CMS32L051 User Manual

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

V1.2.2

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

Documentation Instructions

This manual is a technical reference manual for CMS32L051 microcontroller products, and the technical reference manual is an application note on how to use this series of products, including the structure, functional description, and function description of each functional module. Details such as operating modes and register configuration.

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

The data sheet information is as follows:

CMS32L051xx: CMS32L051xx_datasheet_vx.x.x. pdf

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

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

Contents

Chapter 1 CPU

1.1 Overview

This section briefly introduces the features and debugging features of the ARM Cortex-M0+ core on this product, please refer to the relevant ARM documentation for details.

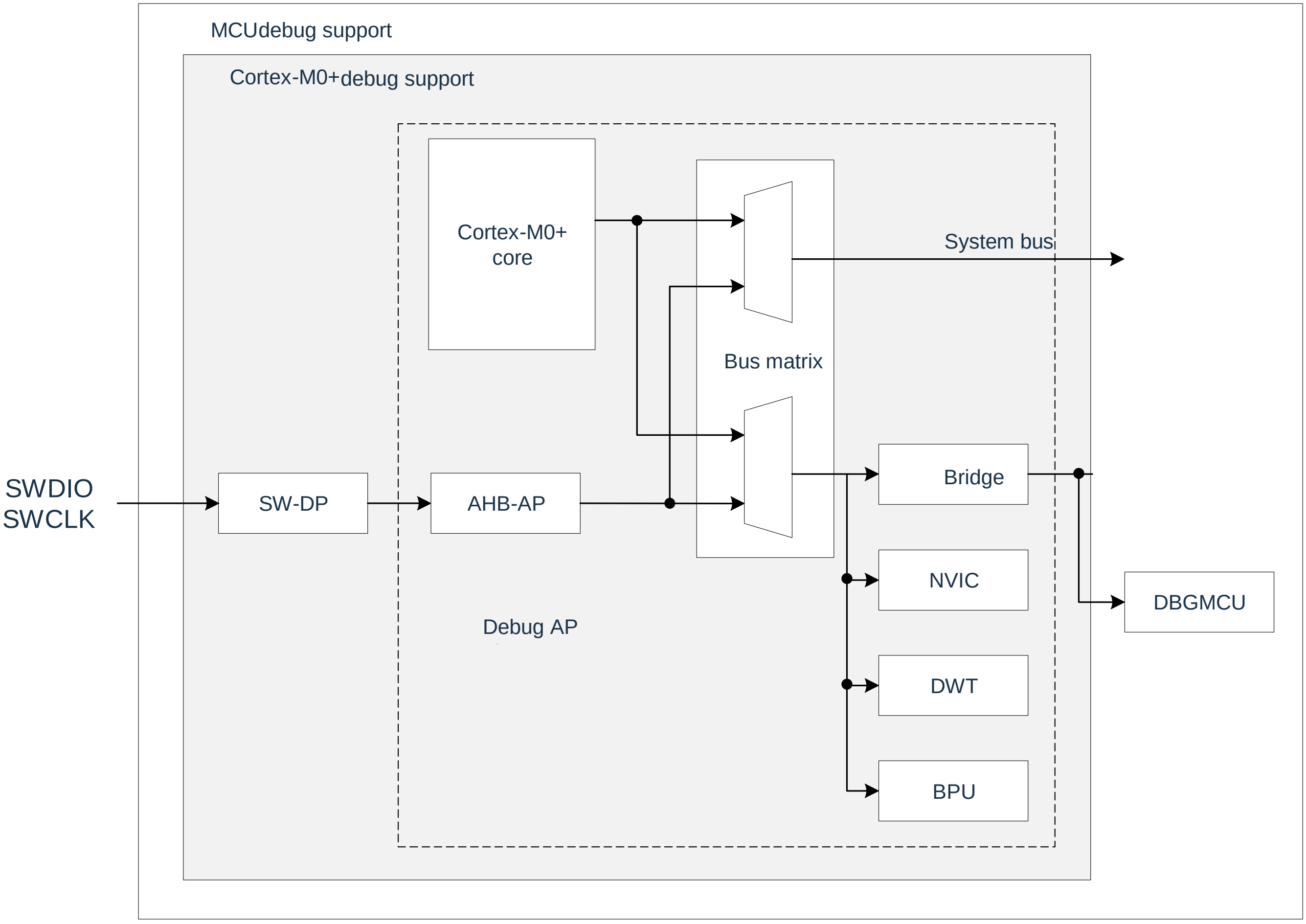
1.2 Cortex-M0+ core features

- The ARM Cortex-M0+ processor is a 32-bit RISC core with 2-stage pipeline that only supports privileged mode
- Single -cycle hardware multiplier
- Nested Vector Interrupt Controller (NVIC)
  - 1 non-maskable interrupt (NMI)
  - Supports 32 maskable interrupt requests (IRQs)
  - 4 interrupt priorities
- System Timer (SysTick) is a 24-bit countdown timer with a choice of f_{CLK} or f_{IL} counting clock
- Vector Table Offset Register (VTOR)
  - The software can write VTOR to relocate the vector table start address to a different location
  - The default value of this register is 0x0000_0000, the low 8 bits are ignored for writes, and read to zero, which means that the offset is 256 bytes aligned.

1.3 Debugging features

- 2-wire SWD debug interface
- Supports pause, resume, and stepping through programs
- Access the processor's core registers and special function registers
- 4 hardware breakpoints (BPUs).
- Unlimited software breakpoints (BKPT instructions).
- 2 Data observation points (DWTs).
- Memory is accessed while the kernel is executing.

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



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

1.4 SWD interface pin

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

Table 1-1 SWD debug port pins

SWD port name	Debugging capabilities	Pin assignment
SWCLK	Serial clock	P137
SWDIO	Serial data input/output	P40

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

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

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

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

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

SWDIS	SWD debug interface status
0	The SWD debug interface is enabled. P40 cannot be used as a GPIO when the debugger is connected (because the ENO and DOUT of the IOBUF are controlled by the debugger at this time)
1	The SWD debug interface is disabled. The P40 can be used as a GPIO

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

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

Note 1: The timer peripheral modules of this product include: Timer4, a universal timer unit.

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

1.5 ARM reference document

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

- Cortex-M0®+ Technical Reference Manual (TRM)
- ARM® debug interface V5
- ARM® CoreSight Design Kit Version r1p1 Technical Reference Manual
- ARM® CoreSight™ MTB-M0+ Technical Reference Manual

Chapter 2 Pin Function

2.1 Port function

Refer to the data sheets for each product family.

2.2 Port multiplexing function

Refer to the data sheets for each product family.

2.3 Registers for controlling port functions

The port function 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 Output Multiplexing Function Configuration Register (PxxCFG).
- Port Input Multiplexing Function Configuration Register (TI10PCFG, TI11PCFG, TI12PCFG, TI13PCFG, INTP0PCFG, INTP1PCFG, INTP2PCFG, INTP3PCFG, SDI00PCFG, SCLKI00PCFG, SS00PCFG, SDI20PCFG, SCLKI20PCFG, SDAA0PCFG, SCLA0PCFG, RXD1PCFG).
- SPI Port Multiplexing Configuration Register (SPIPCFG)

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

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

port		Bit name								48 Pins	40 Pins	32 Pins	24 Pins	20 Pins
		PMxx register	Pxx register	PSETxx register	PCLRxx register	PUxx register	PDxx register	POMxx register	PMCxx register					
Port 0	0	PM00	P00	PSET00	PCLR00	PU00	PD00	POM00	PMC00	○	○	○	—	—
	1	PM01	P01	PSET01	PCLR01	PU01	PD01	POM01	PMC01	○	○	○	—	—
Port 1	0	PM10	P10	PSET10	PCLR10	PU10	PD10	POM10	PMC10	○	○	○	○	○
	1	PM11	P11	PSET11	PCLR11	PU11	PD11	POM11	PMC11	○	○	○	○	○
	2	PM12	P12	PSET12	PCLR12	PU12	PD12	POM12	PMC12	○	○	○	○	○
	3	PM13	P13	PSET13	PCLR13	PU13	PD13	POM13	PMC13	○	○	○	○	○
	4	PM14	P14	PSET14	PCLR14	PU14	PD14	POM14	PMC14	○	○	○	○	○
	5	PM15	P15	PSET15	PCLR15	PU15	PD15	POM15	PMC15	○	○	○	○	—
	6	PM16	P16	PSET16	PCLR16	PU16	PD16	POM16	PMC16	○	○	○	—	—
	7	PM17	P17	PSET17	PCLR17	PU17	PD17	POM17	PMC17	○	○	○	—	—
Port 2	0	PM20	P20	PSET20	PCLR20	PU20	PD20	POM20	PMC20	○	○	○	○	○
	1	PM21	P21	PSET21	PCLR21	PU21	PD21	POM21	PMC21	○	○	○	○	○
	2	PM22	P22	PSET22	PCLR22	PU22	PD22	POM22	PMC22	○	○	○	○	○
	3	PM23	P23	PSET23	PCLR23	PU23	PD23	POM23	PMC23	○	○	○	○	○
	4	PM24	P24	PSET24	PCLR24	PU24	PD24	POM24	PMC24	○	○	—	○	—
	5	PM25	P25	PSET25	PCLR25	PU25	PD25	POM25	PMC25	○	○	—	—	—
	6	PM26	P26	PSET26	PCLR26	PU26	PD26	POM26	PMC26	○	—	—	—	—
	7	PM27	P27	PSET27	PCLR27	PU27	PD27	POM27	PMC27	○	—	—	—	—

Note 1.(-A) is limited to CMS32L051xx-A series products.

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

port		Bit name								48 Pins	40 Pins	32 Pins	24 Pins	20 Pins
		PMxx register	Pxx register	PSETxx register	PCLRxx register	PUxx register	PDxx register	POMxx register	PMCxx register					
Port 3	0	PM30	P30	PSET30	PCLR30	PU30	PD30	POM30	PMC30	○	○	○	—	—
	1	PM31	P31	PSET31	PCLR31	PU31	PD30	POM31	PMC31	○	○	○	—	—
Port 4	0	PM40	P40	PSET40	PCLR40	PU40	—	POM40	—	○	○	○	○	○
	1	PM41	P41	PSET41	PCLR41	PU41	—	POM41	—	○	—	—	—	—
Port 5	0	PM50	P50	PSET50	PCLR50	PU50	PD50	POM50	PMC50	○	○	○	—	—
	1	PM51	P51	PSET51	PCLR51	PU51	PD51	POM51	PMC51	○	○	○	—	—
Port 6	0	PM60	P60	PSET60	PCLR60	PU60[Note2]	PD60[Note2]	POM60[Note 2]	PMC60[Note 2]	○	○	—	—	—
	1	PM61	P61	PSET61	PCLR61	PU61[Note2]	PD61[Note2]	POM61[Note 2]	PMC61[Note 2]	○	○	—	—	—
	2	PM62	P62	PSET62	PCLR62	PU62	PD62	POM62	PMC62	○	—	—	—	—
	3	PM63	P63	PSET63	PCLR63	PU63	PD63	POM63	PMC63	○	—	—	—	—
Port 7	0	PM70	P70	PSET70	PCLR70	PU70	PD70	POM70	PMC70	○	○	○	○	—
	1	PM71	P71	PSET71	PCLR71	PU71	PD71	POM71	PMC71	○	—	—	○	—
	2	PM72	P72	PSET72	PCLR72	PU72	PD72	POM72	PMC72	○	○	○	○	—
	3	PM73	P73	PSET73	PCLR73	PU73	PD73	POM73	PMC73	○	○	○	○	—
	4	PM74	P74	PSET74	PCLR74	PU74	PD74	POM74	PMC74	○	○	○	○	—
	5	PM75	P75	PSET75	PCLR75	PU75	PD75	POM75	PMC75	○	○	—	○	—
Port 12	0	PM120	P120	PSET120	PCLR120	PU120	PD120	POM120	PMC120	○	○	○	—	—
	1	PM121	P121	PSET121	PCLR121	—	—	—	—	○	○	○	○	○
	2	PM122	P122	PSET122	PCLR122	—	—	—	—	○	○	○	○	○
	3	PM123	P123	PSET123	PCLR123	—	—	—	—	○	○	—	—	○
	4	PM124	P124	PSET124	PCLR124	—	—	—	—	○	○	—	—	○
Port 13	0	PM130	P130	PSET130	PCLR130	PU130	PD130	POM130	PMC130	○	—	—	—	—
	6	PM136	P136	PSET136	PCLR136	PU136	PD136	POM136	PMC136	○	○	○	—	○
	7	PM137	P137	PSET137	PCLR137	PU137	—	POM137	—	○	○	○	○	○
Port 14	0	PM140	P140	PSET140	PCLR140	PU140	PD140	POM140	PMC140	○	○	—	—	—
	6	PM146	P146	PSET146	PCLR146	PU146	PD146	POM146	PMC146	○	—	—	—	—
	7	PM147	P147	PSET147	PCLR147	PU147	PD147	POM147	PMC147	○	○	○	—	—

Note: 1. (-A) indicates that it is limited to CMS32L051xx-A series products.

2. It is limited to CMS32L051xx-S series products. For products other than the CMS32L051xx-S series, ports P60, P61 are dedicated N-channel open drain output ports, which do not need to be configured with POM registers and do not have their own pull-up and pull-down functions and must be used with external pull-up resistors, and can only be used as digital pins.

2.3.1 Port mode register (PMxx)

When a port is used as a digital channel, this is the register that sets its input/output in bits. After the reset signal is generated, the ports except the P130 port default to the input state. When using a port pin as a pin for the multiplexed function, it must be set with reference to “2.5 Register Settings When Using the Multiplexing Function”.

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

Figure 2-1 Format of port mode register

symbol	7	6	5	4	3	2	1	0	offset address	after reset	R/W
PM0	1	1	1	1	1	1	PM01	PM00	0x020	FFH	R/W
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	0x021	FFH	R/W
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20	0x022	FFH	R/W
PM3	1	1	1	1	1	1	PM31	PM30	0x023	FFH	R/W
PM4	1	1	1	1	1	1	PM41	PM40	0x024	FFH	R/W
PM5	1	1	1	1	1	1	PM51	PM50	0x025	FFH	R/W
PM6	1	1	1	1	PM63	PM62	PM61	PM60	0x026	FFH	R/W
PM7	1	1	PM75	PM74	PM73	PM72	PM71	PM70	0x027	FFH	R/W
PM12	1	1	1	PM124	PM123	PM122	PM121	PM120	0x02C	FFH	R/W
PM13	PM137	PM136	1	1	1	1	1	PM130	0x02D	FEH	R/W
PM14	PM147	PM146	1	1	1	1	1	PM140	0x02E	FFH	R/W

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

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

2.3.2 Port register (Pxx).

This is the register that sets the value of the port output latch in bits. Reading this register in input mode gives the pin level, while reading it in output mode gives the value of the port's output latch. After the reset signal is generated, the value of the register becomes “00H”.

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

Figure 2-2 Format of port register

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

Pmn	m=0~7, 12~1 4, n=0~7	
	Control of output data (output mode)	Reading of input data (input mode)
0	Output “0”.	Input low level.
1	Output “1”.	Input high level.

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

2. It indicates that it is limited to CMS32L051xx-S series products only. When the products are not CMS32L051xx-S series, ports P60 and P61 are dedicated N-channel open-drain output ports, which can only output “0” and “Hiz “.

2.3.3 Port set control register (PSETxx)

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

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

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

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

2.3.4 Port clear control register (PCLRxx)

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

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

Figure 2-4 Format of port clear control register

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

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

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

2.3.5 Pull-up resistor selection register (PUxx)

Selection register for internal pull-up resistors. The internal pull-up resistor can only be used in bits for bits specified by the pull-up resistor select register using the pin using the internal pull-up resistor and the POMmn bit is “0” and set to input mode (PMmn=1). For bits set to output mode, independent of the setting of the pull-up resistor selection register, the internal pull-up resistor is not connected. The same is true when used as the output pin for the multiplexing function or when set to an analog function.

After the reset signal is generated, the pull-up function of the four ports P10, P26, P40, and P137 is turned on by default (PU10, PU26, PU40, PU137 Reset Value is “1”), and 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	7	6	5	4	3	2	1	0	address	After reset	R/W
PU0	0	0	0	0	0	0	PU01	PU00	0x030	00H	R/W
PU1	PU17	PU16	PU15	PU14	PU13	PU12	PU11	PU10	0x031	01H	R/W
PU2	PU27	PU26	PU25	PU24	PU23	PU22	PU21	PU20	0x032	40H	R/W
PU3	0	0	0	0	0	0	PU31	PU30	0x033	00H	R/W
PU4	0	0	0	0	0	0	PU41	PU40	0x034	01H	R/W
PU5	0	0	0	0	0	0	PU51	PU50	0x035	00H	R/W
PU6	0	0	0	0	PU63	PU62	PU61[Note2]	PU60[Note2]	0x036	00H	R/W
PU7	0	0	PU75	PU74	PU73	PU72	PU71	PU70	0x037	00H	R/W
PU12	0	0	0	0	0	0	0	PU120	0x03C	00H	R/W
PU13	PU137	PU136	0	0	0	0	0	PU130	0x03D	80H	R/W
PU14	PU147	PU146	0	0	0	0	0	PU140	0x03E	00H	R/W

PUmn	Selection of internal pull-up resistors for Pmn pins (m=0~7, 12~14, n=0~7)
0	Internal pull-up resistors are not connected.
1	Connect an internal pull-up resistor.

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

Note 2. It is limited to CMS32L051xx-S series products. For products other than the CMS32L051xx-S series, ports P60 and P61 do not have pull-up and pull-down functions of their own and must be used with external pull-up resistors.

2.3.6 Pull-down resistor selection register (PDxx)

Selection register for internal pull-down resistors. The internal pull-down resistor can only be used in bits for bits specified by the drop-down resistor select register using the pin using the internal pull-down resistor and the POMmn bit is “0” and set to input mode (PMmn=1). For bits set to output mode, independent of the setting of the pull-down resistor selection register, no internal pull-down resistor is connected. The same is true when used as the output pin for the multiplexing function or when set to an analog function.

After the reset signal is generated, the values of these registers become “00H”.

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-up resistor selection register

symbol	7	6	5	4	3	2	1	0	address	After reset	R/W
PD0	0	0	0	0	0	0	PD01	PD00	0x040	00H	R/W
PD1	PD17	PD16	PD15	PD14	PD13	PD12	PD11	PD10	0x041	00 H	R/W
PD2	PD27	PD26	PD25	PD24	PD23	PD22	PD21	PD20	0x042	00H	R/W
PD3	0	0	0	0	0	0	PD31	PD30	0x043	00H	R/W
PD5	0	0	0	0	0	0	PD51	PD50	0x045	00H	R/W
PD6	0	0	0	0	PD63	PD62	PD61[note2]	PD60[note2]	0x046	00H	R/W
PD7	0	0	PD75	PD74	PD73	PD72	PD71	PD70	0x047	00H	R/W
PD12	0	0	0	0	0	0	0	PD120	0x04C	00H	R/W
PD13	0	PD136	0	0	0	0	0	PD130	0x04D	00H	R/W
PD14	PD147	PD146	0	0	0	0	0	PD140	0x04E	00H	R/W

PDmn	Selection of internal pull-down resistors for the Pmn pin (m=0~3, 5~7, 12~14, n=0~7).
0	Internal pull-down resistors are not connected.
1	Connect an internal pull-down resistor.

