	IICEn	LRELn	WRELn	SPIEn	WTIMn	ACKEn	STTn	SPTn

ICEn	I^{2}C Run Allowed
0	Stop running. Reset the IICA state register n(IICSn) [note 1], and stop the internal operation.
1	Allowed to run.
This bit must be set "1" with the SCLAn and SDAAn lines high.	
Clear Criteria (IICEn=0,1)	Set Criteria (IICEn=1)
· Clear by command. · When reset	· Set by command.

LRELn[Note2,3]	Exit of communication
0	Run normally
1	Exit the current communication and go into standby. Automatically clears "0" after execution. It is use in that case of receive an extension code independent of the local station, etc. The SCLAn line and the SDAAn line become high resistance. The following flags in the IICA control register n0 (IICCTLn0) and the IICA state register n (IICSn) are cleared: ·STTn·SPTn·MSTSn·EXCn·COIn·TRCn·ACKDn·STDn
The standby state is changed to the standby state of the exiting communication and is maintained until the following communication participation conditions are met. · Starts as a master device after a stop condition is detected. · Address matches or receives an extension code after a start condition is detected.	
Clear Criteria (LRELn=0,1)	Set Criteria (LRELn=1)
· Automatically clears after execution. · When reset	· Set by command.

WRELn[Note2,3]	Wait release
0	The wait is not released.
1	Unwait. Automatically clears after the wait is released.
If the WRELn bit (Unwait) is set during the 9th clock wait in the send state (TRCn=1), the SDAAn line becomes High impedance status (TRCn=0,1).	
Clear Criteria (WRELn=0,1)	Set Criteria (WRELn=1)
•Automatically clears after execution. •When reset	•Set by command.

Note 1. Reset the STCFn bit and IICBSYn bit of the IICA shift register n (IICAn), the IICA flag register n (IICFn), and the CLDn bit and DADn bit of the IICA control register n1 (IICCTLn1).

2. The IICEn bit is not valid in the state of '0'.

3. The read values for the LRELn and WRELn bits are always "0".

Notice: The start condition is detected immediately if I^{2}C operation is allowed (IICEn=1) when the SCLAn line is high, the

SDAAn line is low and the digital filter is ON (DFCn=1 in the IICCTLn1 register). In this case, the LRELn bit must be set "1" consecutively by a bit memory operation instruction after I^{2}C operation is allowed (IICEn=1).

Remark: n=0,1

Figure 19-6 Format of IICA control register n0 (IICCTLn0) (2/4)

SPIEn Note 1	Interrupt request generated by allow or disabling stop condition detection
0	prohibition
1	Allow
When the WUPn bit of the IICA control register n1 (IICTLn1) is "1", a stop condition interrupt is not generated even if the SPIEn bit is set to "1".	
Clear Criteria (SPIEn=0,1)	Set Criteria (SPIEn=1)
•Clear by command. •When reset	•Set by command.

WTIMn Note 1	Control of wait and interrupt requests
0	An interrupt request signal is generated at the descending edge of the eighth clock. Master device: After outputting 8 clocks, the clock output is set to a low level to wait. Slave: After 8 clocks are input, set the clock to a low level and wait for the master device.
1	An interrupt request signal is generated at the descending edge of the ninth clock. Master device: After outputting 9 clocks, the clock output is set to a low level to wait. Slave: After 9 clocks are input, set the clock to a low level and wait for the master device.
interrupt is generate at that descending edge of the 9th clock, regardless of the setting of this bit during address transfer; After the address transfer is completed, this bit is set to Effect. The master device enters a waiting state along the 9th clock descent during address transfer. The slave device receiving the local station address is responding A ninth clock down edge after (ACK) enters a waiting state, but a slave device receiving an extension code	
Clear Criteria (WTIMn=0,1)	Set Criteria (WTIMn=1)
•Clear by command. •When reset	•Set by command.

ACKEn Notes1,2	Response control
0	No replies.
1	Allow replies. The SDAAn line is set to a low level during the 9th clock.
Clear Criteria (ACKEn=0,1)	Set Criteria (ACKEn=1)
•Clear by command. •When reset	•Set by command.

Note 1. The bit has an invalid signal in the state with the IICEn bit "0". This bit must be set during this time.

2. The setting value is invalid if the address is not an expander during address transfer. When the device is a slave and the address matches, a response is generated regardless of the set value.

Remark: n=0,1

Figure 19-6 Format of IICA control register n0 (IICCTLn0) (3/4)

<table>
<tr><td>STTn[Notes 1, 2]</td><td colspan="2">Trigger of Start Condition</td></tr>
<tr><td>0</td><td colspan="2">Do not generate a start condition.</td></tr>
<tr><td>1</td><td colspan="2">When the bus is released (standby state, IICBSYn bit is "0"):
If this bit is set to "1", a start condition is generated (as startup of the master device). When a third party is communicating:
•Circumstances in which communication reservation is allowed (IICRSVn=0,1)
Use as start condition reservation flag. If this bit is '1', the start condition is automatically generated after the bus is released.
•IICRSVn=1, where communication reservation is prohibited.
Even if this bit is "1", the STTn bit is cleared and the STTn clear flag (STCFn) is set "1" without generating a start condition. Wait State (Master):
A restart condition is generated after the wait is removed.</td></tr>
<tr><td colspan="3">Precautions for Set Timing:
· Master Receive: Setting this bit to "1" during transmission is prohibited. This bit can only be set to "1" during the wait period after setting the ACKEn bit to "0" and informing the slave that reception has been completed.
· Master Send: The start condition may not be generated properly during the ACK period. This bit must be set to "1" during the wait period after the 9th clock is output.
· Disable from setting "1" at the same time as the trigger of the stop condition (SPTn).
· After setting the STTn bit to "1", it is prohibited to set this bit to "1" again before the clear condition is satisfied.</td></tr>
<tr><td colspan="2">Clear Criteria (STTn=0,1)</td><td>Set Criteria (STTn=1)</td></tr>
<tr><td colspan="2">· Set the STTn bit to "1" in the state where communication reservation is disabled.
· When arbitration fails
· Master device generation start condition.
· Clear due to LRELn bit '1' (Exit Communication)
· When the IICEn bit is "0"
· When reset</td><td>· Set by command.</td></tr>
</table>

Note 1. The bit has an invalid signal in the state with the IICEn bit "0".

2.The read value of the STTn bit is always "0".

Remark: 1. If you read bit1 (STTn) after setting the data, this bit becomes "0".

2. IICRSVn: bit0 for IICA flag register n(IICFn)

STCFn: bit7 for IICA flag register n(IICFn)

3. n=0,1

Figure 19-6 Format of IICA control register n0 (IICCTLn0) (4/4)

<table>
<tr><th>SPTn Note</th><th colspan="2">Trigger of stop condition</th></tr>
<tr><td>0</td><td colspan="2">No stop condition is generated.</td></tr>
<tr><td>1</td><td colspan="2">Generate a stop condition (end of transfer as master device).</td></tr>
<tr><td colspan="3">Precautions for Set Timing:
· Master Receive: Setting this bit to "1" during transmission is prohibited. This bit can be set to "1" only during the wait period after setting the ACKEn bit to "0" and notifying the slave that reception has been completed.
· Master Send: During the ACK period, the stop condition may not be generated properly. This bit must be set to "1" during the wait period after the 9th clock is output.
· Disable from setting "1" at the same time as the trigger of the start condition (STTn).
· The SPTn bit can be set to "1" only in the case of a master device..
· When the WTIMn bit is "0", it must be noted that if the SPTn bit is set to "1" during the wait period after 8 clocks of output, a stop condition is generated during the high level of the 9th clock after the wait is released. The WTIMn bit must be set from "0" to "1" during the wait period after 8 clocks of output and the SPTn bit must be set to "1" during the wait period after the 9th clock of output
· After setting the SPTn bit to "1", it is prohibited to set this bit to "1" again until the clear condition is satisfied.</td></tr>
<tr><td colspan="2">Clear Criteria (SPTn=0,1)</td><td>Set Criteria (SPTn=1)</td></tr>
<tr><td colspan="2">· When arbitration fails
· Automatically clears when a stop condition is detected.
· Clear due to LRELn bit '1' (Exit Communication)
· When the IICEn bit is "0"
· When reset</td><td>· Set by command.</td></tr>
</table>

Note: The read value of the SPTn bit is always "0".

Note that if the bit3 (TRCn) of the IICA state register n (IICSn) is "1" when the bit5 (WRELn) of the IICCTLn0 register is set to "1" at the ninth clock, the SDAAn line is set to a high impedance after clearing the TRCn bit (received state). The wait release when the TRCn bit is '1' (sending state) must be performed by writing IICA shift register n.

Remark: n=0,1

19.3.3 IICA status register n (IICSn)

This is a register that represents the I^{2}C status.

The 8-bit memory operation instruction can read the IICSn register only if the STTn bit is "1" and the wait period. After the reset signal is generated, the value of this register changes to "00H".

Note that the IICSn register is disabled from being read in the WUPn=1 state allowed in deep sleep mode. With WUPn bit '1', regardless of INTIICAn interrupt request, if WUPn bit is changed from '1' to '0', the change in state is not reflected until the next start or stop condition is detected. Therefore, to use wake-up functionality, you must allow (SPIEn=1) interrupts due to the detection of a stop condition and read the IICSn register after the interrupt is detected.

Remark	STTn	:bit1 of IICA control register n0(IICCTLn0)
	WOPn	:bit7 for IICA control register n1(IICCTLn1)

Figure 19-7 Format of IICA state register n(IICSn) (1/3)

after reset: 00H R

symbol	7	6	5	4	3	2	1	0
IICSn	MSTSn	ALDn	EXCn	COIn	TRCn	ACKDn	STDn	SPDn

MSTSn	Acknowledgement flag for master status
0	Slave or Communication Standby
1	master communication status
Clear Criteria (MSTSn=0,1)	Set Criteria (MSTSn=1)
•When a stop condition is detected •When the ALDn bit is "1" •Clear due to LRELn bit '1' (Exit Communication) •When the IICEn bit changes from "1" to "0" (stop running) •When reset	•When generating start conditions

ALDn	Detection of Arbitration Failure
0	indicating that no arbitration has occurred or the arbitration has been won.
1	Indicates arbitration failure. Clear MSTSn bits.
Clear Criteria (ALDn=0,1)	Set Criteria (ALDn=1)
•Automatically clear [note] after reading the IICSn register. •When the IICEn bit changes from "1" to "0" (stop running) · When reset	•When arbitration fails

Note: This bit is cleared even if a bit memory operation instruction is executed for a bit other than the IICSn register. Therefore, when using ALDn bits, you must read the data for the ALDn bits before reading the other bits.

Remark: 1. LRELn :bit6 for IICA control register n0(IICCTLn0)
ICEn :bit7 for IICA control register n0(IICCTLn0)
2. n=0,1

Figure 19-7 Format of IICA state register n(IICSn) (2/3)

EXCn	Receiving detection of extension codes
0	The extension code was not received.
1	An extension code was received.
Clear Criteria (EXCn=0,1)	Set Criteria (EXCn=1)
•When a start condition is detected •When a stop condition is detected •Clear due to LRELn bit '1' (Exit Communication) •When the IICEn bit changes from "1" to "0" (stop running) •When reset	•When the 4-bit height of the received address data is "0000" or "1111" (Set the rising edge of the 8th clock)

COIn	Detection of address matching
0	Different address.
1	Same address.
Clear Criteria (COIn=0,1)	Set Criteria (COIn=1)
•When a start condition is detected •When a stop condition is detected •Clear due to LRELn bit '1' (Exit Communication) •When the IICEn bit changes from "1" to "0" (stop running) •When reset	• When the receive address and the local station address (slave address register n (SVAn)) are the same (set the rising edge of the 8th clock)

TRCn	Transmit/receive status detection
0	In receive state except send state. The SDAAn line is set to a high impedance.
1	Is in a send state. Set to output the value of the SOn latch to the SDAAn line (valid after the descent of the ninth clock in byte 1).
Clear Criteria (TRCn=0,1)	Set Criteria (TRCn=1)
<master and slave devices> •When a stop condition is detected •Clear due to LRELn bit '1' (Exit Communication) •When the IICEn bit changes from "1" to "0" (stop running) •Clear due to WRELn bit of '1' note •When the ALDn bit changes from "0" to "1" (Arbitration Failure) •When resetting •Non-participation in communications (MSTSn, EXCn, COIn=0,1) <Master device> •When the LSB (transmission direction indicator bit) of the first byte outputs "1" <Slave> •When a start condition is detected •When the LSB (transmission direction indicator bit) of the first byte outputs "0".	<Master device> •When generating start conditions •When the LSB (Direction of Transmission Bit) of the 1st byte (Address Transfer) When outputting is "0" (Master transmit) <Slave> • When "1" (slave transmit) is outputted in the LSB (transmission direction indication bit) of the first byte (address transmission) of the master device

Note: When the bit3 (TRCn) of the IICA status register n (IICSn) is "1", if the bit5 (WRELn) of the IICA control register n0 (IICCTLn0) is set to "1" at the ninth clock, the SDAAn line is set to high impedance after the TRCn bit (reception state) is cleared. The wait release when the TRCn bit is '1' (sending state) must be performed by writing IICA shift register n.

Remark: 1.LRELn: bit6 for IICA control register n0(IICCTLn0)
ICEn: bit7 for IICA control register n0(IICCTLn0)
2. n=0,1

Figure 19-7 Format of IICA state register n(IICSn) (3/3)

ACKDn	Detection of Acknowledgements (ACK)
0	No ACK detected.
1	An ACK detected.

Clear Criteria (ACKDn=0,1)	Set Criteria (ACKDn=1)
•When a stop condition is detected •When the first clock of the next byte goes up •Clear due to LRELn bit '1' (Exit Communication) •When the IICEn bit changes from "1" to "0" (stop running) •When reset	•When the SCLAn line is set to a low level by the 9th clock rising edge of the SDAAn line

STDn	Detection of start condition
0	No start condition detected.
1	A start condition was detected, indicating that it is during address transfer.

Clear Criteria (STDn=0,1)	Set Criteria (STDn=1)
•When a stop condition is detected •When the first clock of the next five-minute section after address transfer rises •Clear due to LRELn bit '1' (Exit Communication) •When the IICEn bit changes from "1" to "0" (stop running) •When reset	•When a start condition is detected

SPDn	Detection of stop condition
0	No stop condition detected.
1	A stop condition is detected, the master device ends communication and the bus is released.

Clear Criteria (SPDn=0,1)	Set Criteria (SPDn=1)
•When the first clock of the address transfer byte rises after the start condition is detected after this bit •When the WUPn bit changes from "1" to "0 •When the IICEn bit changes from "1" to "0" (stop running) •When reset	•When a stop condition is detected

Note 1. LRELn: bit6 for IICA control register n0(IICCTLn0)

ICEn: bit7 for IICA control register n0(IICCTLn0)

2. n=0,1

19.3.4 IICA flag register n (IICFn)

This is a register that sets the I2C run mode and represents the I2C bus status.

The IICFn register is set by an 8-bit memory operation instruction. However, only the STTn clear flag (STCFn) and I2C bus status flag (IICBSYn) can be read.

The communication reservation function is allowed or disabled by the IICRSVn bit setting, and the initial value of the IICBSYn bit is set by the STCENn bit. IICRSVn bits and STCENn bits can only be written if I2C is disabled (bit7(IICEn)=0 for IICA control register n0(IICCTLn0). Only IICFn registers can be read after a run is allowed. After the reset signal is generated, the value of this register changes to "00H".

Figure 19-8 Format of IICA flag register n(IICFn)

After reset: 00H R/W Note

symbol	7	6	5	4	3	2	1	0
IICFn	STCFn	IICBSYn	0	0	0	0	STCENn	IICRSVn

STCFn	STTn clear flag
0	Release start condition.
1	The STTn flag cannot be cleared by issuing a start condition.
Clear Criteria (STCFn=0,1)	Set Criteria (STCFn=1)
•Clear due to STTn bit '1' •When the IICEn bit is "0" •When reset	•When the STTn bit is cleared to "0" because the start condition cannot be issued in a state set to disable IICRSVn=1

ICBSYn	I^2C bus status flag
0	Bus Release Status (Initial Communication Status at STCENn=1)
1	Bus Communication Status (Initial Communication Status at STCENn=0)
Clear Criteria (IICBSYn=0,1)	Set Criteria (IICBSYn=1)
•When a stop condition is detected •When the IICEn bit is "0" •When reset	•When a start condition is detected •Set IICEn Bit when STCENn bit is '0'

STCENn	Initial Allow Trigger
0	After allowing run (IICEn=1), the start condition is allowed to be generated by detecting the stop condition.
1	After allowing run (IICEn=1), the start condition is allowed to be generated without detecting the stop condition.
Clear Criteria (STCENn=0,1)	Set Criteria (STCENn=1)
· Clear by command. · When a start condition is detected · When reset	· Set by command.

IICRSVn	communication reservation function prohibition bit
0	Allow communication appointments.
1	No communication appointments.
Clear Criteria (IICRSVn=0,1)	Set Criteria (IICRSVn=1)
· Clear by command. · When reset	· Set by command.

Note: bit6 and bit7 are read-only bits.

Notice: 1. STCENn bits can only be written when the (IICEn=0,1) is stopped.

2. If the STCENn bit is "1", the bus is considered as IICBSYn=0,1 regardless of the actual bus state, so to avoid breaking other traffic when issuing the first start condition (STTn=1), it is necessary to confirm that there is no third party communicating.

3. Write the IICRSVn only when (IICEn=0,1).

Remark: 1. STTn: bit1 for IICA control register n0 (IICCTLn0)

2. IICEn: bit7 for the IICA control register n0 (IICCTLn0)

19.3.5 IICA control register n1 (IICCTLn1)

This is a register used to set the I^{2}C run mode and detect the status of the SCLAn and SDAAn pins.

The IICCTLn1 register is set by an 8-bit memory operation instruction. However, only CLDn and DADn bits can be read.

In addition to the WUPn bit, the IICCTLn1 register must be set when I2C is disabled (bit7(IICEn)=0 for IICA control register n0(IICCTLn0).

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

Figure 19-9 Format of IICA control register n1 (IICCTLn1) (1/2)

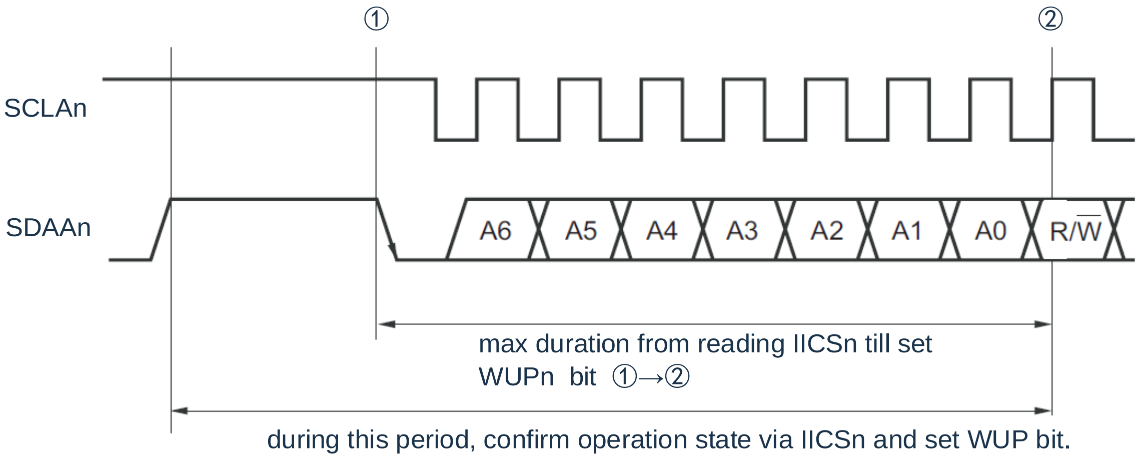
After reset: 00H R/W Note 1

symbol	7	6	5	4	3	2	1	0
IICCTLn1	WUPn	0	CLDn	DADn	SMCn	DFCn	0	PRSn

<table>
<tr><td>WOPn</td><td colspan="2">Control of address matching wake-up</td></tr>
<tr><td>0</td><td colspan="2">In deep sleep mode, stop the address matching wake-up function from running.</td></tr>
<tr><td>1</td><td colspan="2">In deep sleep mode, address matching wake-up functionality is allowed to run.</td></tr>
<tr><td colspan="3">To transfer to the deep sleep mode by setting the WUPn bit to "1", at least 3 f_{MCK} clocks must elapse after the WUPn bit is set to "1", and then the deep sleep instruction must be executed (refer to "Figure 14-28 Flow When Setting the WUPn Bit to "1"). 14-28 Flow when the WUPn bit is set to "1"). The WUPn bit must be cleared to "0" after the address is matched or the extension code is received. You can participate in subsequent communications by clearing the WUPn bit to "0" (it is necessary to clear the WUPn bit to "0" to release the wait and write and send data).
In the state where the WUPn bit is "1", the interrupt timing when the address is matched or the extension code is received is the same as the interrupt timing when the WUPn bit is "0".
(Delay difference of sampling error is generated according to the clock). In addition, when the WUPn bit is "1", no stop condition interrupt is generated even if the SPIEn bit is set to "1".</td></tr>
<tr><td colspan="2">Clear Criteria (WUPn=0,1)</td><td>Set Criteria (WUPn=1)</td></tr>
<tr><td colspan="2">• Clear by instructions (after address matching or receive an extension code).</td><td>• Set by instruction (MSTSn=0,1, EXCn=0,1, COIn=0,1, and STDn=0,1 (not participating in communication) Note 2.</td></tr>
</table>

Note that 1.bit4 and bit5 are read-only bits.

2. During the period shown below, the IICA status register n (IICSn) needs to be acknowledged and placed.


①
②
SCLAn
SDAAn
A6
A5
A4
A3
A2
A1
A0
R/W̄
max duration from reading IICSn till set WUPn bit ①→②
during this period, confirm operation state via IICSn and set WUP bit.

Remark: n=0,1

Figure 19-9 Format of IICA control register n1 (IICCTLn1) (2/2)

CLDn	SCLAn pin level detection (only valid if IICEn bit is "1")
0	The SCLAn pin is detected as low.
1	High SCLAn pin level detected.
Clear Criteria (CLDn=0,1)	Set Criteria (CLDn=1)
•When the SCLAn pin is low •When the IICEn bit is "0" •When reset	•When the SCLAn pin is high level

DADn	SDAAn pin level detection (only valid if IICEn bit is "1")
0	The SDAAn pin is detected as low.
1	High SDAAn pin level detected.
Clear Criteria (DADn=0,1)	Set Criteria (DADn=1)
•When the SDAAn pin is low •When the IICEn bit is "0" •When reset	•When the SDAAn pin is high level

SMCn	Switch of operation mode
0	Run in standard mode (maximum transfer rate: 100kbps).
1	Run in Fast Mode (Maximum Transfer Rate: 400kbps) or Enhanced Fast Mode (Maximum Transfer Rate: 1Mbps).

DFCn	Operation control of digital filter
0	Digital filter OFF
1	Digital Filter ON
Digital filter must be used in fast mode or enhanced fast mode. A digital filter is used to eliminate noise. Whether the DFCn bit is set to "1" or "0", the transmission clock remains unchanged.	

PRSn	Control of operating clock (f_{MCK})
0	Select f_{CLK} (1MHz≤f_{CLK}≤20MHz).
1	Select f_{CLK}/2 (20MHz<f_{CLK}).

Notice: 1. The maximum operating frequency of the $_{\text{IICA}}$ runtime clock (f_{MCK}) is 20MHz (Max.) The bit0 (PRSn) of the IICA control register n1 (IICCTLn1) must be set to "1" only if fCLK exceeds 20 MHz.

2. In the case of setting the transfer clock, you must note the minimum operating frequency of the fCLK. The fCLK minimum operating frequency of the serial interface IICA depends on the operation mode.

Quick mode: f_{CLK}=3.5MHz (Min.)
Enhanced fast mode: f_{CLK}=10MHz(Min.)
Standard mode: f_{CLK}=1MHz (Min.)

Remark: 1. IICEn: bit7 for IICA control register n0 (IICCTLn0)
2. n=0,1

19.3.6 IICA Low level width setting register n (IICWLn)

This register controls the SCLAn pin signal low level width (tLOW) and the SDAAn pin signal output by the serial interface IICA.

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

IICWLn register must be set when I2C is disabled (bit7(IICEn)=0 for IICA control register n0(IICCTLn0).

After the reset signal is generated, the value of this register changes to FFH.

For IICWLn register set-up methods, refer to "19.4.2 Method for setting transmission clock through IICWLn register and IICWHn register".

The data retention time is 1/4 of the time set by IICWLn.

Figure 19-10 Format of IICA low level width setting register n(IICWLn)

After Reset: FFH R/W

symbol	7	6	5	4	3	2	1	0
IICWLn								

19.3.7 IICA high level width setting register n (IICWHn)

This register controls the SCLAn pin signal high level width and the SDAAn pin signal output by the serial interface IICA.

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

IICWHn register must be set when I2C is disabled (bit7(IICEn)=0 for IICA control register n0(IICCTLn0).

After the reset signal is generated, the value of this register changes to FFH.

Figure 19-11 Format of IICA high level width setting register n(IICWHn)

After Reset: FFH R/W

symbol	7	6	5	4	3	2	1	0
ICWHn								

Remark: 1. Refer to 19.4.2 (1) for the setting method of the master transmission clock; Refer to 19.4.2 (2) for the dependent IICWLn register and IICWHn register set-up methods.

2. n=0,1

19.3.8 Registers for controlling the IICA pin port function

This product can multiplex the pin function of IICAn to multiple ports.

SCALn pins and SDAAn pins can be configured to ports separately by setting the port multiplexing function configuration registers (SCLAnPCFG and SDAAnPCFG). (n=0,1)

Set the bits of the Port Mode Control Register (PMCxx) and the Port Mode Register (PMxx) corresponding to these two ports to "0".

After these two ports are configured for multiplexing of IICA pins, the N-channel open drain output (VDD/EVDD withstand) mode of the ports is guaranteed to be turned on automatically by design, i.e. the POMxx register does not need to be set by the user.

See "Chapter 2 Port Function" for details.

19.4 Function of I2C bus mode

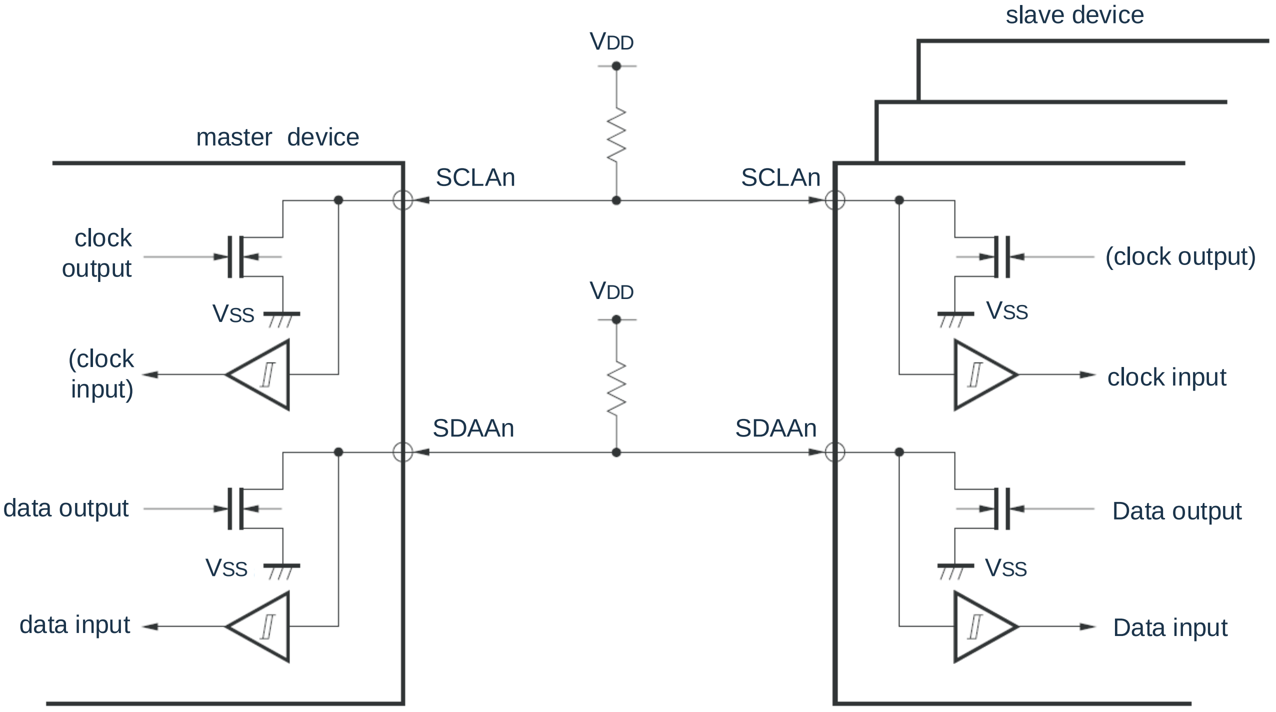
19.4.1 Pin structure

The serial clock pin (SCLAn) and the serial data bus pin (SDAAn) are structured as follows.

(1) SCLAn......Input/output pin for serial clock
The output of the master device and the slave device are N-channel drain open circuit output, and the input is Schmidt input.
(2) SDAAn......Input/output multiplexing pin for serial data
The output of the master device and the slave device are N-channel drain open circuit output, and the input is Schmidt input.

Since the output of the serial clock line and the serial data bus is an open-circuit output of the N-channel drain, an external pull-up resistor is required.

Figure 19-12 Pinout diagram



Remark: n=0, 1

19.4.2 Method for setting transmission clock through IICWLn register and IICWHn register

(1) Setting method for main control party transmitting clock

$$\text{Transmission clock} = \frac{f_{MCK}}{\text{IICWL} + \text{IICWH} + f_{MCK}(t_R + t_F)}$$

At this point, the best values for the IICWLn register and the IICWHn register are as follows: (Decimal portion of all set values rounded)

- Fast mode

$$\text{IICWLn}=\frac{0.52}{\text{transmission clock}}\times f_{MCK}$$

$$\text{IICWHn}=(\frac{0.48}{\text{transmission clock}}-t_R-t_F)\times f_{MCK}$$

- Standard mode

$$\text{IICWLn}=\frac{0.47}{\text{transmission clock}}\times f_{MCK}$$

$$\text{IICWHn}=(\frac{0.53}{\text{transmission clock}}-t_R-t_F)\times f_{MCK}$$

- Enhanced fast mode

$$\text{IICWLn}=\frac{0.50}{\text{transmission clock}}\times f_{MCK}$$

$$\text{IICWHn}=(\frac{0.50}{\text{transmission clock}}-t_R-t_F)\times f_{MCK}$$

(2) Slave IICWLn register and IICWHn register setting method (Decimal portion of all set values rounded)

- Fast mode
  IICWLn=1.3μs×f_{MCK}
  IICWHn=(1.2μs-t_R-t_F) ×f_{MCK}
- Standard mode
  IICWLn=4.7μs×f_{MCK}
  IICWHn=(5.3μs-t_R-t_F) ×f_{MCK}
- Enhanced fast mode
  IICWLn=0.50μs ×f_{MCK}
  IICWHn=(0.50μs-t_R-t_F) f_{MCK}

Note 1. The maximum operating frequency of the IICA runtime clock (f_{MCK}) is 20MHz (Max.). The bit0 (PRSn) of the IICA control register n1 (IICCTLn1) must be set to "1" only if f_{CLK} exceeds 20 MHz.

2. In the case of setting the transfer clock, you must note the minimum operating frequency of the fCLK. The fCLK minimum operating frequency of the serial interface IICA depends on the operation mode.

Fast mode: f_{CLK}=3.5MHz (Min.)
Enhanced fast mode: f_{CLK}=10MHz(Min.)
Standard mode:f_{CLK}=1MHz (Min.)

Note 1. Since the rising time (tR) and falling time (tF) of the SDAAn signal and SCLAn signal differ by pull up resistance and wiring capacitance.

2. IICWLn : IICA low-level width setting register n

ICWHn : IICA high-level width setting register n
t_F : SDAAn and SCLAn signal falling times
t_R rise time of SDAAn and SCLAn signals
f_{MCK} : IICA operation clock frequency

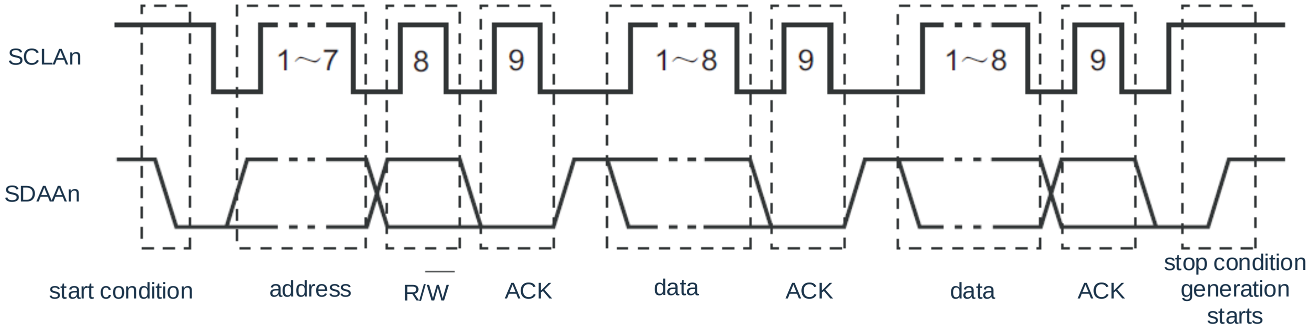
3. n=0,1

19.5 Definition and control method of I²C bus

The following describes the serial data communication format and the signal used for the I²C bus.

The timing of each transmission of "start condition", "address", "data" and "stop condition" generated on the serial data bus of the I²C bus is shown in the figure below.

Figure 19-13 Serial data transfer timing of I²C bus



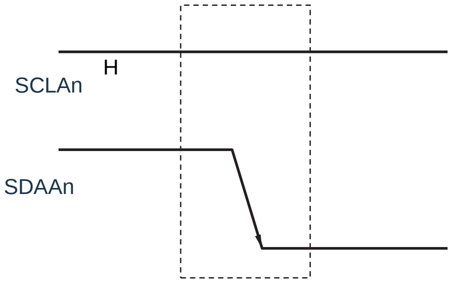
The master generates a start condition, a slave address, and a stop condition.

Both the master device and the slave device can generate an acknowledgement (ACK) (in general, the receiver outputs 8 bits). The master device continuously outputs a serial clock (SCLAn). However, the slave device can extend the low level period of the SCLAn pin and insert the wait.

19.5.1 Start condition

When the SCLAn pin is high, if the SDAAn pin changes from high to low, a start condition is generated. The SCLAn pin and the SDAAn pin start condition is the signal generated when the master device starts serial transfer to the slave device. When used as a slave, a start condition is detected.

Figure 19-14 Start Condition



The start condition is output if the IICA control register n0 (bit0=1 of IICCTLn0) is set to '1'. If a start condition is detected, set the bit1 (STDn) of the IICSn register to "1".

Remark n=0,1

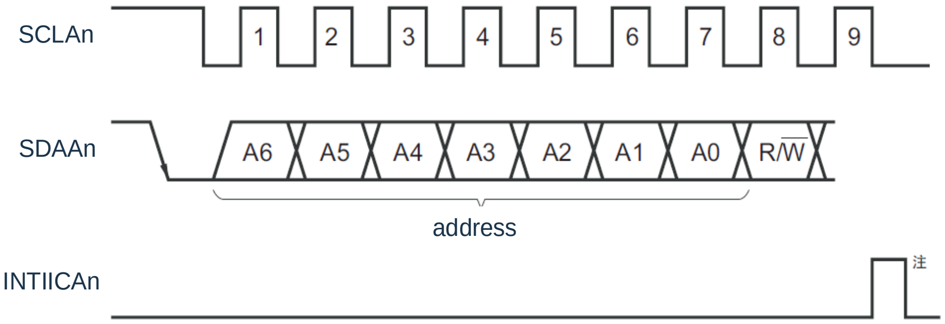
19.5.2 Address

The subsequent 7 bits of data for the start condition are defined as the address.

The address is a 7-bit data output by the master device in order to select a specific slave device from the multiple slave devices connected to the bus. Therefore, the slave device on the bus need to be set to completely different addresses.

The slave device detects the start condition by hardware and checks whether the 7-bit data is the same as the slave address register n (SVAn). At this time, if the 7-bit data and the SVAn register have the same value, the slave device is selected to communicate with the master device before generating a start condition or a stop condition.

Figure 19-15 Address



Note: If data other than the local station address or extension code is received during the slave run, no INTIICAn is generated.

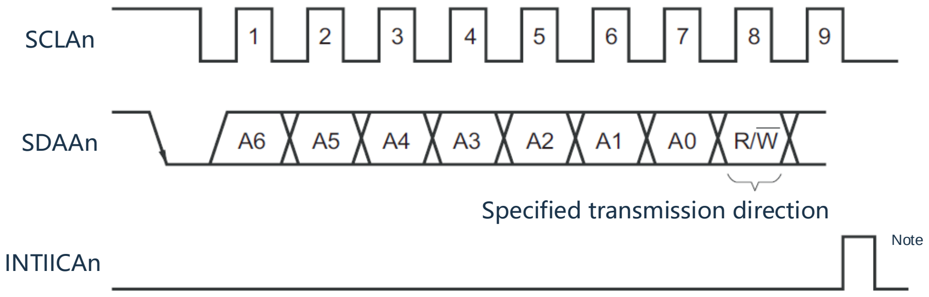
If 8 bits of data constituting the delivery direction described in "19.5.3 Assignment of transmission direction" are written to IICA shift register n(IICAn). The received address is written to the IICAn register. The slave addresses are assigned to the IICAn register as high as 7 bits.

19.5.3 Assignment of transmission direction

The master device transmits 1 bits of data specifying the transmission direction after the 7-bit address.

When the designated bit of the transmission direction is '0', the main control device transmits data to the slave device; When the specified bit of this transfer direction is "1", the master receives data from the slave.

Figure 19-16 Assignment of transmission direction



Note: If you receive data other than the local station address or the expansion code at the slave run-time, no INTIICAn is generated.

Remark: n=0,1

19.5.4 Acknowledge (ACK)

The serial data status of the sender and receiver can be confirmed by an acknowledgement (ACK). The receiver returns a reply each time it receives 8 bits of data.

Typically, the sender receives a reply after sending 8-bit data. When the receiver returns a reply, it is considered that the reply has been received normally and the processing is continued. Detection of acknowledgements that can be confirmed by bit2(ACKDn) of the IICA status register n(IICSn). When the master control device receives the last data in the receiving state, the stop condition is generated without returning the response. When a response is not returned after receiving data from a secondary device, the main control device outputs a stop condition or a restart condition to abort transmission. The reason for not returning an acknowledgement is as follows:

(1) No normal reception.

(2) The receipt of final data has been ended.

(3) There is no receiver specified by address.

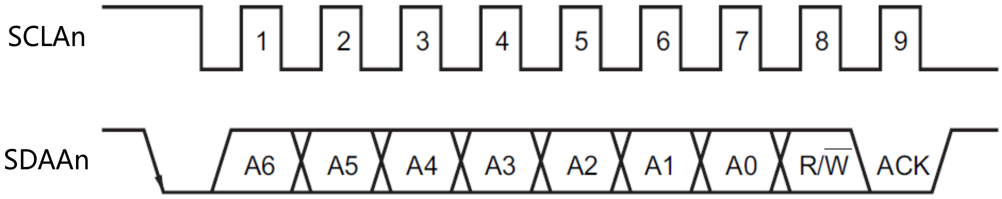
The receiver sets the SDAAn line at a low level at the 9th clock to generate a response (normal reception).

Set bit2 (ACKEn) of the IICA control register n0 (IICCTLn0) to "1" to enable automatic generation of an answer. Set bit3 (TRCn) of the IICSn register with the 8th bit of data following the 7-bit address information. In the case of reception (TRCn=0,1), it is usually necessary to set the ACKEn bit to "1".

When a slave receive run (TRCn=0,1) cannot receive data or does not need the next data, the ACKEn bit must be cleared to "0".

When the next data is not needed in the main receive run (TRCn=0,1), the ACKEn bit must be cleared to "0".

Figure 19-17 Acknowledgement



When the address of the local station is received, the response is generated automatically regardless of the value of the ACKEn bit. When an address of a non-local station is received, no reply is generated (NACK).

When an extension code is received, an ACK is generated by setting the ACKEn bit to "1" in advance. The ACK generation method for receiving data varies depending on the setting of the wait timing as follows.

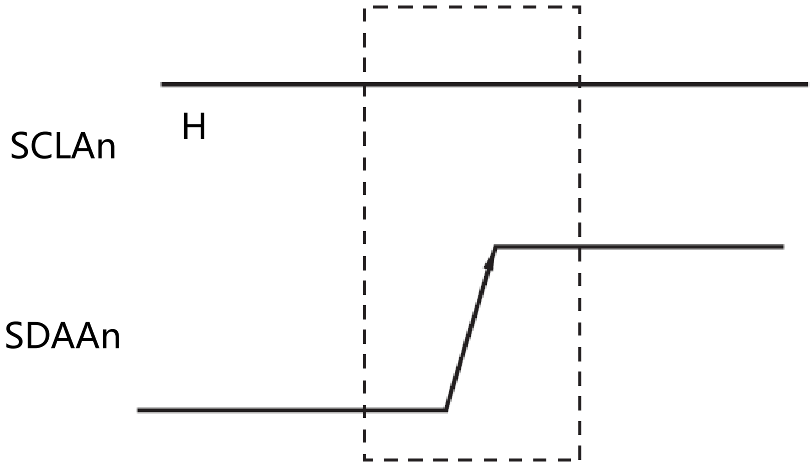
- When 8 clocks of wait are selected (bit3 (WTIMn) = 0 in the IICCTLn0 register): an ACK is generated synchronously with the 8th clock falling edge of the SCLAn pin by setting the ACKEn bit to "1" before releasing the wait.
- When 9 clocks of wait are selected (bit3 (WTIMn) = 1 in the IICCTLn0 register): an ACK is generated by setting the ACKEn bit to "1" in advance.

Remarks: n=0,1

19.5.5 Stop condition

When the SCLAn pin is a high level, a stop condition is generated if the SDAAn pin changes from a low level to a high level. The stop condition is a signal generated when the master device finishes serial transfer to the slave device. A stop condition is detected when used as a slave device.

Figure 19-18 Stop condition



A stop condition is generated if the bit0(SPTn) of the IICA control register n0(IICCTLn0) is set to '1'. If a stop condition is detected, the bit0 (SPDn) of IICA status register n (IICSn) is set "1" and a INTIICAn is generated when the bit4 (SPIEn) of the IICCTLn0 register is "1.

Remark: n=0, 1

19.5.6 Waiting

Notify the other master or slave to prepare the sending/receiving of data by waiting (waiting state).

Notify the other party that it is waiting by placing the SCLAn pin at a low level. If both the master device and the slave device wait states are released, the next transfer can begin.

Figure 19-19 Waiting (1/2)

(1) When the master device waits for 9 clocks and the slave device waits for 8 clocks (Master: transmit, slave: receive, ACKEn=1)

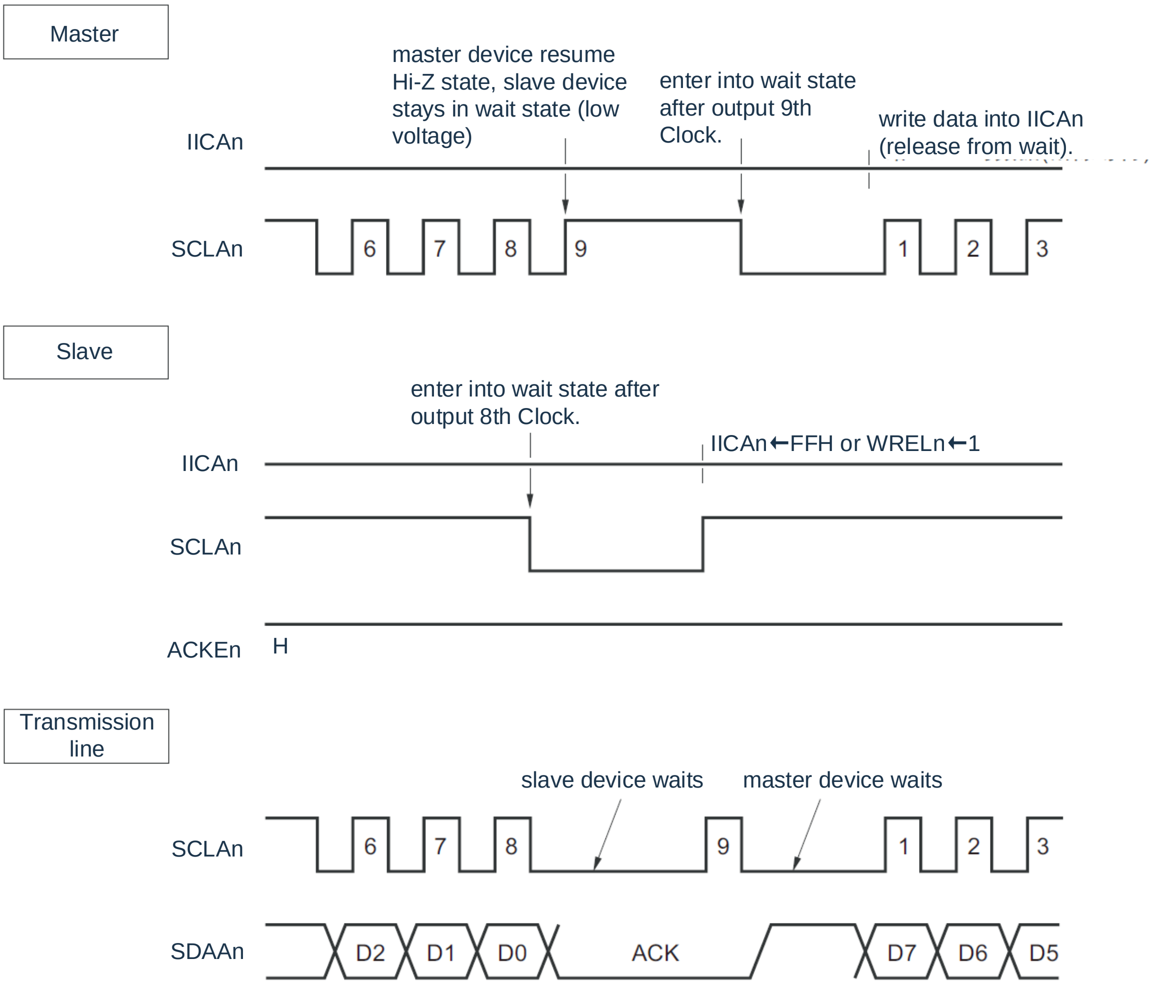
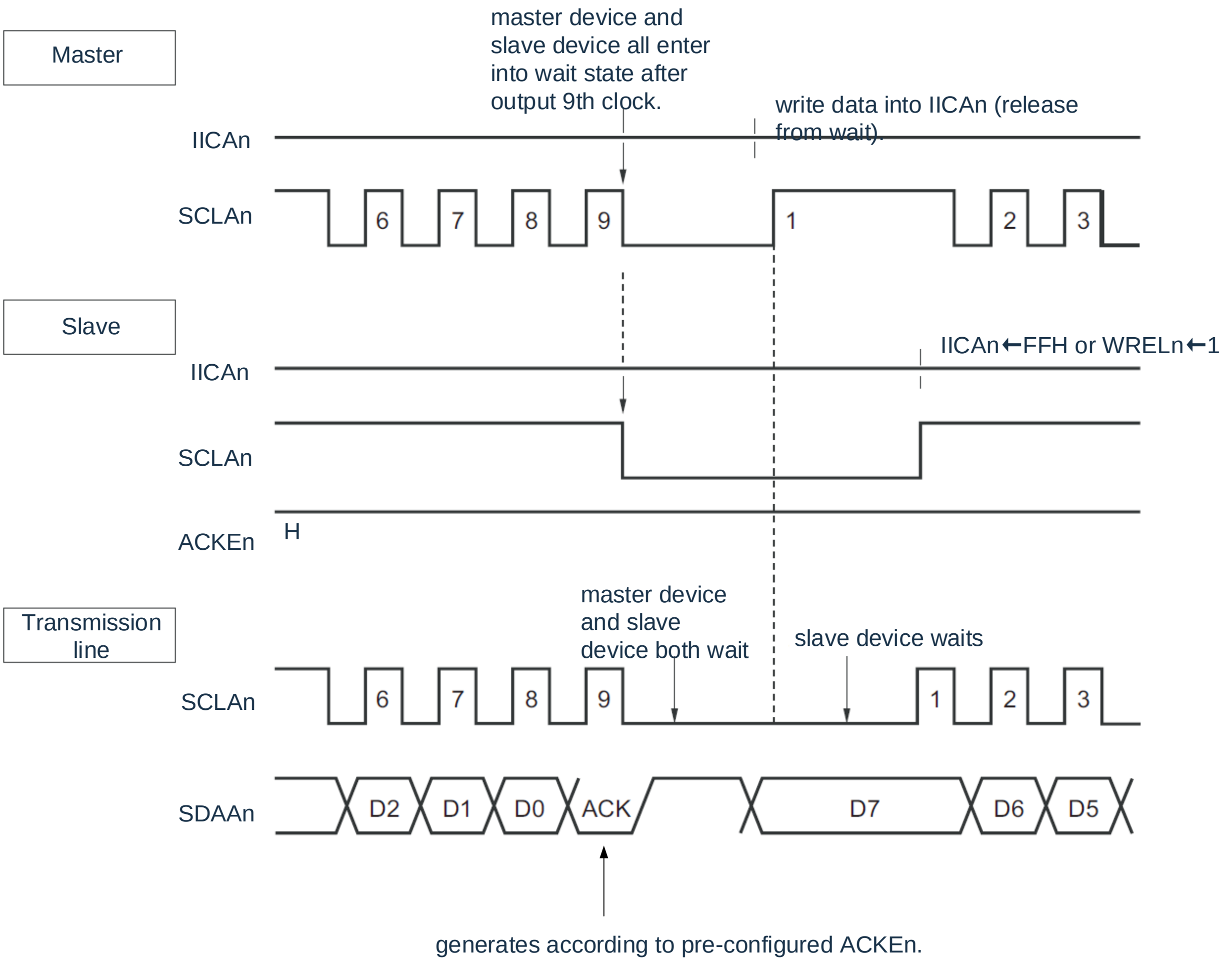

Master
master device resume Hi-Z state, slave device stays in wait state (low voltage)
enter into wait state after output 9th Clock.
write data into IICAn (release from wait).
IICAn
SCLAn
6
7
8
9
1
2
3
Slave
enter into wait state after output 8th Clock.
IICAn←FFH or WRELn←1
IICAn
SCLAn
ACKEn
H
Transmission line
slave device waits
master device waits
SCLAn
6
7
8
9
1
2
3
SDAAn
D2
D1
D0
ACK
D7
D6
D5

Figure 19-19 Waiting (2/2)

(2) When the master and slave are waiting for 9 clocks
(Main control equipment: Send, slave: Receive, ACKEn=1)


Master
master device and slave device all enter into wait state after output 9th clock.
write data into IICAn (release from wait).
IICAn
SCLAn
6
7
8
9
1
2
3
Slave
IICAn←FFH or WRELn←1
IICAn
SCLAn
ACKEn
H
Transmission line
master device and slave device both wait
slave device waits
SCLAn
6
7
8
9
1
2
3
SDAAn
D2
D1
D0
ACK
D7
D6
D5
generates according to pre-configured ACKEn.

Remark: ACKEn:bit2 for the IICA control register n0 (IICCTLn0)
WRELn:bit5 for the IICA control register n0 (IICCTLn0)

A wait state is automatically generated by setting the bit3 (WTIMn) of the IICA control register n0 (IICCTLn0). In general, at the receiver, if the bit5 (WRELn) of the IICCTLn0 register is "1" or writes "FFH" to the IICA shift register n (IICAn) At the sender, if data is written to the IICAn register, the wait is canceled. The master device also removes the wait by:

- Set the bit1 (STTn) of the IICCTLn0 register to 1.
- Set the bit0 (SPTn) of the IICCTLn0 register to 1.

Remark: n=0, 1

19.5.7 Waitinging release method

In general, that I^2C can relieve the wait by the following treatment.

- Write data to IICA shift register n(IICAn).
- Set the bit5(WRELn) of the IICA control register n0(IICCTLn0)(wait released).
- Set the bit1 (STTn) of the IICCTLn0 register (generate a start condition) Note.
- Set the bit0 (SPTn) of the IICCTLn0 register (generate a stop condition) Note.

Note: Limited to master devices.

If these waiting undoing processes are performed, I^2C unwaits and restarts communication. To send data (including addresses) after the unwait, you must write data to the IICAn register.

To receive data after unwaiting or end sending data, the bit5 (WRELn) of the IICCTLn0 register must be set "1". To generate a restart condition after unwaiting, the bit1(STTn) of the IICCTLn0 register must be set "1". To generate a stop condition after unwaiting, the bit0(SPTn) of the IICCTLn0 register must be set "1". You can perform only one unprocess on a single wait.

For example, if data is written to the IICAn register after the wait is released by clearing the WRELn bit by "1", the timing of the change of the SDAAn line may conflict with the timing of the write of the IICAn register, resulting in an incorrect value being output to the SDAAn line. In addition to these processes, if the communication is aborted in the middle of the communication, the communication is stopped by clearing the ICEn bit to "0", so that the wait can be canceled. If the I^2C bus state is deadlocked due to noise, setting bit6 (LRELn) of the IICCTLn0 register to "1" exits the communication, thus releasing the wait.

Notice: If the wait release process is performed when the WUPn bit is "1", the wait is not released.

Remark: n=0,1

19.5.8 Generation timing and waiting control of interrupt request (INTIICAn)

By setting a bit3 (WTIMn) of the IICA control register n0 (IICCTLn0), INTIICAn is generated in the timing shown in Table 19-2 and is subjected to wait control.

Table 19-2 Generation timing and waiting control of INTIICAn

WTIMn	Slave operation			Master operation		
	address	data receive	data transmit	address	data receive	data transmit
0	9 Note 1, 2	8 Note 2	8 Note 2	9	8	8
1	9 Note 1, 2	9 Note 2	9 Note 2	9	9	9

Note 1. The slave device generates a INTIICAn signal and enters a waiting state at the descent edge of the 9th clock only if the received address and the set address of the slave address register n (SVAn) are identical.

At this point, a reply is generated regardless of the setting of the bit2 (ACKEn) of the IICCTLn0 register. The slave device receiving the spreading code generates a INTIICAn at the descending edge of the 8th clock. If that address is different aft the restart, INTIICAn is generate at the descending edge of the 9th clock, but does not enter the wait state.

2. If the received address and the content of the slave address register n (SVAn) are different and no extension code is received, INTIICAn is not generated.

Note: The numbers in the table represent the number of clocks for the serial clock. Both interrupt request and wait control are synchronized with the descent of the serial clock.

(1) Transmission and reception of addresses

- Slave Run: Regardless of the WTIMn bit, the interrupt and waiting timing is determined according to the conditions of note 1 and note 2 above.
- Master Run: Independently of the WTIMn bit, the descending edge of the 9th clock generates a sequence of interruptions and waits.

(2) Data reception

- Master Run/Slave Run: The WTIMn bits determine the timing of interrupts and waits.

(3) Data transmission

- Master Run/Slave Run: The WTIMn bits determine the timing of interrupts and waits.

Remark: n=0,1

(4) Method of wait releasing

There are four methods for the release of the waiting:

- Write data to IICA shift register n(IICAn).
- Set the bit5(WRELn) of the IICA control register n0(IICCTLn0)(wait released).
- Set the bit1 (STTn) of the IICCTLn0 register (generate a start condition) Note.
- Set the bit0 (SPTn) of the IICCTLn0 register (generate a stop condition) Note.

Note Limited to master devices.

When selecting a wait of 8 clocks (WTIMn=0,1), you need to decide whether to generate a response before canceling the wait.

(5) Test of stop condition

If a stop condition is detect, INTIICAn (SPIEn=1 only case) occurs.

19.5.9 Method for detecting address matching

In I^{2}C bus mode, the master device can select a specific slave device by sending a slave address. It can automatically detect address matching through hardware. When the master device sends the same slave address and the same set address of the slave address register n (SVAn) or when only the spreading code is received, a INTIICAn interrupt request is generated.

19.5.10 Error detection

In I^{2}C bus mode, since the state of the serial data bus (SDAAn) during transmission is taken to the IICA shift register n(IICAn) of the transmitting device, it is possible to detect transmission errors by comparing the IICA data before starting transmission with the data after transmission. At this time, if two data are different, a transmission error is determined to have occurred.

Remark: n=0, 1

19.5.11 Extension code

(1) When the 4th bit of the receiving address is '0000' or '1111', the expansion code receiving flag (EXCn) is set '1' and an interrupt request (INTIICAn) is generated at the descending edge of the eighth clock. Does not affect the local station address stored in the slave address register n (SVAn).

(2) When the setting value of the SVAn register is "11110xx0", the following setting occurs if "11110xx0" is sent from the master device. However, an interrupt request (INTIICAn) is generated along the descending edge of the eighth clock.

- Same 4-bit data: EXCn=1
- 7-bit data identical :COIn=1

Note EXCn: bit5 for IICA status register n(IICSn).
COIn: bit4 for IICA status register n(IICSn).

(3) The processing after the interrupt request is different because of the subsequent data of the extension code, and is processed by software. If an expansion code is received at a slave run, communication is attended even if the address is different. For example, if you do not want to run as a slave after receiving the extension code, you must set bit6(LRELn) of IICA control register n0(IICCTLn0) to "1".

Table 19-3 Bit definitions for major extension codes

slave address	R/W bits	Description
0000000	0	General call address
11110xx	0	10-bit Slave address specification (when address is authenticated)
11110xx	1	10-bit Slave Address specification (when issuing read commands after the same address)

Note 1. Refer to the I^{2}C bus specification issued by NXP for expansion codes other than those listed above.
2. n=0, 1

19.5.12 Arbitration

When a plurality of master devices generate start conditions at the same time (When the STTn bit is set to "1" before the STDn bit is changed to "1".). This run is called arbitration.

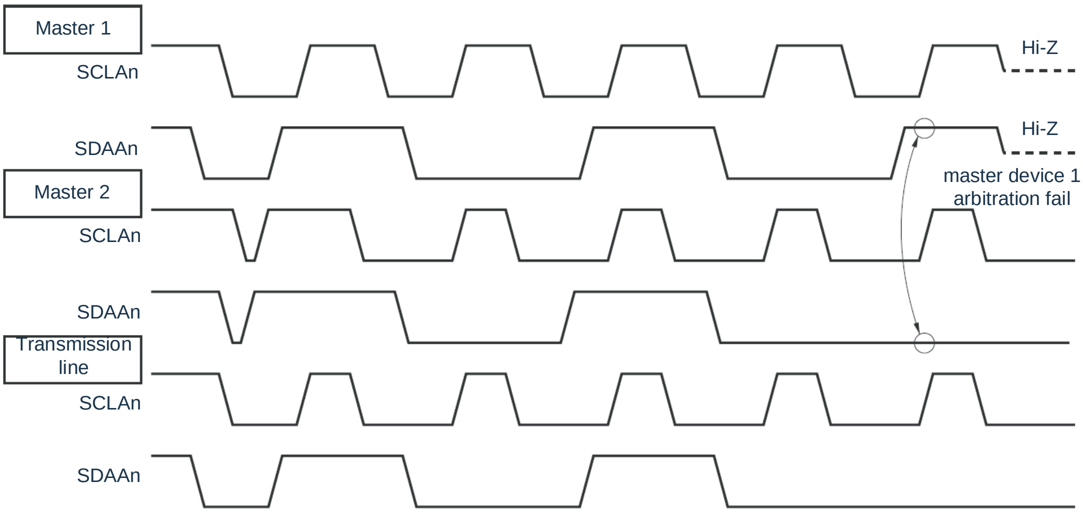
When arbitration failure occurs, the master control device sets ALDn of IICA state register n(IICSn) to "1", and sets SCLAn and SDAAn lines to high impedance state, releasing the bus.

When the next interrupt request occurs (for example: Stop condition is detected at 8 or 9 clock), and arbitration failure is detected by software with ALDn bit '1'.

For the generation sequence of interrupt requests, refer to "19.5.8 Generation timing and waiting control of interrupt request (INTIICAn)."

Remark: STDn: bit1 for IICA status register n(IICSn).
STTn: bit1 for the IICA control register n0 (IICCTLn0)

Figure 19-20 Example of arbitration timing



Remark: n=0, 1

Table 19-4 Status during arbitration and interrupt request generation timing

Status when arbitration occurs	Generation timing of interrupt requests
During address transmission	At falling edge of eighth or ninth clock following byte transfer[Note 1]
Read/write data after address transmission	
During extension code transmission	
Read/write data after extension code transmission	
During data transmission	
During ACK transfer period after data transmission	
When restart condition is detected during data transfer	
When stop condition is detected during data transfer	When stop condition is generated (when SPIEn = 1)[Note 2]
When data is at low level while attempting to generate a restart condition	At falling edge of eighth or ninth clock following byte transfer[Note 1]
When stop condition is detected while attempting to generate a restart condition	When stop condition is generated (when SPIEn = 1)[Note 2]
When data is at low level while attempting to generate a stop condition	At falling edge of eighth or ninth clock following byte transfer[Note 1]
When SCLAn is at low level while attempting to generate a restart condition	

Note: 1. When the WTIMn bit (bit 3 of IICA control register n0 (IICCTLn0)) = 1, an interrupt request occurs at the falling edge of the ninth clock. When WTIMn = 0 and the extension code’s slave address is received, an interrupt request occurs at the falling edge of the eighth clock.

2. When there is a chance that arbitration will occur, set SPIEn = 1 for master device operation.

Remark: 1. SPIEn: bit4 for IICA control register n0 (IICCTLn0)

2. n=0,1

19.5.13 Wake-up function

This is a slave function of I²C, which is the function of generating an interrupt request signal (INTIICAn) upon receipt of the local station address and expansion code. In the case of different addresses, the unnecessary INTIICAn signal is not generated, thereby improving the processing efficiency. If a start condition is detected, a wake-up standby state is entered. Since the master device (which has generated a start condition) may also become a slave device due to arbitration failure, the master device enters a wake-up standby state while sending an address.

To use the wake-up function in deep sleep mode, the WUPn bit must be set to "1". The address can be received regardless of the running clock. Even in this case, an interrupt request signal (INTIICAn) is generated upon receipt of the local station address and the expansion code. After this interrupt occurs, the WUPn bit is cleared '0' by the instruction, returning to the normal run.

The flow when the WUPn bit is set to "1" is shown in Figure 19-21, and the flow when the WUPn bit is set to "0" by address matching is shown in Figure 19-22.

Figure 19-21 Flow when setting the WUPn bit to "1"

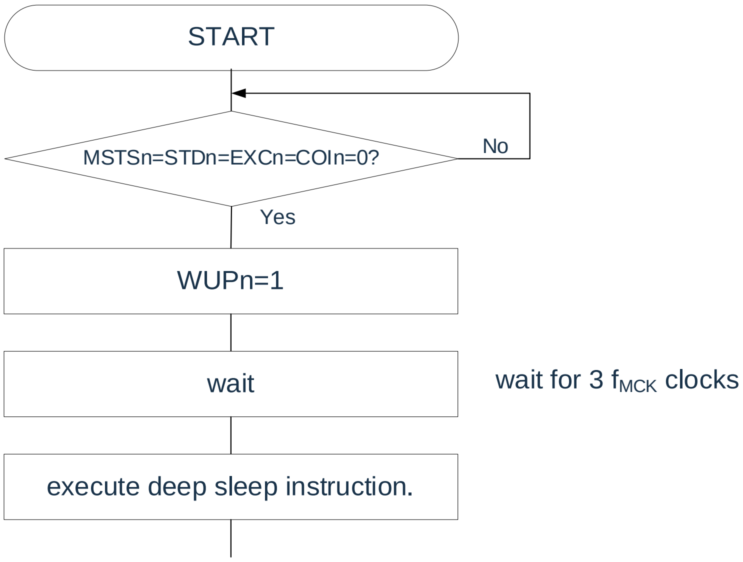
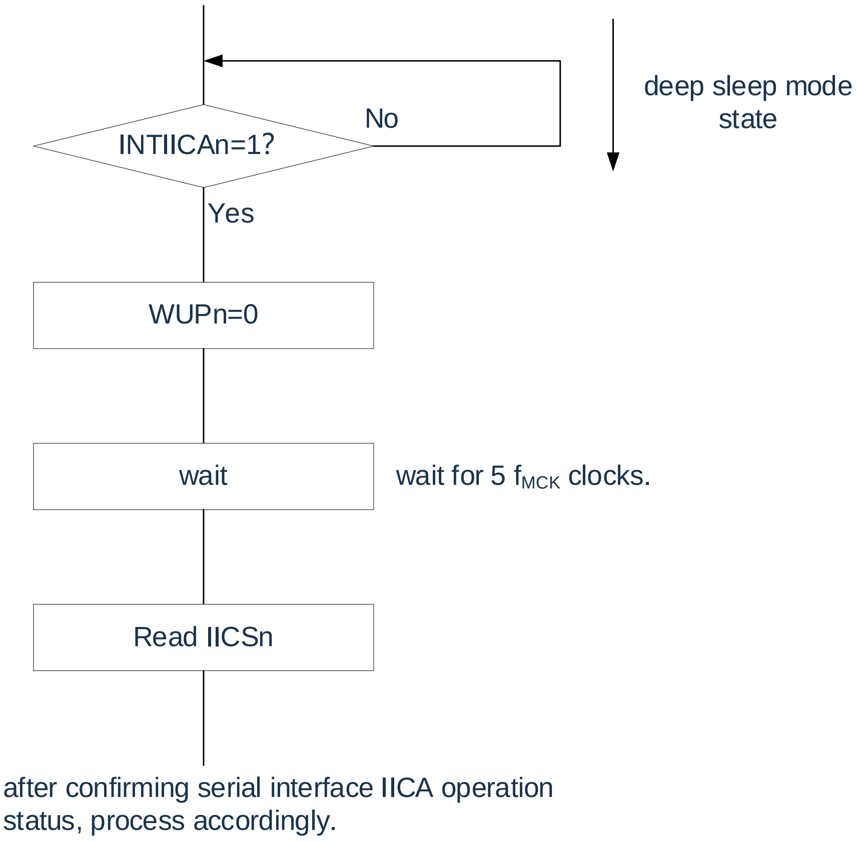


Figure 19-22 Process of "0" WUPn by address matching, including extension code reception



In addition to the interrupt request (INTIICAn) from the serial interface IICA, deep sleep mode must be removed through the following process.

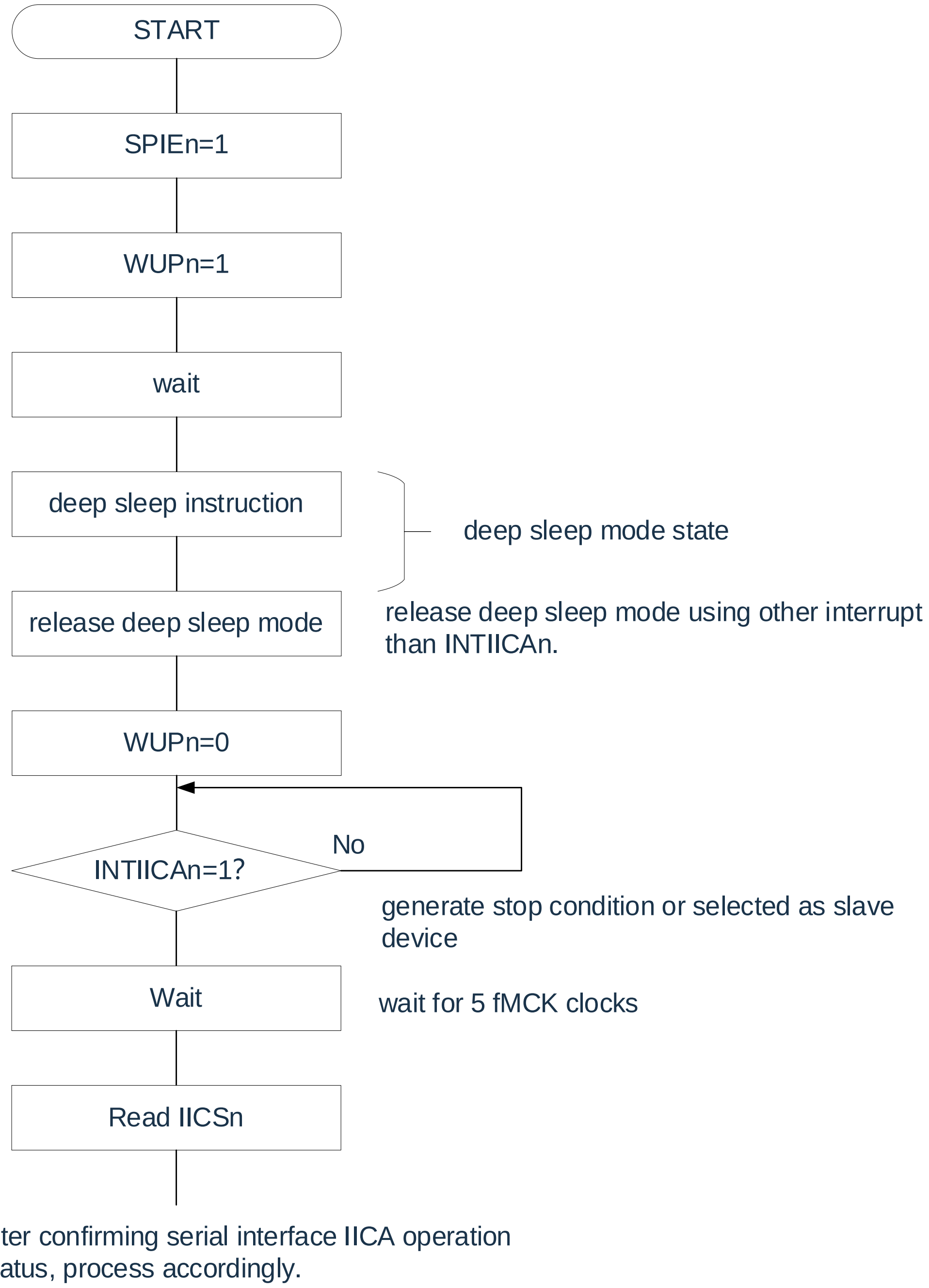
- The next IIC communication is the main control equipment running situation: Figure 19-23 Process
- The next IIC communication is when the slave is running:

Returns from INTIICAn Interrupt: The procedure is the same as in Figure 19-22.

Returns from an interrupt other than an INTIICAn interrupt: You must keep the WUPn bit "1" running before a INTIICAn interrupt is generated.

Remark: n=0, 1

Figure 19-23 Operating as a master device after deep sleep mode is removed by an interrupt other than INTIICAn


START
SPIEn=1
WUPn=1
wait
deep sleep instruction
deep sleep mode state
release deep sleep mode
release deep sleep mode using other interrupt than INTIICAn.
WUPn=0
INTIICAn=1?
No
generate stop condition or selected as slave device
Wait
wait for 5 fMCK clocks
Read IICSn
after confirming serial interface IICA operation status, process accordingly.

Remark: n=0, 1

19.5.14 Communication reservation

(1) The case where the communication reservation function is allowed (IICA flag register n(IICFn) bit0(IICRSVn)=0)

When the next master control communication is performed in the state of not joining the bus, a start condition can be sent when the bus is released by a communication reservation. The non-join bus at this time includes the following two states:

- When the arbitration result is neither master nor slave
- bit6(LRELn) of IICA control register n0(IICCTLn0) is set '1' after receiving the spreading code and releasing the bus

If the bit1(STTn) of the IICCTLn0 register is set 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.

The IICCTLn0 register's bit4 (SPIEn) is set to '1'. After detecting the release of the bus (stopping condition) by the generated interrupt request signal INTIICAn, communication is automatically initiated as the master. The data written to the IICAn register is invalid before stopping conditions are detected.

When the STTn bit is set to "1", the bus status determines whether to operate as a start condition or as a communication reservation.

- Bus is on release...........................Build Start Condition
- When the bus is not in a released state (standby state).....communication reservation

After the STTn bit is set to "1" and a wait time has elapsed, the MSTSn bit (bit 7 of the IICA status register n (IICSn)) confirms whether or not to operate as a communication reservation.

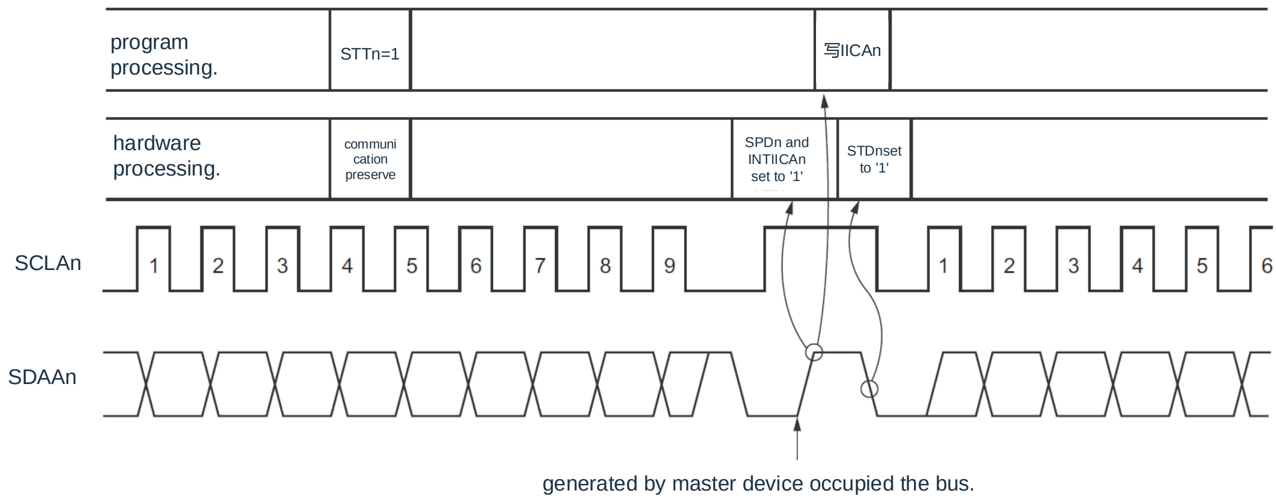
Waiting times calculated by the following equation must be ensured by software:

Waiting time from setting the STTn bit to "1" until the MSTSn flag is recognized:
(Set value for IICWLn+Set value for IICWHn+4)/ f_{MCK}+t_F×2

Remark: 1.IICWLn : IICA low-level width setting register n
ICWHn : IICA high-level width setting register n
t_F : SDAAn and SCLAn signal falling times
f_{MCK} : IICA operation clock frequency
2. n=0,1

The timing of the communication reservation is shown in the figure below.

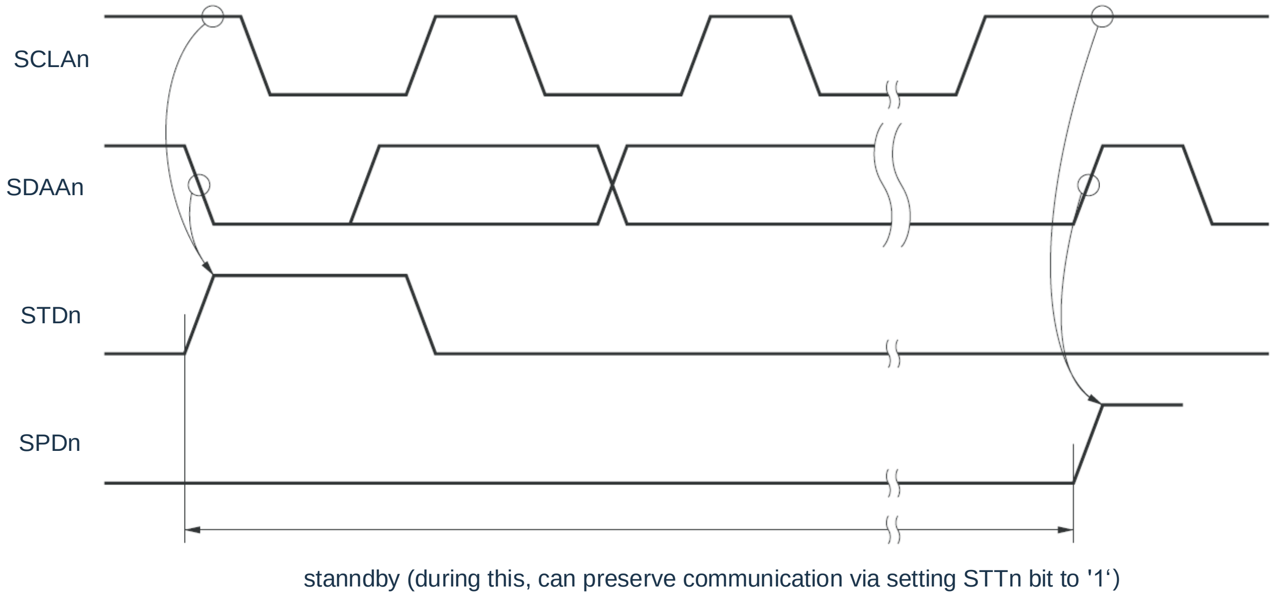
Figure 19-24 Timing of communication appointments



Remark: ICAn : IICA shift register n
STTn : bit1 of IICA control register n0(IICCTLn0)
STDn : bit1 for IICA state register n(IICSn)
SPDn : bit0 for IICA state register n(IICSn)

The communication reservation is accepted by the timing shown in Figure 19-25. After the bit1(STDn) of the IICA status register n(IICSn) becomes '1' and before the stop condition is detected, the bit1(STTn) of the IICA control register n0(IICCTLn0) is set '1' for communication reservation.

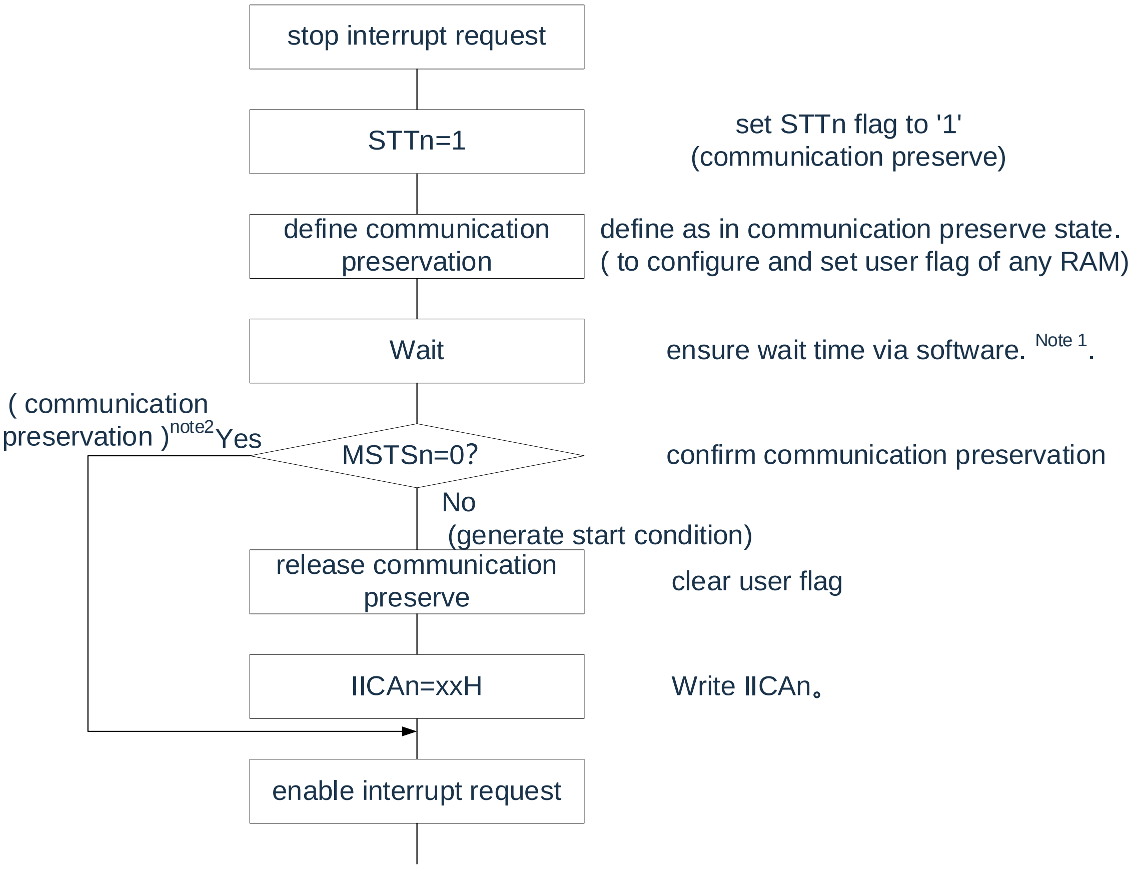
Figure 19-25 Timing for accepting communication reservations


SCLAn
SDAAn
STDn
SPDn
standby (during this, can preserve communication via setting STTn bit to '1')

The steps of the communication reservation are shown in Figure 19-26.

Remark n=0,1

Figure 19-26 Steps for communication reservation


stop interrupt request
STTn=1
set STTn flag to '1'
(communication preserve)
define communication preservation
define as in communication preserve state.
(to configure and set user flag of any RAM)
Wait
ensure wait time via software. Note 1.
(communication preservation)note2
Yes
MSTSn=0?
confirm communication preservation
No
(generate start condition)
release communication preserve
clear user flag
IICAn=xxH
Write IICAn。
enable interrupt request

Note 1. The wait time is as follows: (Set value for IICWLn + Set value for IICWHn +4)/ f_{MCK}+t_Fx2

2. Write the IICA shift register n (IICAn) by stopping the conditional interrupt request when the communication reservation is running.

Remark: 1.STTn : bit1 of IICA control register n0(IICCTLn0)

MSTSn :bit7 of IICA state register n(IICSn)

ICAn : IICA shift register n

IICWLn : IICA low-level width setting register n

ICWHn : IICA high-level width setting register n

t_F : SDAAn and SCLAn signal falling times

f_{MCK} : IICA operation clock frequency

2. n=0,1

(2) When communication reservation is disabled (bit 0 (IICRSVn) of IICA flag register n (IICFn) = 1)

When bit 1 (STTn) of IICA control register n0 (IICCTLn0) is set to 1 when the bus is not used in a communication during bus communication, this request is rejected and a start condition is not generated. The following two statuses are included in the status where bus is not used:

- When the arbitration result is neither master nor slave
- Not run as slave after receiving extension code (bit6 (LRELn) of IICCTLn0 register is set "1" without returning a reply, releasing bus after exiting communication)

The STCFn (bit 7 of the IICFn register) can be used to confirm whether a start condition is generated or a request is rejected. Since it takes 5 f_{MCK} clocks from the time when the STTn bit is "1" to the time when the STCFn bit is set to "1", this time must be ensured by software.

Remark n=0,1

19.5.15 Other cautions

(1) When STCENn bit is "0"

Immediately after I²C is allowed to run (IICEn=1), the actual bus state is considered as the communication state (IICBSYn=1). In order to perform the master control communication in a state where no stop condition is detected, the stop condition must be generated, and the master control communication is performed after the bus is released. For multi-master, master communication cannot be performed in a state where the bus is not released (no stop condition is detected). Generate stop conditions in the following order:

① Set the IICA control register n1 (IICCTLn1).
② Set the bit7(IICEn) of IICA control register n0(IICCTLn0) to "1.
③ Set bit0 (SPTn) of IICCTLn0 register to "1".

(2) When STCENn bit is "1"

Immediately after I²C is allowed to run (IICEn=1), the actual bus state is considered as a released state (IICBSYn=0,1). Therefore, it is necessary to confirm that the bus has been released in order not to destroy other communications when generating the first start condition (STTn=1).

(3) I2C communications with other equipment in progress

When the SDAAn pin is low and the SCLAn pin is high, if I²C is allowed to run and participate in communication halfway, the macro of I2C is considered to be SDAAn pin changed from high to low (start condition detected). If the value on the bus is a value recognized as an expansion code at this time, a reply is returned that interferes with I2C communication. To avoid this, you must start I²C in the following order:

① The bit4 (SPIEn) of the IICCTLn0 register is cleared "0", and the interrupt request signal (INTIICAn) is prohibited.
② Set bit7(IICEn) of the IICCTLn0 register to '1' to allow I²C.
③ Wait for the start condition to be detected.
④ Set bit 6 (LRELn) of the IICCTLn0 register to 1 before ACK is returned (4 to 72 cycles of f_{MCK} after setting the IICEn bit to 1), to forcibly disable detection.

(4) After the STTn bit and the SPTn bit (bit1 and bit0 of the IICCTLn0 register) are set, reset before "0" is prohibited.

(5) If a communication reservation is made, the SPIEn bit (bit4 of the IICCTLn0 register) must be set "1" to generate an interrupt request. After generating an interrupt request, the transmission is started by writing communication data to the IICA shift register n (IICAn). If an interrupt does not occur when a stop condition is detected, the wait state is stopped because an interrupt request is not generated at the start of communication. However, it is not necessary to set the SPIEn bit to 1 when the MSTSn bit (bit 7 of the IICA status register n (IICSn)) is detected by software.

Remark n=0,1

19.5.16 Communication operation

The following 3 run steps are shown through the flowchart.

(1) Master operation of single master control system

The flow chart used as the master device in a single master system is shown below.

This process is broadly divided into Initial Settings and Communication Processing." The Initial Set-up section is executed at start-up, and the Communication Processing section is executed after preparation required for communication if communication with the slave device is required.

(2) Master operation of multi-master systems

In the multi-master system of I2C bus, it is impossible to judge whether the bus is released or in use. Here, if the data and the clock are at a high level within a certain period of time (1 frame), the bus is taken into communication as a released state. This process is broadly divided into Initial Set-up, Communication Waiting, and Communication Handling. A process that is designated as a slave device due to an arbitration failure is omitted here, only for use as the master device. After executing the "Initial Settings" section at start-up, add to the bus and wait for communication requests from the master or slave. The actual communication is the "communication processing" section, which supports arbitration with other master devices in addition to data transmission and reception with the slave devices.

(3) Slave operation

An example of using as an I2C bus slave is shown below.

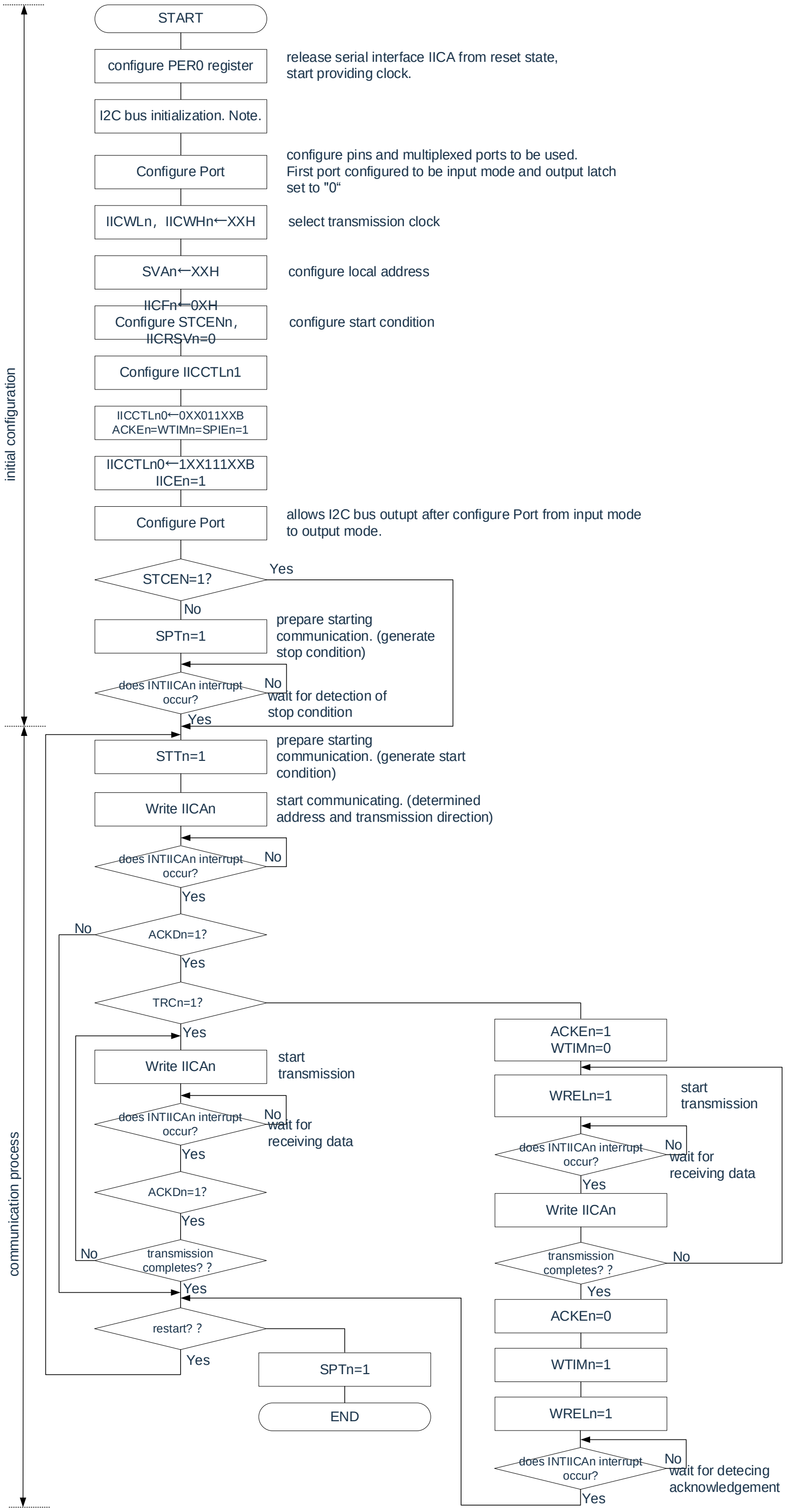
When used as a slave, the operation starts with an interrupt. Perform the Initial Settings section at start-up, and then wait for the INTIICAn interrupt by "Communication Waiting." If a INTIICAn interrupt occurs, a communication state is determined and a flag is transmitted to a main processing section.

Perform the required "communication processing" by examining the respective flags.

Remark n=0,1

(1) Master operation of single master control system

Figure 19-27 Master operation of single master control system

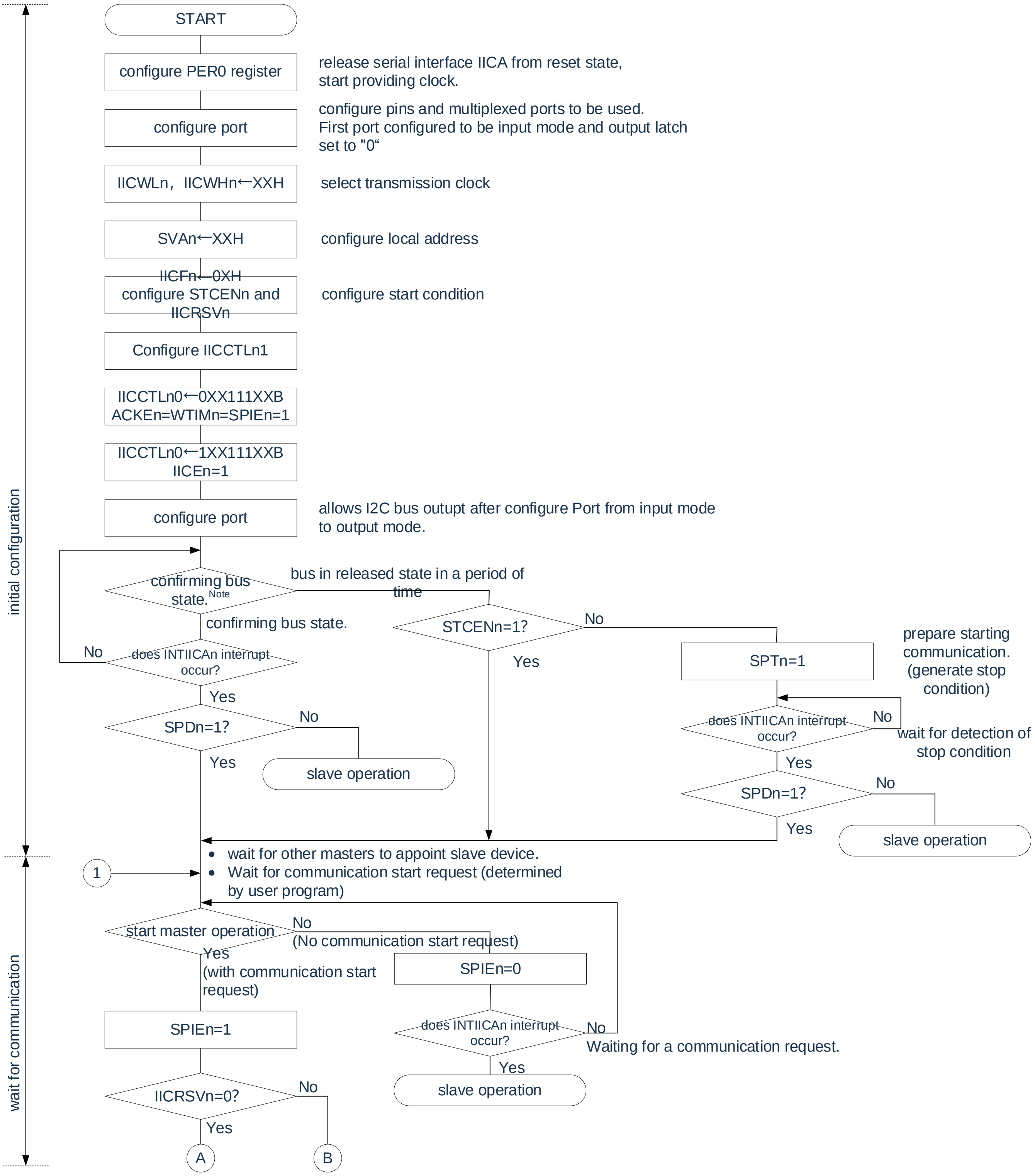

initial configuration
START
configure PER0 register
release serial interface IICA from reset state,
start providing clock.
I2C bus initialization. Note.
Configure Port
configure pins and multiplexed ports to be used.
First port configured to be input mode and output latch set to "0"
IICWLn，IICWHn←XXH
select transmission clock
SVAn←XXH
configure local address
IICFn←0XH
Configure STCENn，
IICRSVn=0
configure start condition
Configure IICCTLn1
IICCTLn0←0XX011XXB
ACKEn=WTIMn=SPIEn=1
IICCTLn0←1XX111XXB
IICEn=1
Configure Port
allows I2C bus outupt after configure Port from input mode to output mode.
STCEN=1?
Yes
No
SPTn=1
prepare starting communication. (generate stop condition)
does INTIICAn interrupt occur?
No
wait for detection of stop condition
Yes
communication process
STTn=1
prepare starting communication. (generate start condition)
Write IICAn
start communicating. (determined address and transmission direction)
does INTIICAn interrupt occur?
No
Yes
No
ACKDn=1?
Yes
TRCn=1?
Yes
Write IICAn
start transmission
does INTIICAn interrupt occur?
No
wait for receiving data
Yes
ACKDn=1?
Yes
No
transmission completes? ?
Yes
restart? ?
Yes
SPTn=1
END
ACKEn=1
WTIMn=0
WRELn=1
start transmission
does INTIICAn interrupt occur?
No
wait for receiving data
Yes
Write IICAn
transmission completes? ?
No
Yes
ACKEn=0
WTIMn=1
WRELn=1
does INTIICAn interrupt occur?
No
wait for detecing acknowledgement
Yes

Note: The I2C bus (SCLAn and SDAAn pins are high level) must be released based on the specification of the product in communication. For example, if the EEPROM is in a low level output to the SDAAn pin, the SCLAn pin must be set as the output port.

Remark 1. The format sent and received must conform to the specifications of the product in communication.
2. n=0,1

(2) Master operation of multi-master systems

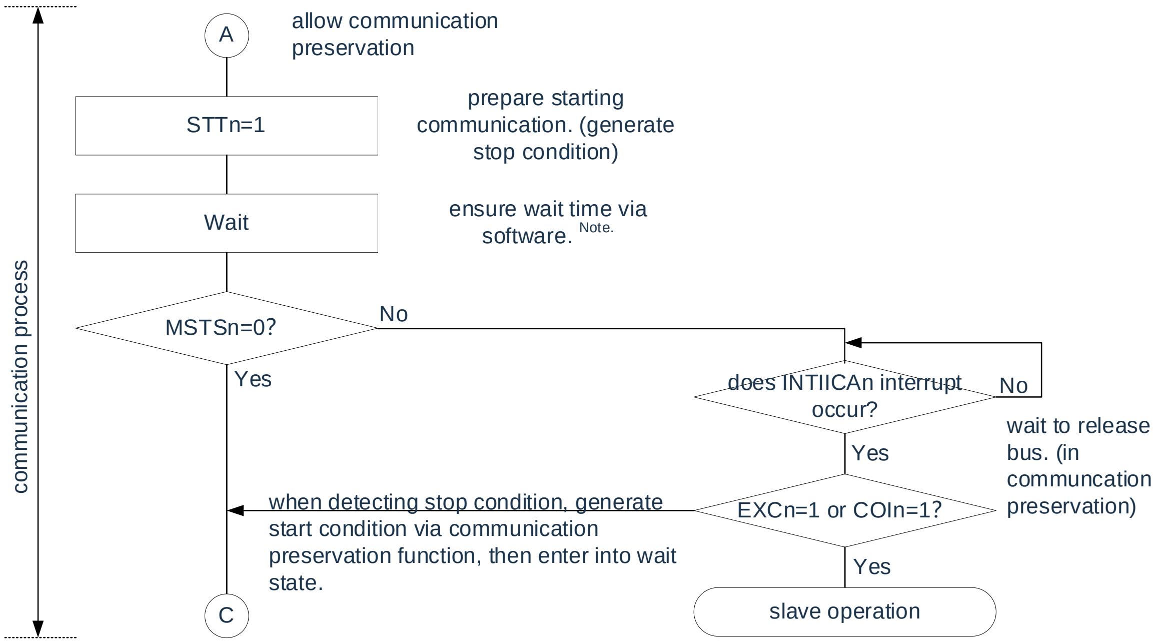
Figure 19-28 Master operation of multi-master systems (1/3)


initial configuration
wait for communication
START
configure PER0 register
release serial interface IICA from reset state, start providing clock.
configure port
configure pins and multiplexed ports to be used. First port configured to be input mode and output latch set to "0“
IICWLn，IICWHn←XXH
select transmission clock
SVAn←XXH
configure local address
IICFn←0XH configure STCENn and IICRSVn
configure start condition
Configure IICCTLn1
IICCTLn0←0XX111XXB ACKEn=WTIMn=SPIEn=1
IICCTLn0←1XX111XXB IICEn=1
configure port
allows I2C bus outupt after configure Port from input mode to output mode.
confirming bus state.Note
bus in released state in a period of time
confirming bus state.
No
does INTIICAn interrupt occur?
Yes
STCENn=1?
No
Yes
SPTn=1
prepare starting communication. (generate stop condition)
does INTIICAn interrupt occur?
No
wait for detection of stop condition
Yes
SPDn=1?
No
Yes
slave operation
SPDn=1?
No
Yes
slave operation
1
wait for other masters to appoint slave device.
Wait for communication start request (determined by user program)
start master operation
No
(No communication start request)
Yes
(with communication start request)
SPIEn=0
SPIEn=1
does INTIICAn interrupt occur?
No
Waiting for a communication request.
Yes
slave operation
IICRSVn=0?
No
Yes
A
B

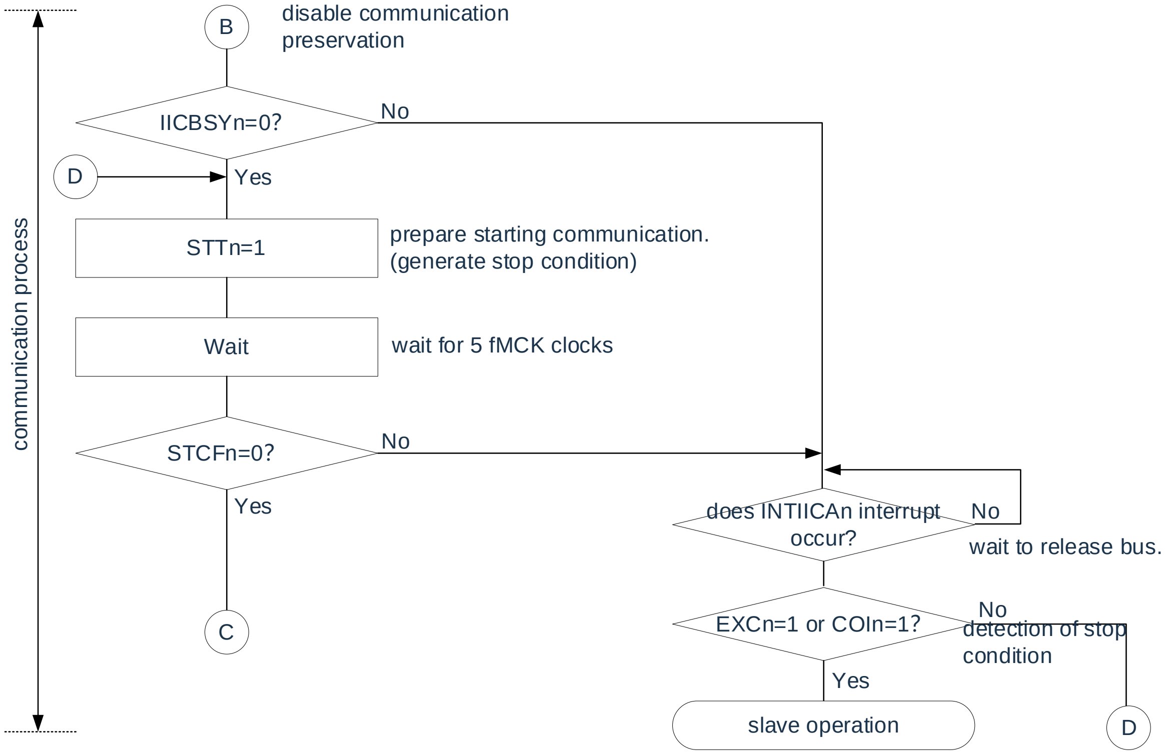
Note：You must verify that the bus is in a released state for a certain time (for example, 1 frame) (CLDn bit=1, DADn bit=1). When the SDAAn pin is fixed at a low level, it must be determined whether to release the I^{2}C bus (SCLAn and SDAAn pins are high).