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

Note 2 It is limited to CMS32L051xx-S series products. For products other than the CMS32L051xx-S series, ports P60 and P61 do not have pull-up and pull-down functions of their own and must be used with external pull-up resistors.

2.3.7 Port output mode register (POMxx)

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

After the reset signal is generated, the values of these registers become “00H”.

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 set to N-channel open-drain output mode (POMmn=1), no internal pull-up resistor is connected.

Figure 2-7 Format of port output mode register

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

POMmn	Selection of output mode for the Pmn pin (m=0~3, 5~7, 12~14, n=0~7).
0	Usual output mode
1	N-channel open-drain output mode

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

2. It is limited to CMS32L051xx-S series products. For products other than the CMS32L051xx-S series, ports P60, P61 are dedicated N-channel open drain output ports and do not need to be configured with POM registers.

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

2.3.8 Port mode control register (PMCxx)

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

After the reset signal is generated, P10, P26, P130 are used as digital channels by default (PMC10, PMC26, PMC130 reset value is “0”,), and other ports are used as analog channels by default. P40, P41, P60, P61, P122~P124, P137 only has digital function and cannot be used as an analog channel.

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 2-8 Format of port mode control register

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

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

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

2. It is limited to CMS32L051xx-S series products. For products other than the CMS32L051xx-S series, ports P60, P61 can only be used as digital pins.

3. Ports P40, P41, P121~P124, P137 are not supported as analog channels.

4. P10, P26, P130 are used as digital channels by default after reset.

5. In addition to the above ports, other ports are used as analog channels by default after reset.

2.3.9 Port output multiplexing function configuration register (PxxCFG)

The port output multiplexing configuration register enables the output functions of a subset of peripheral modules to be mapped to any port. Reset value of the port output multiplexing function configuration register is “00H”, in which case the port is the default concurrent function and GPIO function.

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

Figure 2-9 List of port output multiplexing function configuration registers

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

P130CFG	0x048	R/W	00H
P136CFG	0x04e	R/W	00H
P137CFG	0x04f	R/W	00H
P140CFG	0x050	R/W	00H
P146CFG	0x056	R/W	00H
P147CFG	0x057	R/W	00H

Figure 2-10 Format of port output multiplexing function configuration register

symbol	7	6	5	4	3	2	1	0	address	after reset	R/W
PxxCFG	0	0	0	0	pxxcfg[3:0]				see figure above	00H	R/W

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

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

Table2-2 Configuration method for the concurrent output function

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

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

Configuration Instructions:

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

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

The Port Input Multiplexing Configuration Register enables the mapping of the input functions of peripheral modules to individual ports. Reset Value of the port input multiplexing function configuration register is “00H”. 20 dual input functions (TI10, TI11, TI12, TI13, INTP0, INTP1, INTP2, INTP3, SDI00, RXD0, SDA00, SCLKI00, SS00, SDI20, RXD2, IrRXD, SCLKI20, SDAA0, SCLA0, RXD1) maps to any port. Other than these 20 types of concurrent inputs can only be input from a fixed port.

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

Figure 2-11 List of port input multiplexing function configuration registers

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

Figure 2-12 Format of port input multiplex function configuration register

symbol	7	6	5	4	3	2	1	0	address	after reset	R/W
xxPCFG	0	0	xxpcfg[5:0]						See figure above	00H	R/W

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

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

Table 2-3 Configuration method of concurrent input function

Function name		Input/output	xxxPCFP[5:0]	PMCxx	PMxx	POMxx	Pxx	Remark
Simulation function		Input/output	x	1	x	×	×	All analog functions are directed to fixed ports only and are not configurable, Refer to the data sheets for each product family
GPIO		input	x	0	1	×	×	
Concurrent inputs that can be mapped arbitrarily	TI10	input	Configure the TI10PCFG	0	1	×	×	Can be mapped to any port
	TI11	input	Configure the TI11PCFG	0	1	×	×	Can be mapped to any port
	TI12	input	Configure the TI12PCFG	0	1	×	×	Can be mapped to any port
	TI13	input	Configure the TI13PCFG	0	1	×	×	Can be mapped to any port
	INTP0	input	Configure INTP0PCFG	0	1	×	×	By default, P136 is used and can be mapped to any port
	INTP1	input	Configure INTP1PCFG	0	1	×	×	P50 is used by default and can be mapped to any port
	INTP2	input	Configure INTP2PCFG	0	1	×	×	P51 is used by default and can be mapped to any port
	INTP3	input	Configure INTP3PCFG	0	1	×	×	P30 is used by default and can be mapped to any port
	SCLKI00	input	Configure SCLKI00PCFG	0	1	×	×	Can be mapped to any port
	SDI00/RxD0	input	Configure SDI00PCFG	0	1	×	×	Can be mapped to any port
	SDA00	bidirectional		0	0	1	1	Can be mapped to any port, except P121 to P124
	SS00	input	Configure SS00PCFG	0	1	×	×	Can be mapped to any port
	RxD1	input	Configure the RXD1PCFG	0	1	×	×	Can be mapped to any port
	SCLKI20	input	Configure SCLKI20PCFG	0	1	×	×	Can be mapped to any port
	SDI20/RxD2/IrRXD	input	Configure the SDI20PCFG	0	1	×	×	Can be mapped to any port
	SCLA0	bidirectional	Configure SCLA0PCFG	0	0	1*	0	It can be mapped to any port, except for P121~P124POMxx automatic setting 1, no software configuration is required
	SDAA0	bidirectional	Configure SDAA0PCFG	0	0	1*	0	It can be mapped to any port, except for P121~P124POMxx automatic setting 1, no software configuration is required
Concurrent input mapped to a fixed port	TI00	input	x	0	1	×	×	Fixed use P00
	TI01	input	x	0	1	×	×	Fixed use P16
	TI02	input	x	0	1	×	×	Fixed use P17
	TI03	input	x	0	1	×	×	Fixed use P31
	SCLKI01	input	x	0	1	×	×	Fixed use P75
	SDI01	input	x	0	1	×	×	Fixed use P74
	SDA01	bidirectional	x	0	0	1	1	Fixed use P74
	SCLKI11	input	x	0	1	×	×	Fixed use P10
	SDI11	input	x	0	1	×	×	Fixed use P11
	SDA11	bidirectional	x	0	0	1	1	Fixed use P11
	SDA20	bidirectional	x	0	0	1	1	Fixed use P14
	SCLKI21	input	x	0	1	×	×	Fixed use P70
	SDI21	input	x	0	1	×	×	Fixed use P71
	SDA21	bidirectional	x	0	0	1	1	Fixed use P71
	KR0	input	x	0	1	×	×	Fixed use P70
	KR1	input	x	0	1	×	×	Fixed use P71
	KR2	input	x	0	1	×	×	Fixed use P72
	KR3	input	x	0	1	×	×	Fixed use P73
	KR4	input	x	0	1	×	×	Fixed use P74
	KR5	input	x	0	1	×	×	Fixed use P75

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

Configuration Instructions:

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

2.3.11 SPI port multiplexing configuration register (SPIPCFG)

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

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

Figure 2-13 Format of port input multiplex configuration register

symbol	7	6	5	4	3	2	1	0	address	after reset	R/W
SPIPCFG	0	0	0	0	0	0	spipcfg[1:0]		0x07E	00H	R/W

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

Table 2-4 SPI communication port configuration method

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

2.4 Handling of unused pins

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

Table 2-5 Handling of each unused pin

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

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

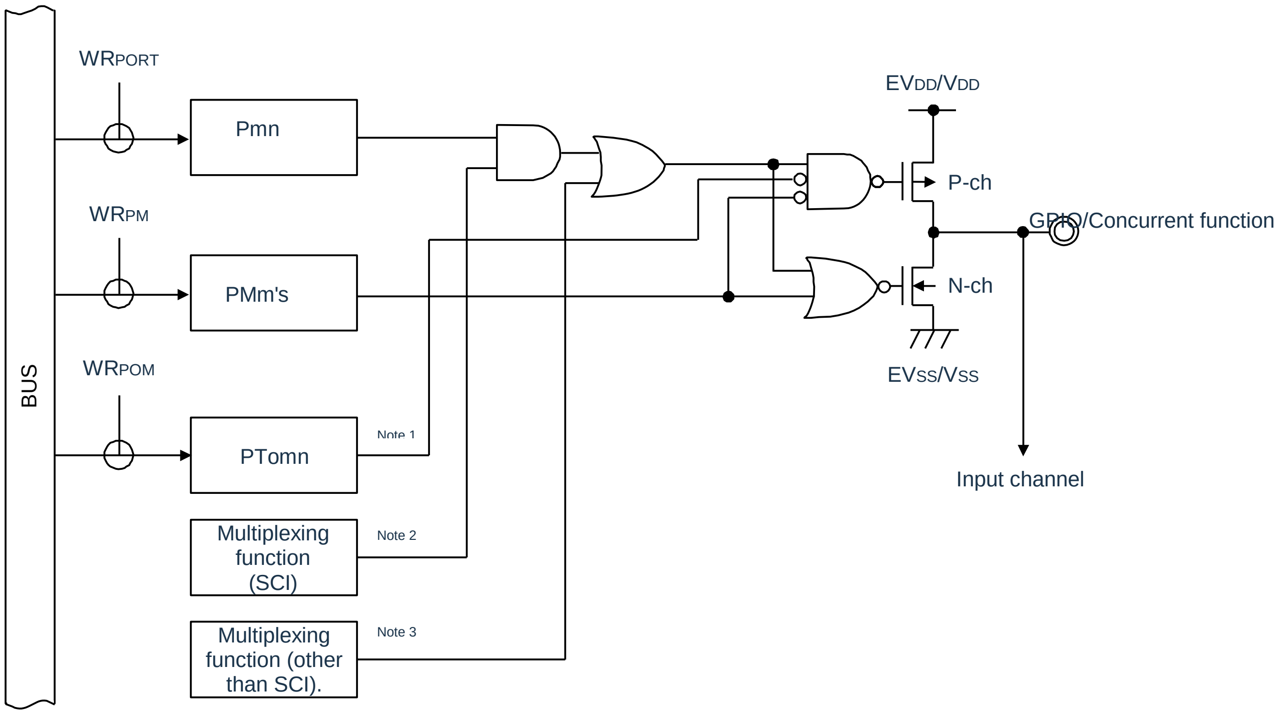
2.5 Register setting when using the multiplexed function

2.5.1 Basic idea when using the multiplexed output feature

First, for analog pins, the port mode control register (PMCxx) sets whether the pin 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. The output of the SCI function that is multiplexed with the output latch output of the port is input to the AND gate, the output of the AND gate is input to the OR gate, and the other input connections of the OR gate are multiplexed Output of SCI functions (output of timer, RTC, clock/buzzer, IICA, etc.). When such a pin is used as a port function or a multiplexed function, the unused multiplexing function cannot affect the output of the function to be used. The basic idea of setting at this point is shown in Table 2-6.

Figure 2-14 Basic structure of pin output



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

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

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

Table 2-6 Basic principal of configuration

The pin output function used	Output settings for unused multiplexing functions		
	Port functionality	The output function of SCI	Output function other than SCI
Port output function	—	High level output (1).	Low level output (0).
The output function of SCI	High(1)	—	Low level output (0).
Output function other than SCI	Low(0)	High level output (1).	Low level output (0) Note

Note: Because it is possible to multiplex output functions other than multiple SCIs with one pin, the output of the unused multiplexing function needs to be set to Low level (0). For details of the specific setting method, please refer to “2.5.2 Example of register settings using port functions and multiplexing functions“.

2.5.2 Example of register settings using port functions and multiplexing functions

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

Table 2-7 Example of register settings when using the P00 to P01 pin function

Pin name	Features used		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output Multiplexing Configuration Register)	xxPCFG (Input Multiplexing Configuration Register)	SPIPCFG	remark
	The feature name	Input/output								
P00	P00	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P00CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI11/VCIN10	Analog channel	1	×	×	×	×	×	×	
	TI00	input	0	1	×	×	×	×	×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P00CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P00CFG=4’h0	Configuring SDI00PCFG/ SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P01	P01	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P01CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI10/VCIN11	Analog channel	1	×	×	×	×	×	×	
	TO00	output	0	0	0	0	P01CFG=4’h0	×	×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P01CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P01CFG=4’h0	Configuring SDI00PCFG/ SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9

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

Pin name	Features used		PMC xx	PMx x	Px x	POM xx	PxxCFG (Output Multiplexing Configuration Register)	xxPCFG (Input Multiplexing Configuration Register)	SPIPCF G	remark
	The feature name	Input/output								
P10	P10	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P10CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI9	Analog channel	1	×	×	×	×	×	×	
	SCLK11	input	0	1	×	×	×	×	×	
		output	0	0	1	0	P10CFG=4’h0	×	×	
	epwmo00	output	0	0	0	0	P10CFG=4’h0	×	×	Please refer to 2.5.3
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0/1	Configure P10CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P10CFG=4’h0	Configuring SDI00PCFG/ SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P11	P11	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P11CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI8	Analog channel	1	×	×	×	×	×	×	
	SDI11	input	0	1	×	×	×	×	×	
	SDA11	bidirectional	0	0	1	1	P11CFG=4’h0	×	×	
	epwmo01	output	0	0	0	0	P11CFG=4’h0	×	×	Please refer to 2.5.3
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P11CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P11CFG=4’h0	Configuring SDI00PCFG/ SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P12	P12	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P12CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI13	Analog channel	1	×	×	×	×	×	×	
	SDO11	output	0	0	1	0	P12CFG=4’h0	×	×	
	epwmo02	output	0	0	0	0	P12CFG=4’h0	×	×	Please refer to 2.5.3
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P12CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P12CFG=4’h0	Configuring SDI00PCFG/ SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P13	P13	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P13CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI16	Analog channel	1	×	×	×	×	×	×	
	epwmo03	output	0	0	0	0	P13CFG=4’h0	×	×	Please refer to 2.5.3
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P13CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P13CFG=4’h0	Configuring SDI00PCFG/ SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9

Pin name	Features used		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output Multiplexing Configuration Register)	xxPCFG (Input Multiplexing Configuration Register)	SPIPCFG	remark
	The feature name	Input/output								
P14	P14	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P14CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI17	Analog channel	1	×	×	×	×	×	×	
	SDA20	bidirectional	0	0	1	1	P14CFG=4’h0	SDI20PCFG=6’h00	×	
	epwmo04	output	0	0	0	0	P14CFG=4’h0	×	×	Please refer to 2.5.3
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P14CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P14CFG=4’h0	Configuring SDI00PCFG/ SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P15	P15	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P15CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI18	Analog channel	1	×	×	×	×	×	×	
	CLKBUZ1	output	0	0	0	0	P15CFG=4’h0	×	×	
	epwmo05	output	0	0	0	0	P15CFG=4’h0	×	×	Please refer to 2.5.3
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P15CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P15CFG=4’h0	Configuring SDI00PCFG/ SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P16	P16	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P16CFG=4’h0	×	≠2’b01	
		N-channel open-drain output	0	0	0/1	1			≠2’b01	
	ANI19	Analog channel	1	×	×	×	×	×	×	
	TI01	input	0	1	×	×	×	×	×	
	TO01	output	0	0	0	0	P16CFG=4’h0	×	≠2’b01	
	SPI_MOSI	input	0	1	×	×	×	×	2’b01	Please refer to 2.3.11
		output	0	0	0	0	×	×		
	epwmo06	output	0	0	0	0	P16CFG=4’h0	×	≠2’b01	Please refer to 2.5.3
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P16CFG	×	≠2’b01	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P16CFG=4’h0	Configuring SDI00PCFG/ SDAA0PCFG/SCLA0PCFG	≠2’b01	Please refer to 2.3.9
P17	P17	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P17CFG=4’h0	×	≠2’b01	
		N-channel open-drain output	0	0	0/1	1			≠2’b01	
	ANI20	Analog channel	1	×	×	×	×	×	×	
	TI02	input	0	1	×	×	×	×	×	
	TO02	output	0	0	0	0	P17CFG=4’h0	×	≠2’b01	
	SPI_MISO	input	0	1	×	×	×	×	2’b01	Please refer to

		output	0	0	0	0	×	×		2.3.11
	epwmo07	output	0	0	0	0	P17CFG=4’h0	×	≠2’b01	Please refer to 2.5.3
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P17CFG	×	≠2’b01	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P17CFG=4’h0	Configuring SDI00PCFG/ SDAA0PCFG/SCLA0PCFG	≠2’b01	Please refer to 2.3.9

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

Pin name	Features used		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output Multiplexing Configuration Register)	xxPCFG (Input Multiplexing Configuration Register)	SPIPCFG	Remark
	The feature name	Input/output								
P20	P20	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P20CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI0/AVREFP/VCIN12	Analog channel	1	×	×	×	×	×	×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P20CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P20CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P21	P21	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P21CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI1/AVREFM/VCIN13	Analog channel	1	×	×	×	×	×	×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P21CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P21CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P22	P22	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P22CFG=4’h0	×	≠2’b11	
		N-channel open-drain output	0	0	0/1	1			≠2’b11	
	ANI2/AVREFM/VCIN0	Analog channel	1	×	×	×	×	×	×	
	SPI_MOSI	input	0	1	×	×	×	×	2’b11	Please refer to 2.3.11
		output	0	0	0	0	×	×		
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P22CFG	×	≠2’b11	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P22CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	≠2’b11	Please refer to 2.3.9
P23	P23	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P23CFG=4’h0	×	≠2’b11	
		N-channel open-drain output	0	0	0/1	1			≠2’b11	
	ANI3	Analog channel	1	×	×	×	×	×	×	
	SPI_MISO	input	0	1	×	×	×	×	2’b11	Please refer to 2.3.11
		output	0	0	0	0	×	×		
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P23CFG	×	≠2’b11	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P23CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	≠2’b11	Please refer to 2.3.9

Pin name	Features used		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output Multiplexing Configuration Register)	xxPCFG (Input Multiplexing Configuration Register)	SPIPCFG	remark
	The feature name	Input/output								
P24	P24	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P24CFG=4'h 0	×	≠2'b11	
		N-channel open-drain output	0	0	0/1	1			≠2'b11	
	ANI4	Analog channel	1	×	×	×	×	×	×	
	SPI_SCK	input	0	1	×	×	×	×	2'b11	Please refer to 2.3.11
		output	0	0	0	0	×	×		
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P24CFG	×	≠2'b11	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P24CFG=4'h 0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	≠2'b11	Please refer to 2.3.9
P25	P25	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P25CFG=4'h 0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI5	Analog channel	1	×	×	×	×	×	×	
	SPI_NSS	input	0	1	×	×	×	×	2'b11	Please refer to 2.3.11
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P25CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P25CFG=4'h 0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P26	P26	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P26CFG=4'h 0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI6	Analog channel	1	×	×	×	×	×	×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P26CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P26CFG=4'h 0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P27	P27	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P27CFG=4'h 0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI7	Analog channel	1	×	×	×	×	×	×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P27CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P27CFG=4'h 0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9

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

Pin name	Features used		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output Multiplexing Configuration Register)	xxPCFG (Input Multiplexing Configuration Register)	SPIPCFG	remark
	The feature name	Input/output								
P30	P30	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P30CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI21	Analog channel	1	×	×	×	×	×	×	
	INTP3	input	0	1	×	×	×	INTP3PCFG=6'h00	×	INTP3 can also be mapped to other ports, see 2.3.9
	RTC1HZ	output	0	0	0	0	P30CFG=4’h0	×	×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P30CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P30CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P31	P31	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P31CFG=4’h0	×	≠2’b10	
		N-channel open-drain output	0	0	0/1	1			≠2’b10	
	ANI22	Analog channel	1	×	×	×	×	×	×	
	TI03	input	0	1	×	×	×	×	×	
	TO03	output	0	0	0	0	P31CFG=4’h0	×	≠2’b10	
	SPI_SCK	input	0	1	×	×	×	×	2’b10	Please refer to 2.3.11
		output	0	0	0	0	×	×		
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P31CFG	×	≠2’b10	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P31CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	≠2’b10	Please refer to 2.3.9

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

Pin name	Features used		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output Multiplexing Configuration Register)	xxPCFG (Input Multiplexing Configuration Register)	SPIPCFG	remark
	The feature name	Input/output								
P40	P40	input	-	1	×	×	×	×	×	
		output	-	0	0/1	0	P40CFG=4’h0	×	×	
		N-channel open-drain output	-	0	0/1	1			×	
	Mappable concurrent inputs	input	-	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	-	0	0/1	0	Configure P40CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	-	0	0/1	1	P40CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P41	P41	input	-	1	×	×	×	×	×	
		output	-	0	0/1	0	P41CFG=4’h0	×	×	
		N-channel open-drain output	-	0	0/1	1			×	
	Mappable concurrent inputs	input	-	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	-	0	0/1	0	Configure P41CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	-	0	0/1	1	P41CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9

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

Pin name	Features used		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output Multiplexing Configuration Register)	xxPCFG (Input Multiplexing Configuration Register)	SPIPCFG	remark
	The feature name	Input/output								
P50	P50	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P50CFG=4'h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI23	Analog channel	1	×	×	×	×	×	×	
	INTP1	input	0	1	×	×	×	×	×	INTP1 can also be mapped to other ports, see 2.3.9
	VCOUT1	output	0	0	0	0	P50CFG=4'h0	×	×	
	SPI_NSS	input	0	1	×	×	×	×	2'b01	Please refer to 2.3.11
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure the P50CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P50CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P51	P51	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P51CFG=4'h0	×	≠2'b01	
		N-channel open-drain output	0	0	0/1	1			≠2'b01	
	ANI24	Analog channel	1	×	×	×	×	×	×	
	INTP2	input	0	1	×	×	×	×	×	INTP2 can also be mapped to other ports, see 2.3.9
	SPI_SCK	input	0	1	×	×	×	×	2'b01	Please refer to 2.3.11
		output	0	0	0	0	×	×		
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure the P51CFG	×	≠2'b01	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P51CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	≠2'b01	Please refer to 2.3.9

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

Pin name	Features used		PMC xx	PMxx	Pxx	POMxx	PxxCFG (Output Multiplexing Configuration Register)	xxPCFG (Input Multiplexing Configuration Register)	SPIPCFG	remark
	The feature name	Input/output								
P60	P60	input	-	1	×	×	×	×	×	
		N-channel open-drain output	-	0	0/1	-	P60CFG=4’h0	×	×	
	Mappable concurrent inputs	input	-	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	-	0	0/1	1	P60CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P61	P61	input	-	1	×	×	×	×	×	
		N-channel open-drain output	-	0	0/1	1	P61CFG=4’h0	×	×	
	Mappable concurrent inputs	input	-	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	-	0	0/1	1	P61CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P62	P62	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P62CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI27	Analog channel	1	×	×	×	×	×	×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P62CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P62CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P63	P63	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P63CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI28	Analog channel	1	×	×	×	×	×	×	
	SPI_NSS	input	0	1	×	×	×	×	2’b10	Please refer to 2.3.11
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P63CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P63CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9

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

Pin name	Features used		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output Multiplexing Configuration Register)	xxPCFG (Input Multiplexing Configuration Register)	SPIPCFG	remark
	The feature name	Input/output								
P70	P70	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P70CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI29	Analog channel	1	×	×	×	×	×	×	
	KR0	input	0	1	×	×	×	×	×	
	SCLK21	input	0	1	×	×	×	×	×	
		output	0	0	1	0	P70CFG=4’h0	×	×	
	SCL21	output	0	0	1	0	P70CFG=4’h0	×	×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure the P70CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P70CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P71	P71	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P71CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI30	Analog channel	1	×	×	×	×	×	×	
	KR1	input	0	1	×	×	×	×	×	
	SDI21	input	0	1	×	×	×	×	×	
	SDA21	bidirectional	0	0	1	1	P71CFG=4’h0	×	×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure the P71CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P71CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P72	P72	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P72CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI31	Analog channel	1	×	×	×	×	×	×	
	KR2	input	0	1	×	×	×	×	×	
	SDO21	output	0	1	×	×	×	×	×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P72CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P72CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9

Pin name	Features used		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output Multiplexing Configuration Register)	xxPCFG (Input Multiplexing Configuration Register)	SPIPCFG	remark
	The feature name	Input/output								
P73	P73	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P73CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI32	Analog channel	1	×	×	×	×	×	×	
	KR3	input	0	1	×	×	×	×	×	
	SDO01	output	0	1	×	×	×	×	×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure the P73CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P73CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG /SCLA0PCFG	×	Please refer to 2.3.9
P74	P74	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P74CFG=4’h0	×	≠2’b10	
		N-channel open-drain output	0	0	0/1	1			≠2’b10	
	ANI33	Analog channel	1	×	×	×	×	×	×	
	KR4	input	0	1	×	×	×	×	×	
	SDI01	input	0	1	×	×	×	×	×	
	SDA01	bidirectional	0	0	1	1	P74CFG=4’h0	×	≠2’b10	
	SPI_MOSI	input	0	1	×	×	×	×	2’b10	Please refer to 2.3.11
		output	0	0	0	0	×	×		
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure P74CFG	×	≠2’b10	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P74CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG /SCLA0PCFG	≠2’b10	Please refer to 2.3.9
P75	P75	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P75CFG=4’h0	×	≠2’b10	
		N-channel open-drain output	0	0	0/1	1			≠2’b10	
	ANI34	Analog channel	1	×	×	×	×	×	×	
	KR5	input	0	1	×	×	×	×	×	
	SCLK01	input	0	1	×	×	×	×	×	
		output	0	0	1	0	P75CFG=4’h0	×	≠2’b10	
	SCL01	output	0	0	1	0	P75CFG=4’h0	×	≠2’b10	
	SPI_MISO	input	0	1	×	×	×	×	2’b10	Please refer to 2.3.11
		output	0	0	0	0	×	×		
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure the P75CFG	×	≠2’b10	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P75CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG /SCLA0PCFG	≠2’b10	Please refer to 2.3.9

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

Pin name	Features used		PMC xx	PM xx	Pxx	POMxx	PxxCFG (Output Multiplexing Configuration Register)	xxPCFG (Input Multiplexing Configuration Register)	SPIPCFG	remark
	The feature name	Input/output								
P120	P120	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P120CFG=4'h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI14	Analog channel	1	×	×	×	×	×	×	
	VCOUT0	output	0	0	0	0	P120CFG=4'h0	×	×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure the P120CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P120CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P121	P121	input	-	1	×	-	×	×	×	
		output	-	0	0/1	-	P121CFG=4'h0	×	×	
		N-channel open-drain output	-	0	0/1	-			×	
	X1	-	-	×	×	-	×	×	×	EXCLK=0, OSCSEL=1
	Mappable concurrent inputs	input	-	1	×	-	×	Configure xxPCFG	×	For EXCLK=0 and OSCSEL=0, please refer to 2.3.9
	Mappable concurrent output	output	-	0	0/1	-	Configure the P121CFG	×	×	For EXCLK=0 and OSCSEL=0, please refer to 2.3.10
P122	P122	input	-	1	×	-	×	×	×	
		output	-	0	0/1	-	P122CFG=4'h0	×	×	
		N-channel open-drain output	-	0	0/1	-			×	
	X2	-	-	×	×	-	×	×	×	EXCLK=0, OSCSEL=1
	EXCLK	input	-	×	×	-	×	×	×	EXCLK=1, OSCSEL=1
	Mappable concurrent inputs	input	-	1	×	-	×	Configure xxPCFG	×	For EXCLK=0 and OSCSEL=0, please refer to 2.3.9
	Mappable concurrent output	output	-	0	0/1	-	Configure the P122CFG	×	×	For EXCLK=0 and OSCSEL=0, please refer to 2.3.10
P123	P123	input	-	1	×	-	×	×	×	
		output	-	0	0/1	-	P123CFG=4'h0	×	×	
		N-channel open-drain output	-	0	0/1	-			×	
	XT1	-	-	×	×	-	×	×	×	EXCLKS=0, OSCSELS=1
	Mappable concurrent inputs	input	-	1	×	-	×	Configure xxPCFG	×	For EXCLKS=0 and OSCSELS=0, refer to 2.3.9
	Mappable concurrent output	output	-	0	0/1	-	Configure the P123CFG	×	×	Please refer to 2.3.10 for EXCLKS=0 and OSCSELS=0
P124	P124	input	-	1	×	-	×	×	×	
		output	-	0	0/1	-	P124CFG=4'h0	×	×	
		N-channel	-	0	0/1	-			×	

		open-drain output								
	XT2	-	-	×	×	-	×	×	×	EXCLKS=0, OSCSELS=1
	EXCLKS	input	-	×	×	-	×	×	×	EXCLKS=1, OSCSELS=1
	Mappable concurrent inputs	input	-	1	×	-	×	Configure xxPCFG	×	For EXCLKS=0 and OSCSELS=0, refer to 2.3.9
	Mappable concurrent output	output	-	0	0/1	-	Configure the P124CFG	×	×	Please refer to 2.3.10 for EXCLKS=0 and OSCSELS=0

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

Pin name	Features used		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output Multiplexing Configuration Register)	xxPCFG (Input Multiplexing Configuration Register)	SPIPCFG	remark
	The feature name	Input/output								
P130	P130	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P130CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI35	Analog channel	1	×	×	×	×	×	×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure the P130CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P130CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P136	P136	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P136CFG=4’h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI36	Analog channel	1	×	×	×	×	×	×	
	INTP0	input	0	1	×	×	×	×	×	INTP0 can also be mapped to other ports, see 2.3.9
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure the P136CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P136CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P137	P137	input	-	1	×	×	×	×	×	
		output	-	0	0/1	0	P137CFG=4’h0	×	×	
		N-channel open-drain output	-	0	0/1	1			×	
	Mappable concurrent inputs	input	-	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	-	0	0/1	0	Configure the P137CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	-	0	0/1	1	P137CFG=4’h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9

Table 2-17 Example of register configuration of using P140, P146, P147 pin function

Pin name	Features used		PMCxx	PMxx	Pxx	POMxx	PxxCFG (Output Multiplexing Configuration Register)	xxPCFG (Input Multiplexing Configuration Register)	SPIPCFG	remark
	The feature name	Input/output								
P140	P140	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P140CFG=4'h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure the P140CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P140CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P146	P146	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P146CFG=4'h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI15	Analog channel	1	×	×	×	×	×	×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure the P146CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P146CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9
P147	P147	input	0	1	×	×	×	×	×	
		output	0	0	0/1	0	P147CFG=4'h0	×	×	
		N-channel open-drain output	0	0	0/1	1			×	
	ANI12/IVREF0	Analog channel	1	×	×	×	×	×	×	
	Mappable concurrent inputs	input	0	1	×	×	×	Configure xxPCFG	×	Please refer to 2.3.9
	Mappable concurrent output	output	0	0	0/1	0	Configure the P147CFG	×	×	Please refer to 2.3.10
	Mappable bidirectional communication (SDA00/SDAA0/SCLA0)	bidirectional	0	0	0/1	1	P147CFG=4'h0	Configure SDI00PCFG/SDAA0PCFG/SCLA0PCFG	×	Please refer to 2.3.9

2.5.3 EPWM port configuration method

When using the EPWM output control circuit function, the EPWM output pin is fixed and mapped to P10~P17 as follows:

The port name	function	Input/output	PxxCFP	PMCxx	PMxx	POMxx	Pxx	remark
P10	epwmo00	output	P10CFG=4'h0	0	0	0	0	
P11	epwmo01	output	P11CFG=4'h0	0	0	0	0	
P12	epwmo02	output	P12CFG=4'h0	0	0	0	0	
P13	epwmo03	output	P13CFG=4'h0	0	0	0	0	
P14	epwmo04	output	P14CFG=4'h0	0	0	0	0	
P15	epwmo05	output	P15CFG=4'h0	0	0	0	0	When epwmo05 output, keep CLKBUZ1 output “0”
P16	epwmo06	output	P16CFG=4'h0	0	0	0	0	
P17	epwmo07	output	P17CFG=4'h0	0	0	0	0	When epwmo07 output, please keep TO02 output “0”

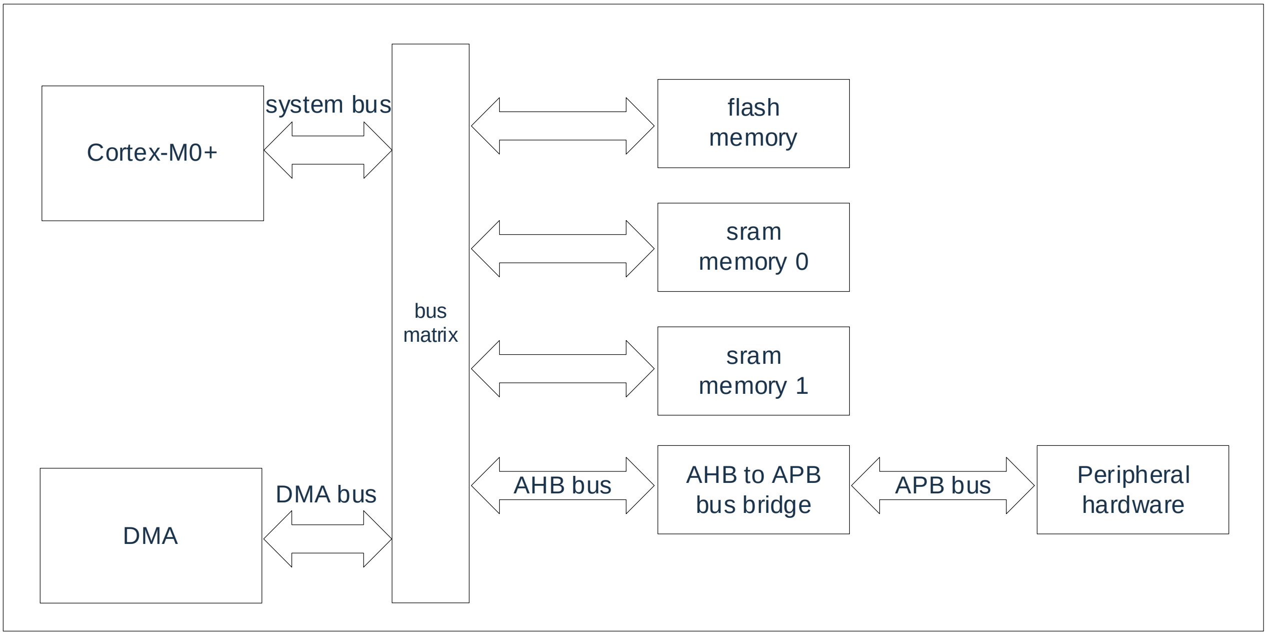
Chapter 3 System Structure

3.1 Overview

This product system consists of the following parts:

- 2 AHB bus master:
  - Cortex-M0+
  - Enhanced DMA
- 4 AHB bus slavas:
  - FLASH memory
  - SRAM memory 0
  - SRAM memory 1
  - AHB to APB Bridge, containing all APB interface peripherals

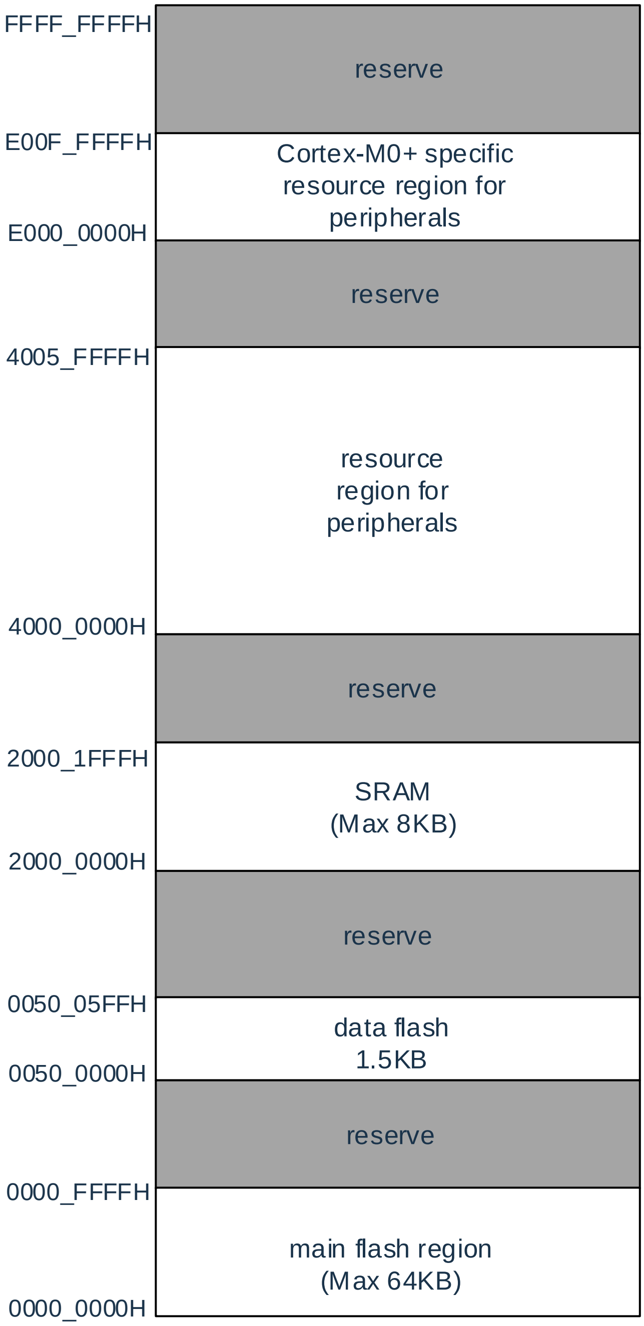
Figure 3-1 Schematic diagram of the system structure



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

3.2 System address partition

Figure 3-2 Schematic diagram of address area partition



Peripheral address assignment

Table 3-1 Start addresses of the peripheral's register group

Start address	Peripheral	Remark
0x4000_0000 - 0x4000_4FFF	Reserved	
0x4000_5000 - 0x4000_5FFF	DMA	
0x4000_6000 - 0x4000_6FFF	Interrupt control	
0x4000_7000 - 0x4001_8FFF	Reserved	
0x4001_9000 - 0x4001_9FFF	Reserved	
0x4001_A000 - 0x4001_FFFF	Reserved	
0x4002_0000 - 0x4002_03FF	FLASH control	
0x4002_0400 - 0x4002_0FFF	Clock control	
0x4002_1000 - 0x4002_1001	Watchdog timer	
0x4002_1002 - 0x4002_1800	Reserved	
0x4002_1800 - 0x4002_1BFF	High-speed CRC	See Chapter 26: Safety Functions
0x4002_1C00 - 0x4002_1FFF	Clock control	
0x4002_2000 - 0x4003_FFFF	Reserved	
0x4004_0000 - 0x4004_0FFF	GPIO	
0x4004_1100 - 0x4004_19FF	Serial communication unit	
0x4004_1A00 - 0x4004_1CFF	Serial interface IICA	
0x4004_1D00 - 0x4004_1FFF	Timer array 0	
0x4004_2000 - 0x4004_21FF	Timer array 1	
0x4004_2200 - ff 0x4004_23	Reserved	
0x4004_2FF 400 - ff 0x4004_27	SPI	
0x4004_2 FF 800 - 0x4004_31	Reserved	
0x4004_3200 - 0x4004_32FF	Generic CRC	See Chapter 26: Safety Functions
0x4004_3300 - 0x4004_33FF	Reserved	
0x4004_3400 - 0x4004_37FF	Linkage controller	
0x4004_3C00 - 0x4004_3FFF	Reserved	
0x4004_4000 - ff 0x4004_43	IrDA	
0x4004_4FF 400 - FF 0x4004_47	EPWM	
0x4004_4800 - 0x4004_4EFF	Reserved	
0x4004_4F00 - 0x4004_4FFF	Real-time clock	
0x4004_5000 - 0x4004_53FF	AD converter	
0x4004_5400 - 0x4004_5AFF	Reserved	
0x4004_5B00 - 0x4004_5BFF	External interrupt control	
0x4008_0000 - 0x4008_01FF	Reserved	
0x4008_0200 - 0xDFFF_FFFF	Reserved	

Chapter 4 Clock Generation Circuit

The presence or absence of resonator connection pins for master system clock/external clock input pins, resonator connection pins for subsystem clocks/external clock input pins varies depending on the product.