Remark: n=0,1

Figure 19-28 Master operation of multi-master systems (2/3)

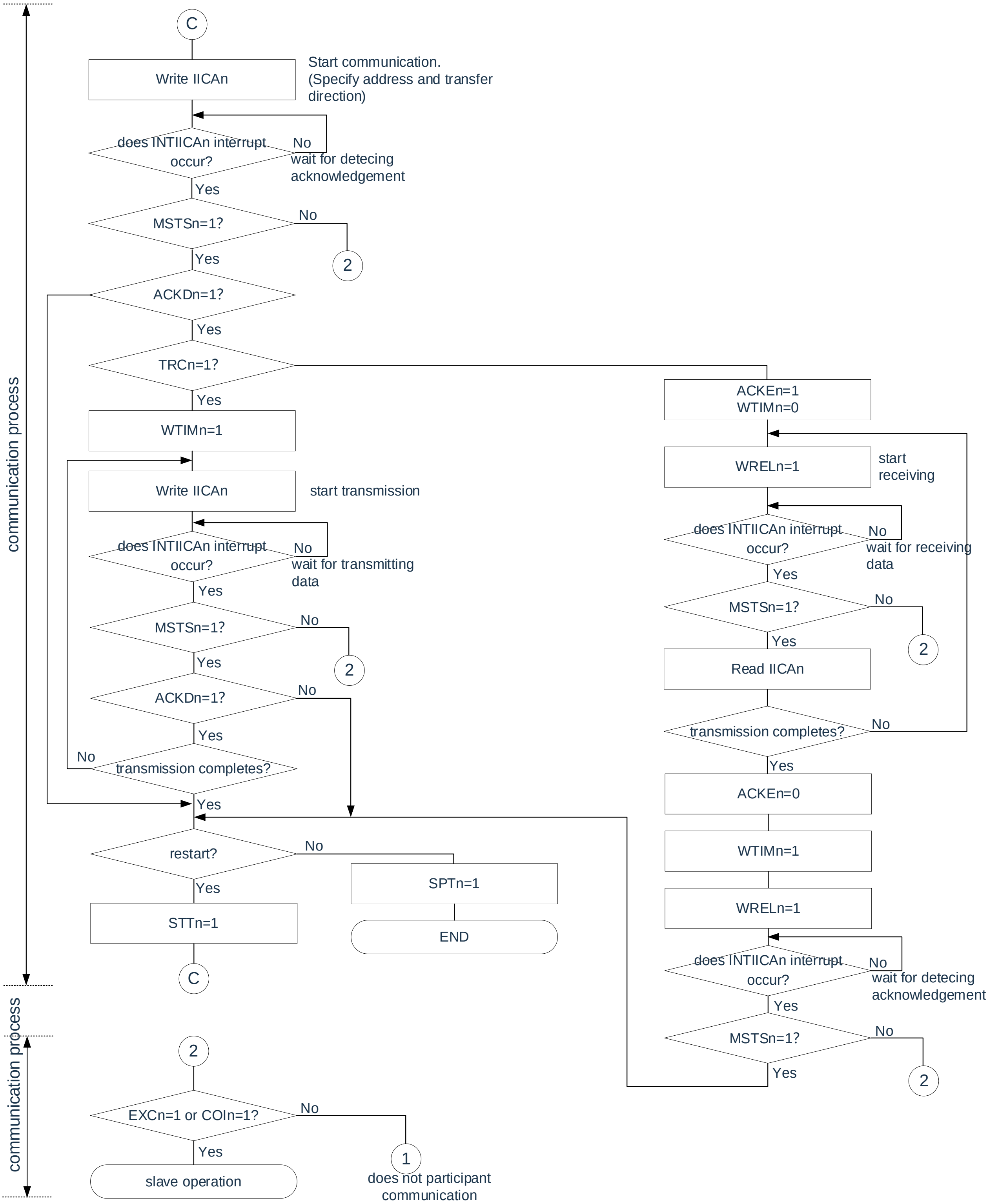

communication process
A
allow communication preservation
STTn=1
prepare starting communication. (generate stop condition)
Wait
ensure wait time via software. Note.
MSTSn=0?
No
Yes
does INTIICAn interrupt occur?
No
wait to release bus. (in communcation preservation)
Yes
EXCn=1 or COIn=1?
when detecting stop condition, generate start condition via communication preservation function, then enter into wait state.
Yes
slave operation
C

Note wait time as following:
(IICWLn configured value+IICWHn configured value+4)/fMCK+tF× 2


communication process
B
disable communication preservation
IICBSYn=0?
No
D
Yes
STTn=1
prepare starting communication. (generate stop condition)
Wait
wait for 5 fMCK clocks
STCFn=0?
No
Yes
does INTIICAn interrupt occur?
No
wait to release bus.
EXCn=1 or COIn=1?
No
detection of stop condition
Yes
slave operation
C
D

Note: 1. IICWLn :IICA low-level width setting register n
ICWHn :IICA high-level width setting register n
t_F SDAAn and SCLAn signal falling times
f_{MCK} : IICA operation clock frequency
2. n=0,1

Figure 19-28 Master operation of multi-master systems (3/3)



Remark: 1. The format of delivery and receipt must conform to the specifications of the product in communication.

2. In the case of multi-master system as master device, the MSTSn bit must be read each time INTIICAn interrupt occurs to confirm arbitration results.

3. In case d is used as slave in a multi-master system, the state must be confirmed by IICA state register n (IICFn) and IICA flag register n.

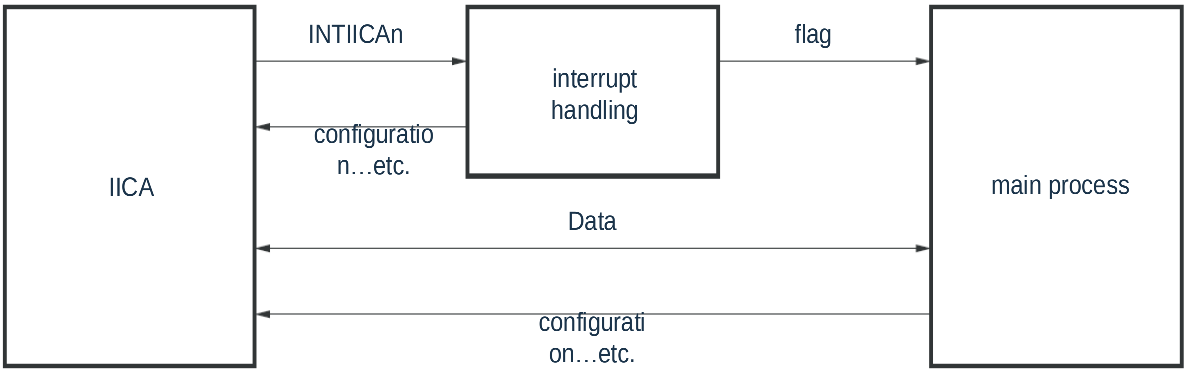
4. n=0,1

(3) Slave operation

The processing steps for salve operation are as follows.

Slave operation is essentially event driven and therefore require processing through INTIICAn interrupts (requiring significant change processing of operational states such as stop condition detection in communications).

In this specification, assuming that data communication does not support an expander code, the INTIICAn interrupt process only performs a state transfer process and the actual data communication is performed by the main process unit.



Therefore, the following three marks are prepared and the marks are transferred to the main processing unit instead of INTIICAn for data communication processing.

(1) Communication mode flag

This flag indicates the following two communication states:

- Clear mode: No status for data communication
- Communication mode: The state of data communication in progress (detection of valid addresses ~ detection of stop conditions, no response detected from master devices, different addresses)

(2) Ready flag

This flag indicates that data communication is possible. In normal data communication, it is set by the interrupt processing section and cleared by the main processing section as in the case of the INTIICAn interrupt. When communication starts, the flag is cleared by the interrupt processing section. However, when the 1st data is sent, the interrupt processing section does not set the ready flag, so the 1st data is sent without clearing the flag (address matching is interpreted as the next data request).

(3) Communication direction flag

This flag indicates the direction of communication and is the same value as the TRCn bit.

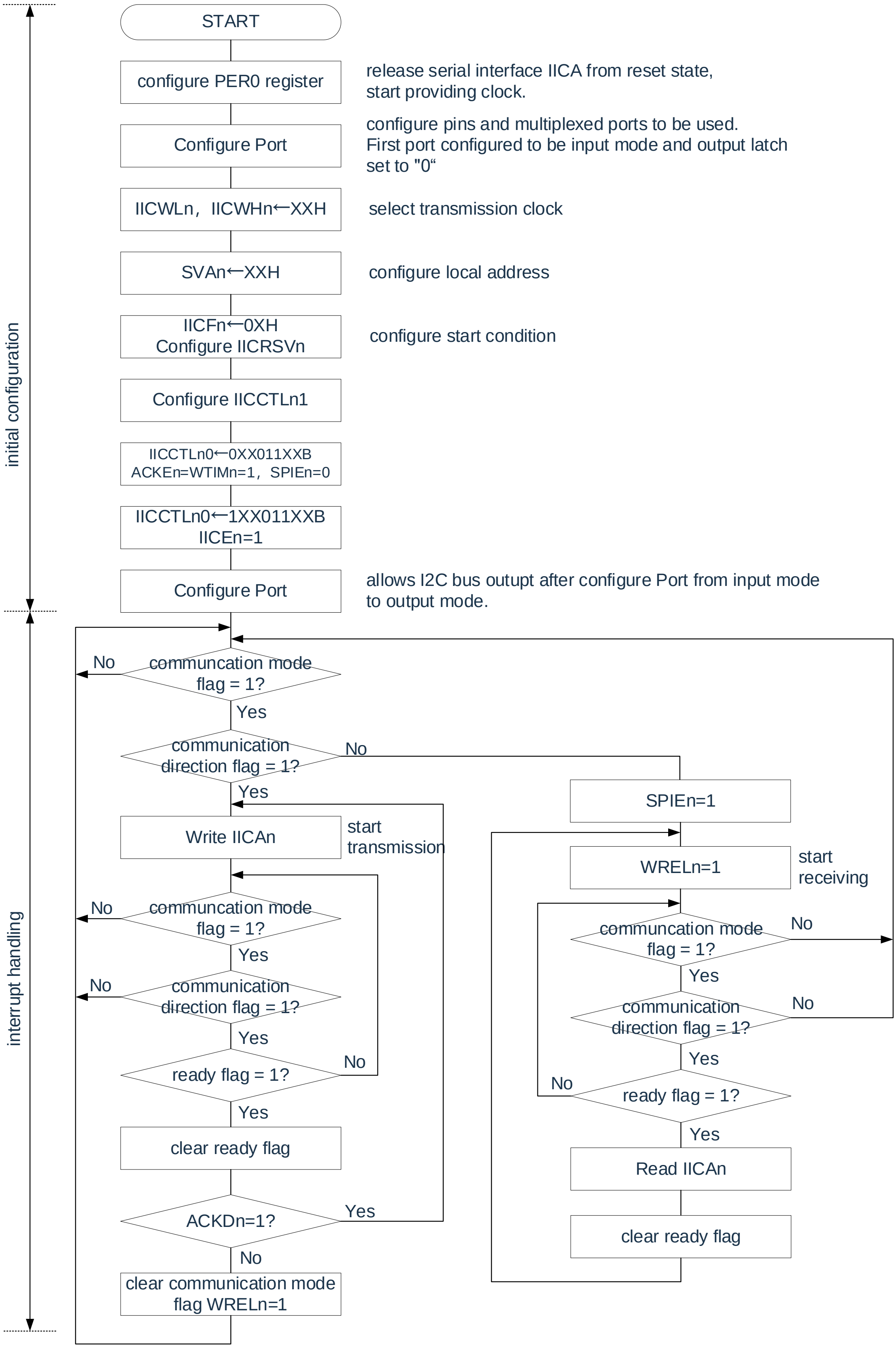
Remark n=0,1

The main process part of the slave operation is run as follows.

Start the serial interface IICA, and wait to become communicable. If the state becomes communicable, the communication mode flag and the ready flag are used to communicate (the state is confirmed here by the flag because the stop condition and the start condition are processed by interrupt).

At the time of transmission, the transmission is repeated until the master device does not return an acknowledgement. If the master does not return a reply, the communication is terminated. Upon receipt, a desired amount of data is received. If the communication ends, no reply is returned on the next data. Thereafter, the master device generates a stop condition or a restart condition, thereby exiting the communication state.

Figure 19-29 Slave operation step (1)


START
configure PER0 register
release serial interface IICA from reset state,
start providing clock.
Configure Port
configure pins and multiplexed ports to be used.
First port configured to be input mode and output latch set to "0"
IICWLn，IICWHn←XXH
select transmission clock
SVAn←XXH
configure local address
IICFn←0XH
Configure IICRSVn
configure start condition
Configure IICCTLn1
IICCTLn0←0XX011XXB
ACKEn=WTIMn=1，SPIEn=0
IICCTLn0←1XX011XXB
IICEn=1
Configure Port
allows I2C bus outuptafter configure Port from input mode to output mode.
initial configuration
interrupt handling
communcation mode flag = 1?
No
Yes
communication direction flag = 1?
No
Yes
Write IICAn
start transmission
communcation mode flag = 1?
No
Yes
communication direction flag = 1?
No
Yes
ready flag = 1?
No
Yes
clear ready flag
ACKDn=1?
Yes
No
clear communication mode flag WRELn=1
SPIEn=1
WRELn=1
start receiving
communcation mode flag = 1?
No
Yes
communication direction flag = 1?
No
Yes
ready flag = 1?
No
Yes
Read IICAn
clear ready flag

Note 1. The format of delivery and receipt must conform to the specifications of the product in communication.

2.n=0,1

An example of the steps for a slave device to process through INTIICAn interrupts is shown below (assuming no processing with an expander in this scenario). The status is interrupted by INTIICAn and the following processing is performed.

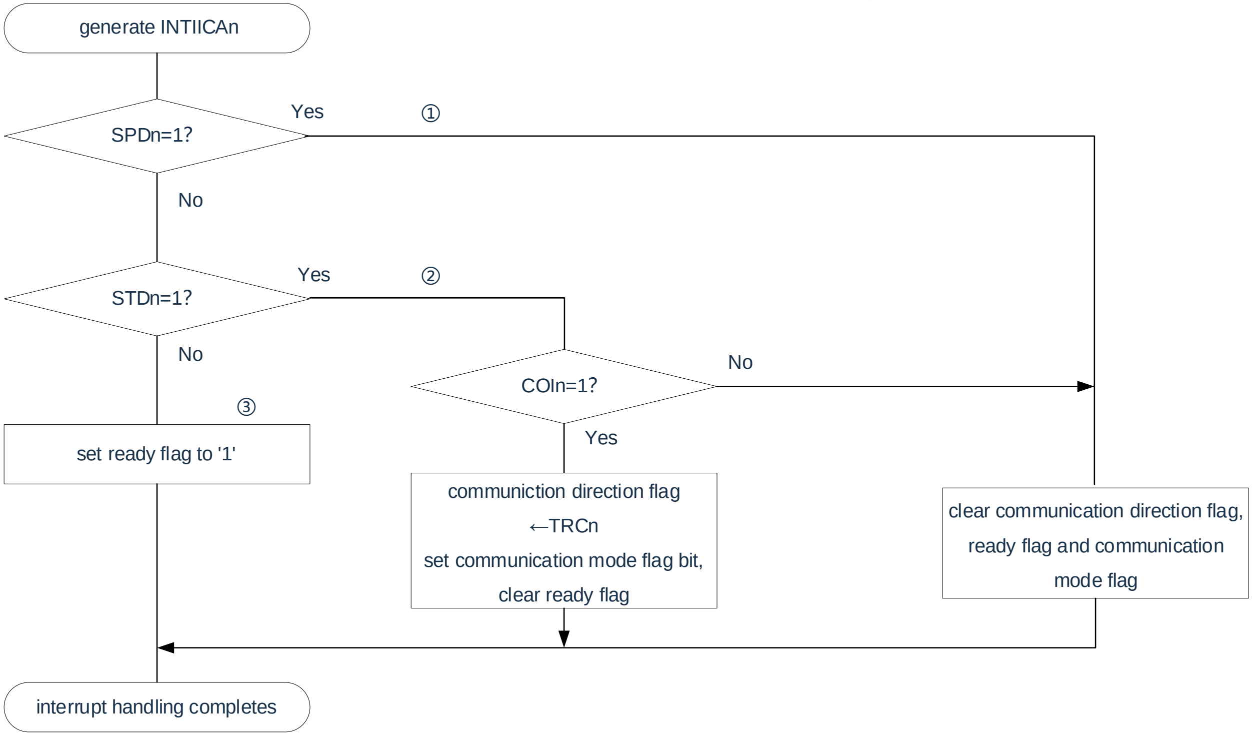
① If a stop condition is generated, the communication is ended.

② If you generate a start condition, confirm the address. If the addresses are different, the communication is terminated. If the addresses are the same, set to communication mode and unwaits, and returns from the interrupt (clear the readiness flag).

③ When sending and receiving data, the I2C bus remains in a waiting state and returns from an interrupt as long as the ready flag is set.

Remark The above ①~③ correspond to ①~③ of "Figure 19 30 slave operation step (2)".

Figure 19-30 Slave operation step (2)


generate INTIICAn
SPDn=1?
Yes
①
No
STDn=1?
Yes
②
No
③
set ready flag to '1'
COIn=1?
No
Yes
communiction direction flag ←TRCn
set communication mode flag bit,
clear ready flag
clear communication direction flag,
ready flag and communication mode flag
interrupt handling completes

Remark n=0,1

19.5.17 Timing of I^{2}C interrupt request (INTIICAn) generation (INTIICAn)

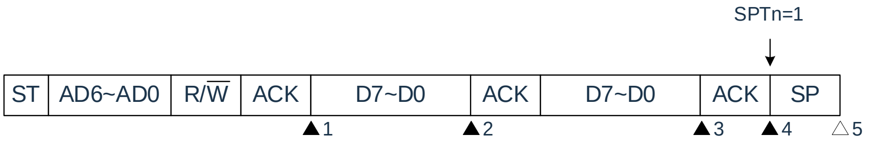
The values for the sending and receiving sequence of the data, the generation sequence of the INTIICAn interrupt request signal, and the IICA status register n (IICSn) when the INTIICAn signal is generated are shown below.

Note 1. ST : Start Condition
AD6~AD0: Address
$R/\overline{W}$: Assignment of Routing Direction
ACK : Acknowledgement
D7~D0 : data
SP : stop condition
2. n=0,1

(1) Master operation

(a) Start~Address~Data~Data~Stop (Transmit and receive)

(i) When WTIMn=0,1

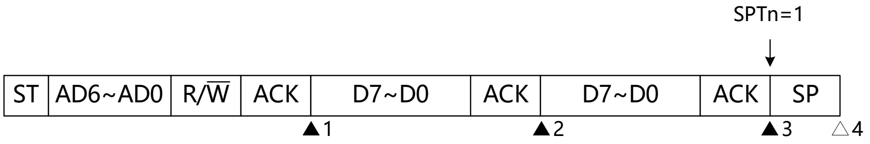


▲1:IICSn=1000X110B
▲2:IICSn=1000X000B
▲3:IICSn=1000X000B(set WTIMn bit to“1”) Note
▲4:IICSn=1000XX00B(set SPTn bit to “1”)
△5:IICSn=00000001B

Note: to generate stop condition, must set WTIMn bit to '1' and modify INTIICAn interrupt requet signal generation timing sequence.

Remark	▲	must generate
	△	only generate while SPIEn bit is '1'
	X	any

(ii) When WTIMn=1



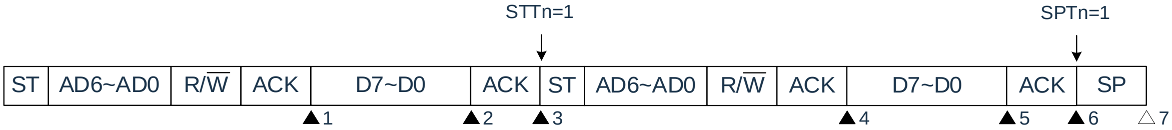
▲1:IICSn=1000X110B
▲2:IICSn=1000X100B
▲3:IICSn=1000XX00B(set SPTn bit to “1”)
△4:IICSn=00000001B

Remark	▲	must generate
	△	only generate while SPIEn bit is '1'
	X	any

Remark n=0,1

(b) Start~Address~Data~Start~Address~Data~Stop (Restart)

(i) When WTIMn=0,1



▲1:IICSn=1000X110B

▲2:IICSn=1000X000B(set WTIMn bit to“1”) Note1

▲3:IICSn=1000XX00B(set WTIMn bit to “0”. Note 2 and set STTn bit to“1”)

▲4:IICSn=1000X110B

▲5:IICSn=1000X000B(set WTIMn bit to“1”) Note3

▲6:IICSn=1000XX00B(set SPTn bit to “1”)

△7:IICSn=00000001B

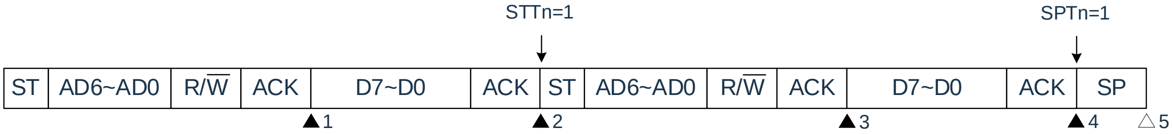
Note1. to generate start condition, must set WTIMn bit to '1' and modify INTIICAn interrtupt request signal generation timing sequence.

2. To recover original configuration, must set WTIMn bit to '0'.

3. to generate stop condition, must set WITIMn bit to '1' and modify INTIICAn interrupt request signal generation timing sequence.

Remark	▲	must generate
	△	only generate while SPIEn bit is '1'
	X	Any

(ii) When WTIMn=1



▲1:IICSn=1000X110B

▲2:IICSn=1000XX00B(set STTn bit to “1”)

▲3:IICSn=1000X110B

▲4:IICSn=1000XX00B(set SPTn bit to “1”)

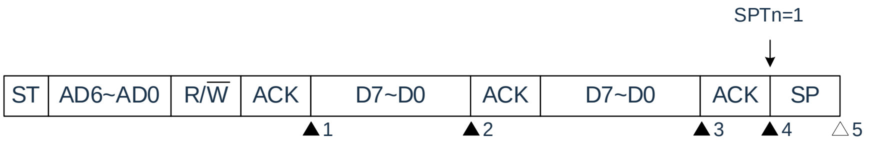
△5:IICSn=00000001B

Remark	▲	must generate
	△	only generate while SPIEn bit is '1'
	X	any

Remark n=0,1

(c) Start~Code~Data~Data~Stop (Transmit extension code)

(i) When WTIMn=0,1



▲1:IICSn=1010X110B

▲2:IICSn=1010X000B

▲3:IICSn=1010X000B(set WTIMn bit to“1”) Note

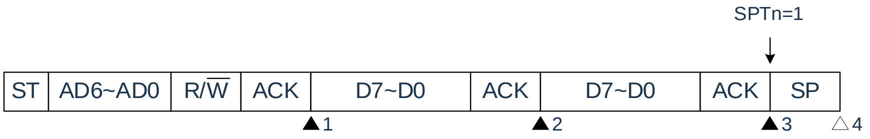
▲4:IICSn=1010XX00B(set SPTn bit to “1”)

△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) When WTIMn=1



▲1:IICSn=1010X110B

▲2:IICSn=1010X100B

▲3:IICSn=1010XX00B(set SPTn bit to “1”)

△4:IICSn=00000001B

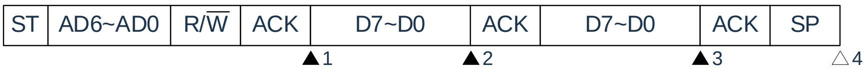
Remark		
	▲	must generate
	△	only generate while SPIEn bit is '1'
	X	any

Remark n=0,1

(2) Slave operation (When receiving slave addresses)

(a) Start~Address~Data~Data~Stop

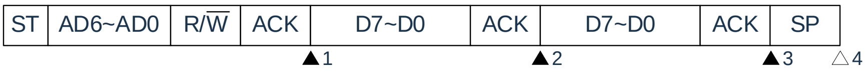
(i) When WTIMn=0,1



▲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) When WTIMn=1



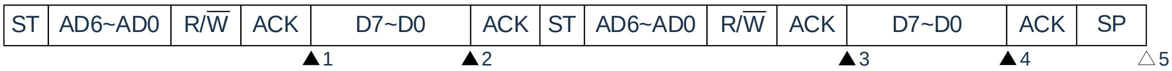
▲1:IICSn=0001X110B
▲2:IICSn=0001X100B
▲3:IICSn=0001XX00B
△4:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

Remark n=0,1

(b) Start~Address~Data~Start~Address~Data~Stop

(i) When WTIMn=0,1 (same as SVAn after restart)



▲1:IICSn=0001X110B

▲2:IICSn=0001X000B

▲3:IICSn=0001X110B

▲4:IICSn=0001X000B

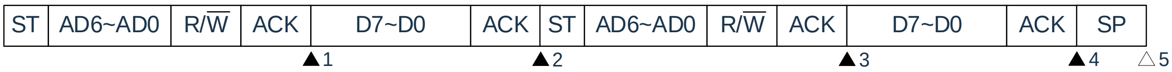
△5:IICSn=00000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

(ii) When WTIMn=1 (same as SVAn after restart)



▲1:IICSn=0001X110B

▲2:IICSn=0001XX00B

▲3:IICSn=0001X110B

▲4:IICSn=0001XX00B

△5:IICSn=00000001B

Remark ▲ must generate

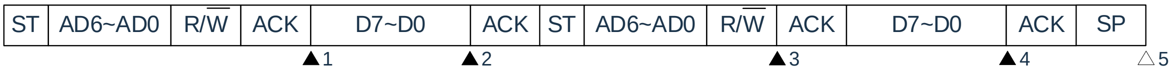
△ only generate while SPIEn bit is '1'

X any

Remark n=0,1

(c) Start~Address~Data~Start~Code~Data~Stop

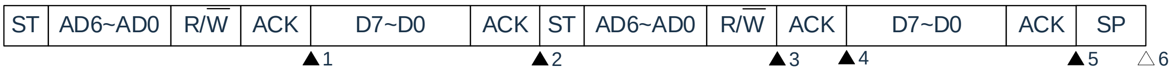
(i) When WTIMn=0,1 (different address after restart (extension code))



▲1:IICSn=0001X110B
▲2:IICSn=0001X000B
▲3:IICSn=0010X010B
▲4:IICSn=0010X000B
△5:IICSn=00000001B

Remark	▲	must generate
	△	only generate while SPIEn bit is '1'
	X	any

(ii) When WTIMn=1 (different address after restart (extension code))



▲1:IICSn=0001X110B
▲2:IICSn=0001XX00B
▲3:IICSn=0010X010B
▲4:IICSn=0010X110B
▲5:IICSn=0010XX00B
△6:IICSn=00000001B

Remark	▲	must generate
	△	only generate while SPIEn bit is '1'
	X	any

Remarks: n=0,1

(d) Start~Address~Data~Start~Address~Data~Stop

(i) When WTIMn=0,1 (different address after restart (not extension code))

ST	AD6~AD0	R/$\overline{W}$	ACK	D7~D0	ACK	ST	AD6~AD0	R/$\overline{W}$	ACK	D7~D0	ACK	SP
			▲1	▲2					▲3			△4

▲1:IICSn=0001X110B
▲2:IICSn=0001X000B
▲3:IICSn=00000110B
△4:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

(ii) When WTIMn=1 (different address after restart (not extension code))

ST	AD6~AD0	R/$\overline{W}$	ACK	D7~D0	ACK	ST	AD6~AD0	R/$\overline{W}$	ACK	D7~D0	ACK	SP
			▲1		▲2				▲3			△4

▲1:IICSn=0001X110B
▲2:IICSn=0001XX00B
▲3:IICSn=00000110B
△4:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

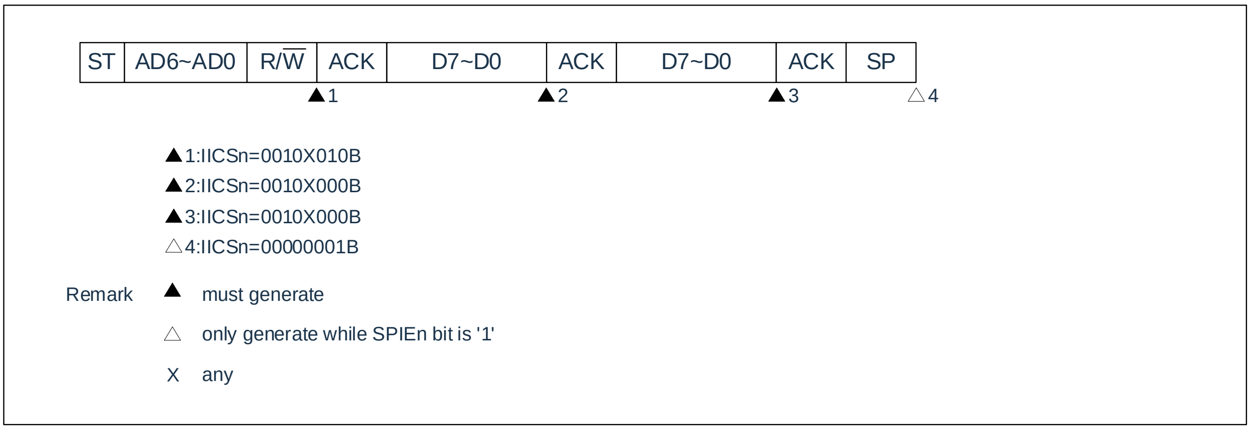
Remarks: n=0,1

(3) Slave operation (When receiving extension code)

Always participate in communication when you receive an expansion code.

(a) Start~Code~Data~Data~Stop

(i) When WTIMn=0,1



▲1:IICSn=0010X010B

▲2:IICSn=0010X000B

▲3:IICSn=0010X000B

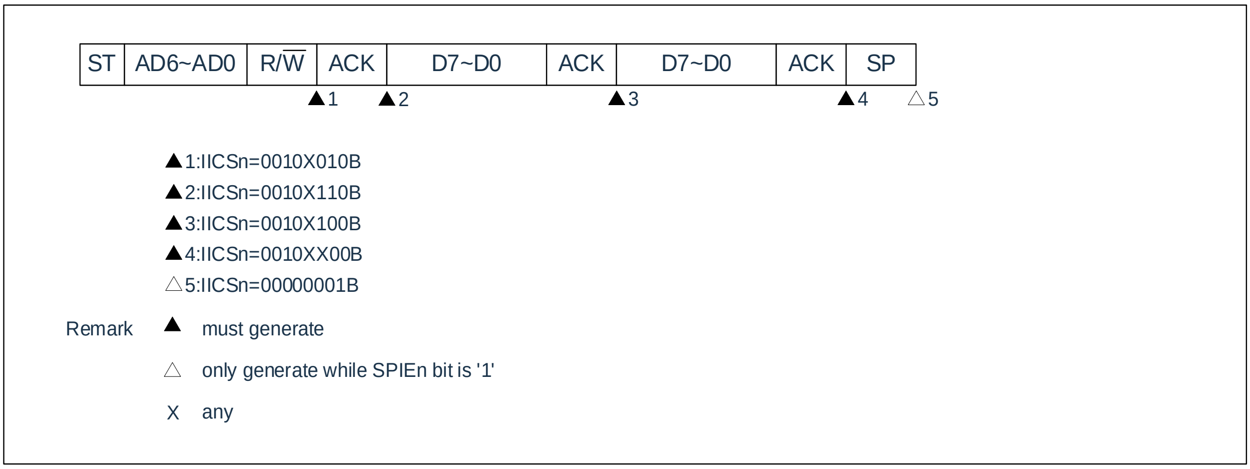
△4:IICSn=00000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

(ii) When WTIMn=1



▲1:IICSn=0010X010B

▲2:IICSn=0010X110B

▲3:IICSn=0010X100B

▲4:IICSn=0010XX00B

△5:IICSn=00000001B

Remark ▲ must generate

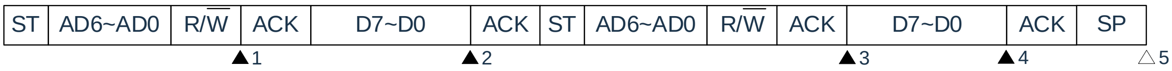
△ only generate while SPIEn bit is '1'

X any

Remarks: n=0,1

(b) Start~Code~Data~Start~Address~Data~Stop

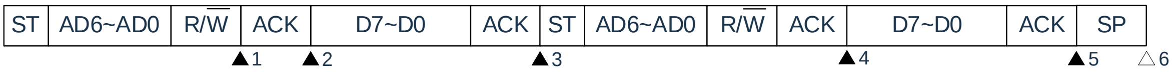
(i) When WTIMn=0,1 (Same as SVAn after restart)



▲1:IICSn=0010X010B
▲2:IICSn=0010X000B
▲3:IICSn=0001X110B
▲4:IICSn=0001X000B
△5:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

(ii) When WTIMn=1 (Same as SVAn after restart)



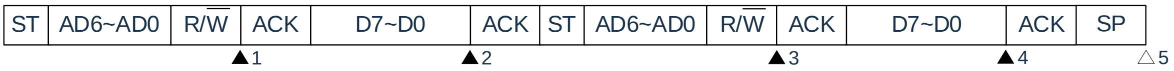
▲1:IICSn=0010X010B
▲2:IICSn=0010X110B
▲3:IICSn=0010XX00B
▲4:IICSn=0001X110B
▲5:IICSn=0001XX00B
△6:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

Remarks: n=0,1

(c) Start~Code~Data~Start~Code~Data~Stop

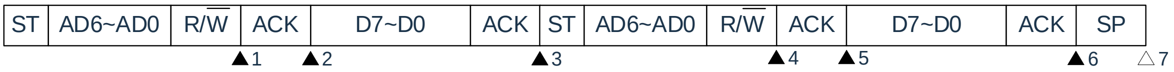
(i) When WTIMn=0,1 (Receive extension code after restart)



▲1:IICSn=0010X010B
▲2:IICSn=0010X000B
▲3:IICSn=0010X010B
▲4:IICSn=0010X000B
△5:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

(ii) When WTIMn=1 (Receive extension code after restart)



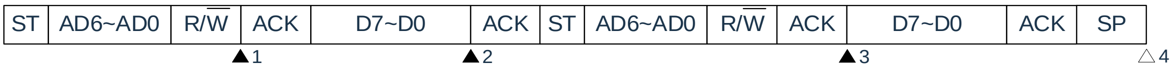
▲1:IICSn=0010X010B
▲2:IICSn=0010X110B
▲3:IICSn=0010XX00B
▲4:IICSn=0010X010B
▲5:IICSn=0010X110B
▲6:IICSn=0010XX00B
△7:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

Remarks: n=0,1

(d) Start~Code~Data~Start~Address~Data~Stop

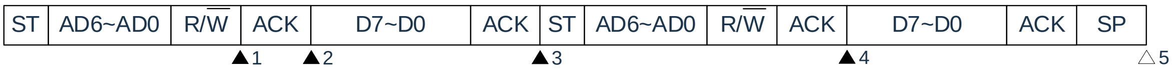
(i) When WTIMn=0,1 (Different address after restart (not extension code))



▲1:IICSn=0010X010B
▲2:IICSn=0010X000B
▲3:IICSn=00000X10B
△4:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

(ii) When WTIMn=1 (Different address after restart (not extension code))



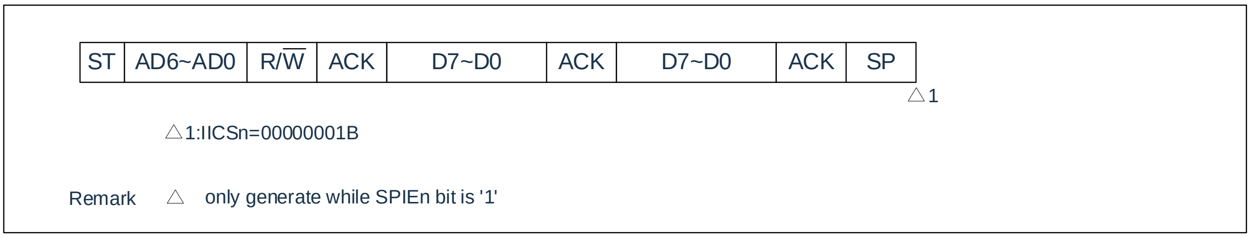
▲1:IICSn=0010X010B
▲2:IICSn=0010X110B
▲3:IICSn=0010XX00B
▲4:IICSn=00000X10B
△5:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

Remarks: n=0,1

(4) Operation without participation in communications

(a) Start~Code~Data~Data~Stop



△1:IICSn=00000001B

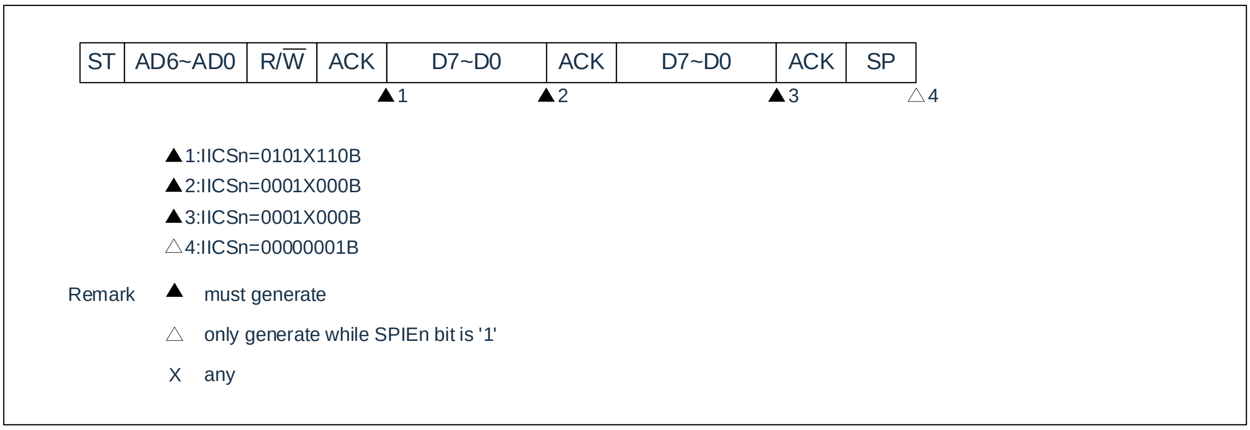
Remark △ only generate while SPIEn bit is '1'

(5) Failed arbitration operation (runs as a slave after a failed arbitration)

When used as a master device in a multi-master system, the MSTSn bit must be read each time a INTIICAn interrupt request signal is generated to confirm arbitration.

(a) Arbitration failure during sending slave address data

(i) When 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

Remarks: n=0,1

(ii) When WTIMn=1

ST	AD6~AD0	R/$\overline{W}$	ACK	D7~D0	ACK	D7~D0	ACK	SP
			▲1		▲2		▲3	△4

▲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) Arbitration failure during transmitting an extension code

(i) When WTIMn=0,1

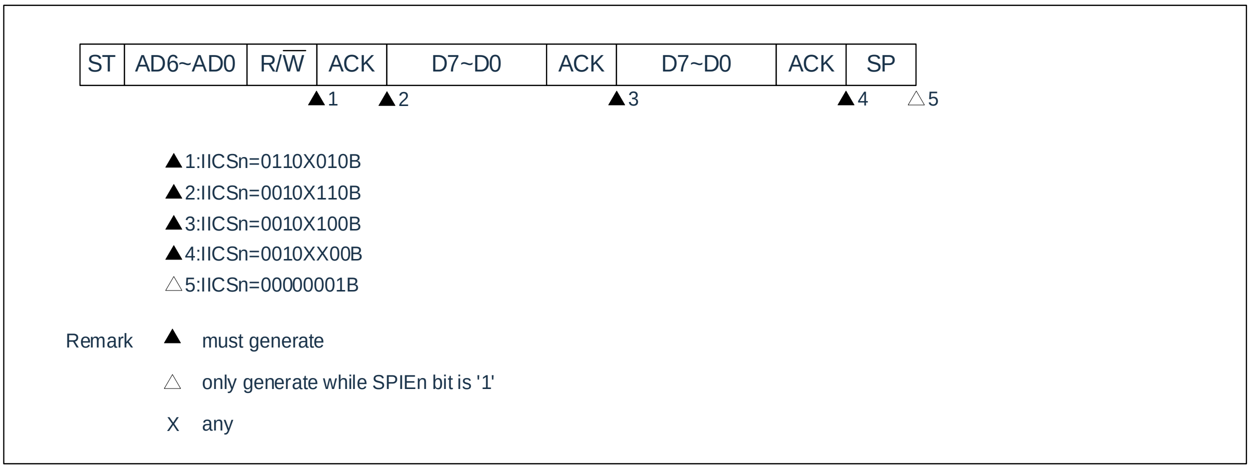
ST	AD6~AD0	R/$\overline{W}$	ACK	D7~D0	ACK	D7~D0	ACK	SP
		▲1		▲2		▲3		△4

▲1:IICSn=0110X010B
▲2:IICSn=0010X000B
▲3:IICSn=0010X000B
△4:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

Remarks: n=0,1

(ii) When WTIMn=1



ST	AD6~AD0	R/$\overline{W}$	ACK	D7~D0	ACK	D7~D0	ACK	SP
		▲1	▲2		▲3		▲4	△5

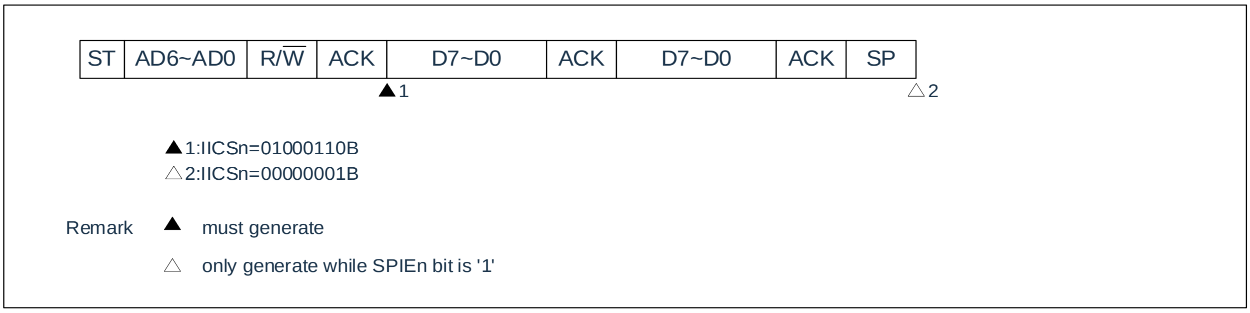
▲1:IICSn=0110X010B
▲2:IICSn=0010X110B
▲3:IICSn=0010X100B
▲4:IICSn=0010XX00B
△5:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

(6) Failed arbitration operation (not participating in communications after a failed arbitration)

When used as a master device in a multi-master system, the MSTSn bit must be read each time a INTIICAn interrupt request signal is generated to confirm arbitration.

(a) Arbitration failure in the process of sending slave address data (WTIMn=1)



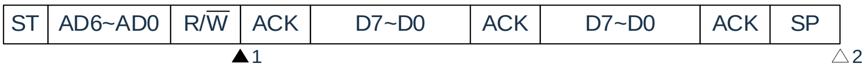
ST	AD6~AD0	R/$\overline{W}$	ACK	D7~D0	ACK	D7~D0	ACK	SP
			▲1					△2

▲1:IICSn=01000110B
△2:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'

Remarks: n=0,1

(b) Arbitration failure during the sending of an extension code



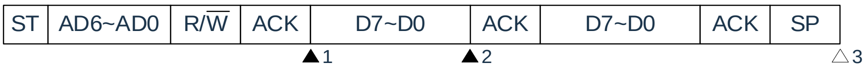
▲1:IICSn=01000110B
set LRELn bit to '1' via software
△2:IICSn=00000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

(c) Arbitration failure when transferring data

(i) When WTIMn=0,1



▲1:IICSn=10001110B
▲2:IICSn=01000000B
△3:IICSn=00000001B

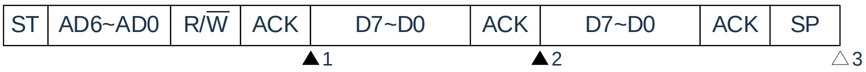
Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

Remarks: n=0,1

(ii) When WTIMn=1



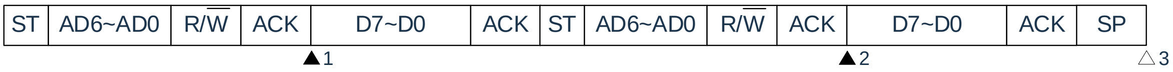
▲1:IICSn=10001110B
▲2:IICSn=01000100B
△3:IICSn=00000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

(d) Arbitration failure due to restart conditions while transmitting data

(i) Non-expansion codes (e.g., SVAn is different)



▲1:IICSn=1000X110B
▲2:IICSn=01000110B
△3:IICSn=00000001B

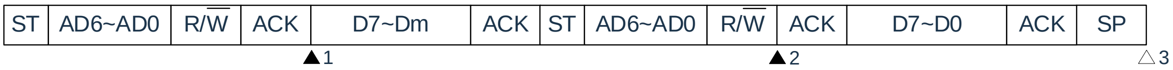
Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

Remarks: n=0,1

(ii) Expansion codes



▲1:IICSn=1000X110B
▲2:IICSn=01000010B
set LRELn bit to '1' via software
△3:IICSn=00000001B

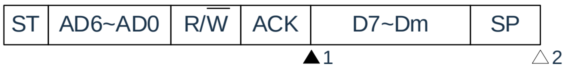
Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

m=0~6

(e) Arbitration failure due to stop conditions while transmitting data



▲1:IICSn=10000110B
△2:IICSn=01000001B

Remark ▲ must generate

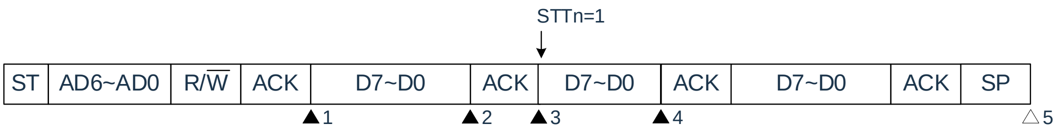
△ only generate while SPIEn bit is '1'

m=0~6

Remarks: n=0,1

(f) Arbitration failure due to low data when trying to generate a restart condition

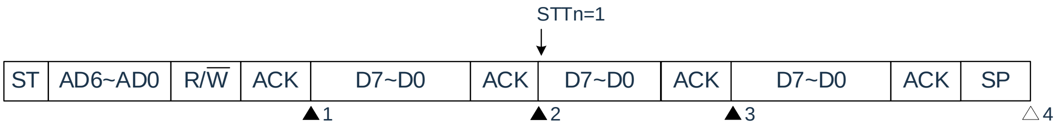
(i) When WTIMn=0,1



▲1:IICSn=1000X110B
▲2:IICSn=1000X000B(set WTIMn bit to“1”)
▲3:IICSn=1000X100B(set WTIMn bit to“0”)
▲4:IICSn=01000000B
△5:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

(ii) When WTIMn=1



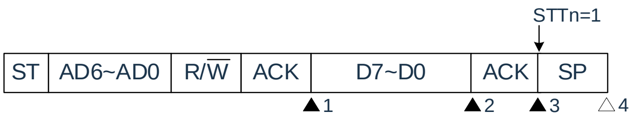
▲1:IICSn=1000X110B
▲2:IICSn=1000X100B(set STTn bit to “1”)
▲3:IICSn=01000100B
△4:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

Remarks: n=0,1

(g) Arbitration failure due to stop conditions when trying to generate restart conditions

(i) When WTIMn=0,1



▲1:IICSn=1000X110B

▲2:IICSn=1000X000B(set WTIMn bit to“1”)

▲3:IICSn=1000XX00B(set STTn bit to “1”)

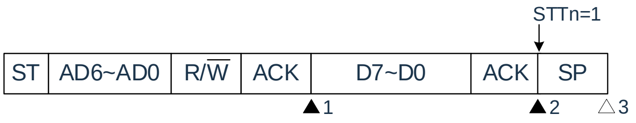
△4:IICSn=01000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

(ii) When WTIMn=1



▲1:IICSn=1000X110B

▲2:IICSn=1000XX00B(set STTn bit to “1”)

△3:IICSn=01000001B

Remark ▲ must generate

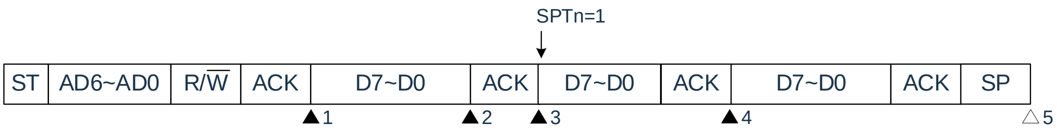
△ only generate while SPIEn bit is '1'

X any

Remarks: n=0,1

(h) Arbitration failure due to low data when trying to generate a stop condition

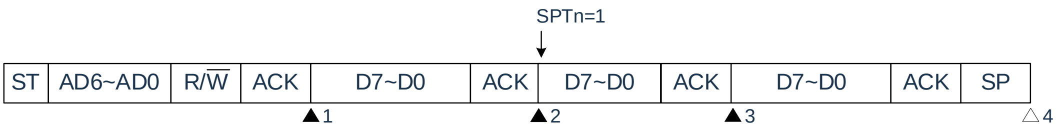
(i) When WTIMn=0,1



▲1:IICSn=1000X110B
▲2:IICSn=1000X000B(set WTIMn bit to“1”)
▲3:IICSn=1000X100B(set WTIMn bit to“0”)
▲4:IICSn=01000100B
△5:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

(ii) When WTIMn=1



▲1:IICSn=1000X110B
▲2:IICSn=1000X100B(set SPTn bit to “1”)
▲3:IICSn=01000100B
△4:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

Remarks: n=0,1

19.6 Timing diagram

In I2C bus mode, a master device selects a slave device as the communication object among a number of slave devices by outputting an address to a serial bus. The master device sends a TRCn bit (bit3 of the IICA status register n (IICSn) representing the data transmission direction after the slave device address and starts serial communication with the slave device. The timing diagram of the data communication is shown in Figure 19-31 and Figure 19-32.

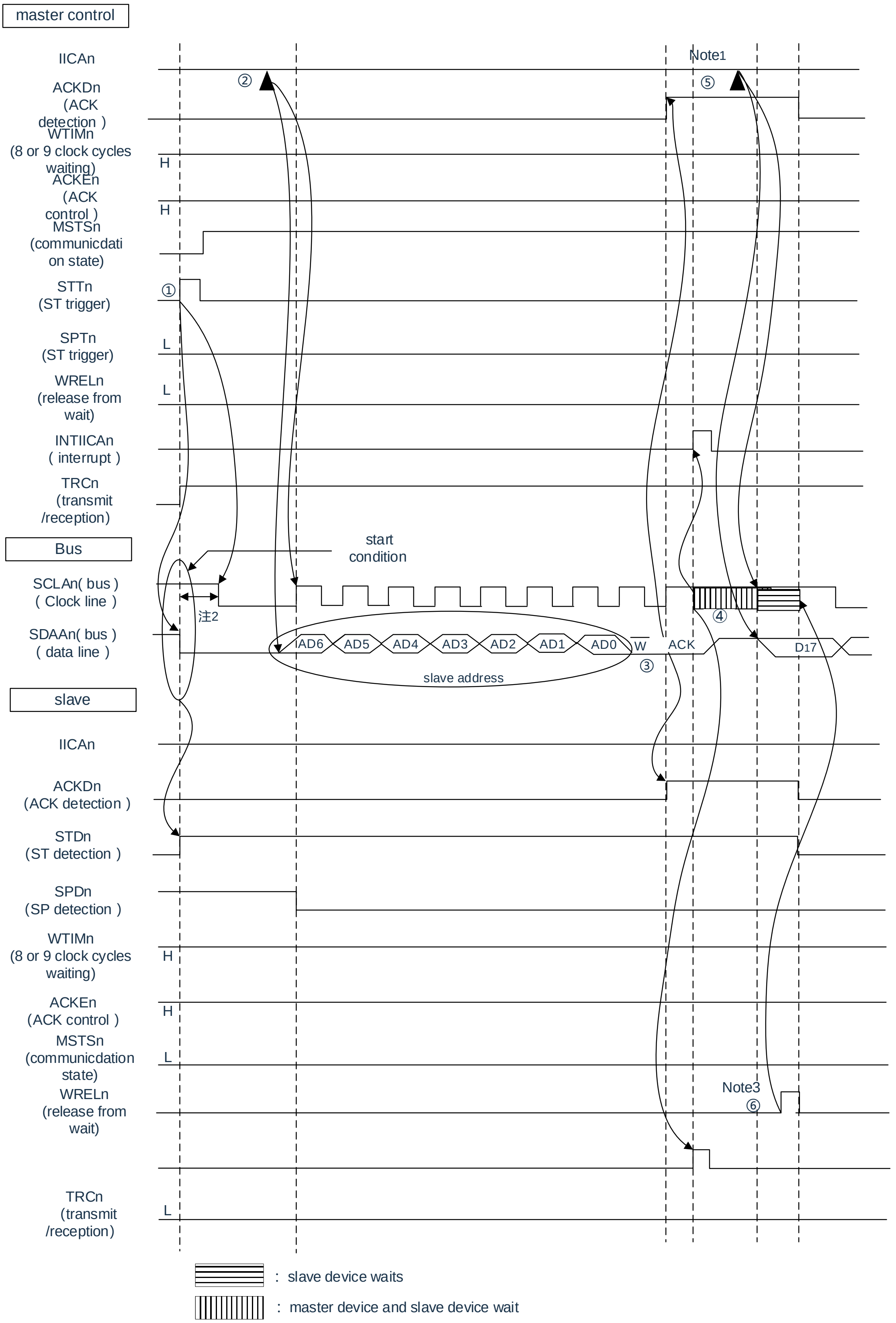
The IICA shift register n (IICAn) is shifted synchronously with the descent edge of the serial clock (SCLAn), and the transmission data is transmitted to the SO latch preferentially from the SDAAn pin.

Take the data entered by the SDAAn pin to IICAn at the rising edge of the SCLAn.

Remarks: n=0,1

Figure 19-31 Communication example of master → slave
(Master: Select 9 clock wait, slave: Select 9 clock wait) (1/4)

(1) Start Condition ~ Address ~ Data



▤ : slave device waits

▥ : master device and slave device wait

Note 1. To release the master from waiting during transmission, you must write data to the IICn instead of setting the WRELn bit.

2. The time to decrease the SDAAn pin signal to the SCLAn pin signal is at least 4.0μs when set to standard mode and at least 0.6μs when set to fast mode.

3. To release the slave from waiting during reception, the IICAn must be set to "FFH" or the WRELn bit must be set.

Figure 19-31 Descriptions of ① to ⑥ of (1) starting condition ~ address ~ data are as follows:

①If the master triggers the setting (STTn=1), the bus data line (SDAAn) drops and generates a start condition (SDAAn changes from 1 to 0). After that, if the start condition is detected, the master enters the master communication state (MSTSn=1), and after holding time the bus clock line drops (SCLAN=0,1), and the communication preparation ends.

②If the master sends the IICA shift register n(IICAn) a write address + W (send), the slave address is sent.

③On the secondary side, if the receiving address and the local station address (value of SVAn) are the same note, an ACK is sent to the host via hardware. The master detects ACK (ACKDn=1) at the rising edge of the 9th clock.

④The master generates an interrupt on the descending edge of the 9th clock (INTIICAn: End of address delivery interrupt). A slave device of the same address enters a waiting state (SCLAn=0,1) and an interrupt is generated (INTIICAn: Address Matching Interrupt) Note.

⑤The host writes sending data to the IICAn register, thus relieving the host from waiting.

⑥If the client unwaits (WRELn=1), the master starts transferring data to the client.

Note: If the address sent and the slave address are different, the slave party does not return ACK(NACK:SDAAn=1) to the host party and does not generate INTIICAn interrupts (address matching interrupts) and does not enter a wait state.

However, the master generates an INTIICAn interrupt for both the ACK and the NACK (the end of address sending interrupt).

Note 1. ①~⑮ in Figure 19-31 are a series of operation steps for data communication through I2C bus.

"(1) Start condition - address - data" in Figure 19-31describes steps ① to ⑥.

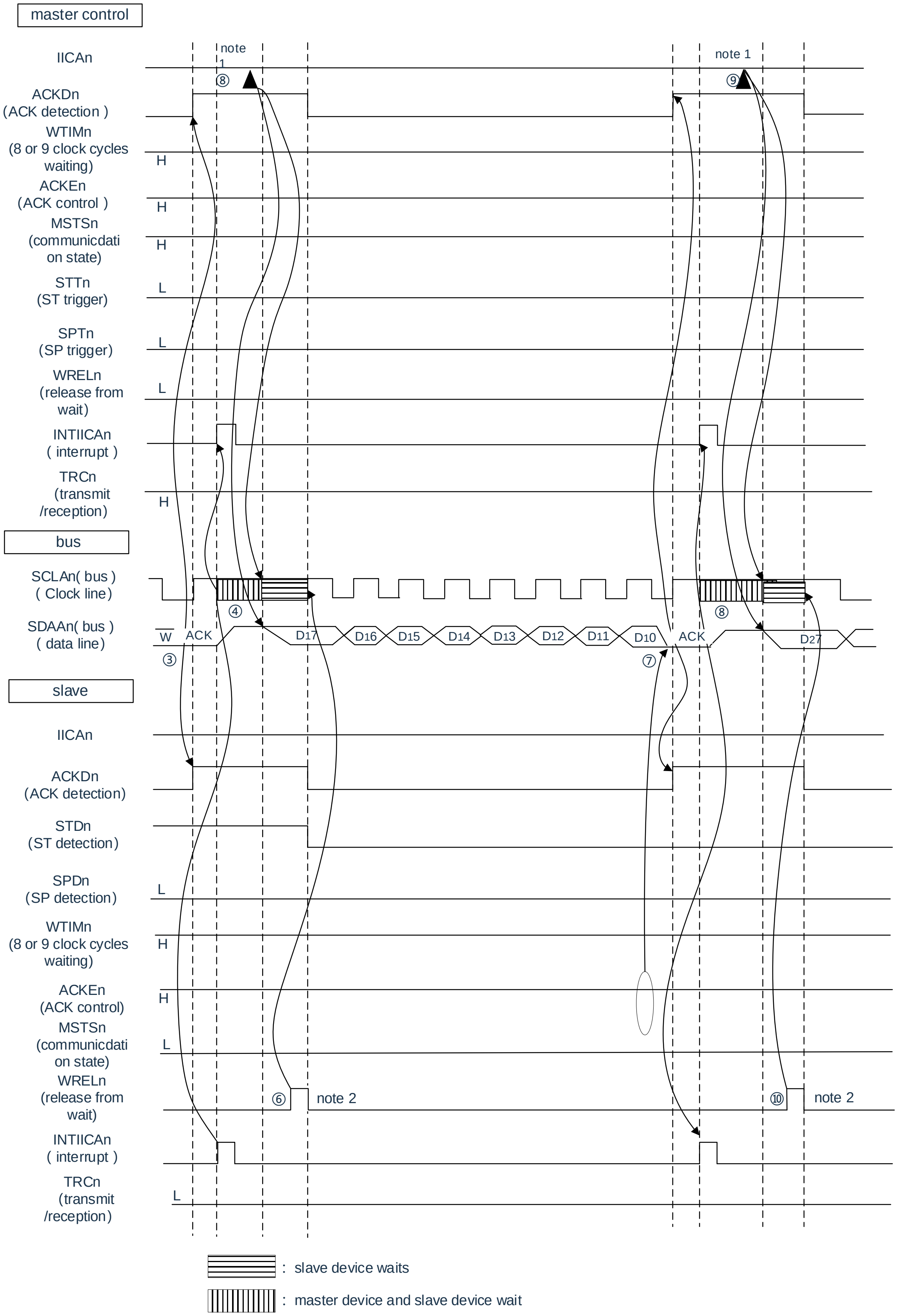
"(2) Address to data to data" in Figure 19-31 describes steps ③ to ⑩.

"(3) Data to data to stop conditions" in Figure 19-31 describes steps ⑦ to ⑮.

2.n=0,1

Figure 19-31 Communication example of master → slave
(Maste: Select 9 clock wait, slave: Select 9 clock wait) (2/4)

(2) Address~Data~Data



Note 1. To release the master from waiting during transmitting, you must write data to the IICAn instead of setting the WRELn bit.

2. To release the slave from waiting during reception, the IICAn must be set to “FFH” or the WRELn bit must be set.

Figure 19-31 Descriptions of ③ to ⑩ for (2) Address ~ Data ~ Data:

③In the secondary party, if the receiving address and the local station address (value of SVAn) are the same note, the ACK is sent to the master via hardware. The master detects ACK (ACKDn=1) at the rising edge of the 9th clock.

④The master generates an interrupt at the descending edge of the 9th clock (INTIICAn: End of address delivery interrupt). A slave device of the same address enters a waiting state (SCLAn=0,1) and an interrupt is generated (INTIICAn: Address Matching Interrupt) Note.

⑤The main controller writes and transmits data to IICA shift register n (IICAn), and eliminates the main controller waiting.

⑥If the slave party unwaits (WRELn=1), the master starts transferring data to the slave party.

⑦After the data transmission is finished, the slave side transmits ACK through hardware to the master side because the ACKEn bit is 1. The master detects ACK (ACKDn=1) at the rising edge of the 9th clock.

⑧The master and slave enter a waiting state (SCLAn=0,1) at the descending edge of the ninth clock, and both generate interrupts (INTIICAn: End of delivery interrupt).

⑨The main control side writes sending data to the IICAn register to release the main control side waiting.

⑩If the slave reads the received data and relieves the wait (WRELn=1), the master starts to transmit data to the slave.

Note If the address sent is different from the slave address, the slave does not return ACK (NACK:SDAAn=1) to the master and does not generate INTIICAn interrupts (address matching interrupts) or enter a wait state.

However, the master generates an INTIICAn interrupt for both the ACK and the NACK (the end of address sending interrupt).

Note 1. ①~⑮ in Figure 19-31 are a series of operation steps for data communication through I2C bus.

"(1) Start condition - address - data" in Figure 19-31describes steps ① to ⑥.

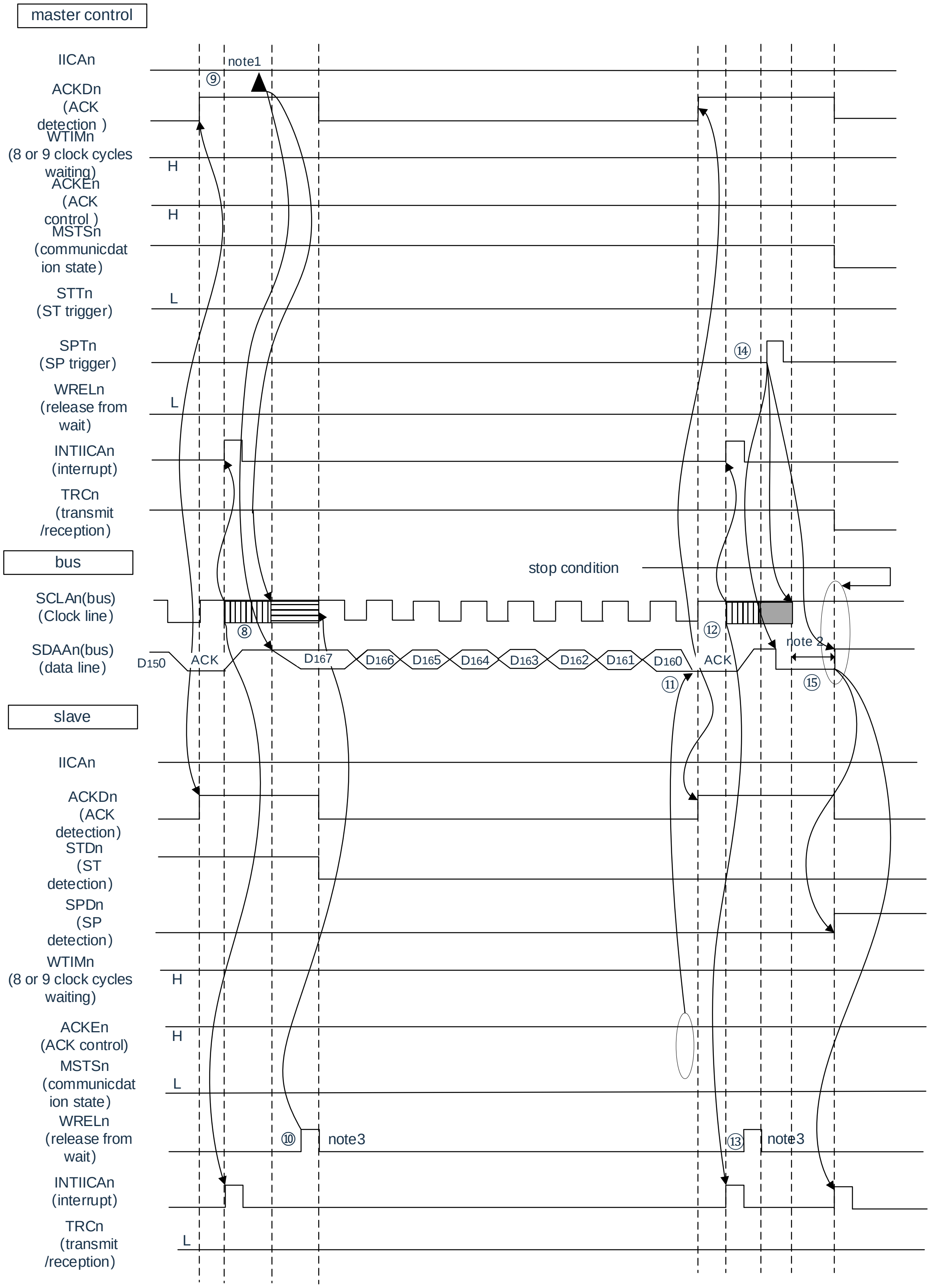
"(2) Address to data to data" in Figure 19-31 describes steps ③ to ⑩.

"(3) Data to data to stop conditions" in Figure 19-31 describes steps ⑦ to ⑮.

2.n=0,1

Figure 19-31 Communication example of master → slave
(Master: Select 9 clock wait, slave: Select 9 clock wait) (3/4)

(2) Data~Data~Stop Condition



▨ : master device waits

☰ : slave device waits

▥ : master device and slave device wait

Note 1. To release the master from waiting during transmission, you must write data to the IICAn instead of setting the WRELn bit.

2. The time from the SCLAn pin signal to generating the stop condition after issuing 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 release the slave from waiting during reception, the IICAn must be set to "FFH" or the WRELn bit must be set.

Figure 19-31 ⑦-⑮ of "(3) Data-Data-Stopping Condition" are described as follows:

⑦After the data transfer is finished, the slave's ACKEn bit is "1", so the master sends ACK through hardware. The master detects ACK (ACKDn=1) at the rising edge of the 9th clock.
⑧The master and the slave enter a waiting state (SCLAn=0) at the descending edge of the 9th clock, and both generate interrupts (INTIICAn: End of delivery interrupt).
⑨The master sends data to the IICA shift register n(IICAn) to release the master from waiting.
⑩If the slave reads the received data and the wait is WRELn=1, the master starts transferring data to the slave.
⑪After the data transfer is completed, the slave (ACKEn=1) sends an ACK to the master via the hardware. The master detects ACK (ACKDn=1) at the rising edge of the 9th clock.
⑫The master and the slave enter a waiting state (SCLAn=0,1) at the descending edge of the 9th clock, and both generate interrupts (INTIICAn: End of delivery interrupt).
⑬The slave reads the received data and unwaits (WRELn=1).
⑭If the master controller sets the stop condition trigger (SPTn=1), clear the bus data line (SDAAn=0,1) and set the bus clock line (SCLAN=1), set the bus data line (SDAAn=1) after the preparation time of the stop condition, and generate the stop condition (change SDAAn from "0" to "1" through SCLAN=1).
⑮If a stop condition is generated, the slave party detects the stop condition and generates an interrupt (INTIICAn: Stop condition interrupt).

Note 1. ①~⑮ in Figure 19-31 are a series of operation steps for data communication through I2C bus.

"(1) Start condition - address - data" in Figure 19-31describes steps ① to ⑥.

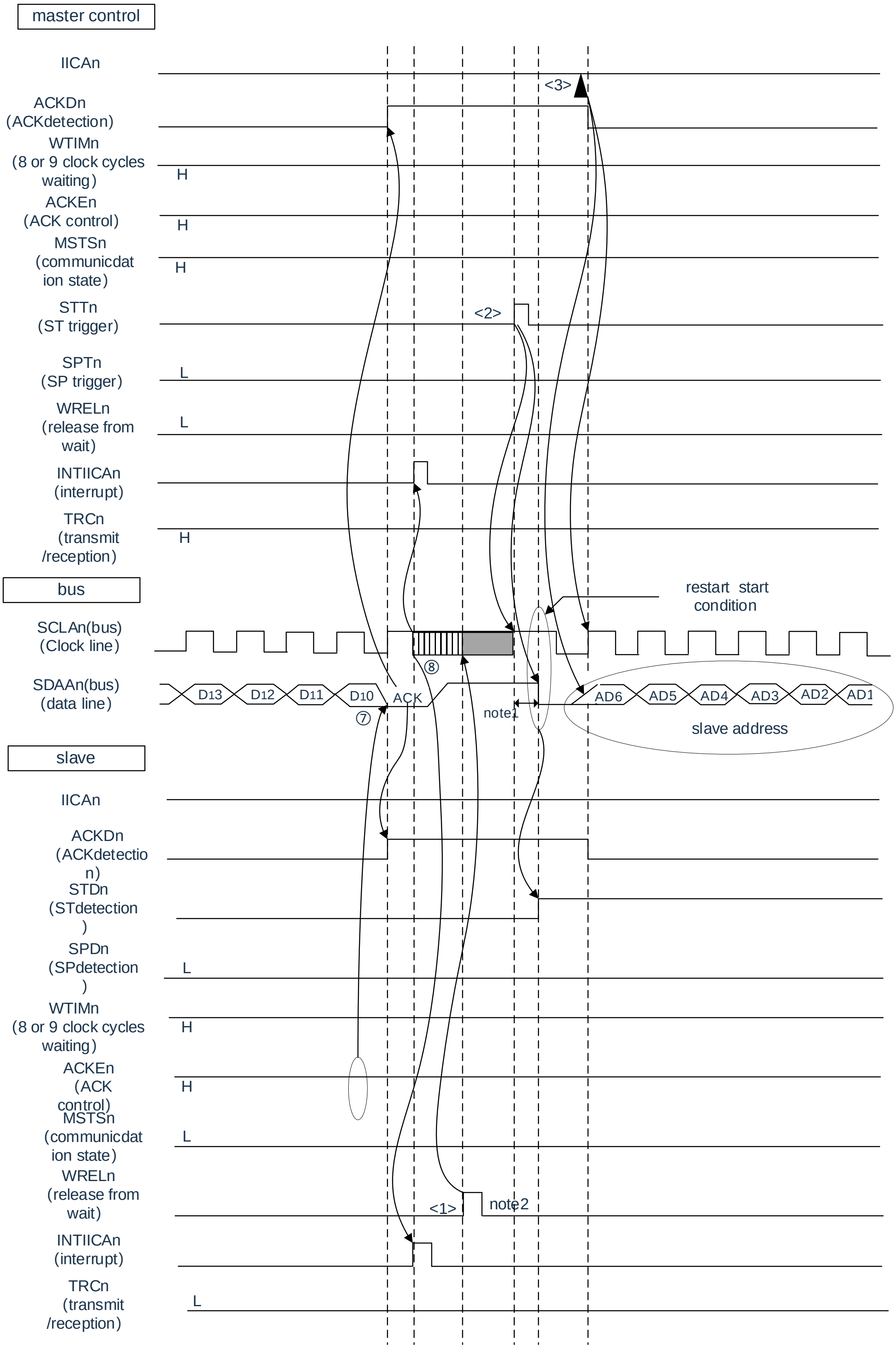
"(2) Address to data to data" in Figure 19-31 describes steps ③ to ⑩.

"(3) Data to data to stop conditions" in Figure 19-31 describes steps ⑦ to ⑮.

2.n=0,1

Figure 19-31 Communication example of master → slave
(Master: Select 9 clock wait, slave: Select 9 clock wait) (4/4)

(4) Data ~ Restart Condition ~ Address



▨ : master device waits

☰ : slave device waits

▥ : master device and slave device waits

Note 1. The time from the SCLAn pin signal to the generation start condition after the release restart condition is at least 4.7μs in standard mode and at least 0.6μs in fast mode.

2. To release the slave from waiting during reception, the IICAn must be set to "FFH" or the WRELn bit must be set.

Figure 19-31 The operation instructions for "(4) Data~Restart Condition~Address" are as follows. After steps ⑦ and ⑧ are executed, <1>~<3> is executed, thereby returning to data sending step ③.

⑦ After the data transmission is finished, the slave side transmits ACK through hardware to the master side because the ACKEn bit is 1. The master detects ACK (ACKDn=1) at the rising edge of the 9th clock.

⑧ The master and slave enter a waiting state (SCLAn=0,1) at the descending edge of the ninth clock, and both generate interrupts (INTIICAn: End of delivery interrupt).

<1> The client reads the received data and unwaits (WRELn=1).

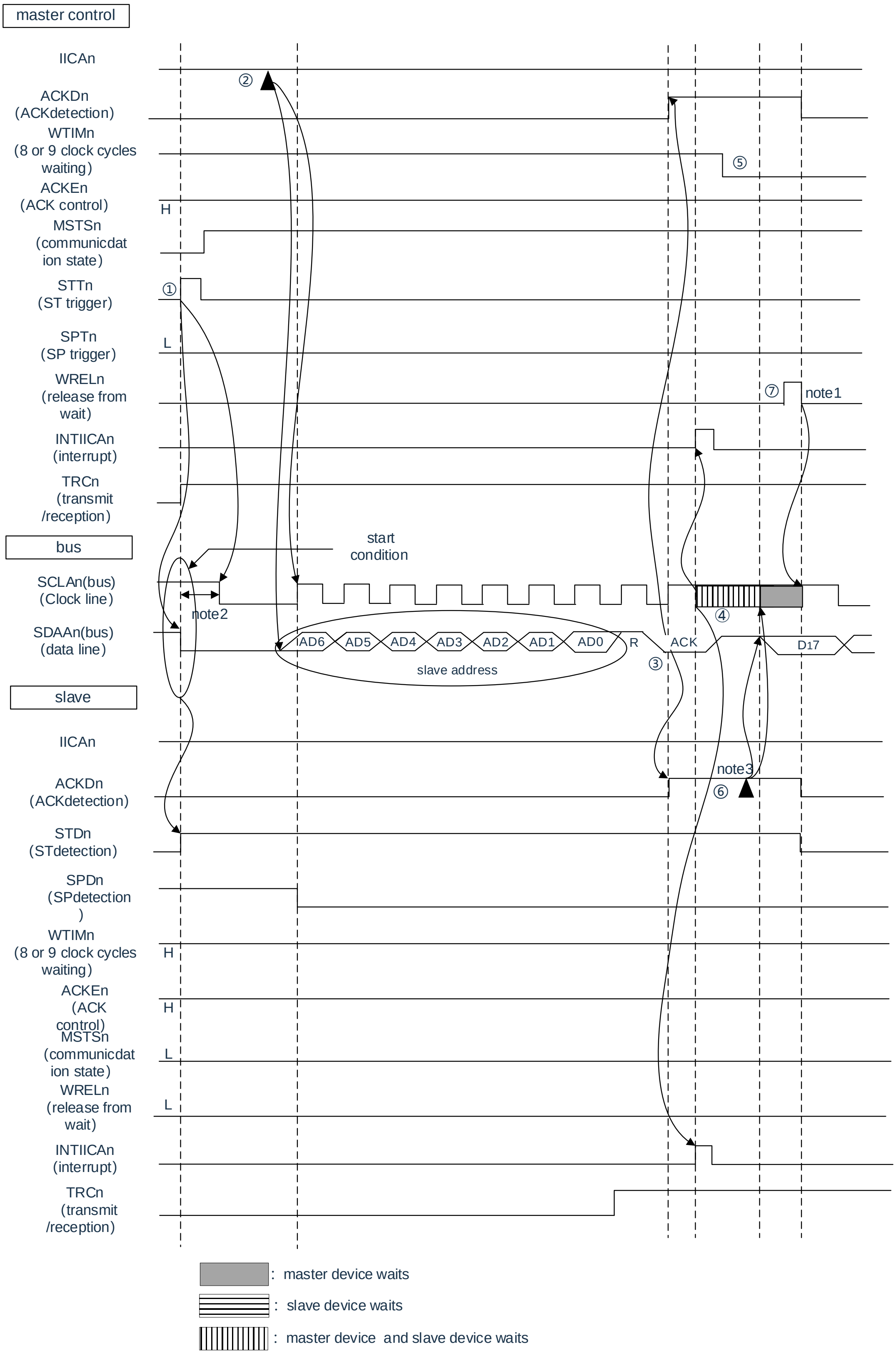
<2> If the master triggers the start condition (STTn=1) again, the bus clock line rises (SCLAn=1) and the bus data line falls (SDAAn=0,1) after the preparation time of the restart condition to generate the start condition (0). Then, if a start condition is detected, the bus clock line descends (SCLAn=0,1) after a hold time, and communication preparations are ended.

<3> If the master sends the IICA shift register n(IICAn) a write address +R/W, the slave address is sent.

Remark n=0,1

Figure 19-32 Communication example of slave ➞ master
(Master: select 8 clock waiting, slave: select 9 clock waiting) (1/3)

(1) Start Condition ~ Address ~ Data



Note 1. To release the master from waiting during reception, the IICAn must be set to "FFH" or the WRELn bit must be set.

2. The time to decrease the SDAAn pin signal to the SCLAn pin signal is at least 4.0μs when set to standard mode and at least 0.6μs when set to fast mode.

3. To release the slave from waiting during transmission, you must write data to the IICAn instead of setting the WRELn bit.

Figure 19-32 Descriptions of ① to ⑦ of "1th Condition ~ Address ~ Data" are as follows:

①If the master triggers the setting (STTn=1), SDAAn descends to generate the starting condition (SCLAn=1 changes SDAAn from "1"). After that, if the start condition is detected, the master enters the master communication state (MSTSn=1), and the bus clock line descends (SCLAn=0,1) after holding time.

②If the master sends the IICA shift register n(IICAn) a write address + R (receive), the slave address is sent.

③In the secondary party, if the receiving address and the local station address (value of SVAn) are the same note, the ACK is sent to the master via hardware. The master detects ACK (ACKDn=1) at the rising edge of the 9th clock.

④The master generates an interrupt at the descending edge of the 9th clock (INTIICAn: End of address delivery interrupt). A slave device of the same address enters a waiting state (SCLAn=0,1) and an interrupt is generated (INTIICAn: Address Matching Interrupt) Note.

⑤The main controller changed the waiting time to the eighth clock (WTIMn=0,1).

⑥The slave party writes sending data to the IICAn register, and the slave party is relieved from waiting.

⑦The main control unit removes the waiting (WRELn=1) and starts the data transmission from the slave equipment.

Note If the address sent is different from the slave address, the slave does not return ACK (NACK:SDAAn=1) to the master and does not generate INTIICAn interrupts (address matching interrupts) or enter a wait state.

However, the master generates an INTIICAn interrupt for both the ACK and the NACK (the end of address sending interrupt).

Note 1. Figure 19-32 shows a series of operation steps for data communication through I2C bus.

"(1) Start condition - address - data" in Figure 19-32 describes steps ① - ⑦.

"(2) Address to data to data" in Figure 19-32 describes steps ③ to ⑫.

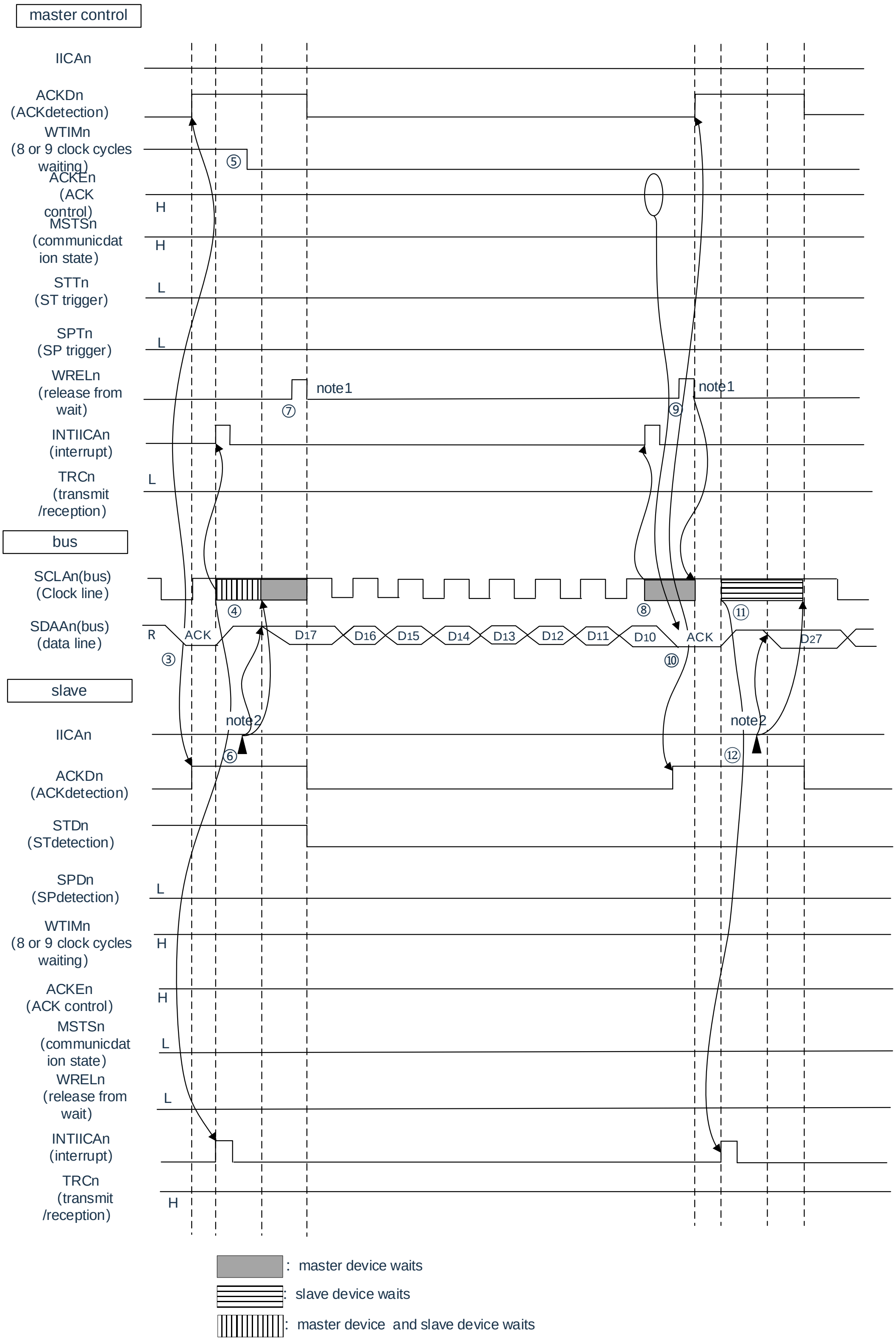
"(3) Data to data to stop conditions" in Figure 19-32 describes steps ⑧ to ⑲.

2.n=0,1

Figure 19-32 Communication example of slave → master

(Master: select 8 clock waiting, slave: select 9 clock waiting) (2/3)

(2) Address~Data~Data



Note 1. To release the master from waiting during reception, the IICAn must be set to "FFH" or the WRELn bit must be set.

2. To release the slave from waiting during transmission, you must write data to the IICAn instead of setting the WRELn bit.

Figure 19-32 ③ to ⑫ of "(2) Address-Data"   are described as follows:

③In the secondary party, if the receiving address and the local station address (value of SVAn) are the same note, the ACK is sent to the master via hardware. The master detects ACK (ACKDn=1) at the rising edge of the 9th clock.

④The master generates an interrupt at the descending edge of the 9th clock (INTIICAn: End of address delivery interrupt). A slave device of the same address enters a waiting state (SCLAn=0,1) and an interrupt is generated (INTIICAn: Address Matching Interrupt) Note.

⑤The main controller changed the waiting time to the eighth clock (WTIMn=0,1).

⑥The slave party writes the transmission data to the IICA shift register n(IICAn), and relieves the slave party.

⑦The main control unit removes the waiting (WRELn=1) and starts the data transmission from the slave equipment.

⑧The master enters a waiting state (SCLAn=0,1) at the descending edge of the eighth clock and generates an interrupt (INTIICAn: End of delivery interrupt). Because the ACKEn bit of the master is "1", the slave is sent an ACK through the hardware.

⑨The main control side reads the received data and relieves the waiting (WRELn=1).

⑩The slave detects an ACK (ACKDn=1) at the rising edge of the 9th clock.

⑪The slave enters a waiting state (SCLAn=0,1) at the descent edge of the 9th clock and generates an interrupt (INTIICAn: End of delivery interrupt).

⑫If the slave writes sending data to the IICAn register, the slave is relieved from waiting and the data transfer from the slave to the master is started.

Note: If the address sent and the slave address are different, the slave party does not return ACK(NACK:SDAAn=1) to the host party and does not generate INTIICAn interrupts (address matching interrupts) and does not enter a wait state.

However, the master generates an INTIICAn interrupt for both the ACK and the NACK (the end of address sending interrupt).

Note 1. Figure 19-32 shows a series of operation steps for data communication through I2C bus.

"(1) Start condition - address - data" in Figure 19-32 describes steps ① - ⑦.

"(2) Address to data to data" in Figure 19-32 describes steps ③ to ⑫.

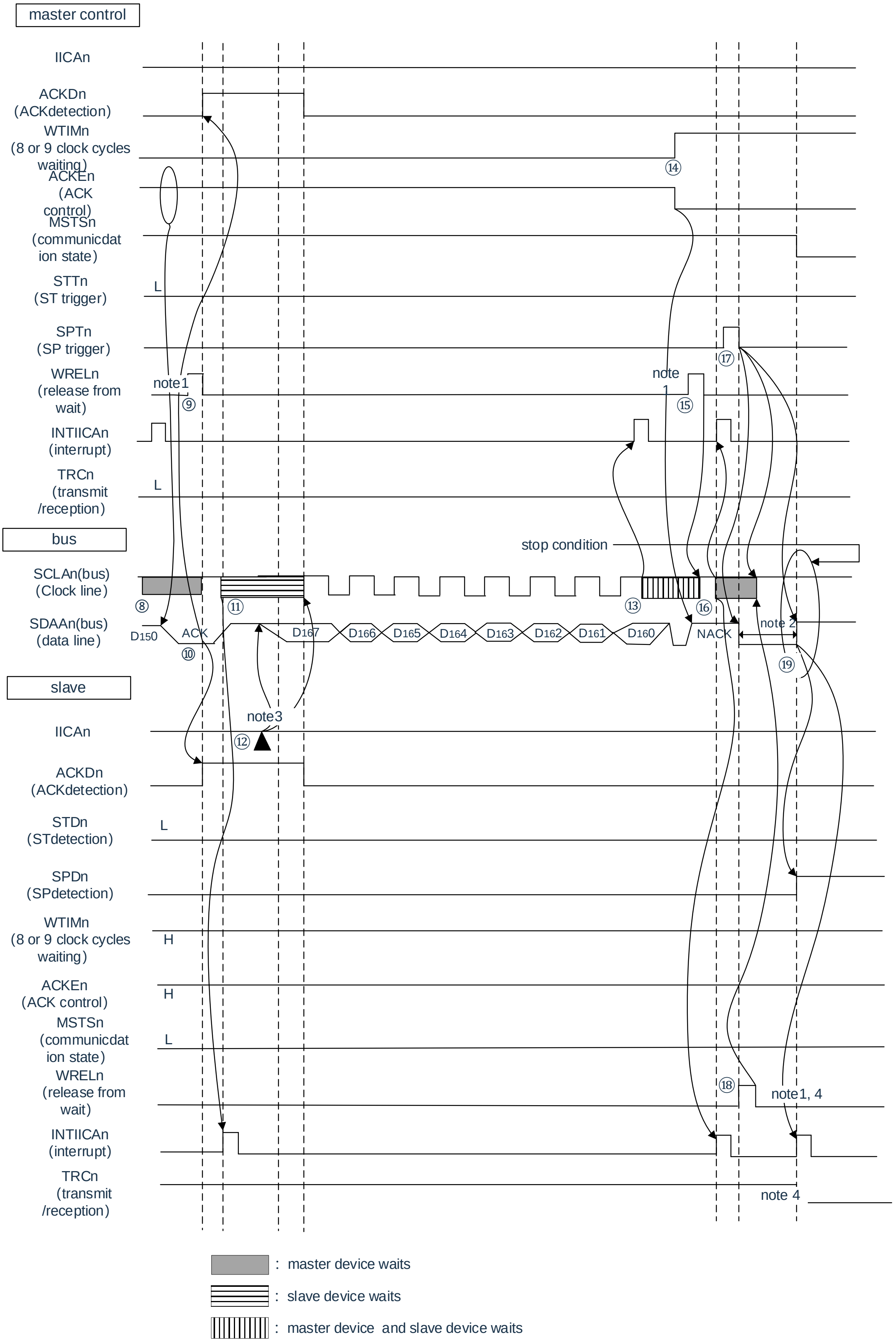
"(3) Data to data to stop conditions" in Figure 19-32 describes steps ⑧ to ⑲.

2.n=0,1

Figure 20-32 Communication example of slave → master

(Master: select 8 clock waiting, slave: select 9 clock waiting) (3/3)

(3) Data~Data~Stop Condition



Note 1. To unwait, you must either place the IICAn in "FFH" or set the WRELn bit.
2. The time from the SCLAn pin signal to generating the stop condition after issuing 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 release the slave from waiting during transmission, you must write data to the IICAn instead of setting the WRELn bit.
4. During the slave's transmit, the TRCn bit is cleared if the wait is released by the setting of the WRELn bit.

Figure 19-32 Descriptions of ⑧ to ⑲ for (3) Data~Data~Stopping Conditions are as follows:

⑧. The master enters a waiting state (SCLAn=0,1) at the descending edge of the eighth clock and generates an interrupt (INTIICAn: End of delivery interrupt). Because the ACKEn bit of the master is "0", the slave is sent an ACK through the hardware.

⑨. The master reads the received data and unwaits (WRELn=1).

⑩. The slave detects ACK (ACKDn=1) at the rising edge of the 9th clock.

⑪. The slave enters a waiting state (SCLAn=0,1) along the descending edge of the 9th clock, and generates an interrupt (INTIICAn: End of delivery interrupt).

⑫. If the slave writes sending data to the IICA shift register n (IICAn), the slave waits and starts data transmission from the slave to the master.

⑬. The master generates an interrupt on the descending edge of the 8th clock (INTIICAn: End of Transfer interrupt) and enter a wait state (SCLAn=0,1). Because of the ACK control (ACKEn=1), the bus data line at this stage becomes low level (SDAAn=0,1).

⑭. The master is set to NACK Acknowledgement (ACKEn=0,1) and changes the wait time to the ninth clock (WTIMn=1). If the master unwaits (WRELn=1), the slave detects NACK (ACKDn=0,1) on the rising edge of the 9th clock.

⑮. The master and the slave enter a waiting state (SCLAn=0,1) at the descending edge of the 9th clock, and both generate interrupts (INTIICAn: End of delivery interrupt).

⑯. If the master issues a stop condition (SPTn=1), the bus data line (SDAAn=0,1) is cleared and the master waits. After that, the master is in standby until the bus clock line is SCLAn=1.

⑰. The slave stops sending after confirming the NACK, and in order to end the communication, the wait is canceled (WRELn=1). If the slave party is relieved of waiting, the bus clock line is set (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

⑲. (SDAAn=1), and then issue the stop condition (SDAAn changed from "0" to "1" via SCLAn=1). If a stop condition is generated, the slave party detects the stop condition and generates an interrupt (INTIICAn: Stop condition interrupt).

Chapter 20 Enhanced DMA

20.1 Function of DMA

The DMA is a function of transferring data between memories without using a CPU. Starting DMA for data transfer is interrupted by peripheral functions. When DMA and CPU access the same unit in FLASH, SRAM0, SRAM1 or peripheral module simultaneously, their bus use rights are higher. When DMA and CPU access different units in FLASH, SRAM0, SRAM1 or peripheral module respectively, they do not interfere with each other.

The specifications of the DMA are shown in Table 20-1.

Table 20-1 Specifications for DMA (1/2)

Item		Specifications
Boot source		Up to 37 boot sources
Distributable control data		Group 40
Transmissionable address space	Address space	full address range space
	Source	Full address range space optional
	Objectives	Full address range space optional
Maximum Transfers	Normal mode	65535 times
	Repeat mode	65535 times
Maximum transfer block size	Normal mode (8-bit transfer)	65535 bytes
	Normal mode (16-bit Transfer)	131070 bytes
	Normal mode (32-bit Transfer)	262140 bytes
	Repeat mode	65535 bytes
Transfer unit		8-/16-32-bit
Transfer mode	Normal mode	Ends after the transfer of the DMACTj register from "1" to "0".
	Repeat mode	After the transfer of the DMACTj register from "1" to "0", the address of the repeat region is initialized and the DMRLDj The value of the register reloads into the DMACTj register and continues to be transferred.
Address control	Normal mode	Fixed or incremental
	Repeat mode	Fix or increment the address of the non-repeating area.
Priority of the start-up source		Refer to “20-5 DMA boot source and vector addresses”

Table 20-1 Specifications for DMA (2/2)

Item		Specifications
Interrupt request	Normal mode	When the DMACTj register is transferred from "1" to "0", the CPU is requested to start the source interrupt and interrupts.
	Repeat mode	When the DMACRj register has RPTINT bit '1' allowing interrupt generation) and the DMACTj register changes.
Transfer Start		If the DMAENi0 to DMAENi7 bits of the DMAENi register are set to "1" (startup is allowed), data transmission starts each time a DMA startup source occurs.
Transfer stop	Normal mode	· Set DMAENi0~DMAENi7 to "0" (Disable boot). · When the DMACTj register changes from "1" to "0" data transfer ends
	Repeat mode	· Set DMAENi0~DMAENi7 to "0" (Disable boot). · End of data transfer when the RPTINT bit is "1" (interrupts allowed) and the DMACTj register changes from "1"

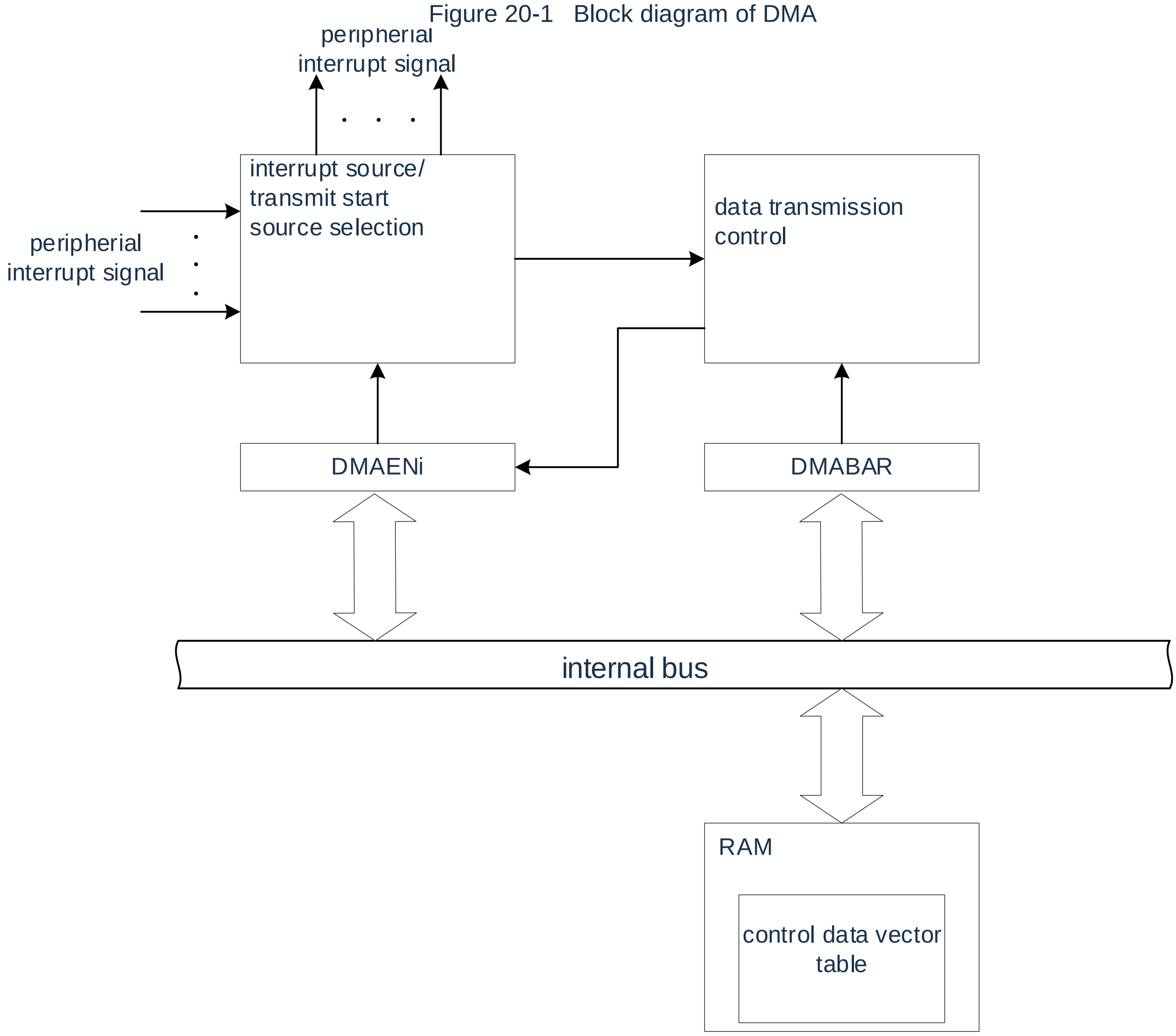
Note In deep sleep mode, flash memory cannot be used as a source for DMA transfer because it stops working.

Remark i=0~4, j=0~39

20.2 Structure of DMA

The block diagram for DMA is shown in Figure 20-1

Figure 20-1 Block diagram of DMA



20.3 Registers for controlling DMA

The registers that control the DMA are shown in Table 20-2.

Table 20-2 Registers for controlling DMA

Register name	Symbol
Peripheral Enable Register 1	PER1
DMA Boot Enable Register 0	DMAEN0
DMA Boot Enable Register 1	DMAEN1
DMA Boot Enable Register 2	DMAEN2
DMA Boot Enable Register 3	DMAEN3
DMA Boot Enable Register 4	DMAEN4
DMA base address register	DMABAR

The control data of the DMA is shown in Table 20-3.

The control data of the DMA is allocated in the DMA control data area of the RAM. The DMA control data area and the 704-byte area containing the DMA vector table area (the starting address of the stored control data) are set through the DMABAR register.

Table 20-3 Control data for DMA

Register name	Symbol
DMA control register j	DMACRj
DMA block size register j	DMBLSj
DMA transfer number register j	DMACTj
DMA transfer times reload register j	DMRLDj
DMA source address register j	DMSARj
DMA destination address register j	DMDARj

Remark j=0~39

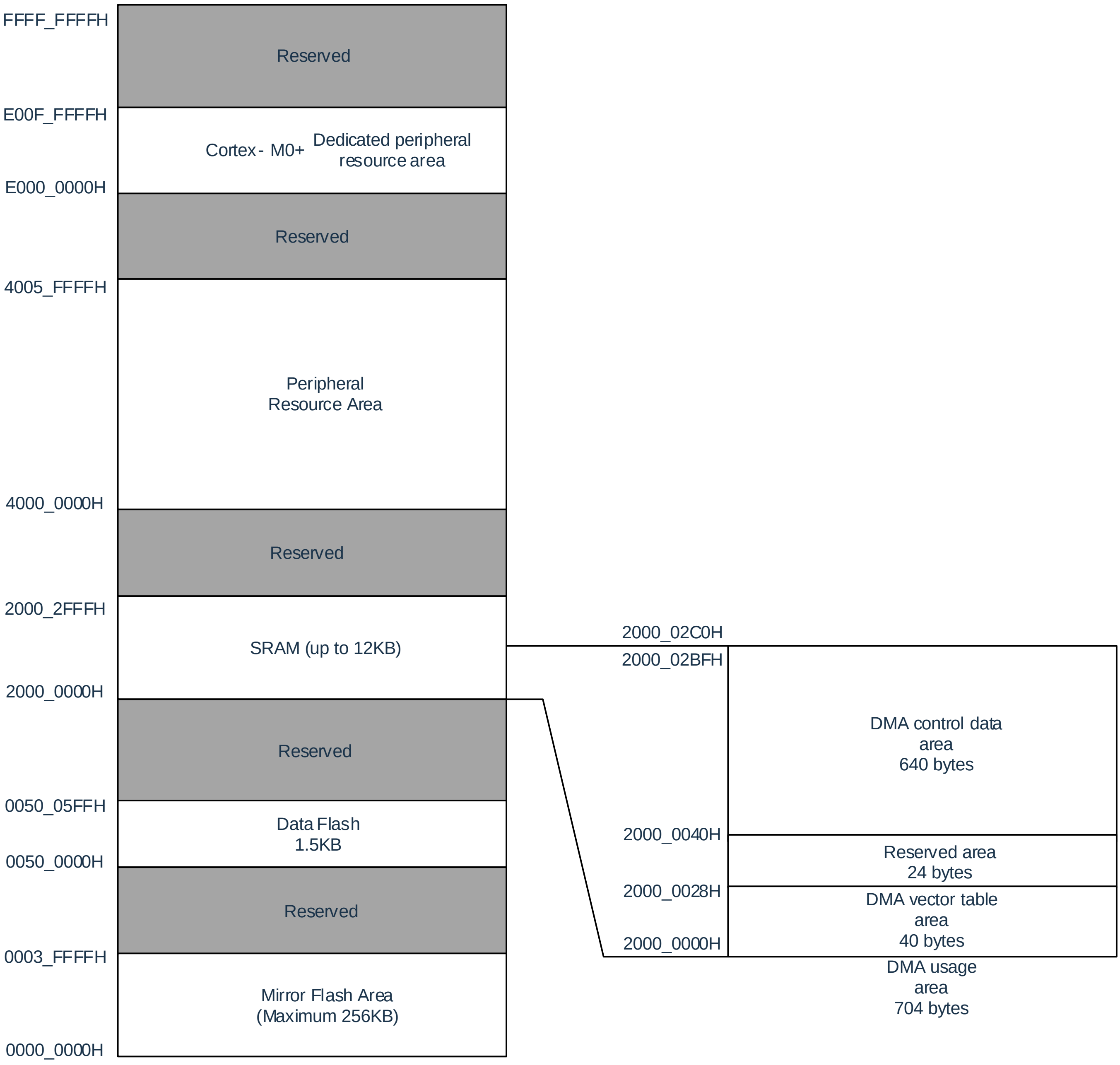
20.3.1 Allocation of DMA control data area and DMA vector table area

A 704-byte region of the control data and the vector table assigned to the DMA is set to the RAM region through the DMABAR register.

An example of a memory image with the DMABAR register set to "20000000H" is shown in Figure 20-2.

Space not used by DMA in 640 bytes of the DMA control data area can be used as RAM.

Figure 20-2 Example of memory image when the DMABAR register is set to "20000000H



20.3.2 Controlling data allocation

Starting from the starting address, control data is allocated in the order of DMACRj, DMBLSj, DMACTj, DMRLDj, DMSARj, DMDARj registers.

The starting address is set by the DMABAR register, and the lower 10 bits are set by the vector tables assigned by each starting source.

The distribution of control data is shown in Figure 20-3.

Note 1. You must change the data for the DMACRj,DMBLSj,DMACTj, DMRLDj, DMSARj, DMDARj registers when the DMAENi0~DMAENi7 bit of the corresponding DMAENi (i=0~4) is "0".