4.1 Function of the clock generation circuit

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

(1) Main system clock

① X1 oscillation circuit

The clock from f = 1 to 20MHz can be oscillated by connecting a resonator to the X1 and X2 pins, and it can be used to enter deep sleep mode or set MSTOP the bit (bit7 of the clock operating state control register (CSC)) stops the oscillation.

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

Can be used from fHOCO=64MHz, 48MHz, 32MHz, 24MHz, via option byte (000C2H). 16MHz, 12MHz, 8MHz, 6MHz, 4MHz, 3MHz and 2MHz (TYP) selects the frequency for oscillation. After the reset is released, the CPU must start running with this high-speed internal oscillator clock. Oscillation can be stopped by entering deep sleep mode or setting the HIOSTOP bit (bit0 of the CSC register). The frequency of the option byte setting can be changed through the frequency selection register (HOCODIV) of the high-speed internal oscillator. For frequency settings, refer to “Figure 4-10 Format of “.

In addition, an external master system clock (fEX = 1 to 20MHz) can be provided by the EXCLK/X2/P122 pin and the input to the external master system clock can be disabled by entering deep sleep mode or by setting the MSTOP bit.

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

(2) Subsystem clock

① XT1 oscillation circuit

The clock of fXT=32.768kHz can be oscillated by connecting a 32.768kHz resonator to the XT1 pin and XT2 pin, and the XTSTOP bit (clock operating status control register (CSC) bit6) to stop the oscillation.

In addition, an external subsystem clock (f_{EXS}=32.768kHz) can be provided by EXCLKS/XT2/P1 pin 24, and the input of the external subsystem clock can be invalidated by setting the XTSTOP bit.

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

Clock oscillation of f_{IL}=15kHz can be made.

You can use the low-speed internal oscillator clock as the system clock.

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

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

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

Note f_X: X1 clock oscillation frequency

f_{HOCO}: Clock frequency of a high-speed internal oscillator).

f_{ISH}: The clock frequency of the high-speed internal oscillator

f_{EX}: External master system clock frequency

f_{XT}: XT1 clock oscillation frequency

f_{EXS}: External subsystem clock frequency

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

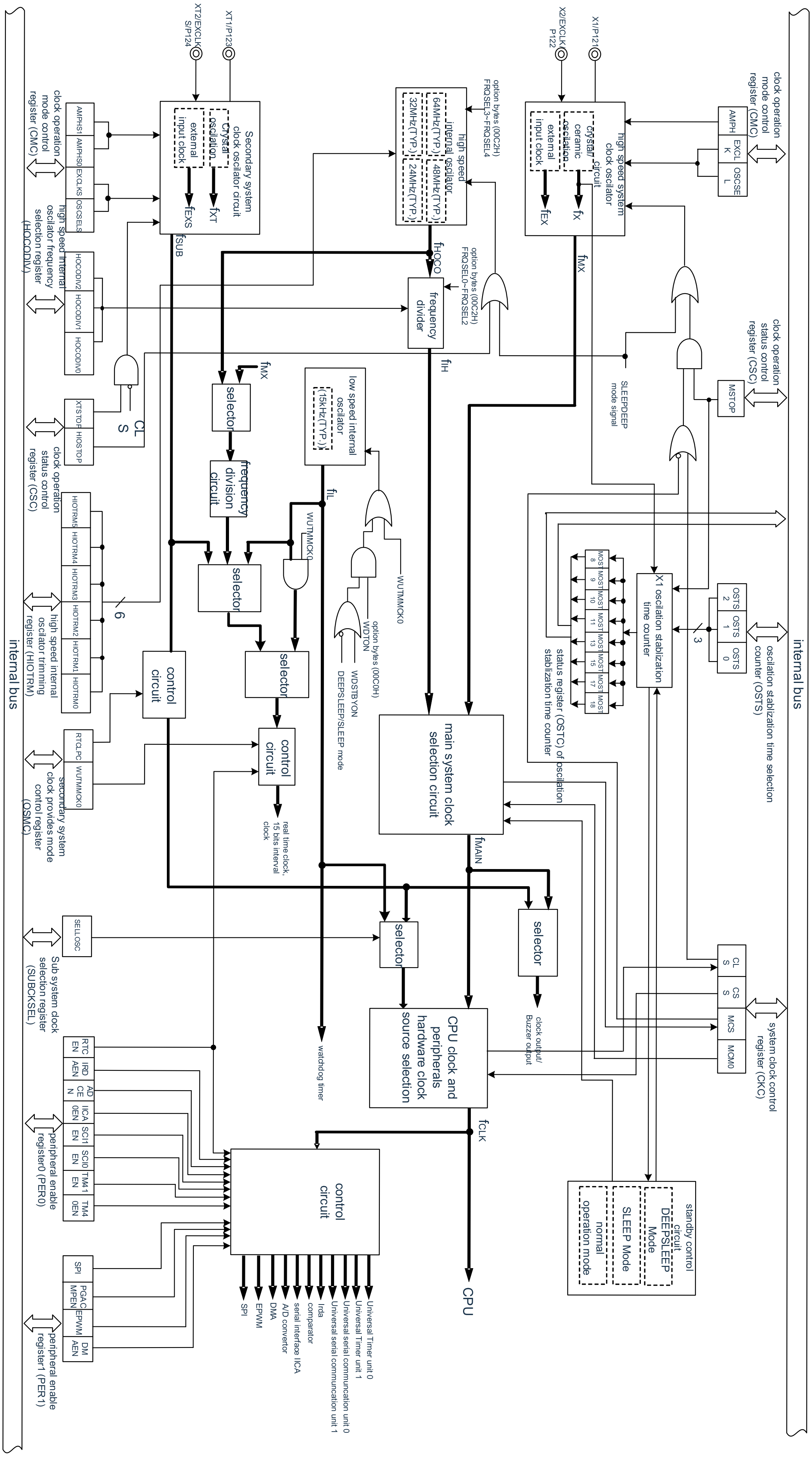
4.2 Structure of clock generation circuit

The clock generation circuit consists of the following hardware.

Table 4-1 Structure of the clock generation circuit

item	structure
Control registers	Clock operation mode control register (CMC). System clock control registers (CKCs). Clock operating status control registers (CSCs). Oscillate the status register (OSTC) of the settling time counter Oscillation settling time selection register (OSTS). Peripheral enable registers 0, 1 (PE R 0, PE R 1). Subsystem clock provides a mode control register (OSMC). Frequency selection register (HOCODIV) for a high-speed internal oscillator The trimming register (HIOTRM) of the high-speed internal oscillator Subsystem Clock Selection Register (SUBCKSEL)
Oscillation circuit	X1 oscillation circuit XT1 oscillation circuit High-speed internal oscillator Low-speed internal oscillator

Figure 4-1 Block diagram of clock generation circuit


internal bus
clock operation mode control register (CMC)
AMPH
EXCLK
OSCSEL
X1/P121
X2/EXCLK/P122
high speed system clock oscillator
crystal ceramic oscillation
external input clock
fX
fEX
fMX
option bytes (000C2H) FRQSEL3~FRQSEL4
high speed internal oscillator
64MHz(TYP.)
48MHz(TYP.)
32MHz(TYP.)
24MHz(TYP.)
fHOCO
option bytes (000C2H) FRQSEL0~FRQSEL2
frequency divider
fIH
clock operation status control register (CSC)
MSTOP
SLEEPDEEP mode signal
XT1/P123
XT2/EXCLKS/P124
Secondary system clock oscillator circuit
Crystal oscillation
external input clock
fXT
fEXS
fSUB
AMPHS1
AMPHS0
EXCLKS
OSCSELS
clock operation mode control register (CMC)
HOCODIV2
HOCODIV1
HOCODIV0
high speed internal oscillator frequency selection register (HOCODIV)
XTSTOP
HIOSTOP
CLS
clock operation status control register (CSC)
HIOTRM5
HIOTRM4
HIOTRM3
HIOTRM2
HIOTRM1
HIOTRM0
6
high speed internal oscillator trimming register (HIOTRM)
RTCLPC
WUTMMCK0
secondary system clock provides mode control register (OSMC)
SELLOSC
Sub system clock selection register (SUBCKSEL)
fMX
selector
frequency division circuit
low speed internal oscillator
(15kHz(TYP.))
fIL
WUTMMCK0
WUTMMCK0
option bytes (000C0H) WDTON
WDSTBYON
DEEPSLEEP/SLEEP mode
selector
selector
control circuit
control circuit
real time clock, 15 bits interval clock
main system clock selection circuit
fMAIN
selector
selector
clock output/ Buzzer output
CPU clock and peripherals hardware clock source selection
watchdog timer
fCLK
CPU
X1 oscillation stabilization time counter
MOST 8
MOST 9
MOST 10
MOST 11
MOST 13
MOST 15
MOST 17
MOST 18
status register (OSTC) of oscillation stabilization time counter
OSTS2
OSTS1
OSTS0
3
oscillation stabilization time selection counter (OSTS)
CLS
CSS
MCS
MCM0
system clock control register (CKC)
standby control circuit
DEEPSLEEP Mode
SLEEP Mode
normal operation mode
RTCEN
IRDAEN
ADCEN
ICA0EN
SCI1EN
SCI0EN
TM41EN
TM40EN
peripheral enable register0 (PER0)
SPI
PGACMPEN
EPWM
DMAEN
peripheral enable register1 (PER1)
control circuit
Universal Timer unit 0
Universal Timer unit 1
Universal serial communication unit 0
Universal serial communication unit 1
Irda
comparator
serial interface IICA
A/D convertor
DMA
EPWM
SPI

remark	f_{X}	: X1 clock oscillation frequency
	f_{HOCO}	: The clock frequency of the high-speed internal oscillator
	f_{IH}	: The clock frequency of the high-speed internal oscillator
	f_{EX}	: The external master system clock frequency
	f_{MX}	: High-speed system clock frequency
	f_{MAIN}	: Master system clock frequency
	f_{XT}	: XT1 clock oscillation frequency
	f_{EXS}	: External subsystem clock frequency
	f_{SUB}	: The secondary system clock frequency
	f_{CLK}	: The clock frequency of the CPU/peripheral hardware
	f_{IL}	: The clock frequency of the 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 registers (CKC)
- Clock operation status control register (CSC)
- Status register of the oscillation stabilization time counter (OSTC)
- Oscillation stabilization time selection register (OSTS)
- Peripheral enable registers 0, 1 (PER 0, PER1)
- Subsystem clock supply mode control register (OSMC)
- High-speed internal oscillator frequency selection register (HOCODIV)
- High-speed internal oscillator trim register (HIOTRM)
- Subsystem clock selection register (SUBCKSEL)

Note: Assigned registers and bits vary from product to product. You must set an initial value for unassigned bits.

4.3.1 Clock operation mode control register (CMC).

This is the register that sets the operating mode of the X1/P 121, X2/EXCLK/P 122, XT1/P123, XT2/EXCLKS/P1 24 pins and selects the oscillation circuit gain.

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

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

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

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

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

EXCLK	OSCSEL	High speed system clock pin operation mode	X1/P121 pins	X2/EXCLK/P122 pins
0	0	Port mode	Input/output ports	
0	1	X1 oscillation mode	Connect a crystal or ceramic resonator.	
1	0	Port mode	Input/output ports	
1	1	External clock input mode	Input/output ports	External clock input

EXCLKS	OSCSELS	Subsystem clock pin operating mode	XT1/P123 pins	XT2/EXCLKS/P124 Pins
0	0	Port mode	Input/output ports	
0	1	XT1 oscillation mode	Connect a crystal resonator.	
1	0	Port mode	Input/output ports	
1	1	External clock input mode	Input/output ports	External clock input

AMPHS1	AMPHS0	Oscillation mode selection for XT1 oscillation circuitry
0	0	Low power oscillation (default)
0	1	Usual oscillations
1	0	Ultra-low power oscillation
1	1	Disable the setting.

AMPH	Control of the oscillation frequency of the X1 clock
0	1MHz≤f_X≤10MHz
1	10MHz＜f_X≤20MHz

Note: The EXCLKS bit, OSCSELS bit, AMPHS 1 bit, and AMPHS0 bits are only initialized on power-on reset and remain unchanged on other resets.

Note 1. After the reset is released, the CMC register can only be written once via the 8-bit memory operation command. When using the CMC register at the initial value (“00H”), in order to prevent malfunction when the program is out of control (it cannot be recovered if a value other than “00H” is miswritten), the CMC register must be set after the reset is released “00H”.

2. The CMC register must be set after the reset is released and before the X1 or XT1 oscillation is started by setting the clock operating state control register (CSC).

3. When the X1 clock oscillation frequency exceeds 10MHz, the AMPH position must be “1”.

4. The AMPH bit, the AMPHS1 bit and the AMPHS0 bit must be set after the reset has been released and with fIH selected as the state of f_{CLK} (before switching f_{CLK} to f_{MX} or f_{SUB}).

5. The oscillation settling time of f_{XT} must be counted by software.

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

Note f_X: X1 clock oscillation frequency

4.3.2 System clock control register (CKC)

This is the register that selects the CPU/peripheral hardware clock and the main system clock.
The CKC register is set via an 8-bit memory operation instruction.
After the reset signal is generated, the value of this register becomes “00H”.

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

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

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

CLS	Status of the CPU/peripheral hardware clock (f_{CLK})
0	Master system clock (f_{MAIN}).
1	Subsystem clock (f_{SUB}).

CSS Note2	CPU/peripheral hardware clock (f_{CLK}) selection
0	Master system clock (f_{MAIN}).
1	Subsystem clock (f_{SUB}).

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

MCM0 Note 2	Operational control of the main system clock (f_{MAIN}).
0	The high-speed internal oscillator clock (f_{IG}) is selected as the main system clock (f_{MAIN}).
1	Select the high-speed system clock (f_{MX}) as the main system clock (f_{MAIN}).

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

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

Note $f_{HOCO:}$ The clock frequency of a high-speed internal oscillator

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

$f_{MX:}$ High-speed system clock frequency

$f_{MAIN:}$ Main system clock frequency

$f_{SUB:}$ Subsystem clock frequency

Note 1 You must set bit0~3 to “0”.

2. Provide a clock for CSS bit settings for the CPU and peripheral hardware. If you change the CPU clock, you change the clocks of the peripheral hardware at the same time (except for the real-time clock, the 15-bit interval timer, the clock output/buzzer output, and the watchdog timer). Therefore, if you want to change the clock of the CPU/peripheral hardware, you must stop each peripheral function.

3. If the subsystem clock is used as the peripheral hardware clock, the operation of the A/D converter and IICA cannot be guaranteed. For the operating characteristics of peripheral hardware, please refer to the chapters and data sheets of each peripheral hardware.

4.3.3 Clock operation status control register (CSC)

This is a register that controls the operation of the high-speed system clock, the high-speed internal oscillator clock, and the secondary system clock (except for the low-speed internal oscillator clock). Set the CSC register via 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 operating status control register

Address: 40020401H After reset: C0H R/W

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

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

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

HIOSTOP	Operation control of the high-speed internal oscillator clock
0	High-speed internal oscillator runs
1	High-speed internal oscillator stops

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

2. After the reset is released and before the MSTOP position is “0”, the oscillation settling time selection register (OSTS) must be set. However, when using the OSTS register with the initial value, there is no need to set the OSTS register.

3. To start the X1 oscillation by setting the MSTOP bit, the oscillation settling time of the X1 clock must be confirmed through the state register (OSTC) of the oscillation settling time counter.

4. To start the XT1 oscillation by setting the XSTOP bit, you must wait through the software for the oscillation settling time required by the subsystem clock.

5. The clock selected as the CPU/Peripheral Hardware Clock (fCLK) cannot be stopped through the CSC registers.

6. For the register flag setting and pre-stop conditions for stopping clock oscillation (external clock input is invalid), refer to Table 4-2.

Table 4-2 Clock stop method

clock	Conditions before the clock stops (invalid external clock input)	Set the flag of the CSC register
X1 clock	The CPU/peripheral hardware clock operates on a clock other than the high-speed system clock. (CLS=0 and MCS=0, or CLS=1).	MSTOP=1
External master system clock		
XT1 clock	The CPU/peripheral hardware clock operates on a clock other than the secondary system clock. (CLS=0)	XTSTOP=1
External subsystem clock		
High-speed internal oscillator clock	The CPU/peripheral hardware clock runs on a clock other than the high-speed internal oscillator clock. (CLS=0 and MCS=1, or CLS=1).	HIOSTOP=1

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

This is the register that indicates the counting status of the oscillation settling time counter of the X1 clock. It is possible to confirm the oscillation settling time of the X1 clock in the following cases.

- When the CPU clock is a high-speed internal oscillator clock or a subsystem clock and the oscillation of the X1 clock begins
- The OST register can be read via 8-bit memory operation instructions when the CPU clock is a high-speed internal oscillator clock and the sleep mode is released after moving to deep sleep mode in the state of X1 clock oscillation.

The value of this register becomes “00H” by resetting the signal generation, entering deep sleep mode, or the MSTOP bit (bit7 of the clock operating state control register (CSC)) is “1”.

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

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

Figure 4-5 Format of status register of the oscillation stabilization time counter (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

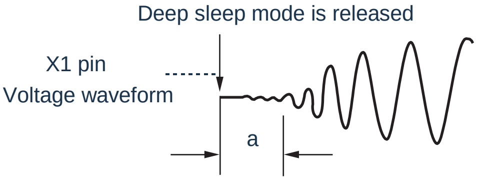
MOST 8	MOST 9	MOST 10	MOST 11	MOST 13	MOST 15	MOST 17	MOST 18	Oscillation steady-time state		
									f_X=10MHz	f_X=20MHz
0	0	0	0	0	0	0	0	Less than $2^8/f_X$	Less than 25.6us	Less than 12.8us
1	0	0	0	0	0	0	0	At least $2^8/f_X$	At least 25.6u s	At least 12.8u s
1	1	0	0	0	0	0	0	At least $2^9/f_X$	At least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

Note 1 After the above time, you start with MOST8 bits and change to “1” and remain in the state of “1”.

2. The oscillation stabilization time counter only counts during the oscillation stabilization time set by the oscillation stabilization time selection register (OSTS). In the following cases, the oscillation settling time of the OSTS register must be set to be greater than the count value confirmed by the OSTC register.

- When the CPU clock is a high-speed internal oscillator clock or a subsystem clock and oscillation of the X1 clock is to begin
- When the CPU clock is a high-speed internal oscillator clock and is released from deep sleep mode after being shifted to deep sleep mode with the X1 clock oscillating (so it must be noted that the OSTC register after being released from deep sleep mode only sets the state for the oscillation settling time set in the OSTS register).

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



Note f_X: X1 clock oscillation frequency

4.3.5 Oscillation stabilization time selection register (OSTS)

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

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

If you switch the CPU clock from the high-speed internal oscillator clock or the subsystem clock to the X1 clock, or if the CPU clock is a high-speed internal oscillator clock and is decommissioned after transferring to deep sleep mode while the X1 clock oscillates, the oscillation settling time must be confirmed by the oscillation settling time counter's status register (OSTC).

The time set by the OSTS register can be confirmed through the OSTC register.

Set the OSTS registers via 8-bit memory operation instructions. After the reset signal is generated, the value of this register changes to “07H”.

Figure 4-6 Format of oscillation stabilization time selection 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

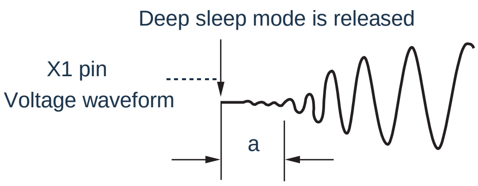
OSTS2	OSTS1	OSTS0	Selection of oscillation stabilization time		
				f_X=10MHz	f_X=20MHz
0	0	0	28/fX	25.6us	12.8us
0	0	1	29/fX	51.2us	25.6us
0	1	0	210/fX	102us	51.2us
0	1	1	211/fX	204us	102us
1	0	0	213/fX	819us	409us
1	0	1	215/fX	3.27ms	1.63ms
1	1	0	217/fX	13.1ms	6.55ms
1	1	1	218/fX	26.2ms	13.1ms

Note 1 To change the settings of the OSTS register, you must change it before placing the MSTOP position of the clock running state control register (CSC) “0”.

2. The oscillation settling time counter only counts during the oscillation stabilization time set by the OSTS register. In the following cases, the oscillation settling time of the OSTS register must be set to be greater than the count value confirmed by the OSTC register after the oscillation has begun.

- When the CPU clock is a high-speed internal oscillator clock or a subsystem clock and oscillation of the X1 clock is to begin
- When the CPU clock is a high-speed internal oscillator clock and deep sleep mode is removed after moving to deep sleep mode in the state of X1 clock oscillation (so it must be noted that the OSTS register after deep sleep mode is dismissed, only OSTS is set.) The state of the oscillation set by the register within the stable time).

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

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

This is the register that sets the clock that is allowed or disallowed for each peripheral hardware. Reduce power consumption and noise by stopping clocking unused hardware.

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

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

The PER0 register and the PER1 register are set via 8-bit memory operation instructions.

After the reset signal is generated, the values of these registers become “00H”.

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

Address: 40020 420H After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
PER0	RTCEN Note	IRDAEN	ADCEN	IICAEN	SCI1EN	SCI0EN	TM41EN	TM40EN

RTCEN	Provides control of the input clock of a real-time clock (RTC) and a 15-bit interval timer
0	Stop supplying the input clock. • Cannot write the SFR used by the real-time clock (RTC) and 15-bit interval timer. • The real-time clock (RTC) and 15-bit interval timer are in the reset state.
1	An input clock is provided. • Ability to 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 on power-on reset and remains unchanged on other resets.

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

Address: 40020 420H After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	IRDAEN	ADCEN	IICAEN	SCI1EN	SCI0EN	TM41EN	TM40EN

IRDAEN	Provides control of the input clock of the serial interface IRDA
0	Stop supplying the input clock. • IRDA cannot be written using SFR. • IRDA is in a reset state.
1	An input clock is provided. • Ability to read and write SFR used by Ｉ RDA.

ADCEN	Provides control of the input clock of the A/D converter
0	Stop supplying the input clock. • Cannot write A/D converters using SFR. • The A/D converter is in a reset state.
1	An input clock is provided. • SFR can read and write to A/D converters used.

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

SCI1EN	Provides control of the input clock of universal serial communication unit 1
0	Stop supplying the input clock. • Cannot write SFR used by universal serial communication unit 1. • Universal serial communication unit 1 is in reset state.
1	An input clock is provided. • SFR used by universal serial communication unit 1 can read and write.

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

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

Address: 40020 420H After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	IRDAEN	ADCEN	IICAEN	SCI1EN	SCI0EN	TM41EN	TM40EN

TM41EN	Provides control of the input clock of the general-purpose timer unit 1
0	Stop supplying the input clock. • Cannot write the SFR used by universal timer unit 1. • General purpose timer unit 1 is in a reset state.
1	An input clock is provided. • SFR can read and write to the SFR used by universal timer unit 1.

TM40EN	Provides control of the input clock of the general-purpose timer unit 0
0	Stop supplying the input clock. • Cannot write the SFR used by universal timer unit 0. • General purpose timer unit 0 is in the reset state.
1	An input clock is provided. • SFR used by universal timer unit 0 can read and write.

Figure 4-8 Format of peripheral enable register 1 (PER1)

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

symbol	7	6	5	4	3	2	1	0
PER1	SPIEN	0	-	0	DMAEN	EPWMEN	0	0

SPIEN	Provides control of the input clock of the SPI
0	Stop supplying the input clock. • SPI cannot run.
1	An input clock is provided. • SPI can run.

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

EPWMIN	Provides control of the EPWM's input clock
0	Stop supplying the input clock. • EPWM does not run.
1	An input clock is provided. • EPWM can run.

4.3.7 Subsystem clock supply mode control register (OSMC)

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

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

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

OSMC registers are set via 8-bit memory operation instructions.

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

Figure 4-9 Format of subsystem clock supply mode control register (OSMC)

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

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

RTCLPC	The settings in deep sleep mode and sleep mode where the CPU runs at the subsystem clock
0	Allows the subsystem clock to be supplied to peripheral functions (For peripheral functions that are allowed to operate, please refer to Table 19-1 to Table 19-3.)
1	Stop supplying the subsystem clock for the real-time clock and peripheral functions other than the 15-bit interval timer.

WUTMMCK0	Selection of the operating clock for the real-time clock, 1-5-bit interval timer
0	• The subsystem clock is the real-time clock and the operating clock of the 1 5-bit interval
1	• The low-speed internal oscillator clock is the operating clock of the real-time clock and the 15-bit interval timer.

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

This is the register that changes the high-speed internal oscillator frequency set by option byte (000C2H). However, the frequency that can be selected varies depending on the value of the FRQSEL 4 and FRQSEL3 bits of the option byte (000C2H).

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

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

Figure 4-10 Format of high-speed internal oscillator frequency selection register (HOCODIV)

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

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

HOCODIV2	HOCODIV1	HOCODIV0	Selection of clock frequency for high-speed internal oscillators	
			FRQSEL4,3=00	FRQSEL4,3=01

0	0	0	fIH=48MHz
			fHOCO=48MHz
				fIH=64MHz
				fHOCO=64MHz

0	0	1	fIH=24MHz
			fHOCO=48MHz
				fIH=32MHz
				fHOCO=64MHz

0	1	0	fIH=12MHz
			fHOCO=48MHz
				fIH=16MHz
				fHOCO=64MHz

0	1	1	fIH=6MHz
			fHOCO=48MHz
				fIH=8MHz
				fHOCO=64MHz

1	0	0	fIH=3MHz
			fHOCO=48MHz
				fIH=4MHz
				fHOCO=64MHz

1	0	1	Disable the setting.	fIH=2MHz
				fHOCO=64MHz

Others			Settings are prohibited	

Note 1 The HOCODIV register must be set in the state of selecting a high-speed internal oscillator clock (f_{IH}) 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 transfer time:

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

4.3.9 High-speed internal oscillator trim register (HIOTRM)

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

Note If the temperature and voltage at the V_{DD} pin change after the accuracy is corrected, the frequency changes. In the event of changes in temperature and voltage at the V_{DD} pin, correction is required before the accuracy of the required frequency or periodically.

Figure 4-11 Format of high-speed internal oscillator trim register (HIOTRM)

Address: 40021C00H After reset: Note R/W

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

HIOTRM5	HIOTRM4	HIOTRM3	HIOTRM2	HIOTRM1	HIOTRM0	High-speed 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 adjustment value at the time of shipment.

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

4.3.10 Subsystem clock selection register (SUBCKSEL)

The SUBCKSEL register is the register that selects the subsystem clock fSUB and the low-speed internal oscillator clock FIL.

The SUBCKSEL registers are set via 8-bit memory operation instructions.

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

Figure 4-12 Format of subsystem clock selection register (SUBCKSEL)

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

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

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

4.4 System clock oscillation circuit

4.4.1 X1 oscillation circuit

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

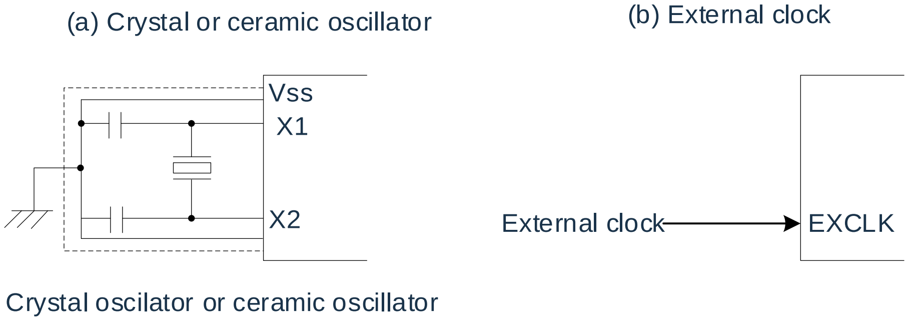
When using the X1 oscillation circuit, the following settings must be made to bit 7 and bit 6 (EXCLK, OSCSEL) of the Clock Operation Mode Control Register (CMC):

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

When the X1 oscillation circuit is not used, it must be set to port mode (EXCLK, OSCSEL=0, 0). Also, when it is not used as an input and output port, please refer to “Table 2-3 for the handling of each unused pin”.

An example of an external circuit for an X1 oscillation circuit is shown in Figure 4-13.

Figure 4-13 Example of an external circuit for the X1 oscillation circuit


(a) Crystal or ceramic oscillator
Vss
X1
X2
Crystal oscilator or ceramic oscillator
(b) External clock
External clock
EXCLK

Considerations are shown on the following page.

4.4.2 XT1 oscillation circuit

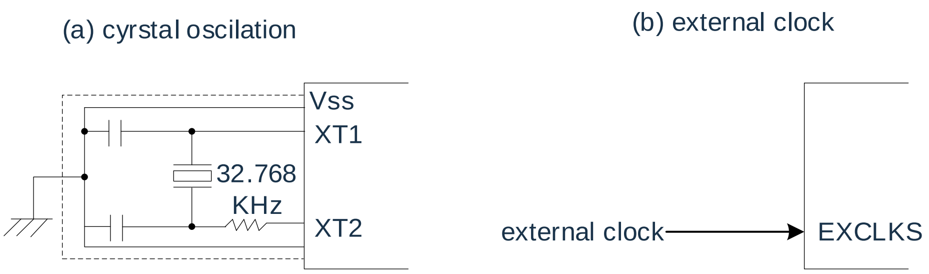
The XT1 oscillation circuit is via a crystal resonator (32.768kHz (TYP.)) connected to the XT1 pin and the XT2 pin to oscillate. When using the XT1 oscillation circuit, bit4 (OSCSELS) of the clock mode control register (CMC) must be set to “1” to also input an external clock, which must be given the EXCLKS pin input clock signal.

When using the XT1 oscillation circuit, bit5 and bit4 (EXCLKS, OSCSELS) of the clock operation mode control register (CMC) must be controlled) to make the following settings:

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

When the XT1 oscillation circuit is not used, it must be set to port mode (EXCLKS, OSCSELS=0, 0). Also, when it is not used as an input and output port, please refer to “Table 2-3 Handling of Unused Pins”. An example of an external circuit for an XT1 oscillation circuit is shown in Figure 4-14.

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



Note When using the X1 oscillation circuit and the XT1 oscillation circuit, in order to avoid the influence of the routing capacitance, etc., the dotted lines in Figure 4-13 and Figure 4-14 must be routed as follows:

• Routing must be kept as short as possible.

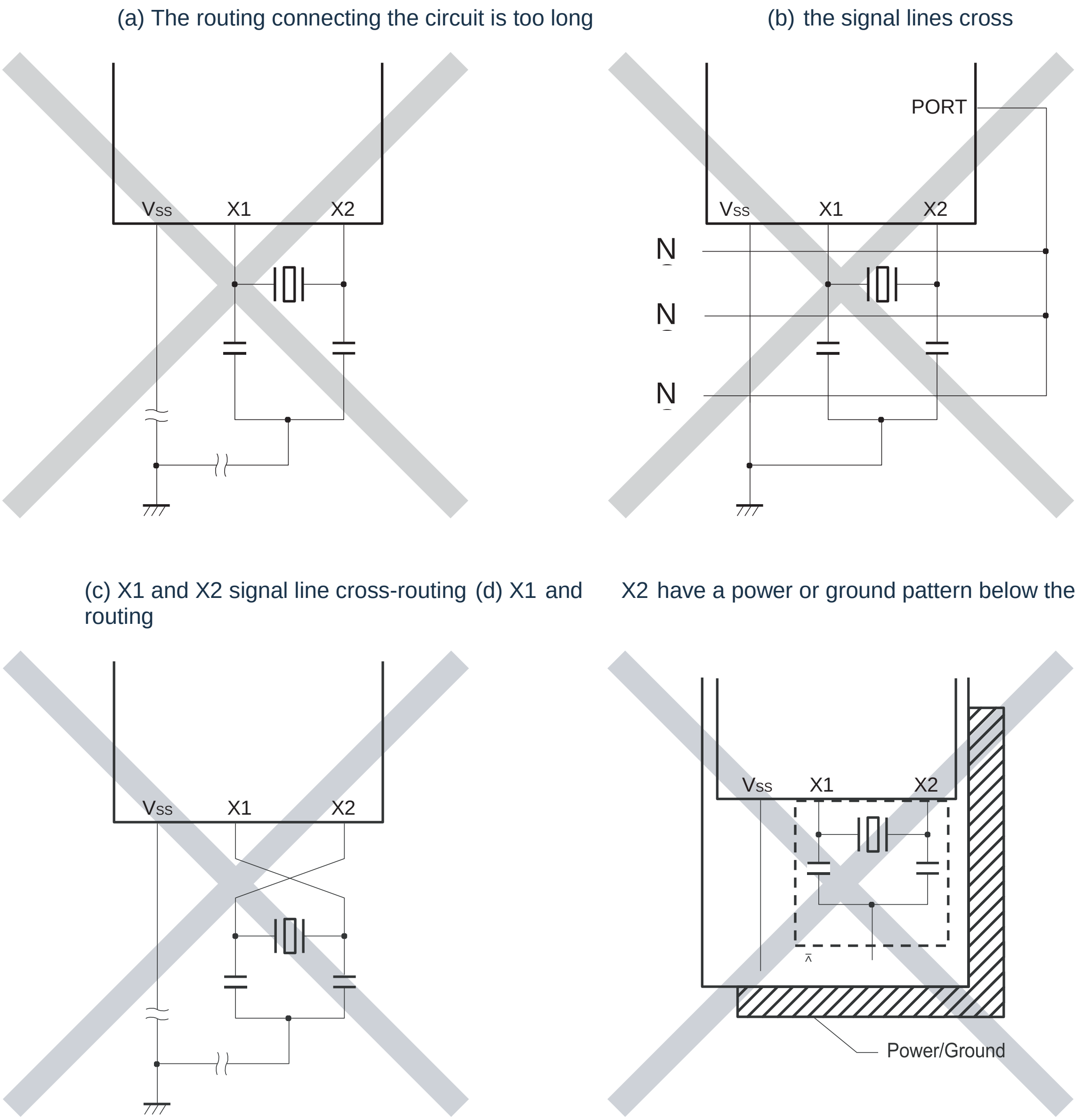
• Cannot cross with other signal lines and cannot approach routing that has a large current flow with variations.

• The capacitor grounding position of the oscillation circuit must always be kept at the same potential as the V_{SS}, and the ground pattern of large currents flowing through must not be grounded.

• The signal cannot be removed from the oscillation circuit.

An example of an incorrect resonator connection is shown in Figure 4-15.

Figure 4-15 Example of an incorrect resonator connection (1/2).