2. Access to DMACRj, DMBLSj, DMACTj, DMRLDj, DMSARj, and DMDARj cannot be performed via DMA transfer.

Figure 20-3 Control data allocation (DMABAR is set to 2000000H)

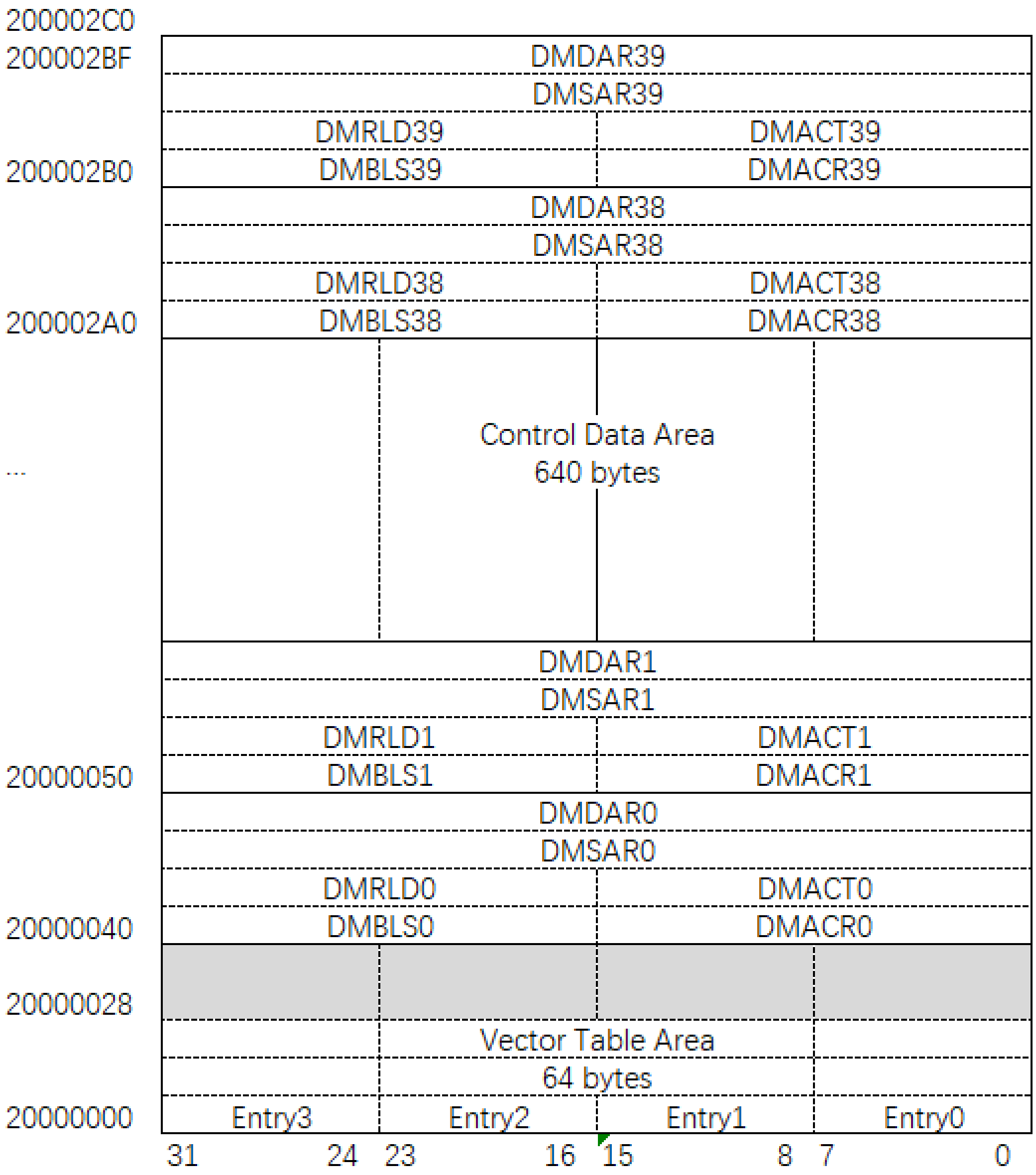


Table 20-4 Starting address of control data

j	Address
19	baseaddr+170H
18	baseaddr+160H
17	baseaddr+150H
16	baseaddr+140H
15	baseaddr+130H
14	baseaddr+120H
13	baseaddr+110H
12	baseaddr+100H
11	baseaddr+F0H
10	baseaddr+E0H
9	baseaddr+D0H
8	baseaddr+C0H
7	baseaddr+B0H
6	baseaddr+A0H
5	baseaddr+90H
4	baseaddr+80H
3	baseaddr+70H
2	baseaddr+60H
1	baseaddr+50H
0	baseaddr+40H

j	Address
39	baseaddr+2B0H
38	baseaddr+2A0H
37	baseaddr+290H
36	baseaddr+280H
35	baseaddr+270H
34	baseaddr+260H
33	baseaddr+250H
32	baseaddr+240H
31	baseaddr+230H
30	baseaddr+220H
29	baseaddr+210H
28	baseaddr+200H
27	baseaddr+1F0H
26	baseaddr+1E0H
25	baseaddr+1D0H
24	baseaddr+1C0H
23	baseaddr+1B0H
22	baseaddr+1A0H
21	baseaddr+190H
20	baseaddr+180H

Remark: baseaddr: Setting values of the DMABAR register

20.3.3 Vector table

Once the DMA is started, control data assigned to the DMA control data area is read by data read from a vector table assigned to each start source.

The DMA start source and vector address are shown in Table 20-5. The vector table of each start source has 1 byte to save the data from "00H" to "27H", and 1 group of data is selected from the 40 groups of control data. The high 22 bits of the vector address are set by the DMABAR register, and the low 10 bits are assigned to "00H" to "2H" of the corresponding start source.

Note The starting address of the DMA control data area set in the vector table must be changed if the DMAENi0~DMAENi7 bit of the corresponding DMAENi (i=0~4) register is '0'.

Figure 20-4 Starting address and vector table for control data

When the DMABAR register is set to "2000000H"

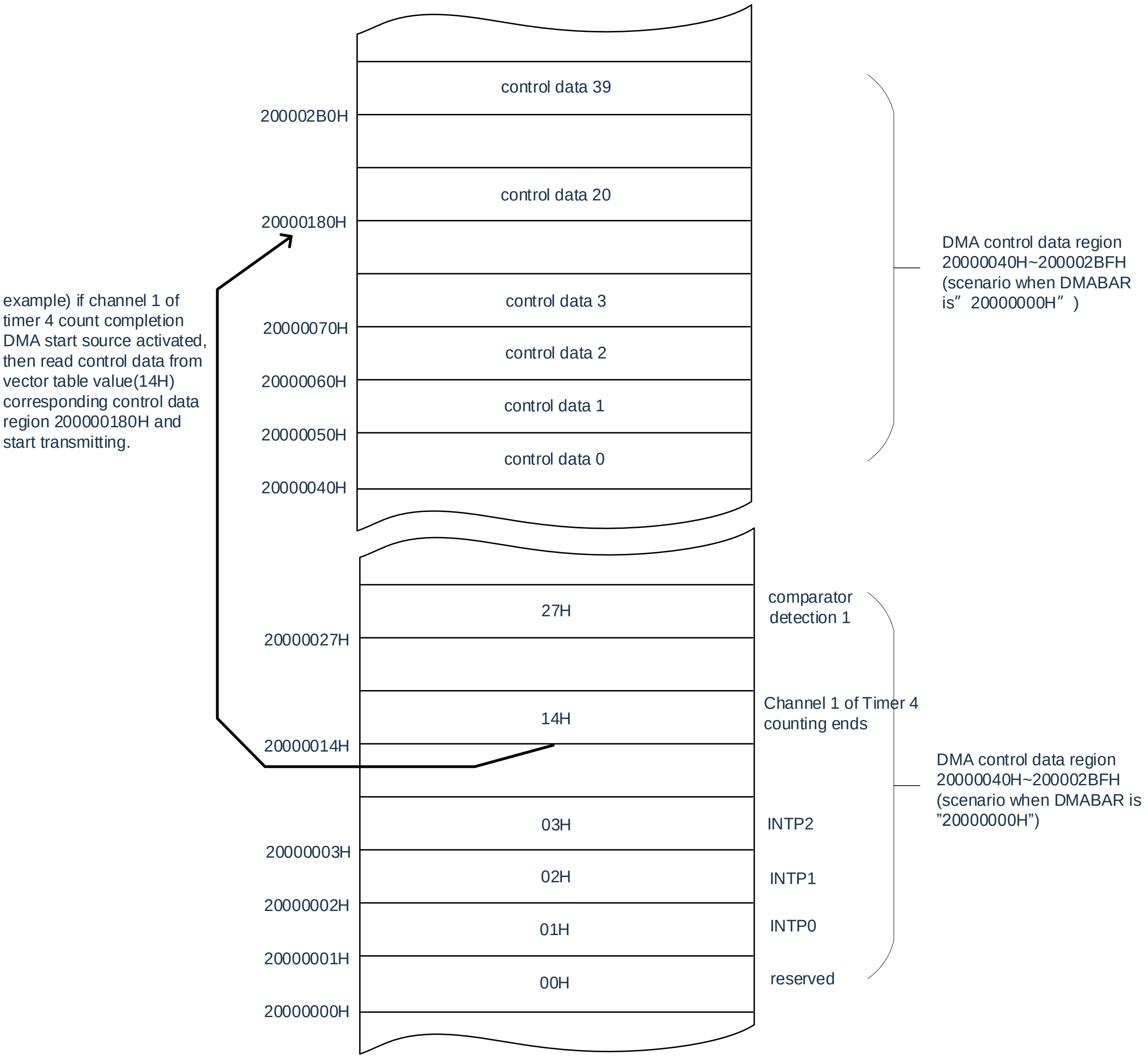


Table 20-5 DMA boot source and vector addresses

DMA boot source (interrupt request generation source)	Source No.	Vector address	Priority
End of Flash Read/Write Erase	0	DMABAR REGISTER SETTING ADDRESS+00H	High
INTP0	1	DMABAR REGISTER SETTING ADDRESS+01H	
INTP1	2	DMABAR REGISTER SETTING ADDRESS+02H	
INTP2	3	DMABAR REGISTER SETTING ADDRESS+03H	
INTP3	4	DMABAR REGISTER SETTING ADDRESS+04H	
INTP4	5	DMABAR REGISTER SETTING ADDRESS+05H	
INTP5	6	DMABAR REGISTER SETTING ADDRESS+06H	
INTP6	7	DMABAR REGISTER SETTING ADDRESS+07H	
INTP7	8	DMABAR REGISTER SETTING ADDRESS+08H	
Key Input	9	DMABAR REGISTER SETTING ADDRESS+09H	
End of A/D conversion	10	DMABAR REGISTER SETTING ADDRESS+0AH	
End of transmission sent by UART0/end of transmission by SSPI00 or buffer null/end of transmission of IIC00	11	DMABAR REGISTER SETTING ADDRESS+0BH	
End of transmission received by UART0/end of transmission by SSPI01 or buffer null/end of transmission of IIC01	12	DMABAR REGISTER SETTING ADDRESS+0CH	
End of transmission sent by UART1/end of transmission by SSPI10 or buffer null/end of transmission for IIC10	13	DMABAR REGISTER SETTING ADDRESS+0DH	
End of transmission received by UART1/end of transmission by SSPI11 or buffer null/end of transmission of IIC11	14	DMABAR REGISTER SETTING ADDRESS+0EH	
End of transmission sent by UART2/end of transmission by SSPI20 or buffer null/end of transmission of IIC20	15	DMABAR REGISTER SETTING ADDRESS+0FH	
End of transmission received by UART2/end of transmission by SSPI21 or buffer null/end of transmission of IIC21	16	DMABAR REGISTER SETTING ADDRESS+10H	
IICA0 communication ends.	17	DMABAR REGISTER SETTING ADDRESS+11H	
End of IICA1 communication.	18	DMABAR REGISTER SETTING ADDRESS+12H	
End of SPIHS0 transmission	19	DMABAR REGISTER SETTING ADDRESS+13H	
End of SPIHS1 transmission	20	DMABAR REGISTER SETTING ADDRESS+14H	
End of count or capture of channel 0 of Timer4	21	DMABAR REGISTER SETTING ADDRESS+15H	
End of count or capture of channel 1 of Timer4	22	DMABAR REGISTER SETTING ADDRESS+16H	
End of count or capture of channel 2 of Timer4	23	DMABAR REGISTER SETTING ADDRESS+17H	
End of count or capture of channel 3 of Timer4	24	DMABAR REGISTER SETTING ADDRESS+18H	
End of count or capture of channel 0 of Timer8	26	DMABAR REGISTER SETTING ADDRESS+1AH	Low
End of count or capture of channel 1 of Timer8	27	DMABAR REGISTER SETTING ADDRESS+1BH	

End of count or capture of channel 2 of Timer8	28	DMABAR REGISTER SETTING ADDRESS+1CH	
End of count or capture of channel 3 of Timer8	29	DMABAR REGISTER SETTING ADDRESS+1DH	
End of count or capture for channel 4 of Timer8	30	DMABAR REGISTER SETTING ADDRESS+1EH	
dma tx request for ssi	31	DMABAR REGISTER SETTING ADDRESS+1FH	
dma rx request for ssi	32	DMABAR REGISTER SETTING ADDRESS+20H	
dma rt request for ssi	33	DMABAR REGISTER SETTING ADDRESS+21H	
15-bit interval timer generates counting interrupts	34	DMABAR REGISTER SETTING ADDRESS+22H	
USB D0FIFO transfer request	35	DMABAR REGISTER SETTING ADDRESS+23H	
USB D1FIFO transfer request	36	DMABAR REGISTER SETTING ADDRESS+24H	
Comparator detection 0	37	DMABAR REGISTER SETTING ADDRESS+25H	
Comparator detection1	38	DMABAR REGISTER SETTING ADDRESS+26H	
DMA transfer request for qspi	39	DMABAR REGISTER SETTING ADDRESS+27H	

20.3.4 Peripheral enable register 1 (PER1)

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

You must set bit3 (DMAEN) to "1" when using DMA.

The PER1 register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "00H".

Figure 20-5 Format of peripheral enable register 1(PER1)

Address: 0x4002081A After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
PER1	DACEN	TMBEN	PGACMPEN	TMMEN	DMAEN	PWMPEN	TMCEN	TMAEN

DMAEN	Providing control of an input clock of a DMA
0	Stop provide an input clock. · DMA cannot run.
1	Provides an input clock. · DMA can run.

20.3.5 DMA control register j(DMACRj) (j=0~39)

The DMACRj register controls the mode of operation of the DMA.

Figure 20-6 Format of DMA control register j(DMACRj)

Address Refer to 20.3.2 Controlling Data Allocation. After reset: indefinite value R/W

Symbol:	15	14	13	12	11	10	9	8
DMACRj	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	SZ		RPTINT	CHEN	DAMOD	SUMMER	RPTSEL	MODE

FIFO	FIFO block transfer control
0	Not FIFO block transfer
1	It is a FIFO block transfer where the source address (SAMOD=0) or destination address (DAMOD=0) is absolutely fixed

SZ	Selection of transmission data length
00	8-bit
01	16-bit
10	32-bit
11	Disable settings

RPTINT	Enable/disable for repeat mode interrupt
0	Interrupts are prohibited.
1	Interrupts are enabled.
The setting for the RPTINT bit is not valid when the MODE bit is '0' (normal mode).	

CHNE	Chain transmission enable/disable
0	Disable chain transmission.
1	Enable chain transmission.
The CHNE bit of the DMACR 39 register must be "0" (Chain transfer is disabled).	

DAMOD	Control of transfer destination address
0	Fixed
1	Incremental
The DAMOD bit is not set when the MODE bit is "1" and the RPTSEL bit is "0".	

SAMOD	Control of transfer source address
0	Fixed
1	Incremental
The SAMOD bit is not set when the MODE bit is "1" (repeat mode) and RPTSEL bit is "1".	

RPTSEL	Selection of repeat area
0	The transfer target is a repeat area.
1	The transfer source is a repeat area.
The setting for the RPTSEL bit is not valid when the MODE bit is '0' (normal mode).	

MODE	Selection of transfer modes
0	Normal mode
1	Repeat mode

Note Access to the DMACRj register is not possible through DMA transfers.

20.3.6 DMA block size register j(DMBLSj) (j=0~39)

This register sets the block size of the transfer data started 1 time.

Figure 20-7 Format of DMA block size register j(DMBLSj)

Address Refer to 20.3.2 Controlling Data Allocation. After reset: indefinite value R/W

Symbol:	15	14	13	12	11	10	9	8
DMBLSj	DMBLSj15	DMBLSj14	DMBLSj13	DMBLSj12	DMBLSj11	DMBLSj10	DMBLSj9	DMBLSj8
	7	6	5	4	3	2	1	0
	DMBLSj7	DMBLSj6	DMBLSj5	DMBLSj4	DMBLSj3	DMBLSj2	DMBLSj1	DMBLSj0

DMBLSj	Transmission Block Size		
	8-bit transmission	16-bit transmission	32-bit transmission
00H	Disable from setting	Disable from setting	Disable from setting
01H	1 byte	2 bytes	4 bytes
02H	2 bytes	4 bytes	8 bytes
03H	3 bytes	6 bytes	12 bytes
. . .	. . .	. . .	. . .
FDH	253 bytes	506 bytes	1012 bytes
FEH	254 bytes	508 bytes	1016 bytes
FFH	255 bytes	510 bytes	1020 bytes
. . .	. . .	. . .	. . .
FFFFFH	65535 bytes	131070 bytes	262140 bytes

Note 1. Access to the DMBLSj register cannot be performed via DMA transfer.

20.3.7 DMA transfer times register j(DMACTj) (j=0~39)

This register sets the number of data transfers for DMA. Each time you initiate a DMA transfer, you reduce 1.

Figure 20-8 Format of 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 20.3.2 Controlling Data Allocation. After reset: indefinite value R/W

DMACTj	Number of transfers
00H	Disable from setting
01H	1 time
02H	2 times
03H	3 times
. . .	. . .
FDH	253 times
FEH	254 times
FFH	255 times
. . .	. . .
FFFFFH	65535 times

Note 1. Access to the DMACTj register cannot be performed via DMA transfer.

20.3.8 DMA transfer times reload register j (DMRLDj) (j=0~39)

This register sets the initial value of the transfer number register in repeat mode. In repeat mode, the value of this register must be the same as the initial value of the DMACT register because it is reloaded into the DMACT register.

Figure 20-9 Format of DMA transfer number reload register j(DMRLDj)

Address Refer to 20.3.2 Controlling Data Allocation. After reset: indefinite value R/W

Symbol:	15	14	13	12	11	10	9	8
DMRLDj	DMRLDj15	DMRLDj14	DMRLDj13	DMRLDj12	DMRLDj11	DMRLDj1 0	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 performed via DMA transfer.

20.3.9 DMA source address register j(DMSARj) (j=0~39)

This register specifies the transmission source address when the data is transferred.

When the SZ bit of the DMACRj register is '01' (16-bit transfer), the lowest bit is ignored and processed as an even address.

When the SZ bit of the DMACRj register is "10" (32-bit transfer), the low 2 bits are ignored and processed as word addresses.

Figure 20-10 Format of DMA source address register j(DMSARj)

Address Refer to 20.3.2 Controlling Data Allocation. After reset: indefinite value R/W

symbol	31	30	29	28	27	26	25	24
DMSARj	DMSARj31	DMSARj30	DMSARj29	DMSARj28	DMSARj27	DMSARj26	DMSARj25	DMSARj24
	23	22	21	20	19	18	17	16
	DMSARj23	DMSARj22	DMSARj21	DMSARj20	DMSARj19	DMSARj18	DMSARj17	DMSARj16
	15	14	13	12	11	10	9	8
	DMSARj15	DMSARj14	DMSARj13	DMSARj12	DMSARj11	DMSARj10	DMSARj9	DMSARj8
	7	6	5	4	3	2	1	0
	DMSARj7	DMSARj6	DMSARj5	DMSARj4	DMSARj3	DMSARj2	DMSARj1	DMSARj0

Note 1. Access to the DMSARj register cannot be performed via DMA transfer.

20.3.10 DMA destination address register j(DMDARj) (j=0~39)

This register specifies the destination address of the transfer when the data is transferred.

When the SZ bit of the DMACRj register is '01' (16-bit transfer), the lowest bit is ignored and processed as an even address.

When the SZ bit of the DMACRj register is "10" (32-bit transfer), the low 2 bits are ignored and processed as word addresses.

Figure 20-11 Format of DMA destination address register j(DMDARj)

Address Refer to 20.3.2 Controlling Data Allocation. After reset: indefinite value R/W

symbol	31	30	29	28	27	26	25	24
DMDARj	DMDARj31	DMDARj30	DMDARj29	DMDARj28	DMDARj27	DMDARj26	DMDARj25	DMDARj24
	23	22	21	20	19	18	17	16
	DMDARj23	DMDARj22	DMDARj21	DMDARj20	DMDARj19	DMDARj18	DMDARj17	DMDARj16
	15	14	13	12	11	10	9	8
	DMDARj15	DMDARj14	DMDARj13	DMDARj12	DMDARj11	DMDARj10	DMDARj9	DMDARj8
	7	6	5	4	3	2	1	0
	DMDARj7	DMDARj6	DMDARj5	DMDARj4	DMDARj3	DMDARj2	DMDARj1	DMDARj0

Note: Access to the DMDARj register cannot be made via DMA transfer.

20.3.11 DMA boot enable register i (DMAENi) (i=0~4)

This is an 8-bit register that controls whether or not starting DMA through each interrupt source. The correspondence of that interrupt source and the DMAENi0~DMAENi7 bit is shown in Table 20-6.

The DMAENi register can be set by an 8-bit memory operation instruction.

Note 1. You must change the DMAENi0~DMAENi7 bit at a location that does not produce a start source for the bit.
2. Access to the DMAENi register cannot be performed via DMA transfer.
3. The assigned function varies from product to product, and the bit without the assigned function must be set to "0".

Figure 20-12 Format of DMA boot enable register i(DMAENi) (i=0~4)

Addresses: 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	DMAENj7	DMAENj6	DMAENj5	DMAENj4	DMAENj3	DMAENj2	DMAENj1	DMAENj0

DMAENi7	Enable i7 for DMA boot
0	Disable boot.
1	Enable boot.
The DMAENi7 bit changes to "0" depending on the condition of the end of the transmission interrupt.	

DMAENi6	Enable i6 for DMA boot
0	Disable boot.
1	Enable boot.
The DMAENi6 bit changes to "0" depending on the condition of the end of the transmission interrupt.	

DMAENi5	Enable i5 for DMA boot
0	Disable boot.
1	Enable boot.
The DMAENi5 bit changes to "0" depending on the condition of the end of the transmission interrupt.	

DMAENi4	Enable i4 for DMA boot
0	Disable boot.
1	Enable boot.
The DMAENi4 bit changes to "0" depending on the condition of the end of the transmission interrupt.	

DMAENi3	Enable i3 for DMA boot
0	Disable boot.
1	Enable boot.
The DMAENi3 bit changes to "0" depending on the condition of the end of the transmission interrupt.	

DMAENi2	Enable i2 for DMA boot
0	Disable boot.
1	Enable boot.
The DMAENi2 bit changes to "0" depending on the condition of the end of the transmission interrupt.	

DMAENi1	Enable i1 for DMA boot
0	Disable boot.
1	Enable boot.
The DMAENi1 bit changes to "0" depending on the condition of the end of the transmission interrupt.	

DMAENi0	Enable i0 for DMA boot
0	Disable boot.
1	Enable boot.
The DMAENi0 bit changes to "0" depending on the condition of the end of the transmission interrupt.	

Table 20-6 Correspondence of interrupt source and DMAENi0 to DMAENi7 bits

Register	DMAENi7 bit	DMAENi6 bit	DMAENi5 bit	DMAENi4 bit	DMAENi3 bit	DMAENi2 bit	DMAENi1 bit	DMAENi0 bit
DMAEN0	INTP6	INTP5	INTP4	INTP3	INTP2	INTP1	INTP0	End of Flash erase/write
DMAEN1	UART 2 transmit end/ SSPI20 transfer end or buffer null/ IIC20 transfer end	UART 1 receive end/ SSPI11 transfer end or buffer null/ IIC11 transfer end	UART 1 transmit end/ SSPI10 transfer end or buffer null/ IIC10 transfer end	UART 0 receive end/ SSPI01 transfer end or buffer null/ IIC01 transfer end	UART 0 transmit end/ SSPI00 transfer end or buffer null/ IIC00 transfer end	A/D conversion end	KEY Interrupt	INTP7
DMAEN2	End of count or capture for channel 2 of general purpose timer TM4	End of count or capture for channel 2 of general purpose timer TM4	End of count or capture for channel 0 of general purpose timer TM4	End of SPIHS1 transfer	End of SPIHS0 transfer	End of IICA1 communication	End of IICA0 communication	UART 2 receive end/ SSPI21 transfer end or buffer null/ IIC21 transfer end
DMAEN3	Ssi dma rx requests	Ssi dma tx requests	End of count or capture for channel 4 of general purpose timer TM8	End of count or capture for channel 3 of general purpose timer TM8	End of count or capture for channel 2 of general purpose timer TM8	End of count or capture for channel 1 of general purpose timer TM8	End of count or capture for channel 0 of general purpose timer TM8	End of count or capture for channel 3 of general purpose timer TM4
DMAEN4	Transfer interruption of Lcdb	DMA transfer request for Qspi	Comparator detection1	Comparator detection0	USB D1FIFO transfer request	USB D0FIFO transfer request	15-bit interval timer interrupt	Ssi dma rx requests

Note Bits that are not assigned a function must be set to "0".

Remark i=0~4

20.3.12 DMA base address register (DMABAR)

This is a 32-bit register that sets the vector address that holds the starting address of the DMA control data area and the address of the DMA control data area.

Note 1. The DMABAR register must be changed in a state where all DMA boot sources are set to disable booting.

2. DMABAR registers can only be overwritten 1 time.

3. Access to the DMABAR register cannot be performed via DMA transfer.

4. Refer to the notices in "20.3.1Allocation of DMA control data area and DMA vector table area”

5. Set the register to maintain 1024-byte alignment, that is, set the low 10 bit to zero. DMA hardware ignores low 10 bits.

6. The register can only be accessed by WORD, BYTE and HALFWORD access are ignored.

Figure 20-13 Format of DMA base address register (DMABAR)

Address: 40005008H After reset: 00000000H R/W

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

20.4 Operation of DMA

Once the DMA is started, the control data is read from the DMA control data area, data transfer is performed based on the control data, and the control data transferred is written back. It is possible to store 40 sets of control data to a DMA control data area and to perform transfer of 40 sets of data. The transfer mode has a normal mode and a repeat mode, and the transfer size has 8-bit transfer, 16-bit transfer and 32-bit transfer. When the CHNE bit of the DMACRj (j=0~39) register is '1' (allowing chain transfer), the continuous data transfer (chain transfer) is read by 1 start-up sources.

The transmission source address and the transmission destination address are specified through the 32-bit DMSARj register and the 32-bit DMDARj register, respectively. After the data is transferred, the values of the DMSARj register and the DMDARj register are incremented or fixed according to the control data.

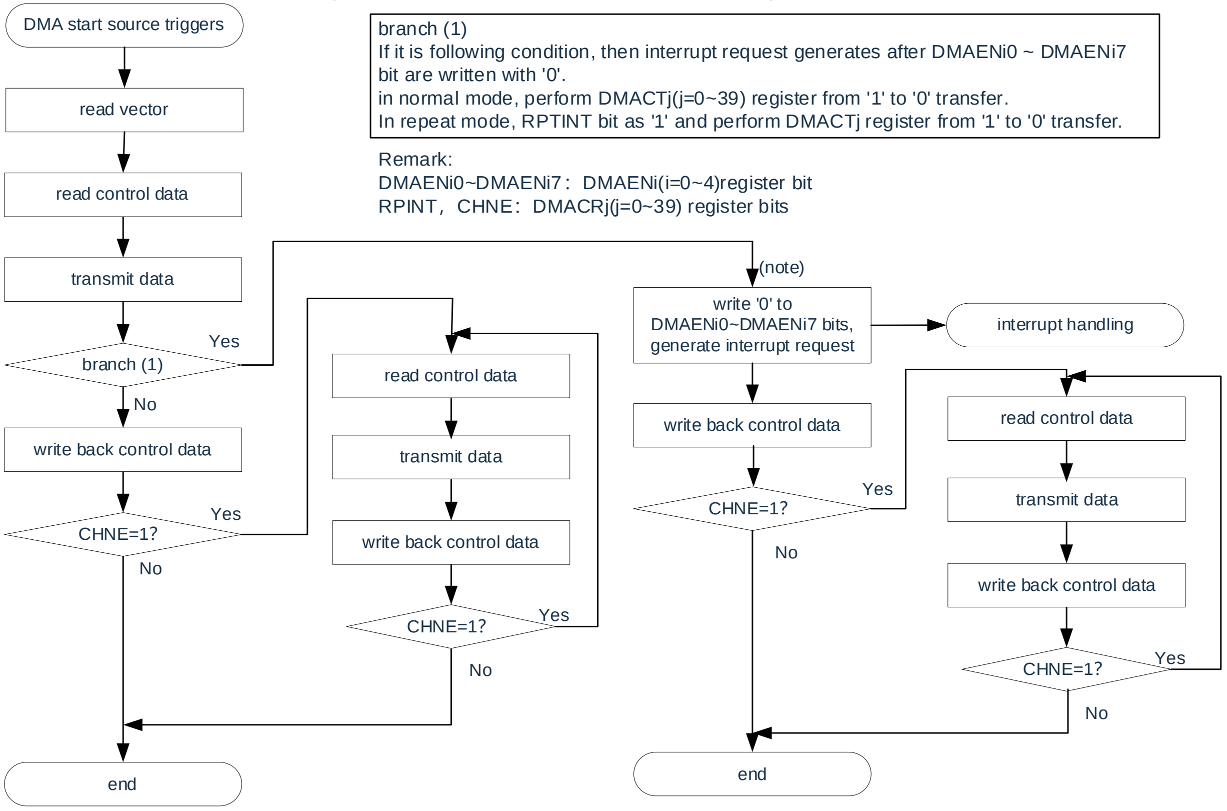
20.4.1 Boot source

The DMA is started by the interrupt signal of the peripheral function, and the interrupt signal to start the DMA is selected by the DMAENi (i=0 ~ 3) register. When the data transfer (the initial transfer is performed continuously when the chain transfer is performed) is set as follows, the DMAENi0 to DMAENi7 bits of the corresponding DMAENi register are set to "0" (disable start) in DMA operation.

- In normal mode, the DMACTj (j=0~39) register is transferred to "0".
- In repeat mode, the RPTINT bit of the DMACRj register is '1' (interrupt allowed) and the DMACTj register is '0' transferred.

The internal flow chart of the DMA is shown in Figure 20-14.

Figure 20-14 Flowchart of internal operation of DMA



branch (1)
If it is following condition, then interrupt request generates after DMAENi0 ~ DMAENi7 bit are written with '0'.
in normal mode, perform DMACTj(j=0~39) register from '1' to '0' transfer.
In repeat mode, RPTINT bit as '1' and perform DMACTj register from '1' to '0' transfer.

Remark:
DMAENi0~DMAENi7： DMAENi(i=0~4)register bit
RPINT， CHNE： DMACRj(j=0~39) register bits

Note: In a data transfer initiated by a setting allowing chain transfer (CHNE=1), the DMAENi0~DMAENi7 bit is not written "0" and no interrupt request is generated.

20.4.2 Normal mode

In the case of 8-bit transmission, the transmission data for 1 start is 1 to 65535 bytes; in the case of 16-bit transmission, the transmission data for 1 start is 2 to 131070 bytes; in the case of 32-bit transmission, the transmission data for 1 start is 4 to 262140 bytes. The number of transmissions is 1 to 65535 times. If the DMACTj (j=0～39) register becomes "0", the interrupt request corresponding to the start-up source is generated to the interrupt controller during DMA operation, and the DMAENi0 to DMAENi7 bits of the corresponding DMAENi (i=0～4) register are set to "0" (disable start).

Register functions and data transfers in normal mode are shown in Table 20-7 and Figure 20-15 respectively.

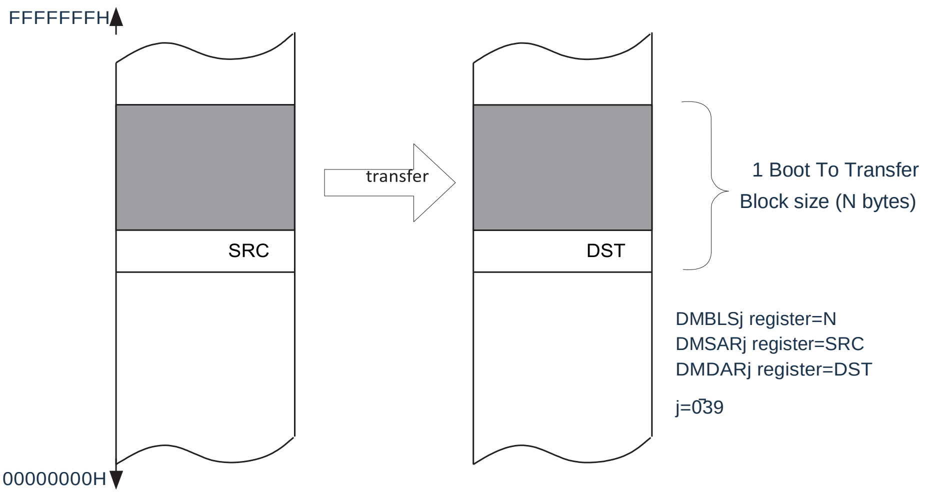
Table 20-7 Register function in normal mode

Register name	Symbol	Features
DMA block size register j	DMBLSj	1 Boot The size of the data block to be transferred
DMA transfer number register j	DMACTj	Number of data transfers
DMA transfer times reload register j	DMRLDj	Do not use Note.
DMA source address register j	DMSARj	The source address where the data is sent
DMA destination address register j	DMDARj	Destination address for data transfer

Note When parity error reset (RPERDIS=0) is allowed to occur through the RAM parity error detection function, initialization must be performed (00H).

Remark j=0~39

Figure 20-15 Data transfer in normal mode



Settings for the DMACR				Control of source address	Control of Target Address	Source address after delivery	Destination address after
DAMOD	SUMME	RPTSEL	MODE				
0	0	X	0	fixed	fixed	SRC	DST
0	1	X	0	incremental	fixed	SRC+N	DST
1	0	X	0	fixed	incremental	SRC	DST+N
1	1	X	0	incremental	incremental	SRC+N	DST+N

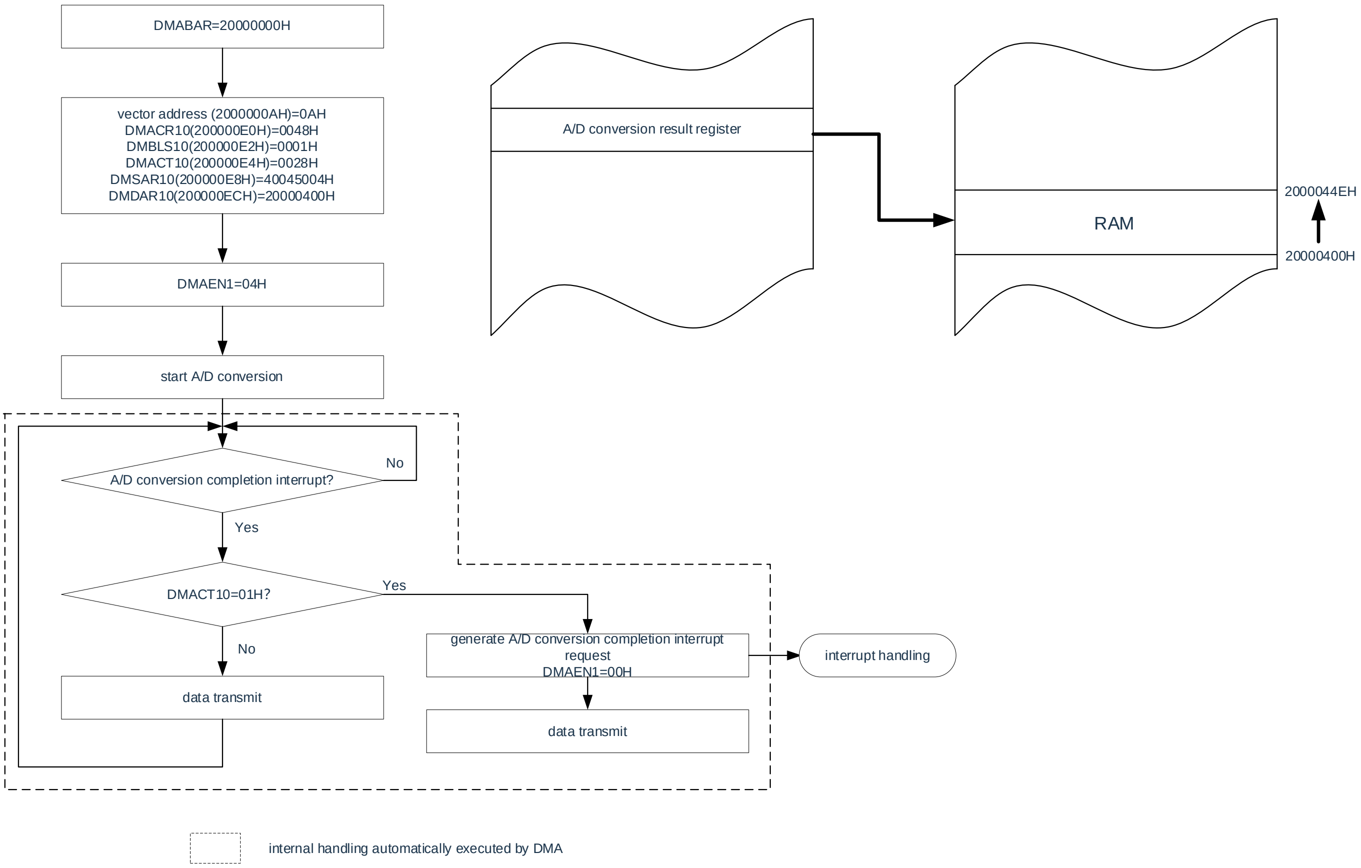
X:"0" or "1"

(1) Example of normal mode 1: Continuous reading A/D conversion results

The DMA is started by an A/D conversion end interrupt, and the value of the A/D conversion result register is transferred to the RAM.

- The vector address is assigned in 2000000AH and the control data is assigned in 200000E0H to 2000000EFH.
- Transfer 2 bytes of data from the A/D conversion result register (40045004H, 40045005H) 40 times to 80 bytes from 20000400H to 2000044FH of RAM.

Figure 20-16 Example of normal mode 1: Continuously reading A/D conversion results


DMABAR=20000000H
vector address (2000000AH)=0AH
DMACR10(200000E0H)=0048H
DMBLS10(200000E2H)=0001H
DMACT10(200000E4H)=0028H
DMSAR10(200000E8H)=40045004H
DMDAR10(200000ECH)=20000400H
DMAEN1=04H
start A/D conversion
A/D conversion completion interrupt?
No
Yes
DMACT10=01H?
Yes
No
data transmit
generate A/D conversion completion interrupt request
DMAEN1=00H
data transmit
interrupt handling
A/D conversion result register
RAM
2000044EH
20000400H
internal handling automatically executed by DMA

The value of the DMRLD10 register is not used because it is in normal mode. However, the DMRLD10 register must be initialized (0000H) when parity error reset (RPERDIS=0) is permitted via RAM parity.

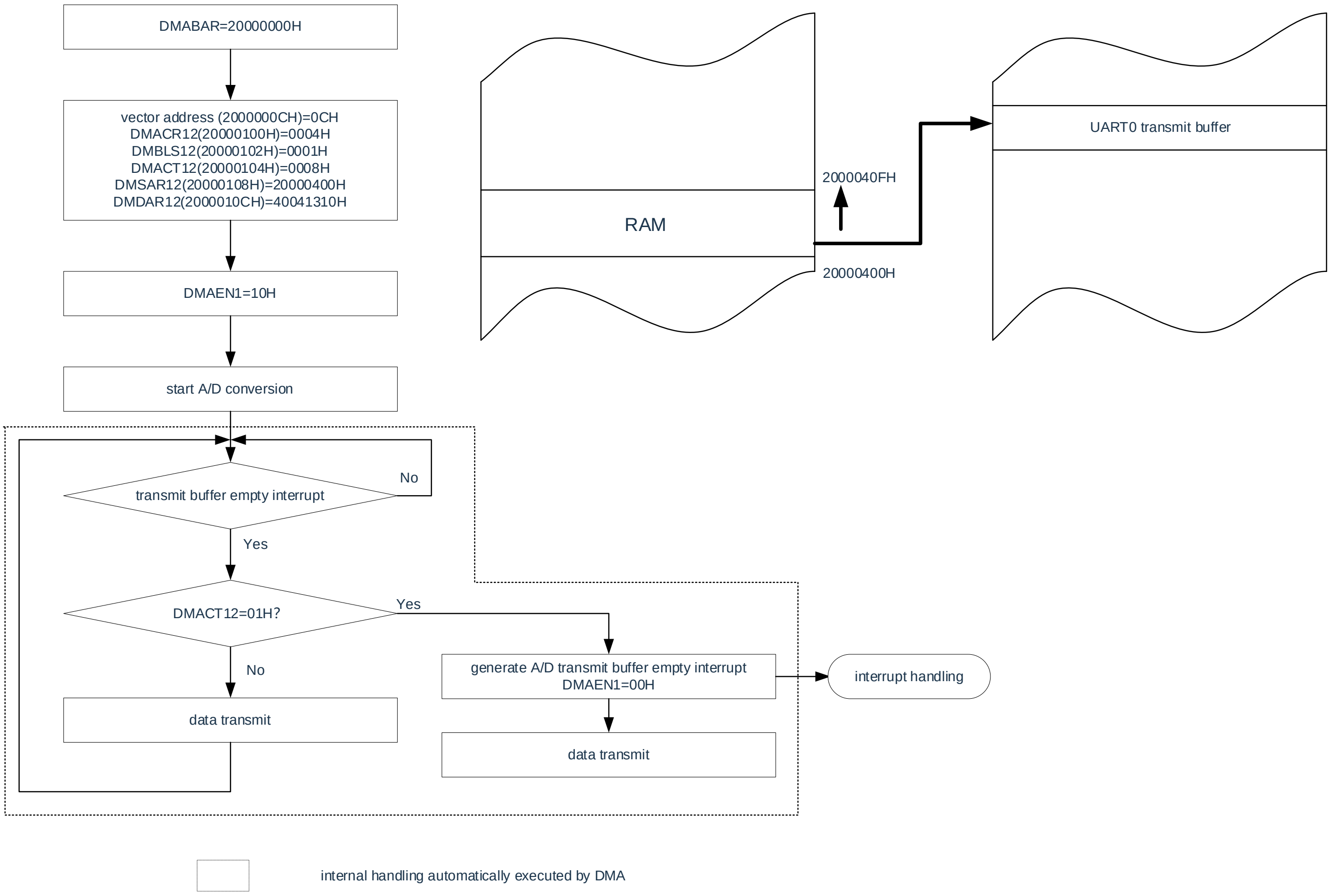
(2) Example of normal mode 2: UART0 continuous transmit

The DMA is initiated over the transmission buffer of the UART0 and the value of the RAM is transferred to the transmission buffer of the UART0.

· The vector address is assigned in 2000000CH, and the control data is assigned in 20000100H~2000010FH.

· Transfer 8 bytes of RAM 20000400H~2000407H to the UART0 transmission buffer (40041310H).

Figure 20-17 Example of normal mode usage 2: UART0 continuous transmit



The value of the DMRLD12 register is not used because it is in normal mode. However, the DMRLD12 register must be initialized (0000H) when parity error reset (RPERDIS=0) is permitted via RAM parity.

The 1st UART0 transmission must be started by software. The DMA is started by an air interrupt in the transmit buffer, and then the second and subsequent transmits are performed automatically.

20.4.3 Repeat mode

The transmission data for a single start is 1 to 65535 bytes. Specify the transmission source or transmission destination as a repeat zone, and the number of transmissions is 1 to 65535 times. Once the specified number of transfers is completed, the DMACTj (j=0 to 39) register and the address designated as the repeat area are initialized, and then the transfer is repeated. When the RPTINT bit of the DMACRj register is "1" (interrupt is allowed) and the DMACTj register becomes "0" for data transfer, an interrupt request corresponding to the start source is generated to the interrupt controller during DMA operation. The DMAENi0 to DMAENi7 of the corresponding DMAENi (i=0 to 4) registers are set to "0" (disable start). When the RPTINT bit of the DMACRj register is "0" (interrupt is prohibited), no interrupt request is generated even if the DMACTj register becomes "0" for data transfer, and the DMAENi0 to DMAENi7 bits remain unchanged at "0".

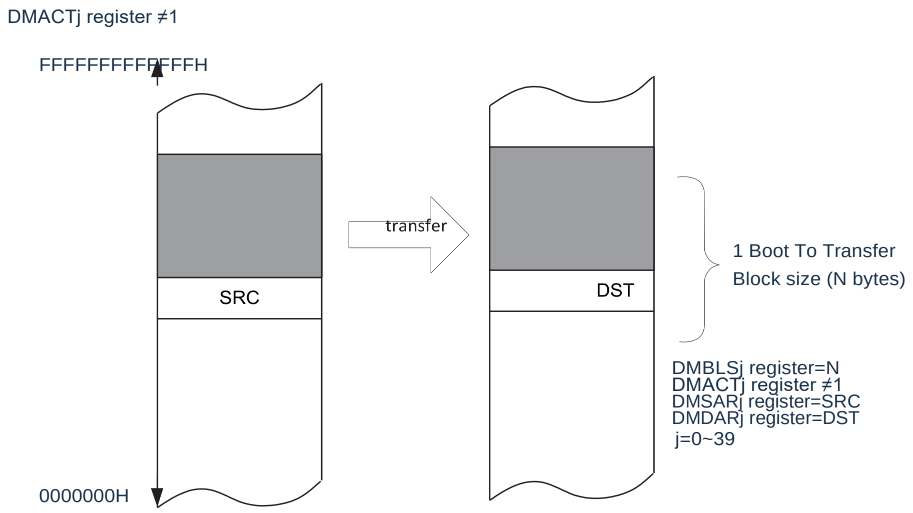
Register functions and data transfer for the repeat mode are shown in Tables 20-8 and Figures 20-18 respectively.

Table 20-8 Repeat mode register function

Register name	Symbol	Features
DMA block size register j	DMBLSj	The size of the data block to be transferred in a single start
DMA transfer times register j	DMACTj	Number of data transfers
DMA transfer times reload register j	DMRLDj	Reload the value of this register into the DMACT register. (initialize the number of transfers of data)
DMA source address register j	DMSARj	The source address where the data is sent
DMA target address register j	DMDARj	Destination address for data transfer

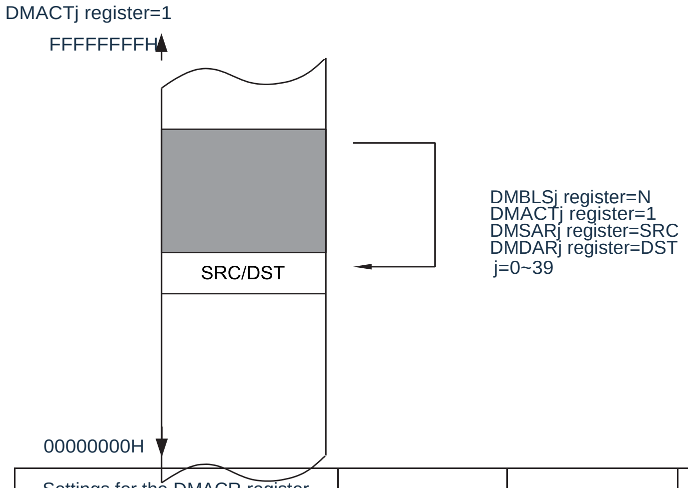
Remark j=0~39

Figure 20-18 Repeat mode data transfer



Settings for the DMACR register				Control of source address	Control of target address	Source address after transfer	Target address after transfer
DAMOD	SUMMER	RPTSEL	MODE				
0	X	1	1	repeat area	fixed	SRC+N	DST
1	X	1	1	repeat area	incremental	SRC+N	DST+N
X	0	0	1	fixed	repeat area	SRC	DST+N
X	1	0	1	incremental	repeat area	SRC+N	DST+N

X:"0" or "1"



Settings for the DMACR register				Control of source address	Control of target address	Source address after transfer	Target address after transfer
DAMOD	SUMMER	RPTSEL	MODE				
0	X	1	1	repeat area	fixed	SRC	DST
1	X	1	1	repeat area	incremental	SRC	DST+N
X	0	0	1	fixed	repeat area	SRC	DST
X	1	0	1	incremental	repeat area	SRC+N	DST

X: '0' or '1'

Note 1. When using repeat mode, you must set the data length of the repeat region within 65535 bytes.

(1) Example of using repeat mode 1: pulse output using stepper motor control of the port

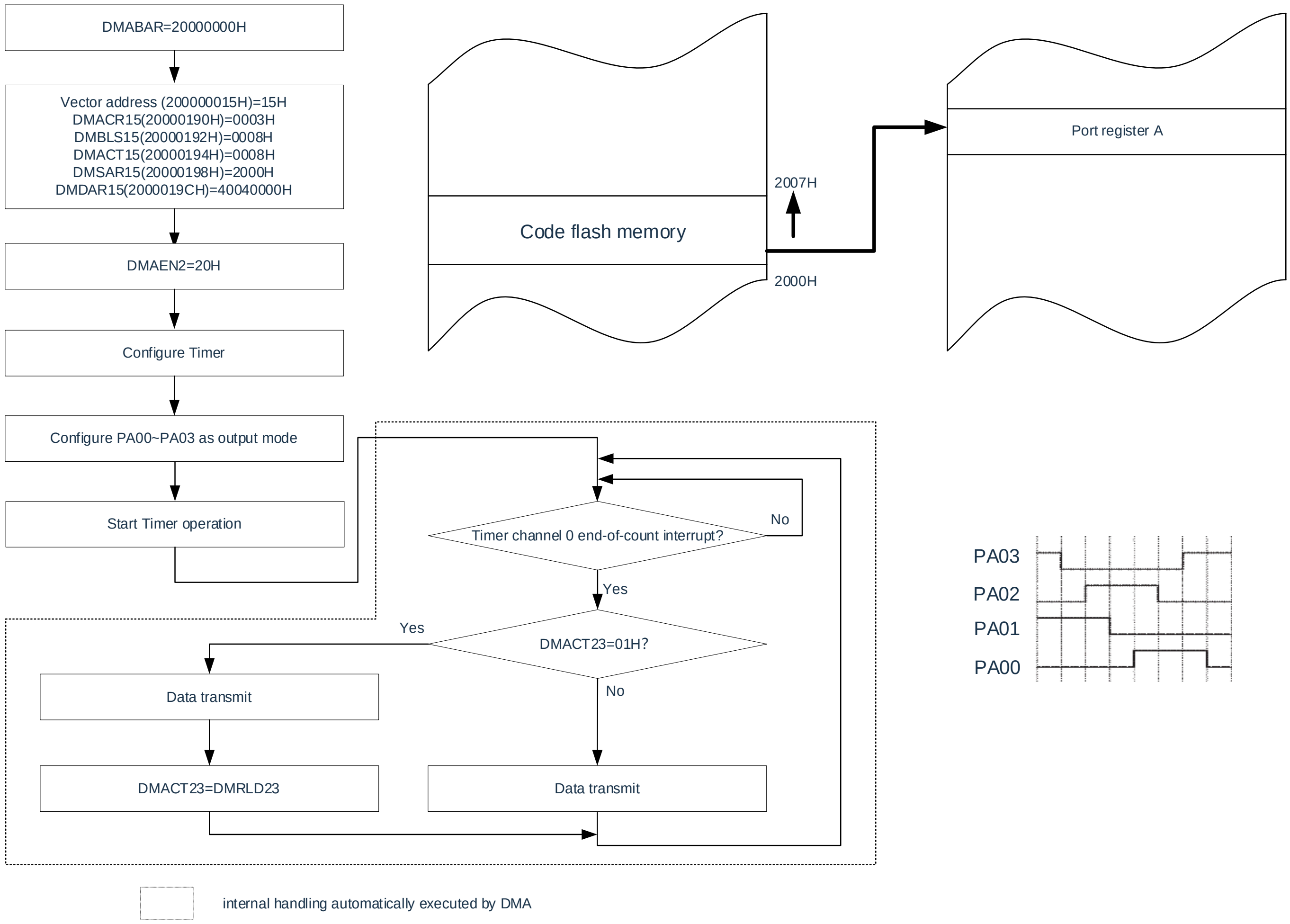
The DMA is started using Timer4's channel 0 interval timer function and the pattern of motor control pulses saved in the code flash is transferred to the general purpose port.

· The vector address is assigned at 20000015H and the control data is assigned at 20000190H~2000019FH.

· Transfer 8 bytes from 02000H to 02007H of code flash to port register A (40040001H).

· Disable repeat mode interrupts.

Figure 20-19 Example of using repeat mode 1: Pulse output using stepper motor control of the port



To stop the output, you must clear the bit5 of the DMAEN2 after stopping the timer running.

20.4.4 Chain transfer

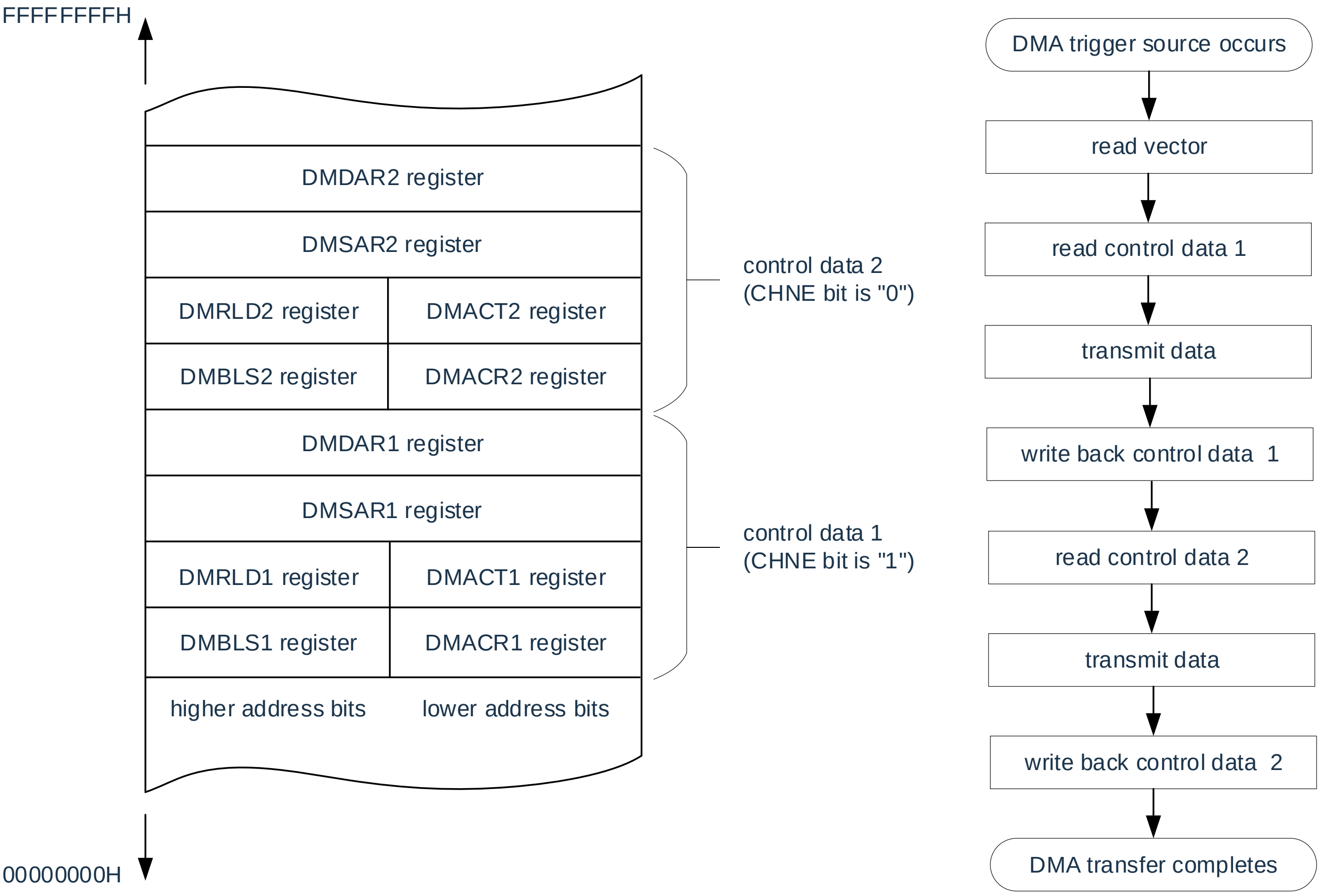
When the CHNE bit of the DMACRj (j=0 to 39) register is "1 (chain transfer allowed), multiple data transfers can be performed continuously by a start source.

Once the DMA is started, control data assigned to the DMA control data area is read by selecting control data from data read from a vector address corresponding to the start source. If the CHNE bit of the read control data is '1' (allowing chain transmission), the transmission is continued by reading the next allocated control data after transmission. Repeat until the control data transfer with the CHNE bit "0" (Chain Transfer Disabled) is complete.

When the plurality of control data is used for chain transmission, the transmission times of the first control data setting are valid, and the transmission times of the second subsequent processing are invalid.

The flow chart for chain transfer is shown in Figure 20-20.

Figure 20-20 Flow chart for chain transfer



Note 1. The CHNE bit of the DMACR39 register must be set to "0" (to disable chain transmission).

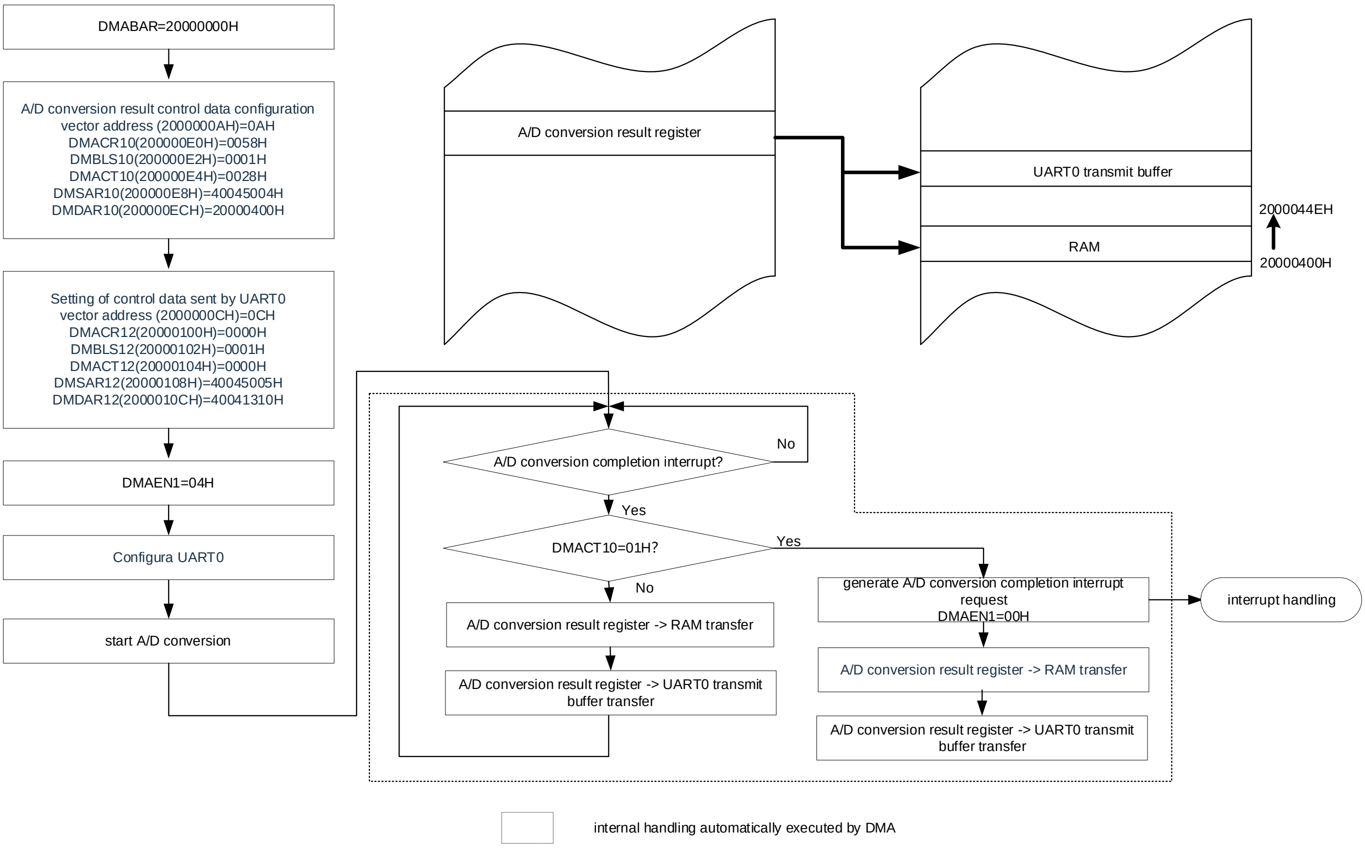
2. When data is transferred after the second time of chain transfer, DMAENi0~DMAENi7 bit of DMAENi (i=0~4) is unchanged as '0' (DMA start disabled) and no interrupt request is generated.

(1) Examples of the use of chain transmission: Continuously taking A/D conversion result for UART0 transmission

The DMA is started by an A/D conversion end interrupt and the A/D conversion result is transmitted to RAM for UART0 transmission.

- The vector addresses are 200000AH and 200000CH respectively.
- The control data of A/D conversion results were distributed in the range of 200000E0H~200000 EFH.
- The control data sent by UART0 is distributed in the range of 20000100H~2000010FH.
- The two-byte data of 40045004H (40045005H) is transferred to 20000400H-2000044FH of RAM, and the high 1-byte (40045005H) of A/D is transferred to UART0 transmission buffer (40041310H).

Figure 20-22 Examples of the use of chain transmission: Continuously taking A/D conversion result for UART0 transmission


DMABAR=20000000H
A/D conversion result control data configuration
vector address (2000000AH)=0AH
DMACR10(200000E0H)=0058H
DMBLS10(200000E2H)=0001H
DMACT10(200000E4H)=0028H
DMSAR10(200000E8H)=40045004H
DMDAR10(200000ECH)=20000400H
Setting of control data sent by UART0
vector address (2000000CH)=0CH
DMACR12(20000100H)=0000H
DMBLS12(20000102H)=0001H
DMACT12(20000104H)=0000H
DMSAR12(20000108H)=40045005H
DMDAR12(2000010CH)=40041310H
DMAEN1=04H
Configura UART0
start A/D conversion
A/D conversion result register
UART0 transmit buffer
RAM
2000044EH
20000400H
A/D conversion completion interrupt?
No
Yes
DMACT10=01H?
Yes
No
A/D conversion result register -> RAM transfer
A/D conversion result register -> UART0 transmit buffer transfer
generate A/D conversion completion interrupt request
DMAEN1=00H
A/D conversion result register -> RAM transfer
A/D conversion result register -> UART0 transmit buffer transfer
interrupt handling
internal handling automatically executed by DMA

20.5 Cautions when using DMA

20.5.1 DMA control data and vector table settings

- The DMA base address register (DMABAR) must be changed in a state where all DMA boot sources are set to disable boot.
- The DMA base address register (DMABAR) can only be overridden 1 time.
- You must change the DMACRj, DMBLSj, DMACTj, DMRLDj, DMSARj, DMDARj register data when the DMAENi0~DMAENi7 bit of the corresponding DMAENi (i=0~4) is '0' (DMA boot disabled) register.
- The starting address of the DMA control data area set in the vector table must be changed when the DMAENi0~DMAENi7 bit of the corresponding DMAENi (i=0~4) register is '0' (DMA boot disabled).

20.5.2 Allocation of DMA control data area and DMA vector table area

Areas where DMA control data and vector tables can be allocated vary depending on the product and usage conditions.

- The stack, DMA control data, and DMA vector table sections cannot overlap.
- The DMRLD register must be initialized (0000H) even when normal mode is used when parity error reset (RPERDIS=0) is allowed by RAM.

20.5.3 Number of execution clocks for DMA

The execution of the DMA at start-up and the number of clocks required are shown in Table 20-9.

Table 20-9 Execution and number of clocks required when DMA is started

Reading vector	Control data		Reading 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 the control data, refer to "Table 20-10 Number of Clocks Required to Write Back the Control Data".

2. For the number of clocks required to read and write data, refer to "Table 20-11 Number of clocks required to read and write data".

Table 20-10. Number of clocks required to write back the control data

Settings for the DMACR register				Address settings		Control register write-back				Clock Count
DAMOD	SUMMER	RPTSEL	MODE	Source	Objectives	DMACTj register	DMRLDj register	DMSARj register	DMDARj register	
0	0	X	0	fixed	fixed	write-back	write-back	Do not write back	Do not write back	1
0	1	X	0	incremental	fixed	write-back	write-back	write-back	Do not write back	2
1	0	X	0	fixed	incremental	write-back	write-back	Do not write back	write-back	2
1	1	X	0	incremental	incremental	write-back	write-back	write-back	write-back	3
0	X	1	1	repeat area	fixed	write-back	write-back	write-back	Do not write back	2
1	X	1	1		incremental	write-back	write-back	write-back	write-back	3
X	0	0	1	fixed	repeat area	write-back	write-back	Do not write back	write-back	2
X	1	0	1	incremental		write-back	write-back	write-back	write-back	3

Remark j=0~23,X:"0" or "1"

Table 20-11 Number of clocks required to read and write data

Execution status	RAM	Code flash	Data flash	Special function register (SFR)	Extended Special Function Register (2ndSFR)	
					No Wait	Wait
Read data	1	2	4	1	1	1+Waiting number NOTE
write data	1	—	—	1	1	1+Waiting number NOTE

20.5.4 Response time for DMA

The DMA response time is shown in Table 20-12. The DMA response time is the time from the detection of the DMA boot source to the start of the DMA transfer, excluding the number of DMA execution clocks.

Table 20-12 Response time for DMA

	Minimum time	Maximum time
Response time	3 clocks	23 clocks

However, the DMA response may also be delayed in the following cases. The number of delayed clocks varies depending on the condition.

· Maximum response time to execute instructions from internal RAM: 20 clocks

Remark: 1 clock: 1/f_{CLK} (f_{CLK}: CPU/peripheral hardware clock)

20.5.5 Start source for DMA

- You cannot enter the same boot source during the period from the input DMA boot source to the end of the DMA transfer.
- At the location where the DMA boot source was generated, the corresponding DMA boot enable bit for the boot source cannot be operated.
- If that DMA start source transmit contention, the priority is determine when the CPU accept the DMA transfer to start the start source. Refer to the "20.3.3 Vector Table" for starting source priority.
- If DMA start is permitted in one of the following states, DMA transfer is started and an interrupt is generated after transfer. Therefore, the monitor flag (CnMON) of the comparator must be confirmed as necessary to allow the DMA to start.
  - Set to generate interrupt requests by single edge detection of comparator [Note] (CnEDG=0) and by rising edge of comparator (CnEPO=0) and IVCMP>IVREF (or internal reference voltage = 1.45V).
  - Set to generate interrupt requests by single edge detection of the comparator (CnEDG=0) and by falling edge of the comparator (CnEPO=1) and IVCMP < IVREF (or internal reference voltage = 1.45V).

(n=0, 1)

20.5.6 Operation in standby mode

Status	DMA Operation
Sleep mode	Enable operation (not allowed in low power RTC mode).
Deep sleep mode	Able to accept DMA boot source and perform DMA transfer Note 1

Note 1. In deep sleep mode, DMA transfer can be performed after a DMA start source is detected, and return to deep sleep mode after transfer. However, flash memory cannot be set as a transfer source because that code flash and data flash memory stop running in deep sleep mode.

Chapter 21 Linkage Controller(EVENTC)

21.1 Function of EVENTC

EVENTC links events output by each peripheral function to each other by the peripheral function. The event link allows for collaborative operation between peripheral functions without passing through the CPU.

EVENTC has the following features:

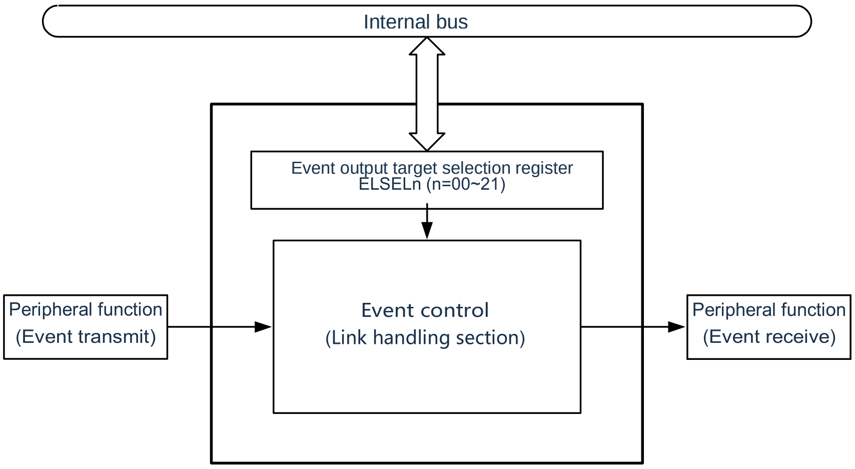
· Depending on the product, it is possible to link the event signals of 15 peripheral functions directly to the specified peripheral function.

· Depending on the product, it is possible to use the event signal as a start-up source for the operation of 1 of the 4 peripheral functions.

21.2 Structure of EVENTC

The EVENTC box is shown in Figure 21-1.

Figure 21-1 Block diagram of EVENTC



21.3 Control register

The controller register is shown in Table 21-1.

Table 21-1 Registers for controlling EVENTC

Register name	Symbol
Event output target selection register 00	ELSEL00
Event output target selection register 01	ELSEL01
Event output target selection register 02	ELSEL02
Event output target selection register 03	ELSEL03
Event output target selection register 04	ELSEL04
Event output target selection register 05	ELSEL05
Event output target selection register 06	ELSEL06
Event output target selection register 07	ELSEL07
Event output target selection register 08	ELSEL08
Event output target selection register 09	ELSEL09
Event output target selection register 10	ELSEL10
Event output target selection register 11	ELSEL11
Event output target selection register 12	ELSEL12
Event output target selection register 13	ELSEL13
Event output target selection register 14	ELSEL14
Event output target selection register 15	ELSEL15
Event output target selection register 16	ELSEL16
Event output target selection register 17	ELSEL17
Event output target selection register 18	ELSEL18
Event output target selection register 19	ELSEL19
Event output target selection register 20	ELSEL20
Event output target selection register 21	ELSEL21

21.3.1 Output target selection register n(ELSELRn) (n=00~14)

The ELSELRn register links each event signal to an event recipient peripheral function (link target peripheral function) when the event is accepted. Multiple event inputs cannot be linked to the same event output target (event recipient). Otherwise, the operation of the event recipient peripheral function may be inconsistent and the event signal may not be normally accepted. In addition, event link generation source and event output target cannot be set to the same function.

The ELSELRn register must be set during the period when all event output peripheral functions do not generate event signals.

The ELSELRn register (n=00~14) and peripheral function are shown in Table 21-2, and the ELSELRn register (n=00~14) are shown in Table 21-3.

Figure 21-2 Format of event output target selection register n (ELSELRn)

Address: 40043400H (ELSELR00)~4004340EH(ELSELR14) After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
ELSELRn	0	0	0	0	0	ELSELn2	ELSELn1	ELSELn0

ELSELn2	ELSELn1	ELSELn0	Selection of event links
0	0	0	Disable event links.
0	0	1	Select the linked peripheral function 1 for operation Note 1.
0	1	0	Select the linked peripheral function 2 for operation Note 1.
0	1	1	Select the linked peripheral function 3 for operation Note 1.
1	0	0	Select the linked peripheral function 4 for operation Note 1.
Others			Disable setting.

Note 1. Refer to “Table 21-3 Correspondence between the set value of the ELSELRn register (n=00 to 14) and the operation when the link target peripheral function accepts the event”.

Table 21-2 Correspondence of ELSELRn registers (n=00 ~ 14) and peripheral functions

Register name	Event generation source (output source for event input n)	Event content
ELSELR00	External interrupt edge detection 0	INTP0
ELSELR01	External interrupt edge detection 1	INTP1
ELSELR02	External interrupt edge detection 2	INTP2
ELSELR03	External interrupt edge detection 3	INTP3
ELSELR04	RTC fixed cycle/alarm clock consistency detection	INTRTC
ELSELR05	End of count/end of capture for Timer4 channel 00	INTTM00
ELSELR06	End of count/end of capture for Timer4 channel 01	INTTM01
ELSELR07	End of count/end of capture for Timer4 channel 02	INTTM02
ELSELR08	End of count/end of capture for Timer4 channel 03	INTTM03
ELSELR09	End of count/end of capture for Timer8 channel 00	INTTM10
ELSELR10	End of count/end of capture for Timer8 channel 01	INTTM11
ELSELR11	End of count/end of capture for Timer8 channel 02	INTTM12
ELSELR12	End of count/end of capture for Timer8 channel 03	INTTM13
ELSELR13	Comparator detection 0	INTCMP0
ELSELR14	Comparator detection 1	INTCMP1

Table 21-3 Correspondence between the set value of the ELSELRn register (n=00 to 14) and the operation when the link target peripheral function accepts the event

ELSELn3~ELSELn0 bits of ELSELRn register	Link target No.	Link target peripheral function	Operation when event is accepted
001B	1	A/D converter	Start A/D conversion.
010B	2	Timer input for Timer4 channel 0 Note 1	Delay counter, measurement of input pulse interval, external event counterc
011B	3	Timer input for Timer4 channel 1Note 2	Delay counter, measurement of input pulse interval, external event counter
100B	4	EPWM output control of truncation sources	Forced pulse output truncation

Note 1. To select timer input for Timer4 channel 0 as link target peripheral function, the running clock of channel 0 must be set to f_{CLK} through timer clock select register 0 (NFEN1) and TI00 pin (TNFEN00=0) and sets the timer input used by channel 0 as the event input signal of the linkage controller by the timer input selection register 0 (TIS0).

2. To select the timer input of Timer4 channel 1 as the linking target peripheral function, the running clock of channel 1 must be set to f_{CLK} through timer clock selection register 0 (TPS0), the noise filter of TI01 pin is set to OFF (TNFEN01=0) through noise filter permit register 1 (NFEN1), and the timer input of channel 1 is set to EVENTC event input signal through timer input selection register 0 (TIS0).

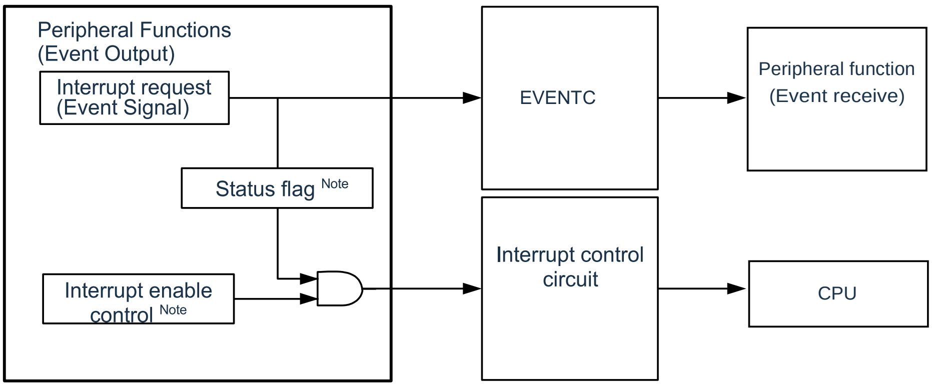
21.4 Operation of EVENTC

The path used for the interrupt request of the interrupt control circuit and the path used for 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 peripheral function operation of the event receiver.

The relationship between interrupt handling and EVENTC is shown in Figure 21-3. This diagram is a relation that takes as an example peripheral functions with the interrupt request status flag and the interrupt enable bit (control enable or disable).

The operation of the peripheral function that receives an event via EVENTC is based on the operation of the recipient peripheral function after receiving the event (refer to "Table 21-3 Correspondence between the set value of the ELSELRn register (n=00 to 14) and the operation when the link target peripheral function accepts the event").

Figure 21-3 Relationship between interrupt handling and EVENTC



Note Some peripheral features do not have this feature.

The responses of the peripheral functions that accept the events are shown in Table 21-4.

Table 21-4 Accept the response of the event's peripheral function

Event acceptance target No.	Function of the event link target	Run after event acceptance	Response
1	A/D converter	A/D conversion	The EVENTC event changes directly to the hardware trigger of the A/D conversion.
2	Timer input for Timer4 channel 0	Delay counter input pulse width measurement of external event counter	Edge detection is perform after 3 or 4 f_{CLK} cycles from that EVENTC event occur.
3	Timer input for Timer4 channel 1	Delay counter input pulse width measurement of external event counter	Edge detection is perform after 3 or 4 f_{CLK} cycles from that EVENTC event occur.
4	EPWM output control of truncated sources	Forced truncation of pulse output	After 2 or 3 EPWM operating clock cycles from the occurrence of an EVENTC event, it becomes a forced truncation state.

Chapter 22 USB 2.0 Full-Speed Module (USBFS)

22.1 Overview

This product provides a USB 2.0 Full-Speed Module (USBFS) that is compatible with the USB 2.0 specification and can be used as either a host controller or a device controller. The USBFS module has a built-in USB transceiver and supports all transfer types defined by the USB 2.0 specification.

The USBFS module supports a maximum of 10 channels of data transfer FIFO, and any endpoint can be configured to channel 1~9 depending on the peripheral or communication requirements.

This product is also compatible with the BC 1.2 specification.

Table 22-1 lists the functions of USBFS, Figure 22-1 shows the functional block diagram, and Table 22-2 lists the I/O ports.

Table 22-1 USBFS Specifications

Parameters	Specification
Features	• USB Device Controller (UDC) and USB 2.0 transceiver supporting host controller, device controller and On-The-Go (OTG) functionality (one channel) • Host and device controller can be switched via software • Self-powered or bus-powered mode can be selected • Supports for Revision 1.2 of the Battery Charging Specification • USB LDO regulator is used to power the internal USB transceiver.
	Host controller features: • Full-speed transfer (12Mbps) and low-speed transfer (1.5Mbps) • SOF and automatic scheduling of packet transfer • Programmable intervals for isochronous and interrupt transfers
	Device controller features: • Full-speed transfer (12Mbps) and low-speed transfer (1.5Mbps) • Control of transfer stage • Device status control function • Automatic response function for SET_ADDRESS requests • SOF interpolation function
Communication data transmission type	• Control transfer • Bulk transfer • Interrupt transfer • Isochronous transfer
Pipe configuration	• FIFO buffer for USB commnunication • Up to 10 pipes can be selected, including the default control pipe • Any endpoint number can be assigned to pipes 1~9.
	Transfer conditions that can be set for each pipe: • Pipe 0: 64-byte single buffer control transfer • Pipe 1 and 2: 64-byte double buffer bulk transfer 256-byte double buffer isochronous transfer • Pipe 3~5: 64-byte double buffer bulk transfer • Pipe 6~9: 64-byte single buffer interrupt transfer
Others	• Complete transfer by counting transaction • Modify BRDY interrupt event notification timing (BFRE) • Automatically clear the FIFO buffer after reading data from the pipe specified by the DnFIFO (n=0, 1) port (DCLRM) • NAK setup function for generating response PIDs at the end of transfer (SHTNAK) • On-chip pull-up and pull-down resistors for USB_DP/USB_DM • HOCO clock that can be used as a USB clock.
Module stop function	Set the module stop state to reduce power consumption

The USBFS block diagram is shown in Figure 22-1.

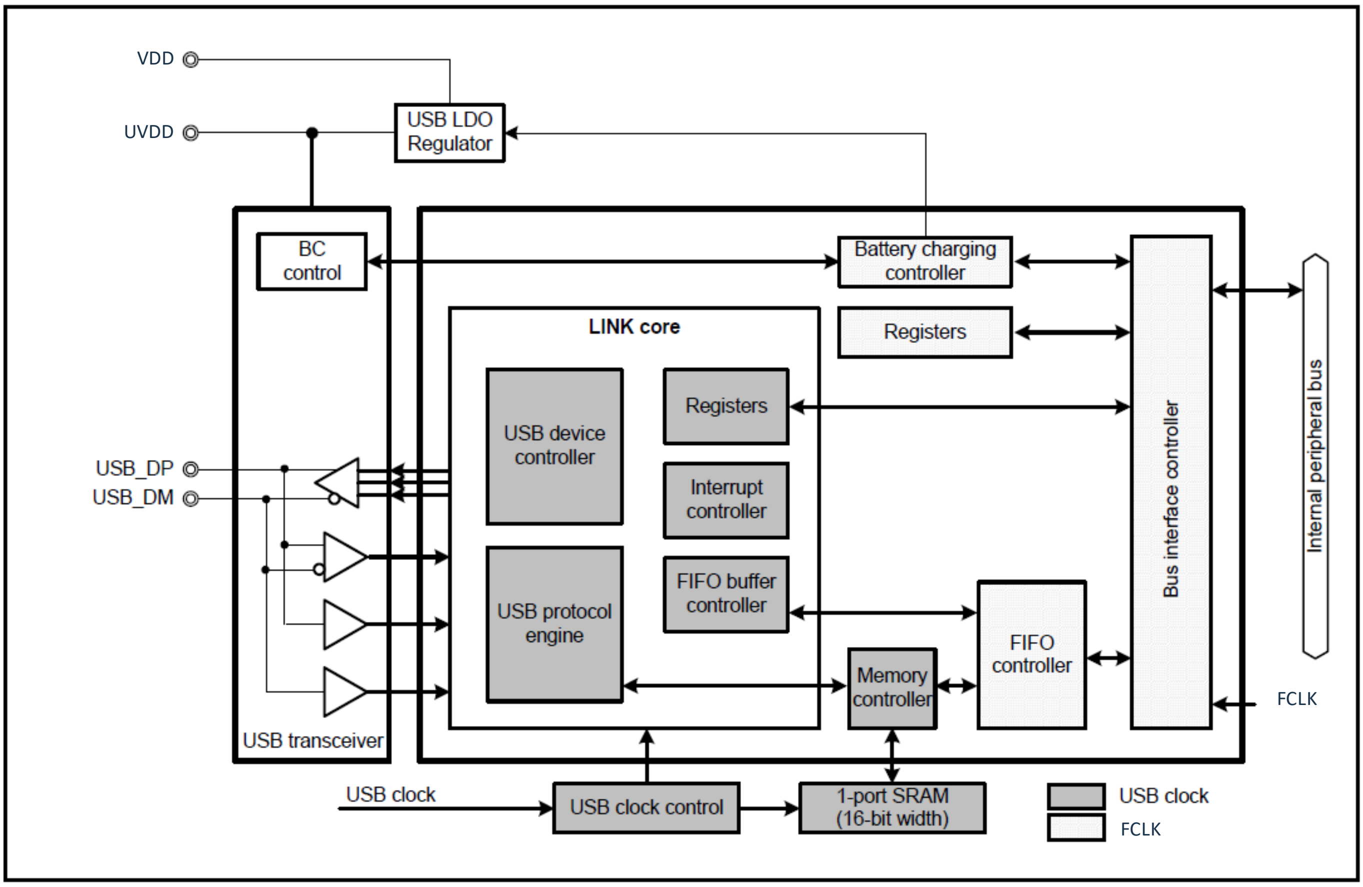


Figure 22-1 Block diagram of USBFS

Table 22-2 lists the I/O ports of the USBFS.

Table 22-2 USBFS port configuration

Port	Pin name	I/O	Function
USBFS	USB_DP	I/O	Built-in D+ port of USB transceiver It must be connected to the D+ port of the USB bus
	USB_DM	I/O	Built-in D- port of USB transceiver It must be connected to the D- port of the USB bus
	USB_VBUS	Input	USB cable connects to monitoring port It must be connected to the VBUS signal of the USB bus. Detects the status of the VBUS port (connected or unconnected)*1 when the USBFS is used as a device controller
	USB_EXICEN	Output	Low-power control signal output to OTG power supply IC
	USB_VBUSEN	Output	VBUS (5V) enable signal Output to external power supply IC
	USB_OVRCURA USB_OVRCURB	Input	Overcurrent port It must be connected to an external overcurrent detection signal. When USBFS is connected to the OTG power supply IC, the overcurrent port must be connected to the VBUS comparator signal.
	USB_ID	Input	The ID input signal must be connected to the MicroAB connector when in OTG mode
Common	UVDD	I/O	Input: Power supply for USB transceiver Output: Output port of the USB LDO. External capacitors must be connected to this port

*1. PA09 is a 5V withstand voltage buffer.

22.2 Register description

22.2.1 Peripheral function enable register (PER2)

The PER2 register is a register to set the peripheral hardware to enable or disable operation. It reduces power consumption by stopping the clock to hardware that is not in use. To use the USB function, USBEN must be set to 1.

For details, see "4.3.8 Peripheral Enable Registers 0, 1, 2 (PER0, PER1, PER2)".

22.2.2 System configuration control register (SYSCFG)

Address: USBFS.SYSCFG 4008 0000h

	b15	b14	13b	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	SCKE	—	CNEN	—	DCFM	DRPD	DPRPU	DMRPU	—	—	USBE
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	USBE	USBFS operation enble	0: Operation disable 1: Operation enable	R/W
b2,b1	—	Reserved	These bits are read as 0, and the write value should be 0.	—
b3	DMRPU	D- resistance control*[1]	0: D- pull-up is disabled. 1: D- pull-up is enabled.	R/W
b4	DPRPU	D+ resistance control*[1]	0: D+ pull-up is disabled. 1: D+ pull-up is enabled.	R/W
b5	DRPD	D+/D- resistance control	0: D+/D- pull-down is disabled. 1: D+/D- pull-down is enabled.	R/W
b6	DCFM	Controller function selection	0: Slave controller selection 1: Host controller selection	R/W
b7	—	Reserved	These bits are read as 0, and the write value should be 0.	—
b8	CNEN	CNEN single-ended receiver enable	0: Single end receiver operation is disabled. 1: Single end receiver operation is enabled.	R/W
b9	—	Reserved	These bits are read as 0, and the write value should be 0.	—
b10	SCKE	USB clock enable*[2]	0: Stop supplying the clock signal to the USB. 1: Enable supplying the clock signal to the USB.	R/W
b15~b11	—	Reserved	These bits are read as 0, and the write value should be 0.	—

*1. Do not enable the DMRPU and DPRPU bits at the same time.

*2. After writing 1 to the SCKE bit, read it and confirm it is set to 1.

USBE bit (USBFS operation enable)

The USBE bit enables or disables operation of the USBFS.

When USBE is rewritten from 1 to 0, it initializes the register bits in Table 22-3. This bit can only be rewritten if the SCKE bit is 1. In host controller mode, the DRPD bit must be set to 1 to eliminate the chattering of the SYSSTS0.LNST[1:0] bit and to confirm that the USB bus state is stable before the bit can be set to 1.

Table 22-3 Registers initialized by writing 0 to the SYSCFG.USBE bit

Selected Function	Register	Bit	Remarks
Device controller	SYSSTS0	LNST[1:0]	The value is retained when the host controller is selected.
	DVSTCTR0	RHST[2:0]	-
	INTSTS0	DVSQ[2:0]	The value is retained when the host controller is selected.
	USBREQ	BREQUEST[7:0], BMREQUESTTYPE[7:0]	The value is retained when the host controller is selected.
	USBVAL	WVALUE[15:0]	The value is retained when the host controller is selected.
	USBINDX	WINDEX[15:0]	The value is retained when the host controller is selected.
	USBLENG	WLENTUH[15:0]	The value is retained when the host controller is selected.
Host controller	DVSTCTR0	RHST[2:0]	-
	FRMNUM	FRNM[10:0]	The value is retained when the device controller is selected.

DMRPU:

The DMRPU bit enables or disables pulling up the D- line when the device controller is selected.

When the DMRPU bit is set to 1 while the device controller is selected, the USBFS forces a pull-up of the D- line to notify the USB host of connection as a low-speed device. Modifying the DMRPU bit from 1 to 0 allows the USB to release the D- line, thus notifying the USB host of disconnection.

This bit should be set to 0 if the host controller is selected.

DPRPU:

The DPRPU bit enables or disables pulling up the D+ line when the device controller is selected.

When the DPRPU bit is set to 1 while the device controller is selected, the USBFS forces a pull-up of the D+ line to notify the USB host of connection as a full-speed device. Modifying the DPRPU bit from 1 to 0 allows the USB to release the D+ line, thus notifying the USB host of disconnection.

This bit should be set to 0 if the host controller is selected.

DRPD:

The DRPD bit enables or disables pulling down D+ and D– lines when the host controller is selected.

This bit should be set to 1 if the host controller is selected, and should be set to 0 if the device controller is selected.

DCFM:

The DCFM bit selects the function of the USBFS.

This bit should be modified when the DMRPU, DPRPU, and DRPD bits are all 0.

CNEN:

Setting the CNEN bit to 1 to enable the single end receiver and set the LNST bit to monitor the status of the D+ and D– lines.

The CNEN bit is used when the USBFS operates as a portable device for battery charging.

SCKE:

The SCKE bit disables or enables supplying 48-MHz clock signals to the USB.

When this bit is 0, only SYSCFG can be read from and written to; the other registers related to the USB cannot be read from or written to.

22.2.3 System configuration status register 0 (SYSSTS0)

Address: USBFS.SYSSTS0 4008 0004h

	b15	b14	13b	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	OVCMON[1:0]		—	—	—	—	—	—	—	HTACT	—	—	—	IDMON	LNST[1:0]	
Reset value	0*1	0*1	0	0	0	0	0	0	0	0	0	0	0	0*1	0	0

Bit	Symbol	Bit name	Description	R/W
b1,b0	LNST[1:0]	USB data line status monitor	To the status of the USB data line. See Table 22-4	R
b2	IDMON	External ID0 input monitor	0:USB_ID port is low 1:USB_ID port is high	R
b5~b3	—	Reserved	This bit is read as 0	—
b6	HTACT	USB host sequencer status monitor	0:Host sequencer is completely stopped. 1:Host sequencer is not completely stopped.	R
b13~b7	—	Reserved	This bit is read as 0	R
b15,b14	OVCMON[1:0]	External USB_OVRCURA/ USB_OVRCURB input monitor	OVCMON[1] bit indicates the status of USB_OVRCURA OVCMON[0] bit indicates the status of USB_OVRCURB	R

*1. Depends on the status of the USB_OVRCURA/USB_OVRCURB and USB_ID pins.

LNST[1:0]:

The LNST[1:0] flags indicate the state of the USB data lines (D+ and D- lines). Refer to Table 22-4.

The LNST[1:0] flags should be read after the connection processing (SYSCFG.DPRPU bit = 1) when the device controller is selected; whereas after enabling pull-down of the lines (SYSCFG.DRPD bit = 1) when the host controller is selected.

HTACT:

The HTACT flag is 0 when the host sequencer of the USBFS is completely stopped.

In host controller mode, check that the HTACT bit is 0 before setting the DVSTCTR0.UACT bit to 0 to put the USBFS in a suspending state or setting the SCKE bit to 0 to stop clock supply during communication.

OVCMON[1:0]:

The OCVMON[1:0] flags indicate the status of overcurrent from an external power supply chip.

Table 22-4 Status of USB data bus lines (D+ Line, D- Line)

LNST[1:0]bit	During full-speed operation	During low-speed operation
00b	SE0	SE0
01b	J-state	K-state
10b	K-state	J-state
11b	SE1	SE1

22.2.4 Device state control register 0 (DVSTCTR0)

Address: USBFS.DVSTCTR0 4008 0008h

	b15	b14	13b	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	HNPBTOA	EXICEN	VBUSEN	WKUP	RWUPE	USBRST	RESUME	UACT	—	RHST[2:0]		
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b2~b0	RHST[2:0]	USB bus reset status	- When the host controller is selected: b2 b0 0 0 0: Communication speed is not determined (powered state or no connection) 1 x x: USB bus reset in progress 0 0 1: Low-speed connection 0 1 0: Full-speed connection - When the device controller is selected: b2 b0 0 0 0: Communication speed is not determined 0 0 1: USB bus reset in progress or low-speed connection 0 1 0: USB bus reset in progress or full-speed connection	R
b3	—	Reserved	This bit is read as 0. The write value should be 0.	—
b4	UACT	USB bus enable	0: Downstream port is disabled (SOF transfer is disabled). 1: Downstream port is enabled (SOF transfer is enabled).	R/W
b5	RESUME	Resume output	0: Resume signal is not output. 1: Resume signal is output.	R/W
b6	USBRST	USB bus reset output	0: USB bus reset signal is not output. 1: USB bus reset signal is output.	R/W
b7	RWUPE	Wakeup detection Enable	0: Downstream port wakeup is disabled. 1: Downstream port wakeup is enabled.	R/W
b8	WKUP	Wakeup output	0: Remote wakeup signal is not output. 1: Remote wakeup signal is output.	R/W
b9	VBUSEN	USB_VBUSEN output pin control	0: External USB_VBUSEN pin outputs low. 1: External USB_VBUSEN pin outputs high.	R/W
b10	EXICEN	USB_EXICEN output pin control	0: External USB_EXICEN pin outputs low. 1: External USB_EXICEN pin outputs high.	R/W
b11	HNPBTOA	Host negotiation protocol (HNP) control	This bit is used when switching from device B to device A while in OTG mode. If the HNPBTOA bit is 1, the internal function control keeps the suspended state until the HNP processing ends even though SYSCFG.DPRPU = 0 or SYSCFG.DCFM = 1.	R/W
b15~b12	—	Reserved	This bit is read as 0. The write value should be 0.	—

RHST[2:0]:

The RHST[2:0] flags indicate the status of the USB bus reset.

When the host controller is selected, the RHST[2:0] bit indicate 100b after the USBRST bit has been set to 1. It fixes the value of the RHST[2:0] flags when 0 is written to the USBRST bit and the USBFS completes SE0 driving.

When the device controller is selected, the RHST[2:0] bit indicate 010b (connection while DPRPU = 1) or 001b (disconnection while DMRPU = 1) when the USBFS detects the USB bus reset, and a DVST interrupt is generated.

UACT bit (USB bus enable)

When set to 1 in host controller mode, the UACT bit enables USB bus operation by controlling the transfer of SOF packets to the USB bus in addition to data and reception. With this bit set to 1, the USBFS starts outputting SOF packets within one frame cycle. When UACT is set to 0, the USB enters an idle state after the SOF packet is output.

With this bit set to 0, the USB enters the idle state after outputting SOF packets.

The USB sets the UACT bit to 0 on any of the following conditions:

- A DTCH interrupt is detected during communication (while UACT = 1).
- An EOFERR interrupt is detected during communication (while UACT = 1).

Writing 1 to this bit should be done at the end of the USB reset processing (writing 0 to the USBRST bit) or at the end of the resume processing from the suspended state (writing 0 to the RESUME bit).

This bit should be set to 0 if the device controller is selected.

RESUME:

The RESUME bit controls the resume signal output when the host controller is selected.

Setting the RESUME bit to 1 allows the USBFS to drive the USB port to the K-state and output the resume signal. The USBFS sets the RESUME bit to 1 on detecting the remote wakeup signal while RWUPE is 1 in the USB suspended state.

With the RESUME bit is 1, the USBFS continues to output the K-state until the software sets this bit to 0. For the times defined in the USB 2.0 specification, the RESUME bit must be 1 (resume cycle). Set this bit to 1 only when the interface is in a suspended state. Write 1 to the UACT bit (write 0 to the RESUME bit) at the same time as the resume processing is completed.

In device controller mode, always set this bit to 0.

USBRST:

The USBRST bit controls the USB bus reset signal output when the host controller is selected. When the host controller is selected, setting this bit to 1 allows the USBFS to drive SE0 of the USB port to reset the USB bus. The USBFS continues to output SE0 while the USBRST bit is 1 until the bit is cleared to 0 by software. The USBRST bit should be 1 (= USB bus reset period) for the time defined by USB Specification 2.0.

Writing 1 to this bit during communication (the UACT bit = 1) or during the resume processing (the RESUME bit = 1) prevents the USBFS from starting the USB bus reset processing until both the UACT and RESUME bits become 0. Write 1 to the UACT bit simultaneously with the end of the USB bus reset processing (writing 0 to the USBRST bit).

This bit should be set to 0 if the device controller is selected.

RWUPE:

The RWUPE bit enables or disables the downstream port peripheral device to use the remote wakeup function (resume signal output) when the host controller is selected. With this bit set to 1, on detecting the remote wakeup signal, the USBFS detects the resume signal (K-state for 2.5 μs) from the downstream port device and performs the resume processing (drives the port to the K-state).

With this bit set to 0, the USBFS ignores the detected remote wakeup signal (K-state) from the peripheral device connected to the USB port. While the RWUPE bit is 1, the internal clock should not be stopped even in the suspended state (SYSCFG.SCKE bit should be set to 1).

This bit should be set to 0 if the device controller is selected.

WKUP:

The WKUP bit enables or disables outputting the remote wakeup signal (resume signal) to the USB bus when the device controller is selected.

The USBFS controls the output time of a remote wakeup signal. When this bit is set to 1, the USBFS sets this bit to 0 after outputting the 10-ms K-state. According to USB Specification 2.0, the USB bus idle state must be kept for 5 ms or longer before a remote wakeup signal is transmitted. If the USB writes 1 to this bit right after detection of the suspended state, the K-state will be output after 2 ms.

Do not write 1 to this bit, unless the device state is in the suspended state (INTSTS0.DVSQ[2:0] = 1xxb) and the USB host enables the remote wakeup signal. When this bit is set to 1, the internal clock must not be stopped even in the suspended state (SYSCFG.SCKE bit = 1). This bit should be set to 0 if the host controller is selected.

HNPBTOA:

The HNPBTOA bit is used when switching from device B to device A while in OTG mode.

If the HNPBTOA bit is 1, the internal function control keeps the suspended state until the HNP processing ends even though the SYSCFG.DPRPU bit = 0 or SYSCFG.DCFM = 1 is set. Even if the falling edge of the D+ signal is detected at this time, no resume (RESM) interrupt is generated.

Write 0 to this bit by software to terminate the HNP processing when connect to the host (pull-up on the target side) or the HNP processing timeout is detected.

22.2.5 CFIFO port register (CFIFO/CFIFOL) D0FIFO port register (D0FIFO/D0FIFOL) D1FIFO port register (D1FIFO/D1FIFOL)

(1) When MBW bit is 1

Address: USBFS.CFIFO 4008 0014h, USBFS.D0FIFO 4008 0018h, USBFS.D1FIFO 4008 001Ch

	b15	b14	13b	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	FIFOPORT[15:0]															
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(2) When MBW bit is 0

Address: USBFS.CFIFOL 4009 0014h, USBFS.D0FIFOL 4009 0018h, USBFS.D1FIFOL 4009 001Ch

	b7	b6	b5	b4	b3	b2	b1	b0
	FIFOPORT[7:0]							
Reset value:	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b15~b0	FIFOPORT[15:0]*1	FIFO port	This port is used for reading receive data from the FIFO buffer and writing transmit data to the FIFO buffer.	R

* 1. The valid bits depend on the MBW setting (CFIFOSEL.MBW, D0FIFOSEL.MBW and D1FIFOSEL.MBW) and the BIGEND setting (CFIFOSEL.BIGEND, D0FIFOSEL.BIGEND and D1FIFOSEL.BIGEND) in the relevant port select registers. See Table 22-5 and Table 22-6.

Three FIFO ports are available:

- CFIFO
- D0FIFO
- D1FIFO

Each FIFO port is configured with:

- Port registers (CFIFO, D0FIFO or D1FIFO) to handle reading data from and writing data to the FIFO buffer
- Port select register (CFIFOSEL,D0FIFOSEL or D1FIFOSEL) for selecting the pipe assigned to the FIFO port
- Port control Register (CFIFOCTR,D0FIFOCTR or D1FIFOCTR)

Each FIFO port has the following constraints:

- Access to the FIFO buffer for DCP control transfers via the CFIFO port
- Access to FIFO buffers for DMA or DTC transfers via D0FIFO or D1FIFO ports
- The CPU can also access the D0FIFO and D1FIFO ports.
- The pipe number selected in the CURPIPE[3:0] bits of the Port Select Register cannot be changed when using a function specific to the FIFO port (such as the DMA or DTC transfer function).
- Configure the registers of the FIFO port does not affect other FIFO ports
- The same pipe must not be assigned to two or more FIFO ports
- There are two FIFO buffer states, one state grants access to the CPU and the other grants access to the Serial Interface Engine (SIE). When the SIE has access privileges, the CPU cannot access the FIFO buffer.

FIFOPORT[15:0]:

When the FIFOPORT bit is accessed, the USBFS reads incoming data from the FIFO buffer or writes transmit data to the FIFO buffer. The FIFO port register can be accessed only when the FRDY bit in the corresponding port control register (CFIFOCTR, D0FIFOCTR or D1FIFOCTR) is 1.

Valid bits depend on the MBW setting (CFIFOSEL.MBW, D0FIFOSEL.MBW and D1FIFOSEL.MBW) and the BIGEND setting (CFIFOSEL.BIGEND, D0FIFOSEL.BIGEND and D1FIFOSEL.BIGEND) in the relevant port select registers. See Table 22-5 and Table 22-6.

Table 22-5 Endian operation in 16-bit access

CFIFOSEL.BIGEND bit D0FIFOSEL.BIGEND bit D1FIFOSEL.BIGEND bit	Bit[15:8]	Bit[7:0]
0	N+1 data	N+0 data
1	N+0 data	N+1 data

Table 22-6 Endian operation in 8-bit access

CFIFOSEL.BIGEND bit D0FIFOSEL.BIGEND bit D1FIFOSEL.BIGEND bit	Bit [15:8]	Bit [7:0]
0	Access prohibited [*1]	N+0 data
1	Access prohibited [*1]	N+0 data

*1. Writing to or reading from areas to which access is prohibited is prohibited.

22.2.6 CFIFO port select register (CFIFOSEL) D0FIFO port select register (D0FIFOSEL) D1FIFO port select register (D1FIFOSEL)

CFIFOSEL

Address: USBFS.CFIFOSEL 4008 0020h

	b15	b14	13b	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RCNT	REW	—	—	—	MBW	—	BIGEND	—	—	ISEL	—	CURPIPE[3:0]			
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b3~b0	CURPIPE[3:0]	CFIFO port access pipe specification	b3 b0 0 0 0 0: DCP (Default control pipe) 0 0 0 1: Pipe 1 0 0 1 0: Pipe 2 0 0 1 1: Pipe 3 0 1 0 0: Pipe 4 0 1 0 1: Pipe 5 0 1 1 0: Pipe 6 0 1 1 1: Pipe 7 1 0 0 0: Pipe 8 1 0 0 1: Pipe 9 Settings other than above are prohibited.	R/W
b4	—	Reserved	This bit is read as 0. The write value should be 0.	R
b5	ISEL	CFIFO port access direction when DCP is selected	0: Reading from the buffer memory is selected. 1: Writing to the buffer memory is selected.	R/W
b7, b6	—	Reserved	This bit is read as 0. The write value should be 0.	R
b8	BIGEND	CFIFO port endian control	0: Little endian 1: Big endian	R/W
b9	—	Reserved	This bit is read as 0. The write value should be 0.	R
b10	MBW	CFIFO port access bit width	0: 8-bit width 1: 16-bit width	R/W
b13~b11	—	Reserved	This bit is read as 0. The write value should be 0.	R
b14	REW	Buffer pointer rewind	0: The buffer pointer is not rewound. 1: The buffer pointer is rewound.	R/W*1
b15	RCNT	Read count mode	0: The DTLN[8:0] bit (CFIFOCTR.DTLN[8:0], D0FIFOCTR.DTLN[8:0], D1FIFOCTR.DTLN[8:0]) is cleared when all of the receive data has been read from the CFIFO. In double buffer mode, the DTLN[8:0] bit value is cleared when all the data has been read from only a single plane.) 1: The DTLN[8:0] bit is decremented each time the receive data is read from the CFIFO.	R/W

*1. Only 0 can be read.

The same pipe should not be specified by the CURPIPE[3:0] bits in the CFIFOSEL, D0FIFOSEL, and D1FIFOSEL registers. When the CURPIPE[3:0] bits in the D0FIFOSEL and D1FIFOSEL registers are set to 0000b, no pipe is selected.

The pipe number should not be changed while the DMA or DTC transfers are enabled.

CURPIPE[3:0]:

The CURPIPE[3:0] bits specify the pipe number using which data is read or written through the CFIFO port. After writing to these bits, read these bits to check that the written value agrees with the read value before proceeding to the next process.

Do not set the same pipe number to the CURPIPE[3:0] bits in the CFIFOSEL, D0FIFOSEL, and D1FIFOSEL registers.

Even if an attempt is made to modify the setting of these bits during access to the FIFO buffer, the current access setting is retained until the access is completed. Then, the modification becomes effective, thus enabling continuous access.

ISEL:

After writing to the ISEL bit with the DCP being a selected pipe, read this bit to check that the written value agrees with the read value before proceeding to the next process. Set this bit and the CURPIPE[3:0] bits simultaneously.

MBW:

The MBW bit specifies the bit width for accessing the CFIFO port.

While the selected pipe is receiving, set both the CURPIPE[3:0] bits and the MBW bits. A write operation to these bits starts an operation to read data from the FIFO buffer; do not change these bits until all data has been read. When reading the FIFO buffer, use the access size set in the MBW bits for the read.

When the selected pipe is transmitting, the bit width cannot be changed from 8-bit width to 16-bit width while data is being written to the buffer memory.

An odd number of bytes can also be written through byte-access control even when 16-bit width is selected.

REW:

The REW bit specifies whether or not to rewind the buffer pointer.

When the selected pipe is receiving, setting the REW bit to 1 while the FIFO buffer is being read allows re-reading the FIFO buffer from the first data. In double buffer mode, re-reading the currently-read FIFO buffer plane from the first data is allowed.

Do not set the REW bit to 1 simultaneously with modifying the CURPIPE[3:0] bits. Before setting the REW bit to 1, be sure to check that the FRDY flag is 1.

To re-write to the FIFO buffer again from the first data for the pipe in the transmitting, use the BCLR bit.

D0FIFOSEL, D1FIFOSEL

Address: USBFS.D0FIFOSEL 4008 0028h, USBFS.D1FIFOSEL 4008 002Ch

	b15	b14	13b	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RCNT	REW	DCLRM	DREQE	—	MBW	—	BIGEN D	—	—	—	—	CURPIPE[3:0]			
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b3 to b0	CURPIPE[3:0]	CFIFO port access pipe specification	b3 b0 0 0 0 0: No pipe specification 0 0 0 1: Pipe: 1 0 0 1 0: Pipe: 2 0 0 1 1: Pipe: 3 0 1 0 0: Pipe: 4 0 1 0 1: Pipe: 5 0 1 1 0: Pipe: 6 0 1 1 1: Pipe: 7 1 0 0 0: Pipe: 8 1 0 0 1: Pipe: 9 Settings other than above are prohibited.	R/W
b7-4	—	Reserved	This bit is read as 0. The write value should be 0.	R
b8	BIGEND	FIFO port endian control	0: Little endian 1: Big endian	R/W
b9	—	Reserved	This bit is read as 0. The write value should be 0.	R
b10	MBW	FIFO port access bit width	0: 8-bit access 1: 16-bit access	R/W
b11	—	Reserved	This bit is read as 0. The write value should be 0.	R
b12	DREQE	DMA transfer request	0: DMA transfer request is disabled. 1: DMA transfer request is enabled.	R/W
b13	DCLRM	Auto buffer memory clear Mode accessed after specified pipe data is read	0: Auto buffer clear mode is disabled. 1: Auto buffer clear mode is enabled.	R/W
b14	REW	Buffer pointer rewind	0: The buffer pointer is not rewound. 1: The buffer pointer is rewound.	R/W*1
b15	RCNT	Read count mode	0: The DTLN[8:0] bit (CFIFOCTR.DTLN[8:0], D0FIFOCTR.DTLN[8:0], D1FIFOCTR.DTLN[8:0]) are cleared when all of the receive data has been read from the CFIFO. In double buffer mode, the DTLN[8: 0] bit value is cleared when all the data has been read from only a single plane.) 1: The DTLN[8:0] flags are decremented each time the receive data is read from the CFIFO.	R/W

*1. Only 0 can be read.

The same pipe should not be specified by the CURPIPE[3:0] bits in the CFIFOSEL, D0FIFOSEL, and D1FIFOSEL registers. When the CURPIPE[3:0] bits in the D0FIFOSEL and D1FIFOSEL registers are set to 0000b, no pipe is selected. The pipe number should not be changed while the DMA or the DTC transfer is enabled.

CURPIPE[3:0]:

The CURPIPE[3:0] bits specify the pipe number using which data is read or written through the D0FIFO port or D1FIFO port. After writing to these bits, read these bits to check that the written value agrees with the read value before proceeding to the next process.Do not set the same pipe number to the CURPIPE[3:0] bits in the CFIFOSEL, D0FIFOSEL, and D1FIFOSEL registers.

Even if an attempt is made to modify the setting of these bits during access to the FIFO buffer, the current access setting is retained until the access is completed. Then, the modification becomes effective, thus enabling continuous access.

MBW:

The MBW bit specifies the bit width for accessing the D0FIFO port or D1FIFO port.

When the selected pipe is receiving, once reading data is started after setting this bit, this bit should not be modified until all the data has been read. When reading the FIFO buffer, use the access size set in the MBW bit for the read. Set both the CURPIPE[3:0] bit and the MBW bit.

When the selected pipe is transmitting, the bit width cannot be changed from 8-bit width to 16-bit width while data is being written to the FIFO memory.

An odd number of bytes can also be written through byte-access control even when 16-bit width is selected.

DREQE:

The DREQE bit enables or disables the DMA transfer request to be issued. To enable DMA transfer requests, set this bit to 1 after setting the CURPIPE[3:0] bit to 1. When modifying the setting of the CURPIPE[3:0] bits, set this bit to 0 first.

DCLRM bit (auto buffer memory clear mode accessed after specified pipe data is read)

The DCLRM bit enables or disables the buffer memory to be cleared automatically after data has been read using the selected pipe.

With this bit set to 1, the USBFS sets the BCLR bit to 1 for the FIFO buffer of the selected pipe on receiving a zero-length packet while the FIFO buffer assigned to the selected pipe is empty, or on receiving a short packet and reading the data while the PIPECFG.BFRE bit is 1.

When using the USBFS with the SOFCFG.BRDYM bit set to 1, set this bit to 0.

REW:

The REW bit specifies whether or not to rewind the buffer pointer.

When the selected pipe is receiving, setting the REW bit to 1 while the FIFO buffer is being read allows re-reading the FIFO buffer from the first data. In double buffer mode, re-reading the currently-read FIFO buffer plane from the first data is allowed.

Do not set the REW bit to 1 simultaneously with modifying the CURPIPE[3:0] bits. Before setting the REW bit to 1, be sure to check that the FRDY bit is 1.

To re-write to the FIFO buffer again from the first data for the pipe in transmitting, use the BCLR bit.

RCNT:

The RCNT bit specifies the read mode for the value in the CFIFOCTR.DTLN bit. When accessing DnFIFO with the PIPECFG.BFRE bit set to 1, set the RCNT bit to 0.

22.2.7 CFIFO port control register (CFIFOCTR) D0FIFO port control register (D0FIFOCTR) D1FIFO port control register (D1FIFOCTR)

Address: USBFS.CFIFOCTR 4008 0022h, USBFS.D0FIFOCTR 4008 002Ah, USBFS.D1FIFOCTR 4008 002Eh

	b15	b14	13b	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	BVAL	BCLR	FRDY	—	—	—	—	DTLN[8:0]								
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b8~b0	DTLN[8:0]	Receive data length	Indicate the length of the receive data. These bits indicate different values depending on the setting of the RCNT bit in the port select register. For details, refer to the description on the DTLN[8:0] bit shown below.	R
b12~b9	—	Reserved	This bit is read as 0. The write value should be 0.	R/W
b13	FRDY	FIFO port ready	0: FIFO port access is disabled. 1: FIFO port access is enabled.	R
b14	BCLR	CPU buffer clear	0: No operation 1: Clear the buffer memory on the CPU.	R/W*1
b15	BVAL	Buffer memory valid	0: Invalid 1: Writing ended	R/W

*1. Only 0 can be read.

Registers CFIFOCTR, D0FIFOCTR, and D1FIFOCTR correspond to CFIFO, D0FIFO, and D1FIFO, respectively.

DTLN[8:0]:

DTLN[8: 0]bit indicates the length of the receive data.

While the FIFO buffer is being read, the DTLN[8:0] flags indicate different values depending on the DnFIFOSEL.RCNT bit (n = 0, 1) value as described below:

- RCNT=0

  The USBFS sets the DTLN[8:0] flags to 1 to indicate the length of the receive data until the CPU or DMA/DTC has read all the received data from a single FIFO buffer plane.

  While the PIPECFG.BFRE bit = 1, even after all data has been read, the USB will retain the length of the received data until the BCLR bit is set to 1.

- RCNT=1

  The USBFS decrements the value indicated by the DTLN[8:0] flags each time data is read from the FIFO buffer. The value is decremented by one when the MBW bit is 0, and by two when the MBW bit is 1.

  The USBFS sets these bits to 0 when all the data has been read from one FIFO buffer. However, in double buffer mode, if data has been received in one FIFO buffer plane before all the data has been read from the other plane, the USB sets these bits to indicate the length of the receive data in the former plane when all the data has been read from the latter plane.

FRDY:

The FRDY flag indicates whether the FIFO port can be accessed by the CPU or DMA.

In the following cases, the USBFS sets the FRDY flag to 1 but data cannot be read via the FIFO port because there is no data to be read. In these cases, set the BCLR bit to 1 to clear the FIFO buffer, and enable transmission and reception of the next data.

- A zero-length packet is received when the FIFO buffer assigned to the selected pipe is empty.
- A short packet is received and the data is completely read while the PIPECFG.BFRE bit = 1.

BCLR:

The BCLR bit should be set to 1 to clear the FIFO buffer on the CPU side for the selected pipe.

When double buffer mode is set for the FIFO buffer assigned to the selected pipe, the USBFS clears only one plane of the FIFO buffer even when both planes are read-enabled.

When the selected pipe is the DCP, setting the BCLR bit to 1 allows the USBFS to clear the FIFO buffer regardless of whether the CPU or SIE has access rights. To clear the buffer when the SIE has access, please set the DCPCTR.PID[1:0] bit to 00b before setting the BCLR bit to 1 (NAK ACK).

When the selected pipe is transmitting, if 1 is written to the BVAL bit and the BCLR bit simultaneously, the USBFS clears the data that has been written before it, enabling transmission of a zero-length packet.

When the selected pipe is not the DCP, writing 1 to the BCLR bit while the FRDY flag in the FIFO port control register is 1 (set by the USBFS).

BVAL:

The BVAL bit should be set to 1 when data has been completely written to the FIFO buffer on the CPU for the pipe selected using the CURPIPE[3:0] bits (selected pipe).

When the selected pipe is transmitting, set the BVAL bit to 1 in the following cases:

- To transmit a short packet, set the BVAL bit to 1 after data has been written.
- To transmit a zero-length packet, set the BVAL bit to 1 before data is written to the FIFO buffer.

Then, the USBFS switches the FIFO buffer from the CPU side to the SIE side, thus enabling transmission.

When data of the maximum packet size has been written for the pipe in continuous transfer mode, the USBFS sets the BVAL bit to 1 and switches the FIFO buffer from the CPU side to the SIE side, thus enabling transmission.

22.2.8 Interrupt enable register 0 (INTENB0)

Address: USBFS.INTENB0 4008 0030h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	VBSE	RSME	SOFE	DVSE	CTRE	BEMPE	NRDYE	BRDYE	—	—	—	—	—	—	—	—
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b7~b0	—	Reserved	These bits are read as 0, and the write value should be 0.	—
b8	BRDYE	Buffer ready interrupt enable	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b9	NRDYE	Buffer not ready response interrupt enable	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b10	BEMPE	Buffer empty interrupt enable	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b11	CTRE	Control transfer stage transition interrupt enable*1	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b12	DVSE	Device state transition interrupt enable*1	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b13	SOFE	Frame number update interrupt enable	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b14	RSME	Resume interrupt enable*1	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b15	VBSE	VBUS interrupt enable	0: Interrupt output disabled 1: Interrupt output enabled	R/W

*1. The RSME, DVSE and CTRE bits can only be set to 1 in device controller mode; do not set these bits to 1 in the host controller

When the status flag in the INTSTS0 register is set to 1 and the corresponding interrupt request enable bit in the INTENB0 register is set to 1, the USBFS issues a USBFS interrupt request.

Regardless of the INTENB0 register setting, the status flag in the INTSTS0 register will be set to 1 in response to a status change that satisfies the relevant condition.

When the relevant status flag in the INTSTS0 register is set to 1, the USBFS interrupt will be requested when the interrupt request enable bit in the INTENB0 register is modified from 0 to 1.

22.2.9 Interrupt enable register 1 (INTENB1)

Address: USBFS.INTENB1 4008 0032h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	OVRCRE	BCHGE	—	DTCHE	ATTCH E	—	—	—	—	EOFER RE	SIGNE	SACKE	—	—	—	PDDET INTE0
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	PDDETINTE0	PDDETINT0 detection interrupt enable	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b3~b1	—	Reserved	These bits are read as 0, and the write value should be 0.	—
b4	SACKE	Setup transaction normal response interrupt enable	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b5	SIGNE	Setup transaction error interrupt enable	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b6	EOFERRE	EOF error detection interrupt enable	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b10~b7	—	Reserved	These bits are read as 0, and the write value should be 0.	—
b11	ATTCHE	Connection detection interrupt enable	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b12	DTCHE	Disconnection detection interrupt enable	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b13	—	Reserved	These bits are read as 0, and the write value should be 0.	—
b14	BCHGE	USB bus change interrupt enable	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b15	OVRCRE	Overcurrent input change interrupt enable	0: Interrupt output disabled 1: Interrupt output enabled	R/W

Note: The bits in the INTENB1 register can be set to 1 only when the host controller is selected; do not set these bits to 1 to enable the corresponding interrupt output when the device controller is selected.

The INTENB1 register specifies the interrupt masks when the host controller is selected, and for the setup transaction.

When the status flag in the INTSTS1 register is set to 1 and the corresponding interrupt request enable bit in the INTENB1 register is set to 1, the USBFS issues a USBFS interrupt request.

Regardless of the INTENB1 register setting, the status flag in the INTSTS1 register will be set to 1 in response to a status change that satisfies the relevant condition.

When the relevant status flag in the INTSTS1 register is set to 1, the USBFS interrupt will be requested when the interrupt request enable bit in the INTENB1 register is modified from 0 to 1.

Do not enable interrupts in device controller mode.

22.2.10 BRDY interrupt enable register (BRDYENB)

Address: USBFS.BRDYENB 4008 0036h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	PIPE9BRDYE	PIPE8BRDYE	PIPE7BRDYE	PIPE6BRDYE	PIPE5BRDYE	PIPE4BRDYE	PIPE3BRDYE	PIPE2BRDYE	PIPE1BRDYE	PIPE0BRDYE
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	PIPE0BRDYE	BRDY interrupt enable for pipe 0	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b1	PIPE1BRDYE	BRDY interrupt enable for pipe 1	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b2	PIPE2BRDYE	BRDY interrupt enable for pipe 2	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b3	PIPE3BRDYE	BRDY interrupt enable for pipe 3	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b4	PIPE4BRDYE	BRDY interrupt enable for pipe 4	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b5	PIPE5BRDYE	BRDY interrupt enable for pipe 5	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b6	PIPE6BRDYE	BRDY interrupt enable for pipe 6	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b7	PIPE7BRDYE	BRDY interrupt enable for pipe 7	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b8	PIPE8BRDYE	BRDY interrupt enable for pipe 8	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b9	PIPE9BRDYE	BRDY interrupt enable for pipe 9	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b15~b10	—	Reserved	These bits are read as 0, and the write value should be 0.	—

The BRDYENB register enables or disables the INTSTS0.BRDY flag to be set to 1 when the BRDY interrupt is detected for each pipe.

When the status flag in the BRDYSTS register is set to 1 and the corresponding PIPEnBRDYE bits (n=0 to 9) in the BRDYENB register are set to 1, the INTSTS0.BRDY flag is set to 1. In this case, the USBFS generates a BRDY interrupt request if the BRDYE bit in INTENB0 is 1.

While at least one PIPEnBRDY flag indicates 1, the USBFS generates the BRDY interrupt request when the corresponding interrupt enable bit in the BRDYENB register is modified from 0 to 1 by software.

22.2.11 NRDY interrupt enable register (NRDYENB)

Address: USBFS.NRDYENB 4008 0038h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	PIPE9N RDYE	PIPE8N RDYE	PIPE7N RDYE	PIPE6N RDYE	PIPE5N RDYE	PIPE4N RDYE	PIPE3N RDYE	PIPE2N RDYE	PIPE1N RDYE	PIPE0N RDYE
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	PIPE0NRDYE	NRDY interrupt enable for pipe 0	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b1	PIPE1NRDYE	NRDY interrupt enable for pipe 1	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b2	PIPE2NRDYE	NRDY interrupt enable for pipe 2	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b3	PIPE3NRDYE	NRDY interrupt enable for pipe 3	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b4	PIPE4NRDYE	NRDY interrupt enable for pipe 4	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b5	PIPE5NRDYE	NRDY interrupt enable for pipe 5	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b6	PIPE6NRDYE	NRDY interrupt enable for pipe 6	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b7	PIPE7NRDYE	NRDY interrupt enable for pipe 7	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b8	PIPE8NRDYE	NRDY interrupt enable for pipe 8	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b9	PIPE9NRDYE	NRDY interrupt enable for pipe 9	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b15~b10	—	Reserved	These bits are read as 0, and the write value should be 0.	R/W

The NRDYENB register enables or disables the INTSTS0.NRDY flag to be set to 1 when the NRDY interrupt is detected for each pipe.

The INTSTS0.NRDY flag is set to 1 when the status flag in the NRDYSTS register is set to 1 and the corresponding PIPEnNRDYE (n=0 to 9) bits in the NRDYENB register are set to 1. In this case, the USBFS generates an NRDY interrupt request if the NRDYE bit in INTENB0 is 1.

While at least one PIPEnNRDY flag indicates 1, the USBFS generates the NRDY interrupt request when the corresponding interrupt enable bit in the NRDYENB register is modified from 0 to 1 by software.

22.2.12 BEMP interrupt enable register (BEMPENB)

Address: USBFS.BEMPENB 4008 003Ah

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	PIPE9B EMPE	PIPE8B EMPE	PIPE7B EMPE	PIPE6B EMPE	PIPE5B EMPE	PIPE4B EMPE	PIPE3B EMPE	PIPE2B EMPE	PIPE1B EMPE	PIPE0B EMPE
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	PIPE0BEMPE	BEMP interrupt enable for pipe 0	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b1	PIPE1BEMPE	BEMP interrupt enable for pipe 1	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b2	PIPE2BEMPE	BEMP interrupt enable for pipe 2	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b3	PIPE3BEMPE	BEMP interrupt enable for pipe 3	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b4	PIPE4BEMPE	BEMP interrupt enable for pipe 4	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b5	PIPE5BEMPE	BEMP interrupt enable for pipe 5	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b6	PIPE6BEMPE	BEMP interrupt enable for pipe 6	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b7	PIPE7BEMPE	BEMP interrupt enable for pipe 7	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b8	PIPE8BEMPE	BEMP interrupt enable for pipe 8	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b9	PIPE9BEMPE	BEMP interrupt enable for pipe 9	0: Interrupt output disabled 1: Interrupt output enabled	R/W
b15~b10	—	Reserved	These bits are read as 0, and the write value should be 0.	R/W

The BEMPENB register enables or disables the INTSTS0.BEMP flag to be set to 1 when the BEMP interrupt is detected for each pipe.

The INTSTS0.BEMP flag is set to 1 when the status flag in the BEMPSTS register is set to 1 and the corresponding PIPEnBEMPE (n=0 to 9) bits in the BEMPENB register are set to 1. In this case, the USBFS generates a BEMP interrupt request if the BEMPE bit in INTENB0 is 1.

While at least one PIPEnBEMP flag in the BEMPSTS register indicates 1, the USBFS generates the BEMP interrupt request when the corresponding interrupt enable bit in the BEMPENB register is modified from 0 to 1 by software.

22.2.13 SOF output configuration register (SOFCFG)

Address: USBFS.SOFCFG 4008 003Ch

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	TRNEN SEL	—	BRDY M	—	EDGES TS	—	—	—	—
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b3~b0	—	Reserved	These bits are read as 0, and the write value should be 0.	R/W
b4	EDGESTS	Edge interrupt output status monitor flag*1	Indicates 1 when the edge interrupt output signal is in the middle of the edge processing.	R
b5	—	Reserved	These bits are read as 0, and the write value should be 0.	R/W
b6	BRDYM	BRDY interrupt status clear timing	0: Software clears the status. 1: The USBFS clears the status when data has been read from the FIFO buffer or data has been written to the FIFO buffer.	R/W
b7	—	Reserved	The bit is read as 0, and the write value should be 0.	R/W
b8	TRNENSEL	Transaction-enabled time select*1	0: Non-low-speed communication 1: Low-speed communication	R/W
b15~b9	—	Reserved	These bits are read as 0, and the write value should be 0.	R/W

*1. Confirm that this bit is 0 before stopping the clock supply to the USBFS.

EDGESTS:

The EDGESTS flag indicates 1 when the edge interrupt output signal is in the middle of the edge processing. Confirm that this flag is 0 before stopping the clock supply to the USBFS.

BRDYM:

The BRDYM bit specifies the timing for clearing the BRDY interrupt status for each pipe.

TRNENSEL:

When the USB port is used for full or low speed communication, the TRNENSEL bit specifies the time at which the USBFS publishes the token in the frame (the time at which the transaction is enabled).

Set the TRNENSEL bit to 1 when a low-speed device is connected. The TRNENSEL bit is valid only when the host controller is selected. This bit should be set to 0 if the device controller is selected.

22.2.14 Interrupt status register0 (INTSTS0)

Address: USBFS.INTSTS0 4008 0040h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	VBINT	RESM	SOFR	DVST	CTRT	BEMP	NRDY	BRDY	VBSTS	DVSQ[2:0]			VALID	CTSQ[2:0]		
Reset value	0	0	0	0	0	0	0	0	0*2	0*3	0*3	0/1*3	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b2~b0	CTSQ[2:0]	Control transfer stage	b2 b0 0 0 0: Idle or setup stage 0 0 1: Control read data stage 0 1 0: Control read status stage 0 1 1: Control write data stage 1 0 0: Control write status stage 1 0 1: Control write (no data) status stage 1 1 0: Control transfer sequence error	R
b3	VALID	USB equest reception	0: Setup packet is not received. 1: Setup packet is received.	R/W*4
b6~b4	DVSQ[2:0]	Device state	b6 b4 0 0 0: Powered state 0 0 1: Default state 0 1 0: Address state 0 1 1: Configured state 1 x x: Suspended state	R
b7	VBSTS	VBUS input status	0: USB_VBUS port is low. 1: USB_VBUS port is high	R
b8	BRDY	Buffer ready interrupt status flag	0: BRDY interrupts are not generated. 1: BRDY interrupts are generated.	R
b9	NRDY	Buffer not ready interrupt status flag	0: NRDY interrupts are not generated. 1: NRDY interrupts are generated.	R
b10	BEMP	Buffer empty interrupt status flag	0: BEMP interrupts are not generated. 1: BEMP interrupts are generated.	R
b11	CTRT	Control transfer stage transition interrupt status flag *5	0: Control transfer stage transition interrupts are not generated. 1: Control transfer stage transition interrupts are generated.	R/W*4
b12	DVST	Device state transition interrupt status flag *5	0: Device state transition interrupts are not generated. 1: Device state transition interrupts are generated.	R/W*4
b13	SOFR	Frame number refresh interrupt status flag	0: SOF interrupts are not generated. 1: SOF interrupts are generated.	R/W*4
b14	RESM	Resume interrupt status flag *5,*6	0: Resume interrupts are not generated. 1: Resume interrupts are generated.	R/W*4
b15	VBINT	VBUS interrupt status flag *6	0: VBUS interrupts are not generated. 1: VBUS interrupts are generated.	R/W*4

x: Don’t care

*1. The value is 0 when the MCU is reset and 1 after a USB bus reset.

*2. The value is 1 when the USB_VBUS pin is high and 0 when the USB_VBUS pin is low.

*3. The value is 000b when the MCU is reset and 001b after a USB bus reset.

*4. To clear the VBINT, RESM, SOFR, DVST, CTRT, or VALID flag, write 0 only to the bits to be cleared; write 1 to the other bits. Do not write 0 to the status bits indicating 0.

*5. The status of the RESM, DVST, and CTRT flags are changed only when the device controller is selected. Set the corresponding interrupt enable bits to 0 (disabled) when the host controller is selected.

*6. A change in the status indicated by the VBINT and RESM bits can be detected by the USBFS even while the clock supply is stopped (the SCKE bit = 0), and the USBFS will request an interrupt when the corresponding interrupt request bit is 1. The clock is supplied before the state is cleared by software.

CTSQ[2:0]:

In host controller mode, the read value of the CTSQ[2:0] bits is invalid.

VALID:

In host controller mode, the read value of the VALID bit is invalid.

DVSQ[2:0]:

The DVSQ[2:0] bits are initialized by a USB bus reset. In host controller mode, the value read is invalid.

BRDY:

The BRDY bit indicates the BRDY interrupt status.

When the BRDY interrupt state (PIPEnBRDY=1, n=0 to 9) is detected on at least one pipe with BRDY interrupts enabled (BRDYENB.PIPEnBRDYE=1), the USBFS sets BRDY bit to 1.

For the conditions for PIPEnBRDY status assertion, refer to 22.3.3.1 BRDY interrupt.

The USBFS sets the BRDY flag to 0 when 0 is written by software to all the PIPEnBRDY flags corresponding to the PIPEnBRDYE bits that have been set to 1. The BRDY flag cannot be set to 0 even if 0 is written to this bit by software.

NRDY:

The USBFS sets the NRDY bit to 1 when at least one of the PIPENRDY bits (n=0 to 9) associated with the PIPEnNRDYE bit (n=0 to 9) is set to 1. (USBFS detects the NRDY interrupt status of at least one pipe in a pipes whose software is set to allow NRDY interrupt output)

For the conditions for PIPEnNRDY status assertion, refer to 22.3.3.2 NRDY interrupt.

The USBFS sets the NRDY flag to 0 when 0 is written by software to all the PIPEnNRDY bits corresponding to the PIPEnNRDYE bits that have been set to 1. The NRDY flag cannot be set to 0 even if 0 is written to this bit by software.

BEMP:

The BEMP bit indicates the BEMP interrupt status.

The USBFS sets the BEMP bit to 1 when the BEMP interrupt status (PIPEnBEMP=1, n=0 to 9) is detected on at least one pipe with BEMP interrupts enabled (BEMPENB.PIPEnBEMPE=1).

For the conditions for PIPEnBEMP status assertion, refer to 22.3.3.3 BEMP interrupt.

The USBFS sets the BEMP to 0 when 0 is written by software to all the PIPEnBEMP bitss corresponding to the PIPEnBEMPE bits that have been set to 1.The BEMP flag cannot be set to 0 even if 0 is written to this bit by software.

CTRT:

When the device controller is selected, the USBFS updates the value of the CTSQ[2:0] flags and sets the CTRT flag to 1 on detecting a change in the control transfer stage. When a control transfer stage transition interrupt is generated, clear the status before the USBFS detects the next control transfer stage transition.

When the host controller is selected, the read value is invalid.

DVST:

When the device controller is selected, the USBFS updates the DVSQ[2:0] value and sets the DVST flag to 1 on detecting a change in the device state.

When a device state transition interrupt is generated, clear the status before the USBFS detects the next device state transition.

When the host controller is selected, the read value is invalid.

SOFR:

In host controller mode, the USBFS sets the SOFR bit to 1 when the frame number is updated (when the DVSTCTR0.UACT bit is set to 1 by software). a SOFR interrupt is detected every 1ms.

Even if a damaged SOF packet is received from the USB host, the USBFS can detect SOFR interruptions through the internal interpolation function.

RESM:

When the device controller is selected, the USBFS sets the RESM flag to 1 on detecting the falling edge of the signal on the USB_DP pin in the suspended state (DVSQ[2:0] = 1xxb).

When the host controller is selected, the read value is invalid.

VBINT:

The USBFS sets the VBINT flag to 1 on detecting a level change (high to low or low to high) in the USB_VBUS pin input value. The USBFS sets the VBSTS flag to indicate the USB_VBUS pin input value. When the VBUS interrupt is generated, use software to repeat reading the VBSTS flag until the same value is read three or more times, and eliminate chattering.

22.2.15 Interrupt status flag1 (INTSTS1)

Address: USBFS.INTSTS1 4008 0042h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	OVRC R	BCHG	—	DTCH	ATTCH	—	—	—	—	EOFER R	SIGN	SACK	—	—	—	PDDET INT0
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	PDDETINT0	Portable device detection interruption status	0: PDDET0 detection interrupts are not generated. 1: PDDET0 detection interrupts are generated.	R/W *1
b3~b1	—	Reserved	These bits are read as 0. The write value should be 0.	—
b4	SACK	setup transaction normal response interrupt status	0: SACK interrupts are not generated. 1: SACK interrupts are generated.	R/W *1
b5	SIGN	Setup transaction error interrupt status	0: SIGN interrupts are not generated. 1: SIGN interrupts are generated.	R/W *1
b6	EOFERR	EOF error detection interrupt status	0: EOFERR interrupts are not generated. 1: EOFERR interrupts are generated.	R/W *1
b10~b7	—	Reserved	These bits are read as 0. The write value should be 0.	—
b11	ATTCH	Connection detection interrupt status	0: ATTCH interrupts are not generated. 1: ATTCH interrupts are generated.	R/W *1
b12	DTCH	USB disconnection detection interrupt status	0: DTCH interrupts are not generated. 1: DTCH interrupts are generated.	R/W *1
b13	—	Reserved	These bits are read as 0. The write value should be 0.	—
b14	BCHG	USB bus change interrupt status*2	0: BCHG interrupts are not generated. 1: BCHG interrupts are generated.	R/W *1
b15	OVRCR	Overcurrent input change interrupt status *2	0: OVRCR interrupts are not generated. 1: OVRCR interrupts are generated.	R/W *1

*1. To clear the status indicated by the bit in the INTSTS1 register, write 0 only to the bits to be cleared; write 1 to the other bits.

*2. Even if clock supply is stopped (SYSCFG.SCKE=0), the USBFS will still detect a state change in the OVRCR or BCHG bit and will request an interrupt when the corresponding interrupt request bit is 1. The clock is supplied (SYSCFG.SCKE=1) before this state is cleared by software. When clock supply is stopped (SYSCFG.SCKE bit = 0), no other interrupts can be detected.

The INTSTS1 register is used to confirm the status of each interrupt when the host controller is selected.

PDDETINTE0:

Indicates the status of the portable device detection interrupt when the host controller is selected. This bit is set to 1 when the USBFS detects when a level change (high to low or low to high) occurs in the input value to the VDPDET pin of the USB physical layer transceiver (PHY). The USBFS sets the PDDETSTS0 flag to indicate the VDPDET input value. When the PDDETINT interrupt is generated, use software to repeat reading the PDDETSTS0 bit until the same value is read three or more times, and eliminate chattering.

SACK:

Indicates the status of the setup transaction normal response interrupt when the host controller is selected.

The USBFS detects the SACK interrupt when ACK response is returned from the peripheral device during the setup transactions issued by the USBFS, and sets the SACK flag to 1. Here, if the corresponding interrupt enable bit has been set to 1 by software, the USBFS generates the SACK interrupt.

When the device controller is selected, the read value is invalid.

SIGN:

Indicates the status of the setup transaction error interrupt when the host controller is selected.

The USBFS detects the SIGN interrupt when ACK response is not returned from the peripheral device three consecutive times during the setup transactions issued by this module, and sets the SIGN flag to 1. Here, if the corresponding interrupt enable bit has been set to 1 by software, the USBFS generates an interrupt.

Specifically, the USBFS detects the SIGN interrupt when any of the following response conditions occur for three consecutive setup transactions.

- Timeout is detected by the USBFS when the peripheral device has returned no response.
- A damaged ACK packet is received.
- A handshake other than ACK (NAK, NYET, or STALL) is received.

EOFERR:

Indicates the status of the EOF error detection interrupt when the host controller is selected.

The USBFS detects the EOFERR interrupt on detecting that communication is not completed at the EOF2 timing prescribed by USB Specification 2.0, and sets the EOFERR flag to 1. Here, if the corresponding interrupt enable bit has been set to 1 by software, the USBFS generates an interrupt.

Upon detection of an EOFERR interrupt, the USBFS will control the hardware as follows, regardless of the corresponding interrupt enable bit setting:

- Modifies the DVSTCTR0.UACT bit for the port in which an EOFERR interrupt has been detected to 0.
- Puts the port in which an EOFERR interrupt has been generated into the idle state.

The software must terminate all pipes that are currently communicating and re-enumerate the USB ports.

When the device controller is selected, the read value is invalid.

ATTCH:

indicates the status of the USB connection detection interrupt when the host controller is selected.

The USBFS detects an ATTCH interrupt and sets this bit to 1 when the J- or K-state of the full or low signal level is detected within 2.5µs. If the corresponding interrupt enable bit is set to 1 by software, the USBFS generates an interrupt.

Specifically, the USBFS detects the ATTCH interrupt on any of the following conditions:

- K-state, SE0, or SE1 changes to J-state, and J-state continues for 2.5 µs.
- J-state, SE0, or SE1 changes to K-state, and K-state continues for 2.5 µs.

When the device controller is selected, the read value is invalid.

DTCH:

Indicates the status of the USB disconnection detection interrupt when the host controller is selected.

The USBFS detects the DTCH interrupt on detecting USB bus disconnection, and sets the DTCH flag to 1. Here, if the corresponding interrupt enable bit has been set to 1 by software, the USBFS generates an interrupt. The USBFS detects bus disconnection based on USB Specification 2.0.

When a DTCH interrupt is detected, the USBFS controls the hardware as follows, regardless of the corresponding interrupt enable bit setting:

- Modifies the DVSTCTR0.UACT bit for the port in which a DTCH interrupt has been detected to 0.
- Puts the port in which a DTCH interrupt has been generated into the idle state.

The software must terminate all pipes that are currently performing communication and invoke a wait state to connect to the USB port (waiting for the ATTCH interrupt to be generated).

When the device controller is selected, the read value is invalid.

BCHG:

Indicates the status of the USB bus change interrupt when the host controller is selected.

The USBFS detects the BCHG interrupt when a change in the full-speed or low-speed signal level occurs on the USB port (a change from J-state, K-state, or SE0 to J-state, K-state, or SE0), and sets the BCHG flag to 1. Here, if the corresponding interrupt enable bit has been set to 1 by software, the USBFS generates an interrupt.

The USBFS sets the LNST[1:0] bit to indicate the current input state of the USB port. When the BCHG interrupt is generated, use software to repeat reading the LNST[1:0] bit until the same value is read three or more times, and eliminate chattering.

A change in the USB bus state can be detected even while the internal clock supply is stopped.

When the device controller is selected, the read value is invalid.

OVRCR:

Indicates the status of the USB_OVRCURA and USB_OVRCURB input pin change interrupt.

The USBFS detects the OVRCR interrupt when a change (high to low or low to high) occurs in at least one of the input values to the USB_OVRCURA and USB_OVRCURB pins, and sets the OVRCR flag to 1. Here, if the corresponding interrupt enable bit has been set to 1 by software, the USBFS generates the interrupt.

22.2.16 BRDY interrupt status register (BRDYSTS)

Address: USBFS.BRDYSTS 4008 0046h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	PIPE9B RDY	PIPE8B RDY	PIPE7B RDY	PIPE6B RDY	PIPE5B RDY	PIPE4B RDY	PIPE3B RDY	PIPE2B RDY	PIPE1B RDY	PIPE0B RDY
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	PIPE0BRDY	BRDY interrupt status flag for pipe 0*2	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b1	PIPE1BRDY	BRDY interrupt status flag for pipe 1*2	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b2	PIPE2BRDY	BRDY interrupt status flag for pipe 2*2	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b3	PIPE3BRDY	BRDY interrupt status flag for pipe 3*2	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b4	PIPE4BRDY	BRDY interrupt status flag for pipe 4*2	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b5	PIPE5BRDY	BRDY interrupt status flag for pipe 5*2	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b6	PIPE6BRDY	BRDY interrupt status flag for pipe 6*2	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b7	PIPE7BRDY	BRDY interrupt status flag for pipe 7*2	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b8	PIPE8BRDY	BRDY interrupt status flag for pipe 8*2	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b9	PIPE9BRDY	BRDY interrupt status flag for pipe 9*2	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b15~b10	—	Reserved	These bits are read as 0. The write value should be 0.	R/W

*1. When the SOFCFG.BRDYM bit is set to 0, to clear the status indicated by the bit in the BRDYSTS register, write 0 only to the bits to be cleared; write 1 to the other bits.

*2. When the SOFCFG.BRDYM bit is set to 0, clearing BRDY Interrupts before accessing the FIFO.

22.2.17 NRDY interrupt status register (NRDYSTS)

Address: USBFS.NRDYSTS 4008 0048h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	PIPE9N RDY	PIPE8N RDY	PIPE7N RDY	PIPE6N RDY	PIPE5N RDY	PIPE4N RDY	PIPE3N RDY	PIPE2N RDY	PIPE1N RDY	PIPE0N RDY
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	PIPE0NRDY	NRDY interrupt status flag for pipe 0	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b1	PIPE1NRDY	NRDY interrupt status flag for pipe 1	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b2	PIPE2NRDY	NRDY interrupt status flag for pipe 2	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b3	PIPE3NRDY	NRDY interrupt status flag for pipe 3	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b4	PIPE4NRDY	NRDY interrupt status flag for pipe 4	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b5	PIPE5NRDY	NRDY interrupt status flag for pipe 5	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b6	PIPE6NRDY	NRDY interrupt status flag for pipe 6	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b7	PIPE7NRDY	NRDY interrupt status flag for pipe 7	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b8	PIPE8NRDY	NRDY interrupt status flag for pipe 8	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b9	PIPE9NRDY	NRDY interrupt status flag for pipe 9	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b15~b10	—	Reserved	These bits are read as 0. The write value should be 0.	R/W

*1. To clear the status indicated by the bits in the NRDYSTS register, write 0 only to the bits to be cleared; write 1 to the other bits.

22.2.18 BEMP interrupt status register (BEMPSTS)

Address: USBFS.BEMPSTS 4008 004Ah

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	PIPE9B EMP	PIPE8B EMP	PIPE7B EMP	PIPE6B EMP	PIPE5B EMP	PIPE4B EMP	PIPE3B EMP	PIPE2B EMP	PIPE1B EMP	PIPE0B EMP
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	PIPE0BEMP	BEMP interrupt status flag for pipe 0	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b1	PIPE1BEMP	BEMP interrupt status flag for pipe 1	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b2	PIPE2BEMP	BEMP interrupt status flag for pipe 2	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b3	PIPE3BEMP	BEMP interrupt status flag for pipe 3	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b4	PIPE4BEMP	BEMP interrupt status flag for pipe 4	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b5	PIPE5BEMP	BEMP interrupt status flag for pipe 5	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b6	PIPE6BEMP	BEMP interrupt status flag for pipe 6	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b7	PIPE7BEMP	BEMP interrupt status flag for pipe 7	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b8	PIPE8BEMP	BEMP interrupt status flag for pipe 8	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b9	PIPE9BEMP	BEMP interrupt status flag for pipe 9	0: Interrupts are not generated. 1: Interrupts are generated.	R/W *1
b15~b10	—	Reserved	These bits are read as 0. The write value should be 0.	R/W

*1. To clear the status indicated by the bit in the BEMPSTS register, write 0 only to the bits to be cleared; write 1 to the other bits.

22.2.19 Frame number register (FRMNUM)

Address: USBFS.FRMNUM 4008 004Ch

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	OVRN	CRCE	—	—	—	FRNM[10:0]										
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b10~b0	FRNM[10:0]	Frame number	Latest frame number	R
b13~b11	—	Reserved	These bits are read as 0. The write value should be 0.	R/W
b14	CRCE	Receive data error	0: No error 1: An error occurred	R/W*1
b15	OVRN	Overrun/underrun detection status	0: No error 1: An error occurred	R/W*1

*1. To clear the status, write 0 only to the bits to be cleared; write 1 to the other bits.

FRNM[10:0]:

The FRNM[10:0] bits indicate the latest frame number for the USBFS after the issuing of an SOF packet every 1 ms or writing to the FRNM[10:0] flags at the SOF packet reception.

CRCE:

The CRCE bit is set to 1 when a CRC error or a bit stuffing error occurs during isochronous transfer. The USBFS generates an internal NRDY interrupt when a CRC error is detected in host controller mode.

To clear the CRCE bit, write 0 to this bit of the FRMNUM register and 1 to the other bits.

OVRN:

When an overrun or underrun error occurs during isochronous transmission, the OVRN bit is set to 1. To clear the OVRN bit, write 0 to this bit and 1 to the other bits of the FRMNUM register.

In host controller mode, the OVRN bit is set to 1 in any of the following cases:

- For isochronous transmitting pipe, an OUT token is issued before all transferred data is written to the FIFO buffer
- For isochronous receiving pipe, an IN token is issued when there is no empty FIFO buffer plane

In device controller mode, the OVRN bit is set to 1 in any of the following cases:

- For isochronous transmitting pipe, the IN token is received before all transfer data is written to the FIFO buffer
- For isochronous receive pipelines, IN tokens are received when there is no empty FIFO buffer plane

22.2.20 USB request type register (USBREQ)

Address: USBFS.USBREQ 4008 0054h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	BREQUEST[7:0]								BMREQUESTTYPE[7:0]							
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b7~b0	BMREQUESTTYPE[7:0]	Request type	These bits store the USB request bmRequestType value.	R/W*1
b15~b8	BREQUEST[7:0]	Request	These bits store the USB request bRequest value.	R/W*1

*1. In device controller mode, this register is readable, not writable. In host controller mode, the register can be read and written.

USBREQ stores the setup requests used to control the transfer.

USBREQ is initialized by USB bus reset.

BMREQUESTTYPE[7:0]:

The BMREQUESTTYPE[7:0] bits hold the bmRequestType value of the USB request.

- Host controller mode:

  Set these bits to the value of the USB request data in the transfer setup transaction. Do not overwrite when the DCPCTR.SUREQ bit is 1.

- Device controller mode:

  These bits indicate the value of the USB request data received in the setup transaction; writes are invalid.

BREQUEST[7:0]:

The BREQUEST[7:0] bits hold the bRequest value of the USB request.

- Host controller mode:

  Set these bits to the value of the USB request data in the Set Transfer Transaction. do not overwrite when the DCPCTR.SUREQ bit is 1.

- Device controller mode:

  These bits indicate the value of the USB request data received in the setup transaction; writes are invalid.

22.2.21 USB request value register (USBVAL)

Address: USBFS.USBVAL 4008 0056h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	WVALUE[15:0]															
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b15~b0	WVALUE[15:0]	Value	Save the value of the USB request wValue.	R/W[*1]

*1. In device controller mode, this register is readable, not writable. In host controller mode, this register can be read and written.

USBVAL is initialized by USB bus reset.

WVALUE[15:0]:

The WVALUE[15:0] bits store the wValue value of the USB request.

- Host controller mode:

Set these bits to the wValue value of the USB request for transmission in the setup transaction. Do not overwrite when the DCPCTR.SUREQ bit is 1.

- Device controller mode:

These bits indicate the wValue value of the USB request received in the setup transaction. The write is invalid.

22.2.22 USB request index register (USBINDX)

Address: USBFS.USBINDX 4008 0058h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	WINDEX[15:0]															
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b15~b0	WINDEX[15:0]	Index	These bits store the USB request wIndex value.	R/W[*1]

*1. In device controller mode, this register is readable, not writable. In host controller mode, the register can be read and written.

The USBINDX register stores setup requests for control transfers.

The USBINDX register is initialized by a USB bus reset.

WINDEX[15:0]:

These bits hold the value of WINDEX [15:0] bits of a USB request.

- Host controller mode:

Set these bits to the wIndex value of the USB request for transmission in the setup transaction. Do not overwrite when the DCPCTR.SUREQ bit is 1.

- Device controller mode:

These bits indicate the wIndex value of the USB request received in the setup transaction, and the write is invalid.

22.2.23 USB request length register (USBLENG)

Address: USBFS.USBLENG 4008 005Ah

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	WLENTUH[15:0]															
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b15~b0	WLENTUH[15:0]	Length	These bits store the USB request wLength value.	R/W[*1]

*1. In device controller mode, this register is readable, not writable. In host controller mode, the register can be read and written.

USBLENG stores the setup requests used to control the transfer.

USBLENG is initialized by USB bus reset.

WLENTUH[15:0]:

The WLENTUH[15:0] bits hold the wLength value of the USB request:

- Host controller mode:

In the transfer setup transaction, set these bits to the wLength value in the USB request. Do not override the DCPCTR.SUREQ bit when it is 1.

- Device controller mode:

These bits indicate the wLength value in the USB request received in the receive setup transaction, and the write is invalid.

22.2.24 DCP configuration register (DCPCFG)

Address: USBFS.DCPCFG 4008 005Ch

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	-	-	-	-	-	-	-	-	SHTNA K	-	-	DIR	-	-	-	-
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b3~b0	—	Reserved	These bits are read as 0. The write value should be 0.	R/W
b4	DIR	Transfer direction *1	0: Data receiving direction 1: Data transmitting direction	R/W
b6,b5	—	Reserved	These bits are read as 0. The write value should be 0.	R/W
b7	SHTNAK	Pipe disabled at end of transfer *1	0: Pipe continued at the end of transfer 1: Pipe disabled at the end of transfer	R/W
b15~b8	—	Reserved	These bits are read as 0. The write value should be 0.	R/W

*1. Set this bit only when the PID is NAK. Before setting this bit, check if the DCPCTR.PBUSY bit is 0, then change the DCPCTR.PID[1:0] bits of the DCP from BUF to NAK. If the USBFS changes the PID[1:0] bits to NAK, there is no need to check the PBUSY bit by software.

DIR:

When the host controller is selected, the DIR bit sets the transfer direction of the data stage and status stage. When the device controller is selected, the DIR bit should be set to 0.

SHTNAK:

The SHTNAK bit specifies whether the selected pipe changes the PID to NAK after the transfer has ended on reception. Valid only for the selected pipe on reception.

When the SHTNAK bit is 1, the USBFS changes the DCPCTR.PID[1:0] bits of the DCP to NAK when it determines that the transmission has ended. The USBFS determines that the transfer has ended when:

- A short packet (including a zero-length packet) is successfully received.

22.2.25 DCP maximum packet size register (DCPMAXP)

Address: USBFS.DCPMAXP 4008 005Eh

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	DEVSEL[3:0]				-	-	-	-	-	MXPS[6:0]						
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b6~b0	MXPS[6:0]	Maximum Packet Size*1	These bits set the maximum amount of data (maximum packet size) in payloads for the DCP. b6 b0 0 0 0 1 0 0 0: 8 bytes 0 0 1 0 0 0 0: 16 bytes 0 0 1 1 0 0 0: 24 bytes 0 1 0 0 0 0 0: 32 bytes 0 1 0 1 0 0 0: 40 bytes 0 1 1 0 0 0 0: 48 bytes 0 1 1 1 0 0 0: 56 bytes 1 0 0 0 0 0 0: 64 bytes 1 0 0 1 0 0 0: 72 bytes 1 0 1 0 0 0 0: 80 bytes 1 0 1 1 0 0 0: 88 bytes 1 1 0 0 0 0 0: 96 bytes 1 1 0 1 0 0 0: 104 bytes 1 1 1 0 0 0 0: 112 bytes 1 1 1 1 0 0 0: 120 bytes Settings other than above are prohibited.	R/W
b11~b7	—	Reserved	These bits are read as 0. The write value should be 0.	R/W
b15~b12	DEVSEL[3:0]	Device Select*2	b15 b12 0 0 0 0: Address 0000 0 0 0 1: Address 0001 0 0 1 0: Address 0010 0 0 1 1: Address 0011 0 1 0 0: Address 0100 0 1 0 1: Address 0101 Settings other than above are prohibited.	R/W

*1. When the PID is NAK, only MXPS[6:0] bits are set to 1. Before setting these bits, check that the DCPCTR.PBUSY bits are 0 and then change the DCPCTR.PID[1:0] bits of the DCP from BUF to NAK. If the USBFS changes the PID[1:0] bits to NAK, it is not necessary to check the PBUSY bits by software. After modifying the MXPS[6:0] bits and setting the DCP to the CURPIPE[3:0] bits in the Port Select Register, the buffer is cleared by setting the BCLR bit in the Port Control Register to 1.

*2. Set the DEVSEL[3:0] bits to 1 only if the PID is NAK and the DCPCTR.SUREQ bit is 0. Before setting these bits, check that the DCPCTR.PBUSY bit are 0, and then change the DCPCTR.PID[1:0] bits from BUF to DCP of NAK. If USBFS changes the PID[1:0] bits to NAK, there is no need to check the PBUSY bits by software.

MXPS[6:0]:

The MXPS[6:0] bits specify the maximum amount of data (maximum packet size) in payloads for the DCP. The initial value of the bits is 40h (64 bytes). Ensure that the setting of the MXPS[6:0] bits is in compliance with USB Specification 2.0. Do not write to the FIFO buffer or set PID to BUF while the setting of the MXPS[6:0] bits is 0.

DEVSEL[3:0]:

In host controller mode, the DEVSEL[3:0] bits specify the address of the target peripheral device used to control the transfer. Firstly, set the device address in the corresponding DEVADDn (n=0 to 5) register, and then set these bits to the appropriate value. For example, before setting the DEVSEL[3:0] bits to 0010b, set the DEVADD2 register to that address at first.

In device controller mode, set these bits to 0000b.

22.2.26 DCP control register (DCPCTR)

Address: USBFS.DCPCTR 4008 0060h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	BSTS	SUREQ	—	—	SUREQ CLR	—	—	SQCLR	SQSET	SQMO N	PBUSY	—	—	CCPL	PID[1:0]	
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b1,b0	PID[1:0]	Response PID	b1 b0 0 0: NAK response 0 1: BUF response (depending on the buffer state) 1 0: STALL response 1 1: STALL response	R/W
b2	CCPL	Control transfer end enable	0: Invalid 1: Completion of control transfer is enabled.	R/W
b4,b3	—	Reserved	These bits are read as 0. The write value should be 0.	R/W
b5	PBUSY	Pipe Busy	0: DCP is not used for the transaction. 1: DCP is used for the transaction.	R
b6	SQMON	Sequence toggle bit monitor	0: DATA0 1: DATA1	R
b7	SQSET	Sequence toggle bit set*2	Set the sequence toggle bit in the DCP transmission: 0: Invalid (writing 0 is invalid) 1: Set the expected value of the next transaction to DATA1.	R/W*1
b8	SQCLR	Sequence toggle bit clear *2	Clear the sequence toggle bit in the DCP transmission: 0: Invalid (writing 0 is invalid) 1 Set the expected value of the next transaction to DATA0.	R/W*1
b10,b9	—	Reserved	These bits are read as 0. The write value should be 0.	R/W
b11	SUREQCLR	SUREQ bit clear	To clear the SUREQ bit in host controller mode: 0: Invalid (writing 0 is invalid) 1: Clear SUREQ to 0	R/W*1
b13,b12	—	Reserved	These bits are read as 0. The write value should be 0.	R/W
b14	SUREQ	Setup token transfer	To set up token transfer in host controller mode: 0: Invalid (writing 0 is invalid) 1: Transmit the setup package.	R/W
b15	BSTS	Buffer status flag	0: Buffer access is disabled. 1: Buffer access is enabled.	R

*1. Only 0 can be read.

*2. When the PID is NAK, only the SQSET and SQCLR bits are set to 1. Before setting these bits, check that the PBUSY bit is 0 and then change the PID[1:0] bits of the DCP from BUF to NAK. If the USBFS changes the PID[1:0] bits to NAK, it is not necessary to check the PBUSY bits by software.

PID[1:0]:

The PID[1:0] bits control the response type of the USBFS during control transfer.

To change the PID [1:0] setting from NAK to BUF in host controller mode:

- When setting the transmitting direction:

a. Write all transmit data to the FIFO buffer when DVSTCTR0.UACT bit is 1 and PID is NAK

b. Set the PID[1:0] bits to 01b (BUF), and then the USBFS executes the OUT transaction.

- When setting the receiving direction:

a. When DVSTCTR0.UACT bit is 1 and PID is NAK, check if the FIFO buffer is empty (or clear the buffer).

b. Set the PID[1:0] bits to 01b (BUF), and then the USBFS executes the IN transaction

USBFS changes the PID [1:0] setting in the following cases:

- When the software sets the PID[1:0] bits to BUF (01b) and the USBFS receives more data than MaxPacketSize, the USBFS sets PID[1:0] to STALL (11b)
- The USBFS sets the PID[1:0] bits to NAK (00b) when three consecutive receive errors (e.g. CRC errors) are detected
- After receiving the STALL handshake, the USBFS sets the PID [1:0] to STALL (11b).
- Device controller mode, the USBFS changes the PID [1:0] setting if
- When the setup packet is received, the USBFS sets the PID[1:0] to NAK (00b). The USBFS then sets the INTSTS0.VALID bit to 1 and cannot change the PID[1:0] setting until the software clears the VALID bit to zero.
- When the software sets the PID[1:0] bits to BUF (01b) and the USBFS receives more data than MaxPacketSize, the USBFS sets PID[1:0] to STALL (11b)
- The USBFS sets the PID [1:0] to STALL (1xb) when a control transfer sequence error is detected.
- - The USBFS sets the PID [1:0] to NAK when a USBFS bus reset is detected.

USBFS does not check the PID [1:0] setting when processing SET_ADDRESS requests.

The PID[1:0] bits are initialized by a USB bus reset.

CCPL:

In device controller mode, setting the CCPL bit to 1 enables the status phase of the control transfer to be completed. When the PID[1:0] bit of the associated bit is set to BUF and the CCPL bit is set to 1 by software, the USBFS completes the control transfer status phase.

During control read transfer, the USBFS transmits the ACK handshake in response to the OUT transaction from the USB host, and transmits the zero-length packet in response to the IN transaction from the USB host during control write or no-data control transfer. However, on detecting the SET_ADDRESS request, the USBFS operates in auto response mode from the setup stage up to the status stage completion irrespective of the setting of the CCPL bit.

The USBFS modifies the CCPL bit from 1 to 0 on receiving a new setup packet. 1 cannot be written to the CCPL bit by software while the INTSTS0.VALID flag is 1. The CCPL bit is initialized by a USB bus reset.

When the host controller is selected, be sure to write 0 to the CCPL bit.

PBUSY:

The PBUSY flag indicates whether DCP is used or not for the transaction when USBFS changes the PID[1:0] bits from 01b (BUF) to 00b (NAK). The USBFS modifies the PBUSY flag from 0 to 1 upon start of the USBFS transaction for the relevant pipe, and modifies the PBUSY flag from 1 to 0 upon completion of one transaction.

Reading the PBUSY flag after the PID[1:0] bits have been set to 00b (NAK) by software allows checking whether modification of the pipe settings is possible.

For details, refer to section 22.3.4.1, Pipe Control Register Switching Procedures.

SQMON:

The SQMON flag indicates the expected value of the sequence toggle bit for the next transaction during the DCP transfer.

The USBFS allows the SQMON flag to toggle upon normal completion of the transaction. However, the SQMON flag is not allowed to toggle when a DATA-PID mismatch occurs during the transfer in the receiving direction.

When the device controller is selected, the USBFS sets the SQMON flag to 1 (specifies DATA1 as the expected value) upon successful reception of the setup packet.

When the device controller is selected, the USBFS does not reference the SQMON flag during the IN/OUT transaction of the status stage, and does not allow the SQMON flag to toggle upon normal completion.

SQSET:

The SQSET bit specifies DATA1 as the expected value of the sequence toggle bit for the next transaction during the DCP transfer. The SQSET bit indicates 0.

Do not set the SQCLR and SQSET bits to 1 simultaneously.

SQCLR:

The SQCLR bit specifies DATA0 as the expected value of the sequence toggle bit for the next transaction during the DCP transfer. The SQCLR bit indicates 0.

Do not set the SQCLR and SQSET bits to 1 simultaneously.

SUREQCLR:

In host controller mode, setting the SUREQCLR bit to 1 will clear the SUREQ bit to 0. This bit indicates 0.

If the transfer stops when the SUREQ bit is set to 1 in a setup transaction, set the SUREQCLR bit to 1 via software. this is not necessary at the end of a normal setup transaction, as USBFS will automatically clear the SUREQ bit to 0.

When the DVSTCTR0.UACT bit is 0, the SUREQ bit can only be controlled by the SUREQCLR bit. When UACT is 0, communication is suspended or no transmission is performed because a bus disconnection is detected.

In device controller mode, be sure to write 0 to the SUREQ bit.

SUREQ:

The USBFS transmits the setup packet by setting the SUREQ bit to 1 when the host controller is selected. After completing the setup transaction process, the USBFS generates either the SACK or SIGN interrupt and sets the SUREQ bit to 0. The USBFS also sets the SUREQ bit to 0 when software sets the SUREQCLR bit to 1.

Before setting the SUREQ bit to 1, set the DCPMAXP.DEVSEL[3:0] bits, registers USBREQ, USBVAL, USBINDX, and USBLENG appropriately to transmit the desired USB request in the setup transaction. Before setting this bit to 1, check that the PID[1:0] bits for the DCP are set to 00b (NAK). After setting the SUREQ bit to 1, do not modify the DCPMAXP.DEVSEL[3:0] bits, registers USBREQ, USBVAL, USBINDX, or USBLENG until the setup transaction is completed (the SUREQ bit = 1). Write 1 to the SUREQ bit only when transmitting the setup token; for other purposes, write 0.

When the device controller is selected, be sure to write 0 to the SUREQ bit.

BSTS:

Indicates whether DCP FIFO buffer access is enabled or disabled. The meaning of the BSTS flag depends on the setting of ISEL bit in the port select register as shown below:

- When the ISEL bit = 0, the BSTS flag indicates whether the received data can be read from the buffer.
- When the ISEL bit = 1, the BSTS flag indicates whether the data to be transmitted can be written to the buffer.

22.2.27 Pipe window select register (PIPESEL)

Address: USBFS.PIPESEL 4008 0064h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	PIPESEL[3:0]			
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b3~b0	PIPESEL[3:0]	Pipe window select	b3 b0 0 0 0 0: No pipe selected 0 0 0 1: Pipe 1 0 0 1 0: Pipe 2 0 0 1 1: Pipe 3 0 1 0 0: Pipe 4 0 1 0 1: Pipe 5 0 1 1 0: Pipe 6 0 1 1 1: Pipe 7 1 0 0 0: Pipe 8 1 0 0 1: Pipe 9 Settings other than above are prohibited.	R/W
b15~b4	—	Reserved	These bits are read as 0. The write value should be 0.	R/W

PIPE1 to PIPE 9 should be set using registers PIPESEL, PIPECFG, PIPEMAXP, PIPEPERI, PIPEnCTR, PIPEnTRE, and PIPEnTRN (n=0~9).

After selecting the pipe using the PIPESEL register, functions of the pipe should be set using registers PIPECFG, PIPEMAXP, and PIPEPERI. PIPEnCTR, PIPEnTRE, and PIPEnTRN can be set regardless of the pipe selection in the PIPESEL register.

PIPESEL[3:0]:

The PIPESEL[3:0] bits select the pipe number corresponding to registers PIPECFG, PIPEMAXP, and PIPEPERI which data are written to or read from.

Selecting a pipe number through the PIPESEL[3:0] bits allows writing to and reading from registers PIPECFG, PIPEMAXP, and PIPEPERI which correspond to the selected pipe number.

When PIPESEL[3:0] = 0000b, 0 is read from all of the bits in registers PIPECFG, PIPEMAXP, and PIPEPERI. Writing to these bits is invalid.

22.2.28 Pipe configuration register (PIPECFG)

Address: USBFS.PIPECFG 4008 0068h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TYPE[1:0]		—	—	—	BFRE	DBLB	—	SHTNA K	—	—	DIR	EPNUM[3:0]			
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b3~b0	EPNUM[3:0]	Endpoint number *1	These bits specify the endpoint number for the selected pipe. Setting 0000b means an unused pipe.	R/W
b4	DIR	Transfer direction *2,*3	0: Receiving direction 1: Transmitting direction	R/W
b6,b5	—	Reserved	These bits are read as 0. The write value should be 0.	—
b7	SHTNAK	Pipe disabled at end of transfer *1	0: Pipe assignment continued at the end of transfer 1: Pipe assignment disabled at the end of transfer	R/W
b8	—	Reserved	These bits are read as 0. The write value should be 0.	—
b9	DBLB	Double buffer mode*2,*3	0: Single buffer 1: Double buffer	R/W
b10	BFRE	BRDY interrupt operation specification *2,*3	0: BRDY interrupt upon transmitting or receiving data 1: BRDY interrupt upon completion of reading data	R/W
b13~b11	—	Reserved	These bits are read as 0. The write value should be 0.	—
b15,b14	TYPE[1:0]	Transfer type *1	• Pipe 1 and 2 b15 b14 0 0: Pipe not used 0 1: Bulk transfer 1 0: Setting prohibited 1 1: Isochronous trnasfer • Pipe 3~5 b15 b14 0 0: Pipe not used 0 1: Bulk transfer 1 0: Setting prohibited 1 1: Setting prohibited. • Pipe 6~9 b15 b14 0 0: Pipe not used 0 1: Setting prohibited 1 0: Interrupt transfer 1 1: Setting prohibited	R/W

*1. The TYPE[1:0], SHTNAK and EPNUM[3:0] bits can be set only when the PID is NAK. Before setting these bits, check that the PIPEnCTR.PBUSY bit is 0, then change the PIPEnCTR.PID[1:0] bit from 01b (BUF) to 00b (NAK). If the USBFS changes the PID[1:0] bits to 00 (NAK), there is no need to check the PBUSY bits by software.

*2. The BFRE, DBLB, and DIR bits should be set only before the PID is NAK and the pipe is selected in the CURPIPE[3:0] bits of the Port Selection Register. Before setting these bits, check that the PIPEnCTR.PBUSY bit is 0, and then change the PIPEnCTR.PID[1:0] bit from 01b (BUF) to 00b (NAK). If the USBFS changes the PID[1:0] bits to 00 (NAK), there is no need to check the PBUSY bits by software.

*3. To change the BFRE, DBLB, or DIR bits after completing USB communication on the selected pipe, in addition to the constraints described in *2, clear the FIFO buffer allocated to the selected pipe by writing 1 and then 0 to the PIPEnCTR.ACLRM bit successively by software.

The PIPECFG specifies the transfer type, FIFO buffer access direction, and endpoint number for pipes 1 through

9. It also selects the single or double buffer mode and whether to continue or disable the pipe operation at the end of the transfer.

EPNUM[3:0]:

EPNUM[3: 0] bits specify the endpoint number for the selected pipe. Setting 0000b means an unused pipe.

Do not make the settings such that the combination of the settings of the DIR and EPNUM[3:0] bits should be the same for two or more pipes. EPNUM[3:0] bits = 0000b can be set for all of the pipes.

DIR:

The DIR bit specifies the direction of transmission for the selected pipe.

When the software sets this bit to 0, the USBFS uses the selected pipe for receiving. When the software sets this bit to 1, the USBFS uses the selected pipe for transmitting.

SHTNAK:

The SHTNAK bit specifies whether the PIPEnCTR.PID[1:0] bit is changed to 00b (NAK) at the end of transmission when the selected pipe is set in the receive direction. This bit is valid for pipes 1 through 5 in the receive direction.

When the software sets this bit of the receive pipe to 1, the USBFS changes the PIPEnCTR.PID[1:0] bits corresponding to 00b (NAK) on determining the end of the transfer. The USBFS determines that the transfer has ended on any of the following conditions:

- A short packet (including a zero-length packet) is successfully received.
- The transaction counter is used and the number of packets specified by the counter are successfully received.

DBLB:

The DBLB bit selects either single or double buffer mode for the FIFO buffer used by the selected pipe. The DBLB bit is valid when PIPE1 to PIPE5 are selected.

BFRE:

The BFRE bit specifies the BRDY interrupt generation timing from the USBFS to the CPU with respect to the selected pipe.

When the BFRE bit has been set to 1 by software and the selected pipe is in the receiving direction, the USBFS detects the transfer completion and generates the BRDY interrupt on having read the relevant packet.

When the BRDY interrupt is generated with the above conditions, 1 should be written to the BCLR bit in the port control register by software. The FIFO buffer assigned to the selected pipe is not enabled for reception until 1 is written to the BCLR bit.

When the BFRE bit has been set to 1 by software and the selected pipe is in the transmitting direction, the USBFS does not generate the BRDY interrupt.

For details, refer to section 22.3.3.1 BRDY interrupt.

TYPE[1:0]:

The TYPE[1:0] bits select the transfer type for the pipe selected by the PIPESEL.PIPESEL[3:0] bits. Before setting the PID to BUF and before starting USB communication using the selected pipe, set the TYPE[1:0] bits to a value other than 00b.

22.2.29 Pipe maximum packet size register (PIPEMAXP)

Address: USBFS.PIPEMAXP 4008 006Ch

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	DEVSEL[3:0]				—	—	—	MXPS[8:0]								
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b8~b0	MXPS[8:0]	Maximum packet size *2	• Pipe 1 and 2: 1 byte (001h) to 256 bytes (100h) • Pipe 3~5: 8 bytes(008h),16 bytes(010h) 32(020h),64 bytes(040h) (Bits [8:7] and [2:0] are not supported.) • Pipe 6~9: 1 byte (001h) to 64 bytes (040h) (Bits [8:7] are not supported)	R/W
b11~b9	—	Reserved	These bits are read as 0. The write value should be 0.	—
b15~b12	DEVSEL[3:0]	Device selection *3	b3 b0 0 0 0 0: Address 0000 0 0 0 1: Address 0001 0 0 1 0: Address 0010 0 0 1 1: Address 0011 0 1 0 0: Address 0100 0 1 0 1: Address 0101 Settings other than above are prohibited.	R/W

*1. The value of these bits is 0000h when no pipe is selected with the PIPESEL.PIPESEL[3:0] bits and 0040h when a pipe is selected.

*2. Set the MXPS[8:0] bits only before the PID is NAK and the pipe is selected in the CURPIPE[3:0] bits of the Port Selection Register. Before setting these bits, check that the PIPEnCTR.PBUSY bit is 0, and then change the PIPEnCTR.PID[1:0] bit from 01b (BUF) to 00b (NAK). If the USBFS changes the PID[1:0] bits to 00 (NAK), there is no need to check the PBUSY bits by software.

*3. The DEVSEL[3:0] bits can be set only when the PID is NAK. Before setting these bits, check that the PIPEnCTR.PBUSY bit is 0, and then change the PIPEnCTR.PID[1:0] bit from 01b (BUF) to 00b (NAK). If the USBFS changes the PID[1:0] bits to 00 (NAK), there is no need to check the PBUSY bits by software.

The PIPEMAXP specifies the maximum packet size for pipes 1 to 9.

MXPS[8:0]:

The MXPS[8:0] bits specify the maximum data payload (maximum packet size) for the selected pipe.

These bits should be set to the appropriate value for each transfer type based on USB Specification 2.0. Note that the maximum value of Pipe 1 and Pipe 2 is 256. While MXPS[8:0] = 000h, do not write to the FIFO buffer or do not set the PID[1:0] bits to 01b (BUF). These operations are not valid.

DEVSEL[3:0]:

In host controller mode, the DEVSEL[3:0] bits specify the address of the target device to be used for USB communication. First set the device address in the associated DEVADDn (n=0 to 5) register, and then set these bits to the corresponding values. For example, to set the DEVSEL[3:0] bits to 0010b, first set the address of the target device in the DEVADD2 register.

In device controller mode, set these bits to 0000b.

22.2.30 Pipe cycle control register (PIPEPERI)

Address: USBFS.PIPEPERI 4008 006Eh

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	IFIS	—	—	—	—	—	—	—	—		IITV[2:0]		
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b2~b0	IITV[2:0]*1	Interval error detection interval	Specify the interval error detection timing for the selected pipe in terms of frames, which is expressed as nth power of 2.	R/W
b11~b3	—	Reserved	These bits are read as 0. The write value should be 0.	—
b12	IFIS	Isochronous IN buffer flush	0: The buffer is not flushed. 1: The buffer is flushed.	R/W
b15~b13	—	Reserved	These bits are read as 0. The write value should be 0.	—

*1. The IITV[2:0] bits can be set only when the PID is NAK. Before setting these bits, check that the PBUSY bit is 0, and then change the PID[1:0] bit from 01b (BUF) to 00b (NAK). If the USBFS changes the PID[1:0] bits to 00 (NAK), it is not necessary to check the PBUSY bits by software.

The PIPEPERI register selects whether the buffer is flushed or not when an interval error occurred during isochronous IN transfer, and sets the interval error detection interval for PIPE1 to PIPE9.

IITV[2:0]:

To change the IITV[2:0] bits to another value after they are set and USB communication is performed, set the PIPEnCTR.PID[1:0] bits to 00b (NAK) and then set the PIPEnCTR.ACLRM bits to 1 to initialize the interval timer.

The IITV[2:0] bits are invalid for pipe 3 to pipe 5, and set the IITV[2:0] bits to 000 for pipe 3 to pipe 5.

IFIS:

The IFIS bit specifies whether to flush the buffer when the pipe selected by the PIPESEL.PIPESEL[3:0] bits is used for isochronous IN transfers.

In device controller mode, when the selected pipe is used for isochronous IN transfers, the USBFS automatically clears the FIFO buffer if the USBFS fails to receive an IN token from the USB host within the frame interval set by the IITV[2:0] bits. When specified as double buffered (PIPECFG.DBLB=1), USBFS clears only the data in the plane used earlier.

The USBFS clears the FIFO buffer immediately after receiving a SOF packet when the USBFS is expecting to receive a frame with an IN token. Even if the SOF packet is damaged, the USBFS also clears the FIFO buffer at the right timing to receive the SOF packet by using the internal interpolation function.

Set this bit to 0 when the host controller function is selected, and to 0 when the selected pipe is not used for isochronous transmission.

22.2.31 PIPEn control registers (PIPEnCTR)(n=1~9)

PIPEnCTR(n=1~5)

Address:USBFS.PIPE1CTR 4008 0070h, USBFS.PIPE2CTR 4008 0072h, USBFS.PIPE3CTR 4008 0074h, USBFS.PIPE4CTR 4008 0076h, USBFS.PIPE5CTR 4008 0078h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	BSTS	INBUFM	—	—	—	ATREPM	ACLRM	SQCLR	SQSET	SQMO N	PBUSY	—	—	—	PID[1:0]	
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b1,b0	PID[1:0]	Response PID	b1 b0 0 0: NAK response 0 1: BUF response (depending on the buffer state) 1 0: STALL response 1 1: STALL response	R/W
b4 to b2	—	Reserved	These bits are read as 0. The write value should be 0.	—
b5	PBUSY	Pipe busy flag	0: The relevant pipe is not used for the transaction. 1: The relevant pipe is used for the transaction.	R
b6	SQMON	Sequence toggle bit confirmation	0: DATA0 1: DATA1	R
b7	SQSET	Sequence toggle bit set *2	Set the sequence toggle bit of pipe n: 0: Invalid (writing 0 is invalid) 1: Specify DATA1.	R/W*1
b8	SQCLR	Sequence toggle bit clear *2	Clear the sequence toggle bit of pipe n: 0: Invalid (writing 0 is invalid) 1: Specify DATA0.	R/W*1
b9	ACLRM	Auto buffer clear mode *3	0: Dsiabled 1: Enabled (all buffers are initialized)	R/W
b10	ATREPM	Auto Response Mode *2	0: Auto response is disabled. 1: Auto response is enabled.	R/W
b13 to b11	—	Reserved	These bits are read as 0. The write value should be 0.	—
b14	INBUFM	Transmit buffer monitor	0: There is no data to be transmitted in the buffer memory 1: There is data to be transmitted in the buffer memory	R
b15	BSTS	Buffer status	0: Buffer access by the CPU is disabled. 1: Buffer access by the CPU is enabled.	R

*1. Only 0 can be read, and only 1 can be written.

*2. Set the ATREPM bit or write 1 to the SQCLR or SQSET bit only when the PID is NAK. Before setting these bits, check that the PBUSY bit is 0 and then change the PID[1:0] bit from 01b (BUF) to 00b (NAK). If the USBFS changes the PID[1:0] bits to 00 (NAK), it is not necessary to check the PBUSY bits by software.

*3. The ACLRM bit can be set only before the PID is NAK and the pipe is selected in the CURPIPE[3:0] bits of the Select Port Selection Register. Before setting this bit, check that the PBUSY bit is 0, and then change the PID[1:0] bit from 01b (BUF) to 00b (NAK). If USBFS changes the PID[1:0] bit to 00 (NAK), there is no need to check the PBUSY bit by software.

The PIPEnCTR register can be set regardless of the pipe selection in the PIPESEL register.

PID[1:0]:

The PID[1:0] bits specify the response type for the next transaction of the relevant pipe.

The default setting of PID[1:0] is NAK. Change the PID[1:0] setting to BUF to use the associated pipe for USBFS transfer. And Table 22-7 and Table 22-8 show the basic operation of USBFS according to the setting of PID[1:0] (when there is no error in communication data preservation).

During USBFS communication on the selected pipe, after changing the PID [1:0] setting from BUF to NAK by software, check that the PBUSY bit is 1 to determine if the USBFS transfer on the selected pipe has actually entered the NAK state. If the USBFS changes the PID [1:0] bits to NAK, there is no need to check the PBUSY bits by software.

The USBFS modifies the setting of the PIPEnCTR.PID [1:0] bits in the following cases:

- When the selected pipe is receiving and the software sets the PIPECFG.SHTNAK bit of the selected pipe to 1, the USBFS sets the PID to NAK after recognizing the completion of the transmission
- The USBFS sets the PID to STALL (11b) when it receives a packet with a payload that exceeds the maximum packet size of the selected pipe
- The USBFS sets the PID to NAK when a USB bus reset is detected in device controller mode
- In host controller mode, the USBFS sets the PID to NAK when it detects three consecutive receive errors (e.g. CRC errors)
- When the STALL handshake is received in host controller mode, the USBFS sets the PID to STALL (11b).

To specify the response type, set the PID [1:0] bits as follows:

- To make a transition from NAK (00b) to STALL, set 10b.
- To make a transition from BUF (01b) to STALL, set 11b.
- To make a transition from STALL (11b) to NAK, set 10b and then 00b.
- To make a transition from STALL to BUF, first to NAK, then to BUF.

Table 22-7 USBFS operation in host controller mode based on PID [1:0] settings

PID[1:0]value	Transfer type	Transfer direction (DIR bit)	USBFS operation
00b(NAK)	Not dependent on settings	Not dependent on settings	No token issued
01b(BUF)	Batch or interrupt transfers	Not dependent on settings	A token is issued when the DVSTCTR0.UACT bit is 1 and the FIFO buffer associated with the selected pipeline is ready for transmitting and receiving. No token is issued when the DVSTCTR0.UACT bit is 0 or when the FIFO buffer associated with the selected pipeline is not ready for transmitting or receiving.
	Isochronous transfer	Not dependent on settings	Issue a token regardless of the status of the FIFO buffer associated with the selected pipe
10b(STALL) or 11b(STALL)	Not dependent on settings	Not dependent on settings	No token issued

Table 22-8 USBFS operation in device controller mode based on PID [1:0] settings

PID[1:0]value	Transfer type	Transfer direction (DIR bit)	USBFS operation
00b(NAK)	Batch or interrupt transfers	Not dependent on settings	Return NAK in response to USB host token
	Isochronous transfer	Not dependent on settings	Respond to the USB host token without returning any content
01b(BUF)	Bulk Transfer	Receiving direction (DIR=0)	If the FIFO buffer associated with the selected pipe is ready to receive, the data is received and an ACK is returned in response to the OUT token from the USB host.
	Interrupt transfer	Receiving direction (DIR=0)	If the FIFO buffer associated with the selected pipe is ready to receive, the data is received and an ACK is returned in response to the OUT token from the USB host.
	Batch or interrupt transfers	Transmitting direction (DIR=1)	If the FIFO buffer associated with the selected pipe is ready to be transmitted, transmit data in response to a token from the USB host. Otherwise, return NAK.
	Isochronous transfer	Receiving direction (DIR=0)	If the FIFO buffer associated with the selected pipe is ready to receive, the data is received in response to an OUT token from the USB host. Otherwise, the data is discarded.
	Isochronous transfer	Transmitting direction (DIR=1)	If the associated FIFO buffer is ready to transmit, transmits data in response to a token from the USB host. Otherwise, transmits zero-length packets.
10b(STALL) or 11b(STALL)	Batch or interrupt transfers	Not dependent on settings	Return STALL in response to a token from the USB host

PBUSY:

The PBUSY bit indicates whether the selected pipe is currently being used for a transaction.

USBFS changes the PBUSY bit from 0 to 1 at the beginning of a USBFS transaction for the selected pipe, and changes the PBUSY bit from 1 to 0 at the completion of a transaction.

After setting the PID to NAK, reading the PBUSY bit with software checks if the pipeline settings can be changed. For more information, see 22.3.4.1 Pipe control register switching procedures.

SQMON:

The SQMON flag indicates the expected value of the sequence toggle bit for the next transaction of the relevant pipe.

When the relevant pipe is not for the isochronous transfer, the USBFS allows the SQMON flag to toggle upon normal completion of the transaction. However, the SQMON flag is not allowed to toggle when a DATA-PID mismatch occurs during the transfer in the receiving direction.

SQSET:

Setting the SQSET bit to 1 through software allows the USBFS to set DATA1 as the expected value of the sequence toggle bit of the relevant pipe. The USBFS sets the SQSET bit to 0.

SQCLR:

Setting the SQCLR bit to 1 through software allows the USBFS to set DATA0 as the expected value of the sequence toggle bit of the relevant pipe. The USBFS sets the SQCLR bit to 0.

ACLRM:

Enables or disables auto buffer clear mode for the relevant pipe. To delete the information in the FIFO buffer assigned to the relevant pipe completely, write 1 and then 0 to the ACLRM bit continuously.

Table 22-9 shows the information cleared by writing 1 and 0 to the ACLRM bit continuously and the cases in which clearing the information is necessary.

Table 22-9 USBFS clears data when ACLRM=1

NO.	Set the data cleared by the ACLRM bit	Cases in which data needs to be cleared
1	All data in the FIFO buffer allocated to the selected pipe	When initializing the selected pipeline
2	The interval count value when the selected pipe is for isochronous transfer	When resetting the interval count value
3	Internal flags associated with PIPECFG.BFRE	When changing the setting of PIPECFG.BFRE
4	FIFO buffer switching control	When changing the setting of PIPECFG.DBLB
5	Internal flags associated with transaction counts	When the forced transaction counting function is terminated

ATREPM:

The ATREPM enables or disables auto response mode for the relevant pipe.

When the device controller is selected and the relevant pipe is for bulk transfer, the ATREPM bit can be set to 1. When the ATREPM bit is set to 1, the USBFS responds to the token from the USB host as described below:

- When the relevant pipe is for bulk IN transfer (the PIPECFG.TYPE[1:0] bits = 01b and the PIPECFG.DIR bit = 1):

  a) When the ATREPM = 1 and PID =BUF, the USBFS transmits a zero-length packet in response to the IN token.

  b)The USBFS updates (allows toggling of) the sequence toggle bit (DATA-PID) each time the USBFS receives ACK from the USB host. In a single transaction, IN token is received, zero-length packet is transmitted, and then ACK is received. In this case, the USBFS does not generate the BRDY or BEMP interrupt.

- When the relevant pipe is for bulk OUT transfer (the PIPECFG.TYPE[1:0] = 01b and the PIPECFG.DIR = 0)

- When ATREPM=1 and PID=BUF,USBFS returns NAK in response to OUT token and generates the NRDY interrupt.

For USB communication in auto response mode, set the ATREPM bit to 1 while the FIFO buffer is empty. Do not write to the FIFO buffer during USB communication in auto response mode. When the relevant pipe is for isochronous transfer, be sure to set the ATREPM bit to 0.

When the host controller is selected, be sure to set the ATREPM bit to 0.

INBUFM:

The INBUMFM indicates the relevant FIFO buffer status when the relevant pipe is in the transmitting direction.

When the relevant pipe is transmitting (the PIPECFG.DIR bit = 1), the USBFS sets the INBUFM flag to 1 when the CPU or DMA completes writing data to at least one FIFO buffer plane.

The USBFS sets the INBUFM flag to 0 when transmitting the data from the FIFO buffer plane. In double buffer mode (PIPECFG.DBLB = 1), the USBFS sets the INBUFM bit to 0 when it has finished transmitting data from both FIFO buffer planes and the CPU or DMA has not finished writing data from one FIFO buffer plane.

The INBUFM flag indicates the same value as the BSTS flag when the relevant pipe is receiving (PIPECFG.DIR = 0).

BSTS:

The BSTS bit indicates the FIFO buffer status of the selected pipe.

The meaning of the BSTS bits depends on the PIPECFG.DIR, PIPECFG.BFRE and DnFIFOSEL.DCLRM settings, as shown in Table 22-10.

Table 22-10 BSTS bit operation

DIR value	BFRE value	DCLRM value	BSTS bit function
0	0	0	Set to 1 when the received data can be read from the FIFO buffer, and set to 0 when the data reading is completed.
		1	Prohibit settings
	1	0	Set to 1 when the received data can be read from the FIFO buffer; set to 0 when the BCLR bit in the port control register is set to 1 by the software after the data reading is completed.
		1	Set to 1 when the received data can be read from the FIFO buffer, and set to 0 when the data reading is completed.
1	0	0	Can be set to 1 when the transmitted data is written to the FIFO buffer and to 0 when the data is finished.
		1	Prohibit settings
	1	0	Prohibit settings
		1	Prohibit settings

PIPEnCTR(n=6~9)

Address:USBFS.PIPE6CTR 4008 007Ah,USBFS.PIPE7CTR 4008 007Ch,USBFS.PIPE8CTR 4008 007Eh, USBFS.PIPE9CTR 4008 0080h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	BSTS	—	—	—	—	—	ACLRM	SQCLR	SQSET	SQMON	PBUSY	—	—	—	PID[1:0]	
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b1,b0	PID[1:0]	Response PID	b1 b0 0 0: NAK response 0 1: BUF response (depending on the buffer state) 1 0: STALL response 1 1: STALL response	R/W
b4~b2	—	Reserved	These bits are read as 0. The write value should be 0.	—
b5	PBUSY	Pipe busy	0: The selected pipe is not used for the transaction. 1: The selected pipe is used for the transaction.	R
b6	SQMON	Sequence toggle bit confirmation	0: DATA0 1: DATA1	R
b7	SQSET	Set sequence toggle bit*2	Set the sequence toggle bit of pipe n: 0: Invalid (writing 0 is invalid) 1: Specify DATA1.	R/W*1
b8	SQCLR	Clear sequence toggle bit*2	Clear the sequence toggle bit of pipe n: 0: Invalid (writing 0 is invalid) 1: Specify DATA0.	R/W*1
b9	ACLRM	Auto buffer clear mode *3	0: Disable 1: Enable(Initialize all buffers)	R/W
b14~b10	—	Reserved	These bits are read as 0. The write value should be 0.	—
b15	BSTS	Buffer status	0: Disable CPU buffer access 1: Enable CPU buffer access	R

*1. Only 0 can be read, and only 1 can be written.

*2. Set the ATREPM bit or write 1 to the SQCLR or SQSET bit only when the PID is NAK. Before setting these bits, check that the PBUSY bit is 0 and then change the PID[1:0] bit from 01b (BUF) to 00b (NAK). If the USBFS changes the PID[1:0] bits to 00 (NAK), it is not necessary to check the PBUSY bits by software.

*3. The ACLRM bit can be set only before the PID is NAK and the pipe is selected in the CURPIPE[3:0] bits of the Select Port Selection Register. Before setting this bit, check that the PBUSY bit is 0, and then change the PID[1:0] bit from 01b (BUF) to 00b (NAK). If USBFS changes the PID[1:0] bit to 00 (NAK), th ere is no need to check the PBUSY bit by software.

PID[1:0]:

The PID[1:0] bits specify the response type for the next transaction in the selected pipe.

The default setting of PID[1:0] is NAK. Change the PID[1:0] setting to BUF to use the associated pipe for USBFS transfer. Table 22-7 and Table 22-8 show the basic operation of USBFS according to the setting of PID[1:0] (when there is no error in the transmitted and received packets).

During USBFS communication on the selected pipe, after changing the PID[1:0] setting from BUF to NAK by software, check that the PBUSY bit is 1 to determine if the USBFS transfer on the selected pipe has actually entered the NAK state. If the USBFS changes the PID[1:0] bits to NAK, there is no need to check the PBUSY bits by software.

The USBFS modifies the setting of the PIPEnCTR.PID [1:0] bits in the following cases:

- USBFS sets the PID to STALL (11b) when it receives a packet with a payload that exceeds the maximum packet size of the selected pipe
- USBFS sets the PID to NAK when a USB bus reset is detected in device controller mode
- In host controller mode, the USBFS sets the PID to NAK when it detects three consecutive receive errors (e.g. CRC errors)
- When the STALL handshake is received in host controller mode, the USBFS sets the PID to STALL (11b).

To specify the response type, set the PID [1:0] bits as follows:

- To make a transition from NAK (00b) to STALL, set 10b.
- To make a transition from BUF (01b) to STALL, set 11b.
- To make a transition from STALL (11b) to NAK, set 10b and then 00b.
- To make a transition from STALL to BUF, first to NAK, then to BUF.

PBUSY:

The PBUSY flag indicates whether the relevant pipe is being currently used or not for the transaction.

The USBFS modifies the PBUSY flag from 0 to 1 upon start of the USBFS transaction for the relevant pipe, and modifies the PBUSY flag from 1 to 0 upon completion of one transaction.

Reading the PBUSY flag after the PID has been set to NAKby software allows checking whether modification of the pipe settings is possible.

SQMON:

The SQMON flag indicates the expected value of the sequence toggle bit for the next transaction of the relevant pipe.

The USBFS allows the SQMON flag to toggle upon normal completion of the transaction. However, the SQMON flag is not allowed to toggle when a DATA-PID mismatch occurs during the transfer in receiving.

SQSET:

The SQSET bit should be set to 1 to set DATA1 as the expected value of the sequence toggle bit for the next transaction of the relevant pipe. The USBFS sets the SQSET bit to 0.

SQCLR:

Setting the SQCLR bit to 1 by software causes the USBFS to clear the desired value of the sequence switch bit for the next transaction of the selected pipe to DATA0. The USBFS sets the SQCLR bit to 0.

ACLRM:

The ACLRM bit enables or disables the automatic buffer clear mode for the selected pipe. To completely clear the data in the FIFO buffer assigned to the selected pipe, write 1 and then 0 to the ACLRM bit consecutively.

Table 22-11 lists the data cleared by successive writing 1 and 0 to the ACLRM bits and the cases where this processing is required.

Table 22-11 USBFS clears data when ACLRM=1

NO.	Set the data cleared by the ACLRM bit	Cases in which data needs to be cleared
1	All data in the FIFO buffer allocated to the selected pipe	When initializing the selected pipe
2	Interval count value when the selected pipe is used to interrupt transfer and the host controller is selected	When resetting the interval count value
3	Internal flags associated with PIPECFG.BFRE	When changing the setting of PIPECFG.BFRE
4	Internal flags associated with transaction counts	When the forced transaction counting function is terminated

BSTS:

The BSTS bit indicates the FIFO buffer status of the selected pipe.

The meaning of the BSTS bits depends on the PIPECFG.DIR, PIPECFG.BFRE and DnFIFOSEL.DCLRM settings, as shown in Table 22-10.

22.2.32 PIPEn transaction counter enable register (PIPEnTRE) (n=1~5)

Address: USBFS.PIPE1TRE 4008 0090h, USBFS.PIPE2TRE 4008 0094h, USBFS.PIPE3TRE 4008 0098h, USBFS.PIPE4TRE 4008 009Ch, USBFS.PIPE5TRE 4008 00A0h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	TRENB	TRCLR	—	—	—	—	—	—	—	—
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b7~b0	—	Reserved	These bits are read as 0. The write value should be 0.	—
b8	TRCLR	Transaction counter clear	0: invalid (writing 0 is invalid) 1: The current counter value is cleared.	R/W
b9	TRENB	Transaction counter enable	0: Transaction counter is disabled. 1: Transaction counter is enabled.	R/W
b15~b10	—	Reserved	These bits are read as 0. The write value should be 0.	—

Note: When the PID is NAK, set each bit in PIPEnTRE. Before setting these bits, check the PIPEnCTR.PBUSY bits for 0 and then change the PIPEnCTR.PID[1:0] bits of the selected pipe from BUF to NAK. If the USBFS changes the PID[1:0] bits to NAK, it is not necessary to check the PBUSY bits by software.

TRCLR:

When the TRCLR bit is set to 1, the USBFS clears the current value of the transaction counter associated with the selected pipe, and then sets the TRCLR bit to 0.

TRENB:

The TRENB bit enables or disables the transaction counter.

For the receive pipes, after setting the total number of packets to be received to the PIPEnTRN.TRNCNT[15:0] bits by software, setting the TRENB bit to 1 will enable the USBFS to do the following control when receiving the same number of packets as the TRNCNT[15:0] setting value:

- When the PIPECFG.SHTNAK bit is 1, the USBFS changes the PID bit of the associated pipe to NAK after receiving the same number of packets as the TRNCNT [15:0] setting value.
- When the PIPECFG.BFRE bit is 1, the USBFS will read the BRDY interrupt after receiving the same number of packets as the TRNCNT[15:0] setting value, and then read the last received data.

For the pipe in transmitting, set the TRENB bit to 0.

When the transaction counter is not used, set the TRENB bit to 0. When the transaction counter is used, set the TRNCNT[15:0] bits before setting the TRENB bit to 1. Set the TRENB bit to 1 before receiving the first packet to be counted by the transaction counter.

22.2.33 PIPEn transaction counter register (PIPEnTRN) (n=1~5)

Address: USBFS.PIPE1TRN 4008 0092h, USBFS.PIPE2TRN 4008 0096h, USBFS.PIPE3TRN 4008 009Ah, USBFS.PIPE4TRN 4008 009Eh, USBFS.PIPE5TRN 4008 00A2h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TRNCNT[15:0]															
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b15~b0	TRNCNT[15:0]	Transaction counter	When written to: Specifies the total of packets (number of transactions) to be received in corresponding PIPE. When read from: If the PIPEnTRE.TRENB bit is 0, it indicates the number of transactions specified. If the PIPEnTRE.TRENB bit is 1, it indicates the number of transactions currently counted.	R/W

The PIPEnTRN register retains the setting by a USB bus reset.

TRNCNT[15:0]:

The USBFS increments the value of the TRNCNT[15:0] bits by one when all of the following conditions are satisfied:

- PIPEnTRE.TRENB=1
- (TRNCNT[15:0] set value ≠ current counter value + 1) on receiving the packet.
- The payload of the received packet agrees with the setting of the PIPEMAXP.MXPS[8:0] bits.

The USBFS sets the value of the TRNCNT[15:0] bits to 0 when any of the following conditions are met:

When all of the following conditions are met:

- PIPEnTRE.TRENB=1
- (TRNCNT[15:0] set value = current counter value + 1) on receiving the packet.
- The payload of the received packet agrees with the setting of the PIPEMAXP.MXPS[8:0] bits.

When all of the following conditions are met.

- PIPEnTRE.TRENB=1
- The USBFS has received a short packet.

When all of the following conditions are met.

- PIPEnTRE.TRENB=1
- The PIPEnTRE.TRCLR bit has been set to 1 by software.

For the pipe in transmitting, set the TRNCNT[15:0] bits to 0. When the transaction counter is not used, set the TRNCNT[15:0] bits to 0.

Setting the number of transactions to be transferred to the TRNCNT[15:0] bits is only enabled when the PIPEnTRE.TRENB bit is 0. To modify the number of transactions to be transferred, set the TRCLR bit to 1 (to clear the current counter value) before setting the PIPEnTRE.TRENB bit to 1.

22.2.34 Device address n configuration register (DEVADDn) (n=0~5)

Address: USBFS.DEVADD0 4008 00D0h, USBFS.DEVADD1 4008 00D2h, USBFS.DEVADD2 4008 00D4h, USBFS.DEVADD3 4008 00D6h, USBFS.DEVADD4 4008 00D8h, USBFS.DEVADD5 4008 00DAh

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	USBSPD[1:0]		—	—				
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b5~b0	—	Reserved	These bits are read as 0. The write value should be 0.	—
b7~b6	USBSPD[1:0]	Transfer speed of communication target devices	b7 b6 0 0: DEVADDn is not used 0 1: Low-speed 1 0: Full-speed 1 1: Prohibited settings	R/W
b15~b8	—	Reserved	These bits are read as 0. The write value should be 0.	—

USBSPD[1:0]:

The USBSPD[1:0] bits specify the USB transfer speed of the communication target peripheral device.

When the host controller is selected, the USBFS refers to the setting of the USBSPD[1:0] bits to generate packets. When the device controller is selected, set these bits to 00b.

22.2.35 USB clock resume control register (CKREC)

Address: USBFS.CKREC 4008 00C4h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	CKRECC
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	CKRECC	Enable clock resume operations *1	0: Clock resumes OFF. 1: Clock resumes ON.	R/W
b15~b1	—	Reserved	These bits are read as 0. The write value should be 0.	—

*1: Clock resume operation enable/disable control is only available when USB is not connected.

CKRECC:

When clock resume operation is enabled, the expected reception interval is not adjusted (adjustment of sof_timer) and the interpolation circuit is executed every 48000 clocks at 48 MHz. This bit is set to 0 when in host controller mode.

22.2.36 USB module control register (USBMC)

Address: USBFS.USBMC 4008 00CCh

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	VDCEN	—	—	—	—	—	—	VDDUS BE
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	VDDUSBE	USB reference power circuit on/off control	0: The USB reference power circuit is turned off. 1: The USB reference power circuit is turned on.	R/W
b1	—	Reserved	These bits are read as 1. The write value should be 1.	—
b6~b2	—	Reserved	These bits are read as 0. The write value should be 0.	—
b7	VDCEN	USB voltage regulator on/off control	0: USB regulator is off. 1: The USB regulator is turned on.	R/W
b15~b8	—	Reserved	These bits are read as 0. The write value should be 0.	—

VDDUSBE:

The USB power supply circuit generates the reference voltage for battery charging. Set this bit to 1 when using the battery charging function.

VDCEN:

The VDCEN bit controls the USB voltage regulator circuit. Set this bit to 1 when using the USB regulator circuit.

22.2.37 BC control register 0 (USBBCCTRL0)

Address: USBFS.USBBCCTRL0 4008 00B0h

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	PDDETSTS0	CHGDETSTS0	BATCHGE0	—	VDMSRCE0	IDPSINKE0	VDPSRCE0	IDMSINKE0	IDPSRCE0	RPDME0
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Bit name	Description	R/W
b0	RPDME0	D- pin pull-down control	0: Pull-down off 1: Pull-down on	R/W
b1	IDPSRCE0	D+ pin IDPSRC output control	0: Stop 1: 10μA output	R/W
b2	IDMSINKE0	D- pin 0.6V input detection (comparator and sink) control	0: Detection off 1: Detection on (comparator and sink current on)	R/W
b3	VDPSRCE0	D+ pin VDPSRC (0.6V) output control	0: No output 1: 0.6V output	R/W
b4	IDPSINKE0	D+ pin 0.6V input detection (comparator and sink) control	0: Detection off 1: Detection on (comparator and sink current on)	R/W
b5	VDMSRCE0	D- pin VDPSRC (0.6V) output control	0: No output 1: 0.6V output	R/W
b6	—	Reserved	This bit is read as 0. The write value should be 0.	—
b7	BATCHGE0	BC (battery charger) function general enable control	0: Disabled 1: Enabled	R/W
b8	CHGDETSTS0	D- pin 0.6V input detection status*1	0: Not detected 1: Detected	R
b9	PDDETSTS0	D+ pin 0.6V input detection status *2	0: Not detected 1: Detected	R
b15~b10	—	Reserved	These bits are read as 0. The write value should be 0.	—

*1. Valid when IDMSINKE0=1

*2. Valid when IDPSINKE0=1

RPDME0:

When using the battery charging function, set this bit to 1 to control the pull-down resistor of the D– pin.

IDPSRCE0:

With this bit set to 1, when the device controller is selected, current output is enabled upon detection of the connection of the data pin and the D+ pin is pulled up.

IDMSINKE0:

With this bit set to 1, when the device controller is selected, the USBFS module detects whether VDMSRC (0.6 V) that is output from the host to D– upon primary detection is connected, or whether VDPSRC (0.6 V) that is output from the device to D+ is connected to the device's D– via the host.

VDPSRCE0:

With this bit set to 1, when the device controller is selected, output is enabled upon primary detection and VDPSRC (0.6 V) is applied to D+.

IDPSINKE0:

With this bit set to 1, when the device controller selected, the USBFS module detects whether VDMSRC (0.6 V) that is output from the device to D- is connected to the device's D+ (DCP) via the host. When the host controller is selected, the USBFS module detects whether VDPSRC (0.6 V) that is output from the device to D+ upon primary detection is connected.

VDMSRCE0:

With this bit set to 1, when the device controller selected, output is enabled upon secondary detection and VDMSRC (0.6 V) is applied to D-. When the host controller is selected, output is enabled upon primary detection and VDMSRC (0.6 V) is applied to D-.

CHGDETSTS0:

When the host controller is selected, this flag is set to 1 if the USBFS module detects whether VDMSRC (0.6 V) that is output from the host to D- during primary detection is connected, or whether VDPSRC (0.6 V) that is output from the device to D+ is connected to the device's D– via the host.

PDDETSTS0:

When the device controller is selected, this flag is set to 1 if the USBFS module detects whether VDMSRC (0.6 V) that is output from the device to D- during secondary detection is connected to the device's D+ (DCP) via the host.

When the host controller is selected, this bit is set to 1 if the USBFS module detects whether VDPSRC (0.6 V) that is output from the device to D+ during primary detection is connected.

22.3 Operation

22.3.1 System control

This section describes the register settings that are necessary for initialization of USBFS and power consumption control.

22.3.1.1 Setting data to the USBFS related register

Setting the SYSCFG.USBE bit to 1 after starting the clock supply to the USB (SYSCFG.SCKE bit = 1) enables and starts USBFS operation.

22.3.1.2 Controller function selection

USBFS can act as a host or device controller.

Use the SYSCFG.DCFM bit to select one of these USBFS functions. The DCFM bit must be changed in the initial settings immediately after reset, or in the D+ pull-up disable state (SYSCFG.DPRPU bit = 0) and the D+ and D- pull-down disable states (SYSCFG.DRPD bit = 0).

22.3.1.3 Controlling USBFS data bus resistors

The USBFS has pull-up and pull-down resistors for the D+ and D- lines. Pull up or pull down these lines by setting the SYSCFG.DPRPU SYSCFG.DMRPU and SYSCFG.DRPD bits.

In Device Controller mode, confirm that a connection to the USB host has been established, then set the SYSCFG.DPRPU bit to 1 to pull up the D+ line (in full-speed communication), or set the SYSCFG.DMRPU bit to 1 to pull up the D- line (in low-speed communication).

During communication with the PC, if the SYSCFG.DPRPU (during full speed) or SYSCFG.DMRPU (during low speed) bit is set to 0, the USBFS disables the pull-up resistor on the USB data line, thereby notifying the USB host of the disconnection.

When the host controller is selected, set the SYSCFG.DRPD bit and pull down the D+ and D- lines.

Table 22-12 Controlling USBFS data bus resistors

SYSCFG setting			USB data bus control		Function
DRPD bit	DPRPU bit	DMRPU bit	D-	D+	
0	0	0	Open	Open	Resistor not used
0	1	0	Open	Pull-up	When operating at full speed as a device controller
0	0	1	Pull-up	Open	When operating at low speed as a device controller
1	0	0	Pull-down	Pull-down	When operating as a host controller
Other settings			—	—	Prohibit setting

22.3.1.4 Example of USBFS power connection

Figure 22-2 shows an example power connection without the USB regulator, and Figure 22-3 and Figure 22-4 show an example power connection when the USB regulator is used.