(a) The routing connecting the circuit is too long

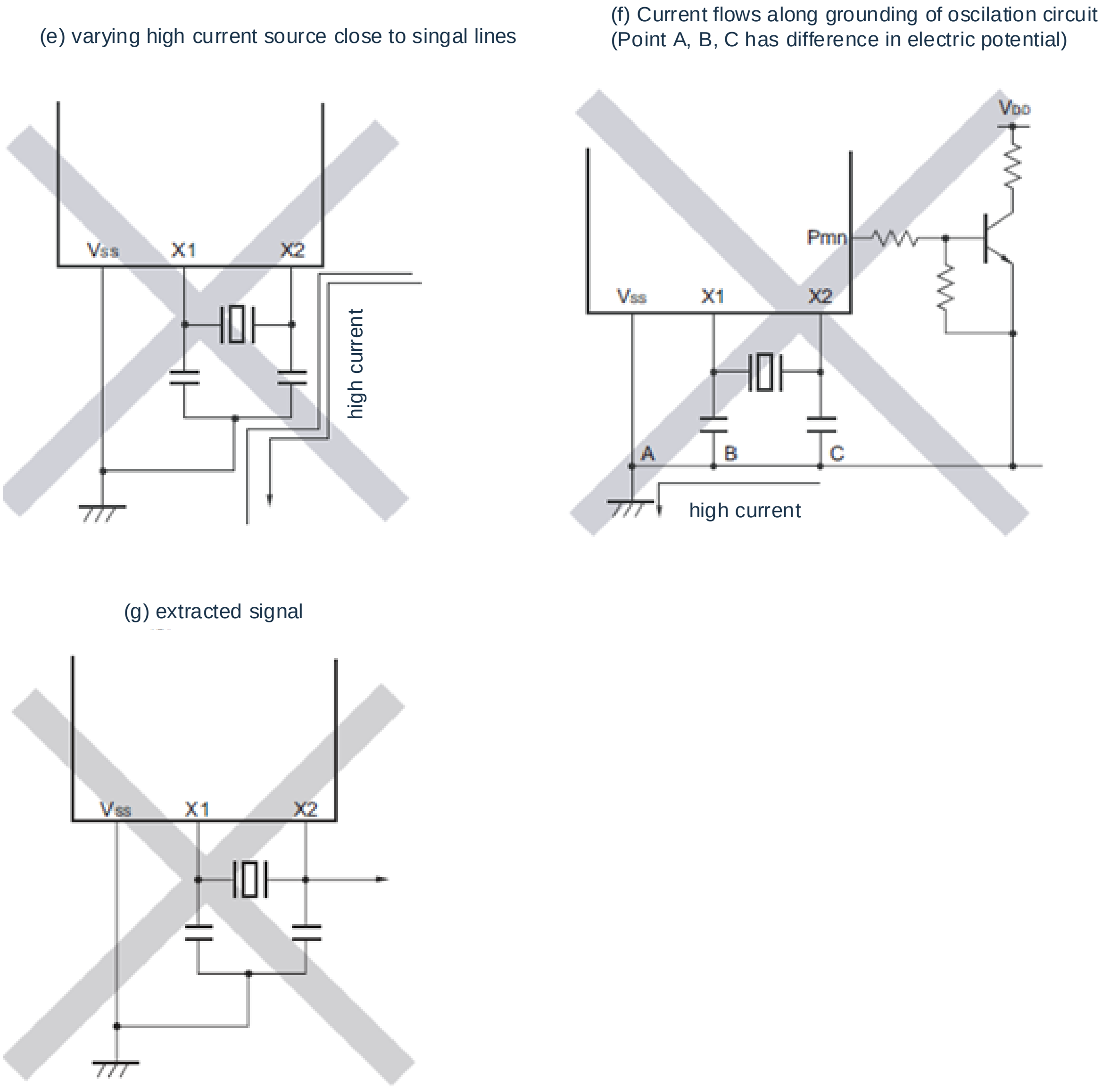
(b) the signal lines cross

(c) X1 and X2 signal line cross-routing (d) X1 and X2 have a power or ground pattern below the routing

In multilayer boards or bi-panels, power or ground graphics cannot be configured below the routing area of the X1 pin, X2 pin, and resonator (dotted in the figure). The routing cannot produce a capacitive component that affects the oscillation characteristics.

Note In the case of a subsystem clock, replace X1 and X2 with XT1 and XT2 respectively when reading, and insert a serial resistor on the XT2 side.

Figure 4-17 Example of an incorrect resonator connection (2/2)



(e) varying high current source close to singal lines

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

(g) extracted signal

Note When X2 and XT1 are routed in parallel, the crosstalk noise of X2 will superimpose on XT1 and cause malfunction.

Note In the case of a subsystem clock, replace X1 and X2 with XT1 and XT2 respectively when reading, and insert a series resistor on the XT2 side.

4.4.3 High-speed internal oscillator

The CMS32L051 has a built-in high-speed internal oscillator. The frequency can be selected from 64MHz, 48MHz, 32MHz, 24MHz, 16MHz, 12MHz, 8MHz, 6MHz, 4MHz, 3MHz and 2MHz via the option byte (000C2H). The oscillation can be controlled via bit 0 (HIOSTOP) of the Clock Operation Status Control Register (CSC).

After the power-on reset is released, the high-speed internal oscillator automatically begins to oscillate.

4.4.4 Low-speed internal oscillator

The CMS32L051 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, and the external reference clock for the SysTick timer, as well as a CPU clock and a peripheral module clock.

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

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

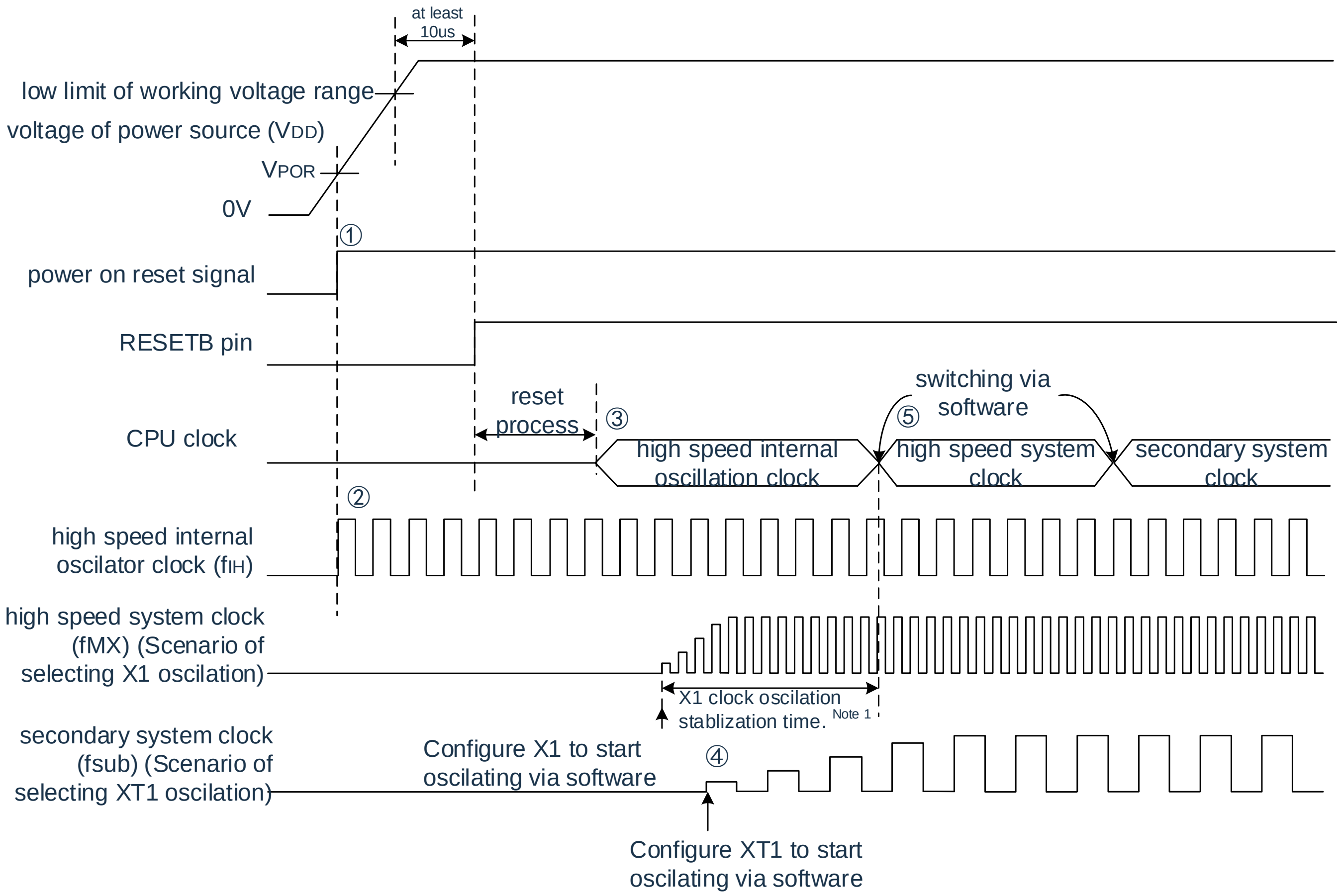
4.5 Operation of clock generation circuit

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

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

After the CMS32L051 is released, the CPU starts operating through the output of a high-speed internal oscillator. The operation of the clock generation circuit when the power is turned on is shown in Figure 4-16.

Figure 4-16 Operation of clock generation circuit when the power is turned on



① After the power is turned on, an internal reset signal is generated through a power-on reset (POL) circuit.

However, the reset state is maintained by voltage detection circuitry or an external reset until the operating voltage range shown in the AC characteristics of the data sheet is reached (the figure above is an example when using an external reset).

② If the reset is released, the high-speed internal oscillator automatically begins to oscillate.

③ After the reset is released, a voltage stabilization wait and reset process occurs, and then the CPU starts operating with a high-speed internal oscillator clock.

④ The X1 clock or the starting oscillation of the XT1 clock must be set via software (see “Example of setting up a 4.6.2 X14.6.2 Example of setting up an X1 oscillation circuit and “4.6.3”).4.6.3 Example of setting up an XT1 oscillation circuit).

⑤ If you want to switch the CPU clock to the X1 clock or the XT1 clock, you must switch through the software settings after waiting for the clock oscillation to stabilize (see the “4.6.2 X1 4.6.2Example of setting up an X1 oscillation circuitand “4.6.3Example of setting up an XT1 oscillation circuit“).

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

Note that if an external clock is used for the EXCLK pin input, there is no need for oscillation stabilization wait times.

4.6 Clock control

4.6.1 Example of setting up a high-speed internal oscillator

The CPU/peripheral hardware clock (fCLK) must run at the high-speed internal oscillator clock after a reset has been released. The frequency of the high-speed internal oscillator can be selected from 64MHz, 48MHz, 32MHz, 24MHz, 16MHz, 12MHz, 8MHz, 6MHz, 4MHz, 3MHz and 2MHz via 4 bits FRQSEL0 to FRQSEL of the option byte (000C2H). In addition, the frequency can be changed via the high-speed internal oscillator's frequency selection register (HOCODIV).

【Option byte setting】

Address: 000C2H

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

FRQSEL4	FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0	The frequency of the high-speed internal oscillator	
					f_{HOCO}	f_{IH}
0	1	0	0	0	64MHz	64MHz
0	0	0	0	0	48MHz	48MHz
0	1	0	0	1	64MHz	32MHz
0	0	0	0	1	48MHz	24MHz
0	1	0	1	0	64MHz	16MHz
0	0	0	1	0	48MHz	12MHz
0	1	0	1	1	64MHz	8MHz
0	0	0	1	1	48MHz	6MHz
0	1	1	0	0	64MHz	4MHz
0	0	1	0	0	48MHz	3MHz
0	1	1	0	1	64MHz	2MHz
Others					Setting is prohibited	

[Setting of the Frequency Selection Register (HOCODIV) of the High Speed Internal Oscillator].

Address: 0x40021C20

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

HOCODIV2	HOCODIV1	HOCODIV0	Selection of clock frequency for high-speed internal oscillators			
			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 the setting.	f_{IH}=2MHz f_{HOCO}=32MHz	f_{IH}=3MHz f_{HOCO}=48MHz	f_{IH}=4MHz f_{HOCO}=64MHz
1	0	1	Disable the setting.	f_{IH}=1MHz f_{HOCO}=32MHz	Disable the setting.	f_{IH}=2MHz f_{HOCO}=64MHz
Others			Setting is prohibited			

4.6.2 Example of setting up an X1 oscillation circuit

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

[Register Setting] The registers must be set in order from (1) to (5).

(1) Put the OSCSEL position of the CMC register “1”, when f_X is greater than or equal to 10MHz, the AMPH Position “1” to make the X1 oscillation circuit run.

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

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

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

	7	6	5	4	3	2	1	0
OSTS	0	0	0	0	0	OSTS2 0	OSTS1 1	OsTS0 0

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

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

(4) Wait through the OSC register for the oscillation of the X1 oscillation circuit to stabilize.

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

	7	6	5	4	3	2	1	0
OSTC	MOST8 1	MOST9 1	MOST10 1	MOST110	MOST13 0	MOST15 0	MOST17 0	MOST18 0

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

	7	6	5	4	3	2	1	0
CKC	CLS0	CSS 0	MCS 0	MCM0 1	0	0	0	0

4.6.3 Example of setting up an XT1 oscillation circuit

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

[Register Setting] The registers must be set in order from (1) to (5).

(1) In deep sleep mode or sleep mode where the CPU is running at the subsystem clock, the RTCLPC position “1” must be applied when only the real-time clock and the 15-bit interval timer are run at the subsystem clock (ultra-low current consumption).

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

(2) Put the OSCSELS position of the CMC register “1” so that the XT1 oscillation circuit runs.

	7	6	5	4	3	2	1	0
CMC	EXCLK0	OSCSEL0	EXCLKS0	OSCSELS1	0	AMPHS1 0/1	AMPHS0 0/1	AMPH0

AMPHS0 and AMPHS 1 bit: Set the oscillation mode of the XT1 oscillation circuit.

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

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

(4) The oscillation stabilization time required by the subsystem clock must be waited for through software and timer functions, etc.

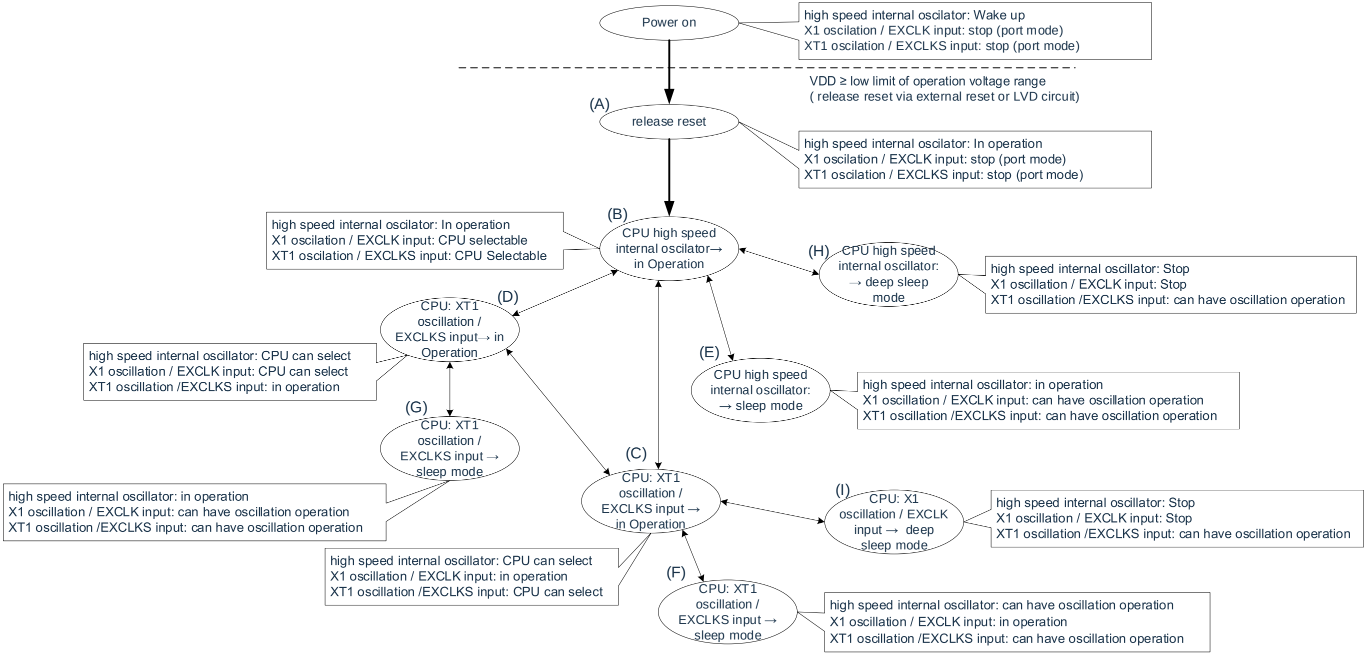
(5) Set the XT1 oscillation clock to the CPU/peripheral hardware clock through the CSS bit of the CKC register.

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

4.6.4 State transition graph of the CPU clock

The CPU clock state transfer diagram of this product is shown in Figure 4-17.

Figure 4-17 State transfer diagram of the CPU clock


Power on
high speed internal oscilator: Wake up
X1 oscilation / EXCLK input: stop (port mode)
XT1 oscilation / EXCLKS input: stop (port mode)
VDD ≥ low limit of operation voltage range
(release reset via external reset or LVD circuit)
(A)
release reset
high speed internal oscilator: In operation
X1 oscilation / EXCLK input: stop (port mode)
XT1 oscilation / EXCLKS input: stop (port mode)
(B)
CPU high speed internal oscilator→ in Operation
high speed internal oscilator: In operation
X1 oscilation / EXCLK input: CPU selectable
XT1 oscilation / EXCLKS input: CPU Selectable
(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
(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
(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
(G)
CPU: XT1 oscillation / EXCLKS input → sleep mode
high speed internal oscillator: in operation
X1 oscillation / EXCLK input: can have oscillation operation
XT1 oscillation /EXCLKS input: can have oscillation operation
(C)
CPU: XT1 oscillation / EXCLKS 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
high speed internal oscillator: CPU can select
X1 oscillation / EXCLK input: in operation
XT1 oscillation /EXCLKS input: CPU can select
(F)
CPU: XT1 oscillation / EXCLKS 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

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

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

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

State transition	Setting of the SFR register
(A)→(B)	There is no need to set the SFR register (the initial state after the reset is released).

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

(Order of setting SFR registers). →

The setting flag of the SFR register State transition	CMC register Note 1			OSTS register	CSC register	OSTC register	CKC register
	EXCLK	OSCSEL	AMPH		MSTOP		MCM0
(A) →(B) →(C) (X1 clock:1MHz≤f_X≤10MHz)	0	1	0	Note 2	0	Confirmation is required	1
(A) →(B) →(C) (X1Clock:10MHz<f_X≤20MHz)	0	1	1	Note 2	0	Confirmation is required	1
(A) →(B) →(C) (External Master Clock)	1	1	×	Note 2	0	No confirmation is required	1

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

2. The following settings must be made for the oscillation settling time of the Oscillation Settling Time Selection Register (OSTS).

• Oscillation settling time of the oscillation settling time counter status register (OSTC) ≤ oscillation settling time set by the OSTS register

Note that the clock must be set after the supply voltage reaches the set clock operable voltage (referring to the electrical characteristics of the data sheet).

(3) After the reset is released (A), the CPU is transferred to the secondary system clock to run (D). (The CPU runs with a high-speed internal oscillator clock (B) immediately after the reset is released.)

(Order of setting SFR registers). →

The setting flag of the SFR register State transition	CMC register note				CSC register	Oscillation stabilizes wait	CKC register
	EXCLKS	OSCSELS	AMPHS1	AMPHS0	XTSTOP		CSS
(A) →(B) →(D) (XT1 clock).	0	1	0/1	0/1	0	need	1
(A) →(B) →(D) (External Subclock)	1	1	×	×	0	need	1

Note that after the reset is released, only one clock operation mode control register (CMC) can be written through the 8-bit memory operation command.

Note 1 ×: Ignore

2.Table 4-3(A)~(I) corresponding to fig Figure 4-17(A)~(I).

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

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

(Order of setting SFR registers). →

Setting flag of the SFR register / State transition	CMC register Note 1			OSTS register	CSC register	OSTC register	CKC register
	EXCLK	OSCSEL	AMPH		MSTOP		MCM0
(B) →(C) (X1 Clock:1MHz≤f_X≤10MHz)	0	1	0	Note 2	0	Confirmation is required	1
(B) →(C) (X1Clock:10MHz<f_X≤20MHz)	0	1	1	Note 2	0	Confirmation is required	1
(B) →(C) (External Master Clock)	1	1	×	Note 2	0	No confirmation is required	1

Not required if set.　　Not required in high-speed system clock
operation.

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

2. The following settings must be made for the oscillation settling time of the Oscillation Settling Time Selection Register (OSTS).

• The expected oscillation settling time of the status register (OSTC) of the oscillation settling time counter≤ the oscillation settling time set by the OSTS register

Note that the clock must be set after the supply voltage reaches the set clock operable voltage (referring to the electrical characteristics of the data sheet).

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

(Order of setting SFR registers). →

The setting flag of the SFR register / State transition	CMC register note			CSC register	Oscillation stabilizes wait	CKC register
	EXCLKS	OSCSELS	AMPHS1, 0	XTSTOP		CSS
(B) →(D) (XT1 clock).	0	1	00: Low power consumption oscillation 01: Usually	0	need	1
(B) →(D) (External Subclock)	1	1	×	0	need	1

Not required if set.　　Not required in the operation of the
subsystem clock.

Note that after the reset is released, only one clock operation mode control register (CMC) can be written through the 8-bit memory operation command. Not required if set.

Note 1 ×: Ignore

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

Table 4-3 Example of CPU transfering and SFR register setting (3/5)

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

(Order of setting SFR registers).

The setting flag of the SFR register State transition	CSC registers	Oscillation accuracy is stable waiting	CKC registers
	HIOSTOP		MCM0
(C) →(B)	0	1us	0

Not required in high-speed internal oscillator clock operation.

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

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

(Order of setting SFR registers).

The setting flag of the SFR register State transition	CSC registers	Oscillation accuracy is stable waiting	CKC registers
	XTSTOP		CSS
(C) →(D)	0	need	1

Not required in the operation of the secondary system clock.

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

(Order of setting SFR registers).

The setting flag of the SFR register State transition	CSC registers	Oscillation accuracy is stable waiting	CKC registers
	HIOSTOP		CSS
(D) (B)	0	1us	0

Not required in high-speed internal oscillator clock operation.

Note 1. Table 4-3 (A) ~ (I) correspond to Figure 4-17 (A) ~ (I).

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

Table 4-3 Example of CPU transfering and SFR register setting (4/5)

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

(Order of setting SFR registers) →

The setting flag of the SFR register State transition	OSTS register	CSC registers	OSTC registers	CKC registers
		MSTOP		CSS
(D) →(C) (X1Clock:1MHz≤f_X≤10MHz)	concentrate	0	Confirmation is required	0
(D) →(C) (X1Clock:10MHz＜f_X≤20MHz)	concentrate	0	Confirmation is required	0
(D) →(C) (External Master Clock)	concentrate	0	No confirmation is required	0

Not required in high-speed system clock operation.

Note The following settings must be made for the oscillation settling time of the Oscillation Settling Time Selection Register (OSTS).

• The expected oscillation stability time of the status register of the oscillation stability time counter (OSTC) ⩽ the oscillation stability time set by the OSTS register

Note that the clock must be set after the supply voltage reaches the set clock operable voltage (referring to the electrical characteristics of the data sheet).

(10)

- The CPU is transferred to sleep mode (E) while the high-speed internal oscillator clock is running (B).
- CPU transitions to sleep mode (F) while running at a high-speed system clock (C).
- The CPU is transferred to sleep mode (G) while the secondary system clock is running (D).

State transition	Set the content
(B) →(s) (C) →(F) (D) →(G)	Execute the WFI instruction.

Remark: (A)~(I) in Table 4-3 correspond to (A)~(I) in Figure 4-17.

Table 4-3 Example of CPU transfering and SFR register setting (5/5)

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

- CPU transitions to deep sleep mode (I) while running at a high-speed system clock (C).

(Set Order) →

State transition		Set the content		
(B) →(M)		Stop it Peripheral features that cannot run in deep sleep mode.	—	The SCR register bit2 (SLEEPDEEP) is set to 1 and the WFI instruction is executed.
(C) →(s)	X1 oscillation		Set the OSTS registers.	
	External clock		—	

Remark: (A)~(I) in Table 4-3 correspond to (A)~(I) in Figure 4-17.

4.6.5 Conditions before CPU clock transfer and processing after transfer

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

Table4-4 Regarding CPU clock transfer (1/2)

CPU clock		Conditions before transfer	Post-transfer processing
Before transferring	After transferring		
High-speed internal oscillator clock	X1 clock	X1 oscillation is stable. • OSCSEL=1, EXCLK=0, MSTOP=0 • After over-oscillation settling time	If the oscillation of the high-speed internal oscillator is stopped (HIOSTOP=1), the operating current can be reduced.
	External master system clock	Set the external clock of the EXCLK pin input to active. • OSCSEL=1, EXCLK=1, MSTOP=0	
	XT1 clock	XT1 oscillation is stable. • OSCSELS=1, EXCLKS=0, XTSTOP=0 • After the oscillation stabilization time	
	External subsystem clock	Set the external clock of the EXCLKS pin input to active. • OSCSELS=1, EXCLKS=1, XTSTOP=0	
X1 clock	High-speed internal oscillator clock	Allows high-speed internal oscillator oscillation. • HIOSTOP=0 • After the oscillation stabilization time	Can stop the oscillation of X1 (MSTOP=1).
	External master system clock	Cannot be transferred.	—
	XT1 clock	XT1 oscillation is stable. • OSCSELS=1, EXCLKS=0, XTSTOP=0 • After the oscillation stabilization time	Can stop the oscillation of X1 (MSTOP=1).
	External subsystem clock	Set the external clock of the EXCLKS pin input to active. • OSCSELS=1, EXCLKS=1, XTSTOP=0	Can stop the oscillation of X1 (MSTOP=1).
External master system clock	High-speed internal oscillator clock	Allows high-speed internal oscillator oscillation. • HIOSTOP=0 • After the oscillation stabilization time	Input to the external master system clock can be set to be invalid (MSTOP=1).
	X1 clock	Cannot be transferred.	—
	XT1 clock	XT1 oscillation is stable. • OSCSELS=1, EXCLKS=0, XTSTOP=0 • After the oscillation stabilization time	Input to the external master system clock can be set to be invalid (MSTOP=1).
	External subsystem clock	Set the external clock of the EXCLKS pin input to active. • OSCSELS=1, EXCLKS=1, XTSTOP=0	Input to the external master system clock can be set to be invalid (MSTOP=1).

Table 4-4 Transfers of CPU clocks (2/2)

CPU clock		Conditions before transfer	Post-transfer processing
Before transferring	After transferring		
XT1 clock	High-speed internal oscillator clock	The high-speed internal oscillator is oscillating and the high-speed internal is chosen The oscillator clock acts as the master 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 chosen as the main system Unified clock. • OSCSEL=1, EXCLK=0, MSTOP=0 • After the oscillation stabilization time • MCS=1	
	External master system clock	Leave the external clock of the EXCLK pin input active and select the high-speed system clock as the master system clock. • OSCSEL=1, EXCLK=1, MSTOP=0 • MCS=1	
	External subsystem clock	Cannot be transferred.	—
External subsystem clock	High-speed internal oscillator clock	The high-speed internal oscillator is oscillating and the high-speed internal is chosen The oscillator clock acts as the master system clock. • HIOSTOP=0, MCS=0	Input to the external subsystem clock can be set to invalid (XTSTOP=1).
	X1 clock	The X1 oscillation is stable and the high-speed system clock is chosen as the main system Unified clock. • OSCSEL=1, EXCLK=0, MSTOP=0 • After the oscillation stabilization time • MCS=1	
	External master system clock	Leave the external clock of the EXCLK pin input 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.	—

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

CPU clock switching (master system clock) can be performed by setting bit6 and bit4 (C SS, MCM0) of the system clock control register (CKC). ↔ Secondary system clock) and the switching of the main system clock (high-speed internal oscillator clock ↔ high-speed system clock).

Instead of making the actual switch immediately after overwriting the CKC registers, several clocks continue to run at the pre-switching clock after changing the CKC registers (refer to Table 4-5~Table 4-7).

It can be determined by bit7 (CLS) of the CKC register whether the CPU is running from the primary system clock or the secondary system clock. The bit5 (MCS) of the CKC register can tell whether the master system clock is running on a high-speed system clock or a high-speed internal oscillator clock.

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

Table 4-5 Time required to switch the master system clock

Clock A	Switch directions	Clock B	remark
f_{IH}	↔	f_{MX}	Reference Table 4-6.
f_{MAIN}	↔	f_{SUB}	Reference Table 4-7.

Table 4-6 f_{IH} f_{MX} maximum number of clocks required

The setting value before switching		The setting value after switching	
MCM0		MCM0	
		0 (f_{MAIN}=f_{IH})	1 (f_{MAIN}=f_{MX})
0 (f_{MAIN}=f_{IH})	f_{MX}≥f_{IH}		2 clocks
	f_{MX}<f_{IH}		2 f_{IH}/f_{MX} clock
1 (f_{MAIN}=f_{MX})	f_{MX}≥f_{IH}	2 f_{MX}/f_{IH}clocks	
	f_{MX}<f_{IH}	2 clocks	

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

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

Note 1 The number of clocks in Table 4-6and Table 4-7is the number of CPU clocks before switchover.

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

For example, the main system clock is switched from the high-speed system clock to the high-speed internal oscillator clock (f-IH=8MHz and fMX=10MHz oscillation is selected).

2fMX/fIH = 2 (10/8) = 2.5 3 clocks

4.6.7 Condition before the clock oscillation stops

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

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

clock	Conditions before the clock stops (invalid external clock input)	The flag setting of the SFR register
High-speed internal oscillator clock	MCS=1 or CLS=1 (The CPU runs on a clock other than the high-speed internal oscillator clock).	HIOSTOP=1
X1 clock	MCS=0 or CLS=1 (The CPU runs on a clock other than the high-speed system clock).	MSTOP=1
External master system clock		
XT1 clock	CLS=0 (The CPU runs on a clock other than the secondary system clock).	XTSTOP=1
External subsystem clock		

4.7 High-speed internal oscillation correction

4.7.1 High-speed internal oscillation self-adjustment function

This function measures the frequency of high-speed internal oscillators with the subsystem clock fSUB (32.768KHz) and corrects the frequency accuracy of high-speed internal oscillators fHOCO in real time.

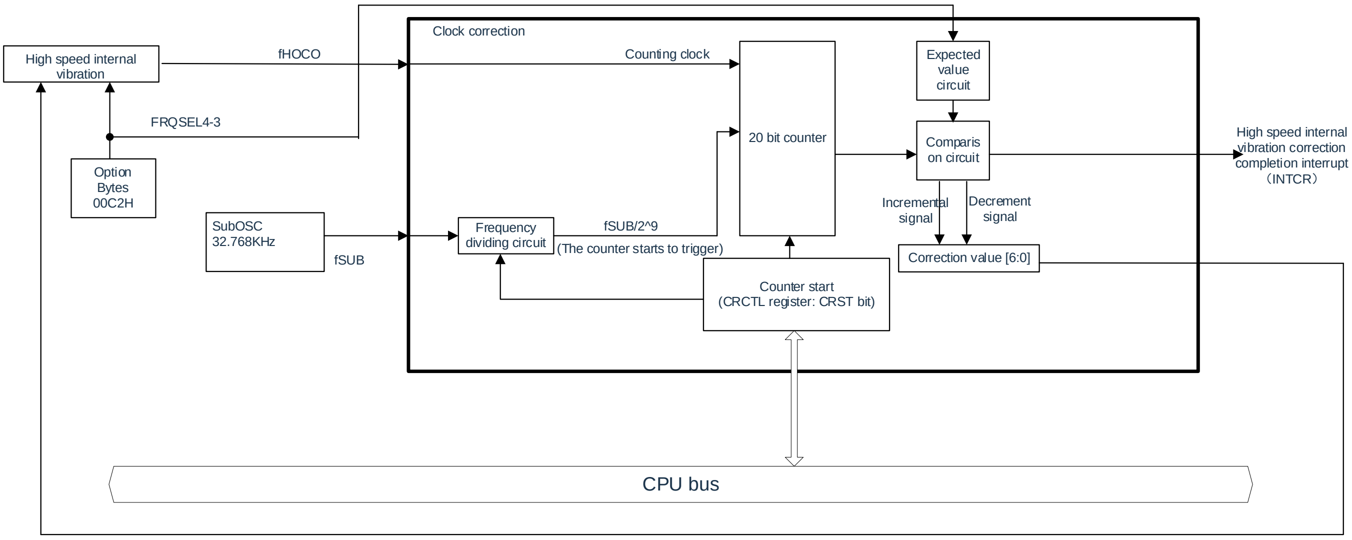
Table 4-9 shows the operation specifications of high-speed internal frequency correction function, and Figure 4-18 shows the block diagram of high-speed internal frequency correction function.

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

item	content
Base clock	・ fSUB/2^9 (subsystem clock 32.768KHz).
Correct the object clock	・ fHOCO (fast inner swing)
Operation mode	・ Continuous Operation mode A mode of continuous high-speed internal vibration frequency correction ・ Interval Operation mode A mode of high-speed internal frequency correction is spaced by using a timer clock end, etc
Clock accuracy adjustment function	・ Correction time: Correction period (31.2ms) X (correction number -0.5) [Note]
interrupt	・ Interrupt when high-speed internal oscillation frequency correction is complete (when interrupt permission is on)

Note: Correction time: Varies depending on the number of corrections.
Correction period: The total time of the frequency determination phase and the frequency correction phase.
Number of corrections: The number of corrections in which the frequency is bundled to the expected range.

Figure 4-18 High-speed internal vibration frequency correction function



4.7.2 Register description

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

Table 4-10 Format of high-speed internal vibration frequency correction function registers

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

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

Control register for high-speed internal oscillation frequency correction.
The HOCOFC register is set via an 8-bit memory operation command.
After the reset signal is generated, the value of this register becomes “00H”.

Figure 4-19 Format of high-speed internal oscillation frequency correction control register (HOCOFC)

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

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

FCMD Note 1	High-speed internal oscillation frequency correction function operation mode
0	Continuous operation mode
1	Interval operation mode

FCIE	Interrupt control completed by high-speed internal oscillation frequency correction
0	There is no interruption after the high-speed internal oscillation frequency correction is completed
1	Interrupts occur after high-speed internal oscillation frequency correction is complete

FCST Note 2	High speed internal frequency correction circuit motion control/status
0	The high-speed internal oscillation frequency correction circuit is in the process of stopping/stopping
1	High-speed internal frequency correction circuit operation starts/in motion
When in continuous operation mode, the software writes 0 to stop the action. When the interval operation mode is complete, the hardware clears the FCST bit.	

Note 1. When the FCST bit is 1, it is forbidden to override the FCMD bit.

2. When writing 1 to the FCST bit, first confirm that the value of the current FCST bit is 0 and then write 1 to it. Since hardware clears priority, when writing 1 to the FCST bit immediately after the interval operation is completed (when the interrupt generation is completed by high-speed internal resonance frequency correction), the operation should be performed at least 1 cycle after the interrupt generation is completed after the high-speed internal resonance frequency correction is completed.
After writing 0 to the FCST bit (the high-speed internal resonance frequency correction circuit Operation stops), fHOCO prohibits writing 1 to the FCST bit for 2 cycles (the high-speed internal resonance frequency correction circuit Operation begins).

Note: Bit 5 to 1 must be written to 0.

4.7.3 Operation description

4.7.3.1 Operation overview

The high-speed internal oscillation frequency correction function uses the subsystem clock (fSUB) as a reference to generate a correction period, measure the frequency of high-speed internal oscillation, and correct the frequency accuracy of high-speed internal oscillator in real time. Clock adjustment for the repetition of the operation of the measurement phase and the frequency correction phase. The correction calculus is performed during the frequency measurement phase, and the correction values that reflect the results of the correction calculus are saved during the frequency correction phase.

Table 4-11 shows the input frequency and correction period of high-speed internal vibration, and Figure 4-20 shows the sequence diagram of high-speed internal vibration frequency correction action (detailed).

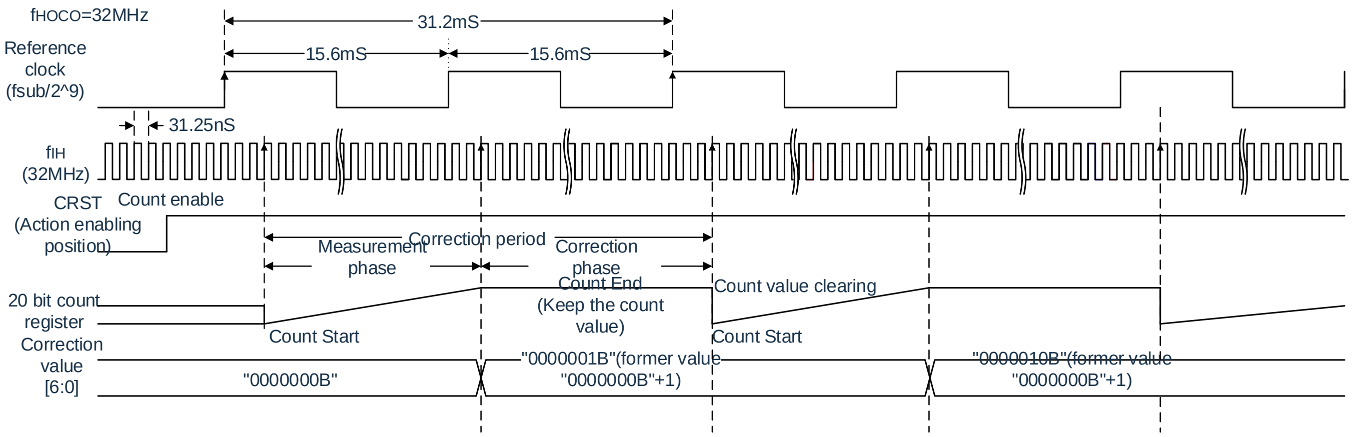
Table 4-11 High Speed Internal Vibration Input Frequency and Correction Period

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

Note: FRQSEL4-FRQSEL3 is option byte0 0C2H bit4-bit3

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

Figure 4-20 High-speed internal vibration frequency correction Operation timing diagram (detailed)



Note: The basic actions of the continuous Operation mode and the interval Operation mode are the same. The difference is whether the removal of FCST bits is controlled by software or hardware. In addition, only the system reset can clearly correct the value.

（1） Continuous operation mode

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

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

After the high-speed internal clock frequency correction action, the rising edge frequency counter of the reference clock (fSUB/2 9) begins counting and stops counting on the rising edge of the next reference clock (fSUB/2^9). (frequency measurement phase).