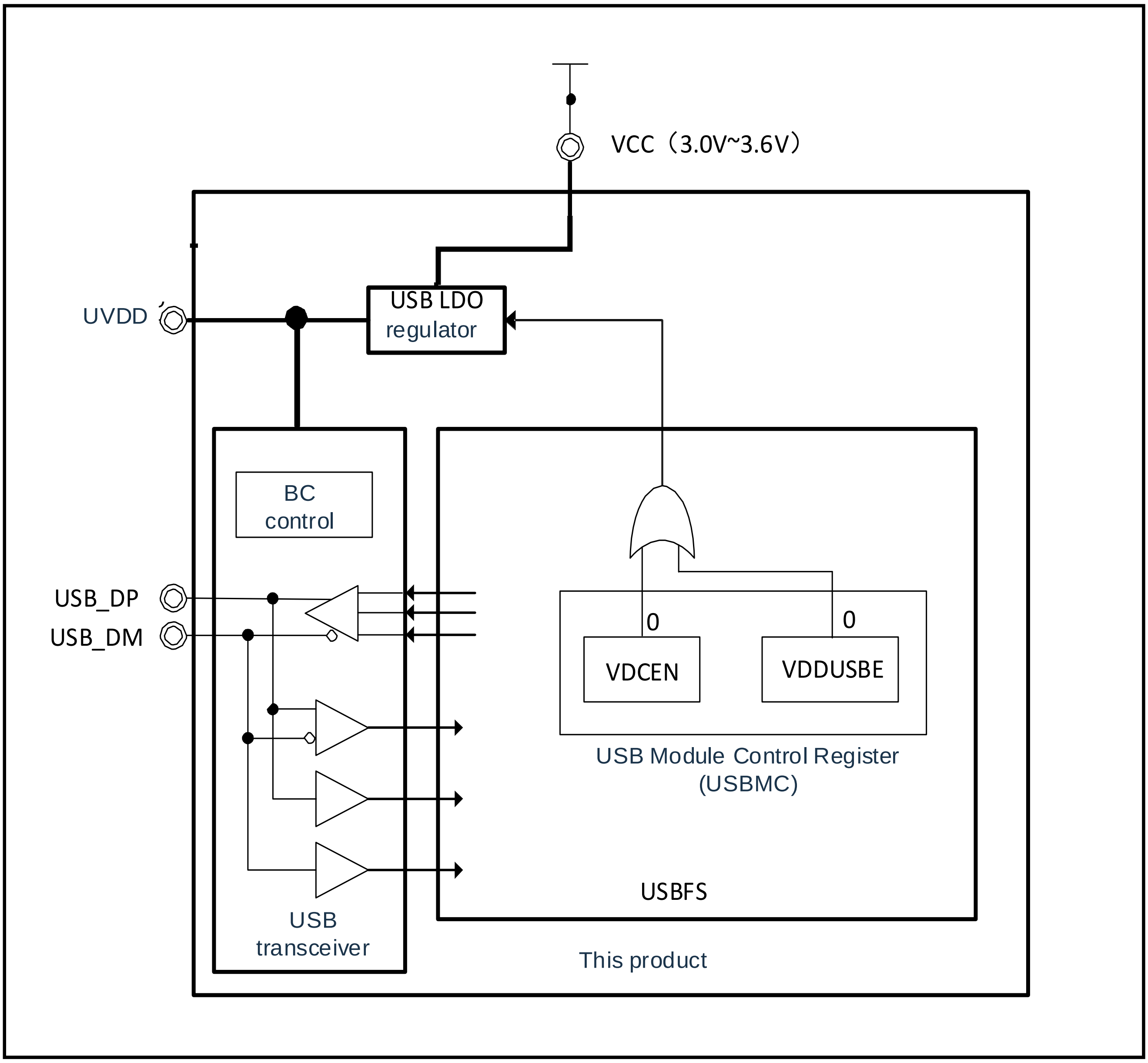
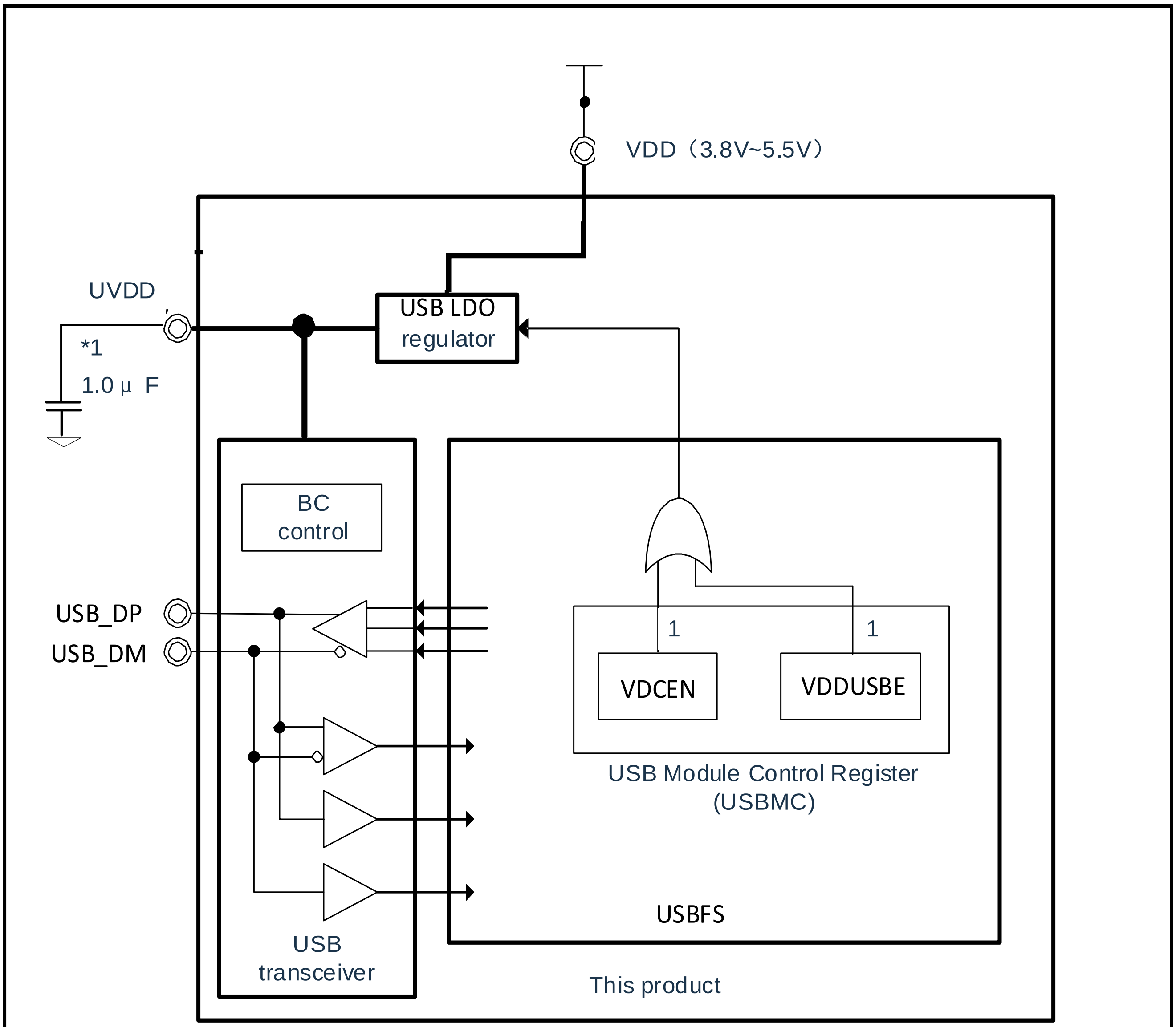
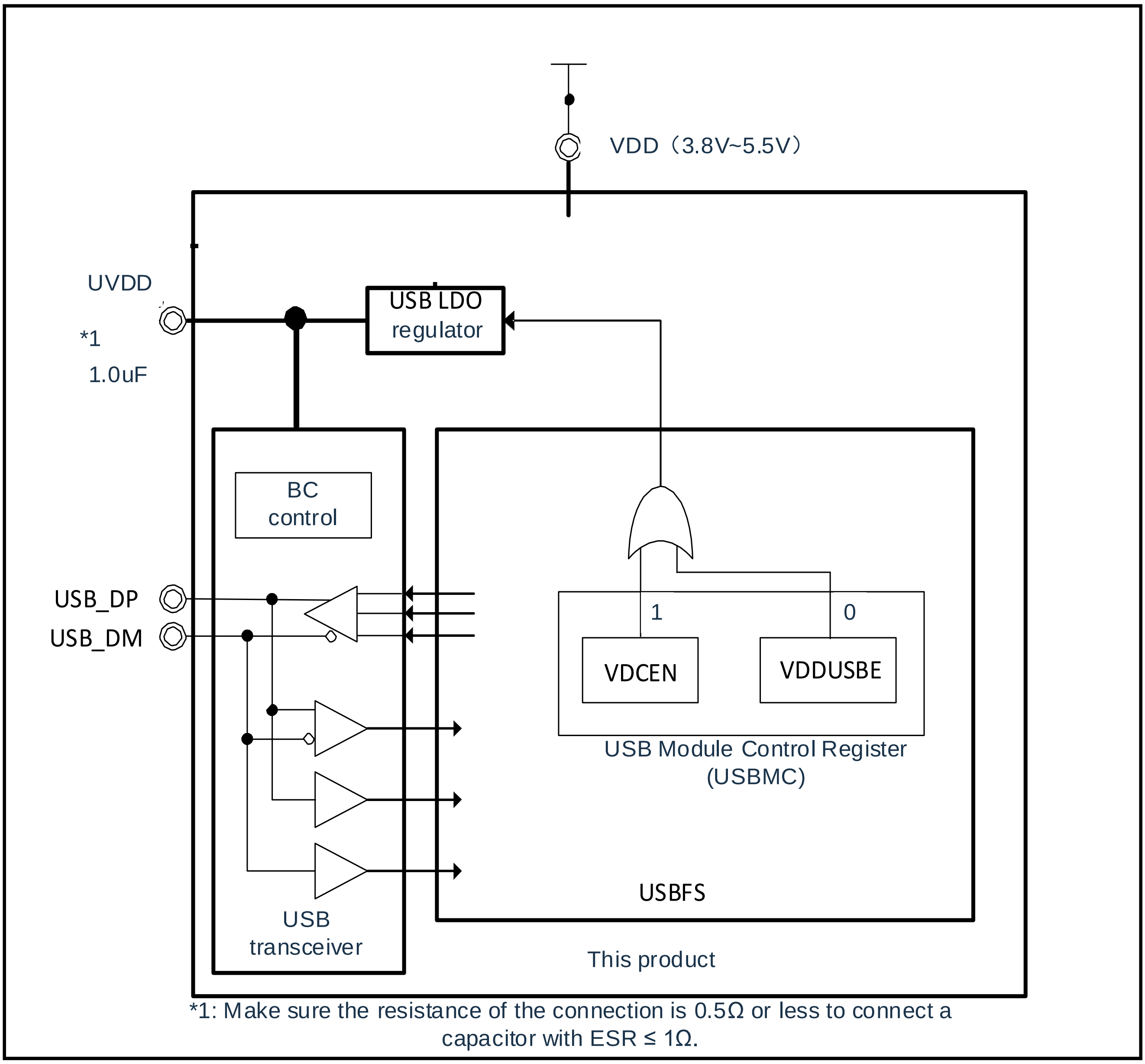


Figure 22-2 Example of power connection without USB LDO regulators



*1：Make sure the resistance of the connection is 0.5Ω or less to connect a capacitor with ESR ≤ 1Ω.

Figure 22-3 Example of power connection with a USB LDO regulator (using BC)



*1: Make sure the resistance of the connection is 0.5Ω or less to connect a capacitor with ESR ≤ 1Ω.

Figure 22-4 Example of power connection with a USB LDO regulator (without BC)

22.3.1.5 Example of USB external connection circuit

When a data line is pulled up, the host will recognize the USB device. The MCU can use an internal pull-up resistor switch for this purpose. Again, bus-powered devices do not need an external voltage regulator, as the MCU provides power in the USB-PHY.

Examples of external circuitry for USB connection are shown in Figure 22-5 and Figure 22-6.

The OTG connection of the USB connector in self-powered state is illustrated in Figure 22-5.

The USBFS controls the pull-up resistors for the D+ line and the pull-down resistors for the D+ and D- lines. Use the SYSCFG.DPRPU and SYSCFG.DRPD bits to select pull-ups and pull-downs for the bus. In device controller mode, if the SYSCFG.DPRPU bit is set to 0 while communicating with the USB host, it disables the pull-up resistors for the USB data lines. The USBFS can use it to notify the USB host device of disconnection.

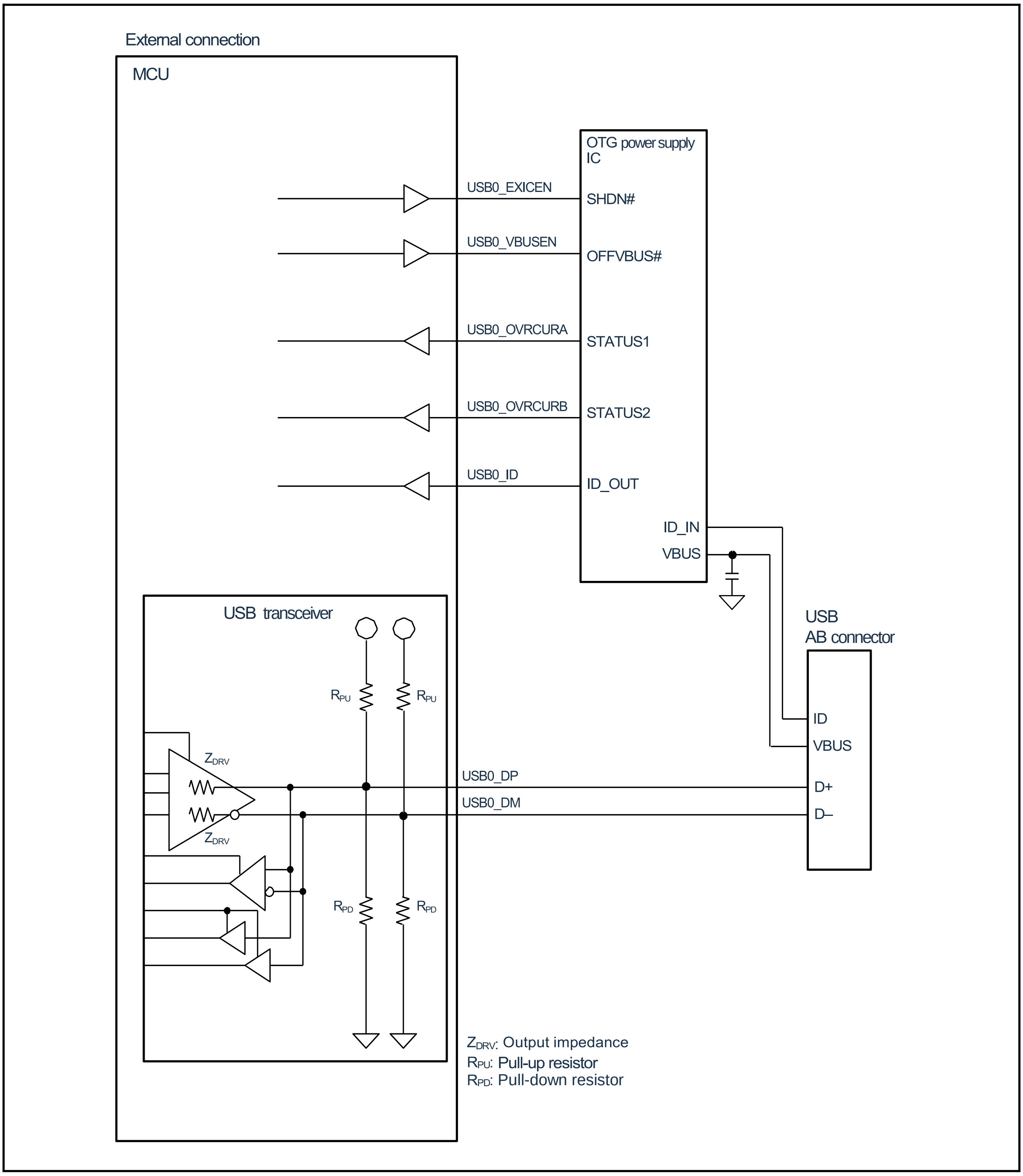
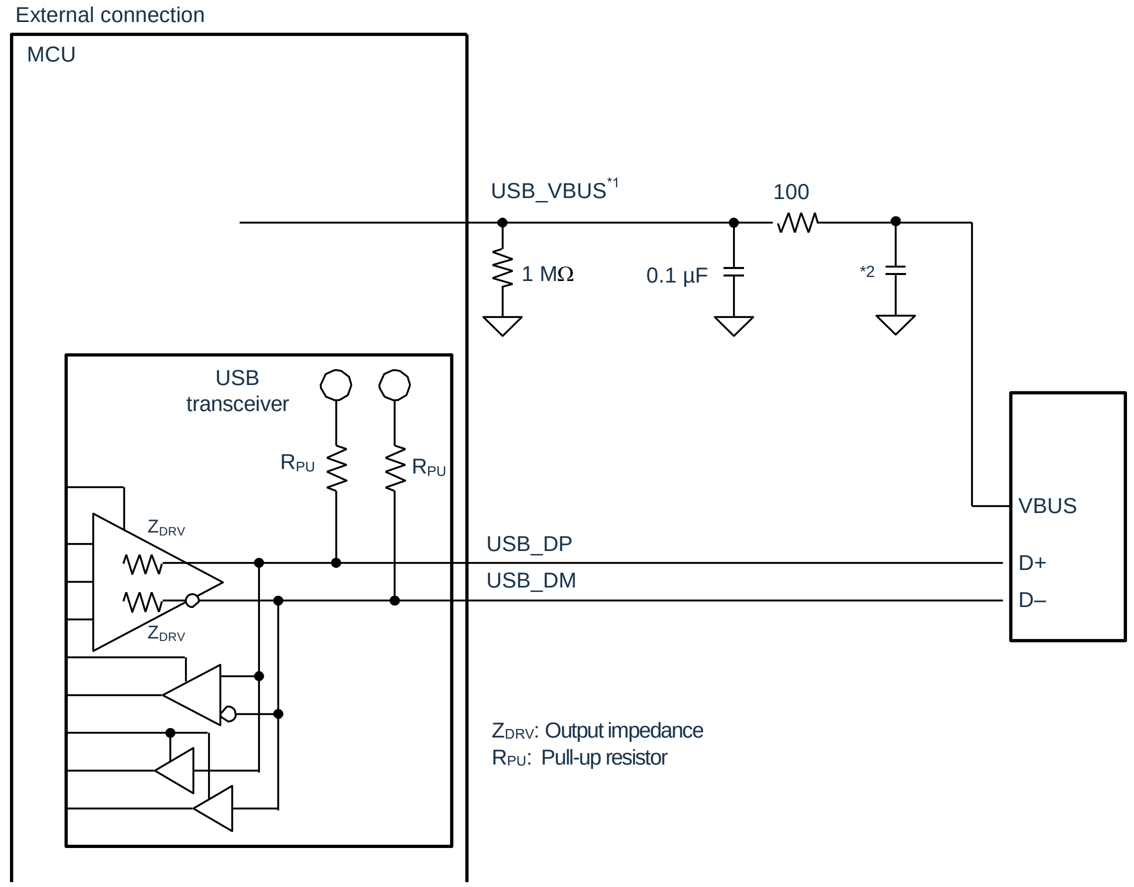


Figure 22-5 Sample OTG connection of USB connector in self-powered state

Figure 22-6 shows an example of functional connection of the USB connector in the self-powered state.

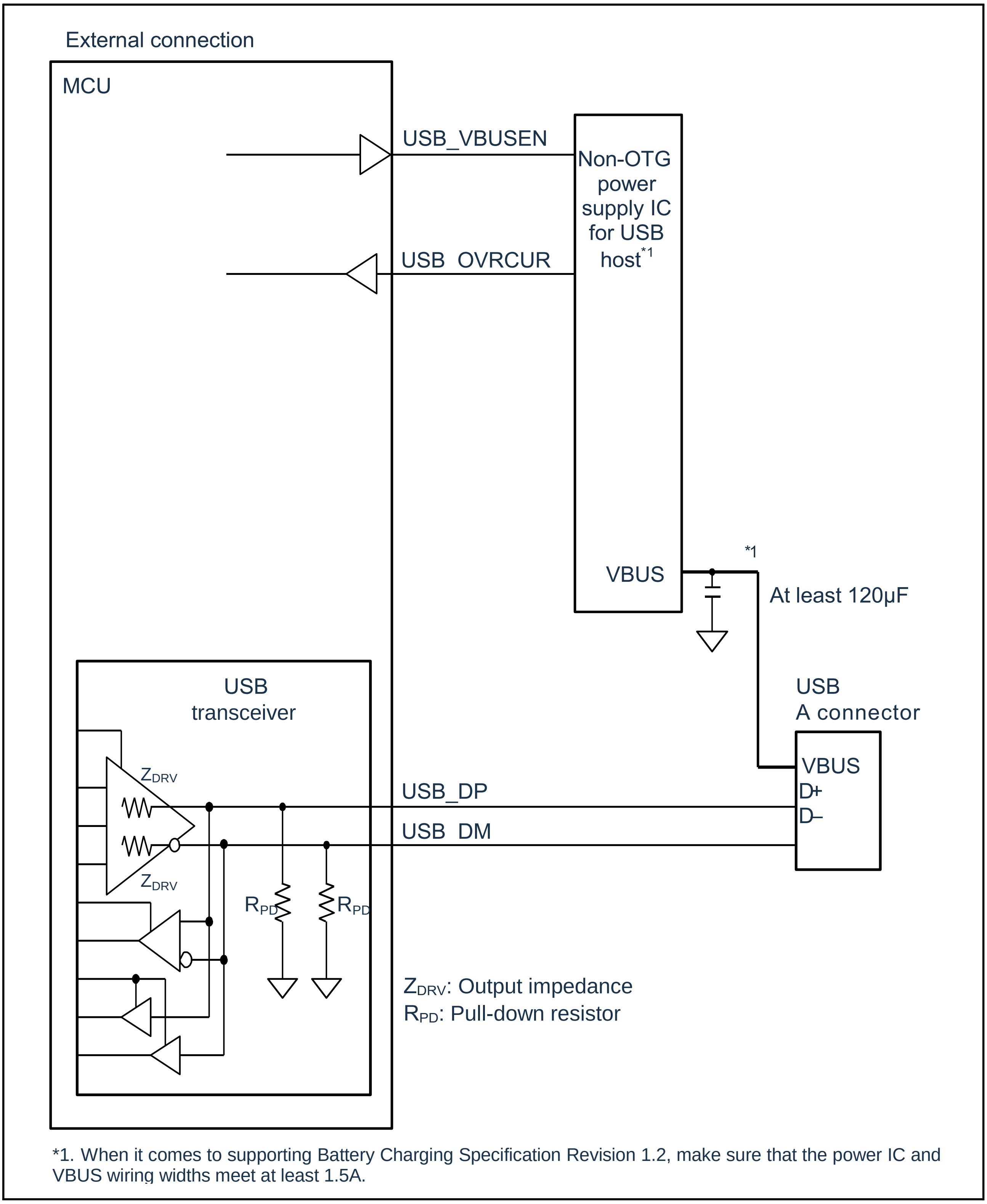


*1. USB_VBUS is 5V withstand voltage.

*2. Design the board so that the total VBUS capacitance ranges from 1.0 to 10 μF.

Figure 22-6 Example of device connection in self-powered state

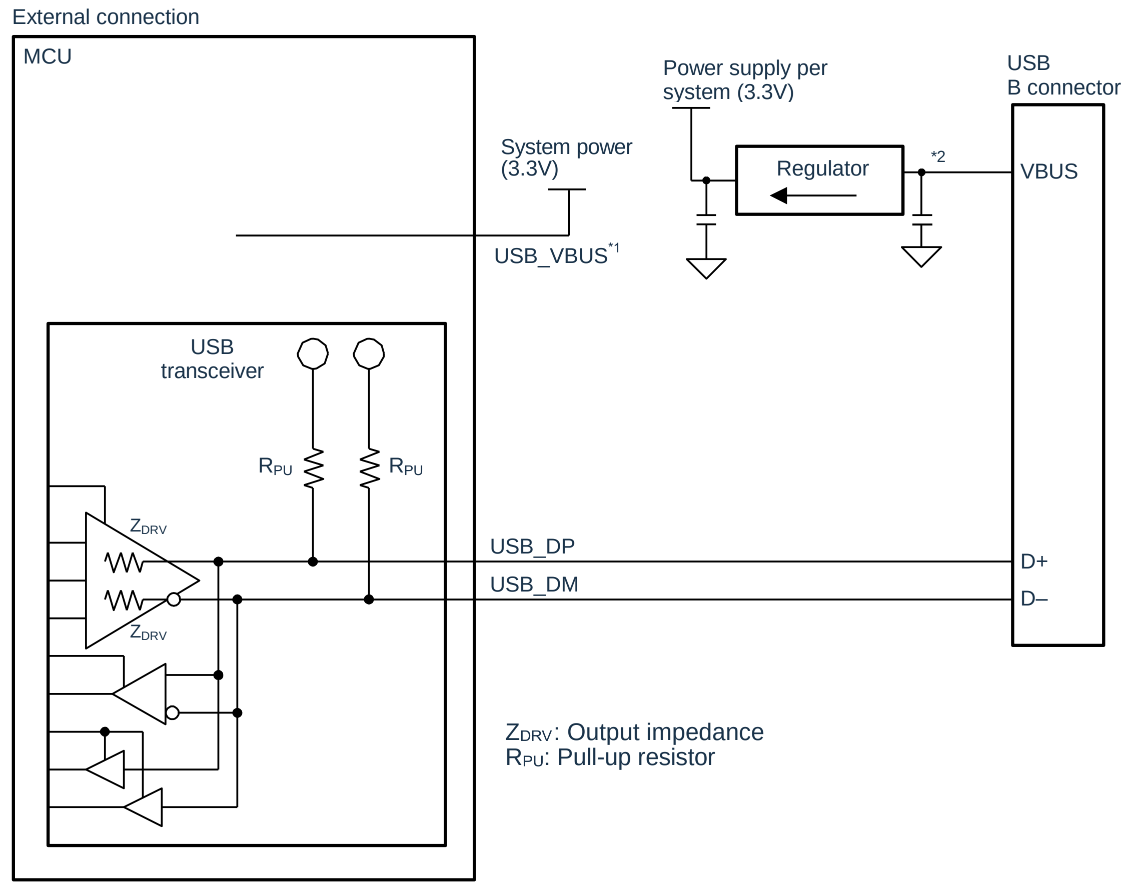
An example of the host connection for the USB connector is shown in Figure 22-7.



*1. When it comes to supporting Battery Charging Specification Revision 1.2, make sure that the power IC and VBUS wiring widths meet at least 1.5A.

Figure 22-7 Example of host connection

An example of the functional connection of the USB connector for bus power state 1 is shown in Figure 22-8.



*1. USB_VBUS is 5V withstand voltage.

*2. Design the board so that the total capacitance of VBUS ranges from 1.0 to 10 μF.

Figure 22-8 Example of device connection for bus power state 1

An example of the functional connection of the USB connector for bus-powered state 2 is shown in Figure 22-9.

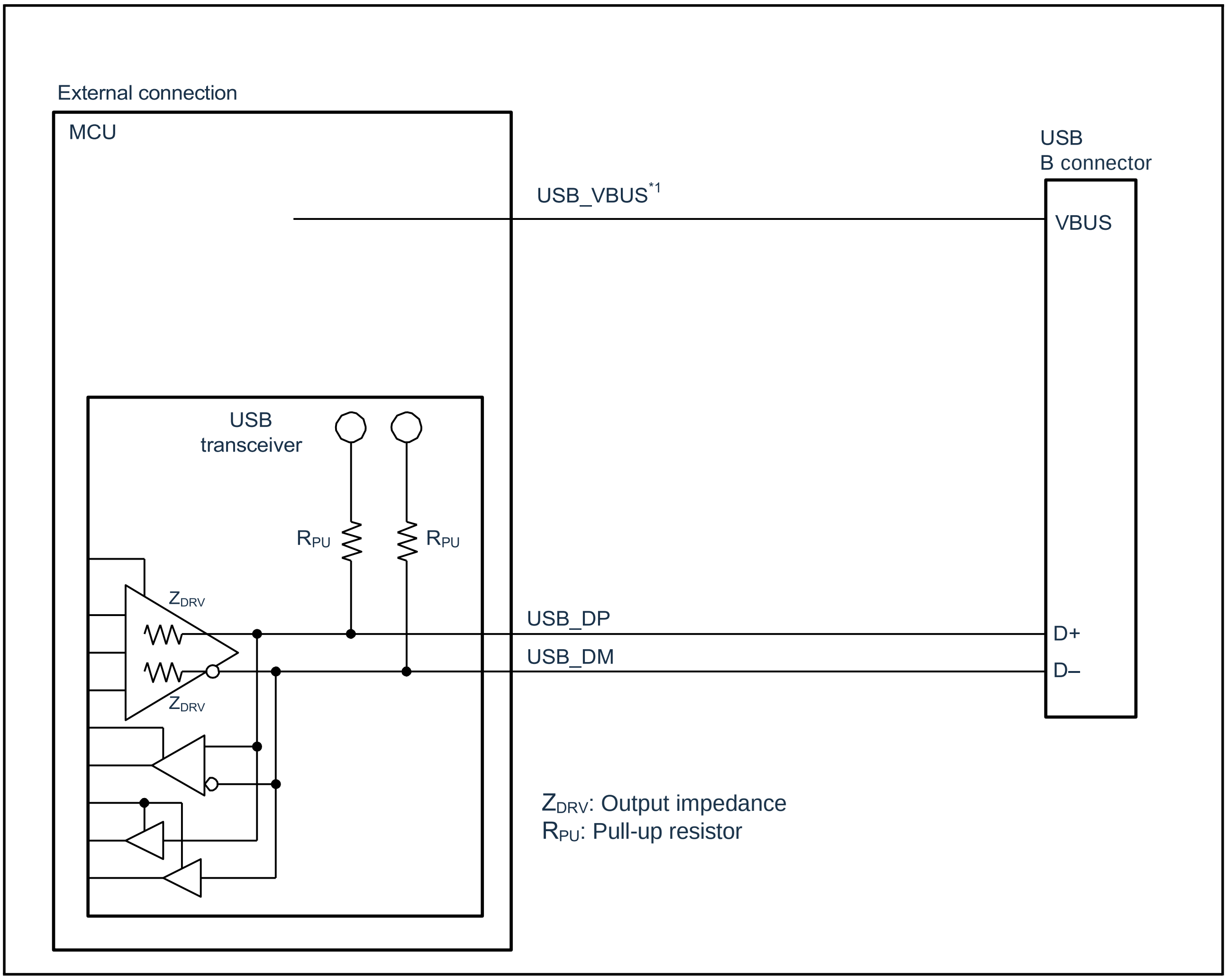
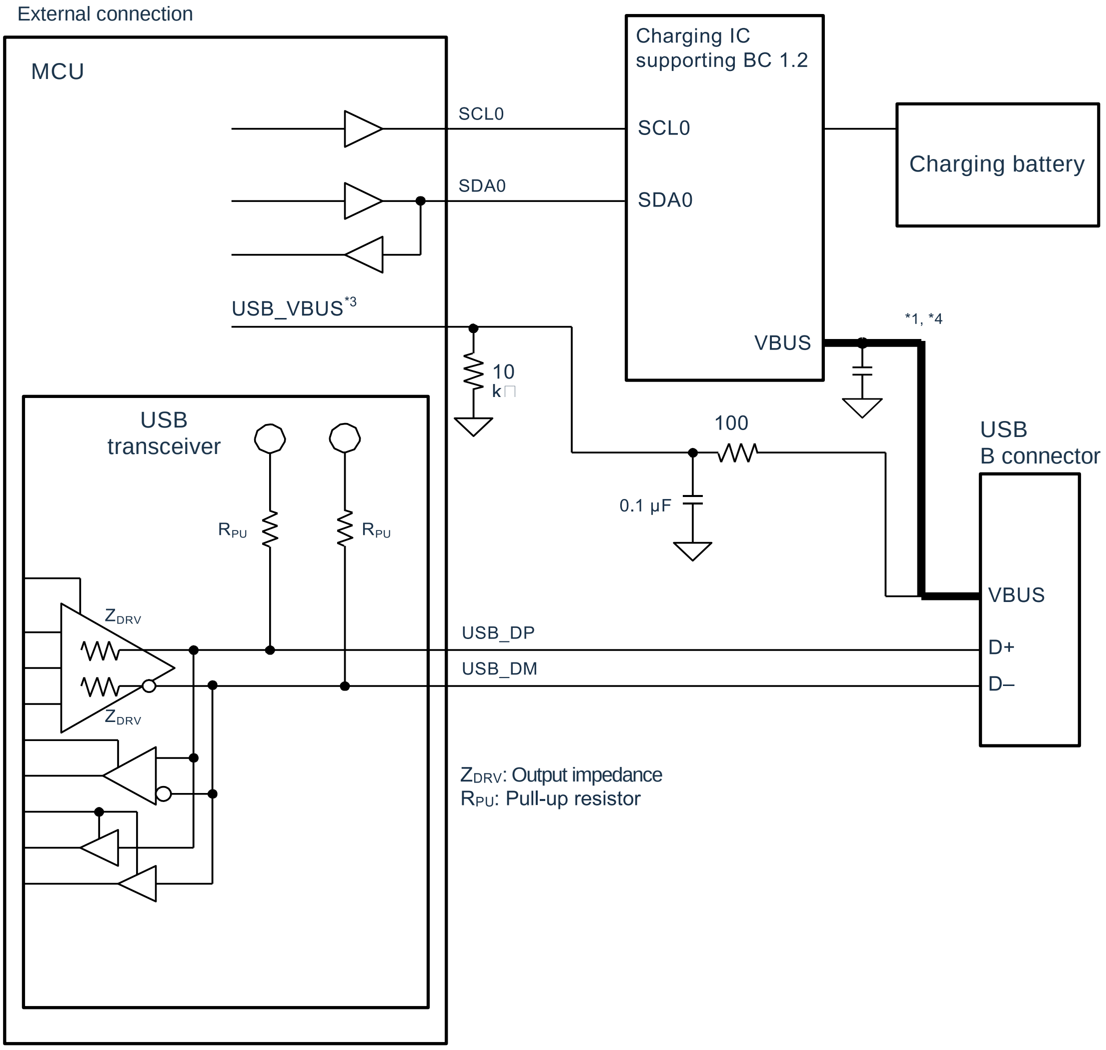


Figure 22-9 Example of device connection for bus power state 2

The examples of external circuits given in this section are simplified circuits, and their operation in every system is not guaranteed.

An example of the functional connection of the USB connector supporting battery charging version 1.2 is shown in Figure 22-10.



*1. When Battery Charging Spec. Rev.1.2 is to be supported, ensure that the VBUS wiring width is enough for at least 1.5 A (shown in bold lines).

*2. Use a resistor value such that the discharging time of VBUS is within 500 ms.

*3. USB_VBUS is 5V withstand voltage.

*4. Design the board so that the total VBUS capacitance ranges from 1.0 to 10 µF.

Figure 22-10 Example of functional connections to support Battery Charging Specification Revision 1.2

22.3.2 Interrupt sources

Table 22-13 lists the interrupt sources in the USBFS. When an interrupt generation condition is satisfied and the interrupt output is enabled using the corresponding interrupt enable register, a USBFS interrupt request is issued the Interrupt Controller (ICU) and an USBFS interrupt will be generated.

Table 22-13 Interrupt sources

Bit to be set	Name	Interrupt source	Applicable controller function	Status flag
VBINT	VBUS interrupt	• When a change in the state of the USB_VBUS input pin has been detected (low to high or high to low)	Host/Device *1	INTSTS0.VBSTS
RESM	Resume interrupt	• When a change in the state of the USB bus has been detected in the suspended state (J-state to K-state or J-state to SE0)	Device	—
SOFR	Frame number update interrupt	Host controller mode: • When an SOF packet with a different frame number has been transmitted Device controller mode: • When an SOF packet with a different frame number has been received	Host/Device	—
DVST	Device state transition interrupt	When a device state transition has been detected (any of the following conditions) • A USB bus reset detected • Suspend state detected • SET_ADDRESS request received • SET_CONFIGURATION request received	Device	INTSTS0.DVSQ[2:0]
CTRT	Control transfer stage transition interrupt	When a stage transition has been detected in control transfer (any of the following conditions) • Setup stage completed • Control write transfer status stage transition • Control read transfer status stage transition • Control transfer completed • A control transfer sequence error occurred	Device	INTSTS0.CTSQ[2:0]
BEMP	Buffer empty interrupt	• When transmission of all data in the buffer memory has been completed and the buffer has become empty • When a packet larger than the maximum packet size has been received	Host/Device	BEMPSTS.PIPEnBEMP
NRDY	Buffer not ready interrupt	Host controller mode: • For issued tokens, a STALL response is received from the peripheral device • When a response has not been received correctly from the peripheral device for the issued token (no response was returned three consecutive times or a packet reception error occurred three consecutive times) • When an overrun/underrun occurred during isochronous transfer Device controller mode: • When NAK has been returned for an IN or OUT token while the PID[1:0] bits are 01b (BUF) • When a CRC error or a bit stuffing error occurred during data reception in isochronous transfer • When an overrun/underrun occurred during data reception in isochronous transfer	Host/Device	NRDYSTS.PIPEnNRDY
BRDY	Buffer ready interrupt	• When the buffer has become ready (reading or writing state)	Host/Device	BRDYSTS.PIPEnBRDY
OVRCR	Overcurrent input change interrupt	• When a change in the state of the USB_OVRCURA or USB_OVRCURB input pin has been detected (low to high or high to low)	Host	INTSTS1.OVRCR
BCHG	Bus change interrupt	• When a change of USB bus state has been detected	Host/Device	SYSSTS0.LNST[1:0]
DTCH	Disconnection detection during	• When disconnection of a peripheral device has been detected in full- speed operation	Host	DVSTCTR0.RHST[1:0]

	full- speed operation			
ATTCH	Device connection detection	When J-state or K-state is detected on the USB port for 2.5 µs. Used for checking whether a peripheral device is connected.	Host	—
EOFERR	EOF error detection	When an EOF error of a peripheral device has been detected	Host	—
SACK	Normal setup operation	When the normal response (ACK) for the setup transaction has been received	Host	—
SIGN	Setup error	When a setup transaction error (no response or ACK packet corruption) was detected three consecutive times	Host	—
PDDEINT0	Portable device detection interrupt	When connection of the portable device has been detected	Host	INTSTS1.PDDETINT0

*1. Though this interrupt can be generated while the host function is selected, it is not usually used with the host function.

Figure 22-11 shows the circuits related to the interrupts in the USBFS.

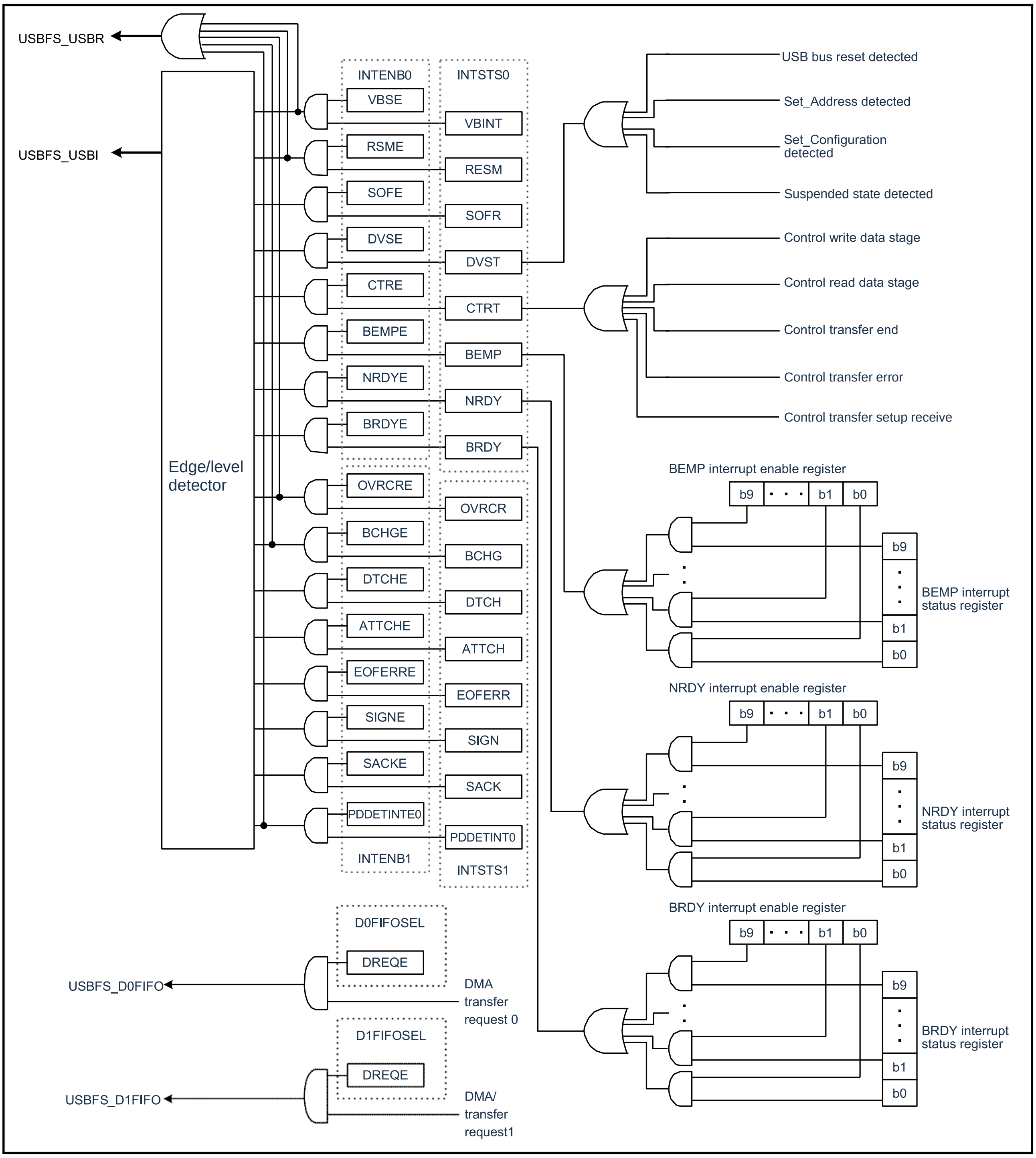


Figure 22-11: USBFS interrupt related circuits

The interrupts generated by USBFS are shown in Table 22-11.

Table 22-14 USBFS interrupts

Interrupt name	Interrupt status flag	DMA trigger	Priority
D0FIFO	DMA transfer request 0	Possible	High
D1FIFO	DMA transfer request 1	Possible	↑
USBFS_USBI	VBUS interrupt, resume interrupt, frame number update interrupt, device state transition interrupt, control transfer stage transition interrupt, buffer empty interrupt, buffer not ready interrupt, buffer ready interrupt, overcurrent input change interrupt, bus change interrupt, disconnection detection during full-speed operation, device connection detection, EOF error detection, normal setup operation, setup error, and portable device detection interrupt	Impossible	Low
USBFS_USBR	VBUS interrupt, resume interrupt, overcurrent input change interrupt, and portable device detection interrupt	Impossible	—

22.3.3 Interrupt descriptions

22.3.3.1 BRDY interrupt

The BRDY interrupt is generated in both host and device controller modes. This section describes the conditions under which USBFS sets the associated bit in BRDYSTS to 1. Under these conditions, USBFS generates a BRDY interrupt if software sets the bit in BRDYENB associated with a given pipe to 1 and sets INTENB0.BRDYE to 1.

The conditions for generating and clearing the BRDY interrupt depend on the settings of the SOFCFG.BRDYM bit and PIPECFG.BFRE bit for each pipe as described below:

1) When SOFCFG.BRDYM=0, PIPECFG.BFRE=0

With these settings, the BRDY interrupt indicates that the FIFO port is accessible.

On any of the following conditions, the USBFS generates an internal BRDY interrupt request trigger and sets 1 to the BRDYSTS.PIPEnBRDY flag corresponding to the pertinent pipe.

a) For the transmitting pipe

- When the DIR bit is changed from 0 to 1 by software.
- When packet transmission is completed using the pertinent pipe while write-access from the CPU to the FIFO buffer for the pertinent pipe is disabled (when the BSTS flag is read as 0).
- When one FIFO buffer is empty on completion of writing data to the other FIFO buffer in double buffer mode.
- No request trigger is generated until completion of writing data to the currently-written FIFO buffer even if transmission to the other FIFO buffer is completed.
- When the hardware flushes the buffer of the pipe for isochronous transfers.
- When 1 is written to the PIPEnCTR.ACLRM bit, which causes the FIFO buffer to make transition from the write-disabled to write-enabled state.

  No request trigger is generated for the DCP (that is, during data transmission for control transfers).

b) For the receiving pipe

- When packet reception is completed successfully thus enabling the FIFO buffer to be read while read-access from the CPU to the FIFO buffer for the pertinent pipe is disabled (when the BSTS flag is read as 0).
- No request trigger is generated for the transaction in which DATA-PID mismatch has occurred.
- When one FIFO buffer is read-enabled on completion of reading data from the other FIFO buffer in double buffer mode.
- No request trigger is generated until completion of reading data from the currently-read FIFO buffer even if reception by the other FIFO buffer is completed.

When the device controller is selected, the BRDY interrupt is not generated in the status stage of control transfers. The PIPEnBRDY interrupt status of the pertinent pipe can be set to 0 by writing 0 to the corresponding PIPEnBRDY flag through software. In this case, 1s should be written to the PIPEnBRDY flags for the other pipes.

Clear the BRDY status before accessing the FIFO buffer.

2) When SOFCFG.BRDYM=0 and PIPECFG.BFRE=1

With these settings, the USBFS generates a BRDY interrupt on completion of reading all data for a single transfer using the pipe in the receiving direction, and sets 1 to the bit in the BRDYSTS register corresponding to the pertinent pipe.

On any of the following conditions, the USBFS determines that the last data for a single transfer has been received.

- When a short packet including a zero-length packet is received.
- When the PIPEn transaction counter register (PIPEnTRN) is used and the number of packets specified by the PIPEnTRN.TRNCNT[15:0] bits are completely received.

When the pertinent data is completely read after any of the above conditions has been satisfied, the USBFS determines that all data for a single transfer has been completely read.

When a zero-length packet is received while the FIFO buffer is empty, the USBFS module determines that all data for a single transfer has been completely read when the FRDY flag in the FIFO port control register is 1 and the DTLN[8:0] flags are 0. In this case, to start the next transfer, write 1 to the BCLR bit in the corresponding port control register through software. With these settings, the USBFS does not detect a BRDY interrupt for the pipe in the transmitting direction.

The PIPEnBRDY interrupt status of the pertinent pipe can be set to 0 by writing 0 to the corresponding BRDYSTS.PIPEnBRDY flag through software. In this case, 1 should be written to the PIPEnBRDY flags for the other pipes.

In this mode, the PIPECFG.BFRE bit setting should not be modified until all data for a single transfer has been processed. When it is necessary to modify the PIPECFG.BFRE bit before completion of processing, all FIFO buffers for the pertinent pipe should be cleared using the PIPEnCTR.ACLRM bit.

3) When SOFCFG.BRDYM=1 and PIPECFG.BFRE=0

With these settings, the BRDYSTS.PIPEnBRDY values are linked to the BSTS flag setting for each pipe. In other words, the BRDY interrupt status bit (PIPEnBRDY) are set to 1 or 0 by the USB depending on the FIFO buffer status.

a) For the transmitting pipe

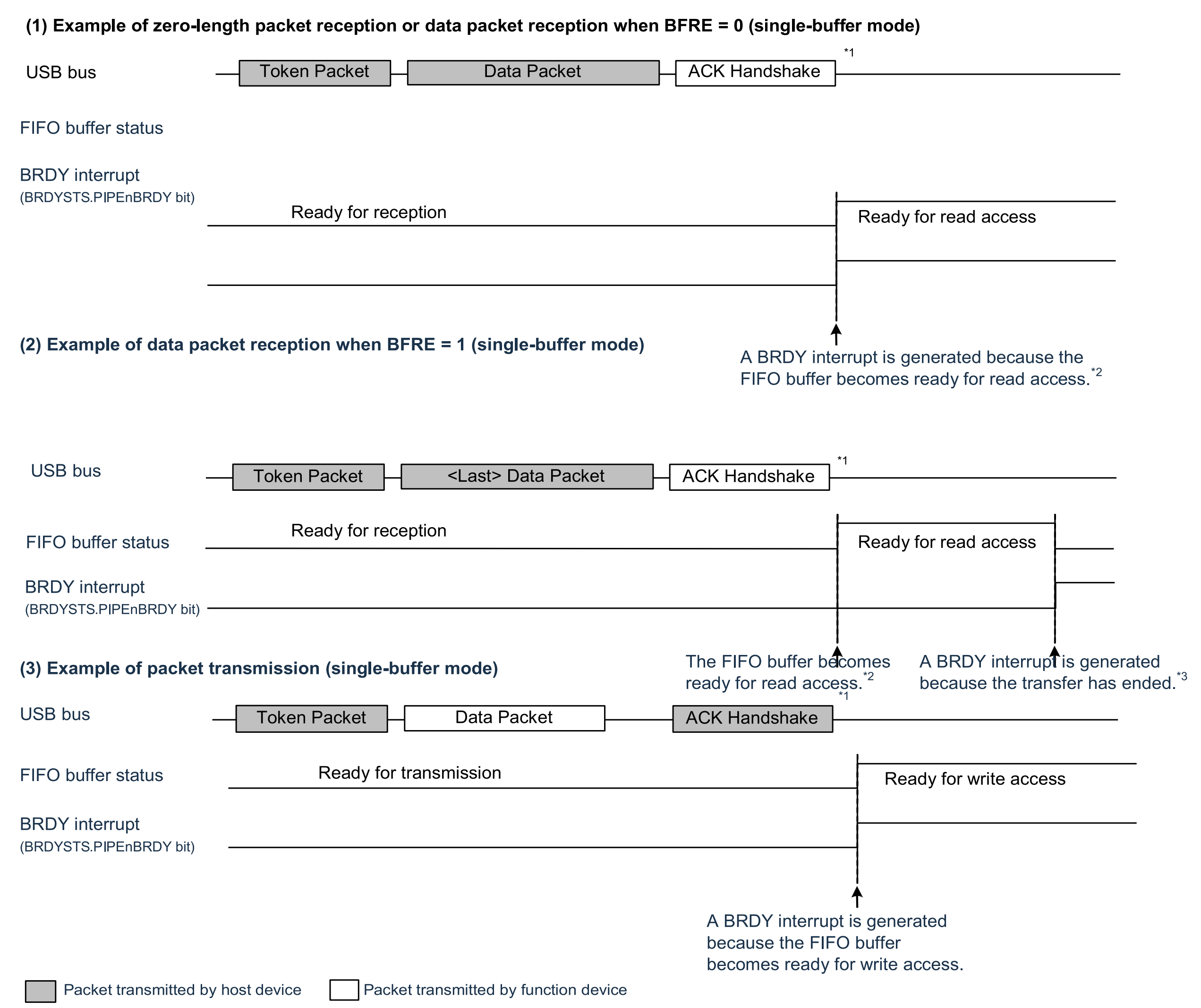
The BRDY interrupt status bit is set to 1 when the FIFO buffer is ready for write access, and are set to 0 when it is not ready. However, the BRDY interrupt is not generated even if the DCP in the transmitting direction is ready for write access.

b) For the receiving pipe

The BRDY interrupt status bit is set to 1 when the FIFO buffer is ready for read access, and are set to 0 when all data have been read (not ready for read access).

When a zero-length packet is received while the FIFO buffer is empty, the pertinent bit is set to 1 and the BRDY interrupt is continuously generated until BCLR = 1 is written through software. With this setting, the PIPEnBRDY cannot be set to 0 through software. When the SOFCFG.BRDYM bit is set to 1, all PIPECFG.BFRE bit (for all pipes) should be set to 0.

The timing for generating a BRDY interrupt is shown in Figure 22-12.


(1) Example of zero-length packet reception or data packet reception when BFRE = 0 (single-buffer mode)
USB bus
Token Packet
Data Packet
ACK Handshake
*1
FIFO buffer status
BRDY interrupt
(BRDYSTS.PIPEnBRDY bit)
Ready for reception
Ready for read access
A BRDY interrupt is generated because the FIFO buffer becomes ready for read access.*2
(2) Example of data packet reception when BFRE = 1 (single-buffer mode)
USB bus
Token Packet
<Last> Data Packet
ACK Handshake
*1
FIFO buffer status
Ready for reception
Ready for read access
BRDY interrupt
(BRDYSTS.PIPEnBRDY bit)
The FIFO buffer becomes ready for read access.*2
A BRDY interrupt is generated because the transfer has ended.*3
(3) Example of packet transmission (single-buffer mode)
USB bus
Token Packet
Data Packet
ACK Handshake
*1
FIFO buffer status
Ready for transmission
Ready for write access
BRDY interrupt
(BRDYSTS.PIPEnBRDY bit)
A BRDY interrupt is generated because the FIFO buffer becomes ready for write access.
Packet transmitted by host device
Packet transmitted by function device

Note 1. The ACK handshake is not used in isochronous transfers.

Note 2. The FIFO buffer becomes ready for read access under the following condition:
When a packet is received while no data remains unread in the FIFO buffer in the CPU.

Note 3. A transfer ends under either of the following conditions:
(1) When a short packet including a zero-length packet is received
(2) When the number of packets specified in the transaction counter are received

Figure 22-12 Timing of BRDY interrupt generation

The condition for clearing the INTSTS0.BDAY bit depends on the SOFCFG.BRDYM bit setting, see Table 22-15.

Table 22-15 **Condition for clearing BRDY bit**

BRDYM bit	Condition for clearing BRDY bit
0	When all bits in BRDYSTS are set to 0 by software
1	When the BSTS bit becomes 0 for all pipes

22.3.3.2 NRDY interrupt

On generating an internal NRDY interrupt request for the pipe whose PID[1:0] bits are set to BUF by software, the USBFS sets the corresponding NRDYSTS.PIPEnNRDY bit to 1. If the corresponding bit in the NRDYENB register has been set to 1 by software, the USBFS sets the INTSTS0.NRDY flag to 1 and generates a USBFS interrupt.

This section describes the conditions under which USBFS generates an internal NRDY interrupt request for a given pipe.

During a setup transaction in host controller mode, no internal NRDY interrupt request is generated. A SACK or SIGN interrupt will be detected during a setup transaction in host controller mode.

No internal NRDY interrupt request is generated during the status phase of a control transfer performed in device controller mode.

1) Host controller mode

a) For the transmitting pipe

On any of the following conditions, the USBFS detects an NRDY interrupt:

- For the pipe for isochronous transfers, when the time to issue an OUT token comes while there is no data to be transmitted in the FIFO buffer. In this case, the USBFS transmits a zero-length packet following the OUT token and sets the bit corresponding to the NRDYSTS.PIPEnNRDY bit and the FRMNUM.OVRN bit to 1.
- During communications other than setup transactions using the pipe for the transfers other than isochronous transfers, when any combination of the following two cases occur three consecutive times:
- No response is returned from the peripheral device (when timeout is detected before detection of the handshake packet from the peripheral device)
- An error is detected in the packet from the peripheral device. In this case, the USBFS sets the bit corresponding to the PIPEnNRDY bit to 1 and modifies the setting of the PID[1:0] bits of the corresponding pipe to NAK.
- During communications other than setup transactions, when the STALL handshake is received from the peripheral device. In this case, the USBFS sets the bit corresponding to the PIPEnNRDY flag to 1 and modifies the setting of the PID[1:0] bits of the corresponding pipe to STALL (11b).

b) For the receiving pipe

- For the pipe for isochronous transfers, when the time to issue an IN token comes while there is no space available in the FIFO buffer. In this case, the USBFS discards the received data for the IN token and sets the PIPEnNRDYbit corresponding to the pipe and the OVRN bit to 1. When a packet error is detected in the received data for the IN token, the USBFS also sets the FRMNUM.CRCE bit to 1.
- For the pipe for the transfers other than isochronous transfers, when any combination of the following two cases occur three consecutive times:
- No response is returned from the peripheral device for the IN token issued by the USBFS (when timeout is detected before detection of the DATA packet from the peripheral device)
- An error is detected in the packet from the peripheral device. In this case, the USBFS sets the PIPEnNRDY bit corresponding to the pipe to 1 and modifies the setting of the PID[1:0] bits of the

corresponding pipe to NAK.

- For the pipe for isochronous transfers, when no response is returned from the peripheral device for the IN token (when timeout is detected before detection of the DATA packet from the peripheral device) or an error is detected in the packet from the peripheral device. In this case, the USBFS sets the PIPEnNRDY flag corresponding to the pipe to 1. The setting of the PID[1:0] bits of the pipe is not modified.
- For the pipe for isochronous transfers, when a CRC error or a bit stuffing error is detected in the received data packet. In this case, the USBFS sets the PIPEnNRDY flag corresponding to the pipe and the CRCE flag to 1.
- When the STALL handshake is received. In this case, the USBFS sets the PIPEnNRDY flag corresponding to the pipe to 1 and modifies the setting of the PID[1:0] bits of the corresponding pipe to STALL.

2) Device controller mde

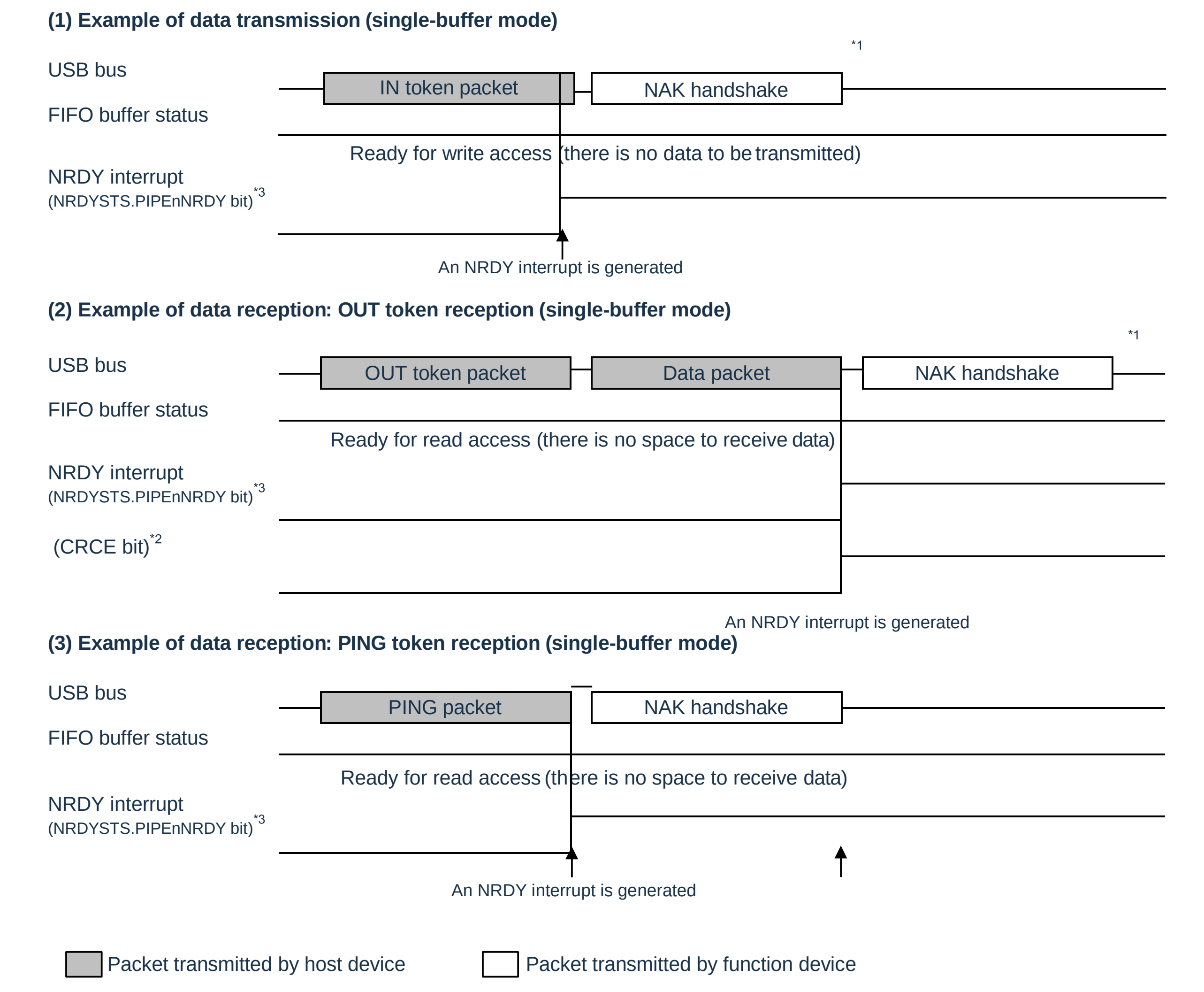
a) For the transmitting pipe

- When an IN token is received while there is no data to be transmitted in the FIFO buffer. In this case, the USBFS generates a NRDY interrupt request at the reception of the IN token and sets the NRDYSTS.PIPEnNRDY flag to 1. For the pipe for the isochronous transfers in which an interrupt is generated, the USBFS transmits a zero-length packet and sets the FRMNUM.OVRN bit to 1.

b) For the receiving pipe

- When an OUT token is received while there is no space available in the FIFO buffer. For the pipe for the isochronous transfers in which an interrupt is generated, the USBFS generates a NRDY interrupt request at the reception of the OUT token and sets the PIPEnNRDY flag and OVRN flag to 1. For the pipe for the transfers other than isochronous transfers in which an interrupt is generated, the USBFS generates a NRDY interrupt request when a NAK handshake is transferred after the data following the OUT token is received, and sets the PIPEnNRDY flag to 1. However, during re-transmission (due to DATA-PID mismatch), the NRDY interrupt request is not generated. In addition, if an error occurs in the DATA packet, the NRDY interrupt request is not generated.
- For the pipe for isochronous transfers, when a token is not received successfully within an interval frame. In this case, the USBFS generates a NRDY interrupt request when SOF is received, and sets the PIPEnNRDY flag to 1.

Figure 22-13 shows the timing of NRDY interrupt generation when the device controller is selected.


(1) Example of data transmission (single-buffer mode)
USB bus
FIFO buffer status
NRDY interrupt
(NRDYSTS.PIPEnNRDY bit)*3
IN token packet
NAK handshake
*1
Ready for write access (there is no data to be transmitted)
An NRDY interrupt is generated
(2) Example of data reception: OUT token reception (single-buffer mode)
USB bus
FIFO buffer status
NRDY interrupt
(NRDYSTS.PIPEnNRDY bit)*3
(CRCE bit)*2
OUT token packet
Data packet
NAK handshake
*1
Ready for read access (there is no space to receive data)
An NRDY interrupt is generated
(3) Example of data reception: PING token reception (single-buffer mode)
USB bus
FIFO buffer status
NRDY interrupt
(NRDYSTS.PIPEnNRDY bit)*3
PING packet
NAK handshake
Ready for read access (there is no space to receive data)
An NRDY interrupt is generated
Packet transmitted by host device
Packet transmitted by function device

Note 1. The handshake is not used in isochronous transfers.
Note 2. The CRCE and OVRN bits change only while the target pipe is set to isochronous transfers.
Note 3. The value of the PIPEnNRDY bit changes to 1 only when the PIPEnPID[1:0] bits are set to 01b (BUF response).

Figure 22-13 Timing for generating NRDY interrupts in device controller mode

22.3.3.3 BEMP interrupt

On detecting a BEMP interrupt for the pipe whose PID[1:0] bits are set to BUF by software, the USBFS sets the corresponding BEMPSTS.PIPEnBEMP flag to 1. If the corresponding bit in the BEMPENB register has been set to 1 by software, the USBFS sets the INTSTS0.BEMP flag to 1 and generates a USBFS interrupt. This section describes the conditions under which the USBFS generates an internal BEMP interrupt request.

1) For the transmitting pipe

When the FIFO buffer of the corresponding pipe is empty on completion of transmission (including zero-length packet transmission). In single buffer mode, an internal BEMP interrupt request is generated simultaneously with the BRDY interrupt for the pipe other than DCP.

However, the internal BEMP interrupt request is not generated on any of the following conditions:

- When the CPU or DMA has already started writing data to the FIFO buffer of the CPU on completion of transmitting data from one FIFO buffer in double buffer mode.
- When the buffer is cleared (emptied) by setting the PIPEnCTR.ACLRM or the BCLR bit in the port control register to 1.
- When IN transfer (zero-length packet transmission) is performed during the control transfer status stage while the device controller is selected.

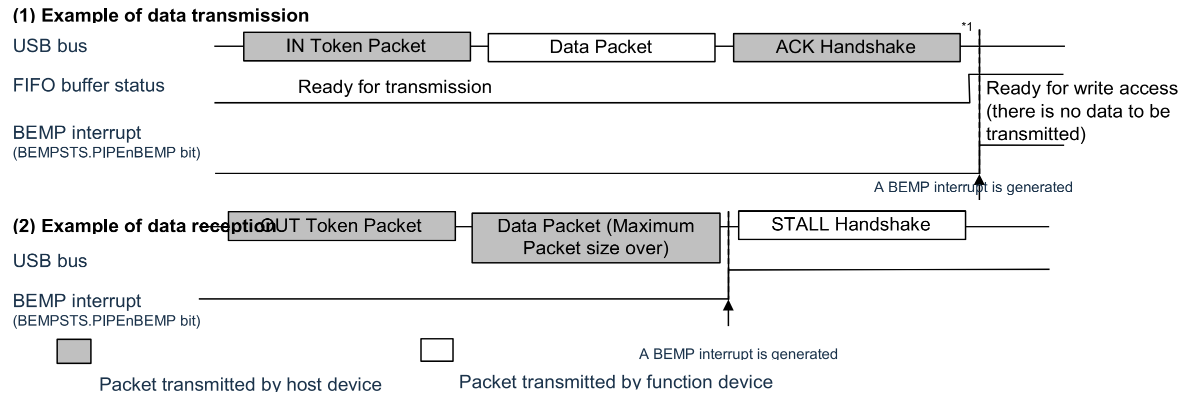
2) For the receiving pipe

When the successfully-received data packet size exceeds the specified maximum packet size. In this case, the USBFS generates a BEMP interrupt request, sets the corresponding BEMPSTS.PIPEnBEMP bit to 1, discards the received data, and modifies the setting of the PID[1:0] bits of the corresponding pipe to STALL (11b). Here, the USBFS returns no response when used as the host controller, and returns STALL response when used as the device controller.

However, the internal BEMP interrupt request is not generated on any of the following conditions:

- When a CRC error or a bit stuffing error is detected in the received data.
- When a setup transaction is being performed:
- Writing 0 to the BEMPSTS.PIPEnBEMP bit clears the status.
- Writing 1 to the BEMPSTS.PIPEnBEMP bit is invalid.

The timing for generating a BEMP interrupt in device controller mode is shown in Figure 22-14.



Note 1. The handshake is not used in isochronous transfers.

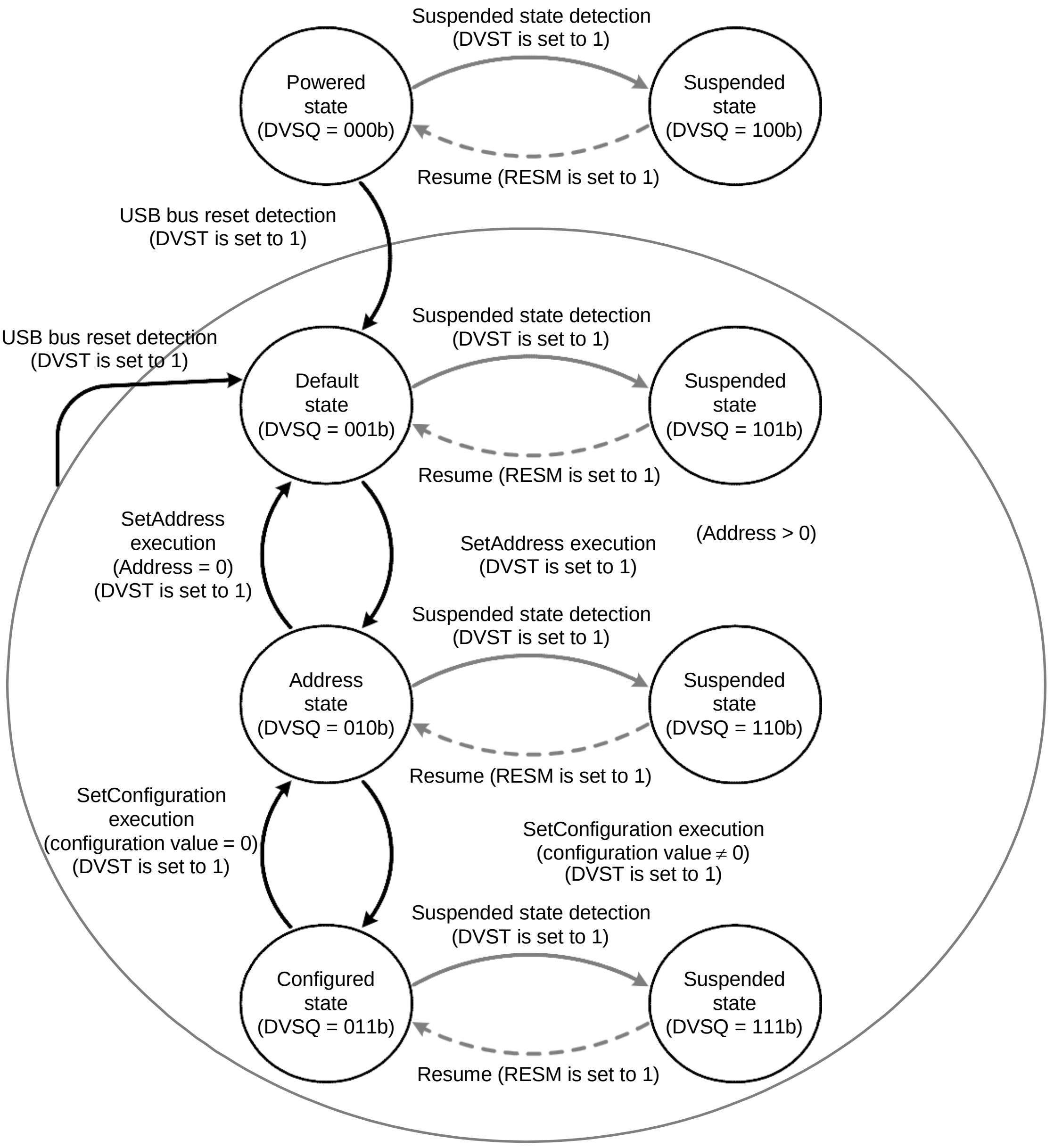
Figure 22-14 Timing for generating BEMP interrupts in device controller mode

22.3.3.4 Device state transition interrupt (device controller mode)

Figure 22-15 is a diagram of device state transitions in the USBFS. The USBFS controls device state and generates device state transition interrupts. However, recovery from the suspended state (resume signal detection) is detected by means of the resume interrupt. The device state transition interrupts can be enabled or disabled individually using INTENB0. The device state to which a transition was made can be confirmed using the INTSTS0.DVSQ[2:0] bits.

When transitioning to the default state, a device state transition interrupt is generated after a USB bus reset is detected.

The USBFS controls the device state and can only generate device state transition interrupts in device controller mode.



Note For a transition indicated by a solid line, the DVST bit is set to 1.
For a resume indicated by a dashed line, the RESM bit is set to 1.

Figure 22-15 Device state transition

22.3.3.5 Control transfer stage transition interrupt (device controller mode)

Figure 22-16 is a diagram of control transfer stage transitions in the USBFS. The USBFS controls the control transfer sequence and generates control transfer stage transition interrupts. The control transfer stage transition interrupts can be enabled or disabled individually using INTENB0. The transfer stage to which a transition was made can be confirmed using the INTSTS0.CTSQ[2:0] bits.

Control transfer stage transition interrupts are generated only when the device controller is selected. The control transfer sequence errors are listed below. If an error occurs, the DCPCTR.PID[1:0] bits are set to 1xb (STALL response).

1) Control read transfer error
   - An OUT token is received while no data has been transferred for the IN token at the data stage.
   - An IN token is received at the status stage.
   - A data packet with DATAPID = DATA0 is received at the status stage.
2) Control write transfer error
   - An IN token is received while no ACK response has been returned for the OUT token at the data stage.
   - A data packet with DATAPID = DATA0 is received for the first data packet at the data stage.
   - An OUT token is received at the status stage
3) Control write no data transfer error
   - An OUT token is received at the status stage.

   At the control write transfer data stage, if the number of receive data exceeds the wLength value of the USB request, it cannot be recognized as a control transfer sequence error. At the control read transfer status stage, packets other than zero-length packets are received by an ACK response and the transfer ends normally.

When a CTRT interrupt occurs in response to a sequence error (INTSTS0.CTRT = 1), CTSQ[2:0] = 110b value is retained until the CTRT flag = 0 is written (the interrupt status is cleared). Therefore, while CTSQ[2:0]= 110b is being held, the CTRT interrupt that ends the setup stage will not be generated even if a new USB request is received. The USBFS retains the setup stage end, and after the interrupt status has been cleared by software, a CTRT interrupt is generated.

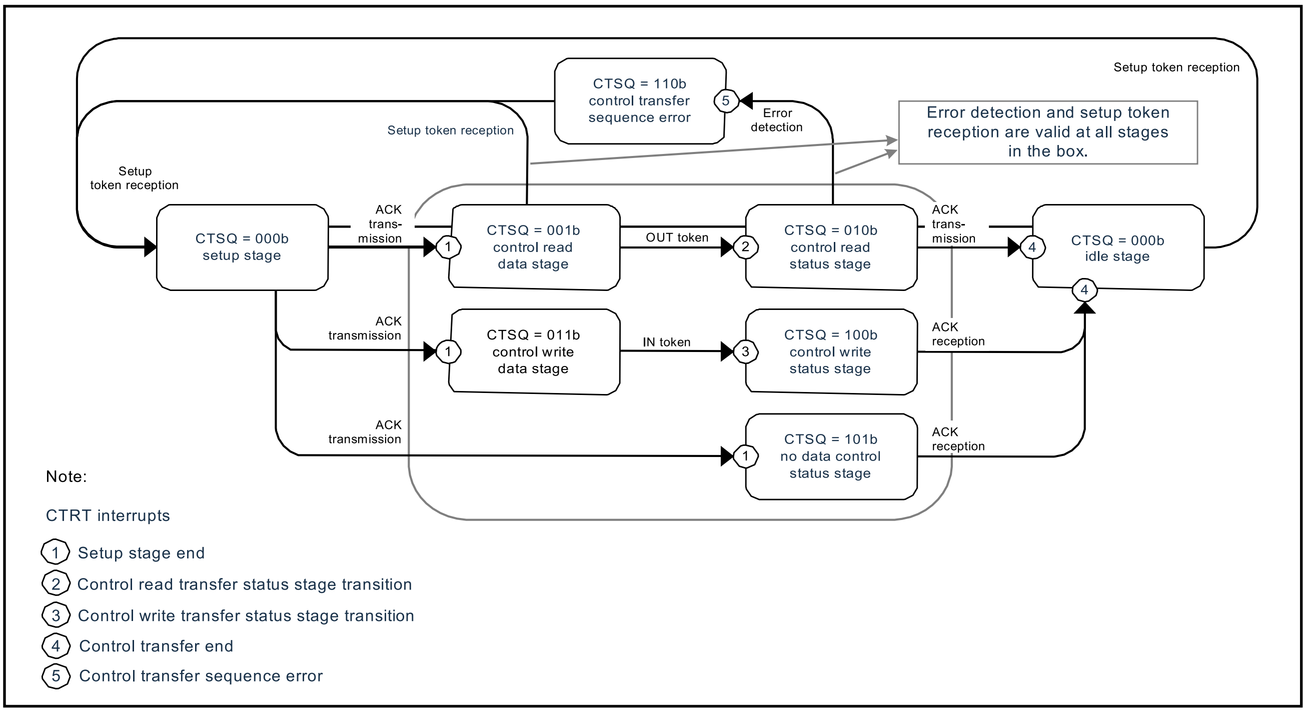


Figure 22-16 Control transfer stage transitions

22.3.3.6 Frame number update interrupt

With the host controller selected, an interrupt is generated at the timing when the frame number is updated.

In device controller mode, a SOFR interrupt is generated when the frame number is updated. If the USBFS detects a new SOF packet during full speed operation, the frame number is updated and a SOFR interrupt is generated.

22.3.3.7 VBUS interrupt

When the USB_VBUS pin level changes, a VBUS interrupt is generated. The level of the USB_VBUS pin can be checked with the INTSTS0.VBSTS bit. Whether the host controller is connected or disconnected can be confirmed using the VBUS interrupt. However, if the system is activated with the host controller connected, the first VBUS interrupt is not generated because there is no change in the USB_VBUS pin level.

22.3.3.8 Resume interrupt

When the device controller is selected, a resume interrupt is generated when the device state is the suspended state and the USB bus state has changed (from J-state to K-state, or from J-state to SE0). Recovery from the suspended state is detected by means of the resume interrupt.

When the host controller is selected, no resume interrupt is generated. Use the BCHG interrupt to detect a change in the USB bus state.

22.3.3.9 OVRCR interrupt

An OVRCR interrupt is generated when the USB_OVRCURA or USB_OVRCURB pin level has changed. The levels of the USB_OVRCURA and USB_OVRCURB pins can be checked with the SYSSTS0.OVCMON[1:0] bits. The external power supply IC can check whether overcurrent has been detected using the OVRCR interrupt.

For OTG connection, whether a change has been detected in the VBUS comparator can be checked using the OVRCR interrupt.

22.3.3.10 BCHG interrupt

A BCHG interrupt is generated when the USB bus state has changed. The BCHG interrupt can be used to detect whether the peripheral device is connected and can also be used to detect a remote wakeup when the host controller is selected. The BCHG interrupt is generated regardless of whether the host controller or device controller is selected.

22.3.3.11 DTCH interrupt

A DTCH interrupt is generated when disconnection of the USB bus is detected while the host controller is selected. The USBFS detects bus disconnection based on USB Specification 2.0.

All pipes in which communications are currently carried out for the pertinent port should be terminated by software and make a transition to the wait state for bus connection to the pertinent port (wait state for ATTCH interrupt generation).

Regardless of the value of the associated interrupt enable bit setting, the USBFS hardware will:

- Modifies the DVSTCTR0.UACT bit for the port in which a DTCH interrupt has been detected to 0.
- Puts the port in which a DTCH interrupt has been generated into the idle state.

22.3.3.12 SACK interrupt

A SACK interrupt is generated when an ACK response for the transmitted setup packet has been received from the peripheral device with the host controller selected. The SACK interrupt can be used to confirm that the setup transaction has been completed successfully.

22.3.3.13 SIGN interrupt

A SIGN interrupt is generated when an ACK response for the transmitted setup packet has not been correctly received from the peripheral device three consecutive times with the host controller selected. The SIGN interrupt can be used to detect no ACK response transmitted from the peripheral device or corruption of an ACK packet.

22.3.3.14 ATTCH interrupt

An ATTCH interrupt is generated when J-state or K-state of the full-speed signal level is detected on the USB port for 2.5 μs with the host controller selected. To be more specific, an ATTCH interrupt is detected on any of the following conditions:

- When K-state, SE0, or SE1 changes to J-state, and J-state continues 2.5 μs.
- When J-state, SE0, or SE1 changes to K-state, and K-state continues 2.5 μs.

22.3.3.15 EOFERR interrupt

An EOFERR interrupt occurs when the USBFS detects that communication has not been completed by the EOF2 timing defined in the USB 2.0 specification.

When interrupt detection is performed, all pipes in which communications are currently carried out for the pertinent port should be terminated by software and perform re-enumeration of the pertinent port. Regardless of the value of the associated interrupt enable bit setting. the USBFS hardware:

- Modifies the DVSTCTR0.UACT bit for the port in which an EOFERR interrupt has been detected to 0.
- Puts the port in which an EOFERR interrupt has been generated into the idle state.

22.3.3.16 Portable device detection interrupt

A portable device detection interrupt is generated when the USBFS detects a level change (high to low or low to high) in the PDDET output from the USB-PHY. When a portable device detection interrupt is generated, use software to repeat reading the PDDETSTS0 bit until the same value is read three or more times, and perform debouncing.

22.3.4 Pipe control

Table 22-16 lists the pipe settings for the USBFS. With USB data transfer, data transfer is carried out using the pipe that the software has associated with the endpoint. The USBFS has ten pipes that are used for data transfer. Appropriate settings should be made for each of the pipes according to the specifications of the system.

Table 22-16 Pipe settings

Register name	Bit name	Setting	Remark
DCPCFG PIPECFG	TYPE	Transfer type	Pipe 1~9 can be set.
	BFRE	BRDY interrupt mode	Pipe 1~5 can be set.
	DBLB	Double buffer selection	Pipe 1~5 can be set.
	DIR	Transfer direction selection	IN or OUT can be set.
	EPNUM	Endpoint number	Pipe 1~9 can be set. A value other than 0000b should be set when the pipe is used.
	SHTNAK	Selection of disabled state for pipe when transfer ends	Pipe 1 and 2: limited to bulk transfer. Pipe 3~5: can be set.
DCPMAXP PIPEMAXP	DEVSEL	Device selection	Used only in host controller mode.
	MXPS	Maximum packet size	Compliant with USB Specification 2.0.
PIPEPERI	IFIS	Buffer flush	Pipe 1 and 2: Limited to isochronous transfer. Pipe 3~9: can not be set.
	IITV	Interval counter	Pipe 1 and 2: Limited to isochronous transfer. Pipe 3~5: can not be set. Pipe 6~9: Used only in host controller mode.
DCPCTR PIPEnCTR	BSTS	Buffer status	For the DCP, receive buffer status and transmit buffer status are switched with the ISEL bit.
	INBUFM	IN buffer monitor	Available only for pipe1 to pipe5.
	SUREQ	SETUP request	Can be set only for the DCP. Can be controlled only in host controller mode.
	SUREQCLR	SUREQ clear	Can be set only for the DCP. Can be controlled only in host controller mode.
	ATREPM	Auto response mode	Pipe 1~5: Can be set only in device controller moe.
	ACLRM	Auto buffer clear	Pipe 1~9 can be set.
	SQCLR	Sequence clear	Clears the data toggle bit.
	SQSET	Sequence set	Sets the data toggle bit.
	SQMON	Sequence monitor	Monitors the data toggle bit.
	PBUSY	Pipe busy status	-
	PID	Response PID	See 22.3.4.6 Response PID.
PIPEnTRE	TRENB	Transaction counter enable	Pipe 1~5 can be set.
	TRCLR	Current transaction counter clear	Pipe 1~5 can be set.
PIPEnTRN	TRNCNT	Transaction counter	Pipe 1~5 can be set.

22.3.4.1 Pipe control register switching procedures

The following bits in the pipe control registers can be modified only when USB communication is prohibited (PID= NAK).

The following shows the registers and bits that should not be modified when USB communication is enabled (PID= BUF):

- Bits in the DCPCFG and DCPMAXP registers
- The SQCLR and SQSET bits in the DCPCTR register
- Bits in registers PIPECFG, PIPEMAXP, and PIPEPERI
- The ATREPM, ACLRM, SQCLR, and SQSET bits in the PIPEnCTR register

In order to modify the above bits in the USB communication enabled (PID= BUF) state, follow the procedure shown below:

1. A request to modify bits in the pipe control register occurs.
2. Modify the PID[1:0] bits corresponding to the pipe to NAK.
3. Wait until the corresponding PBUSY flag is set to 0.
4. Modify the bits in the pipe control register.

The following bits in the pipe control registers can be modified only when the pertinent pipe information has not been set by the CURPIPE[3:0] bits in registers CFIFOSEL, D0FIFOSEL, and D1FIFOSEL.

Registers that should not be set when the CURPIPE[3:0] bits are set:

- Bits in the DCPCFG and DCPMAXP register
- Bits in registers PIPECFG, PIPEMAXP and PIPEPERI

In order to modify pipe information, the CURPIPE[3:0] bits in the port select registers should be set to a pipe other than the pipe to be modified. For the DCP, the buffer should be cleared using the BCLR bit in the port control register after the pipe information is modified.

22.3.4.2 Transfer types

The PIPECFG.TYPE[1:0] bits are used to specify the transfer type for each pipe. The transfer types that can be set for the pipes are as follows.

- DCP: No setting is necessary (fixed at control transfer).
- Pipe 1 and 2: These should be set to bulk transfer or isochronous transfer.
- Pipe 3~5: These should be set to bulk transfer.
- Pipe 6~9: These should be set to interrupt transfer.

22.3.4.3 Endpoint number

The PIPECFG.EPNUM[3:0] bits are used to set the endpoint number for each pipe. The DCP is fixed at endpoint 0. The other pipes can be set from endpoint 1 to endpoint 15.

- DCP: No setting is necessary (fixed at endpoint 0).
- Pipe 1~9: The endpoint numbers from 1 to 15 should be selected and set. These should be set so that the combination of the PIPECFG.DIR bit and EPNUM[3:0] bits is unique

22.3.4.4 Maximum packet size setting

The DCPMAXP.MXPS[6:0] bits and the PIPEMAXP.MXPS[8:0] bits are used to specify the maximum packet size for each pipe. DCP and pipe 1~5 can be set to any of the maximum pipe sizes defined by USB Specification 2.0. For pipe 6~9, 64 bytes are the upper limit of the maximum packet size. The maximum packet size should be set before beginning the transfer (PID=BUF):

- DCP: Set 8, 16, 32, or 64.
- Pipe 1~5: Set 8, 16, 32, or 64 when using bulk transfer.
- Pipe 1 and 2: Set a value between 1 and 256 when using isochronous transfer.
- Pipe 6~9: Set a value between 1 and 64.

22.3.4.5 Transaction counter for pipes 1~5 in reading direction

The USBFS recognizes that a transmission has ended when a specified number of transactions have been completed in the packet receiving direction. There are two transaction counters:

- PIPEnTRN register specifying the number of transactions to be executed T
- - Current counter for internally counting the number of executed transactions.

With the PIPECFG.SHTNAK bit set to 1, when the current counter value matches the specified number of transactions, the corresponding PIPEnCTR.PID[1:0] bits are set to 00b (NAK) and the subsequent transfer is disabled. The transactions can be counted again from the beginning by initializing the current counter of the transaction counter function through the PIPEnTRE.TRCLR bit. The information read from PIPEnTRN differs depending on the setting of the PIPEnTRE.TRENB bit.

- TRENB=0: The specified transaction counter value can be read.
- TRENB=1: The current counter value indicating the internally counted number of executed transactions can be read.

When operating the TRCLR bit, the following should be noted:

- If the transactions are being counted and PID= BUF, the current counter cannot be cleared.
- If there is any data left in the buffer, the current counter cannot be cleared.

22.3.4.6 Response PID

The PID[1:0] bits in the DCPCTR and PIPEnCTR registers are used to set the response PID for each pipe. The following shows the USBFS operation with various response PID settings:

1) Response PID settings when the host controller is selected:

The response PID is used to specify the execution of transactions.

- NAK setting: Using pipes is disabled. No transaction is executed.
- BUF setting: Transactions are executed based on the status of the FIFO buffer.
- For OUT direction: If there are transmit data in the FIFO buffer, an OUT token is issued.
- For IN direction: If there is an area to receive data in the FIFO buffer, an IN token is issued.
- STALL setting: Using pipes is disabled. No transaction is executed.

Note: Setup transactions for the DCP are set with the DCPCTR.SUREQ bit.

2) Response PID settings when the device controller is selected:

The response PID is used to specify the response to transactions from the host.

- NAK setting: The NAK response is returned in response to the generated transaction.
- BUF setting: Responses are made to transactions according to the status of the FIFO buffer.
- STALL setting: The STALL response is returned in response to the generated transaction.

Note: For setup transactions, an ACK response is always returned regardless of the PID[1:0] setting and the USB request is stored in the register.

Sections (3) and (4) describe situations where the USBFS writes to the PID[1:0] bits as a result of a specific transaction.

3) Hardware response PID setting in host controller mode

- NAK setting: In the following cases, PID= NAK is set and issuing of tokens is automatically stopped:
- When a transfer other than isochronous transfer has been performed and an NRDY interrupt is generated. (For details, refer to section 22.3.3.2 NRDY interrupt)
- If a short packet is received when the PIPECFG.SHTNAK bit has been set to 1 for bulk transfer.
- If the transaction counting ends when the SHTNAK bit has been set to 1 for bulk transfer.
- BUF setting: This setting will not be written by the USBFS.
- STALL setting: In the following cases, PID = STALL is set and issuing of tokens is automatically stopped:
- When STALL is received in response to the transmitted token.
- When the size of the receive data packet exceeds the maximum packet size.

4) Hardware response PID setting in device controller mode

- NAK setting: In the following cases, PID = NAK is set and NAK is returned in response to transactions:
- When the SETUP token is received normally (DCP only).
- If the transaction counting ends or a short packet is received when the PIPECFG.SHTNAK bit has been set to 1 for bulk transfer.
- BUF setting: There is no BUF writing by the USBFS.
- STALL setting: In the following cases, PID = STALL is set and STALL is returned in response to transactions:

- When a maximum packet size exceeded error is detected in the received data packet.
- When a control transfer sequence error has been detected (DCP only).

22.3.4.7 Data PID sequence bit

The USBFS automatically toggles the sequence bit in the data PID when data is transferred successfully in the control transfer data stage, bulk transfer, and interrupt transfer. The sequence bit of the next data PID to be transmitted can be confirmed with the SQMON flag in the DCPCTR and PIPEnCTR registers. When data is transmitted, the sequence bit switches at the timing of ACK handshake reception. When data is received, the sequence bit switches at the timing of ACK handshake transmission. The DCPCTR bit and the PIPEnCTR bit can be used to change the data PID sequence bit.

When the device controller has been selected and control transfer is used, the USBFS automatically sets the sequence bit when a stage transition is made. DATA0 is returned when the setup stage is ended. PID = DATA1 is returned in a status stage. Therefore, software settings are not required. However, when the host controller has been selected and control transfer is used, the sequence bit should be set by software at a stage transition.

For the ClearFeature request transmission or reception, the data PID sequence bit should be set by software regardless of whether the host controller or device controller is selected.

22.3.4.8 Response PID = NAK function

The USBFS has a function that disables pipe operation (response PID = NAK) at the timing at which the final data packet of a transaction is received (the USBFS automatically distinguishes this based on reception of a short packet or the transaction counter) by setting the PIPECFG.SHTNAK bit to 1.

When the double buffer mode is being used for the buffer memory, using this function enables reception of data packets in transfer units. If pipe operation has been disabled, software should set the pipe to the enabled state again (response PID= BUF).

The response PID = NAK function can be used only when bulk transfers are used.

22.3.4.9 Auto response mode

With the pipes for bulk transfer (pipe1~5), when the PIPEnCTR.ATREPM bit is set to 1, a transition is made to auto response mode. During an OUT transfer (the PIPECFG.DIR bit = 0), OUT-NAK mode is entered, and during an IN transfer (the DIR bit = 1), null auto response mode is entered.

22.3.4.10 OUT-NAK mode

With the pipes for bulk OUT transfer, NAK is returned in response to an OUT token and an NRDY interrupt is output when the PIPEnCTR.ATREPM bit is set to 1. To make a transition from normal mode to OUT-NAK mode, OUT-NAK mode should be specified in the pipe operation disabled state (For NAK response, PID[1:0]=00b) before enabling pipe operation (For BUF response, PID[1:0]=01b). After pipe operation has been enabled, OUT-NAK mode becomes valid. However, if an OUT token is received immediately before pipe operation is disabled, the token data is normally received, and an ACK is returned to the host.

To make a transition from OUT-NAK mode to normal mode, OUT-NAK mode should be canceled in the pipe operation disabled state (For NAK response, PID[1:0]=00b) before enabling pipe operation. In normal mode, reception of OUT data is enabled.

22.3.4.11 Null auto response mode

With the pipes for bulk IN transfer, zero-length packets are continuously transmitted when the PIPEnCTR.ATREPM bit is set to 1.

To make a transition from normal mode to null auto response mode, null auto response mode should be set in the pipe operation disabled state (response PID = NAK). After pipe operation has been enabled, null auto response mode becomes valid. Before setting null auto response mode, the PIPEnCTR.INBUFM = 0 should be confirmed because the mode can be set only when the buffer is empty. If the INBUFM flag is 1, the buffer should be emptied with the PIPEnCTR.ACLRM bit. While a transition to null auto response mode is being made, data should not be written from the FIFO port.

To make a transition from null auto response mode to normal mode, pipe operation disabled state (response PID = NAK) should be retained for the period of zero-length packet transmission (about 10 µs) before canceling null auto response mode. In normal mode, data can be written from the FIFO port; therefore, packet transmission to the host is enabled by enabling pipe operation (response PID = BUF).

22.3.5 FIFO buffer memory

22.3.5.1 FIFO buffer memory

The USBFS has FIFO buffer memory for data transfer. The memory area used for each pipe is managed by the USBFS. The FIFO buffer memory has two states depending on whether the access right is assigned to the system (CPU side) or the USBFS (SIE side).

1) Buffer status

Table 22-17 and Table 22-18 show the buffer status in the USBFS. The buffer memory status can be confirmed using the DCPCTR.BSTS bit and the PIPEnCTR.INBUFM bit. The transfer direction for the FIFO buffer can be specified using either the PIPECFG.DIR bit or the CFIFOSEL.ISEL bit (when DCP is selected). The INBUFM flag is valid for pipe0~5 in transmitting.

When a transmitting pipe uses the double buffer configuration, software can read the BSTS flag to monitor the FIFO buffer status on the CPU side and the INBUFM bit to monitor the FIFO buffer status on the SIE side. When the BEMP interrupt may not show the buffer empty status because the write access to the FIFO port by the CPU or DMA is slow, software can use the INBUFM bit to confirm the end of transmission.

Table 22-17 Buffer status indicated by the BSTS bit

ISEL or DIR	BSTS	Buffer memory status
0(receiving direction)	0	There is no received data, or data is being received. Reading from the FIFO port is disabled.
0(receiving direction)	1	There is received data, or a zero-length packet has been received. Reading from the FIFO port is allowed. Note: when a zero-length packet is received, reading is not possible and the buffer must be cleared.
1(transmitting direction)	0	The transmission has not been completed. Writing to the FIFO port is disabled.
1(transmitting direction)	1	The transmission has been completed. CPU write is allowed.

Table 22-18 Buffer status indicated by the INBUFM bit

DIR	INBUFM	Buffer memory status
0(receiving direction)	Invalid	Invalid
1(transmitting direction)	0	The transmission has been completed. There is no waiting data to be transmitted.
1(transmitting direction)	1	The FIFO port has written data to the buffer. There is data to be transmitted.

2) FIFO buffer clearing

Table 22-19 shows the clearing methods of the FIFO buffer. The FIFO buffer can be cleared using the BCLR, DnFIFOSEL.DCLRM, and PIPEnCTR.ACLRM bit in the port control register.

Table 22-19 List of buffer clearing methods

FIFO buffer clearing methods	Clearing FIFO Buffer on CPU side	Mode for automatically clearing FIFO buffer after reading specified pipe data	Auto buffer clear mode for discarding all received packets
Register used	CFIFOCTR DnFIFOCTR	DnFIFOSEL	PIPEnCTR
Bit used	BCLR	DCLRM	ACLRM
Clearing condition	Cleared by writing 1	1: Mode valid 0: Mode invalid	1: Mode valid 0: Mode invalid

3) Auto buffer clear mode function

With the USBFS, all received data packets are discarded if the PIPEnCTR.ACLRM bit is set to 1. If a correct data packet has been received, the ACK response is returned to the host controller. The auto buffer clear mode function can be set only in the buffer memory reading direction.

If the ACLRM bit is set to 1 and then to 0, the FIFO buffer of the selected pipe can be cleared regardless of the access direction. An access cycle of at least 100 ns is required for the internal hardware sequence processing time between ACLRM = 1 and ACLRM = 0.

Single or double buffering can be selected for pipes 1~5 in the PIPECFG.DBLB bit.

22.3.5.2 FIFO port functions

Table 22-20 shows the settings for the FIFO port functions. In write access, writing data until the maximum packet size is reached automatically enables transmission of the data. To enable transmission before the maximum packet size is reached, the BVAL bit in the port control register should be set to end writing. To send a zero-length packet, the BCLR bit in the register should be used to clear the buffer and then the BVAL bit set in order to end writing.

In reading, reception of new packets is automatically enabled when all data has been read. Data cannot be read when a zero-length packet has been received (DTLN[8:0] = 0), so the BCLR bit in the register should be used to clear the buffer. The length of the receive data can be confirmed using the DTLN[8:0] bits in the port control register.

Table 22-20 FIFO port register setting

Register name	Bit name	Function
CFIFOSEL DnFIFOSEL	RCNT	Selects DTLN read mode.
	REW	FIFO buffer rewind (re-read, rewrite).
	DCLRM	Automatically clears receive data for a specified pipe after the data has been read (only for DnFIFO).
	DREQE	Enables DMA transfers (only for DnFIFO)
	MBW	Selects FIFO port access bit width
	BIGEND	Selects FIFO port endian.
	ISEL	Selects FIFO port access direction (only for DCP).
	CURPIPE	Selects the current pipe.
CFIFOCTR DnFIFOCTR	BVAL	Ends writing to the buffer memory.
	DCLR	Clears the buffer memory on the CPU side.
	DTLN	Checks the length of receive data.

1) FIFO port selection

Table 22-21 shows the pipes that can be selected with the various FIFO ports. The pipe to be accessed should be selected using the CURPIPE[3:0] bits in the port select register. After the pipe is selected, whether the written value can be correctly read from the CURPIPE[3:0] bits should be checked. If the previous pipe number is read, it indicates that the pipe modification is being executed by the USBFS controller. Access to the FIFO port is initiated by software confirmation that the FRDY bit of the port control register is 1.

In addition, the MBW bit must be set through software to specify the width of the bus to be accessed.The FIFO buffer access direction is set by PIPECFG.DIR. In DCP only, the access direction is determined by the ISEL bit.

Table 22-21 Accessing FIFO ports through pipes

Pipe	Access method	Port that can be used
DCP	CPU access	CFIFO port register
Pipe 1~9	CPU access	CFIFO port register D0FIFO/D1FIFO port register
	DMA access	D0FIFO/D1FIFO port register

2) REW bit

It is possible to temporarily stop access to the pipe currently being accessed, access a different pipe, and then continue processing for the current pipe again. The REW bit in the port select register is used for this processing.

If a pipe is selected through the CURPIPE[3:0] bits in the port select register with the REW bit set to 1, the pointer used for reading from and writing to the FIFO buffer is reset, and reading or writing can be carried out from the first byte. If a pipe is selected with 0 set for the REW bit, data can be read and written in continuation from the previous selection, without the pointer being reset.

Do not change the REW=1 and CURPIPE bit settings at the same time. Make sure FRDY=1 before setting REW=1.

22.3.6 DMA transfers (D0FIFO and D1FIFO ports)

1) DMA transfer overview

For pipes 1 to 9, the FIFO ports can be accessed using DMA. When a pipe buffer set in DMA can be accessed, a DMA transfer request will be output.

Use the DnFIFOSEL.MBW bit to select the transfer unit to the FIFO port, and the DnFIFOSEL.CURPIPE[3:0] bit to select the pipe for DMA transfers. Do not change the selected pipe during a DMA transfer.

2) DnFIFO auto-clear mode (D0FIFO and D1FIFO port read directions)

If the DnFIFOSEL.DCLRM bit is set to 1, the USBFS automatically clears the selected pipe's FIFO buffer when reading data from the FIFO buffer is complete.

Table 22-22 lists the relationship between packet reception and buffer memory clearing processing performed by software for each setting. As shown in the table, the buffer clearing conditions vary depending on the value set for the BFRE bit; however, even if the buffer needs to be cleared, there is no need to clear the buffer through software when using the DCLRM bit. DMA transfers can be performed without software.

DnFIFO auto clear mode can only be set in the FIFO buffer read direction.

Table 22-22 Correlation table for packet reception and buffer memory clearing processing by software

	Register setting			

Buffer state when receiving packets	DCLRM=0
	BFRE=0
		BFRE=1	DCLRM=1
			BFRE=0
				BFRE=1

Buffer full	No need to clear	No need to clear	No need to clear	No need to clear
Receive zero-length packets	Need to clear	Need to clear	No need to clear	No need to clear
Receive normal short packets	No need to clear	Need to clear	No need to clear	No need to clear
End of transaction counter	No need to clear	Need to clear	No need to clear	No need to clear

22.3.7 Control transfers (DCP)

In the data stage of control transfers, data is transferred using the default control pipe (DCP). The DCP FIFO buffer is a 64-byte single buffer and is a fixed area that is shared for both control reading and control writing. The FIFO buffer can be accessed only through the CFIFO port.

22.3.7.1 Control transfers in host controller mode

1) Setup stage

   Registers USQREQ, USBVAL, USBINDX, and USBLENG are the registers that are used to transmit a USB request for setup transactions. Writing setup packet data to the registers and writing 1 to the DCPCTR.SUREQ bit transmits the specified data for setup transactions. Upon completion of the transaction, the SUREQ bit is set to 0. The above USB request registers should not be modified while SUREQ = 1.

   After the attached state of the connected function device is detected, the first setup transaction for the device should be issued by using the sequence described above with the DCPMAXP.DEVSEL[3:0] bits set to 0 and the DEVADD0.USBSPD[1:0] bits set appropriately.

   After the connected function device is shifted to the Address state, setup transactions should be issued by using the sequence described above with the assigned USB address set in the DEVSEL[3:0] bits and the bits in the DEVADDn register corresponding to the specified USB address set appropriately. For example, when PIPEMAXP.DEVSEL[3:0] = 0010b, make appropriate settings in the DEVADD2 register; when PIPEMAXP.DEVSEL[3:0] = 0101b, make appropriate settings in the DEVADD5 register.

   When the setup transaction data has been sent, an interrupt request is generated according to the response received from the peripheral device (SIGN or SACK bit in the INTSTS1 register), by means of which the result of the setup transactions can be confirmed.

   A data packet of DATA0 (USB request) is transmitted as the data packet for a setup transaction regardless of the setting of the DCPCTR.SQMON bit.

2) Data stage

   Data stage is transferred using the DCP FIFO buffer.

   The access direction of the DCP FIFO buffer should be specified using the CFIFOSEL.ISEL bit. The transfer direction should be specified using the DCPCFG.DIR bit.

   For the first data packet of the data stage, the data PID should be transferred as DATA1. Set data PID = DATA1 in the DCPCTR.SQSET bit and the PID = BUF. Completion of data transfer is detected using the BRDY or BEMP interrupt.

   For control write transfers, when the number of data bytes to be sent is an integer multiple of the maximum packet size, software should control so as to send a zero-length packet at the end.

3) Status stage

   Zero-length packet data is transferred in the direction opposite to that in the data stage. As in the data stage, data is transferred using the DCP FIFO buffer. Transactions are done in the same manner as the data stage.

   For the data packets of the status stage, the data PID should be set to DATA1 using the DCPCTR.SQSET bit.

   For reception of a zero-length packet, the received data length should be confirmed using the CFIFOCTR.DTLN[8:0] bits after a BRDY interrupt is generated, and the FIFO buffer should then be cleared using the BCLR bit.

22.3.7.2 Control transmission in device controller mode

1) Setup stage

The USBFS sends an ACK response for a correct setup packet targeted to the USBFS. The operation of the USBFS in the setup stage is described below:

When receiving a new setup packet, the USBFS sets the following bits:

- Set the INTSTS0.VALID flag to 1.
- Set the DCPCTR.PID[1:0] bits to NAK.
- Set the DCPCTR.CCPL bit to 0.

When receiving a data packet right after the setup packet, the USBFS stores the USB request parameters in registers USBREQ, USBVAL, USBINDX, and USBLENG.

Response processing with respect to the control transfer should be carried out after setting the VALID = 0. In the VALID = 1 state, PID = BUF cannot be set, and the data stage cannot be terminated.

Using the function of the VALID flag, the USBFS can suspend the current request processing when receiving a new USB request during a control transfer, and can send a response to the newest request.

In addition, the USBFS automatically detects the direction bit (bit 8 of bmRequestType) and the request data length (wLength) of the received USB request, distinguishes between control read transfer, control write transfer, and no-data control transfer, and controls stage transitions. For a wrong sequence, the sequence error of the control transfer stage transition interrupt is generated, and the software is notified of occurrence of the error. For the stage control of the USBFS, refer to Figure 22-16.

2) Data stage

Data transfers corresponding to received USB requests should be done using the DCP. Before accessing the DCP FIFO buffer, the access direction should be specified using the CFIFOSEL.ISEL bit.

If the transfer data is larger than the size of the DCP FIFO buffer, the data transfer should be carried out using the BRDY interrupt for control write transfers and the BEMP interrupt for control read transfers.

3) Status stage

Control transfers are terminated by setting the DCPCTR.CCPL bit to 1 while the DCPCTR.PID[1:0] bits are set to BUF.

After the above settings have been made, the USBFS automatically executes the status stage in accordance with the data transfer direction determined at the setup stage. The specific procedure is as follows:

- For control read transfers

The USBFS receives zero-length packets from the USB host and sends an ACK response.

- For control write transfers and no-data control transfers

USBFS sends zero-length packets and receives ACK responses from the USB host.

4) Control transfer auto response function

The USBFS automatically responds to a correct SET_ADDRESS request. If any of the following errors occurs in the SET_ADDRESS request, a response from the software is necessary.

- bmRequestType is not 00h: Any transfer other than a control write transfer
- wIndex is not 00h: Request error

- wLength is not 00h: Any transfer other than a no-data control transfer
- wValue is larger than 7Fh: Request error
- INTSTS0.DVSQ [2:0] are 011b (Configured state): Control transfer of a device state error

For all requests other than the SET_ADDRESS request, a response is required from the corresponding software.

22.3.8 Bulk transfers (pipes 1 to 5)

The buffer memory usage (single/double buffer setting) can be selected for bulk transfers.

The USB provides the following functions for bulk transfers.

- BRDY interrupt function (PIPECFG.BFRE bit), refer to section 22.3.3.1, (2) When the SOFCFG.BRDYM = 0 and the PIPECFG.BFRE = 1
- Transaction count function (PIPEnTRE.TRENB, TRCLR, and PIPEnTRN.TRNCNT[15:0] bits: refer to section 22.3.4.5 Transaction counter for pipes 1~5 in reading direction)
- Response PID = NAK function (PIPECFG.SHTNAK bit), refer to section 22.3.4.8 Response PID = NAK function.
- Auto response mode (PIPEnCTR.ATREPM bit), refer to section 22.3.4.9 Auto response mode.

22.3.9 Interrupt transfers (pipes 6 to 9)

In device controller mode, the USBFS performs interrupt transfers according to the timing indicated by the host controller. In host controller mode, software can use an interval counter to set the time for issuing tokens.

22.3.9.1 Interval counter during interrupt transfers in host controller mode

For interrupt transfers, intervals between transactions are set in the PIPEPERI.IITV[2:0] bits. This controller issues interrupt transfer tokens based on the specified intervals.

1) Counter initialization

   The USBFS initializes the interval counter under the following conditions:

   Power-on reset

   Initialize the IITV[2:0] bits.

   - Initialize the FIFO buffer using the PIPEnCTR.ACLRM bit:

     This does not initialize the IITV [2:0] bits, but does initialize the count value. Setting the PIPEnCTR.ACLRM bit to 0 starts the count from the value set in IITV [2:0].

   The interval counter is not initialized in the following cases:

   - USB bus reset or USB suspended:

     The IITV[2:0] bits are not initialized. Setting the DVSTCTR0.UACT bit to 1 will start counting from the value saved before entering the USB bus reset state or the USB suspended state.

2) Operation when transmission/reception is impossible at token issuance timing

   In the following cases, the token will not be generated even at the token generation moment. In this case, it will try to execute the transaction in the next interval.

   - When the PID is set to NAK or STALL
   - When the FIFO buffer is full at the moment of token sending in the receive (IN) direction
   - When there is no data to be sent in the FIFO buffer at the moment the token is sent in the transmitting (OUT) direction.

22.3.10 Isochronous transfers (pipe1 and pipe2)

The USBFS has the following functions for isochronous transfers:

- Notification of isochronous transfer error information
- Interval counter (specified by the PIPEPERI.IITV[2:0] bits)
- Isochronous IN transfer data setup control (IDLY function)
- Isochronous IN transfer buffer flush function (specified by the PIPEPERI.IFIS bit)

22.3.10.1 Error detection in isochronous transfers

The USBFS has a function for detecting the error information described below, so that when errors occur in isochronous transfers, they can be controlled by software. Table 22-23 and Table 22-24 show the priority in which errors are confirmed and the interrupts generated corresponding to errors detected by the USBFS.

a) PID errors

- If the PID of the received packet is invalid.

b) CRC errors and bit stuffing errors

- If an error occurs in the CRC of the received packet or the bit stuffing is invalid

c) Maximum packet size exceeded

- The data of the received packet is larger than the specified maximum packet size.

d) Overrun and underrun errors

In host controller mode:

- When the FIFO buffer is full at the token sending timing in the IN (receiving) direction.
- When there is no data to be sent in the buffer memory at the token sending timing in the OUT (transmitting) direction.

In device controller mode:

- When there is no data to be sent in the FIFO buffer at the token receiving timing in the IN (transmitting) direction.
- When the FIFO buffer is full at the token receiving timing in the OUT (receiving) direction.

e) Interval errors

An interval error is generated on any of the following conditions when the device controller is selected:

- During an isochronous IN transfer, an IN token could not be received in the interval frame.
- During an isochronous OUT transfer, an OUT token could not be received in the interval frame.

Table 22-23 Error detection when a token is transmitted and received

Detection priority	Error	Generated interrupt and status
1	PID errors	No interrupts are generated in both cases when the host controller is selected and the device controller is selected (ignored as a corrupted packet).
2	CRC errors and bit stuffing errors	No interrupts generated in both cases when the host controller is selected and the device controller is selected (ignored as a corrupted packet).
3	Overrun and underrun errors	An NRDY interrupt is generated to set the FRMNUM.OVRN bit to 1 in both cases when the host controller is selected and device controller is selected. When the device controller is selected, a zero-length packet is transmitted in response to IN token. However, no data packets are received in response to OUT token.
4	Interval errors	An NRDY interrupt is generated when the device controller is selected. It is not generated when the host controller is selected.

Table 22-24 Error detection when a data packet is received

Detection priority	Error	Generated interrupt and status
1	PID errors	No interrupts are generated (ignored as a corrupted packet).
2	CRC errors and bit stuffing errors	An NRDY interrupt is generated to set the FRMNUM.CRCE bit to 1 in both cases when the host controller is selected and the device controller is selected.
3	Maximum packet size exceeded errors	A BEMP interrupt is generated to set the PID[1:0] bits to STALL in both cases when the host controller is selected and the device controller is selected.

22.3.10.2 Data PID

When the device controller is selected, the USBFS operates as follows in response to the received PID:

1) IN direction
   - DATA0: Sent as data packet PID
   - DATA1: Not sent
   - DATA2: Not sent
   - mDATA: Not sent
2) OUT direction
   - DATA0: Received normally as data packet PID
   - DATA1: Received normally as data packet PID
   - DATA2: Packets are ignored
   - mDATA: Packets are ignored

22.3.10.3 Interval counter

The isochronous transfer interval can be set using the PIPEPERI.IITV[2:0] bits. The interval counter enables the functions shown in Table 22-25 when the device controller is selected. When the host controller is selected, the token issuance timing is generated. When the host controller is selected, the interval counter operation is the same as that in the interrupt transfer.

Table 22-25 Interval counter function when the device controller is selected

Direction	Function	Conditions for detection
IN	Flushes transmit buffer	When an IN token cannot be successfully received in the interval frame during an isochronous IN transfer
OUT	Notifies that a token not being received	When an OUT token cannot be successfully received in the interval frame during an isochronous OUT transfer

The interval count is carried out when an SOF is received or for interpolated SOFs, so the isochronism can be maintained even if an SOF is damaged. The frame interval that can be set is the 2^{IITV} frames.

1) Counter initialization in device controller mode

   The USBFS initializes the interval counter in the following cases:

   - Power-on reset:

     This initializes the PIPEPERI.IITV[2:0] bits.
   - Use the ACLRM bit to initialize the FIFO buffer:

     This does not initialize the IITV [2:0] bits, but does initialize the count value.

   After initializing the interval counter, when a packet is successfully transmitted, the interval counter starts in the following two cases:

   - When PID=BUF, SOF is received after data is sent in response to an IN token.
   - When PID=BUF, SOF is received after data is received in response to an OUT token.

   The interval counter is not initialized in the following cases:

   - When the PID[1:0] bits are set to NAK or STALL

     The interval timer does not stop. The USBFS attempts transactions at the subsequent interval.
   - When the USB bus is reset or USBFS is suspended

     The IITV[2:0] bits are not initialized. When an SOF has been received, counting is restarted from the value prior to the reception of the SOF.

2) Interval counting and transfer control in host controller mode

The USBFS controls the interval between token issuance operations based on the PIPEPERI.IITV[2:0] bit settings. Specifically, the USBFS issues a token for a selected pipe once every 2^{IITV} frames.

The USBFS starts counting the token issuance interval at the frame following the frame in which the PID[1:0] bits have been set to BUF by software.

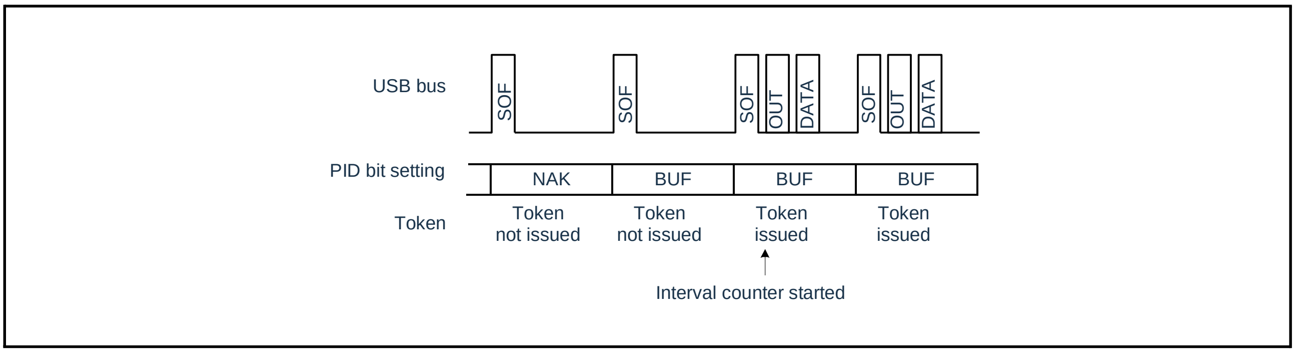


Figure 22-17 Token Issuance When IITV = 0

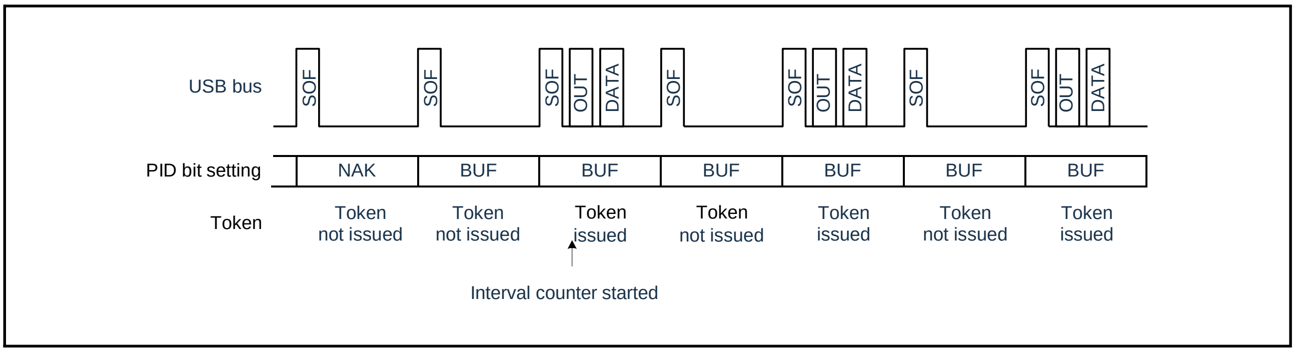


Figure 22-18 Token Issuance When IITV = 1

When the selected pipe is set for isochronous transfers, the USB carries out the following operation in addition to controlling the token issuance interval. The USB issues a token even when the NRDY interrupt generation condition is satisfied.

a) When the selected pipe is for isochronous IN transfers

The USBFS generates an NRDY interrupt when the USBFS issues an IN token but does not receive a packet successfully from a peripheral device (no response or packet error).

When it is time to issue the IN token and the USB module cannot receive data because the FIFO buffer is full (due to the CPU or DMA being too slow to read data from the FIFO buffer), the USB module sets the OVRN bit to 1 and generates an NRDY interrupt.

b) When the selected pipe is for isochronous OUT transfers

When it is time to issue the OUT token and there is no data in the FIFO buffer to be sent (due to the CPU or DMA being too slow to write the data to the FIFO buffer), the USBFS sets the OVRN bit to 1, which generates an NRDY interrupt and sends a zero-length packet.

The token issue interval is reset in any of the following cases:

- The IITV[2:0] bits are initialized when the USBFS is reset via the reset pin.
- When software sets PIPEnCTR.ACLRM to 1

3) Interval counting and transfer control in device controller mode

a) When the selected pipe is for isochronous OUT transfers

The USBFS generates an NRDY interrupt when the USBFS fails to receive a data packet within the interval set by the PIPEPERI.IITV[2:0] bits. The USBFS also generates an NRDY interrupt when the USBFS fails to receive data because of a CRC error or other errors contained in the data packet or because of the FIFO buffer being full.

The NRDY interrupt is generated at the timing of SOF packet reception. Even if the SOF packet is corrupted, the internal interpolation allows the interrupt to be generated at the timing to receive the SOF packet. However, when the IITV bits are set to a value other than 0, the USBFS generates an NRDY interrupt on receiving an SOF packet for every interval after starting interval counting operation.

When the PID[1:0] bits are set to NAK by software after starting the interval timer, the USBFS does not generate an NRDY interrupt on receiving an SOF packet.

The timing to start interval counting depends on the setting of IITV[2:0] bits as shown below:

- When the IITV[2:0] = 0:
- The interval counting starts when software has set the PID[1:0] bits for the selected pipe to BUF.

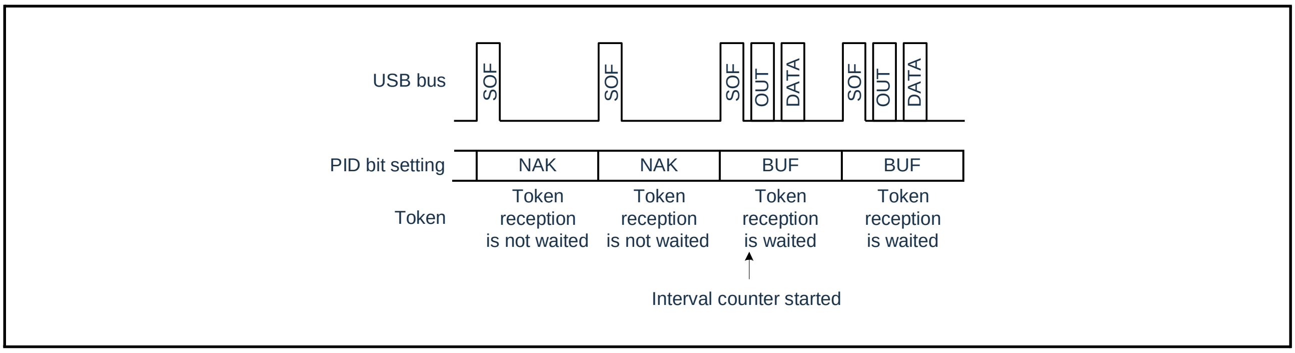

USB bus
SOF
SOF
SOF
OUT
DATA
SOF
OUT
DATA
PID bit setting
NAK
NAK
BUF
BUF
Token
Token reception is not waited
Token reception is not waited
Token reception is waited
Token reception is waited
Interval counter started

Figure 22-19 Relationship between frames and expected token reception when IITV= 0

- When the IITV[2:0] ≠ 000b: The interval counting starts on completion of successful reception of the first data packet after the PID[1:0] bits for the selected pipe have been modified to BUF.

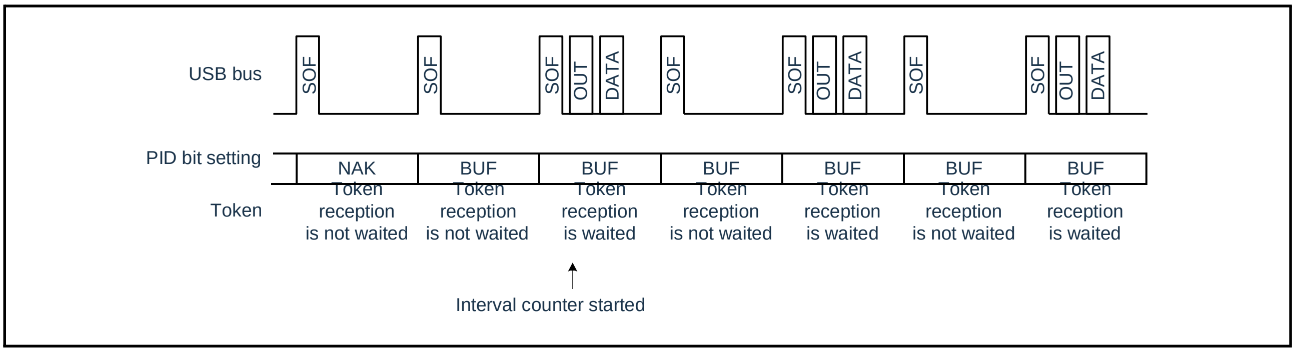

USB bus
SOF
SOF
SOF
OUT
DATA
SOF
SOF
OUT
DATA
SOF
SOF
OUT
DATA
PID bit setting
NAK
BUF
BUF
BUF
BUF
BUF
BUF
Token
Token reception is not waited
Token reception is not waited
Token reception is waited
Token reception is not waited
Token reception is waited
Token reception is not waited
Token reception is waited
Interval counter started

Figure 22-20 Relationship between frames and expected token reception when IITV≠0

b) When the selected pipe is for isochronous IN transfers

The PIPEPERI.IFIS bit should be 1 for this use. When IFIS = 0, the USBFS transmits a data packet in response to the received IN token irrespective of the setting of the PIPEPERI.IITV[2:0] bits.

When IFIS = 1, the USBFS clears the FIFO buffer when the USBFS fails to receive an IN token in the frame at the interval set by the IITV[2:0] bits while there is data to be transmitted in the FIFO buffer.

The USBFS also clears the FIFO buffer when the USBFS fails to receive an IN token successfully because of a bus error such as a CRC error contained in the IN token.

The FIFO buffer is cleared at the timing of SOF packet reception. Even if the SOF packet is corrupted, the internal interpolation allows the FIFO buffer to be cleared at the timing to receive the SOF packet.

As with OUT transfers, the time to start interval counting depends on the IITV[2:0] setting. In device controller mode, the interval is counted under any of the following conditions:

- When a hardware-reset is applied to the USBFS (here, the IITV[2:0] bits are also set to 000b).
- When the PIPEnCTR.ACLRM bit is set to 1 by software.
- When the USBFS detects a USB bus reset.

1) Transmit data setting for synchronized transmission in device controller mode

When using USBFS for isochronous data transfer in device controller mode, data packets are transmitted in the first frame after a SOF packet is detected after writing data to the FIFO buffer. This function is called the isochronous transfer transmission data setup function, and it makes it possible to designate the frame from which transmission began.

In a double buffer configuration, even after the writing of data to both buffers has been completed, transmission will be enabled for only one buffer to which data writing was completed first. Accordingly, even if multiple IN tokens are received, only one packet of data is transmitted from a single buffer.

When the FIFO buffer is ready to send data when an IN token is received, data is transmitted and a normal response is returned. However, if the FIFO buffer is unable to send data, a zero-length packet is sent and an underrun error occurs.

Upon reception of an IN token, if the buffer memory is ready to transmit, it will transmit data and return a normal response. However, if the buffer memory is unable to send data, a zero-length packet is sent and an underrun error occurs.

Figure 22-21 Shows an example of using the isochronous transfer transmission data setup function when IITV=0 (every frame) is set.

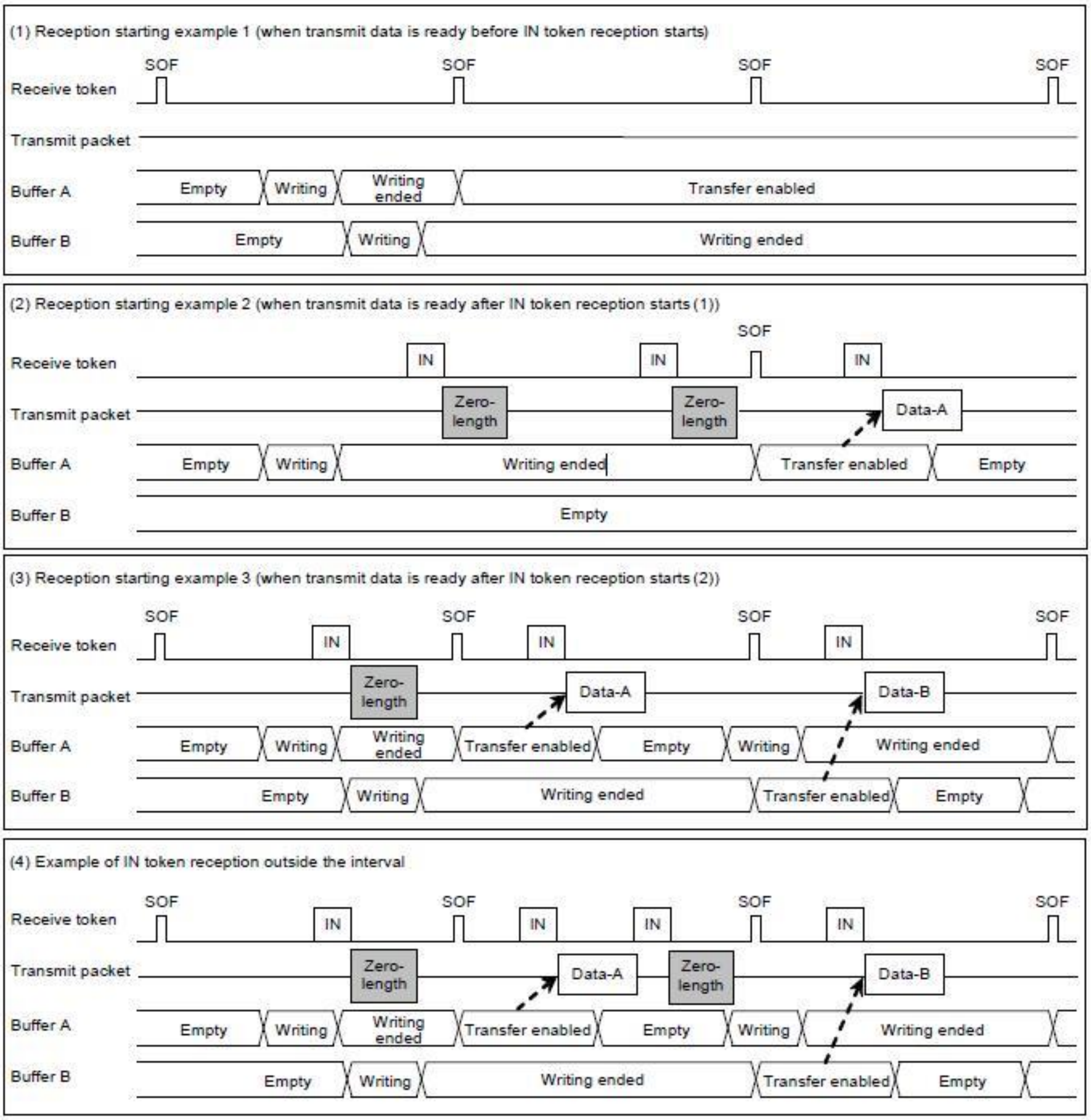


Figure 22-21 Example of data setup function operation

2) Isochronous transfer transmission buffer flush in device controller mode

If an SOF packet of the next frame is received without receiving an IN token in an interval frame during isochronous data transmission, the USB operates as if an IN token had been corrupted, and clears the buffer for which transmission is enabled, putting that buffer in the writing enabled state.

If a double buffer configuration is used and writing to both buffers has been completed, the buffer memory that was cleared is assumed as the data having been sent in the interval frame, and transmission is enabled for the buffer memory that is not cleared with SOF packet reception.

The timing of the buffer flush function depends on the setting of the PIPEPERI.IITV[2:0] bits.

- When IITV=0:

The buffer refresh operation will start from the first frame after the pipe is enabled.

- When IITV≠0:

The buffer flush operation will start after the first normal transaction.

Figure 22-22 shows an example of the buffer flush function. When an unanticipated token is received before the interval frame, the USBFS sends the write data or a zero-length packet as an underrun error according to the data setup state.

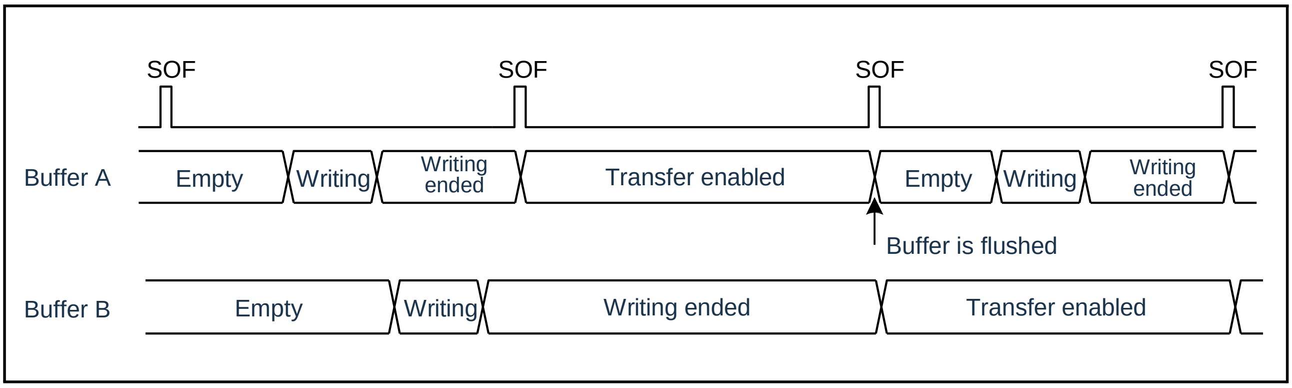


Figure 22-22 Example of buffer flush operation

Figures 22-23 shows an example interval error occurrence. As shown in the figures, there are five types of interval errors. The interval error occurs at ① in the figure and the buffer flush function is activated.

If an interval error occurs during an IN transfers, the buffer flush function is activated; if it occurs during an OUT transfer, an NRDY interrupt is generated. The FRMNUM.OVRN bit should be used to distinguish between NRDY interrupts such as received packet errors and overrun errors.

In response to tokens that are shaded in the figure, responses are sent according to the FIFO buffer status.

- IN direction:
- If the buffer is in the transmission enabled state, the data is transferred as a normal response.
- If the buffer is in the transmission disabled state, a zero-length packet is sent and an underrun error occurs.
- OU direction:
- If the buffer is in the reception enabled state, the data is received as a normal response.
- If the buffer is in the reception disabled state, the data is discarded and an overrun error occurs.

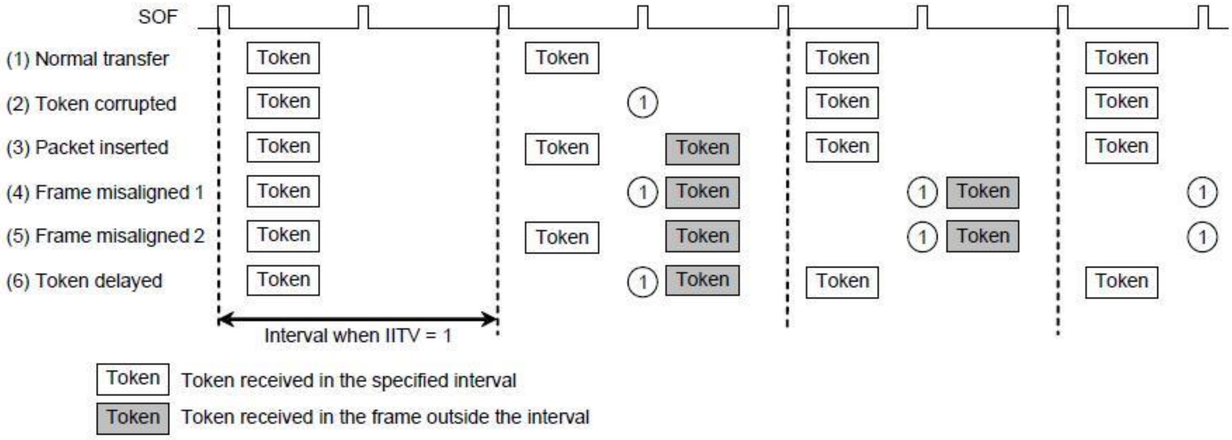


Figure 22-23 Example of interval error occurrence when IITV=1

22.3.11 SOF interpolation function

When the device controller is selected and if data could not be received at intervals of 1 ms because an SOF packet was corrupted or missing, the USB interpolates the SOF. The SOF interpolation operation starts when SYSCFG.USBE=1, SYSCFG.SCKE=1 and the SOF packet is received. The interpolation function will be initialized under the following conditions:

- MCU reset
- USB bus reset
- Suspended state detected

The SOF interpolation operates as follows:

- The interpolation function is not activated until an SOF packet is received.
- After the first SOF packet is received, interpolation is carried out by counting 1 ms with an internal clock of 48 MHz.
- After the second and subsequent SOF packets are received, interpolation is carried out at the previous reception interval.
- Interpolation is not carried out in the suspended state or while a USB bus reset is being received.

The USBFS supports the following functions based on the SOF packet reception. These functions also operate normally with SOF interpolation, if the SOF packet was missing:

- Updating of the frame number
- SOFR interrupt timing
- Isochronous transfer interval count

If an SOF packet is missing, the FRMNUM.FRNM[10:0] bits are not updated.

22.3.12 Pipe schedule

22.3.12.1 Conditions for generating a transaction

When the host controller is selected and the DVSTCTR0.UACT bit has been set to 1, the USBFS generates a transaction under the conditions shown in Table 22-26.

Table 22-26 Conditions for generating transactions

	Conditions for generation				
Transaction	DIR	PID	IITV0	Buffer state	SUREQ
Setup	—*1	—*1	—*1	—*1	Set to 1
Control transfer data stage, status stage, bulk transfer	IN	BUF	Invalid	Receive area exists	—*1
	OUT	BUF	Invalid	Transmit data exists	—*1
Interrupt transfer	IN	BUF	Valid	Receive area exists	—*1
	OUT	BUF	Valid	Transmit data exists	—*1
Isochronous transfer	IN	BUF	Valid	*2	—*1
	OUT	BUF	Valid	*3	—*1

*1. Symbols (-) in the table indicate that the condition is unrelated to the generating of tokens. “Valid” indicates that, for interrupt transfers and isochronous transfers, a transaction is generated only in transfer frames that are based on the interval counter. “Invalid” indicates that a transaction is generated regardless of the interval counter.

*2. This indicates that a transaction is generated regardless of whether there is a receive area. If there is no receive area, however, the received data is discarded.

*3. This indicates that a transaction is generated regardless of whether there is any data to be transmitted. If there is no data to be transmitted, however, a zero-length packet is transmitted.

22.3.12.2 Transfer schedule

This section describes the transfer scheduling within a frame of the USBFS. After the USBFS sends an SOF, the transfer is carried out in the sequence described below:

1. Execution of periodic transfers:

A pipe is searched in the order of PIPE1→PIPE2→PIPE6→PIPE7→PIPE8→PIPE9, and then, if there is a pipe for which an isochronous or interrupt transfer transaction can be generated, the transaction is generated.

2. Setup transactions for control transfers:

The DCP is checked, and if a setup transaction is possible, it is sent.

3. Execution of bulk transfers, control transfer data stages, and control transfer status stages:

A pipe is searched in the order of DCP→PIPE1→PIPE2→ PIPE3→PIPE4→PIPE5, and then, if there is a pipe for which a transaction for a bulk transfer, a control transfer data stage, or a control transfer status stage can be generated, the transaction is generated.

When a transaction is generated, processing moves to the next pipe transaction regardless of whether the response from the peripheral device is ACK or NAK. If there is time for transfer within the frame, step 3 is repeated.

22.3.12.3 Enabling USB communication

Setting the DVSTCTR0.UACT bit to 1 initiates SOF transmission and transaction generation is enabled. Setting the UACT bit to 0 stops SOF transmission and a suspend state is entered. If the setting of the UACT bit is changed from 1 to 0, processing stops after the next SOF is sent.

22.3.13 Battery charging detection processing

It is possible to control the processing for data contact detection (D+ line contact check), primary detection (charger detection), and secondary detection (charger verification), which are defined in the battery charging specification. The following describes required operations for a device device and a host device, individually.

22.3.13.1 Processing when device controller is selected

The following processing is required when operating the USBFS module as a portable device for battery charging:

1. Detect when the data lines (D+ and D-) have made contact and start the processing for primary detection.
2. After primary detection starts, wait 40 ms for masking, and then check the D- voltage level to confirm the primary detection result.
3. If the charger is detected during primary detection, also start secondary detection.
4. After secondary detection starts, wait 40 ms for masking, and then check the D+ voltage level to confirm the secondary detection result.

For step 1, after VBUS is detected using the VBINT and the VBSTS bits:

1. Wait for 300 to 900 ms by software, and then set the VDPSRCE0 and IDMSINKE0 bits in the USBBCCTRL0 register to 1.
2. Set the IDPSRCE0 bit to 1.
3. After a change from high to low on the D+ line is detected using the LNST[1:0] flags, set the IDPSRCE0 bit to 0 and set the VDPSRCE0 and IDMSINKE0 bits *1.

For step 2, set VDPSRCE0 and IDMSINKE0 bit to 1, wait 40 ms, and then use the CHGDETSTS0 bit to verify the primary detection results *2.

For step 3, if the CHGDETSTS0 flag is set to 1 in step 2, verify that the charger is detected, and then set the VDPSRCE0 and IDMSINKE0 bits to 0 and set the VDMSRCE0 and IDPSINKE0 bits to 1.

For step 4, set the VDMSRCE0 and IDPSINKE0 bits to 1 and wait for 40 ms, and then use the PDDETSTS0 bit to verify the secondary detection result.

Figure 22-24 shows the process flow.

*1. The battery charging specification describes two implementation methods of the process flow for data contact detection (D+/D- line contact check). One of the methods is to detect a change to logic low due to the pull-down resistor of the host device when the D+ and D- lines have made contact with the target while the D+ line is held at logic high by applying a current of 7 to 13 uA on the D+ line. The other method is to wait for 300 to 900 ms after VBUS is detected.

*2. During primary detection, when the voltage on the D- line is detected to be 0.25 to 0.4 V or above and 0.8 to 2.0 V or below, the target device is recognized as the host device for battery charging (charging downstream port). When using the CHGDETSTS0 bit to indicate only a USB transceiver with a voltage of 0.25 to 0.4 V or higher on the D-line, add processing that uses the LNST[1:0] bits to check for a voltage of 0.8 V to 2.0 V or less on the D-line as needed.

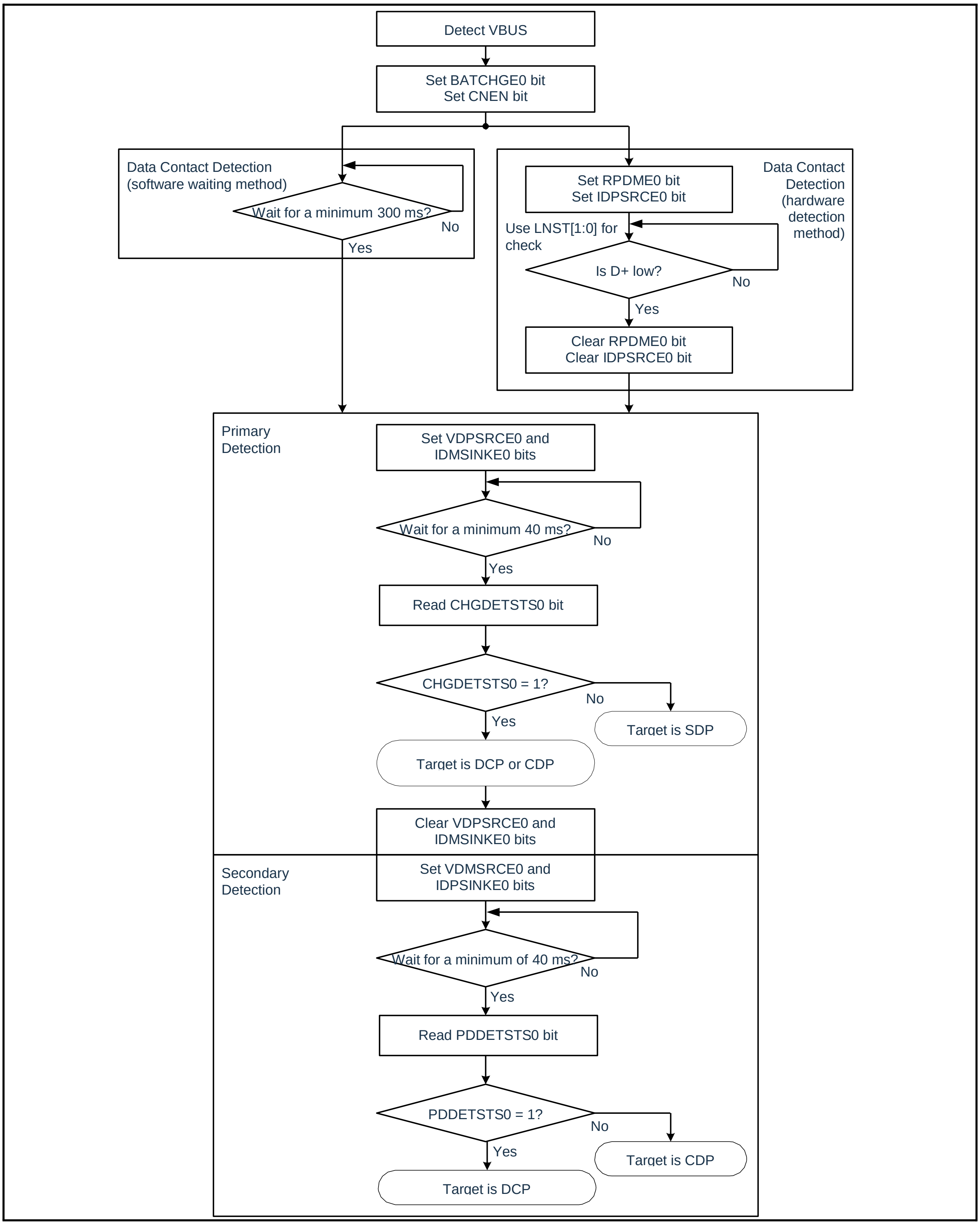


Figure 22-24 Process flow for operating as portable device

22.3.13.2 Processing when host controller is selected

The following processing is required when operating the USBFS module as a charging downstream port for battery charging.

1. Start driving the VBUS.
2. Enable the portable device detection circuit.
3. Monitor the portable device detection signal, and start driving the D- line if the detection signal is high.
4. Detect when the portable device detection signal is low level and stop driving the D- line.

Or, the following processing can also be used in accordance with the battery charging specification:

a. After disconnection is detected, start driving the D- line within 200 ms.

b. After connection is detected, stop driving the D- line within 10 ms.

The D- line must be driven to allow the portable device to detect the primary detection described in section22.3.13.1 Processing when device controller is selected. The above steps 1 to 4 apply when the portable device detection function is provided by hardware. This method is to drive the D- line when the portable device is detected. Steps a and b apply when the portable device function is not provided or available by hardware. Regardless of detection of the portable device, the D- line is driven in the disconnected state and the line is not driven in the connected state. In the battery charging specification, either of these methods can be used.

For steps 3 and 4, after a change in the portable device detection signal is detected using the PDDETINT interrupt, the current signal state can be confirmed by reading the PDDETSTS0 bit. Steps a and b can only be realized with a software timer.

Figure 22-25 shows the process flow for steps 1 to 4 and the process flow for steps a to b, respectively.

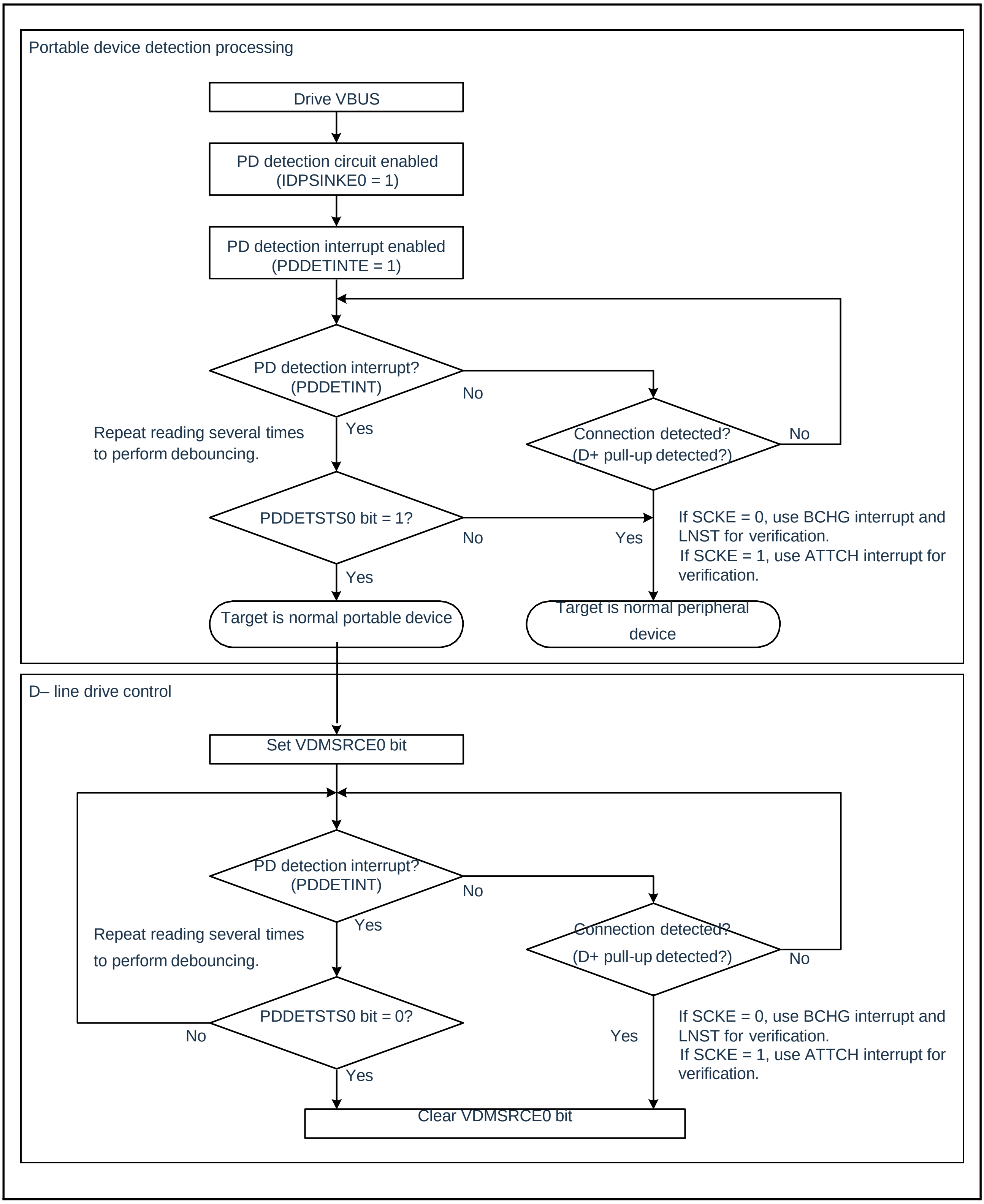

Portable device detection processing
Drive VBUS
PD detection circuit enabled (IDPSINKE0 = 1)
PD detection interrupt enabled (PDDETINTE = 1)
PD detection interrupt? (PDDETINT)
No
Yes
Connection detected? (D+ pull-up detected?)
No
Repeat reading several times to perform debouncing.
PDDETSTS0 bit = 1?
No
Yes
Yes
If SCKE = 0, use BCHG interrupt and LNST for verification.
If SCKE = 1, use ATTCH interrupt for verification.
Target is normal portable device
Target is normal peripheral device
D– line drive control
Set VDMSRCE0 bit
PD detection interrupt? (PDDETINT)
No
Yes
Connection detected? (D+ pull-up detected?)
No
Repeat reading several times to perform debouncing.
PDDETSTS0 bit = 0?
No
Yes
Yes
If SCKE = 0, use BCHG interrupt and LNST for verification.
If SCKE = 1, use ATTCH interrupt for verification.
Clear VDMSRCE0 bit

Figure 22-25 Process flow for operating as charging downstream port (steps 1 to 4)

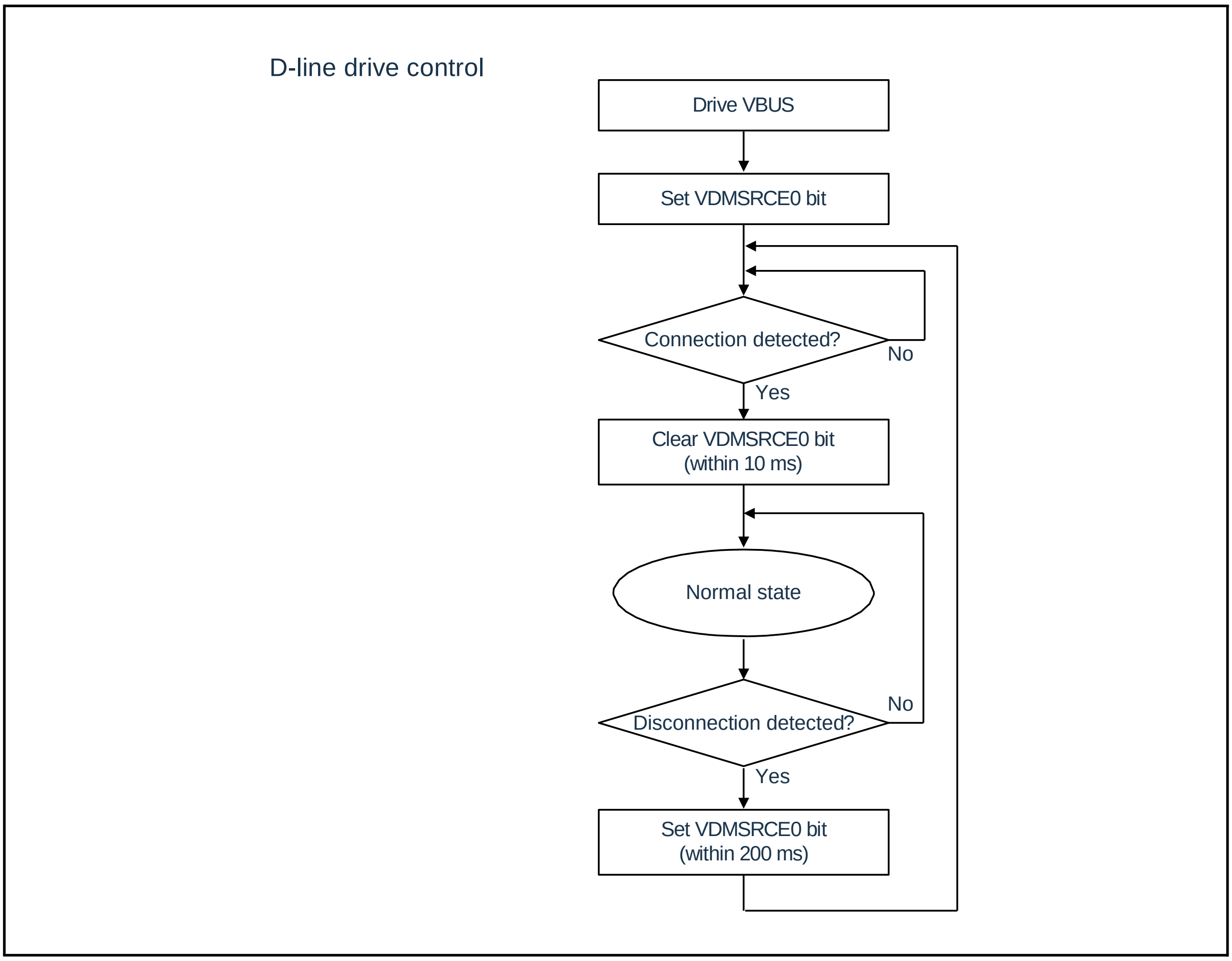


Figure 22-26 Process flow for operating as charging downstream port (steps a to b)

Chapter 23 LCD Bus Interface

The LCD Bus Interface connects the internal bus system to an external LCD Controller/Driver. It provides an asynchronous 8-bit parallel data bus and two control lines.

The LCD Bus Interface supports bidirectional communication. You can send data to and query data from the LCD controller.

23.1 Functions of LCD bus interface

The functions of the LCD Bus Interface are as follows:

- Support of two different control signals modes:
  - mod80 with separate read and write strobe
  - mod68 with read/write signal is controlled by a single pin with different levels.
- Data transfer sequence starts when internal data bus access LBDATA register
- Support 8/16 bit write and read operations
- Programmable transfer speed (max.10 MHz) through
  - selectable clock input
  - programmable transfer time
  - programmable wait states
- DMA trigger generation selectable upon two events (The interrupt can be used as DMA trigger only.)
  - internal data transfer allowed
  - external bus access completed
- Flags that indicate the status of the data register and the progress of data transfer to or from the LCD controller.
- DMA for read and write operations

Notice: When LCD bus is used under EVDDx ≦ VDD, registers of LCD C/D related must be initial value (LCDON=0, SCOC=0, MDSET1-0=00, LCDPFx=0), otherwise the normal operation can not be guaranteed.

Remark: If the concerned pins are configured as LCD Bus Interface pins, change between input and output is performed automatically by LCD Bus Interface to do read and write operations.

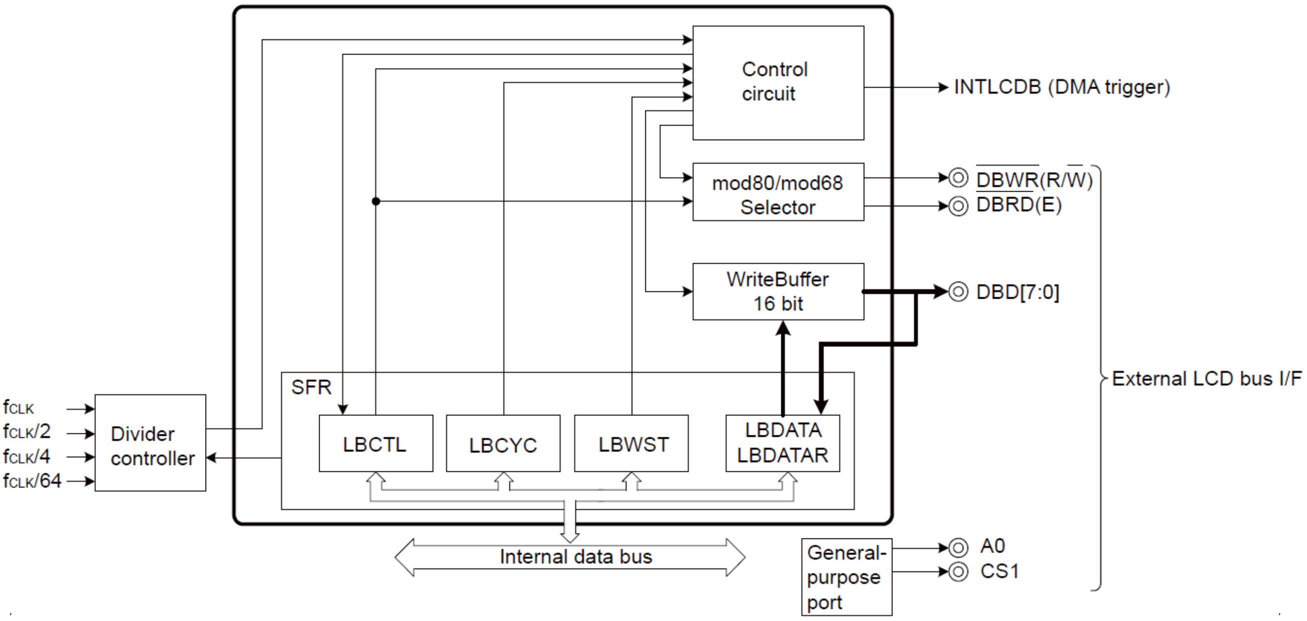
23.2 Configuration of LCD bus interface

The LCD Bus Interface consists of the following hardware:

Table 23-1 Configuration of LCD bus interface

Item	Configuration
Data I/O pins	8 pins (DBD7 to DBD0)
Control pins	$\overline{\text{DBWR}}$, $\overline{\text{DBRD}}$(mod80 (IMD=0)) R/W,E(mod68 (IMD=1))
Data registers	LCD bus interface data register (LBDATA, LBDATAL) LCD bus interface read data register (LBDATAR,LBDATARL)
Control registers	LCD bus interface mode register (LBCTL) LCD bus interface cycle time register (LBCYC) LCD bus interface wait status register (LBWST) Port mode control register (PMCx) Port mode register (PMx) Port register (Px) Peripheral enable register 1 (PER1)

Figure 23-1. Block diagram of LCD bus interface



23.2.1 LCD bus interface data register (LBDATA, LBDATAL)

LBDATA is used to store the data transferred through the LCD bus interface and supports 8-bit and 16-bit read/write operations.

The value of LBDATA after reset is 0000H.

Figure 23-2. Format of LCD bus interface data register (LBDATA, LBDATAL)

Address: 0x40045410 Reset value: 0000H RW

LBDATAL

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

Different ways of operating this register determine different transmission methods for the LCD bus interface.

- 8-bit accesses:

When an 8-bit operation is performed on a register, the LCD bus interface transfers according to 8 bits.

- 16-bit accesses:

When a 16-bit operation is performed on a register, the action is split into two 8-bit operations (the lower 8 bits are transmitted first, then the upper 8 bits), which are transmitted sequentially through the bus interface.

When the data is split into bits and transferred consecutively, the bit order is as follows:

15 8	7 0
2nd 8-bit	1st 8-bit

Write to LBDATA:

A write operation to this register sets the busy flag LBCTL.BYF immediately.

If there is no LCD bus transfer in progress (LBCTL.TPF = 0), the data is copied to the write buffer and LBCTL.BYF is cleared.

If there is a transfer going on (LBCTL.TPF = 1), the data is not copied to the write buffer until the transfer has completed. As soon as the transfer is complete, the data is copied to the write buffer and LBCTL.BYF is cleared. A transfer via the LCD Bus Interface starts as soon as the LBDATA register is copied to the write buffer. This is indicated by INTLCDB (DMA trigger) that becomes active, provided that LBCTL.TCIS = 0.

Read from LBDATA:

A read operation from this register initiates a read transfer via the LCD Bus Interface. The data that is read from the register is always the data that was received during the previous transfer from the LCD Bus Interface.

Notice: 1. An access to an LBDATA register can only point to the even address of the register and is prohibited to point to the odd address of the register.

2. LBCTL.BYF must be zero when accessing this register.

23.2.2 LCD bus interface read data register (LBDATAR,LBDATARL)

LBDATAR is read-only. It contains the data of the last previous read transfer via the LCD Interface. Reading this register does not start a new read transfer on the LCD Bus Interface.

This register supports 16-bit or 8-bit read operations.

When a reset occurs, the reset value of the LBDATAR is 0000H.

Figure 23-3. Format of LCD bus interface read data register (LBDATAR,LBDATARL)

Address: 0x40045412H Reset value: 0000H R/W

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

This register can be read to obtain data that was transferred during a previous read operation to the LBDATA register without initiating a further LCD bus transfer.

Reading the LBDATAR register does not change the status of the LBCTL.BYF and LBCTL.TPF flags.

Note: An access to an LBDATAR register can only point to the even address of the register and is prohibited to point to the odd address of the register.

23.3 Registers for controlling LCD bus interface

The following ten registers are used to control the LCD Bus Interface.:

- LCD bus interface mode register (LBCTL)
- LCD bus interface cycle time register (LBCYC)
- LCD bus interface wait 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)

PER1 is used to enable or disable use of each peripheral hardware macro. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When the LCD Bus Interface is used, be sure to set bit 1 (LCDBEN) of this register to 1.

PER1 can be set by an 8-bit memory manipulation instruction.

Figure 23-4 Peripheral enable register 1 (PER1)

Address: 0x4002081A After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
PER1	XX	XX	XX	XX	XX	XX	LCDBEN	XX

LCDBEN	Control of LCD bus controller clock supply
0	Stops input clock supply. •SFR used by the LCD bus controller cannot be written. •The LCD bus controller is in the reset status.
1	Supplies input clock. •SFR used by LCD bus controller can be read and written.

23.3.2 LCD bus interface mode register (LBCTL)

LBCTL controls the operation of LCD Bus Interface.

LBCTL is set using an 8-bit memory manipulation instruction. Reset signal generation sets LBCTL to 00H.

Figure 23-5. Format of LCD bus interface mode register (LBCTL)

Address:0x40045400 Reset value:00H R/W

Symbol	7	6	5	4	3	2	<1>	<0>
LBCTL	EL	IMD	LBC1	LBC0	TCIS	0	TPF	BYF

EL	Control the level of signal “E” in mod68 mode
0	E is active high; data is read/written on the falling edge.
1	E is active low, data is read/written on the rising edge.

IMD	Mode of external bus interface access selection
0	mod80 mode - control signals are $\overline{WR}$ and $\overline{RD}$
1	mod68 mode - control signals are E and R/$\overline{W}$

LBC1	LBC0	Internal clock selection (SPCLK)
0	0	f_{CLK}
0	1	$f_{CLK}/2$
1	0	$f_{CLK}/4$
1	1	$f_{CLK}/64$

TCIS	INTLCDB (DMA trigger) generation control bit
0	During write access to the bus interface, an INTLCDB is generated as soon as data is transferred from LBDATA to the write buffer. During read access from the bus interface, an INTLCDB is generated as soon as data is available in the LBDATA and LBDATAR registers.
1	An interrupt generated on completion of data transfer on LCD bus interface

TPF	Flag of transfer in progress on external bus interface
0	The external bus interface is idle
1	Data is being transferred on the external bus interface

BYF	Data register busy flag
0	Data can be read or written from/to LBDATA Data can be read from LBDATAR
1	Register LBDATA (LBDATAR) is busy

Notice: 1. Bits 2 must be set to 0.

2. Though the LBCTL.TPF flag is intended to determine the current status of the LCD bus data transfer, reading of this flag may indicate a wrong status by accident.

Therefore, instead of polling the LBCTL.TPF flag it is recommended to use a DMA transfer to load new LCD data into the LCD bus interface data register (LBDATAx).

23.3.3 LCB bus interface cycle control register (LBCYC)

LBCYC register determines the cycle time of the LCD Bus Interface. The cycle time is the duration of one bus access for transferring one 8-bit data. LBCYC is set using an 8-bit memory manipulation instruction.

Reset signal generation sets LBCYC to 00H.

Figure 23-6 Format of LCB bus interface cycle control register (LBCYC)

Address: 0x40045401 Reset value: 02H R/W

Symbol	7	6	5	4	3	2	1	0
LBCYC	0	0	CYC5	CYC4	CYC3	CYC2	CYC1	CYC0

CYC5-CYC0	Cycle time
000000B	Disable settings
000001B	
000010B	Cycle time is 2 ×T
000011B	Cycle time is 3 ×T
:	:
111110B	Cycle time is 62×T
111111B	Cycle time is 63 ×T

Notice: 1. T is the clock period of the selected clock (set by LBC1 and LBC0)

2. LBCYC≥2.

23.3.4 LCB bus interface wait control register (LBWST)

LBWST determines the number of wait states of the LCD Bus Interface. The number of wait states defines the duration of the $\overline{DBWR}$ and $\overline{DBRD}$ signals. This duration must remain below the cycle time.

LBWST is set using an 8-bit memory manipulation instruction.

Reset signal generation sets LBWST to 00H.

Figure 23-7. Format of LCD bus interface wait control register (LBWST)

Address: 0x40045402 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 cycles
00000B	No wait cycle inserted
00001B	1×T
00010B	2 ×T
00011B	3 ×T
:	:
11110B	30 ×T
11111B	31 ×T

Notice: 1. T is the clock period of the selected clock (set by LBC1 and LBC0)

2. WST⩽CYC-2.

23.3.5 Port mode control register

When using the LCD bus interface pins, the control registers for the multiplexed port functions (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) ”, 2.3.6 Pull-down resistor selection register (PDxx) ”, and “2.5 Register settings when using the multiplexing function”.

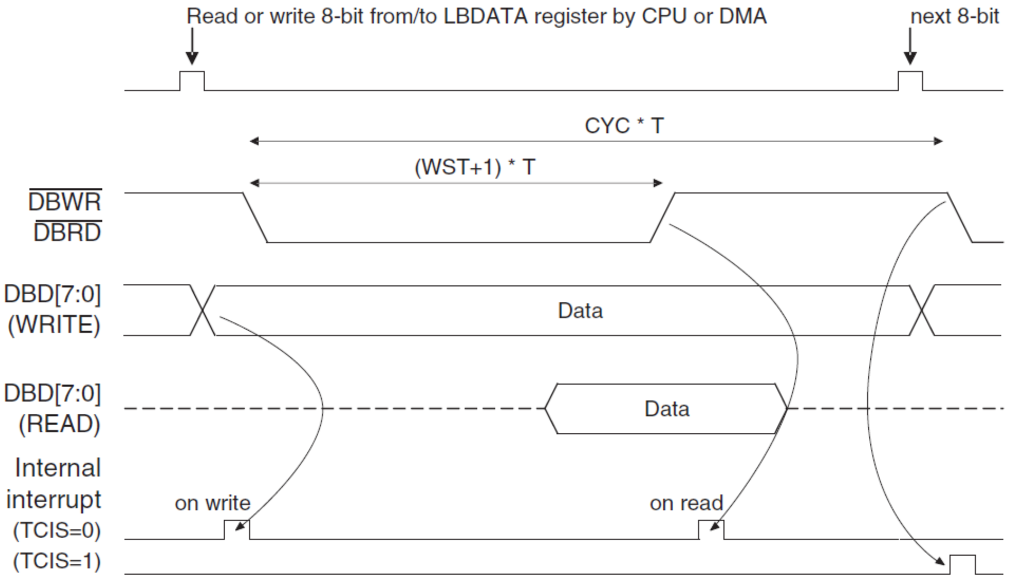
23.4 Operation of timing

This section starts with the general timing and then presents examples of consecutive write and read operations.

23.4.1 Timing dependencies

The following figure shows the general timing when the mod80 mode is used. It illustrates the effect of the LBCYC and LBWST register settings. It explains also the impact of LBCTL.TCIS on the INTLCDB generation.

Figure 23-8. LCD bus interface timing (mod80 mode)



In mod80 mode, $\overline{\text{DBWR}}$ provides the write strobe, and $\overline{\text{DBRD}}$ the read strobe.

Notice: 1. T is the clock period of the internal clock (SPCLK) selected with the LBC1 and LBC0 bits.

2. CYC is the chosen number of clock cycles (LBCYC). Always keep LBCYC > 2.
3. WST is the chosen number of wait states (LBWST). Always keep LBWST < (LBCYC – 2).

The only difference in mod68 mode is, that DBWR provides the read/write R/$\overline{\text{W}}$, and DBRD is replaced with the select enable signal E. The effective level of the select signal E is determined by LBCTL.EL.

23.4.2 LCD bus interface status

When the chip pins are configured for use as LCD data bus interface DB[7:0], the input and output modes of the pins are automatically switched by the LCD module control. After the pin is configured as DB[7:0], the pin is in input mode. When the bus interface operates during a read access, DB[7:0] operates in input mode and also continues to remain in input mode after the read access is completed. During a write access, DB[7:0] operates in output mode and remains in output mode after the write access is completed.

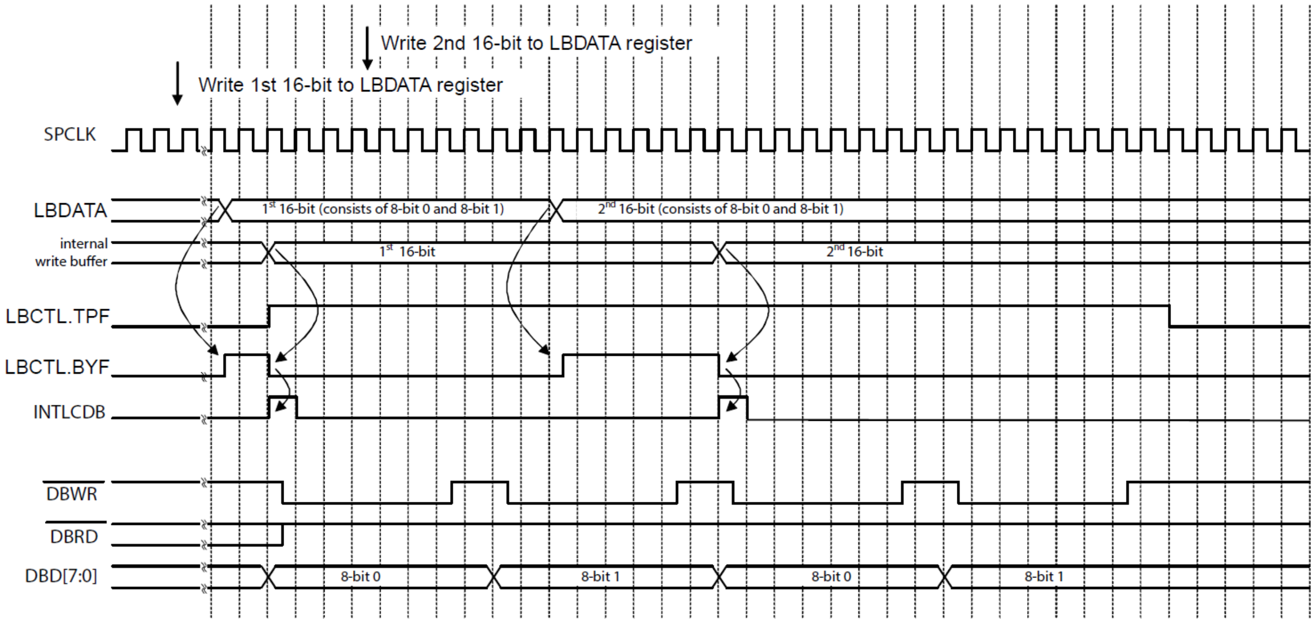
23.4.3 Writing to LCD bus

This section shows typical sequences of writing 16 bits and 8 bits to the LCD bus.

(1). 16-bit writing

16-bit writing transmits two 8-bit data to the external LCD Controller/Driver.

Figure 23-9. Timing (mod80: LBTCTL.IMD = 0): write consecutive 16 bits, LBWST = 5, LBCYC = 8, LBCTL.TCIS = 0


Write 2nd 16-bit to LBDATA register
Write 1st 16-bit to LBDATA register
SPCLK
LBDATA
1st 16-bit (consists of 8-bit 0 and 8-bit 1)
2nd 16-bit (consists of 8-bit 0 and 8-bit 1)
internal write buffer
1st 16-bit
2nd 16-bit
LBCTL.TPF
LBCTL.BYF
INTLCDB
DBWR
DBRD
DBD[7:0]
8-bit 0
8-bit 1
8-bit 0
8-bit 1

Description: The timing diagrams are for functional explanation purposes only without any relevance to the real hardware implementation.

(a) Sequence

① The first 16 bits of LCD data is written to the LBDATA register. The internal bus transfer takes some clocks until the interface register is written. Then the busy flag LBCTL.BYF is set until the data is copied to the write buffer.

② The LBDATA register contents is copied to the write buffer. This clears LBCTL.BYF and causes the INTLCDB output to become active for one clock cycle. Transfer on the LCD bus interface starts with 8-bit data 0. The flag LBCTL.TPF is set to indicate that a transfer is in progress.

③ Caused by the INTLCDB, the DMA writes a second 16 bits to LBDATA. The CPU can write this 16 bits as well after it has checked the busy flag LBCTL.BYF. The internal bus transfer again takes some clock cycles until the LBDATA register is written and LBCTL.BYF is set.

④ Because the transfer on the LCD bus interface is still going on and the LBDATA register contents can not be copied to the write buffer immediately, LBCTL.BYF is set.

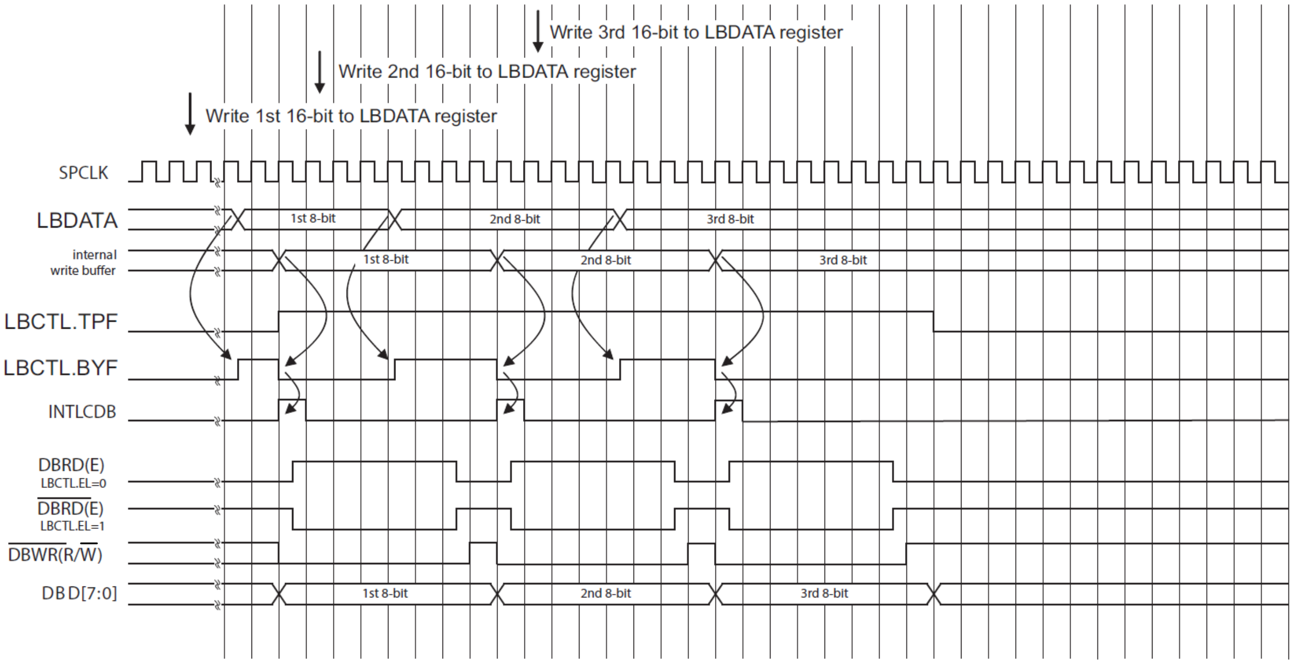
⑤ After the transfer over the LCD bus interface has been completed, the write buffer is filled with the contents of LBDATA. The busy flag LBCTL.BYF is cleared, and the INTLCDB becomes active for one clock cycle.

Filling the write buffer starts a new transfer to the external LCD controller.

(2). 8-bit writing

Writing consecutive 8 bits transmits these 8 bits to the external LCD controller/driver.

Figure 23-10. Timing (mod68 mode: LBTCTL.IMD = 1): write consecutive 8 bits, LBWST = 5, LBCYC = 8, , LBCTL.TCIS = 0



Description: The timing diagrams are for functional explanation purposes only without any relevance to the real hardware implementation.

(b) Sequence

① The first 8-bit of LCD data is written to the LBDATA register. The internal bus transfer takes some clocks until the register of the interface is written. Then the busy flag LBCTL.BYF is set until the data is copied to the write buffer.

② The LBDATA register contents is copied to the write buffer. This clears LBCTL.BYF and causes the INTLCDB output to become active for one clock cycle. Transfer on the LCD bus interface is started with 8 bits of data 0. The flag LBCTL.TPF is set to indicate that a transfer is in progress.

③ Caused by the INTLCDB, the DMA writes a second 8-bit to LBDATA. The CPU can write this 8-bit as well after it has checked the busy flag LBCTL.BYF. The internal bus transfer again takes some clock cycles until the LBDATA register is written and LBCTL.BYF is set.

④ Since the transfer on the LCD bus interface is still going on and the LBDATA register contents can not be copied to the write buffer immediately, the flag LBCTL.BYF remains set.

⑤ After the transfer on the LCD bus interface has been completed, the write buffer is filled with the contents of LBDATA. The busy flag LBCTL.BYF is cleared and the INTLCDB becomes active for one clock cycle.

Filling the write buffer starts a new transfer to the external LCD controller.

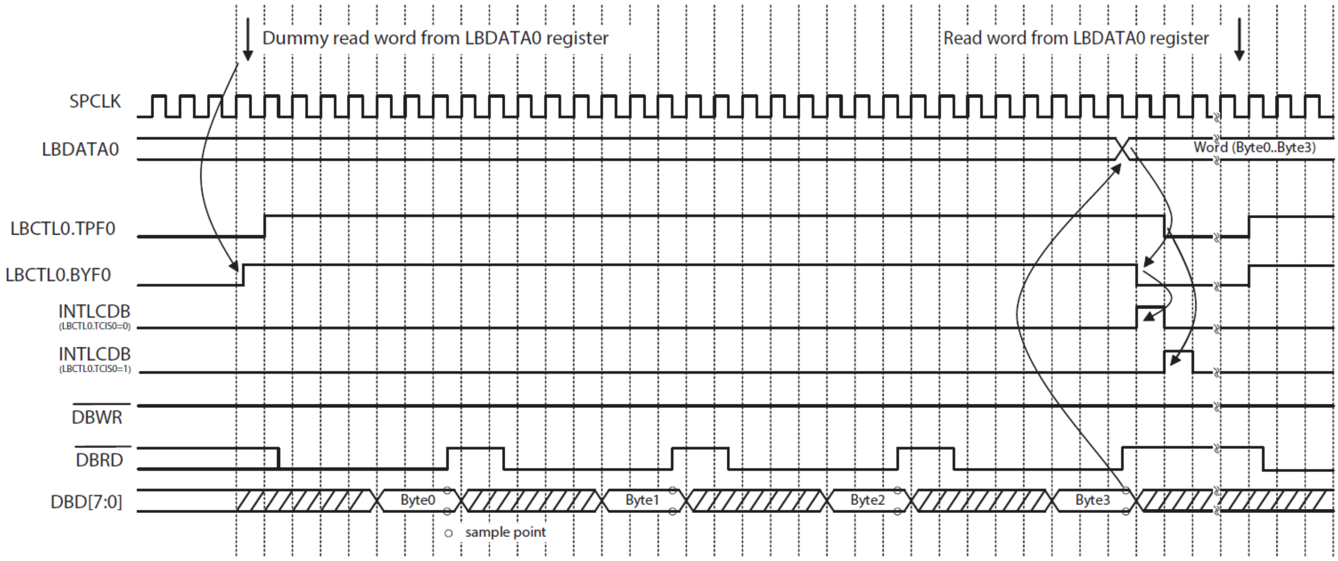
23.4.4 Reading from the LCD bus

You can read from the LCD bus in 8-bit or 16-bit format. The following shows typical sequences of reading 8 bits.

(1). 16-bit reading

The following figure shows 16 bits read operation in mod80 mode.

Figure 23-11(mod80: LBTCTL.IMD=0): Read 16-bit timing
LBWST=5,LBCYC=8,LBCTL.TCIS=0 and 1



Description: The timing diagrams are for functional explanation purposes only without any relevance to the real hardware implementation.

(a) Sequence

① A dummy read to the LBDATA register starts the transfer of four bytes from the external LCD controller. The busy flag LBCTL.BYF is set immediately. The “transfer in progress” flag LBCTL.TPF is set on the rising edge of the clock. The data that is read from LBDATA belongs to a previous transfer and may be ignored.

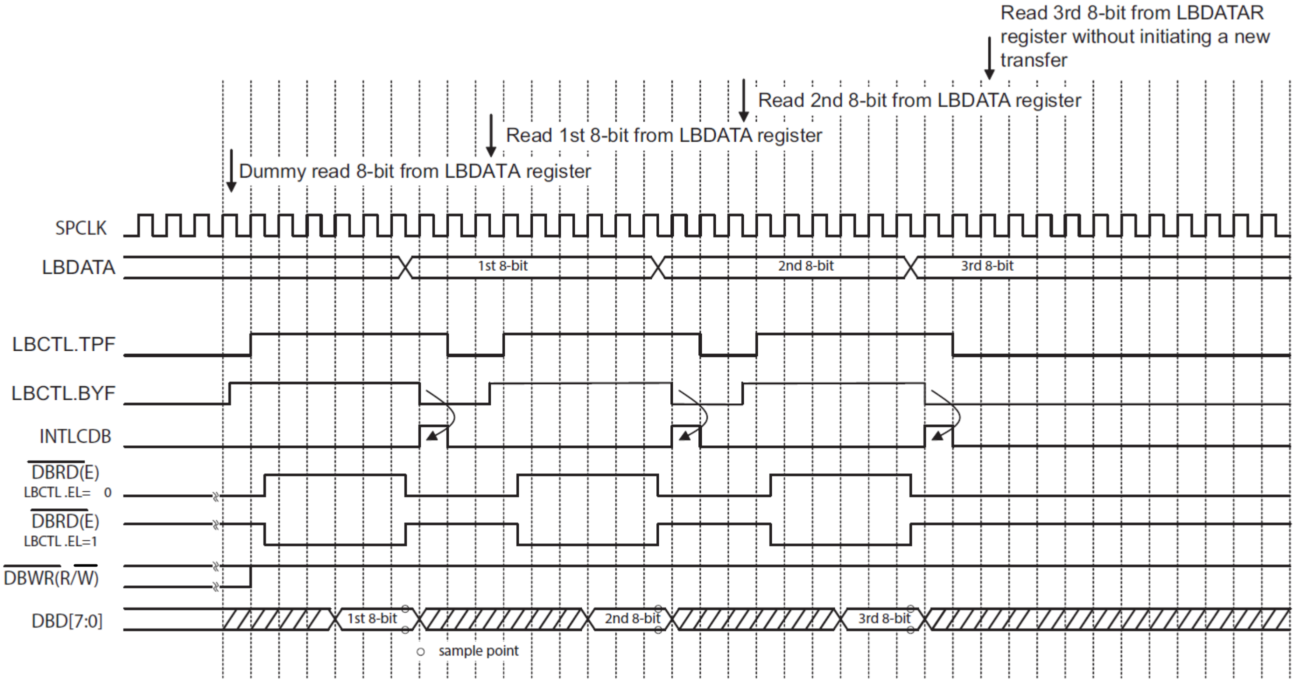
② When the last of the four bytes is sampled and the complete word is available in the LBDATA register, the busy flag LBCTL.BYF is cleared. The LBCTL.TPF flag remains set until the cycle time of the last byte has elapsed.

③ A following read to the LBDATA register provides the LCD controller data and initiates a new transfer.

(2). 8-bit reading

The following figure shows 8 bits read operation in mod68 mode.

Figure 23-12 (mod68: LBTCTL.IMD=1): read consecutive 8 bits
LBWST=4,LBCYC=7,LBCTL.TCIS=0


Read 3rd 8-bit from LBDATAR register without initiating a new transfer
Read 2nd 8-bit from LBDATA register
Read 1st 8-bit from LBDATA register
Dummy read 8-bit from LBDATA register
SPCLK
LBDATA
1st 8-bit
2nd 8-bit
3rd 8-bit
LBCTL.TPF
LBCTL.BYF
INTLCDB
DBRD(E)
LBCTL.EL= 0
DBRD(E)
LBCTL.EL=1
DBWR(R/W)
DBD[7:0]
1st 8-bit
2nd 8-bit
3rd 8-bit
sample point

Description: The timing diagrams are for functional explanation purposes only without any relevance to the real hardware implementation.

(a) Sequence

① A dummy read to the LBDATA register starts the transfer of one 8-bit data from the external LCD controller. The busy flag LBCTL.BYF is set immediately. The “transfer in progress” flag LBCTL.TPF is set on the rising edge of the clock. The data that is read from LBDATA belongs to a previous transfer and may be ignored.

② When the data on the LCD Bus Interface is sampled, the flag bit LBCTL.BYF is cleared and the data is available in LBDATA. The interrupt output INTLCD becomes active for one clock cycle.

③ A new read to LBDATA is performed while the previous transfer has not been finished (cycle time not elapsed). The busy flag LBCTL.BYF is set immediately, but the new transfer is started after the previous one is complete. The “transfer in progress flag” LBCTL.TPF remains set. The data that is read from LBDATA is the first 8-bit LCD data.

④ Again, the data that has been sampled is available in LBDATA and the busy flag LBCTL.BYF is cleared.

⑤ Steps 2 to 4 are repeated until the last 8 bits to be read has been sampled.

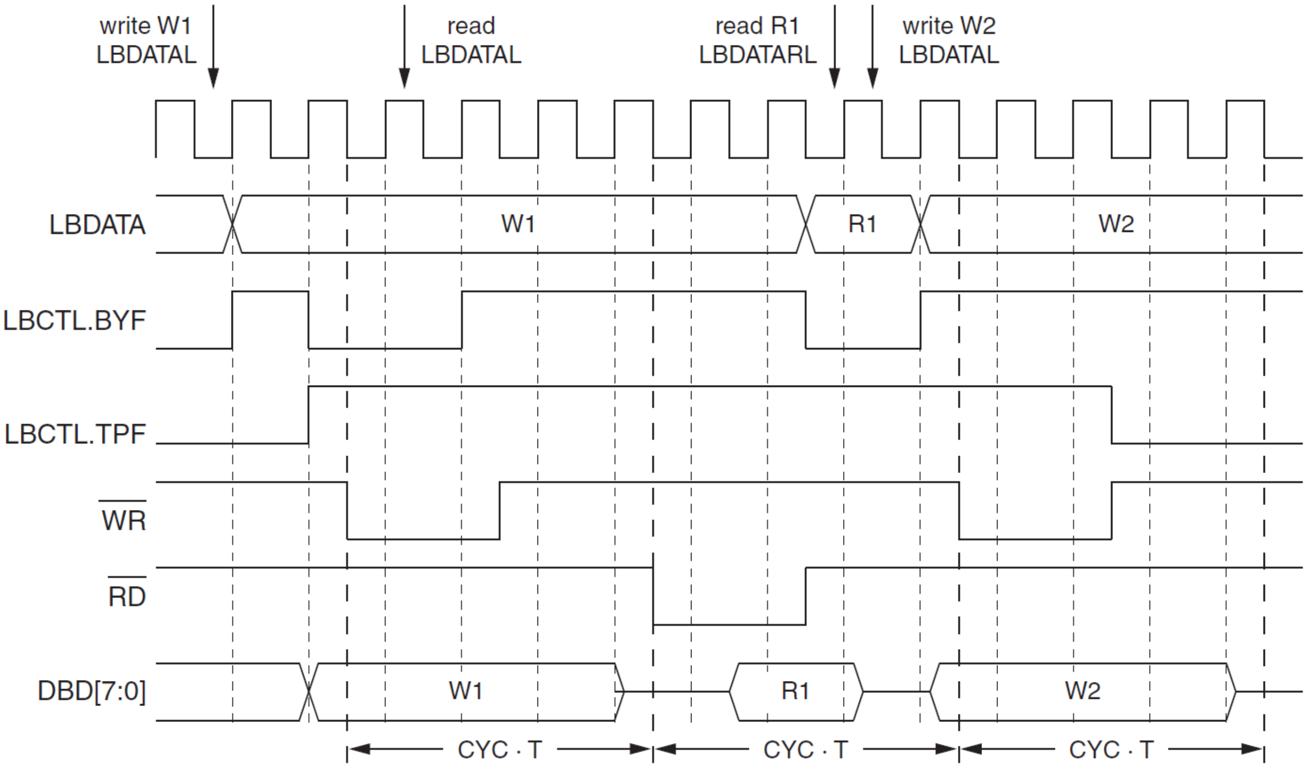
⑥ The last 8 bits is not read from the LBDATA register but from LBDATAR in order to avoid a further read transfer on the LCD bus.

23.4.5 Write-Read-Write sequence on the LCD bus

Figure 23-13 shows an example when a write access to the LCD bus is immediately followed by a read access and vice versa. The example is given in mod80 mode (LBCTL.IMD = 0) with 8-bit transfers.

In mode68 mode (LBCTL.IMD = 1) the timing is equivalent, when the RD strobe is considered as the low active E signal (LBCTL.EL = 1).

Figure 23-13.(mod80: LBTCTL.IMD=0): 8-bit write-read-write, LBWST=4,LBCYC=7,LBCTL.TCIS=0



23.5 Cautions for LCD bus interface

23.5.1 Writing to the LBDATA/ LBDATAL register

When the LCD data bus is transferring, a write operation to the LBDATAx register may cause a data transfer conflict. To avoid this, the following operation must be performed:

Avoid writing the LBDATAx register while the LCD bus is transferring. When a transfer is in progress, to ensure that the LBDATAx register is not written, set LBCTL.TCIS to 1, and depending on whether a bus interface interrupt is generated or not, decide whether or not to write the LBDATAx register.

It is recommended to use DMA transfers to load new LCD data into the LCD bus interface data register (LBDATAx).

23.6 Example of LCD bus interface transfer

23.6.1 Connection example of external LCD driver

Example1.

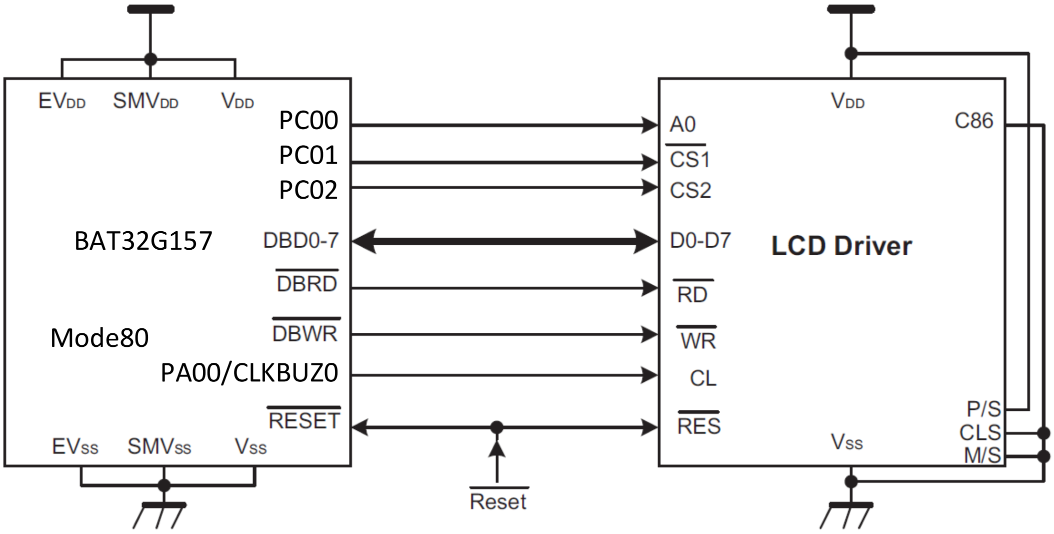
BAT32G157 can be used as a master chip and supply clock from the CLKBUZ0 pin to slave chip (LCD driver) for display clock.

System composition:

- System clock 32MHz, LCDB access cycle 8 MHz ($f_{CLK}/4$)
- Mod68/80
- CL comes from CLKBUZ0 ($f_{CLK}/2^{11}$=15.6kHz)

Figure 23-14. Connection example 1

<Mode80>



<Mode68>

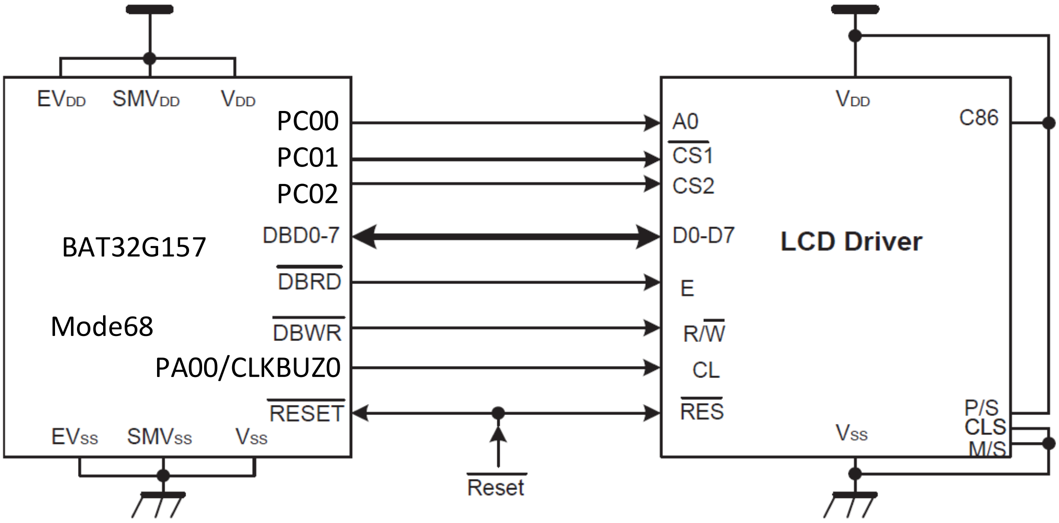


Table 23-2. Connection example 1

No.	LCD driver pin	LCD driver function	Port name
1	A0	To determine D0 to D7 are data or command	PC00 [Note]
2	$\overline{\mathrm{CS1}}$	Chip select 1	PC01 [Note]
3	CS2	Chip select 2	PC02 [Note]
4	D0 to D7	8-bit bi-directional data bus	DBD0 to DBD7
5	$\overline{\mathrm{RD}}$(E)	mod80: read strobe mod 68: enable strobe	_DBRD
6	$\overline{\mathrm{W}}$ R(R/$\overline{\mathrm{W}}$)	mod 80: write strobe mod 68: read/write control	_DBWR
7	CL	Display clock	PA00/CLKBUZ0
8	$\overline{\mathrm{RES}}$	Reset	_RESET

Note: Using PC00, PC01 and PC02 as A0, $\overline{\mathrm{CS1}}$ and CS2 is only an example, other port can also be used.

Example 2.

BAT32G157 can be used as a master chip and supply clock from CLKBUZ0 pin to slave chip (LCD driver) for display clock.

System composition:

- f_{CLK}=6MHz
- PCF21119x mod68
- f_{OSC} comes from CLKBUZ0 (f_{CLK}/16=375kHz)

Figure 23-15 Connection example 2

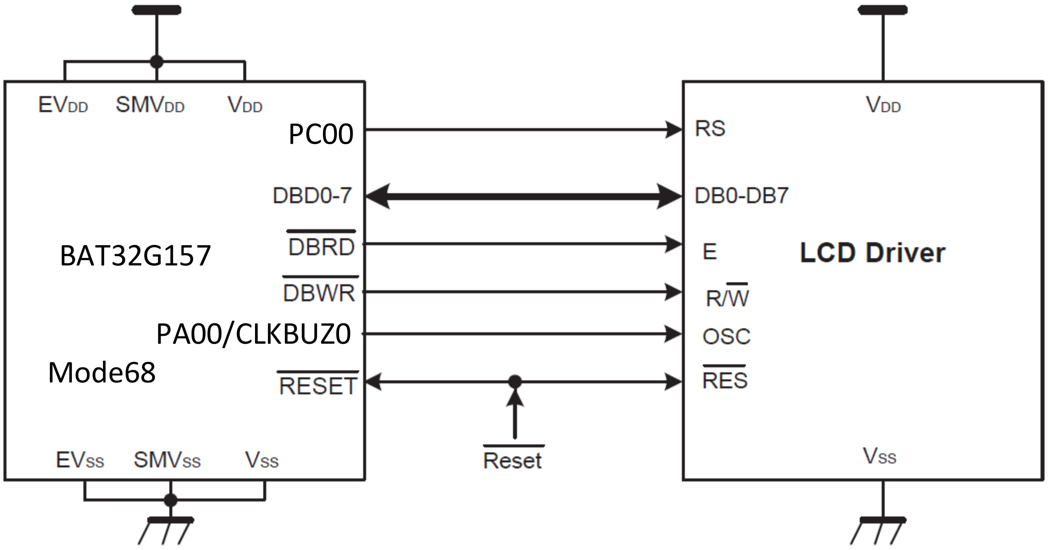


Table 23-3. Connection example 2

No.	LCD driver pin	LCD driver function	Port name
1	RS	Register select	PC00 [Note]
2	DB0 to DB7	8-bit bi-directional data bus	DBD0 to DBD7
3	E	Strobe signal	_DBRD
4	R/$\overline{\mathrm{W}}$	Read/Write control	_DBWR
5	OSC	Oscillator or external clock input	PA00/CLKBUZ0
6	$\overline{\mathrm{RES}}$	Reset	_RESET

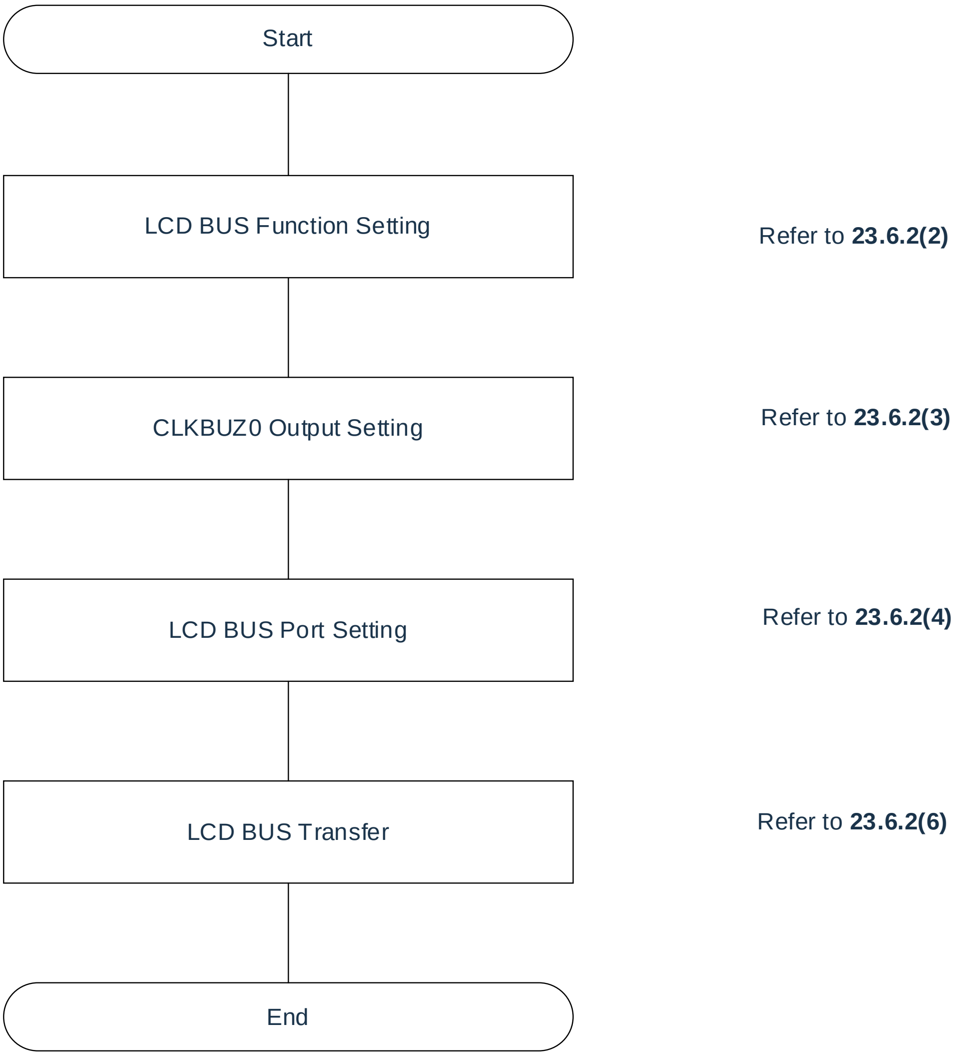
Note: Using PC00 as RS is only an example, other port can also be used.

23.6.2 Operation procedure of LCD BUS transfer

(1). Flow chart (recommended)

This flow chart is the operation procedure of LCD BUS transfer. Every step is described in details at the following sections. (Right side is the section number.)

Figure 23-16 Whole flowchart of LCD BUS transfer

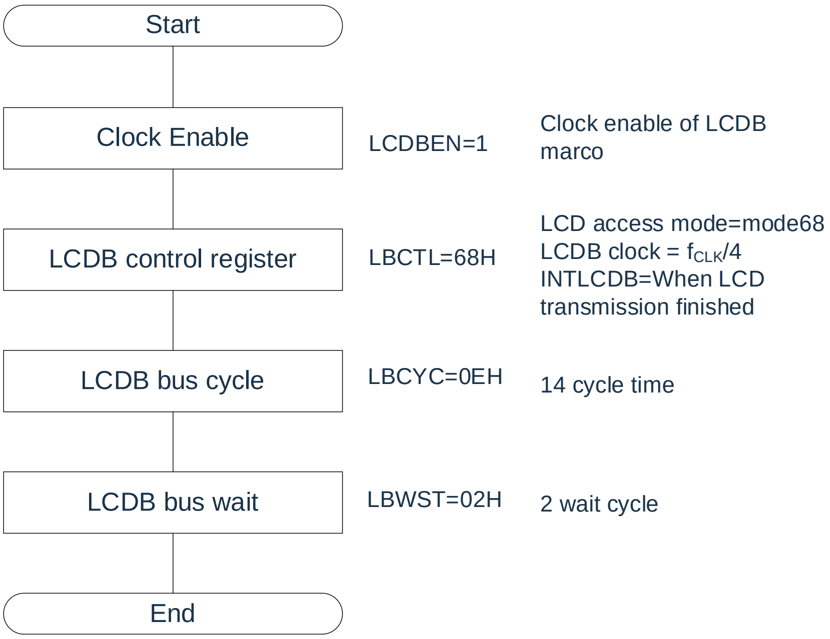


(2). LCD BUS function setting

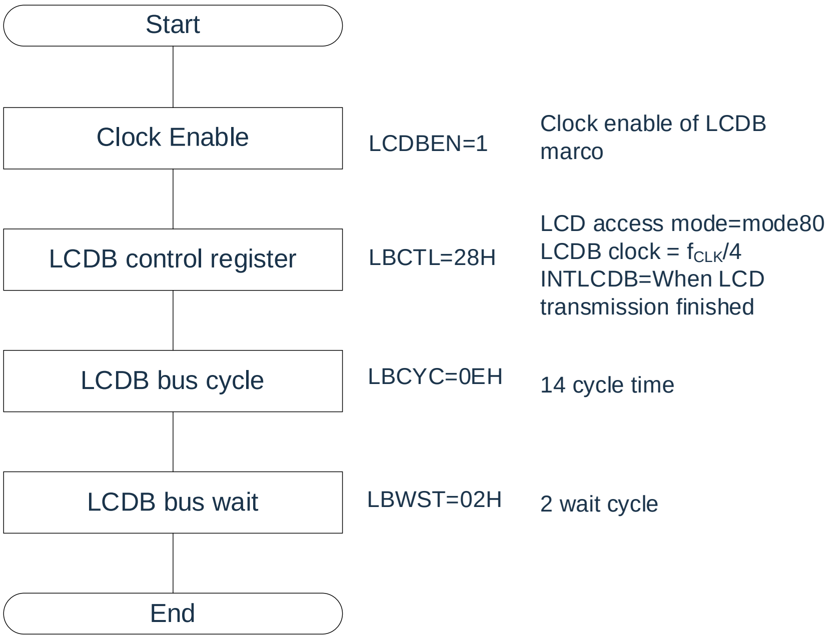
- Enable LCDB module clock
- Set access mode to be mode68 or mode80
- Set internal clock (example: f_{CLK}/4)
- Sets INTLCDB to be generated when LCD transfer ends.
- Set LCDB data transfer cycle (example: “14”).
- Set LCDB data transfer wait cycle (example: “2”).

Figure 23-17 Flowchart of LCD BUS Function Setting

<Mode68>



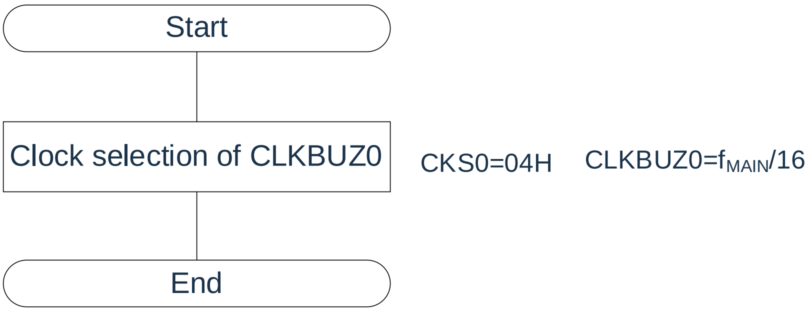
<Mode80>



(3). CLKBUZ0 clock setting

- Set the CLKBUZ0 clock(example: f_{MAIN}/16)
- Set PA00 as CLKBUZ0 output

Figure 23-18 Flowchart of CLKBUZ0 clock setting



When system clock is higher, suitable CLKBUZ0 frequency divided is needed to satisfy the specification of PCF2119x (f_{OSC} = 120 to 450 kHz), or the specification of S1D15E00 (f_{OSC} = 40 kHz (TYP)). See LCD driver data sheet for details.

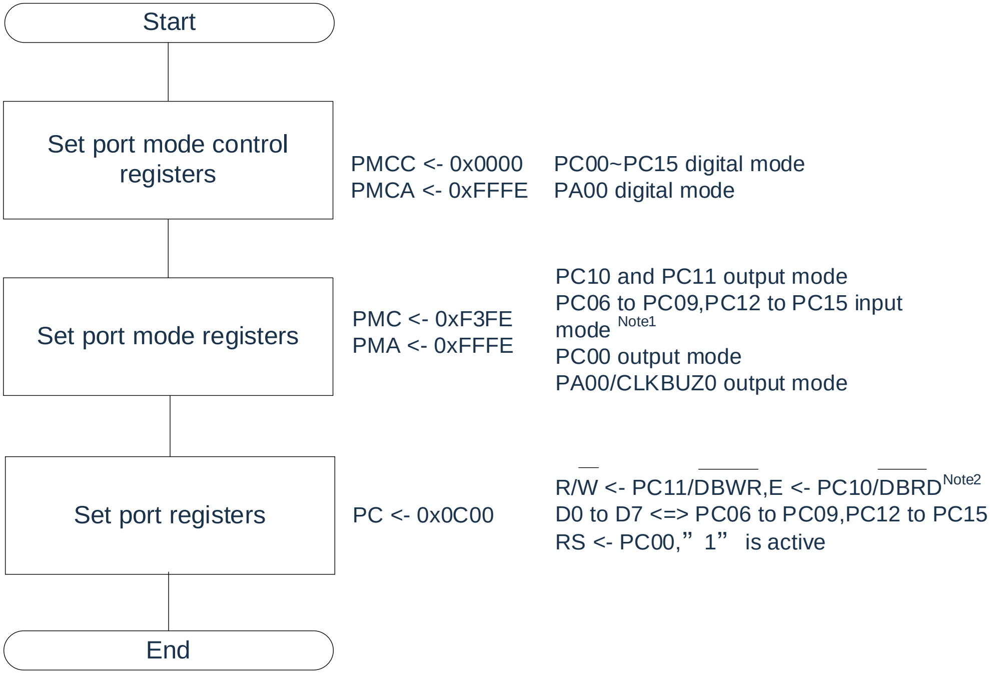
(4). LCD BUS port setting

With the following settings, LCDB is used as a bi-directional bus interface to communicate with the external LCD driver chip.

- Set the PMC register to configure the pin for digital mode.
- Set the pins multiplexed by PM registers $\overline{DBWR}$ and $\overline{DBRD}$ to be output mode.
- Set the pin output latch for pin registers $\overline{DBWR}$ and $\overline{DBRD}$ multiplexed to "1".
- Set PM register DBD0 to DBD7 multiplexed pins set to input mode. (When DBD0 to DBD7 are multiplexed for output, the hardware automatically switches to output mode and does not need to be set by the user)
- Set the pin output latch for pin registers DBD0 to DBD7 multiplexed to "0".

Figure 23-19 Flowchart of LCD BUS port setting

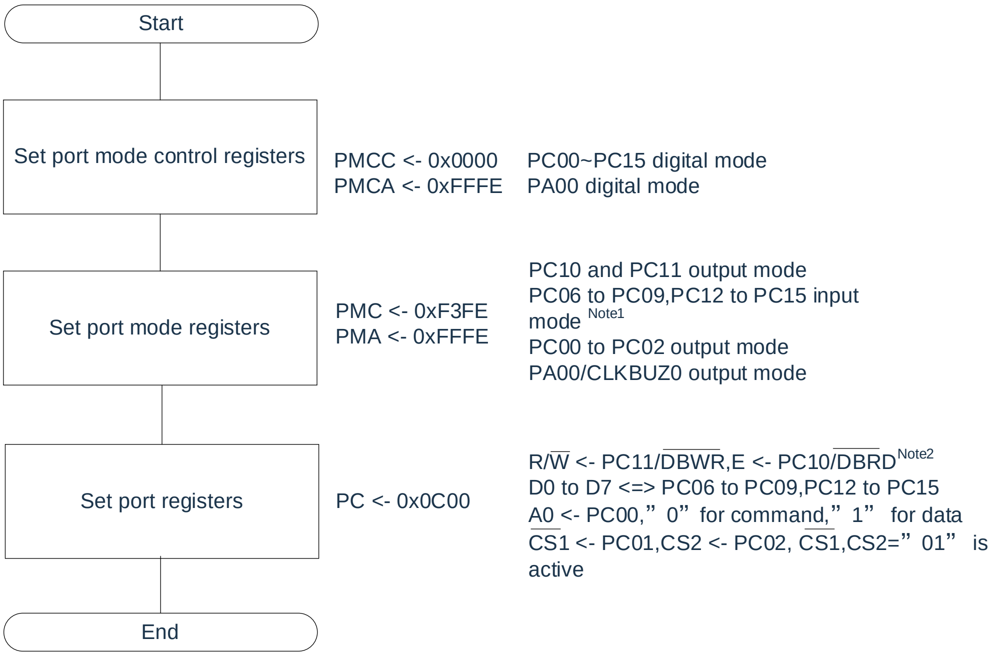
<Mode68(use PCF2119x) >


Start
Set port mode control registers
PMCC <- 0x0000
PMCA <- 0xFFFE
PC00~PC15 digital mode
PA00 digital mode
Set port mode registers
PMC <- 0xF3FE
PMA <- 0xFFFE
PC10 and PC11 output mode
PC06 to PC09,PC12 to PC15 input mode Note1
PC00 output mode
PA00/CLKBUZ0 output mode
Set port registers
PC <- 0x0C00
R/$\overline{W}$ <- PC11/$\overline{DBWR}$,E <- PC10/$\overline{DBRD}$ Note2
D0 to D7 <=> PC06 to PC09,PC12 to PC15
RS <- PC00,” 1” is active
End

Notice: 1. The PC06 to PC09 and PC12 to PC15 pins must be set to input mode in order to function as an LCD bidirectional communication bus.

2. When LBCTL.bit7(LBEL)=0, the PC10 and PC11 pin output latch registers must be set to "1".

<Mode80 (use S1D15E00) >


Start
Set port mode control registers
PMCC <- 0x0000
PMCA <- 0xFFFE
PC00~PC15 digital mode
PA00 digital mode
Set port mode registers
PMC <- 0xF3FE
PMA <- 0xFFFE
PC10 and PC11 output mode
PC06 to PC09,PC12 to PC15 input mode Note1
PC00 to PC02 output mode
PA00/CLKBUZ0 output mode
Set port registers
PC <- 0x0C00
R/$\overline{W}$ <- PC11/$\overline{DBWR}$,E <- PC10/$\overline{DBRD}$Note2
D0 to D7 <=> PC06 to PC09,PC12 to PC15
$\underline{A0}$ <- PC00,” 0” for command,” 1” for data
$\overline{CS1}$ <- PC01,CS2 <- PC02, $\overline{CS1}$,CS2=” 01” is active
End

Notice: 1. PC06 to PC09 and PC12 to PC15 must be set input mode to achieve bi-direction bus.

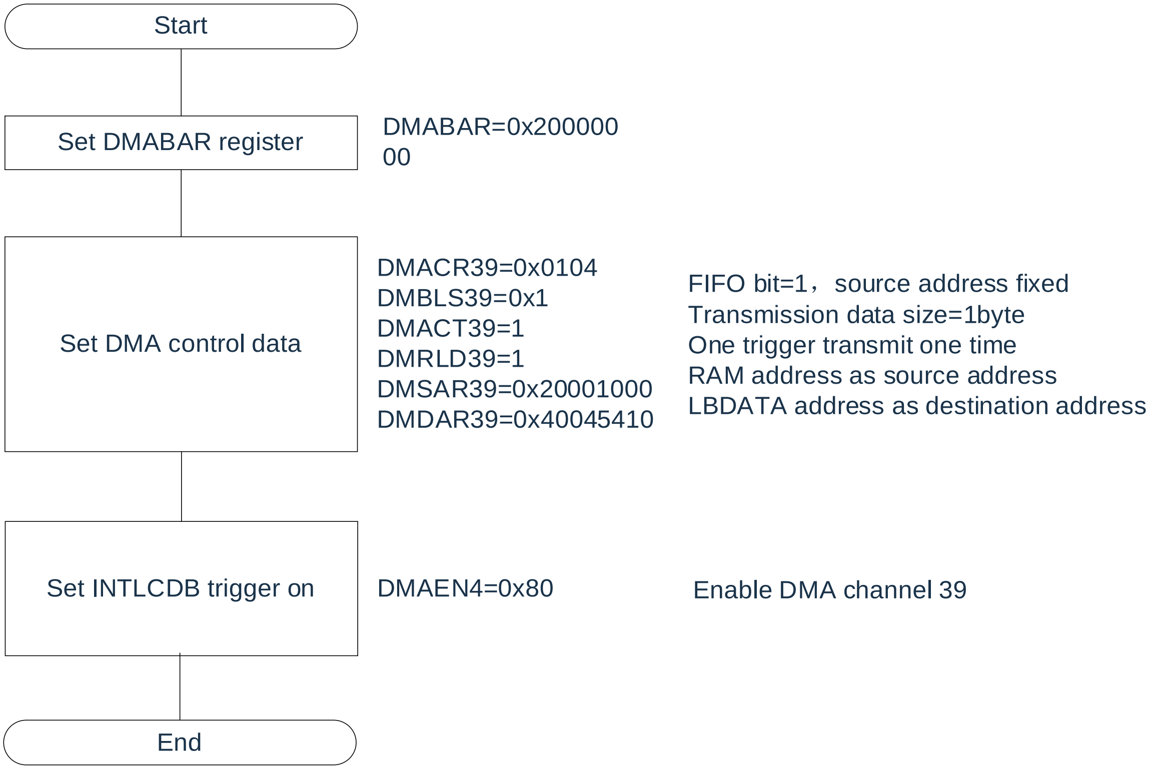
2. The PC10 and PC11 pin output latch registers must be set to "1" due to $\overline{\text{DBRD}}$ and $\overline{\text{DBWR}}$ are active low.

(5). DMA transfer setting

- Set DMABAR register
- Set the control data of DMA channel 39.
- - Set DMA channel 39 to start

Notice: For the first transfer, the contents of the transfer in 0x200001000 need to be written to the LBDATA register by normal write.

Figure 23-20 DMA transfer setup flow



When reading data from the LCD driver chip to the LCDB, set the RAM address to the target address and set the LBDATA address to the source address.

(6). LCD BUS transfer

- Use S1D15E00(EPSON)

The data/command transmitted via DMA should be beforehand stored in a RAM address. Setting content, for example, is shown below. Please refer to the data sheet of S1D15E00 for command details.

RAM address	Value	Command	Description
0x20001000	A0H	ADC select. SEG0→SEG131: 0(H) →Column address→83(H)	LCD driver initial (S1D15E0)
0x20001001	C0H	Common output mode select. Normal scanning direction of COM, COM0 → COM95	
0x20001002	A6H	Display normal/reverse. RAM data: HIGH Potential at LCD On (normal)	
0x20001003	A4H	Display all points ON/OFF Normal display mode.	
0x20001004	61H	Duty cycle set (2 byte) Set duty cycle of 1/8, starting point (block) is 0 (COM0 to 3)	
0x20001005	00H		
0x20001006	81H	Electronic volume (2 byte) The electronic volume register is set 05H (small)	
0x20001007	05H		
0x20001008	40H	Temperature gradient set Temperature gradient is 0.06%/°C.	
0x20001009	8AH	Display starting line set (2 byte) Display starting line is set to 0.	Display setting
0x2000100A	00H		
0x2000100B	B0H	Set the page address The page address is set to 0.	
0x2000100C	10H	Set the column address The higher 4 bits of the display data is 0000B, the	

		lower 4 bits is default (0000B)	
0x2000100D	7FH	Write the display data	The display data is “E”
0x2000100E	49H		
0x2000100F	49H		
0x20001010	49H		
0x20001011	41H		
0x20001012	AFH	Turn ON display	Display ON

- Use PCF2119x(NXP Semiconductors)

Here, only describes the initial routine of LCD driver, the DMA part is omitted, please refer to the former example of S1D15E00. Please refer to the data sheet of PCF2119x for command details.

Value	Command	Description
34H	Function set [Note] 8 bits data length, 2 line×16 characters, 1:18 multiplex drive mode.	LCD driver initialization (PCF2119x)
34H	Function set [Note]	
34H	Function set [Note]	
34H	Function set [Note]	
08H	Display control Display, cursor and character blink are off.	
01H	Clear display Fixed value.	
07H	Entry mode set Address increments by 1, display shifts	

Notice: Same instruction is specified to ensure enough BF checked time.

The flow of LCD BUS transmission without DMA is following:

It takes about 330 μ s (165 driver oscillator cycles) to finish clear display command, but other commands need about 6 μ s (3 driver oscillator cycles) when f_{OSC} = 450 kHz.

Busy flag check operation is carried in PCF2119x. The Busy Flag (BF) indicates the busy state (bit BF = 1) until initialization ends. The busy state lasts 2 ms. The busy flag is output to pin DB7 when RS=0 and R/$\overline{W}$=1.

Pin DB7 of LCD BUS can be used as the busy flag, by reading bit7 of LBDATA/LBDATAR, we can judge whether the driver internal operations are completed or not.

Chapter 24 Interrupt Function

The Cortex-M0+ processor has a built-in nested vector interrupt controller (NVIC), which supports up to 32 interrupt request (IRQ) inputs and one non-maskable interrupt (NMI) input, in addition to multiple internal exceptions.

In this system, the interrupt sources for 32 interrupt request (IRQ) inputs and one non-maskable interrupt (NMI) input are extended for up to 64 interrupt sources and 1 non-maskable interrupt sources. This user's manual only explains the processing in this system. Please refer to the user's manual of Cortex-M0+ processor for the functions of the built-in NVIC of Cortex-M0+ processor.

The actual number of interrupt sources varies by product

24.1 Types of interrupt function

There are 2 types of interrupt functions as follows.

(1) Maskable interrupt

This is a mask-controlled interrupt. If the interrupt mask flag register is not opened, the interrupt request will not be responded even if it is generated

It can generate standby release signals to release deep sleep mode and sleep mode.

Maskable interrupts are divided into external interrupt requests and internal interrupt requests.

(2) Non-maskable interrupt

This is an interrupt that is not controlled by masking. Once an interrupt request is generated, the CPU must respond to it.

24.2 Interrupt source and structure

Refer to Table 24-1 for a list of interrupt sources.

Table 24-1 List of interrupt sources(1/4)

Interrupt handling	Interrupt source No.	Interrupt source		Internal/External	Basic structure type[Note 1]
		Name	Trigger		
Maskable	0	INTLVI	Voltage detection [Note 2]	Internal	(A)
	1	INTP0	Detection of pin input edges	External	(B)
	2	INTP1	Detection of pin input edges		
	3	INTP2	Detection of pin input edges		
	4	INTP3	Detection of pin input edges		
	5	INTUSBI	USB interrupt	Internal	(A)
	6	INTUSBR	USB asynchronous interrupt		
	7	INTST0/INTSSPI00/INTIIC00	UART0 transmit end or buffer null interrupt /SSPI00 transmit end or buffer null interrupt /IIC00 transmit end		
	8	INTSR0/INTSSPI01/INTIIC01	UART0 receive end/SSPI01 receive end or buffer null interrupt/IIC01 receive end		
	9	INTSRE0	A UART0 receive communication error occurred.		
	10	INTST1/INTSSPI10/INTIIC10	UART1 transmit end or buffer null interrupt /SSPI10 transmit end or buffer null interrupt /IIC10 transmit end		
	11	INTSR1/INTSSPI11/INTIIC11	UART1 receive end/SSPI11 transmit end or buffer null interrupt/IIC11 transmit end		
	12	INTSRE1	A UART1 receive communication error occurred.		

Note: 1. The basic structure types (A)~(D) correspond to the (A)~(D) in Figure 24-1.

2. This is the case when bit 7 (LVIMD) of the voltage detection level register (LVIS) is set to "0".

Table 24-1 List of interrupt sources(2/4)

Interrupt handling	Interrupt source No.	Interrupt source		Internal/External	Basic structure type[Note 1]
		Name	Trigger		
Maskable	13	INTST2/INTSSPI20/INTIIC20	UART2 transmit end or buffer null interrupt /SSPI20 transmit end or buffer null interrupt /IIC20 transmit end	Internal	(A)
	14	INTSR2/INTSSPI21/INTIIC21	UART2 receive end/SSPI21 receive end or buffer null interrupt /IIC21 receive end		
	15	INTSRE2	A UART2 receive communication error occurred.		
	16	INTIICA0	IICA0 communication end		
	17	INTIICA1	IICA1 communication end		
	18	INTTM00	Timer channel 00 count end or capture end		
	19	INTTM01	Timer channel 01 count end or capture end		
	20	INTTM02	Timer channel 02 count end or capture end		
	21	INTTM03	Timer channel 03 count end or capture end		
	22	INTAD	A/D conversion end		
	23	INTRTC	Fixed period/alarm consistency detection for real time clocks		
	24	INTKR	Detection of key return signal	External	(C)
	25	INTCMP0	Comparator detect 0	Internal	(A)
	26	INTCMP1	Comparator detect 1		
	27	INTTM10	Timer channel 10 count end or capture end		
	28	INTTM11	Timer channel 11 count end or capture end		
	29	INTTM12	Timer channel 12 count end or capture end		
	30	INTTM13	Timer channel 13 count end or capture end		
	31	INTFL	Flash programming end		

Note: 1. The basic structure types (A)~(D) correspond to the (A)~(D) in Figure 24-1.

Table 24-1 List of interrupt sources(3/4)

Interrupt handling	Interrupt source No.	Interrupt source		Internal/External	Basic structure type[Note 1]
		Name	Trigger		
Maskable	32	INTQSPI	qspi error interrupt	Internal	(A)
	33	INTP4	Detection of pin input edges	External	(B)
	34	INTP5			
	35	INTP6			
	36	INTP7			
	37	INTD0FIFO	USB DMA0 transfer request interrupt	Internal	(A)
	38	INTD0FIFO	USB DMA1 transfer request interrupt	Internal	(A)
	39	Reserved	-	-	-
	40	Reserved	-	-	-
	41	INTSPI0	High-speed SPIHS0 transfer end interrupt	Internal	(A)
	42	Reserved	-	-	-
	43	Reserved	-	-	-
	44	INTSPI1	High-speed SPIHS1 transfer end interrupt	Internal	(A)
	45	INTTM01H	Timer channel 01 count end ot capture end (when the high 8-bit timer is operating)	Internal	(A)
	46	INTTM03H	Timer channel 03 count end ot capture end (when the high 8-bit timer is operating)	Internal	(A)
	47	INTLCDB	LCDB transfer interrupt	Internal	(A)
	48	INTDIV	Divider calculation end	Internal	(A)
	49	Reserved	-	-	-
	50	INTSSIDMART		Internal	(A)
	51	INTSSIDMARX		Internal	(A)
	52	INTSSIDMATX		Internal	(A)
	53	INTSSIINTREQ		Internal	(A)
	54	Reserved	-	-	-
	55	INTIT	Detection of interval signals	Internal	(A)
	56	INTDOCD		Internal	(A)
	57	Reserved	-	-	-
	58	Reserved	-	-	-
	59	INTTM14	Timer channel 14 count end ot capture end	Internal	(A)
	60	INTTM15	Timer channel 15 count end ot capture end	Internal	(A)
	61	INTTM16	Timer channel 16 count end ot capture end	Internal	(A)
	62	INTTM17	Timer channel 17 count end ot capture end	Internal	(A)
	63	INTQSPIDMAREQ	Qspi transfer request interrupt	Internal	(A)

Note: 1.The basic structure types (A)~(D) correspond to the (A)~(D) in Figure 24-1.

Table 24-1 List of interrupt sources(4/4)

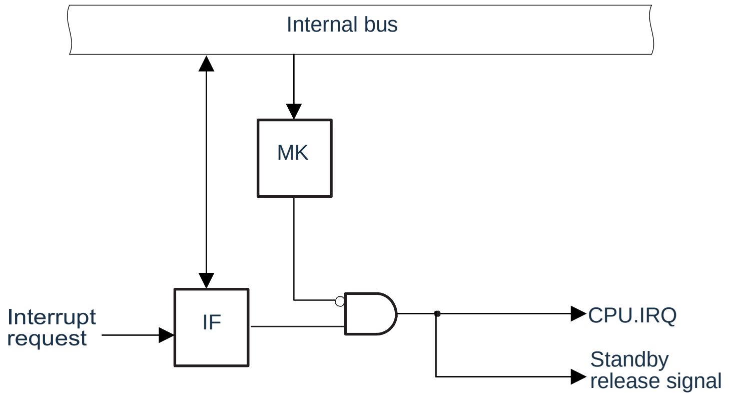
Interrupt handling	Interrupt source No.	Interrupt source		Internal/External	Basic structure type[Note 1]
		Name	Trigger		
Non-maskable	—	INTWDT	Watchdog timer interval interrupt [Note 2]	Internal	(D)

Note: 1. The basic structure types (A)~(D) correspond to the (A)~(D) in Figure 24-1.

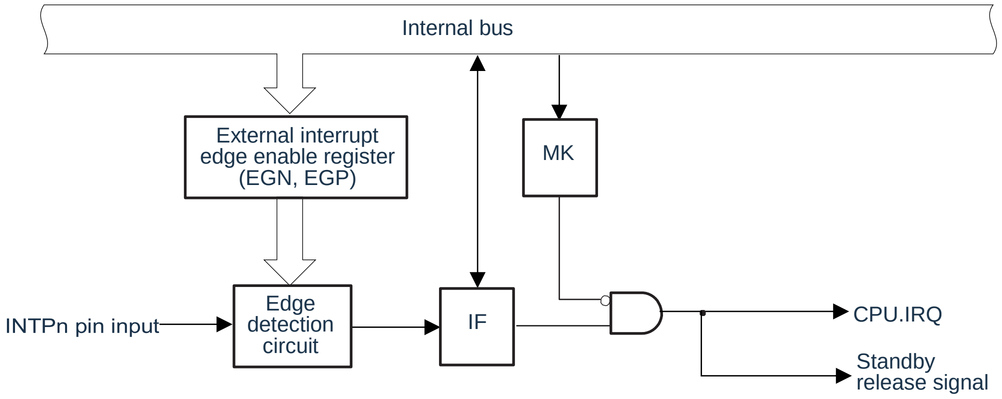
2. This is the case when bit7 (WDTINT) of the option byte (000C0H) is set to "1".

Figure 24-1 Basic structure of interrupt function

(A) Internal maskable interrupt

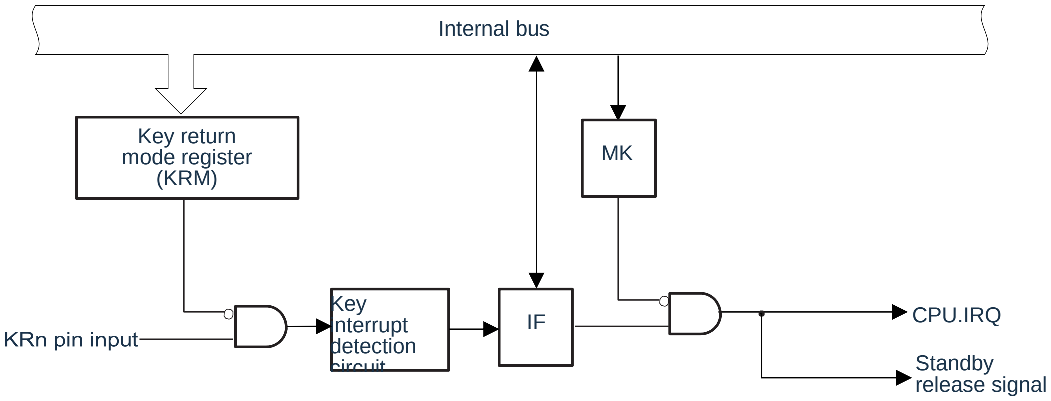


(B) External maskable interrupt (INTPn)



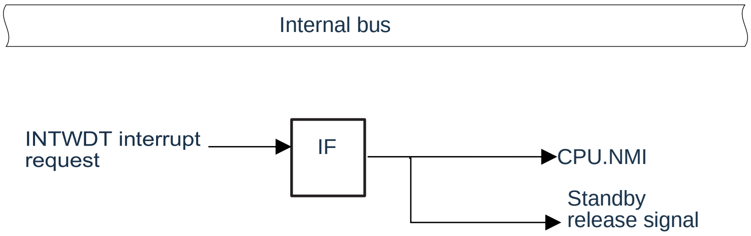
Note: n=0～7

(C) External maskable interrupt (INTKR)


Internal bus
Key return mode register (KRM)
MK
KRn pin input
Key interrupt detection circuit
IF
CPU.IRQ
Standby release signal

Note n=0～7

(D) Nnon-maskable interrupt



Note: The interrupt request flag IF of non-maskable interrupts does not have a physical register and cannot be used to generate interrupt requests by reading or writing registers on the bus.

24.3 Registers for controlling interrupt function

Interrupt function is controlled by the following four registers.

- Interrupt request flag register (IF00~IF31)
- Interrupt Mask Flag Register (MK00~MK31)
- External interrupt rising edge enable register (EGP0)
- External interrupt falling edge enable register (EGN0)

24.3.1 Interrupt request flag register (IF00~IF31)

The interrupt request flag is set to "1" by generating a corresponding interrupt request or executing an instruction. The interrupt request flag is cleared to "0" by generating a reset signal or by executing an instruction.

Set IF00L~IF31L , IF00H~IF31H registers by an 8-bit memory manipulation instruction. Or set IF00~IF31 registers by a 32-bit memory manipulation instruction.

After a reset signal is generated, the value of these registers becomes "0000_0000H".

Figure 24-2 Format of interrupt request flag register (IFm) (m=0~31)

Address: IF00: 40006000H, IF01: 40006004H, IF02: 40006008H, IF03: 4000600CH
IF04: 40006010H, IF05: 40006014H, IF06: 40006018H, IF07: 4000601CH
IF08: 40006020H, IF09: 40006024H, IF10: 40006028H, IF11: 4000602CH
IF12: 40006030H, IF13: 40006034H, IF14: 40006038H, IF15: 4000603CH
IF16: 40006040H, IF17: 40006044H, IF18: 40006048H, IF19: 4000604CH
IF20: 40006050H, IF21: 40006054H, IF22: 40006058H, IF23: 4000605CH
IF24: 40006060H, IF25: 40006064H, IF26: 40006068H, IF27: 4000606CH
IF28: 40006070H, IF29: 40006074H, IF30: 40006078H, IF31: 4000607CH
Reset value: 0000_0000H R/W

	31	30	29	28	27	26	25	24
	Reserved							
	23	22	21	20	19	18	17	16
	Reserved							
	15	14	13	12	11	10	9	8
IFmH	Reserved							IFH
	7	6	5	4	3	2	1	0
IFmL	Reserved							IFL

IFmL	Interrupt request flags 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 flags 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.

Note: 1. The correspondence between the interrupt source and the interrupt request flag register is shown in Table 24-2.

2. The correspondence between the interrupt request flag register and CPU.IRQ is shown in Figure 24-4.

3. The interrupt request flag register is not automatically cleared to zero and the register must be written to zero after an interrupt response.

24.3.2 Interrupt mask flag register (MK00~MK31)

The interrupt mask flag is set to enable or disable the corresponding maskable interrupt processing.

Set MK00L~MK31L, MK00H~MK31H registers by an 8-bit memory manipulation instruction or set MK00~MK31 registers by a 32-bit memory manipulation instruction.

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

Figure 24-3 Format of interrupt request mask register (MKm) (m=0~31)

Address: MK00: 40006100H, MK01: 40006104H, MK02: 40006108H, MK03: 4000610CH
MK04: 40006110H, MK05: 40006114H, MK06: 40006118H, MK07: 4000611CH
MK08: 40006120H, MK09: 40006124H, MK10: 40006128H, MK11: 4000612CH
MK12: 40006130H, MK13: 40006134H, MK14: 40006138H, MK15: 4000613CH
MK16: 40006140H, MK17: 40006144H, MK18: 40006148H, MK19: 4000614CH
MK20: 40006150H, MK21: 40006154H, MK22: 40006158H, MK23: 4000615CH
MK24: 40006160H, MK25: 40006164H, MK26: 40006168H, MK27: 4000616CH
MK28: 40006170H, MK29: 40006174H, MK30: 40006178H, MK31: 4000617CH
Reset value: FFFF_FFFFH R/W

	31	30	29	28	27	26	25	24
	Reserved							
	23	22	21	20	19	18	17	16
	Reserved							
	15	14	13	12	11	10	9	8
MKmH	Reserved							MKH
	7	6	5	4	3	2	1	0
MKmL	Reserved							MKL

MKmL	Interrupt processing control for interrupt sources numbered 0 to 31 [Note1]
0	Enable interrupt processing.
1	Disable interrupt processing.

MKmH	Interrupt processing control for interrupt sources numbered 32~63 [Note2]
0	Enable interrupt processing.
1	Disable interrupt processing.

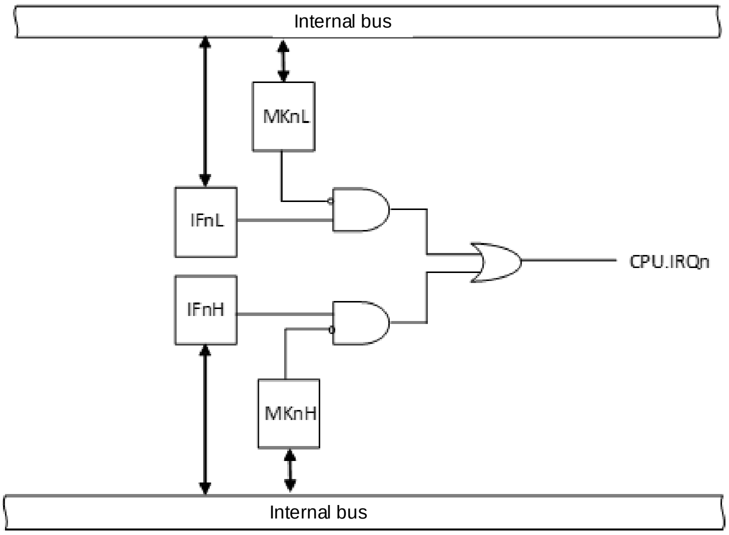
Note: 1. The correspondence between the interrupt source and the interrupt request mask register is shown in Table 24-2.

2. The correspondence between the interrupt request mask register and CPU.IRQ is shown in Figure 24-4.

Table 24-2 Correspondence between interrupt sources and flag registers

No.	Interrupt source	Interrupt request flag register	Interrupt mask flag register	No.	Interrupt source	Interrupt request flag register	Interrupt mask flag register
0	INTLVI	IF00.IFL	MK00.MKL	32	INTQSPI	IF00.IFH	MK00.MKH
1	INTP0	IF01.IFL	MK01.MKL	33	INTP4	IF01.IFH	MK01.MKH
2	INTP1	IF02.IFL	MK02.MKL	34	INTP5	IF02.IFH	MK02.MKH
3	INTP2	IF03.IFL	MK03.MKL	35	INTP6	IF03.IFH	MK03.MKH
4	INTP3	IF04.IFL	MK04.MKL	36	INTP7	IF04.IFH	MK04.MKH
5	INTUSBI	IF05.IFL	MK05.MKL	37	INTD0FIFO	IF05.IFH	MK05.MKH
6	INTUSBR	IF06.IFL	MK06.MKL	38	INTD1FIFO	IF06.IFH	MK06.MKH
7	INTST0/INTSSPI00/INTIIC00	IF07.IFL	MK07.MKL	39	Reserved	IF07.IFH	MK07.MKH
8	INTSR0/INTSSPI01/INTIIC01	IF08.IFL	MK08.MKL	40	Reserved	IF08.IFH	MK08.MKH
9	INTSRE0	IF09.IFL	MK09.MKL	41	INTSPI0	IF09.IFH	MK09.MKH
10	INTST1/INTSSPI10/INTIIC10	IF10.IFL	MK10.MKL	42	Reserved	IF10.IFH	MK10.MKH
11	INTSR1/INTSSPI11/INTIIC11	IF11.IFL	MK11.MKL	43	Reserved	IF11.IFH	MK11.MKH
12	INTSRE1	IF12.IFL	MK12.MKL	44	INTSPI1	IF12.IFH	MK12.MKH
13	INTST2/INTSSPI20/INTIIC20	IF13.IFL	MK13.MKL	45	INTTM01H	IF13.IFH	MK13.MKH
14	INTSR2/INTSSPI21/INTIIC21	IF14.IFL	MK14.MKL	46	INTTM03H	IF14.IFH	MK14.MKH
15	INTSRE2	IF15.IFL	MK15.MKL	47	INTLCDB	IF15.IFH	MK15.MKH
16	INTIICA0	IF16.IFL	MK16.MKL	48	INTDIV	IF16.IFH	MK16.MKH
17	INTIICA1	IF17.IFL	MK17.MKL	49	Reserved	IF17.IFH	MK17.MKH
18	INTTM00	IF18.IFL	MK18.MKL	50	INTSSIDMA	IF18.IFH	MK18.MKH
19	INTTM01	IF19.IFL	MK19.MKL	51	INTSSIDMA	IF19.IFH	MK19.MKH
20	INTTM02	IF20.IFL	MK20.MKL	52	INTSSIDMA	IF20.IFH	MK20.MKH
21	INTTM03	IF21.IFL	MK21.MKL	53	INTSSIINTR	IF21.IFH	MK21.MKH
22	INTAD	IF22.IFL	MK22.MKL	54	Reserved	IF22.IFH	MK22.MKH
23	INTRTC	IF23.IFL	MK23.MKL	55	INTIT	IF23.IFH	MK23.MKH
24	INTKR	IF24.IFL	MK24.MKL	56	INTDOCD	IF24.IFH	MK24.MKH
25	INTCMP0	IF25.IFL	MK25.MKL	57	Reserved	IF25.IFH	MK25.MKH
26	INTCMP1	IF26.IFL	MK26.MKL	58	Reserved	IF26.IFH	MK26.MKH
27	INTTM10	IF27.IFL	MK27.MKL	59	INTTM14	IF27.IFH	MK27.MKH
28	INTTM11	IF28.IFL	MK28.MKL	60	INTTM15	IF28.IFH	MK28.MKH
29	INTTM12	IF29.IFL	MK29.MKL	61	INTTM16	IF29.IFH	MK29.MKH
30	INTTM13	IF30.IFL	MK30.MKL	62	INTTM17	IF30.IFH	MK30.MKH
31	INTFL	IF31.IFL	MK31.MKL	63	INTQSPIDMAREQ	IF31.IFH	MK31.MKH

Figure 24-4　　Relationship between each flag register and CPU.IRQ



24.3.3 External interrupt rising edge enable register (EGP0), External interrupt falling edge enable register (EGN0)

These registers set the active edges of INTP0 to INTP7.

The EGP0 and EGN0 registers are set by an 8-bit memory operation instruction. After a reset signal is generated, the value of these registers changes to "00H".

Figure 24-5 Format of External interrupt rising edge enable register (EGP0), External interrupt falling edge enable register (EGN0)

Address: 40045B38H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
EGP0	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

EGPn	EGNn	Effective edge selection for INTPn pin (n=0 to 7)
0	0	Disable edge detection.
0	1	Falling edge
1	0	Rising edge
1	1	Rising and falling edges

The ports corresponding to the EGPn and EGNn bits are shown in Table 24-3.

Table 24-3 Interrupt request signals corresponding to the EGPn and EGNn bits

Detection enable bits		Interrupt request signal
EGP0	EGN0	INTP0
EGP1	EGN1	INTP1
EGP2	EGN2	INTP2
EGP3	EGN3	INTP3
EGP4	EGN4	INTP4
EGP5	EGN5	INTP5
EGP6	EGN6	INTP6
EGP7	EGN7	INTP7

Notice: If you switch the input port used by the external interrupt function to output mode, an INTPn interrupt may be detected. When switching to output mode, the port mode register (PMxx) must be set to "0" after disabling the detection edge (EGPn, EGNn=0, 0).

Remark: 1. Refer to "2.1 Port function" for the edge detection ports.

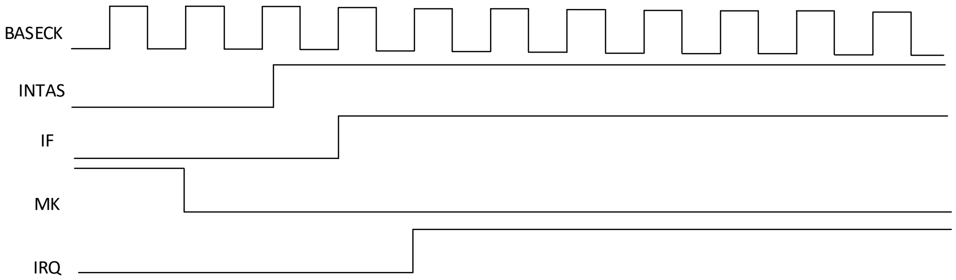
2. n=0～7

24.4 Operation of interrupt handling

24.4.1 Acceptance of maskable interrupt requests

If the interrupt request flag is set to "1" and the mask (MK) flag for the interrupt request is cleared to "0", the interrupt request is accepted and can be passed to the NVIC.

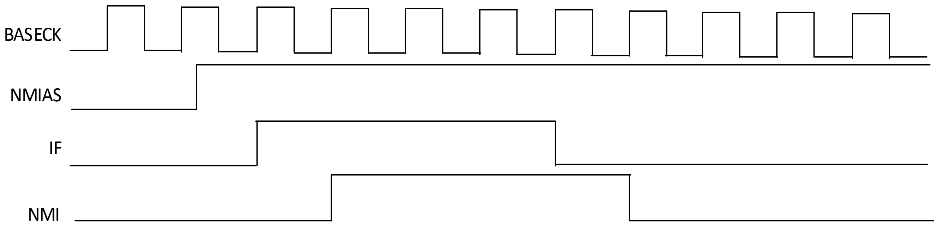
From the time the interrupt request flag is set to 1, to the time the CPU's IRQ is set to 1, it takes only 1 clock.



24.4.2 Acceptance of non-maskable interrupt requests

If a non-maskable interrupt request is generated, the interrupt request flag will be set to "1" and passed directly to the NVIC.

From the time the interrupt request flag is set to 1, to the time the CPU's NMI is set to 1, it takes only 1 clock.



Chapter 25 Key Interrupt Function

The number of channels for key interrupt input varies by product.

25.1 Function of key interrupt

A key interrupt (INTKR) can be generated by inputting a falling edge to the key interrupt input pins (KR0~KR7).

Table 25-1 Assignment of key interrupt detection pins

Key interrupt pin	Key return mode register (KRM)
KR0	KRM0
KR1	KRM1
KR2	KRM2
KR3	KRM3
KR4	KRM4
KR5	KRM5
KR6	KRM6
KR7	KRM7

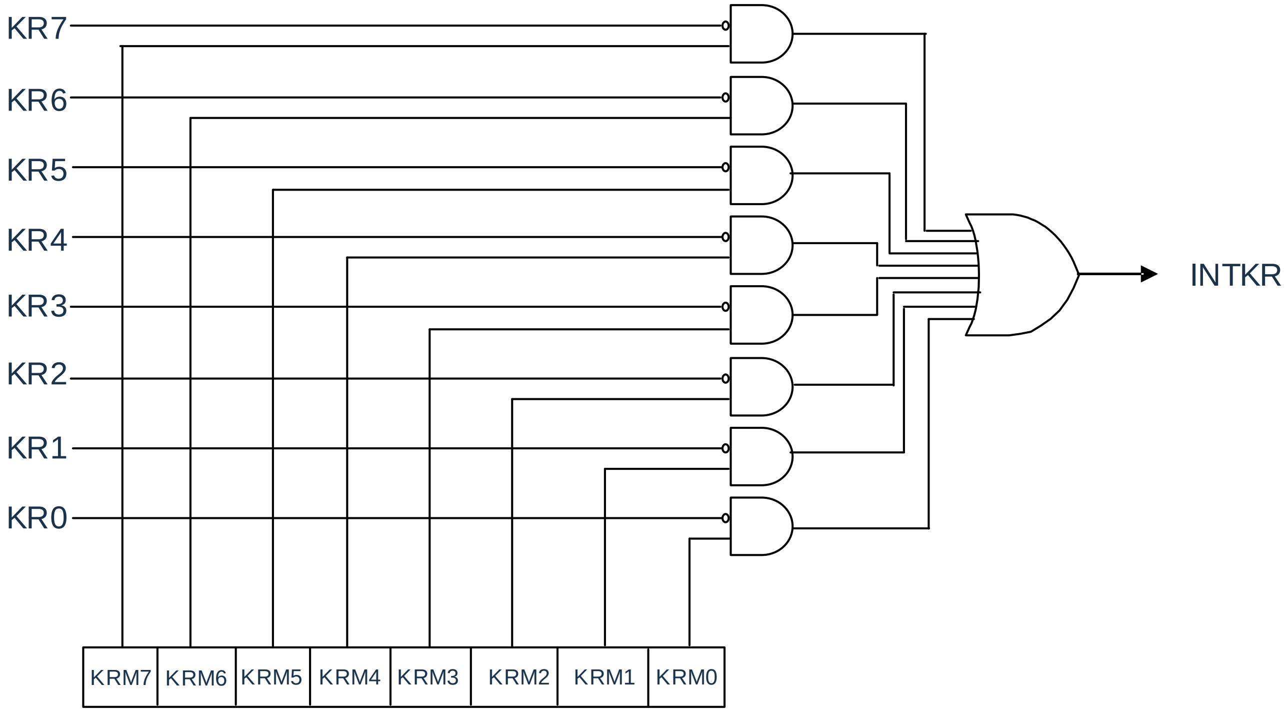
25.2 Structure of key interrupt

Key interrupts are made up of the following hardware.

Table 25-2 Structure of key interrupts

Item	Control register
Control register	Key return mode register (KRM) Port mode register (PMx). Port mode control register (PMCx).

Figure 25-1 Block diagram of key interrupt



Key return mode register

25.3 Registers for controlling key interrupts

The key interrupt function is controlled by the following registers.

- Key return mode register (KRM)
- Port mode register (PMx)

25.3.1 Key return mode register (KRM)

The KRM0~KRM7 bits control the KR0~KR7 signals.

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

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

Figure 25-2 Format of key return mode register (KRM)

Address: 40044B37H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
KRM	KRM7	KRM6	KRM5	KRM4	KRM3	KRM2	KRM1	KRM0

KRMn	Control of key interrupt mode
0	No key interrupt signal is detected.
1	Detects key interrupt signals.

Notice:

1. The internal pull-up resistor can be used by setting the object bit of the pull-up resistor register (PUx) of the key interrupt input pin to "1".
2. If the object bit of the KRM register is entered low on the input pin of the key interrupt, an interrupt is generated. To ignore this interrupt, the KRM register must be set after interrupt processing is disabled by the interrupt mask flag. The interrupt request flag must be cleared after waiting for the key interrupt input's low level width (T_{KR}) (see data sheet) to allow interrupt processing.
3. Unused pins in key interrupt mode can be used as the normal port.

Remark: 1.n=0～7

25.3.2 Port mode register (PMx)

When used as key interrupt input pins (KR0 to KR7), the PMCxx bits must be set to "0" and the PMxx bits to "1" respectively. At this time, the output latch of Pxx can be "0" or "1".

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

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

The internal pull-up resistor is used in bits via the pull-up resistor select register (PUx).

Refer to "2.3.1 Port mode register (PMxx)" for the format of the Port Mode Register.

Chapter 26 Standby Function

26.1 Standby function

The standby function is the function of further reducing the working current of the system, there are two modes below.

(1) Sleep mode

The sleep mode is a mode that stops the CPU running the clock. If a high speed system clock oscillation circuit, a high speed internal oscillator or a sub-system clock oscillation circuit are oscillating before setting a sleep mode, each clock continues to oscillate. While this mode does not allow the operation current to drop to the level of a deep sleep mode, it is a valid mode when you want to restart processing immediately through interrupt requests or when you want to perform frequent intermittent runs.

(2) Deep sleep mode

A deep sleep mode is a mode in which the oscillation of a high speed system clock oscillation circuit and a high speed internal oscillator are stopped and the whole system is stopped. The invention can greatly reduce the working current of the CPU.

Since that deep sleep mode can be released by interrupt request, intermittent operation can also be perform. However, in that case of the X1 clock, since a wait time for ensure oscillation stability is required to cancel the deep sleep mode, a sleep mode must be selected if immediate processing by interrupt request is required.

In either mode, the register, flag and data storage are all maintained as content before the standby mode and also the output latches and output buffers of input/output ports.

Note 1. Deep sleep mode is only available when the CPU is running at the main system clock. The CPU cannot be set to deep sleep mode when running at the secondary system clock. Sleep mode is used regardless of whether the CPU is running at the main system clock or the subsystem clock.

2. When transitioning to deep sleep mode, the WFI instruction must be executed after the peripheral hardware is stopped running at the main system clock.

3. In order to reduce the operation current of A/D converter, the bit7 (ADCS) and bit0 (ADCE) of A/D converter mode register 0 (ADM0) must be cleared to "0", and the WFI instruction must be executed after stopping the A/D converter running.

4. It is possible to choose whether to continue or stop the oscillation of the low-speed internal oscillator in the sleep mode or the deep sleep mode by the option byte. Refer to "Chapter 33 Option Bytes" for details.

26.2 Sleep mode

26.2.1 Sleep mode configuration

When the SLEEPDEEP bit of the SCR register is 0, the WFI instruction is executed and the sleep mode is started. In sleep mode, that CPU stop acting, but the value of the internal register is still maintain, and the peripheral module remain in the state before entering sleep mode. The states of peripheral modules, osillators, etc. in sleep mode are shown in Table 26-1.

Sleep mode can be set whether the CPU clock before setting is a high speed system clock or a high speed internal oscillator clock or a sub-system clock.

Note When the interrupt mask flag is "0" (which allows interrupt handling) and the interrupt request flag is "1" (which generates an interrupt request signal), the interrupt request signal is used to deactivate sleep mode. Therefore, even if the WFI command is executed in this case, it does not shift to sleep mode.

Table 26-1 Operation status in sleep mode (1/2)

Sleep mode setting Item			Execution of WFI instructions while the CPU is running at the main system clock		
			CPU runs with a high-speed internal oscillator clock (f_{IH}) operation or f_{IH}+PLL	CPU runs at X1 clock (f_X) or f_X+PLL	CPU runs on an external main system clock (f_{EX}) operation or f_{EX}+PLL
System clock			Stop to supply clocks to the CPU.		
	Main system clock	f_{IH}	Continues running (cannot be stopped).	Disable running.	
		f_X	Disable running.	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 oscillator clock	f_{IL}	It is set by bit0 (WDSTBYON) and bit4 (WDTON) of the option byte (000C0H) and the WUTMMCK0 bit of the Subsystem Clock Supply Mode Control Register (OSMC). WUTMMCK0=1: oscillate WUTMMCK0=0 and WDTON=0: stop WUTMMCK0=0, WDTON=1 and WDSTBYON=1: oscillate WUTMMCK0=0, WDTON=1 and WDSTBYON=0: stop		
PLL/UPLL			Remain in the state before sleep mode.		
CPU			Stop running.		
Code Flash					
RAM			Stop running (can run when DMA is executed).		
Port (latch)			Remain in the state before sleep mode.		
General-purpose timer unit			Can run.		
Real time clock (RTC)					
15-bit interval timer					
Watchdog timer			Refer to "Chapter 11 Watchdog Timer".		
Clock output/buzzer output			Can run.		
AD converter					
CMP/PGA					
General-purpose serial communication unit (SCI0~2)					
High-speed SPI(SPIHS0, 1)					
Serial interface (IICA0, 1)					
Data transfer controller (DMA)					
Linkage controller			Links between runnable function blocks.		
USBFS			Can run.		
QSPI					
Serial audio interface (SSI)					
LCD BUS interface					
Power-on reset function					
Voltage detection function					
External Interrupts					
CRC operation function	High-speed CRC				
	General CRC		It can be run when DMA is performed in the operation of the RAM area.		
RAM parity check function			It can be run when the DMA is performed.		
SFR protection function					

Remark: Stop running: Automatically stops running when shifting to sleep mode.
Disable running: Stops running before shifting to sleep mode.
f_{IH}: High-speed internal oscillator clock f_{IL}: Low-speed internal oscillator clock
f_X: X1 clock f_{EX}: External main system clock
f_{XT}: XT1 clock f_{EXS}: External subsystem clock

Table 26-1 Operation status in sleep mode (2/2)

Sleep mode setting Item			Execution of WFI instructions while the CPU is running at the subsystem clock	
			CPU running at XT1 clock (F_{XT})	CPU running on external subsystem clock (F_{EXS})
System clock			Stop to supply clocks to the CPU.	
	Main system Clock	f_{IH}	Disable running.	
		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 oscillator clock	f_{IL}	It is set by bit0 (WDSTBYON) and bit4 (WDTON) of the option byte (000C0H) and the WUTMMCK0 bit of the Subsystem Clock Supply Mode Control Register (OSMC). • WUTMMCK0=1: oscillate • WUTMMCK0=0 and WDTON=0: stop • WUTMMCK0=0, WDTON=1 and WDSTBYON=1: oscillate • WUTMMCK0=0, WDTON=1 and WDSTBYON=0: stop	
PLL/UPLL			Remain in the state before sleep mode.	
CPU			Stop running.	
Code Flash				
RAM			Stop running (can run when DMA is executed).	
Port (latch)			Remain in the state before sleep mode.	
General-purpose timer unit			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 10 Watchdog Timer”.	
Clock output/buzzer output			When RTCLPC=0, it can run (otherwise it is prohibited).	
AD converter			Disable running.	
CMP/PGA			Can run.	
General-purpose serial communication unit (SCI0~2)			When RTCLPC=0, it can run (otherwise it is prohibited).	
High-speed SPI(SPIHS0, 1)			Disable running.	
Serial interface (IICA0, 1)			When RTCLPC=0, it can run (otherwise it is prohibited).	
Linkage controller			Links can be made between runnable function blocks.	
USBFS			Disable running.	
QSPI			Disable running.	
Serial audio interface (SSI)			Disable running.	
LCD BUS interface			Disable running.	
Power-on reset function			Can run.	
Voltage detection function				
External Interrupts				
CRC operation function	High-speed CRC		Disable running.	
	General CRC		It can be run when DMA is performed in the operation of the RAM area.	
RAM parity check function			It can be run when the DMA is performed.	
SFR protection function				

Remark: Stop running: Automatically stops running when shifting to sleep mode.

Disable running: Stops running before shifting to sleep mode.

f_{IH}: High-speed internal oscillator clock f_{IL}: Low-speed internal oscillator clock

f_X: X1 clock f_{EX}: External main system clock

f_{XT}: XT1 clock f_{EXS}: External subsystem clock

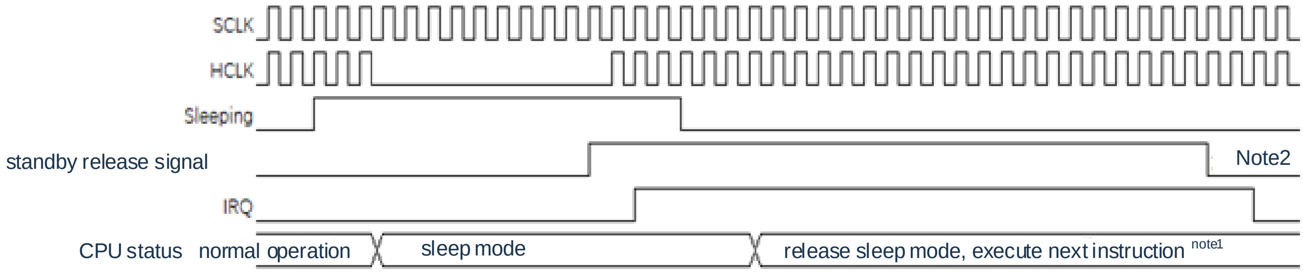
26.2.2 Release of sleep mode

Sleep mode can be released by any interrupt and external reset terminal, POR reset, low voltage detect reset, RAM parity error reset, WDT reset, software reset.

(1) Released by interrupts

When an unmasked interrupt is generated and is in a state that allows interrupts to be accepted, the sleep mode is released and the CPU begins to process the interrupt service program.

Figure 26-1 Release sleep mode by interrupt requests



Note: 1. From the generation of the standby release signal to the release of sleep mode, it takes 16 clocks to start executing the interrupt service program.

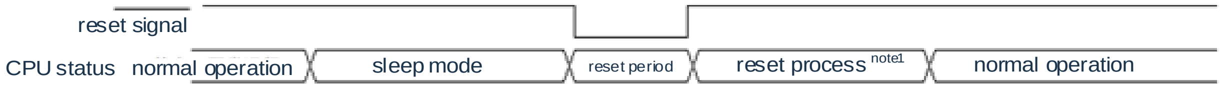
2. The standby release signal cannot clear itself; you must write a register to clear it. It is usually cleared by writing a register in the interrupt service program.

Notice: Before entering sleep mode, only the mask bit corresponding to the interrupt expected to be used to release sleep mode should be cleared to zero.

(2) Released by resetting

When a reset signal is generated, the CPU is in the reset state and the sleep mode is released. As with a normal reset, the program is executed after transferring to the reset vector address.

Figure 26-2 Release sleep mode by resetting



Note 1: For the reset processing, please refer to “Chapter 27 Reset Function ”. For reset processing for power-on reset (POR) circuits and voltage detection (LVD) circuits, refer to “Chapter 28 Power-on Reset Circuit”.

26.3 Deep sleep mode

26.3.1 Deep sleep mode configuration

When the SLEEPDEEP bit of the SCR register is 1, the WFI instruction is executed and deep sleep mode is entered. In this mode, the CPU, most of the peripheral modules, and the oscillator stop running. 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 oscillator in deep sleep mode is shown in Table 26-2.

Deep sleep mode can only be set if the CPU clock before setting is the main system clock.

Notice: When the interrupt mask flag is "0" (allows interrupt processing) and the interrupt request flag is "1" (generating an interrupt request signal), the interrupt request signal is used to release deep sleep mode. Therefore, if the WFI instruction is executed in this case, it is released as soon as it enters deep sleep mode. Returns to operation mode after executing the WFI instruction and after a deep sleep mode release time has elapsed.

Table 26-2 Operation status in deep sleep mode

Deep sleep mode setting Item			Execution of WFI instructions while the CPU is running at the main system clock		
			CPU runs with a high-speed internal oscillator clock (f_{IH}) operation or f_{IH}+PLL	CPU runs at X1 clock (f_X) or f_X+PLL	CPU runs on an external main system clock (f_{EX}) operation or f_{EX}+PLL
System clock			Stop to supply clocks to the CPU.		
	Main system Clock	f_{IH}	Stop		
		f_X			
		f_{EX}			
	Subsystem Clock	f_{XT}	Remain the state before it was set to deep sleep mode.		
		f_{EXS}			
	f_{IL}		Set by bit0 (WDSTBYON) and bit4 (WDTON) of the option byte (000C0H) and the WUTMMCK0 bit of the Subsystem Clock Supply Mode Control Register (OSMC). WUTMMCK0=1: oscillate WUTMMCK0=0 and WDTON=0: stop WUTMMCK0=0, WDTON=1 and WDSTBYON=1: oscillate WUTMMCK0=0, WDTON=1 and WDSTBYON=0: stop		
PLL/UPLL			Stop running.		
CPU			Stop running.		
Code Flash					
RAM					
Port (latch)			Remain the state before it was set to deep sleep mode.		
General-purpose timer unit			Disable running.		
Real time clock (RTC)			Can run.		
15-bit interval timer					
Watchdog timer			Refer to "Chapter 11 Watchdog Timer".		
Clock output/buzzer output			Operation is enabled when the subsystem clock is selected as the count clock and the RTCLPC bit is "0" (otherwise, operation is disabled).		
AD converter			Can perform a wake-up call.		
CMP/PGA			Can run (only without digital filters).		
General-purpose serial communication unit (SCI0~2)			Disable running.		
High-speed SPI(SPIHS0, 1)			Disable running.		
Serial interface (IICA0, 1)			Can wake up by address matching.		
Data transfer controller (DMA)			Can accept DMA boot sources.		
Linkage controller			Links can be made between runnable function blocks.		
USBFS			Disable running.		
QSPI			Disable running.		
Serial audio interface (SSI)			Disable running.		
LCD BUS interface			Disable running.		
Power-on reset function			Can run.		
Voltage detection function					
External Interrupts					
CRC operation function	High-speed CRC		Stop running.		
	General CRC				
RAM parity check function					
SFR protection function					

Remark: Stop running: Automatically stops running when shifting to sleep mode.
Disable running: Stops running before shifting to sleep mode.
f_{IH}: High-speed internal oscillator clock f_{IL}: Low-speed internal oscillator clock
f_X: X1 clock f_{EX}: External main system clock
f_{XT}: XT1 clock f_{EXS}: External subsystem clock

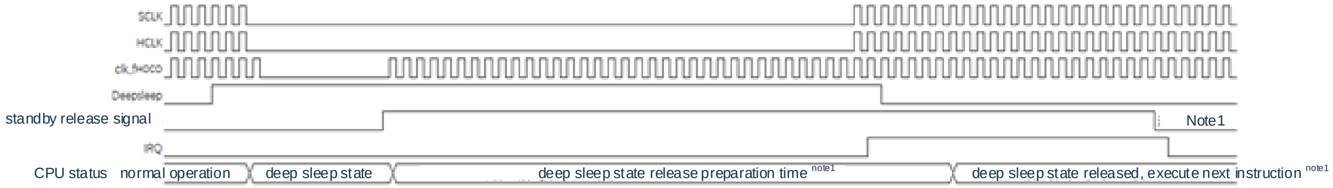
26.3.2 Release of deep sleep mode

Deep sleep mode can be released in 2 ways.

(a) Released by unmasked interrupt request

If an unmasked interrupt request occurs, deep sleep mode is released. After the oscillation stabilization time, if the interrupt is allowed to be accepted, the vector interrupt is processed. If the interrupt acceptance is disabled, the next address is executed.

Figure 26-3 Release deep sleep mode by interrupt requests


SCLK
HCLK
clk_fHOCO
Deepsleep
standby release signal
Note1
IRQ
CPU status
normal operation
deep sleep state
deep sleep state release preparation time note1
deep sleep state released, execute next instruction note1

Note: 1. Standby release signal: For details of the standby release signal, please refer to “Figure 24-1 Basic structure of interrupt function”.

2. Deep sleep release preparation time:

When the CPU clock is a high-speed internal oscillation clock or an external clock input before entering deep sleep mode: at least 20us

When entering deep sleep mode before the CPU clock is a high-speed system clock (X1 oscillation): At least 20us and a longer time in the oscillation settling time (set by OSTS).

Additional LOCKUP time is required when the CPU clock is PLL clock before entering deep sleep mode.

3. Wait: 14 clocks are required from the time CPU.IRQ is valid to the interrupt service program start.

Notice:

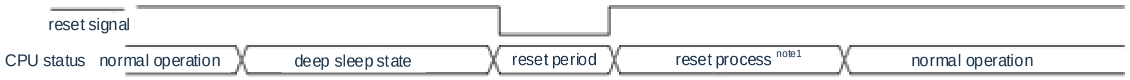
1. Before entering sleep mode, only the mask bits corresponding to the interrupts expected to be used to release sleep mode should be cleared to zero.
2. When the CPU is running at high speed system clock (X1 oscillation) and to shorten the oscillation stabilization time after the deep sleep mode is released, the CPU clock must be temporarily switched to the high-speed internal oscillator clock before the WFI instruction is executed.

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

(b) Released by generating a reset signal

The deep sleep mode is released by generating a reset signal. Then, as with a normal reset, the program is executed after transferring to the reset vector address.

Figure 26-4 Release deep sleep mode by resetting



Note: For reset processing, see “Chapter 27 Reset Function”. For reset processing for power-on reset (POR) circuits and voltage detection (LVD) circuits, see “Chapter 28 Power-on Reset Circuit”.

Chapter 27 Reset Function

The following 7 methods generate a reset signal.

1) External reset input via the RESETB pin.
2) An internal reset is generated by program runaway detection of the watchdog timer.
3) An internal reset is generated by comparing the power supply voltage and the detection voltage of the POR circuit.
4) An internal reset is generated by comparing the power supply voltage of the voltage detection circuit (LVD) with the detection voltage.
5) Internal reset occurs because the system reset request register bit (AIRCR.SYSRESETREQ) is set to 1.
6) Internal reset occurs due to RAM parity error.
7) Internal reset occurs due to access to the illegal memory.

The internal reset and the external reset are the same, and after the reset signal is generated, the program is executed from the user-defined program start address.

When a low level is applied to the RESETB pin, or a program runaway is detected by the watchdog timer, or voltages are detected in the POR and LVD circuits, or the system reset request bit is set, or a RAM parity check error occurs, or an illegal memory is accessed, a reset is generated and each hardware changes to the state shown in Table 27-1.

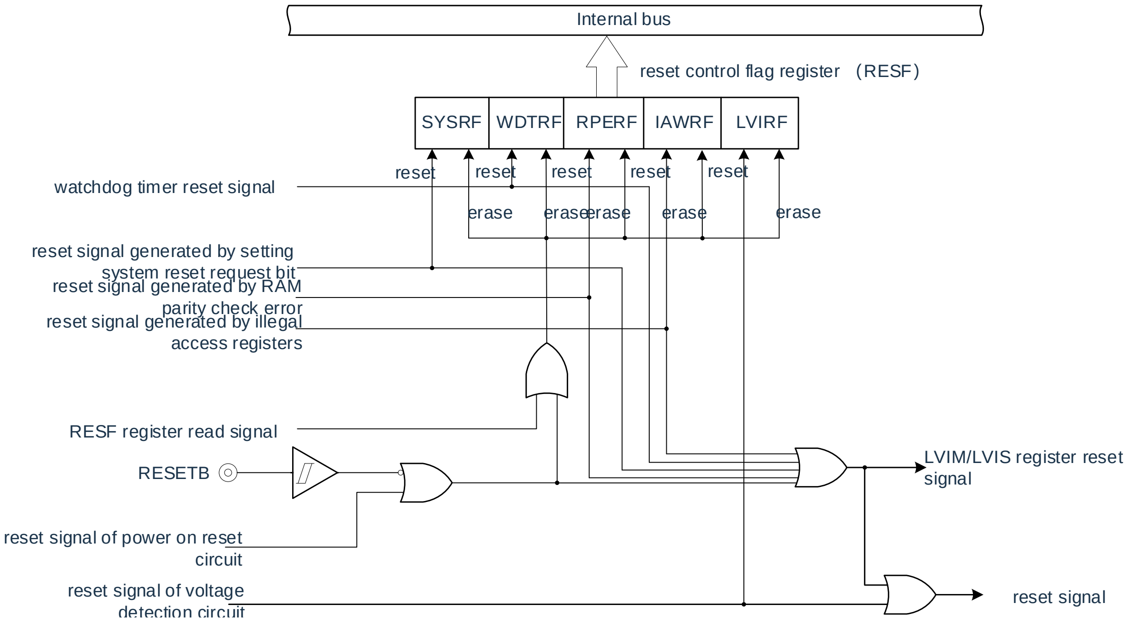
Note 1. When an external reset is performed, a low level of at least 10 μs must be input to the RESETB pin. If an external reset is performed while the supply voltage is rising, the power must be turned on after inputting a low level to the RESETB pin, and it must be held low for at least 10 μs over the operating voltage range shown in the AC Characteristics of the User's Manual, and then input a high level.

2. The oscillation of the X1 clock, XT1 clock, high speed internal oscillator clock and low speed internal oscillator clock is stopped during reset signal generation. Invalid input for external master and external sub system clocks.

3. If a reset occurs, each SFR is initialized, so the port pins change to the following state:

- PA10, PB03, PD00, PH01: High impedance during external reset or POR reset. High during other resets and after accepting a reset (connect internal pull-up resistor).
- Ports other than PPA10, PB03, PD00, PH01: High impedance during reset and after accepting reset.

Figure 27-1 Block diagram for reset function


Internal bus
reset control flag register (RESF)
SYSRF
WDTRF
RPERF
IAWRF
LVIRF
reset
reset
reset
reset
reset
erase
erase
erase
erase
erase
watchdog timer reset signal
reset signal generated by setting system reset request bit
reset signal generated by RAM parity check error
reset signal generated by illegal access registers
RESF register read signal
RESETB
reset signal of power on reset circuit
reset signal of voltage detection circuit
LVIM/LVIS register reset signal
reset signal

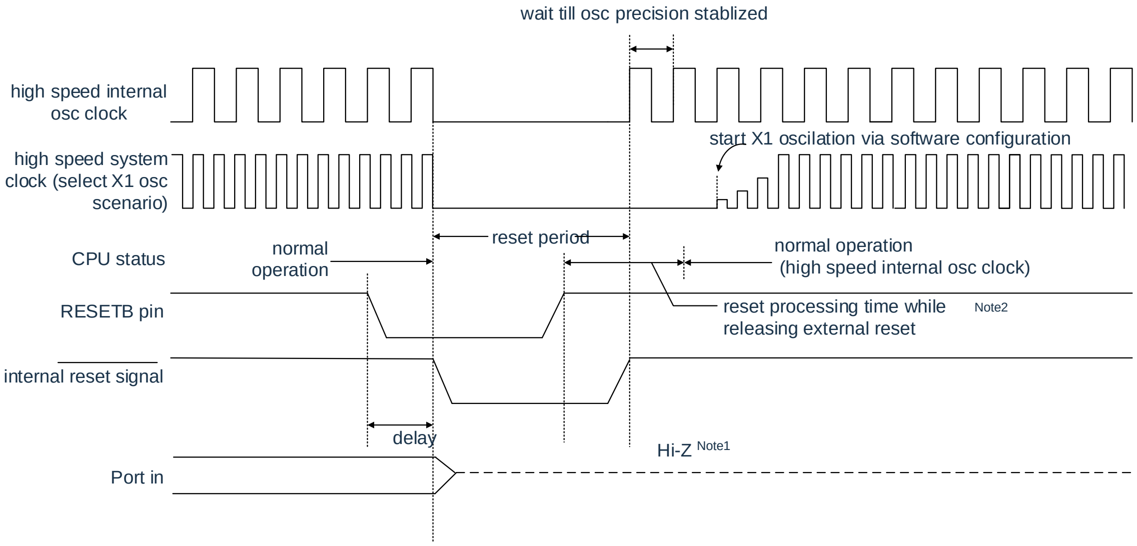
Notice: The internal reset of the LVD circuit does not reset the LVD circuit.

Remark:
1. LVIM: Voltage detection register
2. LVIS: Voltage detection level register

Reset timing

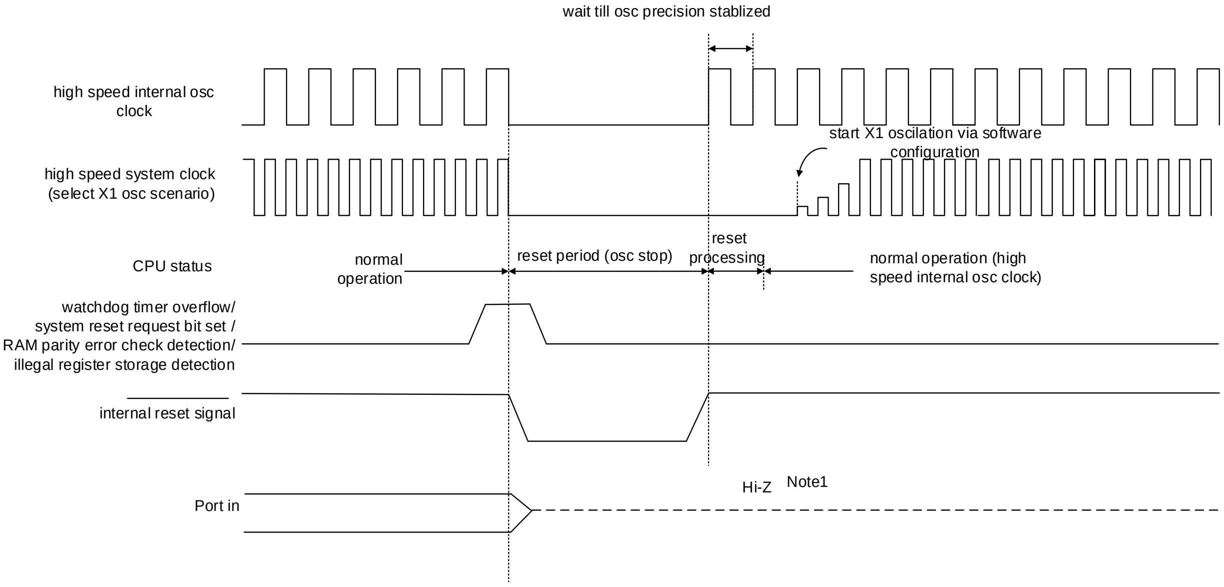
Reset occurs when a low level is input to the RESETB pin. Then, if a high level is inputted to the RESETB, the reset state is released and the program is started to execute with a high speed internal oscillator clock after the reset process is completed.

Figure 27-2 Reset timing for RESETB input


wait till osc precision stablized
high speed internal osc clock
high speed system clock (select X1 osc scenario)
start X1 oscilation via software configuration
CPU status
normal operation
reset period
normal operation
(high speed internal osc clock)
RESETB pin
reset processing time while releasing external reset
Note2
internal reset signal
delay
Hi-Z Note1
Port in

The reset state is automatically released for the reset caused by the overflow of the watchdog timer, the setting of the system reset request bit, the detection of the RAM parity error or the detection of the illegal memory access, and the program is started by the high speed internal oscillator clock after the reset process is finished.

Figure 27-3 Reset timing due to overflow of watchdog timer, set of system reset request bits, detection of RAM parity errors, or detection of illegal memory access


wait till osc precision stablized
high speed internal osc clock
high speed system clock (select X1 osc scenario)
start X1 oscilation via software configuration
CPU status
normal operation
reset period (osc stop)
reset processing
normal operation (high speed internal osc clock)
watchdog timer overflow/ system reset request bit set / RAM parity error check detection/ illegal register storage detection
internal reset signal
Hi-Z Note1
Port in

Note: 1. Ports PA10, PB03, PD00, PH01 become the following states:

- High impedance during external reset or POR reset.
- High during other resets and after accepting a reset (connect internal pull-up resistor)

Notice: The watchdog timer is no exception and is reset when an internal reset occurs.

If $V_{DD} \geqslant V_{POR}$ or $V_{DD} \geqslant V_{LVD}$ is satisfied after the reset for the reset generated by the voltage detection of the POR and LVD circuits, the reset state is released and the program execution starts with the high-speed internal oscillator clock after the reset is processed. For details, refer to "Chapter 28 Power-on Reset Circuit" and "Chapter 29 Voltage Detection Circuit".

Remark: V_{POR}: POR supply voltage rise detection voltage

V_{LVD}: LVD detection voltage

Table 27-1 Operation status during reset

Item			During resetting
System clock			Stop to supply clocks to CPU.
	Main system clock	f_{IH}	Stop running.
		f_X	Stop running (X1 pin and X2 pin are in input port mode).
		f_{EX}	Invalid clock input (pin is in input port mode).
	Sub system clock	f_{XT}	Can run.
		f_{EXS}	Clock input is invalid (pin is in input port mode).
	f_{IL}		Stop running.
PLL/UPLL			
CPU			
Code Flash			Stop running.
RAM			Stop running.
Port (Latch)			High impedance [Note1]
General-purpose timer unit			Stop running.
RTC			
15-bit interval timer			
Watchdog timer			
Clock output/Buzzer output			
A/D converter			
CMP [Note1]			
General-purpose serial communications unit (SCI)			
Serial interface (IICA)			
Data transfer controller (DMA)			
Power-on reset function			Can perform detection runs.
Voltage detection function			Can run on LVD reset. On other resets, stops running.
External interrupt			Stop running.
Key interrupt function			
CRC operation function	High speed CRC		
	General CRC		
RAM parity check function			
SFR protection function			

Note: 1. Ports PA10, PB03, PD00, PH01 become the following states:
High impedance during external reset or POR reset. High during other resets (connect internal pull-up resistor).

Remark

f_{IH}	: High-speed internal oscillator clock	f_X	: X1 oscillator clock
f_{EX}	: External main system clock	f_{XT}	: XT1 oscillator clock
f_{EXS}	: External subsystem clock	f_{IL}	: Low-speed internal oscillator clock

27.1 Registers for confirming the reset source

27.1.1 Reset control flag register (RESF)

The BAT32G157 microcontroller has a variety of internal reset generation sources. The reset control flag register (RESF) holds the reset source where the reset request occurs. The RESF register can be read by an 8-bit memory operation instruction.

The SYSRF, WDTRF, RPERF, IAWRF and LVIRF flags are cleared through the input of RESETB, reset of POR circuit and read of RESF register. To determine the reset source, you must save the value of the RESF register to any RAM and then determine it by its RAM value.

Figure 27-4 Format of reset control flag register (RESF)

Address: 40020440H After reset: Indefinite value Note 1 R

Symbol	7	6	5	4	3	2	1	0
RESF	SYSRF	0	0	WDTRF	0	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	An internal reset request is generated.

WDTRF	An internal reset request generated by watchdog timer (WDT)
0	No internal reset request was generated or the RESF register was cleared.
1	An internal reset request is generated.

RPERF	An internal reset request due to RAM parity error
0	No internal reset request was generated or the RESF register was cleared.
1	An internal reset request is generated.

IAWRF	An internal reset request generated by access illegal memory
0	No internal reset request was generated or the RESF register was cleared.
1	An internal reset request is generated.

LVIRF	An internal reset request generated by a voltage detection circuit (LVD)
0	No internal reset request was generated or the RESF register was cleared.
1	An internal reset request is generated.

Note 1. Different due to reset sources. Refer to Table 27-2.

Notice: When RAM parity error reset (RPERDIS=0) is allowed, the "RAM AREA USED" must be initialized when accessing data; When executing instructions from the RAM area, the area of "RAM area used +10 bytes" must be initialized. By generating a reset, a state allowing generation of a RAM parity error reset (RPERDIS=0) is entered. Refer to "30.3.3 RAM Parity Error Detection" for details.

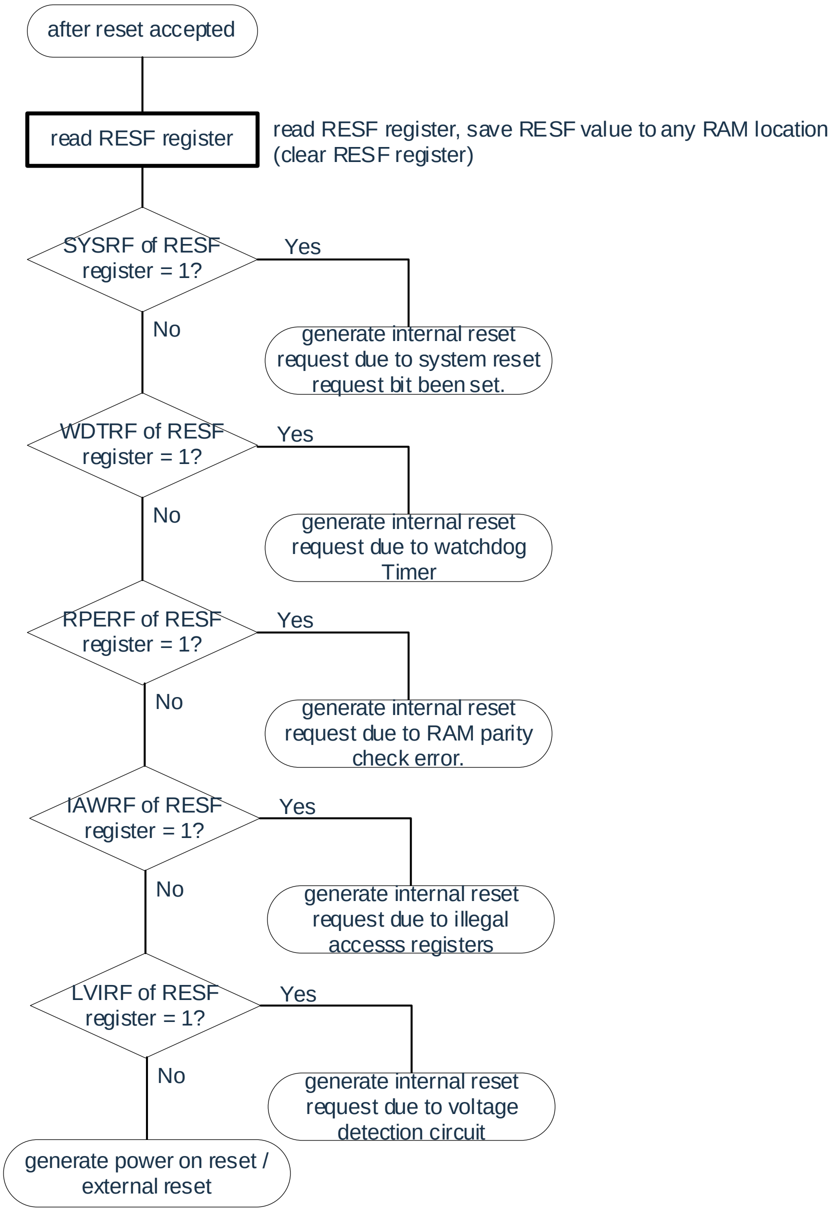
The RESF register status at the time of the reset request is shown in Table 27-2.

Table 27-2 RESF register state when reset request occurs

Reset source Flag	RESETB input	Reset generated by POR	Reset generated by system reset request bit set	Reset generated by WDT	Reset generated by RAM parity error	Reset generated by accessing illegal memory	Reset generated by LVD
SYSRF	Clear to “0”	Clear to “0”	Set to “1”	Keep	Keep	Set to “1”	Set to “1”
WDTRF			Keep	Set to “1”			
RPERF				Keep	Set to “1”		
IAWRF					Keep	Set to “1”	
LVIRF						Keep	Set to “1”

The confirmation steps for the reset sources are shown in Figure 27-5.

Figure 27-5 Confirmation steps for reset sources


after reset accepted
read RESF register
read RESF register, save RESF value to any RAM location
(clear RESF register)
SYSRF of RESF register = 1?
Yes
No
generate internal reset request due to system reset request bit been set.
WDTRF of RESF register = 1?
Yes
No
generate internal reset request due to watchdog Timer
RPERF of RESF register = 1?
Yes
No
generate internal reset request due to RAM parity check error.
IAWRF of RESF register = 1?
Yes
No
generate internal reset request due to illegal accesss registers
LVIRF of RESF register = 1?
Yes
No
generate internal reset request due to voltage detection circuit
generate power on reset / external reset

Note The above procedure is an example of a confirmation step.

Chapter 28 Power-On Reset Circuit

28.1 Function of power-on reset circuit

The power-on reset circuit (POR) has the following functions.

- Internal reset signal is generated when power is turned on.

If the supply voltage (V_{DD}) exceeds the sense voltage (V_{POR}), the reset is released. However, the reset state must be maintained by voltage detection circuitry or an external reset before 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 AC characteristics of the data sheet before the operating voltage range shown. It is reset by means of transfer in deep sleep mode, voltage detection circuitry, or external reset. When restarting operation, you must confirm that the supply voltage has returned to the operating voltage range.

Notice: When the power-on reset circuit generates an internal reset signal, clear the reset control flag register (RESF) to “00H”.

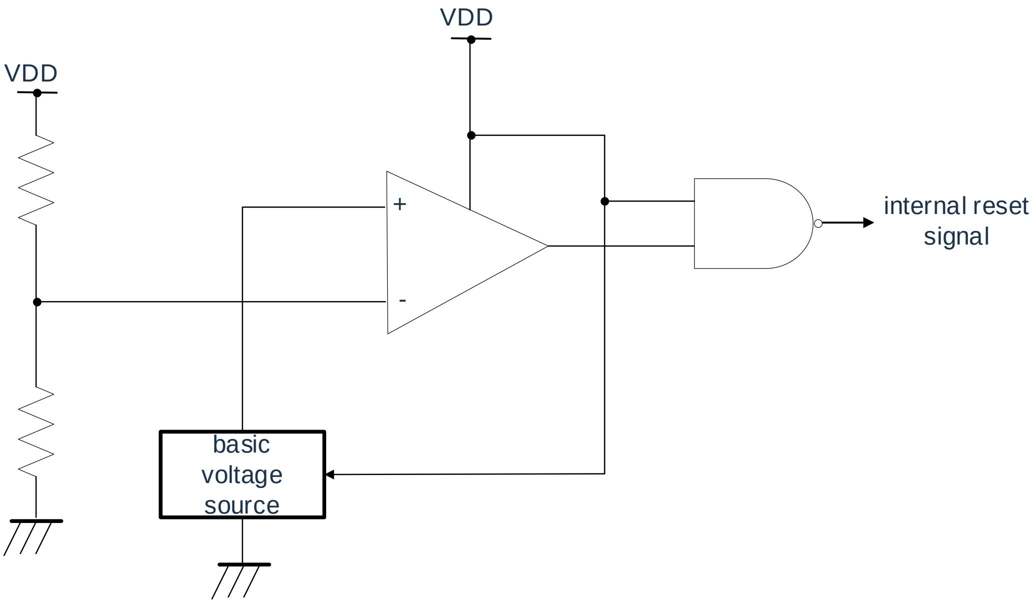
Remark: 1. The BAT32G157 has built-in hardware to generate several internal reset signals. When the internal reset signal is generated by the watchdog timer (WDT), voltage detection (LVD) circuit, system reset request set bit, RAM parity error, or illegal memory access, the flag to indicate the reset source is assigned to the RESF register; when the internal reset signal is generated by the WDT, the LVD, system reset request set, RAM parity error, or illegal memory access, the flag is set to “1” instead of clearing the RESF register to "00H". For details of the RESF register, refer to “Chapter 27 Reset Function”.

2. V_{POR}: POR supply voltage rise detection voltage
V_{PDR}: POR supply voltage drop detection voltage
Refer to the POR circuit characteristics in the datasheet for details.

28.2 Structure of power-on reset circuit

A block diagram of the power-on reset circuit is shown in Figure 28-1.

Figure 28-1 Block diagram of power-on reset circuit

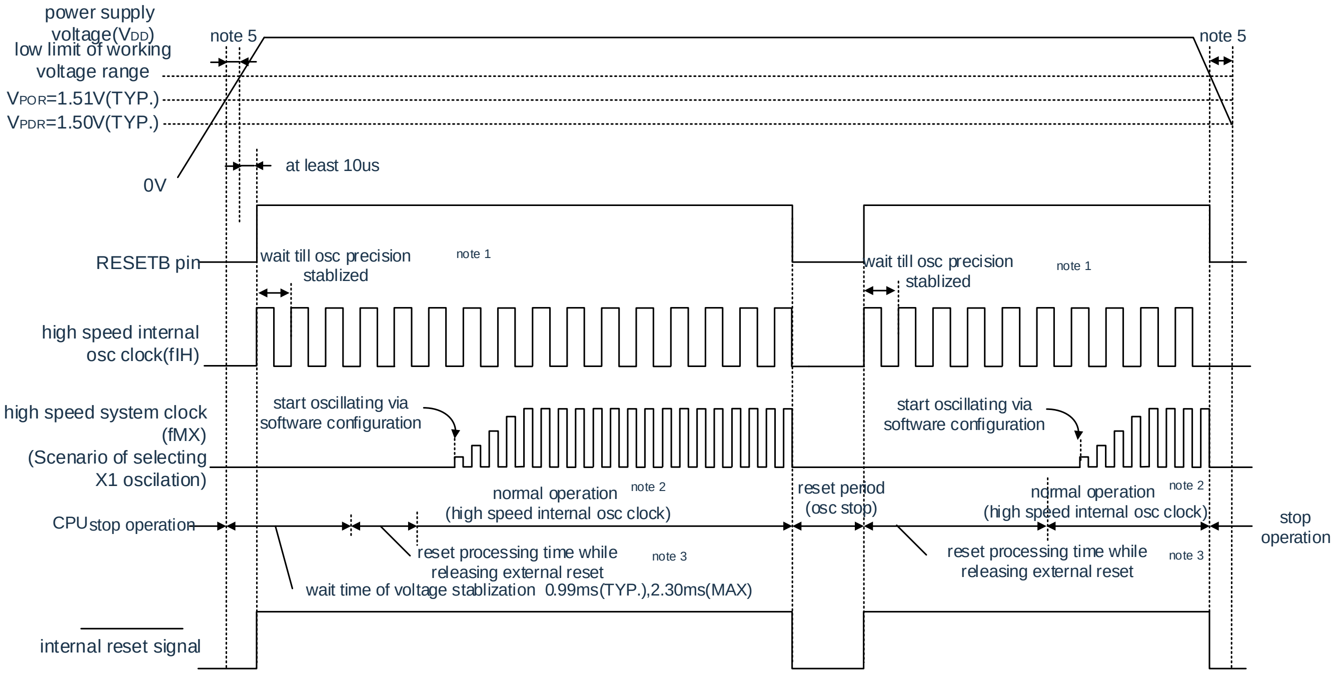


28.3 Operation of power-on reset circuit

The timing of the internal reset signal of the power-on reset circuit and the voltage detection circuit is as follows.

Figure 28-2 Generation timing of internal reset signal for power-on reset circuit and voltage detection circuit (1/3)

(1) When using an external reset input on the RESETB pin


power supply voltage(VDD)
low limit of working voltage range
VPOR=1.51V(TYP.)
VPDR=1.50V(TYP.)
0V
note 5
note 5
at least 10us
RESETB pin
wait till osc precision stablized
note 1
wait till osc precision stablized
note 1
high speed internal osc clock(fIH)
high speed system clock (fMX) (Scenario of selecting X1 oscilation)
start oscillating via software configuration
start oscillating via software configuration
normal operation
note 2
(high speed internal osc clock)
reset period (osc stop)
normal operation
note 2
(high speed internal osc clock)
CPU
stop operation
stop operation
reset processing time while releasing external reset
note 3
wait time of voltage stablization 0.99ms(TYP.),2.30ms(MAX)
reset processing time while releasing external reset
note 3
internal reset signal

Note 1 The internal reset processing time includes the oscillation accuracy stabilization wait time for the high-speed internal oscillator clock.

2. Ability to 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 switching must be made after confirming the oscillation settling time through the status register (OSTC) of the oscillation settling time counter; In the case of an XT1 clock, the switching must be made after confirming the oscillation settling time using the timer function, etc.

3. When the supply voltage rises, the reset state must be maintained by external reset before the supply voltage reaches the operating voltage range shown in the AC characteristics of the data sheet; When the supply voltage drops, it must be reset through deep sleep mode transfer, voltage detection circuitry, or external reset before the supply voltage falls below the operating voltage range. During restart operation, it must be confirmed that the supply voltage returns to the operating voltage range.

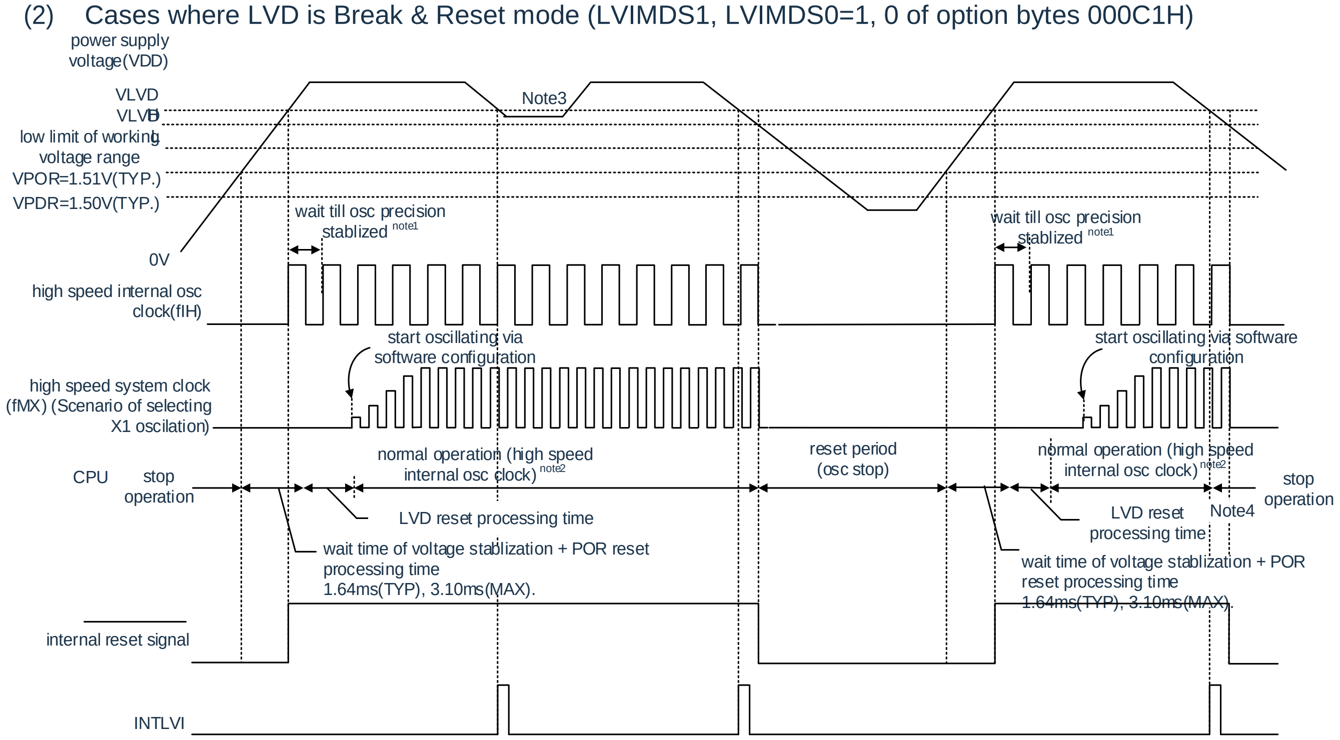
Remark V_{POR}: The POR supply voltage rises to detect the voltage

V_{PDR}: The POR supply voltage drops the detection voltage

Notice When LVD is OFF, you must use an external reset of the RESETB pin. For details, refer to "Chapter 30 Voltage Detection Circuits."

Figure 28-2 Generation timing of internal reset signal for power-on reset circuit and voltage detection circuit (2/3)

(2) Cases where LVD is Break & Reset mode (LVIMDS1, LVIMDS0=1, 0 of option bytes 000C1H)


power supply voltage(VDD)
VLVDH
VLVDL
low limit of working voltage range
VPOR=1.51V(TYP.)
VPDR=1.50V(TYP.)
0V
Note3
wait till osc precision stablized note1
wait till osc precision stablized note1
high speed internal osc clock(fIH)
start oscillating via software configuration
start oscillating via software configuration
high speed system clock (fMX) (Scenario of selecting X1 oscilation)
CPU
stop operation
normal operation (high speed internal osc clock) note2
reset period (osc stop)
normal operation (high speed internal osc clock) note2
stop operation
LVD reset processing time
LVD reset processing time
Note4
wait time of voltage stablization + POR reset processing time 1.64ms(TYP), 3.10ms(MAX).
wait time of voltage stablization + POR reset processing time 1.64ms(TYP), 3.10ms(MAX).
internal reset signal
INTLVI

Note 1. The internal reset processing time includes the oscillation accuracy stabilization wait time for the high-speed internal oscillator clock.

2. Ability to 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 switching must be made after confirming the oscillation settling time through the status register (OSTC) of the oscillation settling time counter; In the case of an XT1 clock, the switching must be made after confirming the oscillation settling time using the timer function, etc.

3. After generating the interrupt request signal (INTLVI), the LVILV bit and the LVIMD bit of the voltage detection level register (LVIS) are automatically set to "1". Therefore, it must be considered that the supply voltage may return to the high voltage detection voltage (V_{LVDH}) or higher without falling below the low voltage detection voltage (V_{LVDL}), and after generating INTLVI, follow the steps in “Figure 29-7 Setting procedure for confirmation/reset of operating voltage”and “Figure 29-8: Initial setting procedure for interrupt & reset mode”.

4. In addition to the "voltage stabilization wait + POR reset process" after reaching VPOR (1.51V(TYP.)), the following "LVD reset process" is required after reaching the LVD detection level (V_{LVDH}) until the start of normal operation ".

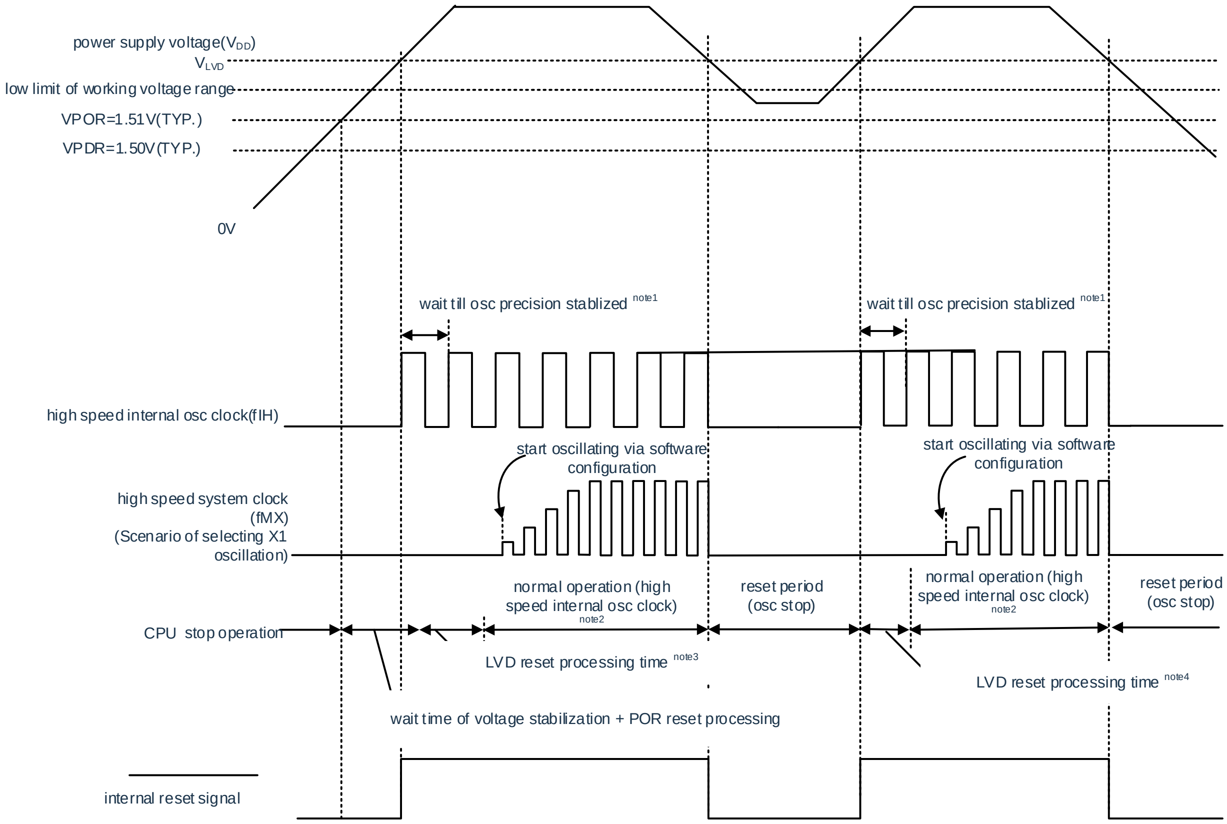
Remark V_{LVDH}, V_{LVDL}: LVD sense voltage

V_{POR} : POR supply voltage rise detection voltage

V_{PDR} : POR supply voltage drop detection voltage

Figure 28-2 Generation timing of internal reset signal for power-on reset circuit and voltage detection circuit (3/3)

(3) LVD reset mode (LVIMDS1, LVIMDS0 of option byte 000C1H = 1, 1).


power supply voltage(V_{DD})
V_{LVD}
low limit of working voltage range
VPOR=1.51V(TYP.)
VPDR=1.50V(TYP.)
0V
wait till osc precision stablized note1
wait till osc precision stablized note1
high speed internal osc clock(fIH)
start oscillating via software configuration
start oscillating via software configuration
high speed system clock (fMX) (Scenario of selecting X1 oscillation)
normal operation (high speed internal osc clock) note2
reset period (osc stop)
normal operation (high speed internal osc clock) note2
reset period (osc stop)
CPU stop operation
LVD reset processing time note3
LVD reset processing time note4
wait time of voltage stabilization + POR reset processing
internal reset signal

Note 1. The internal reset processing time includes the oscillation accuracy stabilization wait time for the high-speed internal oscillator clock.

2. Ability to 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 switching must be made after confirming the oscillation settling time through the status register (OSTC) of the oscillation settling time counter; In the case of an XT1 clock, the switching must be made after confirming the oscillation settling time using the timer function, etc.

3. The time to start running normally except to reach V_{POR} (1.51V (TYP.). In addition to “voltage stabilization waiting +POR reset processing”, it is required after the LVD detection level (V_{LVD}) is reached “LVD Reset Processing”.

4. When the supply voltage drops, if the supply voltage is restored only after the internal reset of the voltage detection circuit (LVD), the “LVD reset process” is required after the LVD sense level (V_{LVD}) is reached.

Remark 1. V_{LVDH}, V_{LVDL}: LVD sense voltage
V_{POR} : The POR supply rise sense voltage
V_{PDR} : The POR supply drop sense voltage

2. When the LVD interrupt mode is selected (LVIMD1, LVIMD0 = 0, 1 for option byte 000C1H), the time from power-on to the start of normal operation is the same as that in "Note 3" of " Figure 28-2 (3/3) LVD Reset Mode".

Chapter 29 Voltage Detection Circuit

29.1 Function of voltage detection circuit

The voltage detection circuit sets the operation mode and detection voltages (V_{LVDH}, V_{LVDL}, V_{LVD}) by the option byte (000C1H). The voltage detection (LVD) circuit has the following functions.

- Compare the supply voltage (V_{DD}) with the detection voltages (V_{LVDH}, V_{LVDL}, V_{LVD}) to generate an internal reset or internal interrupt signal.
- The detection voltage of the power supply voltage (V_{LVDH}, V_{LVDL}) can be selected from 12 detection levels using the option byte (refer to "Chapter 32 Option Byte").
- Can also run in deep sleep mode

When the supply voltage rises, the reset state must be maintained by the voltage detection circuit or external reset before the supply voltage reaches the operating voltage range shown in the AC Characteristics of the datasheet; when the supply voltage falls, it must be set to the reset state by the transfer of the deep sleep mode, the voltage detection circuit, or external reset before the supply voltage falls below the operating voltage range. The operating voltage range depends on the setting of the user option byte (000C2H/010C2H).

a) Interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0 for option bytes)

Two detection voltages (V_{LVDH}, V_{LVDL}) are selected by the option byte 000C1H, a high voltage detection level (V_{LVDH}) for resetting release or interruption generation, and a low voltage detection level (V_{LVDL}) for reset generation.

b) Reset mode (LVIMDS1, LVIMDS0=1, 1 for option bytes)

Use the 1 detection voltage (V_{LVD}) selected by option byte 000C1H for generating or releasing the reset.

c) Interrupt mode (LVIMDS1, LVIMDS0=0, 1 for option bytes)

Use the 1 detection voltage (V_{LVD}) selected by option byte 000C1H to generate an interrupt or to release the reset. The following interrupt signals and internal reset signals are generated in each mode.

Interrupt & Reset mode (LVIMDS1, LVIMDS0=1, 0)	Reset mode (LVIMDS1, LVIMDS0=1, 1)	Interrupt mode (LVIMDS1, LVIMDS0=0, 1)
Generates an interrupt request signal when the operating voltage drops and when $V_{DD} < V_{LVDH}$ is detected; Generates an internal reset when $V_{DD} < V_{LVDL}$ is detected; When $V_{DD} \geqslant V_{LVDH}$ is detected, the internal reset is released.	When $V_{DD} \geqslant V_{LVD}$ is detected, the internal reset is released; When $V_{DD} < V_{LVD}$ is detected, an internal reset is generated.	After a reset occurs, the internal reset state of the LVD is maintained until $V_{DD} \geqslant V_{LVD}$. When $V_{DD} \geqslant V_{LVD}$ is detected, the internal reset of the LVD is released. After the internal reset of the LVD is released, an interrupt request signal (INTLVI) is generated if $V_{DD} < V_{LVD}$ or $V_{DD} \geqslant V_{LVD}$ is detected.

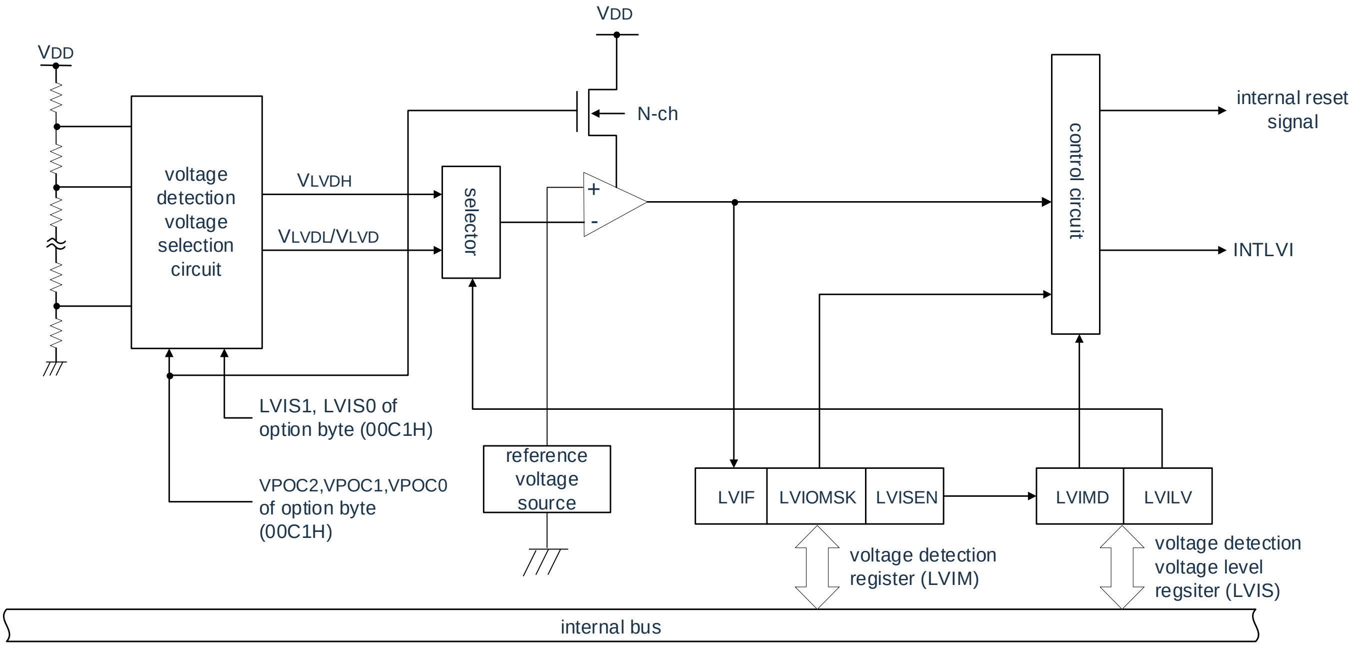
When the voltage detection circuit is operating, it can confirm whether the power supply voltage is greater than or equal to the detection voltage or less than the detection voltage by reading the voltage detection flag (LVIF: bit0 of the voltage detection register (LVIM)).

If a reset occurs, set bit0 (LVIRF) of the reset control flag register (RESF) to "1". For details of the RESF register, refer to "Chapter 27 Reset Function".

29.2 Structure of voltage detection circuit

The block diagram of the voltage detection circuit is shown in Figure 29-1.

Figure 29-1 Block diagram of voltage detection circuit


VDD
VDD
voltage detection voltage selection circuit
VLVDH
VLVDL/VLVD
selector
N-ch
+
-
control circuit
internal reset signal
INTLVI
LVIS1, LVIS0 of option byte (00C1H)
VPOC2,VPOC1,VPOC0 of option byte (00C1H)
reference voltage source
LVIF
LVIOMSK
LVISEN
LVIMD
LVILV
voltage detection register (LVIM)
voltage detection voltage level regsiter (LVIS)
internal bus

29.3 Registers for controlling voltage detection circuit

The voltage detection circuit is controlled by the following registers.

- Voltage detection register (LVIM)
- Voltage detection level register (LVIS)

29.3.1 Voltage detection register (LVIM)

This register is set to enable or disable rewriting of the Voltage Detection Level Register (LVIS) and to confirm the masking status of the LVD outputs. The LVIM register is set by an 8-bit memory operation instruction.

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

Figure 29-2 Format of voltage detection register (LVIM)

Address: 40020441H After reset: 00H Note 1 R/W Note 2

Symbol	7	6	5	4	3	2	1	0
LVIM	LVISEN Note 3	0	0	0	0	0	LVIOMSK	LVIF

LVISENNote3	Setting of enable/disable rewriting of the voltage detection level register (LVIS)
0	Disable overwriting the LVIS register (LVIOMSK=0 (LVD output mask is invalid)).
1	Enable overwriting the LVIS register (LVIOMSK=1 (LVD output mask is valid)).

LVIOMSK	Masking status flag for LVD output
0	The LVD output mask is invalid.
1	The LVD output mask is valid Note4.

LVIF	Voltage detection flag
0	Power supply voltage (V_{DD}) ≥ detection voltage (V_{LVD}) or LVD is OFF.
1	Power supply voltage (V_{DD}) < detection voltage (V_{LVD})

Note:

1) The reset value changes due to the reset source.
When the LVD is reset, the value of the LVIM register is not reset and the original value is kept. Clear the LVISEN to "0" when other reset occurs.

2) Bit0 and bit1 are read-only bits.

3) It can be set only when the interrupt & reset mode is selected (LVIMDS1 bit and LVIMDS0 bit of the option byte are "1" and "0", respectively), and the initial value cannot be changed in other modes.

4) Only when the interrupt & reset mode is selected (LVIMDS1 and LVIMDS0 bits of the option byte are "1" and "0", respectively), the LVIOMSK bit automatically changes to "1" during the following periods, blocking the reset or interrupt generated by the LVD.
   - When LVISEN=1
   - Waiting time from the occurrence of an LVD interrupt until the LVD detection voltage stabilizes
   - Waiting time from changing the value of the LVILV bit until the LVD detection voltage stabilizes

29.3.2 Voltage detection level register (LVIS)

This is a register that sets the voltage detection level.

The LVIS register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to 00H/01H/81H [Note1].

Figure 29-3 Format of voltage detection level register (LVIS)

Address: FFFAAH After reset: 00H/01H/81H [Note 1] R/W

Symbol	7	6	5	4	3	2	1	0
LVIS	LVIMD [Note 2]	0	0	0	0	0	0	LVILV [Note 2]

LVIMD [Note 2]	Operation mode of voltage detection
0	Interrupt mode
1	Reset mode

LVILV [Note 2]	LVD detection level
0	High voltage detection level (V_{LVDH})
1	Low voltage detection level (V_{LVDL} or V_{LVD})

Note: 1. The reset value changes due to the settings of the reset source and option bytes. When LVD reset occurs, do not clear this register '00H'.
When a reset other than LVD occurs, the value of this register is as follows:
- When LVIMDS1, LVIMDS0 of option byte = 1, 0: 00H
- When LVIMDS1, LVIMDS0 of option byte = 1, 1: 81H
- When LVIMDS1, LVIMDS0 of option byte = 0, 1: 01H

2. “0" can be written only when the interrupt & reset mode is selected (LVIMDS1 bit and LVIMDS0 bit of the option byte are "1" and "0" respectively). It cannot be set in other cases. In the interrupt & reset mode, the value is replaced automatically by generating a reset or an interrupt.

Notice: 1. To override the LVIS register, you must follow the steps in Figures 29-7 and 29-8.

2. The operation mode of the LVD and the detection voltage for each mode (V_{LVDH}, V_{LVDL}, V_{LVD}) are selected with the option byte 000C1H. The format of the user option byte (000C1H/010C1H) is shown in Table 29-1. For details of the option byte, refer to "Chapter 32 Option Byte".

Table 29-1 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 setting (Interrupt & Reset Mode)

Detection voltage			Setting value of option byte						
V_{LVDH}		V_{LVDL}	VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rise	drop	drop						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 mentioned above is prohibited.						

- LVD setting (reset mode)

Detection voltage		Setting value of option byte						
V_{LVD}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rise	drop						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 mentioned above is prohibited.						

Remark: 1. The detection voltage is the TYP. value. For details, refer to LVD Circuit Characteristics in the datasheet.

Table 29-1 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 setting (interrupt mode)

Detection voltage		Setting value of option byte						
V_{LVD}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rise	drop						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 mentioned above is prohibited.						

- LVD is OFF (external reset using RESETB pin)

Detection voltage		Setting value of option byte						
V_{LVD}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rise	drop						LVIMDS1	LVIMDS0
—	—	1	×	×	×	×	×	1
—		Setting values other than those mentioned above is prohibited.						

Notice: 1. You must write "1" to bit4.

2. When the power supply voltage rises, the reset state must be maintained through the voltage detection circuit or external reset before the power supply voltage reaches the working voltage range indicated by the AC characteristic of the data manual; When the power supply voltage drops, it must be reset before the power supply voltage falls below the operation voltage range by the deep sleep mode transfer, voltage detection circuit or external reset.

The operation voltage range depends on the setting of the user option byte (000C2H).

Remark:

1. ×: Ignore
2. The detection voltage is the TYP. value. For details, refer to LVD Circuit Characteristics in the datasheet.

29.4 Operation of voltage detection circuit

29.4.1 Settings when used as reset mode

Set the operation mode (reset mode (LVIMDS1, LVIMDS0=1, 1) and the detection voltage (V_{LVD}) by option byte 000C1H. If the reset mode is set, operation starts with the following initial settings.

- Set bit7 (LVISEN) of the voltage detection register (LVIM) to "0" (rewriting of the voltage detection level register (LVIS) is prohibited).
- Set the initial value of the voltage detection level register (LVIS) to "81H". bit7 (LVIMD) is "1" (reset mode). Bit0 (LVILV) is "1" (voltage detection level: V_{LVD}).

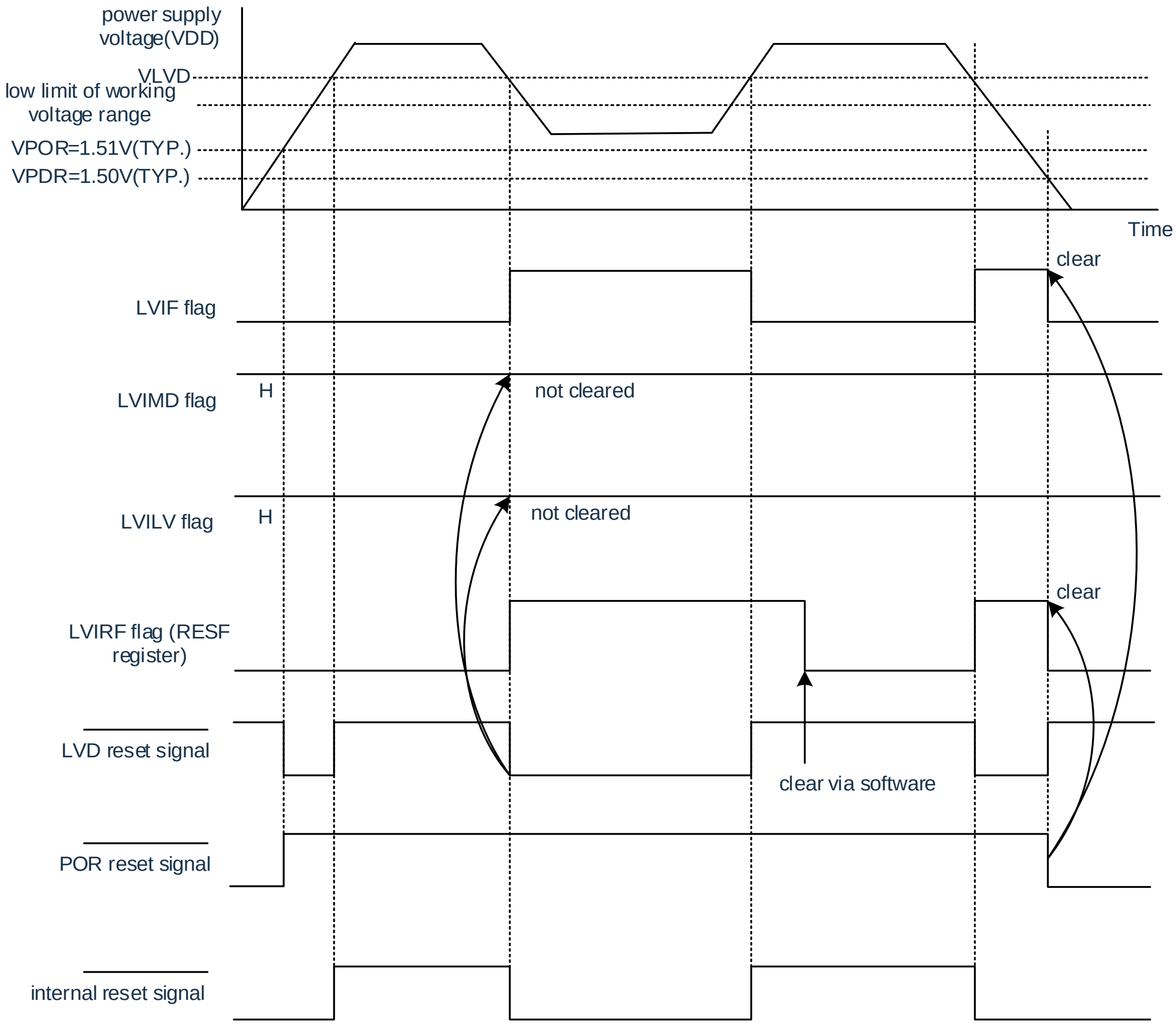
- LVD reset mode operation

  When the power is turned on, the reset mode (LVIMDS1, LVIMDS0=1, 1 of the option byte) maintains the LVD in an internal reset state until the supply voltage (V_{DD}) exceeds the voltage detection level (V_{LVD}). If the supply voltage (V_{DD}) exceeds the voltage detection level (V_{LVD}), the internal reset is released.

  An internal reset of the LVD is generated if the supply voltage (V_{DD}) falls below the voltage detection level (V_{LVD}) when the operating voltage drops.

  The timing of the generation of the internal reset signal for the LVD reset mode is shown in Figure 29-4.

Figure 29-4 Generation timing of internal reset signal (LVIMDS1, LVIMDS0=1, 1 of option bytes)



Remark V_{POR}: POR supply voltage rise detection voltage

V_{PDR}: POR supply voltage drop detection voltage

29.4.2 Settings when used as interrupt mode

The operation mode (interrupt mode (LVIMDS1, LVIMDS0=0, 1)) and detection voltage (V_{LVD}) are set with option byte 000C1H. If the interrupt mode is set, operation starts in the state of the following initial setting.

- Set bit7 (LVISEN) of the voltage detection register (LVIM) to "0" (rewriting of the voltage detection level register (LVIS) is prohibited).
- Set the initial value of the voltage detection level register (LVIS) to "01H". Bit7 (LVIMD) is "0" (interrupt mode).

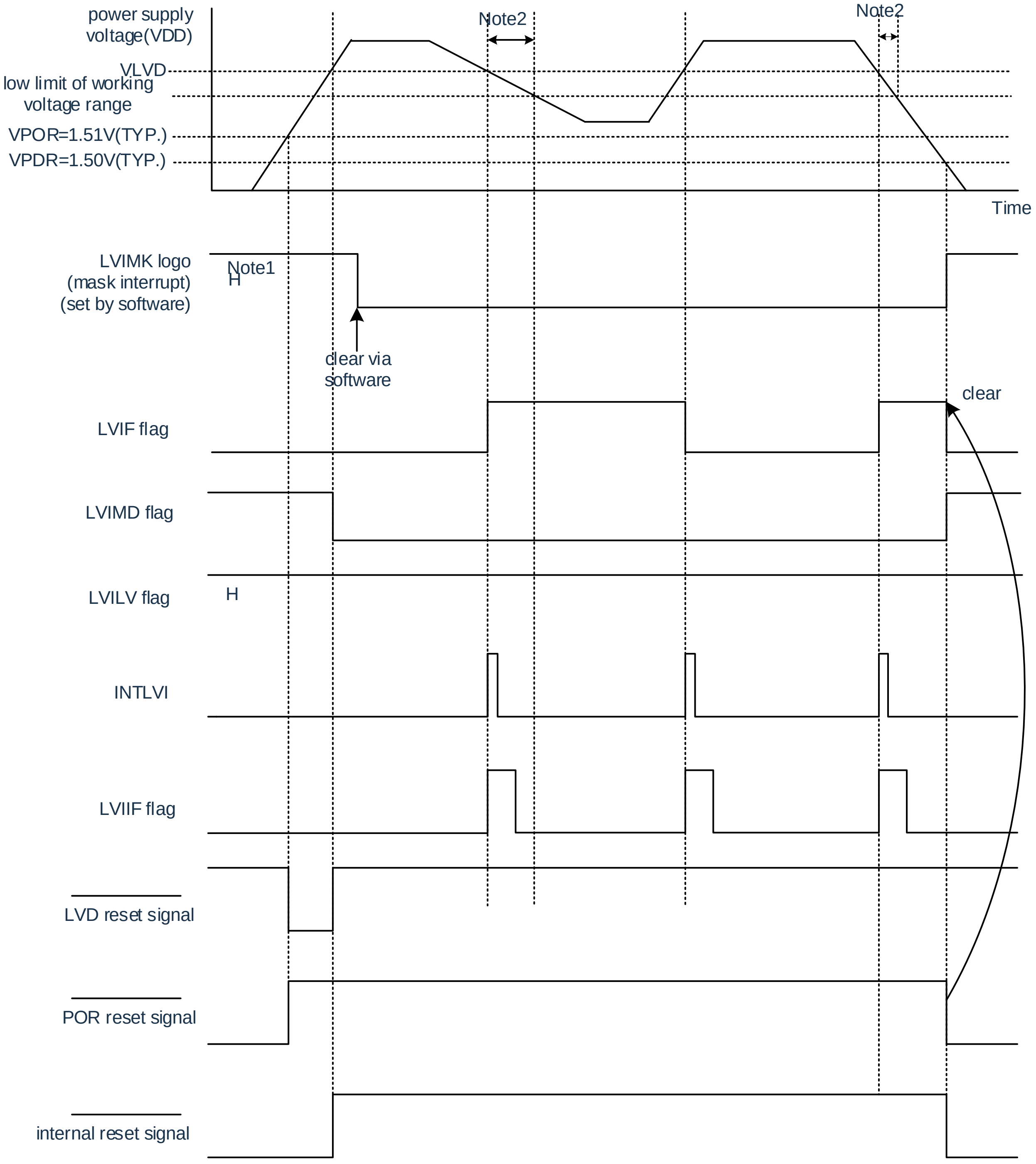
Bit0 (LVILV) is "1" (voltage detection level: V_{LVD}).

- LVD interrupt mode operation

  After generating a reset, the interrupt mode (LVIMDS1, LVIMDS0 = 0, 1 of the option byte) maintains the LVD's internal reset state until the supply voltage (V_{DD}) exceeds the voltage detection level (V_{LVD}). If the supply voltage (V_{DD}) exceeds the voltage detection level (V_{LVD}), the internal reset of the LVD is released. The interrupt request signal (INTLVI) for the LVD is generated if the supply voltage (V_{DD}) exceeds the voltage detection level (V_{LVD}) after the LVD is released from internal reset. When the operating voltage drops, it must be set to the reset state, either by deep sleep mode transfer or external reset, before the operating voltage falls below the operating voltage range shown in the AC characteristics of the datasheet. Upon restarting operation, it must be verified that the supply voltage returns to the operating voltage range.

The timing of the generation of the interrupt request signal for the LVD interrupt mode is shown in Figure 29-5.

Figure 29-5 Timing of interrupt signal generation (LVIMDS1, LVIMDS0 = 0, 1 for option byte)



Note 1. After the reset signal is generated, the LVIMK flag becomes "1".

2. When the operating voltage drops, it must be set to the reset state by a deep sleep mode transfer or external reset before the operating voltage falls below the operating voltage range shown in the AC characteristics of the datasheet. When restarting operation, it must be verified that the supply voltage returns to the operating voltage range.

Remark: V_{POR}: POR power supply voltage rise detection voltage

V_{PDR}: POR power supply voltage drop detection voltage

29.4.3 Settings when used as interrupt & reset mode

The operation mode (interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0)) and detection voltage (V_{LVDH}, V_{LVDL}) are set by option byte 000C1H.

If the interrupt & reset mode is set, operation starts in the state of the following initial settings.

- Set bit7 (LVISEN) of the voltage detection register (LVIM) to "0" (rewriting of the voltage detection level register (LVIS) is prohibited).
- Set the initial value of the voltage detection level register (LVIS) to "00H".
  Bit7 (LVIMD) is "0" (interrupt mode).
  Bit0 (LVILV) is "0" (high voltage detection level: V_{LVDH}).

- LVD Interrupt & Reset mode operation

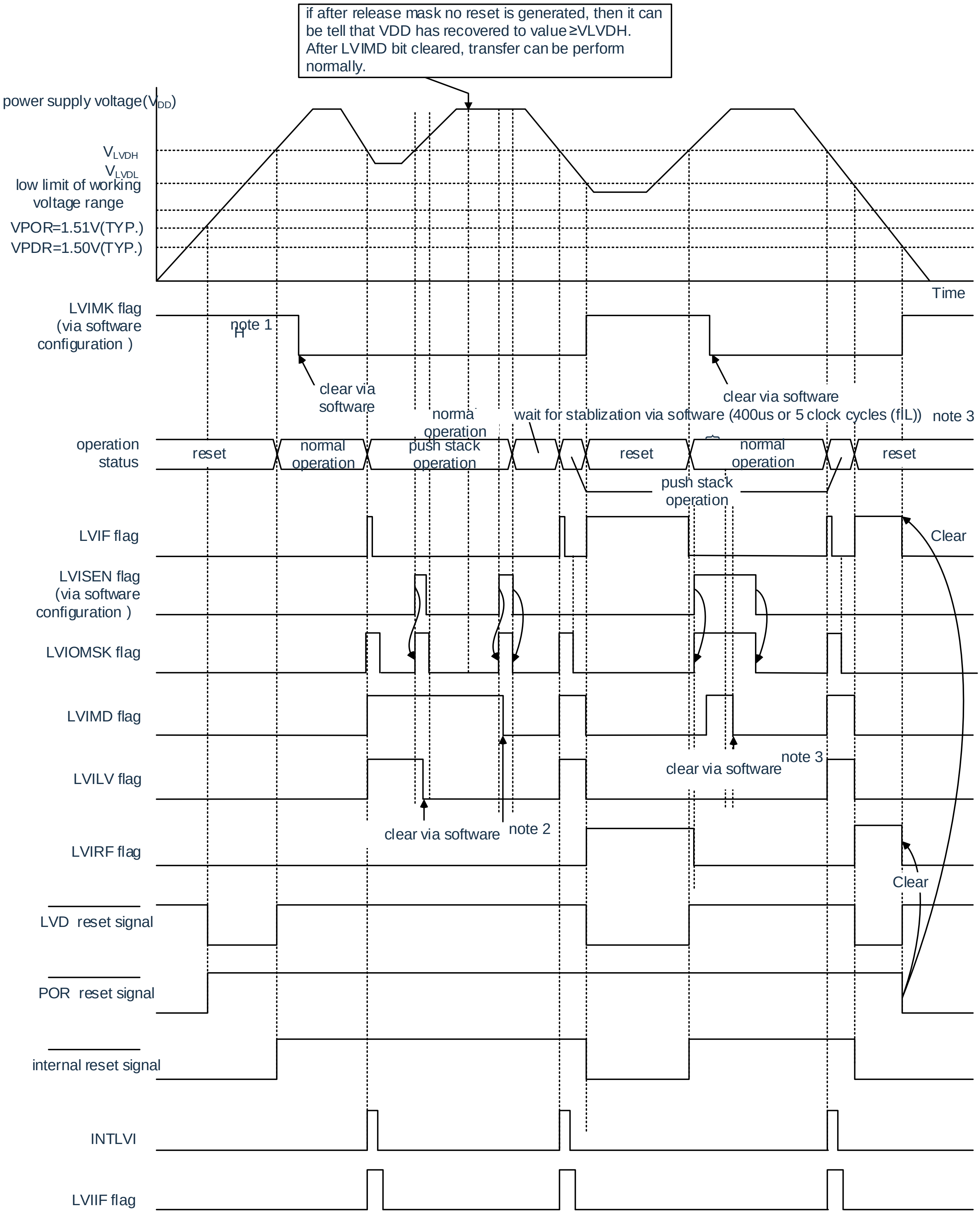
  When power is applied, the interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0 of the option byte) maintains the LVD's internal reset state until the supply voltage (V_{DD}) exceeds the high voltage detection level (V_{LVDH}). If the supply voltage (V_{DD}) exceeds the high voltage detection level (V_{LVDH}), the internal reset is released.

  When the operating voltage drops, if the supply voltage (V_{DD}) falls below the high voltage detection level (V_{LVDH}), the LVD's interrupt request signal (INTLVI) is generated and arbitrary stack processing can be performed. Thereafter, if the supply voltage (V_{DD}) falls below the low voltage detection level (V_{LVDL}), an internal reset of the LVD is generated. However, after the occurrence of INTLVI, the interrupt request signal is not generated even if the supply voltage (V_{DD}) returns to the high voltage detection voltage (V_{LVDH}) or higher in a state that is not lower than the low voltage detection voltage (V_{LVDL}).

  When using the LVD interrupt & reset mode, you must follow the steps in the flowcharts shown in “Figure 29-7 Setting Procedure for Operating Voltage Confirmation/Reset” and “Figure 29-8 Initial Setting Procedure for Interrupt & Reset Mode”.

The timing of the generation of the internal reset and interrupt signals for the LVD interrupt & reset mode is shown in Figure 29-6.

Figure 29-6 Timing of reset & interrupt signal generation (LVIMDS1, LVIMDS0=1, 0 for option byte) (1/2)



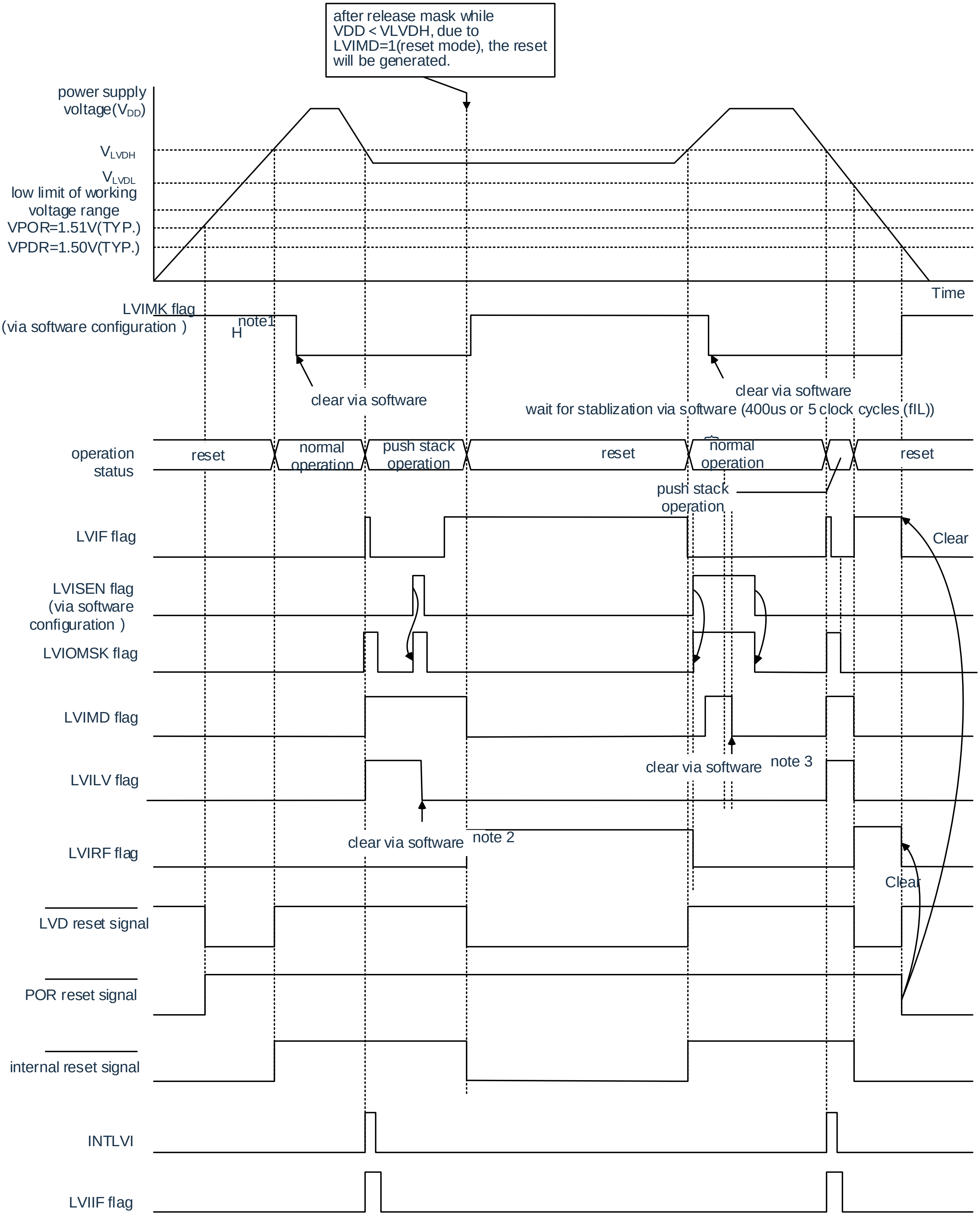
Note 1. After the reset signal is generated, the LVIMK flag becomes "1".

2. When using the interrupt & reset mode, you must follow the “Fig. 29-7 Setting Procedure for Operating Voltage Confirmation/Reset” after an interrupt occurs.

3. When using the interrupt & reset mode, you must follow the "Fig. 29-8 Initial Setting Procedure for Interrupt & Reset Mode" after releasing the reset.

Remark: V_{POR}: POR supply voltage rise detection voltage
V_{PDR}: POR supply voltage drop detection voltage

Figure 29-6 Timing of interrupt & reset signal generation (LVIMDS1, LVIMDS0=1, 0 for option byte) (2/2)



Note:

1. After the reset signal is generated, the LVIMK flag becomes "1".

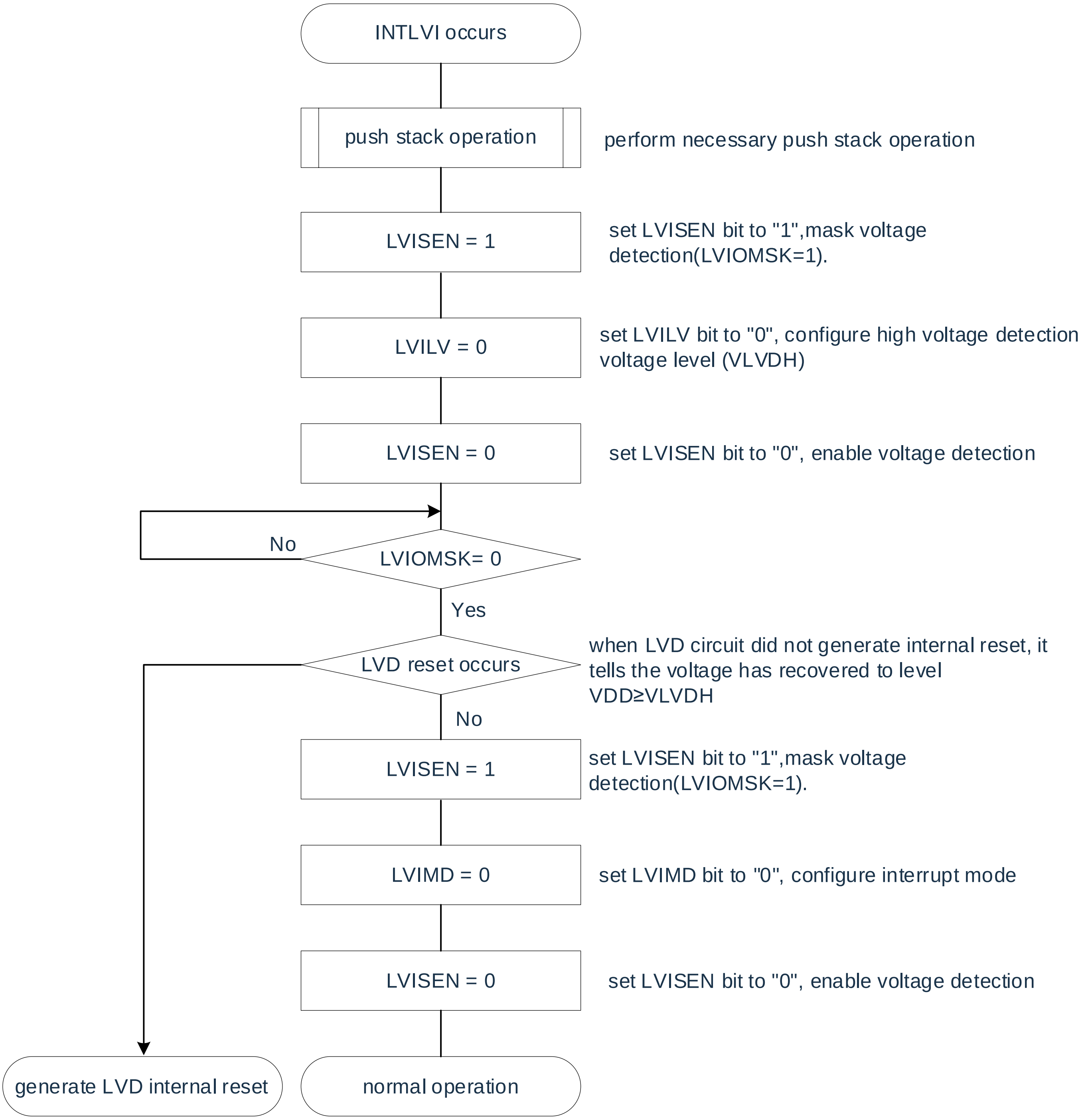
2. When using the interrupt & reset mode, you must follow the “Fig. 29-7 Setting Procedure for Operating Voltage Confirmation/Reset” after an interrupt occurs.

3. When using the interrupt & reset mode, you must follow the "Fig. 29-8 Initial Setting Procedure for Interrupt & Reset Mode" after releasing the reset.

Remark: V_{POR}: POR power supply voltage rise detection voltage

V_{PDR}: POR power supply voltage drop detection voltage

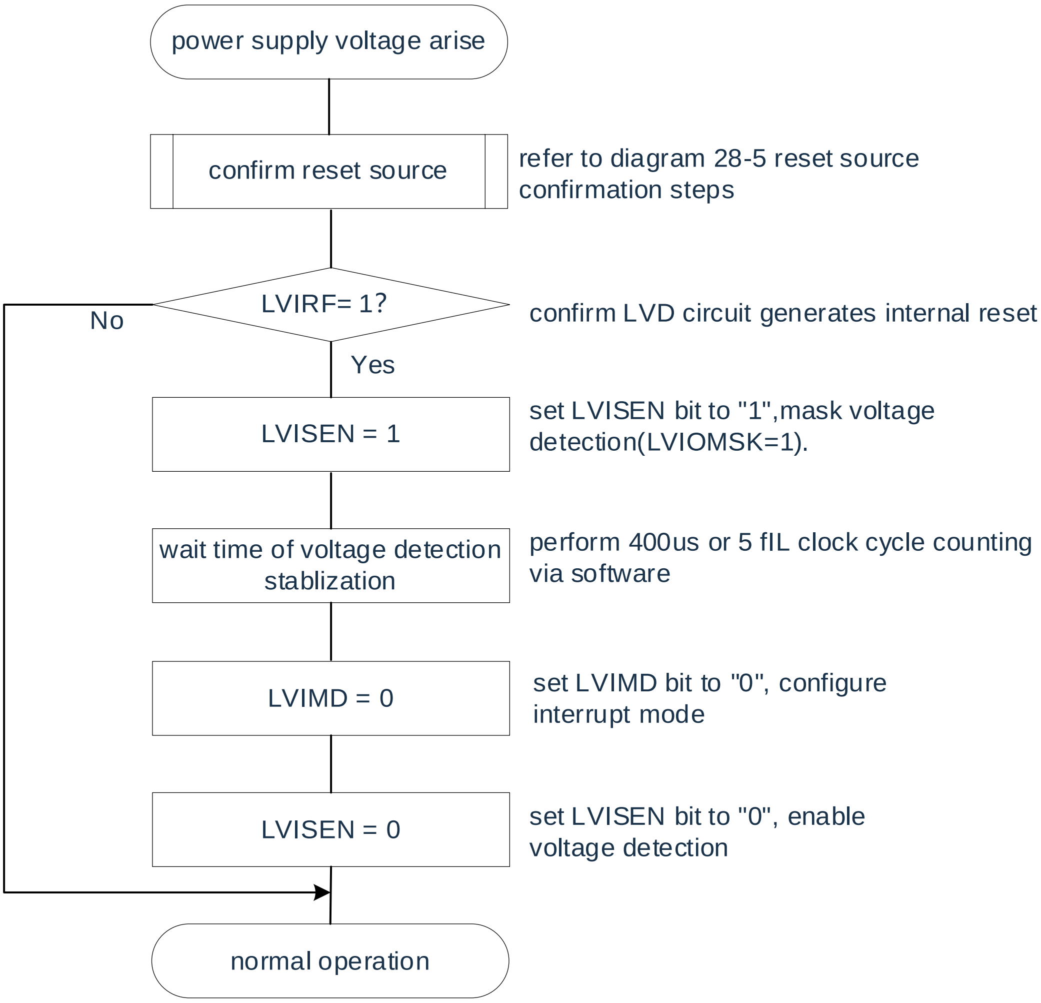
Figure 29-7 Setting procedure for operating voltage confirmation/reset


INTLVI occurs
push stack operation
perform necessary push stack operation
LVISEN = 1
set LVISEN bit to "1",mask voltage detection(LVIOMSK=1).
LVILV = 0
set LVILV bit to "0", configure high voltage detection voltage level (VLVDH)
LVISEN = 0
set LVISEN bit to "0", enable voltage detection
No
LVIOMSK= 0
Yes
LVD reset occurs
when LVD circuit did not generate internal reset, it tells the voltage has recovered to level VDD≥VLVDH
No
LVISEN = 1
set LVISEN bit to "1",mask voltage detection(LVIOMSK=1).
LVIMD = 0
set LVIMD bit to "0", configure interrupt mode
LVISEN = 0
set LVISEN bit to "0", enable voltage detection
generate LVD internal reset
normal operation

If the interrupt & reset mode is set (LVIMDS1, LVIMDS0=1, 0), a voltage detection stabilization wait time of 400 μs or 5 f_{IL} clocks is required after releasing the LVD reset (LVIRF=1). The LVIMD bit must be cleared to "0" for initialization after waiting for the voltage detection to stabilize. During the counting of the voltage detection stabilization wait time and when rewriting the LVIMD bit, the LVISEN bit must be set to "1" to block the reset or interrupt generated by the LVD.

The initial setup procedure for the interrupt & reset mode is shown in Figure 29-8.

Figure 29-8 Initial setting procedure for interrupt & reset mode


power supply voltage arise
confirm reset source
refer to diagram 28-5 reset source confirmation steps
LVIRF= 1?
No
Yes
confirm LVD circuit generates internal reset
LVISEN = 1
set LVISEN bit to "1",mask voltage detection(LVIOMSK=1).
wait time of voltage detection stablization
perform 400us or 5 fIL clock cycle counting via software
LVIMD = 0
set LVIMD bit to "0", configure interrupt mode
LVISEN = 0
set LVISEN bit to "0", enable voltage detection
normal operation

Remark f_{IL}: Low-speed internal oscillator clock frequency

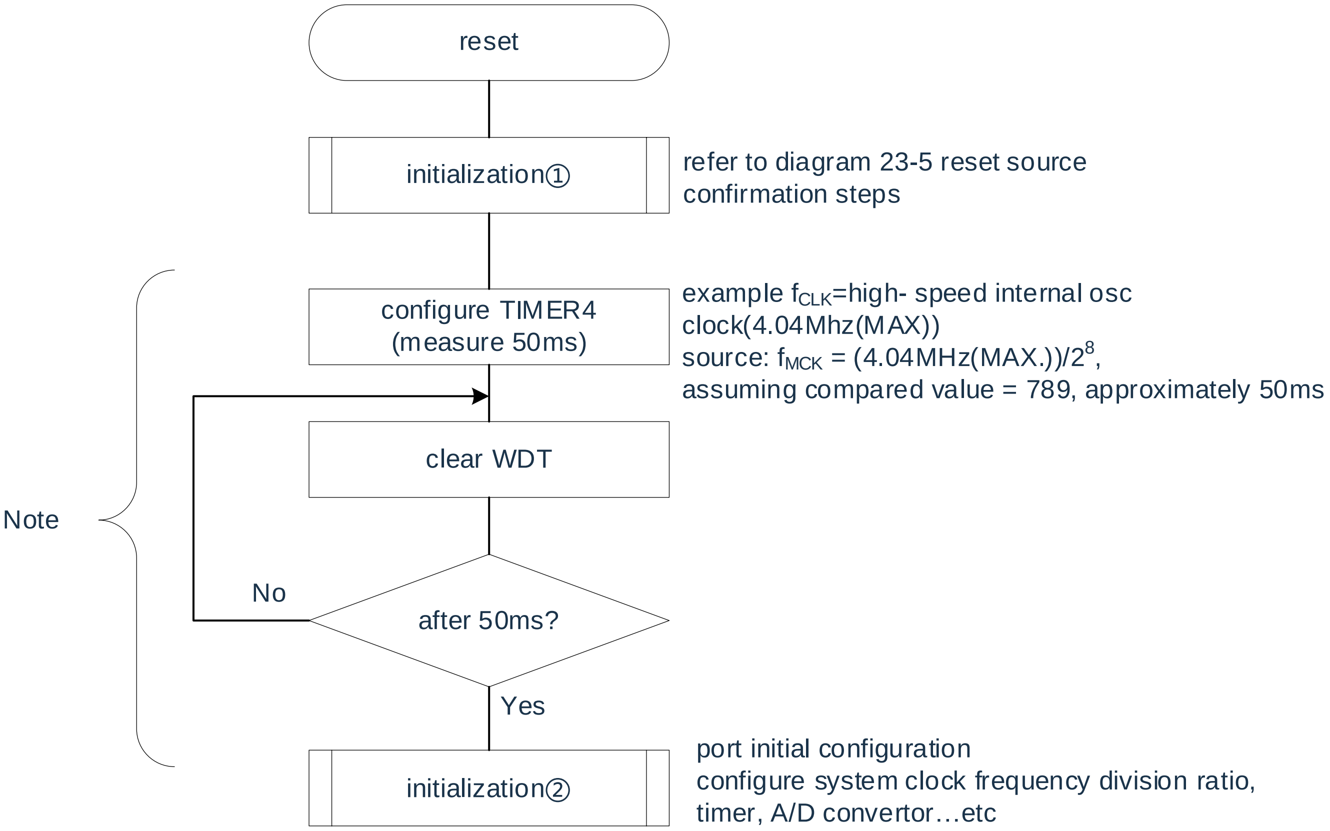
29.5 Cautions for voltage detection circuit

(1) Voltage fluctuations at power-on

For systems where the power supply voltage (V_{DD}) fluctuates for a certain period of time in the vicinity of the LVD detection voltage, it is possible to repeatedly enter the reset state and the reset release state. The time from reset release to the start of microcontroller operation can be arbitrarily set by the following processing.

＜Processing＞ After releasing the reset, the initial setting of ports, etc. must be performed by using the software counter of the timer after waiting for the power supply voltage fluctuation time that varies for each system.

Figure 29-9 Example of software processing when the power supply voltage near the LVD detection voltage does not fluctuate for more than 50ms

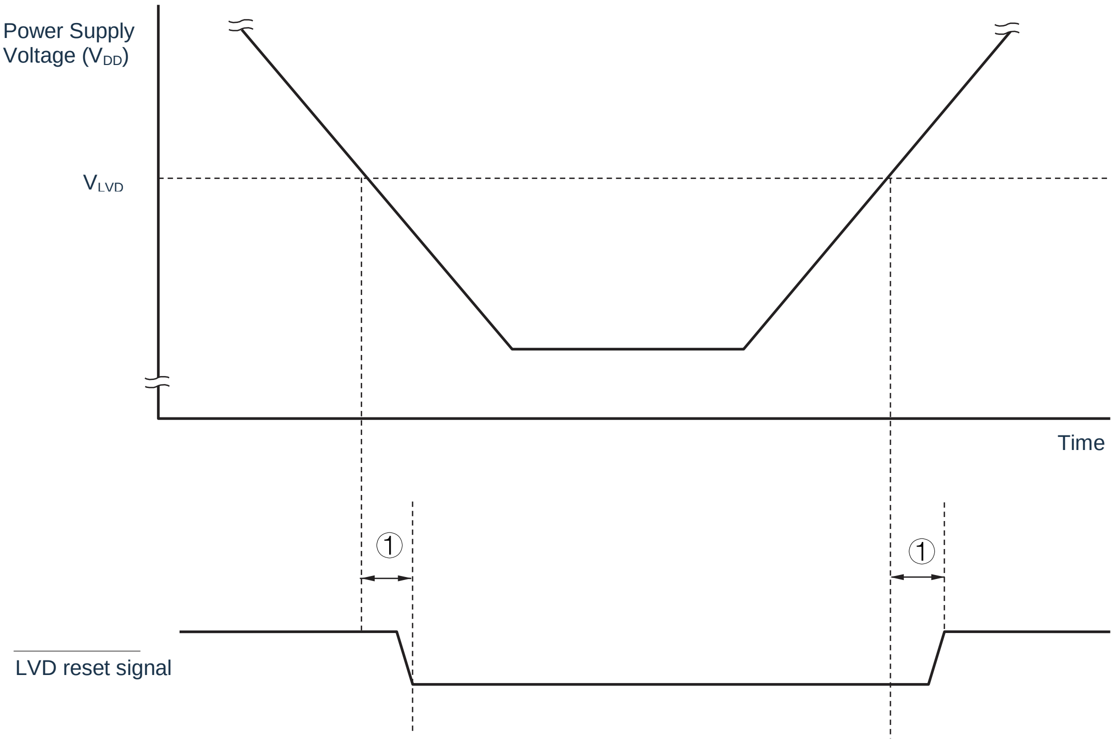

reset
initialization①
refer to diagram 23-5 reset source confirmation steps
configure TIMER4 (measure 50ms)
example f_{CLK}=high- speed internal osc clock(4.04Mhz(MAX))
source: f_{MCK} = (4.04MHz(MAX.))/2^8,
assuming compared value = 789, approximately 50ms
clear WDT
Note
No
after 50ms?
Yes
initialization②
port initial configuration
configure system clock frequency division ratio, timer, A/D convertor…etc

Note If a reset occurs again during this period, it is not transferred to initialization processing ②.

(2) Delay from generation of LVD reset source to generation or release of LVD reset

A delay occurs from the time the supply voltage (VDD) < LVD detection voltage (VLVD) is satisfied until the LVD reset is generated. Similarly, a delay occurs from LVD detection voltage (VLVD) ≤ supply voltage (VDD) until the LVD reset is released (refer to Figure 29-10).

Figure 29-10 Delay from generation of LVD reset source to generation or release of LVD reset



①: Detection delay (300s μs (MAX.))

(3) When the power is turned on with LVD set to OFF

When LVD is set to OFF, an external reset of the RESETB pin must be used.

When performing an external reset, a low level of at least 10 μs must be input to the RESETB pin. If an external reset is performed while the supply voltage is rising, the power must be turned on after inputting a low level to the RESETB pin, and it must be held low for at least 10 μs over the operating voltage range shown in the AC Characteristics of the datasheet, and then input a high level.

(4) Operating voltage drops when LVD is set to OFF and LVD is in interrupt mode.

If the operating voltage drops while the LVD is set to OFF and the LVD interrupt mode is set, it must be set to the reset state by a deep sleep mode transfer or external reset before the operating voltage falls below the operating voltage range shown in the AC characteristics of the datasheet. When restarting operation, it must be verified that the supply voltage returns to the operating voltage range.

Chapter 30 Safety Function

30.1 Overview of safety functions

In response to the IEC60730 and EC61508 safety standards, the following safety features are built into the BAT32G157.

These functions enable the microcontroller to self-diagnose abnormalities and stop operating if an abnormality is detected.

(1) Flash memory CRC operation function (high-speed CRC, general-purpose CRC)

A data error of the flash memory is detected by the CRC operation. The following two CRCs can be used respectively according to different uses and use conditions.

- "High-speed CRC"... In the initialization program, the CPU can be stopped and the entire code flash memory area can be checked at high speed.
- "General-purpose CRC"... In CPU operation, it is not limit to code flash area but can be used for multi-purpose inspection.

(2) RAM parity error detection function

Detects parity errors when reading RAM data.

(3) SFR protection function

Prevents SFR from being overridden due to CPU runaway.

(4) Frequency detection function

A general-purpose timer unit can be used for self-detection of CPU/peripheral hardware clock frequency.

(5) A/D test function

A/D converter self-check can be carried out through A/D conversion of positive (+) reference voltage, negative (negative-reference voltage, analog input channel (ANI), temperature sensor output and internal reference voltage output.

(6) Digital output signal level detection function of input/output port

When the input/output port is in the output mode, the output level of the pin can be read.

30.2 Registers used by safety functions

The following registers are used for each function of the safety function.

Register name	Each function of safety function
• Flash CRC control register (CRC0CTL) • Flash CRC operation result register (PGCRCL)	Flash memory CRC operation (high-speed CRC)
• CRC input register (CRCIN) • CRC data register (CRCD)	CRC operation function (general-purpose CRC)
• RAM parity error control register (RPECTL)	RAM parity error detection
• 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 selection register (PMS)	Digital output signal level detection function of input/output pin

The content of each register is describe in “30.3 Operation of Safety Funcitions”.

30.3 Operation of safety functions

30.3.1 Flash memory CRC operation function (high-speed CRC)

The IEC60730 standard requires confirmation of the data in flash memory, and recommends CRC as a means of confirmation. This high-speed CRC can check the entire code flash memory area in the initial set-up (initialization) program. The high-speed CRC can only be performed in sleep mode of the main system clock through a program in the RAM.

The high speed CRC stops the CPU running and reads 32-bit data from the flash memory through 1 clock to operate. It is therefore characterized by a shorter time to complete the check (e.g. 64KB flash: 512μs@32MHz).

The CRC-generated polynomial corresponds to “$X^{16}+X^{12}+X^{5}+1$” of CRC-16-CCITT.

The high-speed CRC operates in MSB first order from bit 31 to bit 0.

Remark Since that general-purpose CRC is LSB first order, the result of the operation is different.

30.3.1.1 Flash memory CRC control register (CRC0CTL)

This is a register that sets the operational control and operational range of the high speed CRC operator. The CRC0CTL register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "00H".

Figure 30-1 Format of flash CRC control register (CRC0CTL)

Address: 40021810H After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
CRC0CTL	CRC0EN	FEA6	FEA5	FEA4	FEA3	FEA2	FEA1	FEA0

CRC0EN	Control of high-speed CRC operation
0	Stop the operation.
1	Start the operation by executing the WFE instruction.

CRCCHK60	FEA2	FEA1	FEA0	High-speed CRC operation range
0	0	0	0	00000H ~ 1FFBH(8K-4byte)
0	0	0	1	00000H ~ 3FFBH(16K-4byte)
0	0	1	0	00000H ~ 5FFBH(24K-4byte)
0	0	1	1	00000H ~ 7FFBH(32K-4byte)
0	1	0	0	00000H ~ 9FFBH(40K-4byte)
0	1	0	1	00000H ~ BFFBH(48K-4byte)
0	1	1	0	00000H ~ DFFBH(56K-4byte)
0	1	1	1	00000H ~ FFFBH(64K-4byte)
1	0	0	0	00000H ~ EFFFBH(60K-4byte)

Remark The expected value of that CRC operation result for comparison must be stored in the last 4 bytes of the flash memory in advance, so the operation range will be reduced by 4 bytes.

30.3.1.2 Flash CRC operation result register (PGCRCL)

This is a register that stores the results of high speed CRC operations.

The PGCRCL register is set by a 16-bit memory operation instruction.

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

Figure 30-2 Format of flash memory CRC operation result register (PGCRCL)

Address: 0x40021812 After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8
PGCRCL	PGCRC15	PGCRC14	PGCRC13	PGCRC12	PGCRC11	PGCRC10	PGCRC9	PGCRC8
	7	6	5	4	3	2	1	0
	PGCRC7	PGCRC6	PGCRC5	PGCRC4	PGCRC3	PGCRC2	PGCRC1	PGCRC0

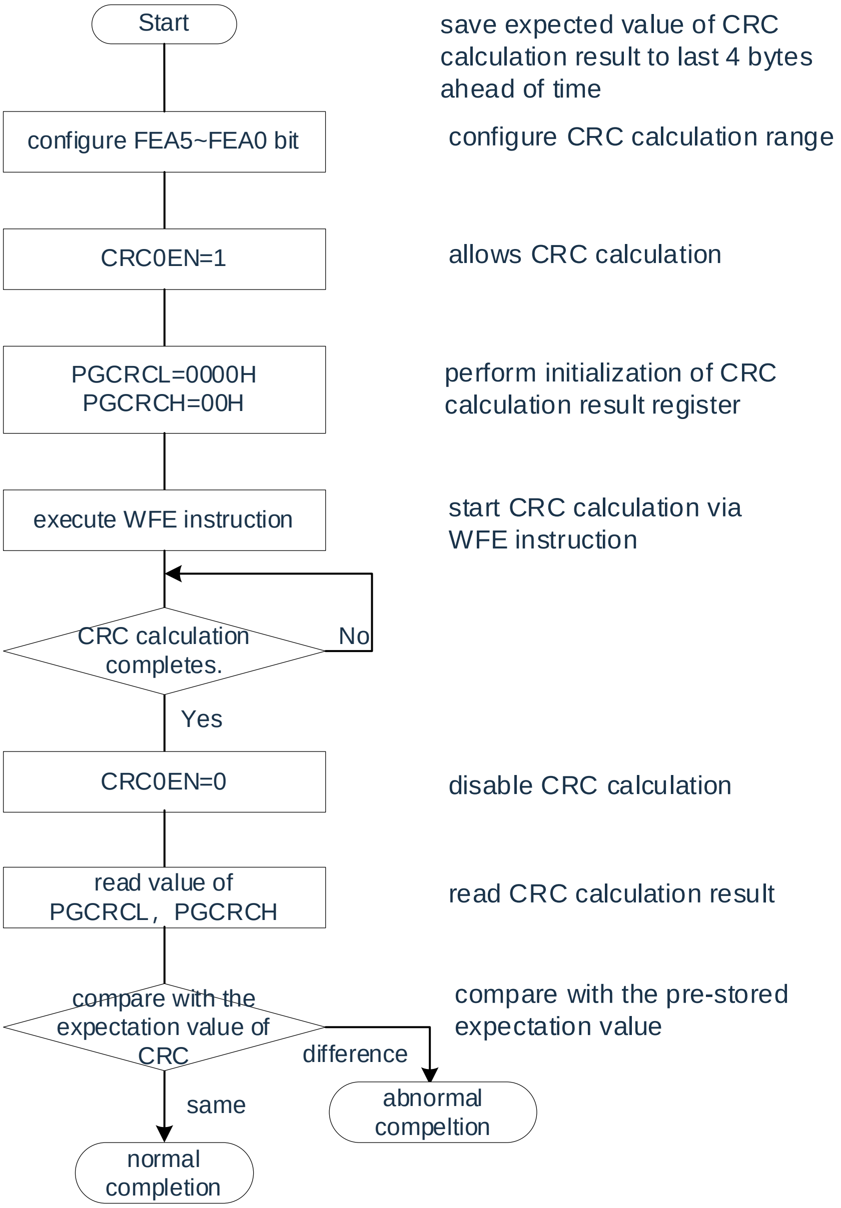
PGCRC15～0	High-speed CRC operation results
0000H～FFFFH	Store the high-speed CRC operation results.

Notice The PGCRCL register can only be written if the CRC0EN (bit7 of the CRC0 CTL register) bit is "1".

The flow chart of the flash CRC operation function (high speed CRC) is shown in Figure 30-3.

<Operation flow>

Figure 30-3 Flowchart for flash memory CRC operation function (high-speed CRC)


Start
save expected value of CRC calculation result to last 4 bytes ahead of time
configure FEA5~FEA0 bit
configure CRC calculation range
CRC0EN=1
allows CRC calculation
PGCRCL=0000H
PGCRCH=00H
perform initialization of CRC calculation result register
execute WFE instruction
start CRC calculation via WFE instruction
CRC calculation completes.
No
Yes
CRC0EN=0
disable CRC calculation
read value of PGCRCL， PGCRCH
read CRC calculation result
compare with the expectation value of CRC
compare with the pre-stored expectation value
difference
abnormal compeltion
same
normal completion

Note 1. The CRC operation is executed only on the code flash.

2. Store the expected CRC operation value in the area below the operation range in the code flash.

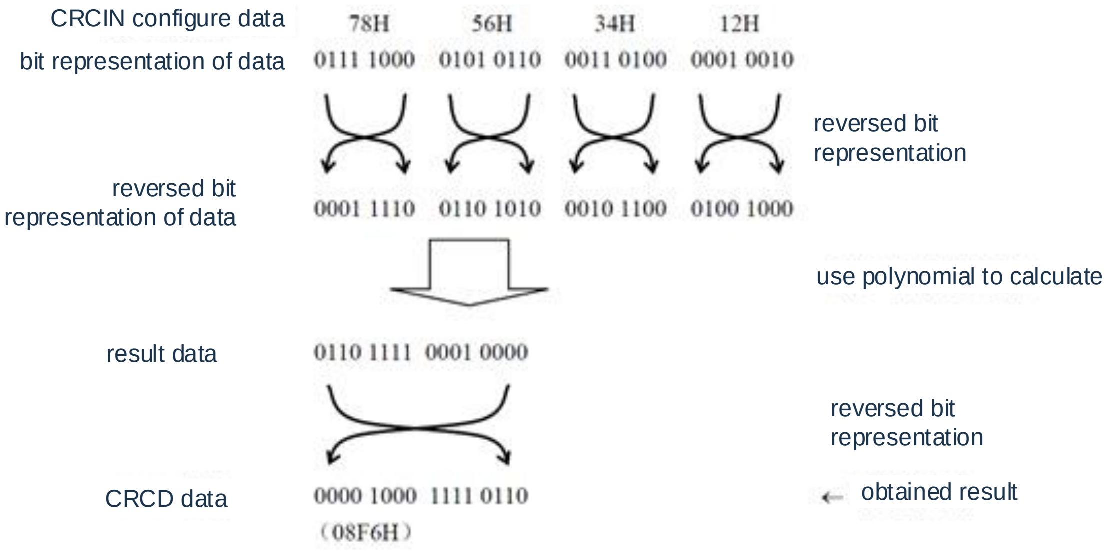
30.3.1.3 CRC operation function (general-purpose CRC)

In order to ensure safety during operation, the IEC61508 standard requires data to be confirmed even in CPU operation.

The general-purpose CRC can perform CRC operation as a peripheral function in CPU operation. The general-purpose CRC is not limited to code flash areas and can be used for multi-purpose inspection. Specify the data to be confirmed by software (user program). The CRC operation in sleep mode can only be used during DMA transfer.

The CRC operation function can be used in the main system clock operation mode or the sub-system clock operation mode.

The CRC generation polynomial uses the “$X^{16}+X^{12}+X^{5}+1$” of CRC-16-CCITT. Since communication is performed with LSB first order, calculation is performed after the bit order of the input data is reversed. For example, when sending data "12345678H" from LSB, the CRCIN register is written to the CRCD register in the order of "78H", "56H, "34H, and "12H". This is the result of a CRC operation for the following bit sequence after reversing the bit sequence of the data "12345678H".



Note Because the debugger rewrites the software break setting line to a break instruction during program execution, the CRC operation result differs if a software break is set in the CRC operation target area.

30.3.1.4 CRC input register (CRCIN)

CRCIN register is an 8-bit register that is used to set the CRC operation data of general-purpose CRC.

The CRCIN register is set by an 8-bit memory operation instruction. After the reset signal is generated, the value of this register changes to "00H".

Figure 30-4 Format of CRC input register (CRCIN)

Address: 400433 ACH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CRCIN								

Bit7~0	Function
00H~FFH	Data input

30.3.1.5 CRC data register (CRCD)

This is a register that holds the results of a generic CRC operation. The range that can be set is “0000H~FFFFH”.

After writing the CRCIN register, the CRC result is saved to the CRCD register after 1 CPU/peripheral hardware clock (fCLK). The CRCD register is set by a 16-bit memory operation instruction.

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

Figure 30-5 Format of CRC data register (CRCD)

Address: 400432FAH After reset: 0000H R/W

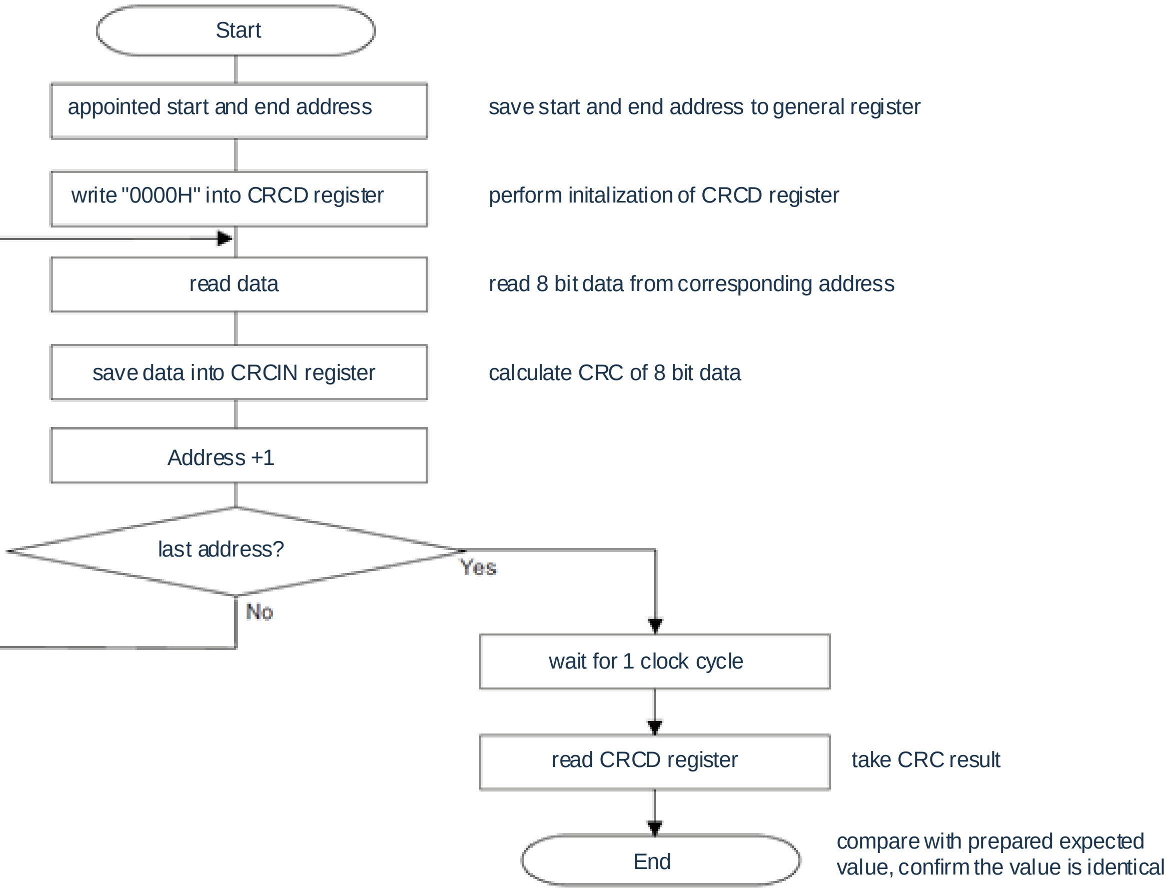
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCD																

Note 1. Read the value written to CRCD register before writing to CRCIN register.

2. If conflict between writing and storing operation result to CRCD register occurs, the writing is ignored.

<Operation flow>

Figure 30-6 Flowchart of CRC operation function (general-purpose CRC)


Start
appointed start and end address
save start and end address to general register
write "0000H" into CRCD register
perform initalization of CRCD register
read data
read 8 bit data from corresponding address
save data into CRCIN register
calculate CRC of 8 bit data
Address +1
last address?
Yes
No
wait for 1 clock cycle
read CRCD register
take CRC result
End
compare with prepared expected value, confirm the value is identical

30.3.2 RAM parity check error detection

The IEC60730 standard requires that RAM data be acknowledged. Therefore, BAT32G157's RAM is appended with 1-bit parity bits per 8 bits. The RAM parity error detection function appends parity bits when writing data and checks parity bits when reading data, and can generate a reset when a parity error occurs.

30.3.2.1 RAM parity error control register (RPECTL)

This register controls the error-confirmed bit of the parity and the reset due to the parity error.

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

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

Figure 30-7 Format of RAM parity error control register (RPECTL)

Address: 40020425H After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
RPECTL	RPERDIS	0	0	0	0	0	0	RPEF

RPERDIS	Parity error reset mask flag
0	Enable parity error resets.
1	Disable parity error resets.

RPEF	Parity error status flag
0	No parity errors occurred.
1	A parity error occurred.

Notice The parity bit is appended when the data is written and checked when the data is read.
Therefore, to allow a RAM parity error reset (RPERDIS=0) to be generated, the "used RAM area" must be initialized when data is accessed and before data is read.
Because of pipelined operation, the CPU performs pre-reading, and a RAM parity error may occur due to the uninitialized RAM area before reading the used RAM area. Therefore, to allow a RAM parity error reset (RPERDIS=0), the "used RAM area +10 bytes" area must be initialized when executing an instruction from the RAM area.

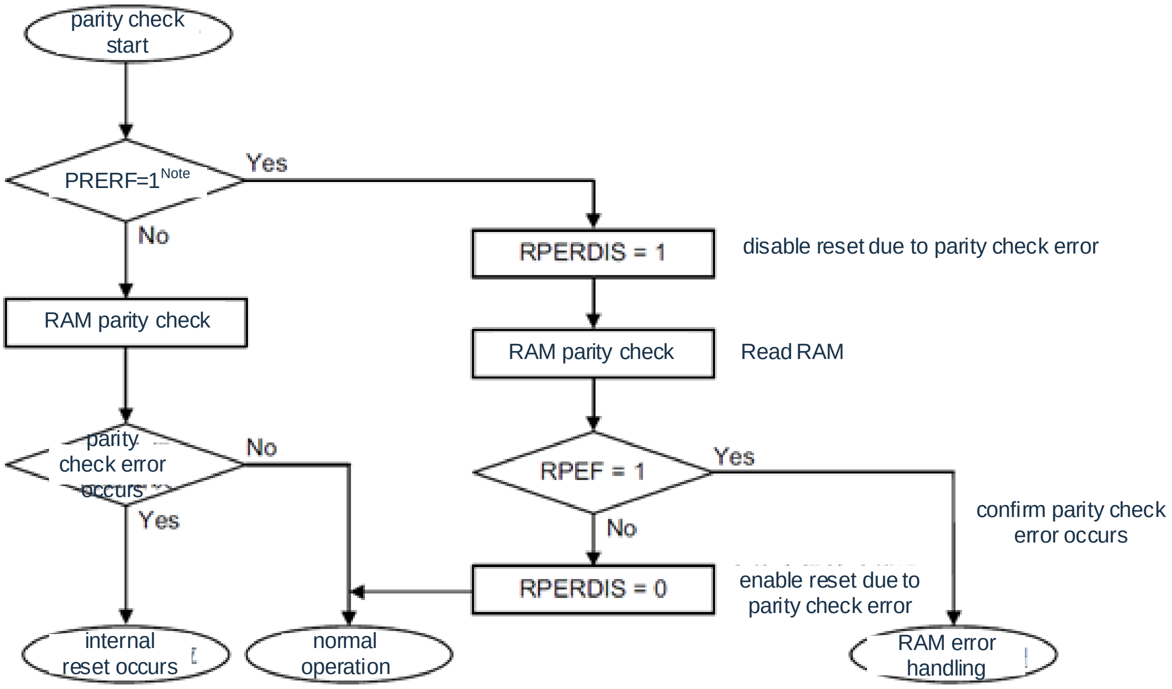
Remark 1. The initial state is allowed for parity error reset (RPERDIS=0).

2. Even if it is set to prohibit generation of parity check error reset (RPERDIS=1), the RPEF flag is set to "1" when a parity check error occurs. If the RPEF bit is set to allow generation of parity check error reset (RPERDIS=0) in the state where the RPEF bit is "1", parity check error reset is generated when RPERDIS is cleared to "0".

3. The RPEF flag in the RPECTL register is set to "1" by a RAM parity error, and the RPEF flag is cleared to "0" by writing "0" or all reset sources. When the RPEF flag is "1", the RPEF flag remains "1" even if the RAM is read without a parity error.

4. The scope of RAM parity detection does not include general-purpose registers.

Figure 30-8 Flow of RAM parity check



Note For confirmation of the internal reset of RAM parity errors, refer to Chapter 27 Reset Features, section.

30.3.3 SFR protection function

In order to ensure safety during operation, the IEC61508 standard requires that even if the CPU is out of control, critical SFRs must be protected from rewriting. The SFR protection function is used to protect data of control registers of comparator function, port function, interrupt function, clock control function, voltage detection circuit and RAM parity error detection function.

When set to SFR protection, the protected SFR writes are invalid but can be read normally.

30.3.3.1 SFR protection control register (SFRGD)

This register controls whether the SFR protection feature is valid.

The SFR protection feature uses GCOMP bits, GPORT bits, GINT bits, and GCSC bits.

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

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

Figure 30-9 Format of SFR protection control register (SFRGD)

Address: 40040478H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
SFRGD	0	0	0	0	GCOMP	GPORT	GINT	GCSC

GCOMP	Protection of control registers for comparator functions
0	Invalid. Can read and write the control registers for the comparator function.
1	Valid. The port-functional control register has invalid write operation and can be read. [Protected SFR]COMPPMDR, COMPFIR, COMPOCR, CVRCTL, CxRVM, PGAxCTL, PGASHMD, CMPSELx

GPORT	Port-functional control register protection
0	Invalid. Can read and write control registers for port functions.
1	Valid. The port-functional control register has invalid write operation and can be read. [Protected SFR]PMxx, PUxx, PIMxx, POMxx, PMCxx, ADPC,PIORx Note

GINT	Protection of register for interrupt function
0	Invalid. Can read and write control registers for interrupt functions.
1	Valid. The write operation of the control register of the interrupt function is invalid and can be read. [Protected SFR]IFxx, MKxx, PRxx, EGPx,EGNx

GCSC	Protection of control registers for clock control function, voltage detection circuit and RAM parity error detection function
0	Invalid. Can read and write the control registers for the clock control function, the voltage detection circuit, and the RAM parity error detection function.
1	Valid. The write operation of the control register of the clock control function, the voltage detection circuit and the RAM parity error detection function is invalid and can be read. [Protected SFR]CMC, CSC, OSTS, CKC, PERx, OSMC, LVIM, LVIS, RPECTL.

Note Pxx (port register) is not protected.

30.3.4 Frequency detection function

The IEC60730 standard requires that the oscillation frequency be confirmed as normal.

The frequency detection function can use the CPU/peripheral hardware clock frequency (f_{CLK}) and judge whether the ratio relation of two clocks is correct.

However, if one clock or two clocks stop oscillating, the ratio relation of two clocks cannot be judged.

<Clocks to compare>

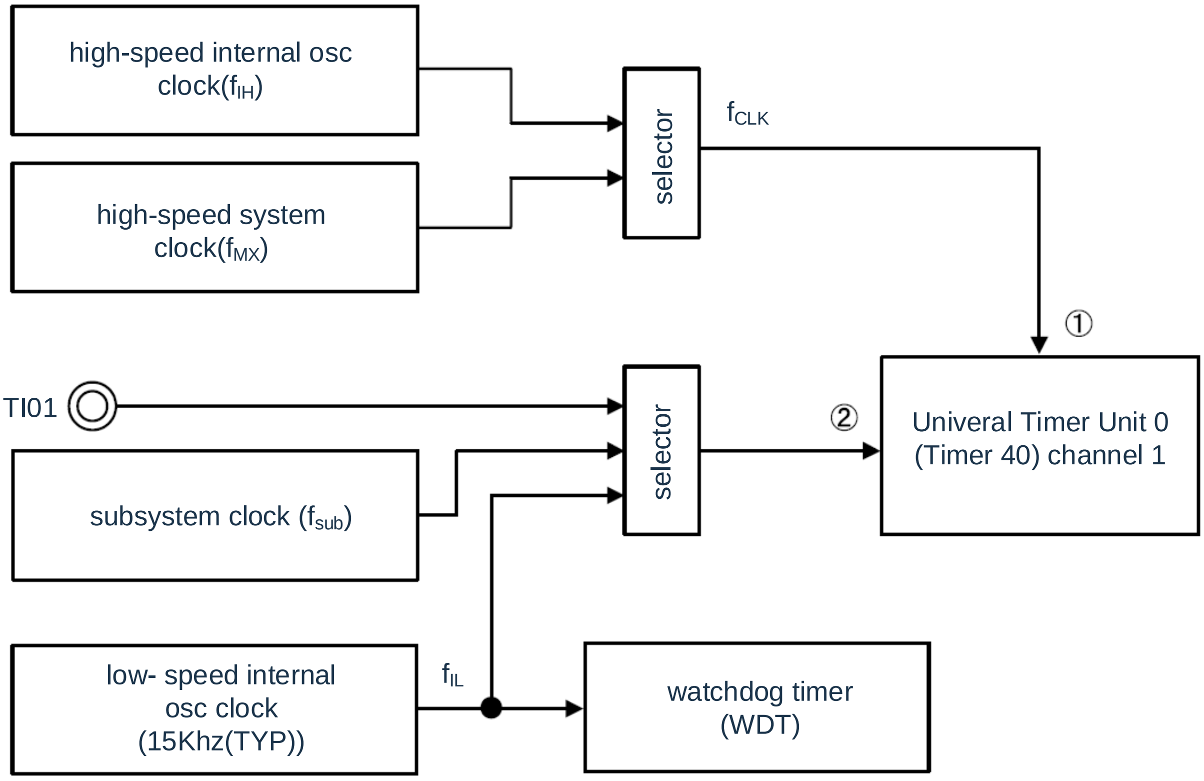
(1) Clock frequency of CPU/peripheral hardware (f_{CLK}):

- High-speed internal oscillator clock (f_{IH})
- High-speed system clock (f_{MX})

(2) Channel 1 input for Timer40:

- Timer input for channel 1 (TI01)
- Low speed internal oscillator clock (f_{IL}:15kHz (TYP.))
- Subsystem clock (f_{SUB}) Note

Figure 30-10 Structure of frequency detection function


high-speed internal osc clock(f_{IH})
high-speed system clock(f_{MX})
selector
f_{CLK}
①
TI01
subsystem clock (f_{sub})
selector
②
Univeral Timer Unit 0 (Timer 40) channel 1
low- speed internal osc clock (15Khz(TYP))
f_{IL}
watchdog timer (WDT)

If the measurement result of the input pulse interval is abnormal, it can be recognized as "clock frequency abnormality". For the measurement method of the input pulse interval, refer to "6.8.4 Operation As Input Pulse Interval Measurement".

Note Only products with a subsystem clock built in can be selected.

30.3.4.1 Timer input select register 0 (TIS0)

For register descriptions, refer to Section 6.3.8.

30.3.5 A/D test function

The IEC60730 standard requires A/D converter testing. This A/D test function confirms whether the A/D converter is operating properly by performing A/D conversion of the A/D converter's positive (+) reference voltage, negative (-) reference voltage, analog input channel (ANI), temperature sensor output voltage, and internal reference voltage.

The analog multiplexer can be confirmed by the following steps:

① Select the ANIx pin as the A/D conversion object (ADTES1, ADTES0=0, 0) through the ADTES register.

② A/D conversion of the ANIx pin (conversion result 1-1).

③ Negative (-) reference voltage of A/D converter is selected as A/D conversion object (ADTES1, ADTES0=1,0) through ADTES register.

④ An A/D conversion is performed on the negative (2-1) reference voltage of the A/D converter.

⑤ Select the ANIx pin as the A/D conversion object (ADTES1, ADTES0=0, 0) through the ADTES register.

⑥ A/D conversion of the ANIx pin (conversion result 1-2).

⑦ Positive (+) reference voltage of A/D converter is selected as A/D conversion object (ADTES1, ADTES0=1,1) through ADTES register.

⑧ A/D conversion of positive (+) reference voltage of A/D converter (conversion result 2-2).

⑨ Select the ANIx pin as the A/D conversion object (ADTES1, ADTES0=0, 0) through the ADTES register.

⑩ A/D conversion of the ANIx pin (conversion result 1-3).

⑪ Confirm that Conversion Results 1-1, Conversion Results 1-2, and Conversion Results 1-3 are identical.

⑫ Confirm that A/D conversion results for Conversion Results 2-1 are all "0" and A/D conversion results for Conversion Results 2. By the above steps, the analog multiplexer can be selected and the wiring is confirmed not broken.

Note 1. During the conversion from ①～⑩, if the analog input voltage is variable, other methods must be used to confirm the analog multiplexer.

2. The conversion results contain errors, so it is necessary to consider the error when comparing the conversion results.

30.3.5.1 A/D test register (ADTES)

The register selects positive (+) reference voltage, negative (-) reference voltage, analog input channel (ANIxx), temperature sensor output voltage and internal reference voltage (1.45V) as AD conversion objects.

When used as an A/D test function, set the following:

- When measuring the zero scale, a negative (-) reference voltage is selected as the A/D conversion object.
- When the full scale is measured, a positive (+) reference voltage is selected as the A/D conversion object.

For register descriptions, refer to 14.2.10.

30.3.5.2 Analog input channel specification register (ADS)

This register specifies the input channel of the analog voltage of the A/D conversion.

To measure the ANIxx, temperature sensor output, or internal reference voltage (1.45V) via the A/D test function, the A/D test register (ADTES) must be set to "00H".

Refer to 14.2.7 for register descriptions.

30.3.6 Digital output signal level detection function of input/output pin

The IEC60730 standard requires that I/O functionality be confirmed.

The digital output signal level detection function of the input/output pin can read the digital output level of the pin when the pin is in the output mode.

30.3.6.1 Port mode selection register (PMS)

This register selects whether the pin is output mode (PMm) with PMmn bit "0" for the read port or the pin output level.

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

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

Figure 30-11 Format of port mode selection register (PMS)

Address: 4004087BH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
PMS	0	0	0	0	0	0	0	PMS0

PMS0	Selection of read data when the pin is in output mode
0	Read the value of the Pmn register.
1	The digital output level of the read pin.

Note 1. For pins that use the pulse output force cutoff function of the timer M to make the pins high impedance, the read value is "0" if the digital output level of the read pin.

Remark m=0~7, 12~14
n=0~7

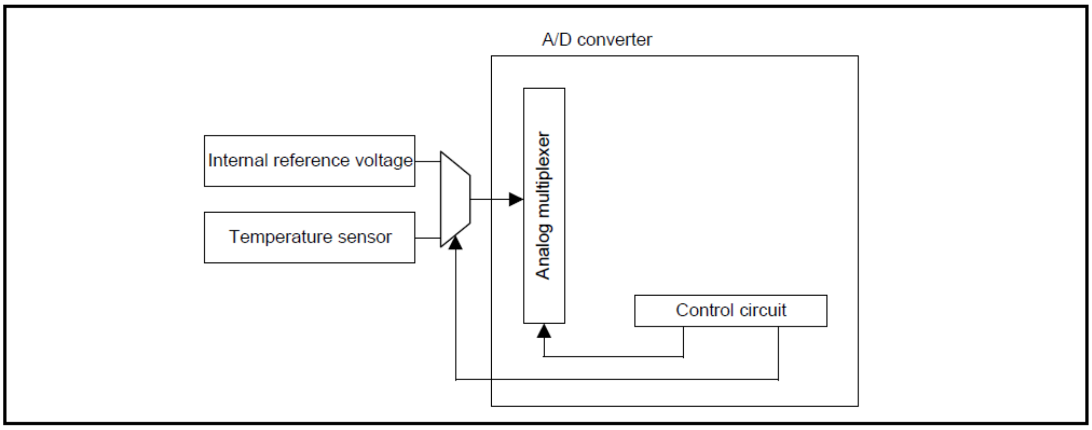
Chapter 31 Temperature Sensor

31.1 Function of temperature sensor

The temperature sensor on the chip can measure and monitor the core temperature of the product, thus ensuring the reliable operation of the product. The output voltage of the temperature sensor is proportional to the core temperature, and a linear relationship between the voltage and the temperature. Its output voltage is supply to ADC for conversion.

Figure 31-1 shows a block diagram of the temperature sensor.

Figure 31-1 Block diagram of temperature sensor



31.2 Registers for temperature sensors

31.2.1 Temperature sensor calibration data register TSN25

Address: 0x0850_0C6C

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	After reset	R/W
TSN25	-	-	-	-	TSN25[11:0]												-	R

A read-only register for recording the calibration data 1 of the temperature sensor, automatically loaded when the power is turned on or reset is started, each chip having its own calibration data.

31.2.2 Temperature sensor calibration data register TSN85

Address: 0x0850_0C68

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	After reset	R/W
TSN85	-	-	-	-	TSN85[11:0]												-	R

A read-only register for recording calibration data 2 of the temperature sensor, automatically loaded when the power supply is turned on or reset is started, each chip having its own calibration data.

31.3 Instructions for using temperature sensors

31.3.1 Principle of using temperature sensor

The temperature (T) is proportional to the sensor voltage output (Vs), so the temperature is calculated as follows:

T = (Vs - V1) / slope + T1

T: Measured temperature (°C)

Vs: Output voltage (V) of temperature sensor in temperature measurement

T1: Temperature measured experimentally at the first point (°C)

V1: Voltage output (V) when the temperature sensor measures T1

T2: Temperature measured experimentally at the second point (˚C)

V2: Voltage output (V) when the temperature sensor measures T2

Slope: Temperature slope of the temperature sensor (V/°C), slope = (V2 - V1)/(T2 - T1).

The characteristics of different sensors are different, so we suggest to measure two different sample temperatures:

1. The voltage V1 output by the temperature sensor at temperature T1 is measured using an A/D converter.
2. The voltage V2 output by the temperature sensor at the second temperature T2 is measured using an A/D converter.
3. The slope of temperature was calculated from both results (slope = (V2 - V1)/(T2 - T1))
4. Subsequently, the temperature (T = (Vs -V1)/ slope + T1) is obtained by substituting the slope into the formula of the temperature characteristic.

31.3.2 Method for using temperature sensor

Method 1: In this product, the TSCDR1 register stores a voltage conversion value (CAL25) of the temperature sensor measured at Ta=Tj=25°C and VDD=3.0v. The TSCDR2 register stores the voltage conversion value (CAL85) of the temperature sensor measured at Ta=Tj=85°C and VDD=3.0v. Using these two sets of values, the temperature slope can be calculated:

slope = (V2 - V1)/(85 - 25).
V1 = 3.0 × CAL25 / 256 [V]
V2 = 3.0 × CAL85 / 256 [V]

Using the above results, the temperature can be calculated according to the following formula:

T = (Vs - V1) / slope + 25 [°C]
T: Measured temperature (℃)
Vs: Output voltage (V) at T temperature of a temperature sensor obtained using an A/D converter

Method 2: If you use the temperature slope given in Electrical Characteristics, you can calculate the measured temperature directly using the following formula:

T = (Vs - V1) / slope + 25 [°C]

Note: The temperature generated by this method is lower than the accuracy measured by the method 1.

Chapter 32 Option Byte

32.1 Functions of option bytes

Address 000C0H~000C8H, 8500004 of the flash memory of BAT32G157 form an option byte area.

The option byte consists of the user option byte (000C0H to 000C2H, 000C4H) and the flash data protection option byte (000C3H, 8500004H, 000C5H to 000C7H). Upon power application or resetting and starting, an option byte is automatically referenced and a specified function is set. When using the product, be sure to set the following functions by using the option bytes. For the bits to which no function is allocated, do not change their initial values.

Notice The option bytes should always be set regardless of whether each function is used.

32.1.1 User option byte (000C0H～000C2H, 000C4H)

(1) 000C0H

- Setting of watchdog timer operation
  - Enabling or disabling of counter operation
  - Enabling or disabling of counter operation in the sleep/deep sleep mode.
  - Watchdog timer overflow time setting
- Watchdog timer window open period setting
- Watchdog timer interval interrupt setting
  - Use or do not use interval interruptions.

(2) 000C1H

- Setting of LVD operation mode
  - Interrupt & reset mode
  - Reset mode
  - Interrupt mode
  - LVD off (by controlling the externally input reset signal on the RESETB pin)
- Setting of LVD detection level (V_{LVDH}, V_{LVDL}, V_{LVD})

Notice: After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in AC Characteristics of the data sheet. This is done by utilizing the voltage detection circuit or controlling the externally input reset signal. After the power supply is turned off, this LSI should be placed in the deep sleep mode, or placed in the reset state by utilizing the voltage detection circuit or controlling the externally input reset signal, before the voltage falls below the operating range.

The range of operating voltage varies with the setting of the user option byte (000C2H).

(3) 000C2H

- Setting of high-speed internal oscillator frequency

Select from 1MHz to 32MHz, 48MHz, and 64MHz.

(4) 000C4H

- Setting of Boot area and BOOT area size
  - The boot area is selected from the main flash area, BOOT area, internal SRAM, and extended FLASH.
  - BOOT area size is selected from 0KB, 4KB, 8KB, 16KB.

32.1.2 Flash memory data protection option byte (000C3H, 000C5H~000C7H, 8500004H)

- Control of on-chip debug operation

  Level0: Enabling read/write/erase operations on flash data via debugger.

  Level1: Chip full erase operations on flash data via debugger are enabled, read and write operations are disabled.

  Level2: Manipulation of flash data via debugger is disabled.

- Data protection for external SPI FLASH

  External SPI FLASH can be encrypted using 16-bit encoding.

32.2 Format of user option byte

Figure 32-1 Format of user option byte (000C0H)

Address: 000C0H

Symbol	7	6	5	4	3	2	1	0
	WDTINT	WINDOW1	WINDOW0	WDTON	WDCS2	WDCS1	WDCS0	WDSTBYON

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

WINDOW1	WINDOW0	Watchdog timer window open period Note1
0	-	Setting prohibited
1	0	75%
1	1	100%

WDTON	Operation control of watchdog timer counter
0	Counter operation disabled (counting stopped after reset)
1	Counter operation enabled (counting started after reset)

WDCS2	WDCS1	WDCS0	Watchdog timer overflow time (f_{IL} = 20 kHz (MAX.))
0	0	0	$2^6/f_{IL}$(3.2ms)
0	0	1	$2^7/f_{IL}$(6.4ms)
0	1	0	$2^8/f_{IL}$(12.8ms)
0	1	1	$2^9/f_{IL}$(25.6ms)
1	0	0	$2^{11}/f_{IL}$(102.4ms)
1	0	1	$2^{13}/f_{IL}$(409.6ms)
1	1	0	$2^{14}/f_{IL}$(819.2ms)
1	1	1	$2^{16}/f_{IL}$(3276.8ms)

WDSTBYON	Operation control for watchdog timer counter (sleep mode)
0	Counter operation stopped in sleep mode Note 2.
1	Counter operation enabled in sleep mode.

Note: 1. The window open period is 100% when WDSTBYON = 0, regardless the value of the WINDOW1 and WINDOW0 bits.

Remark: f_{IL}: Low-speed internal oscillator clock frequency

Figure 32-2 Format of user option byte (000C1H) (1/4)

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD setting (interrupt & reset mode)

Detection voltage			Setting value of option byte						
V_{LVDH}		V_{LVDL}	VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rise	drop	drop						LVIMDS1	LVIMDS0
1.98V	1.94V	1.84V	0	0	1	1	0	1	0
2.09V	2.04V					0	1		
3.13V	3.06V					0	0		
2.61V	2.55V	2.45V		1	0	1	0		
2.71V	2.65V					0	1		
3.75V	3.67V					0	0		
2.92V	2.86V	2.75V		1	1	1	0		
3.02V	2.96V					0	1		
4.06V	3.98V					0	0		
—			Setting values other than those mentioned above is prohibited.						

Notice: Bit4 must be written as "1".

Remark: 1. For details of the LVD circuit, refer to “Chapter 29 Voltage Detection Circuit”.

2. The detection voltage is the TYP. value. For details, refer to LVD Circuit Characteristics in the datasheet.

Figure 32-2 Format of user option byte (000C1H) (2/4)

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD setting (reset mode)

Detection voltage		Setting value of option byte						
V_{LVD}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rise	drop						LVIMDS1	LVIMDS0
1.88V	1.84V	0	0	1	1	1	1	1
1.98V	1.94V		0	1	1	0		
2.09V	2.04V		0	1	0	1		
2.50V	2.45V		1	0	1	1		
2.61V	2.55V		1	0	1	0		
2.71V	2.65V		1	0	0	1		
2.81V	2.75V		1	1	1	1		
2.92V	2.86V		1	1	1	0		
3.02V	2.96V		1	1	0	1		
3.13V	3.06V		0	1	0	0		
3.75V	3.67V		1	0	0	0		
4.06V	3.98V		1	1	0	0		
—		Setting values other than those mentioned above is prohibited.						

Notice: Bit4 must be written as “1”.

Remark: 1. For details of the LVD circuit, refer to “Chapter 29 Voltage Detection Circuit”

2. The detection voltage is the TYP. value. For details, refer to LVD Circuit Characteristics in the datasheet.

Figure 32-2 Format of user option byte (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)

Detection voltage		Setting value of option byte						
VLVD		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rise	drop						LVIMDS1	LVIMDS0
1.88V	1.84V	0	0	1	1	1	0	1
1.98V	1.94V		0	1	1	0		
2.09V	2.04V		0	1	0	1		
2.50V	2.45V		1	0	1	1		
2.61V	2.55V		1	0	1	0		
2.71V	2.65V		1	0	0	1		
2.81V	2.75V		1	1	1	1		
2.92V	2.86V		1	1	1	0		
3.02V	2.96V		1	1	0	1		
3.13V	3.06V		0	1	0	0		
3.75V	3.67V		1	0	0	0		
4.06V	3.98V		1	1	0	0		
—		Setting values other than those mentioned above is prohibited.						

Notice: Bit4 must be written as “1”.

Remark: 1. For details of the LVD circuit, refer to “Chapter 29 Voltage Detection Circuit”.

2. The detection voltage is the TYP. value. For details, refer to LVD Circuit Characteristics in the datasheet.

Figure 32-2 Format of user option byte (000C1H) (4/4)

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- Setting when LVD is OFF (external reset input using RESETB pin)

Detection voltage		Setting value of option byte						
V_{LVDH}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rise	drop						LVIMDS1	LVIMDS0
—	—	1	×	×	×	×	×	1
—		Setting values other than those mentioned above is prohibited.						

Notice: 1.Bit4 must be written as “1”.

2. When the supply voltage rises, the reset state must be maintained by the voltage detection circuit or external reset before the supply voltage reaches the operating voltage range shown in the AC Characteristics of the datasheet; when the supply voltage falls, it must be set to the reset state by the transfer of the sleep mode, the voltage detection circuit, or the external reset before the supply voltage falls below the operating voltage range.

The operating voltage range depends on the setting of the user option byte (000C2H/010C2H).

Remark: 1.×: Ignore

2. For details of the LVD circuit, refer to “Chapter 29 Voltage Detection Circuit”.

3. The detection voltage is the TYP. value. For details, refer to LVD Circuit Characteristics in the datasheet.

Figure 32-3 Format of the user option byte (000C2H)

Address: 000C2H

7	6	5	4	3	2	1	0
1	1	1	FRQSEL4	FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0

FRQSEL4	FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0	High-speed internal oscillator clock frequency	
					f_{HOCO}	f_{IH}
1	1	0	0	0	64MHz	64MHz
1	0	0	0	0	48MHz	48MHz
0	1	0	0	0	32MHz	32MHz
0	0	0	0	0	24MHz	24MHz
0	1	0	0	1	32MHz	16MHz
0	0	0	0	1	24MHz	12MHz
0	1	0	1	0	32MHz	8MHz
0	0	0	1	0	24MHz	6MHz
0	1	0	1	1	32MHz	4MHz
0	0	0	1	1	24MHz	3MHz
0	1	1	0	0	32MHz	2MHz
0	1	1	0	1	32MHz	1MHz
Others					Setting prohibited	

Notice: 1. Bits 7 to 5 must be written as "1".

2. The operating frequency range and operating voltage range vary depending on each operating mode of the flash memory. For details, refer to AC Characteristics in the datasheet.

Figure 32-4 Format of the user option byte (000C4H)

Address: 000C4H

Symbol	7	6	5	4	3	2	1	0
	-	CONFIG[2:0]			-	BOOTSIZE[2:0]		

BOOTSIZE[2:0]	Main flash area	BOOT area Note
001	252K(0000H-3EFFFH)	4K(3F000H-3FFFFH)
010	248K(0000H-3DFFFH)	8K(3E000H-3FFFFH)
011	240K(0000H-3BFFFH)	16K(3C000H-3FFFFH)
Others	256K(0000H-3FFFFH)	-

CONFIG[2:0]	Program startup address setting
001	Boot area startup
010	Extend flash area startup
011	Ram area startup
Others	Main flash area startup

Note: If 4/8/16KB is set for BOOT area, the corresponding address area is protected and cannot be erased or written.

Refer to "Chapter 3 System Structure" for the specific application of the boot area and BOOT area size settings.

32.3 Format of flash memory data protection option byte

The format of the flash data protection option bytes is shown below.

Figure 32-5 Format of flash memory data protection option byte

Address: 000C3H

Symbol	7	6	5	4	3	2	1	0
	OCDEN[7:0]							

Address: 8500004H

Symbol	7	6	5	4	3	2	1	0
	OCDM[7:0]							

OCDM	OCDEN	Control of flash data protection
3C	C3	Manipulation of flash data via debugger is not allowed.
Values other than 3C	C3	Chip full erase operations on flash data via debugger are allowed, read and write operations are not allowed.
Others		Allows read/write/erase operations on flash data via debugger.

Figure 32-6 Format of external SPI FLASH data protection option byte (000C5H~000C7H)

Address: 000C5H

Symbol	7	6	5	4	3	2	1	0
	CRYPEN [7:0]							

Address: 000C6H

Symbol	15	14		1	0
	CRYPCODE[15:0]				

CRYPEN	Control of external SPI FLASH data protection [Note]
5A	Decrypting data when reading SPI FLASH using QSPI
Others	QSPI reads SPI FLASH without decrypting the data.

CRYPCODE	Control of external SPI FLASH data protection
Bit15-0	Encryption code for encrypting data written to SPI FLASH

Note: For data protection of SPI FLASH, please refer to "Chapter 17 QSPI".

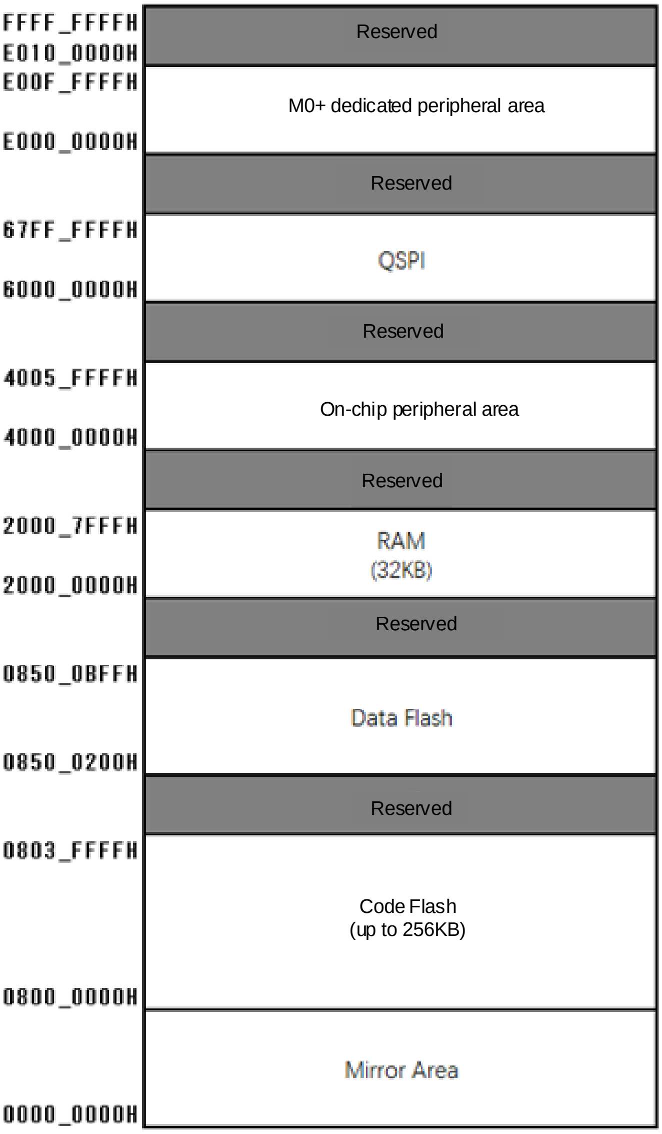
Chapter 33 FLASH Control

Note: The PLL output cannot be used as the system clock of the MCU during the Flash erase/write operation.

33.1 Description of FLASH control functions

This product contains a 256KB FLASH memory, divided into 512 segments, each segment has a capacity of 512 bytes, which can be used as program memory and data memory. This module supports erase, program and read operations on this memory. In addition, this module supports the erase/write protection of the FLASH memory and the write protection of the control registers.

33.2 Structure of flash memory



33.3 Registers for controlling FLASH

The registers that control the FLASH are shown below:

- Flash write protect register (FLPROT)
- Flash operation control register (FLOPMD1,FLOPMD2)
- Flash erase mode control register (FLERMD)
- Flash status register (FLSTS)
- Flash chip erase time control register (FLCERCNT)
- Flash sector erase time control register (FLSERCNT)
- Flash write time control register (FLPROCNT)
- Flash mode time control register (FLNVSCNT/FLPRVCNT/FLERVCNT)

33.3.1 Flash write protect register (FLPROT)

The Flash protection register is used to protect the FLASH operation control registers.

Address: 0x40020020 After reset: 00000000H R/W

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

WRP	Operation register (FLOPMD1/FLOPMD2) write protection
0	Rewriting of FLOPMD1/ FLOPMD2 is not allowed.
1	Rewriting of FLOPMD1/ FLOPMD2 is allowed.

PRKEY[7:1]	WRP write protection
78h	Rewriting of WRP is allowed.
Others	Rewriting of WRP is not allowed.

33.3.2 33.3.2 FLASH operation control register (FLOPMD1, FLOPMD2)

The Flash Operation Control Register is used to set the FLASH erase and write operations.

Address: 0x40020004 After reset: 00000000H R/W

Symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLOPMD1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	-	-	FLOPMD1[7:0]							

Address: 0x40020008 After reset: 00H R/W

Symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLOPMD2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	-	-	FLOPMD2[7:0]							

FLOPMD1	FLOPMD2	OPERATION
55	AA	Erase
AA	55	Write
00	00	Read
Others		Setting prohibited

33.3.3 Flash erase control register (FLERMD)

The Flash Erase Control Register is used to set the type of FLASH erase operation.

Address: 0x4002000C After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
FLERMD	0	0	0	ERMD1	ERMD0	0	0	0

ERMD1	ERMD0	OPERATION
0	0	Sector erase, no hardware checking after erase
1	0	Sector erase, hardware checking after erase
0	1	chip erase Note
1	1	Setting prohibited

Note: Chip erase only erases the code flash area, not the data flash area. And chip erase does not support hardware check.

33.3.4 Flash status register (FLSTS)

The status of the FLASH controller can be queried through the status register.

Address: 0x40020000 After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
FLSTS	0	0	0	0	0	EVF Note	0	OVF Note

OVF	FLASH erase operation complete flag
0	FLASH erase operation is not completed
1	FLASH erase operation is completed

Note: The OVF needs to be cleared by software writing "1". If it is not cleared, the next erase/write operation cannot be performed.

EVF	FLASH erase hardware check error flag
0	No error occurred in hardware check after FLASH erase
1	Hardware check error occurred after FLASH erase

Note: EVF needs to be cleared by writing "1" in software.

33.3.5 Flash chip erase time control register (FLCERCNT)

The FLCERCNT register allows you to set the FLASH chip erase time.

Address: 0x40020010 After reset: indefinite value R/W

Symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLCERCNT	load	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	FLCERCNT[9:0]									

Load	Selection of erase time setting Note
0	Use hardware to set erase time
1	Use software to set erase time (FLCERCNT[9:0])

Note: When the main clock is an internal high-speed OCO or an external input clock <= 20M, you can use hardware to set the time without setting FLCERCNT.

FLCERCNT[9:0]	Software erase time setting
Chip erase time = (CERCNT*2048*Tfclk), need to meet >20ms hardware requirement	

33.3.6 Flash sector erase time control register (FLSERCNT)

The FLASH sector erase time can be set through the FLSERCNT register.

Address: 0x40020014 After reset: indefinite value R/W

Symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLSERCNT	load	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	FLSERCNT[9:0]									

Load	Selection of erase time setting Note
0	Use hardware to set erase time
1	Use software to set erase time (FLSERCNT[9:0])

Note: When the main clock is an internal high-speed OCO or an external input clock <= 20M, you can use hardware to set the time without setting FLSERCNT.

FLSERCNT[9:0]	Software erase time setting
Sector erase time = (SERCNT*256*Tfclk) need to meet >4ms hardware requirement	

33.3.7 Flash write time control register (FLPROCNT)

The FLPROCNT register allows you to set the FLASH WORD write time.

Address: 0x4002001C After reset: indefinite value 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 hardware to set write times
1	Use hardware to set erase times (FLPROCNT[9:0])

Note: When the main clock is an internal high-speed OCO or an external input clock <= 20M, you can use hardware to set the time without setting FLPROCNT.

FLPROCNT[8:0]	Software erase time setting
Write time = (PROCNT*4*Tfclk), need to meet >24us hardware requirements	

Load1	Write action setup time (Tpgs) setting Note
0	Write action setup time using hardware
1	Erase time set using software (FLPGSCNT8:0])

Note: When the main clock is an internal high-speed OCO or an external input clock <= 20M, you can use hardware to set the time without setting FLPGSCNT.

FLPGSCNT[8:0]	Software erase time setting
Write action setup time = (PGSCNT*Tfclk), need to meet >5us hardware requirement	

33.3.8 Flash erase protection control register (FLSECPR)

When a sector is protected, any erase/write operation on that sector is 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	Enable rewriting SECPR[3:0]
Others	Disable rewriting SECPR[3:0]

SECPR	Register SECPR write protection
0001	The 8 sectors from 800_0000H to 800_0FFFH cannot be erased.
0010	The 16 sectors from 800_0000H to 800_1FFFH cannot be erased.
0011	The 32 sectors from 800_0000H to 800_3FFFH cannot be erased.
0100	The 64 sectors from 800_0000H to 800_7FFFH cannot be erased.
0101	The 128 sectors from 800_0000H to 800_FFFFH cannot be erased.
0110	The 256 sectors from 800_0000H to 801_FFFFH cannot be erased.
0111	All 512 sectors are not erasable.
Others	Sector is unprotected and allows erasure

33.4 How to operate FLASH

33.4.1 Sector erase

Sector erase, and the erase time are implemented by hardware or can be configured by FLSERCNT. The operation flow is as follows:

1) Set FLERMD.ERMD0 to 1'b0, select the sector erase mode, and choose to set the value of ERMD1 according to whether or not hardware check is required.

2) Set FLPROT to 0xF1 to unprotect FLOPMD. Then set FLOPMD1 to 0x55 and FLOPMD2 to 0xAA.

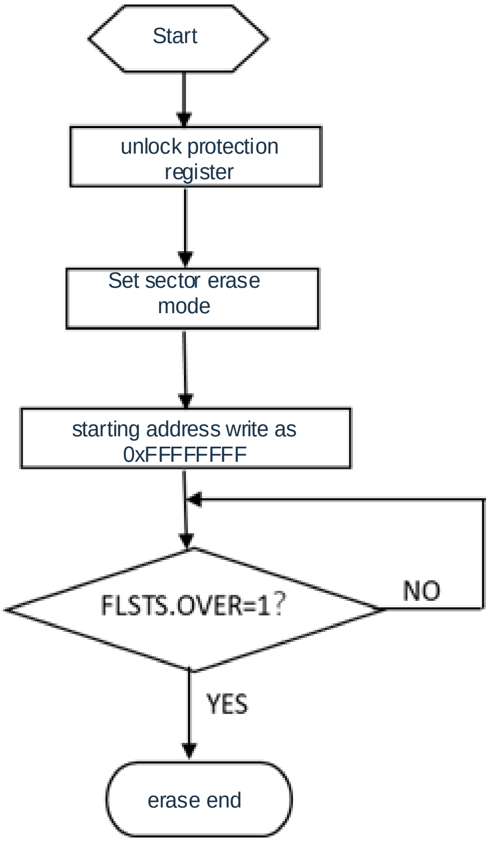
3) Write arbitrary data to the first address of the erase target sector.

Example: *((unsigned long *) 0x00000200)=0xffffffff.

4) Software query status register FLSTS.OVF,OVF=1, indicates that the erase operation is completed.

5) If hardware check after erase is set (ERMD1=1), you can judge FLSTS.EVF by software to check if the check is correct.

6) Before the next operation, the software sets "1" to clear FLSTS.


Start
unlock protection register
Set sector erase mode
starting address write as 0xFFFFFFFF
FLSTS.OVER=1?
NO
YES
erase end

33.4.2 Chip erase

Chip erase, the erase time is realized by hardware or can be configured by FLCERCNT. The operation flow is as follows:

1) Set FLERMD.ERMD0 to 1'b1 to select chip erase mode.
2) Set FLPROT to 0xF1 to unprotect FLOPMD. Then set FLOPMD1 to 0x55 and FLOPMD2 to 0xAA.
3) Write arbitrary data to any address in the code flash area.
4) Software query status register FLSTS.OVF, OVF=1, indicates that the erase operation is completed.
5) Before the next operation, the software sets "1" to clear FLSTS.

33.4.3 Word program

Word programming, the write time is realized by hardware or can be configured by PROCNT. The operation flow is as follows:

1) Set FLPROT to 0xF1 to unprotect FLOPMD. Then set FLOPMD1 to 0xAA and FLOPMD2 to 0x55.
2) Write the corresponding data to the target address.
3) Software query status register FLSTS.OVF,OVF=1, indicates that the write operation is completed.
4) Before the next operation, the software sets "1" to clear FLSTS.

33.5 Flash readinf

The fastest fetch frequency supported by the built-in FLASH of this device is 32MHz. When the HCLK frequency exceeds 32MHz, the hardware will insert 1 wait cycle when the CPU accesses the FLASH.

33.6 Cautions for FLASH operation

- FLASH memory has strict timing requirements for the control signals of the erase and programming operations; unqualified timing of the control signals will result in failure of the erase and programming operations. The setting of erase and write parameters can be realized by hardware or modified by software by modifying parameter registers. When using internal high-speed OCO with MAINOSC/external input clock = 20M, it is recommended to use the erase and write parameters set by hardware without setting parameter registers.
- If the erase/write operation is executed from within FLASH, the CPU stops fetching and the hardware automatically waits for the operation to complete before continuing to the next instruction. If the operation is executed from RAM, the CPU does not stop fetching and can currently continue with the next instruction.
- If the CPU executes an instruction to enter deep sleep while the FLASH is in a programming, the system will wait for the programming action to end before entering deep sleep.

Appendix Revision History

Version	Date	Revised content
V1.00	2020/10/22	Initial version
V1.01	2021/06/24	Corrected some image errors and text errors in section 17.4.1
V1.02	2021/07/13	Proofread and corrected.
V1.03	2022/06/02	Corrected some textual errors in chapters 19.4, 26.3, and 30.3.
V1.0.4	Aug 2023	Corrected some errors in section 22.3.4.1
V1.0.5	2023/9/28	Corrected some errors in Figure 29-1 Block diagram of voltage detection circuitand Figure 29-6 Timing of reset & interrupt signal generation (LVIMDS1, LVIMDS0=1, 0 for option byte) (1/2)
	2024/2/23	Correction to the relevant description of the register RWAIT bit in section 8.3.4 Corrected the number of multi-channel PWM signals in section 6.1.2