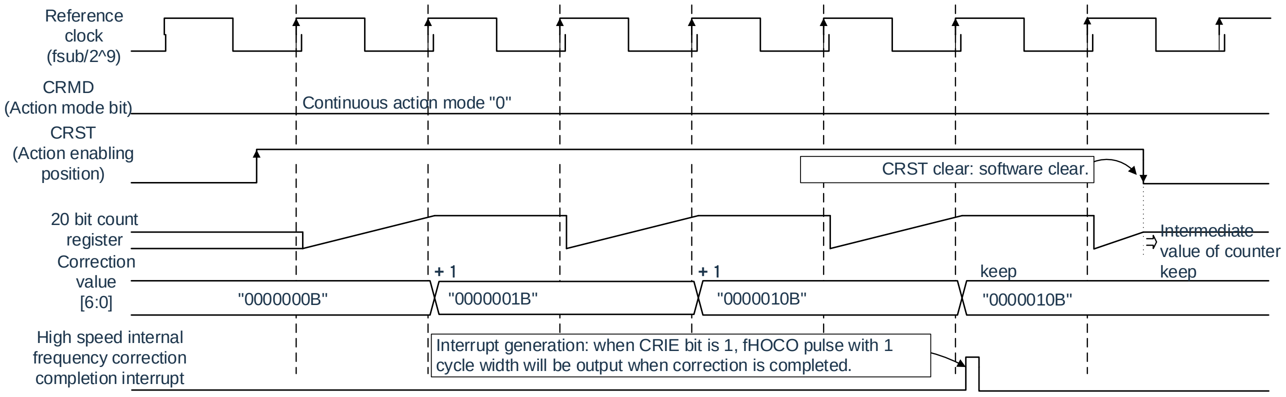
The count value is then compared to the expected value and the correction value adjustment is made as described below. (Frequency correction phase)

- When the count value is greater than expected: the correction value is -1
- The count value is more than expected hours: the correction value is +1
- When the count value is within the expected range: the correction value is maintained (high-speed internal clock frequency correction ends)

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

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

Figure 4-21 Continuous operation mode timing diagram


Reference clock (fsub/2^9)
CRMD (Action mode bit)
Continuous action mode "0"
CRST (Action enabling position)
CRST clear: software clear.
20 bit count register
Intermediate value of counter keep
Correction value [6:0]
+ 1
+ 1
keep
"0000000B"
"0000001B"
"0000010B"
"0000010B"
High speed internal frequency correction completion interrupt
Interrupt generation: when CRIE bit is 1, fHOCO pulse with 1 cycle width will be output when correction is completed.

（2） Interval operation mode

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

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

After the high-speed internal clock frequency correction action, the rising edge frequency counter of the reference clock (fSUB/2 9) begins counting and stops counting on the rising edge of the next reference clock (fSUB/2^9). (frequency measurement phase).

The count value is then compared to the expected value and the correction value adjustment is made as described below. (Frequency correction phase)

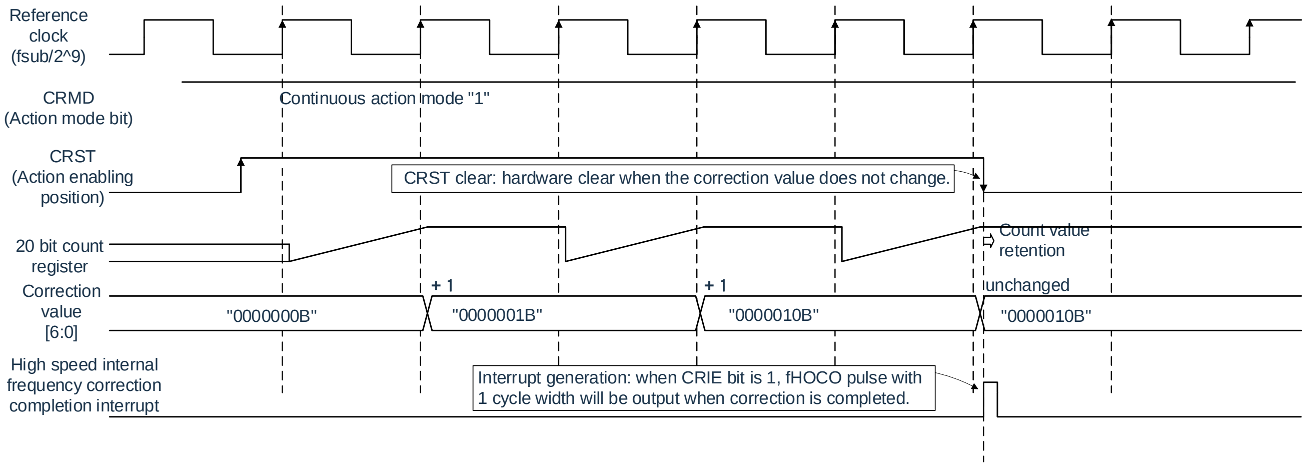
- When the count value is greater than expected: the correction value is -1
- The count value is more than expected hours: the correction value is +1
- When the count value is within the expected range: the correction value is maintained

(high-speed internal clock frequency correction ends)

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

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

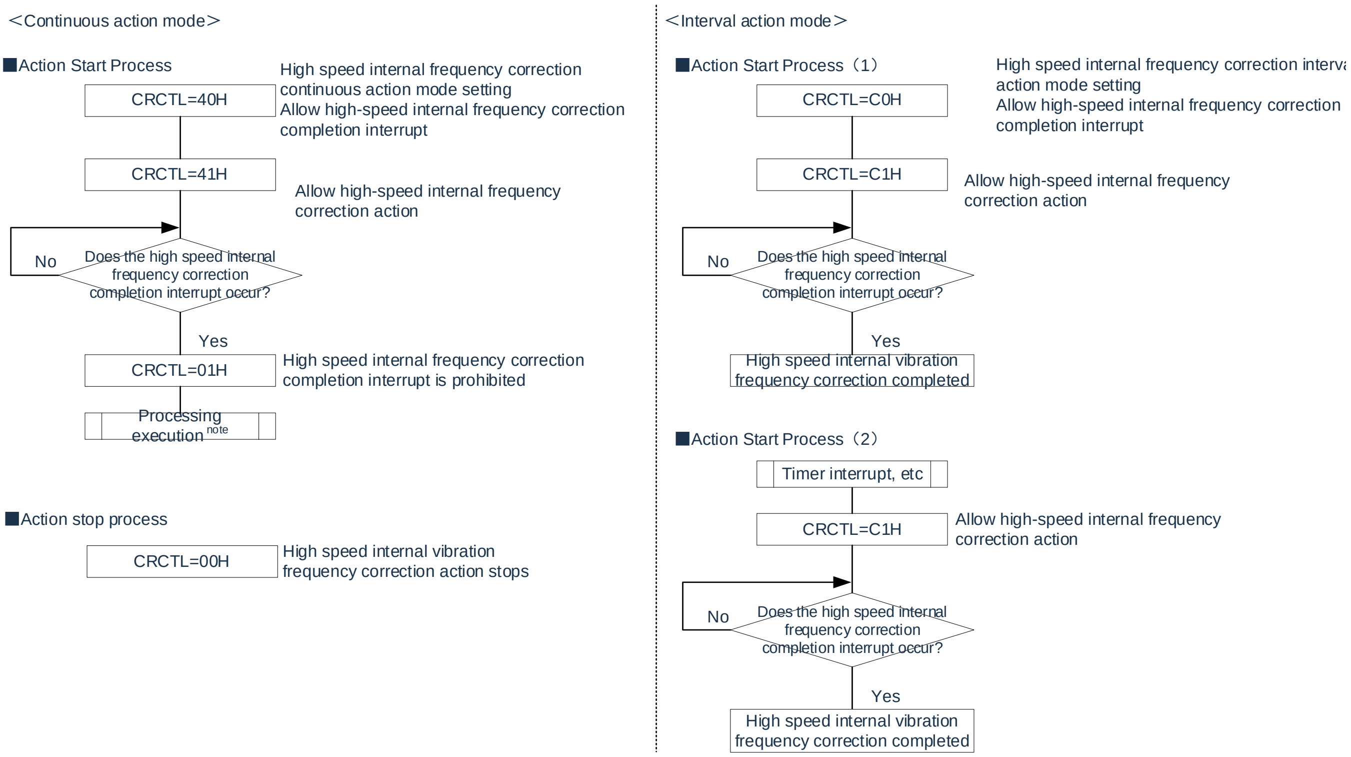
Figure 4-22 Interval operation mode timing diagram


Reference clock (fsub/2^9)
CRMD (Action mode bit)
Continuous action mode "1"
CRST (Action enabling position)
CRST clear: hardware clear when the correction value does not change.
20 bit count register
Count value retention
Correction value [6:0]
+ 1
+ 1
unchanged
"0000000B"
"0000001B"
"0000010B"
"0000010B"
High speed internal frequency correction completion interrupt
Interrupt generation: when CRIE bit is 1, fHOCO pulse with 1 cycle width will be output when correction is completed.

4.7.3.2 Operation setup flow

The operation start/stop flow when the high-speed internal oscillator clock frequency correction function is used is shown in the following figure.

Figure 4-23 Operation mode setting process (example)


＜Continuous action mode＞
■Action Start Process
CRCTL=40H
High speed internal frequency correction continuous action mode setting
Allow high-speed internal frequency correction completion interrupt
CRCTL=41H
Allow high-speed internal frequency correction action
No
Does the high speed internal frequency correction completion interrupt occur?
Yes
CRCTL=01H
High speed internal frequency correction completion interrupt is prohibited
Processing execution note
■Action stop process
CRCTL=00H
High speed internal vibration frequency correction action stops
＜Interval action mode＞
■Action Start Process（1）
CRCTL=C0H
High speed internal frequency correction interva action mode setting
Allow high-speed internal frequency correction completion interrupt
CRCTL=C1H
Allow high-speed internal frequency correction action
No
Does the high speed internal frequency correction completion interrupt occur?
Yes
High speed internal vibration frequency correction completed
■Action Start Process（2）
Timer interrupt, etc
CRCTL=C1H
Allow high-speed internal frequency correction action
No
Does the high speed internal frequency correction completion interrupt occur?
Yes
High speed internal vibration frequency correction completed

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

4.7.4 Precautions for use

4.7.4.1 SFR Access

Regarding the control of the FCST bit in the interval Operation mode, when writing 1 to the FCST bit, you must first confirm that the value of the current FCST bit is 0 before writing 1 to it. Since hardware clears priority, when writing 1 to the FCST bit immediately after the interval operation is completed (when the interrupt generation is completed by high-speed internal resonance frequency correction), the operation should be performed at least 1 cycle after the interrupt generation is completed after the high-speed internal resonance frequency correction is completed.

4.7.4.2 Operation on reset

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

Chapter 5 Universal Timer Unit (Timer4)

This product is equipped with two universal timer units, each containing 4 channels.

Note:

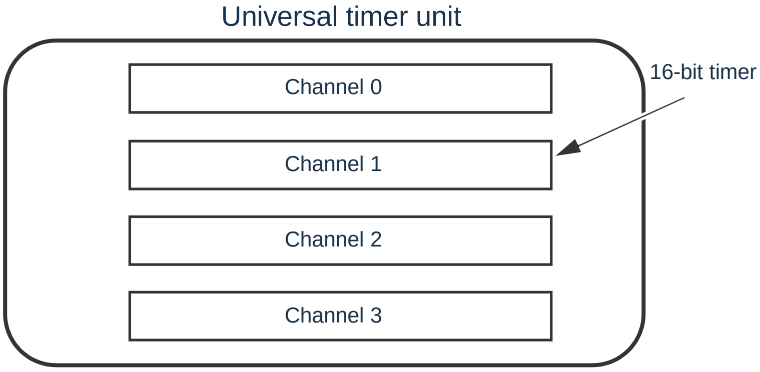
1. The label "m" in the following part of this chapter represents the unit number, this product is equipped with two universal timers Timer4, so m=0,1.

2. The label "n" in the following chapter represents the channel number (n=0~3 in this chapter).

3. The following contents of this chapter are mainly for 48-pin products.

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

Each 16-bit timer, called a "channel", can be used as a separate timer or as a combination of multiple channels as an advanced timer function.



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

Independent channel operation function	Multi-channel linkage operation function
• Interval timer (→refer to 5.8.1). • Square wave output (→refer to 5.8.1). • External event counter (→refer to 5.8.2). • Crossover (→refer to 5.8.3). • Measurement of input pulse intervals (→refer to 5.8.4). • Measurement of the high and low level widths of the input signal (→refer to 5.8.5). • Delay counter (→refer to 5.8.6).	• Single-trigger pulse output (→refer to 5.9.1). • PWM output (→refer to 5.9.2). • Multiple PWM outputs (→refer to 5.9.3).

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

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

5.1 Function of universal timer unit

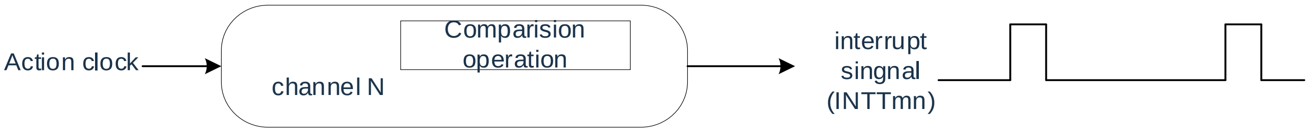
The universal timer unit has the following functions:

5.1.1 Independent channel operation function

The independent channel operation function is a function that can use any channel independently regardless of the operation mode of other channels.

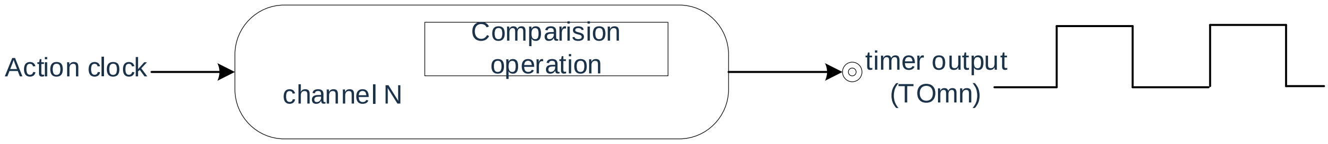
(1) Interval timer

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



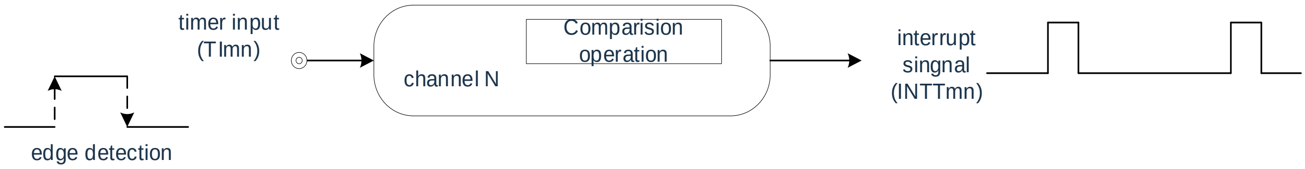
(2) Square wave output

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



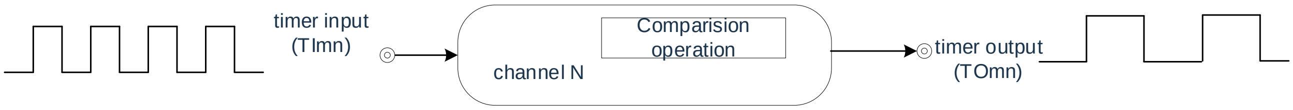
(3) External event counters

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



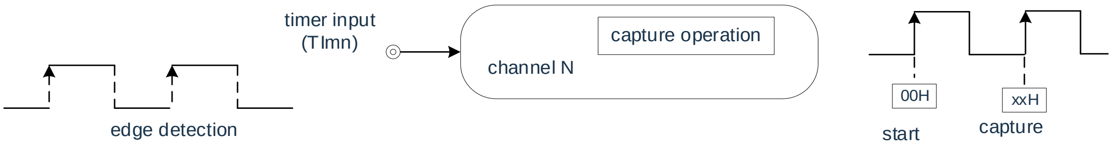
(4) Frequency division function (channel 0 of unit 0 only).

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



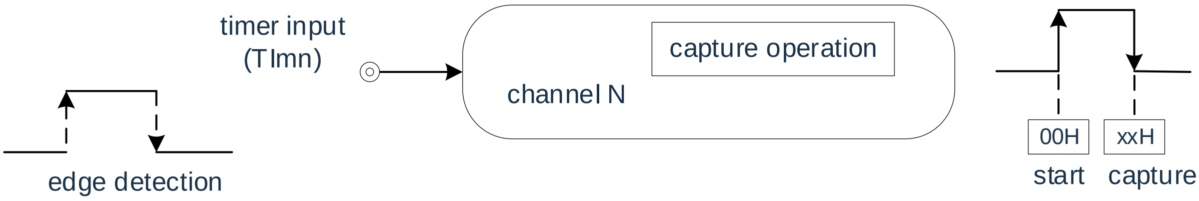
(5) Measurement of input pulse intervals

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



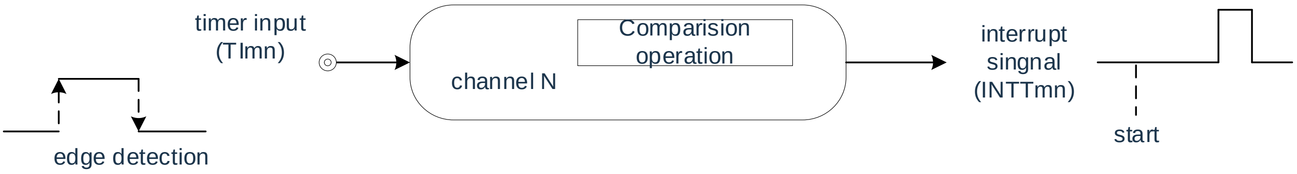
(6) Measurement of the high and low level width of the input signal

The high or low width of the input signal is measured by counting on one edge of the input signal at the timer input pin (TImn) and the count value is captured on the other edge.



(7) Latency counter

The effective edge of the input signal at the timer input pin (TImn) begins to count and an interrupt occurs after any delay period has elapsed.



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

2. For the timer input/output pins of channels 0 to 3 to be configurable, please refer to “Chapter 2 Pin Functions”.

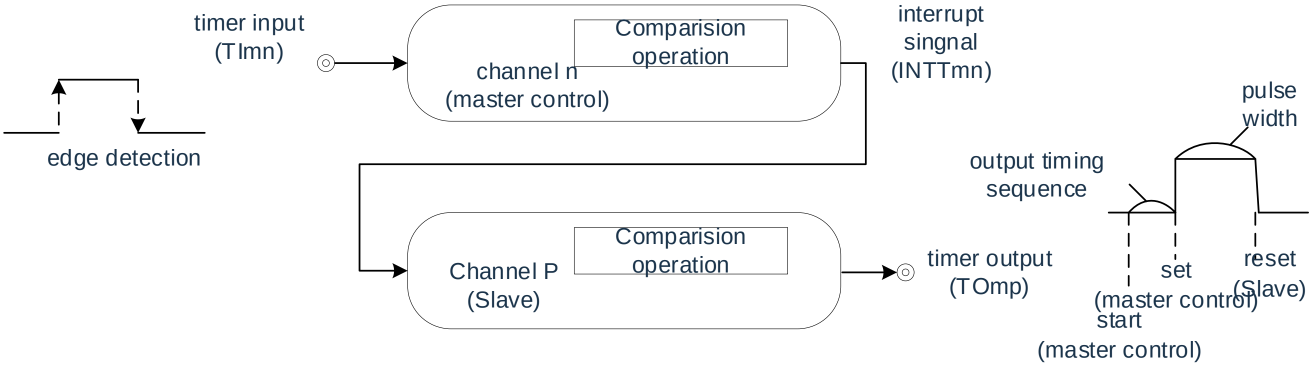
5.1.2 Multi-channel linkage operation function

The multi-channel linkage operation function is a combination of the master channel (the reference timer for the main control period) and the slave channel (the timer that operates in accordance with the master channel).

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

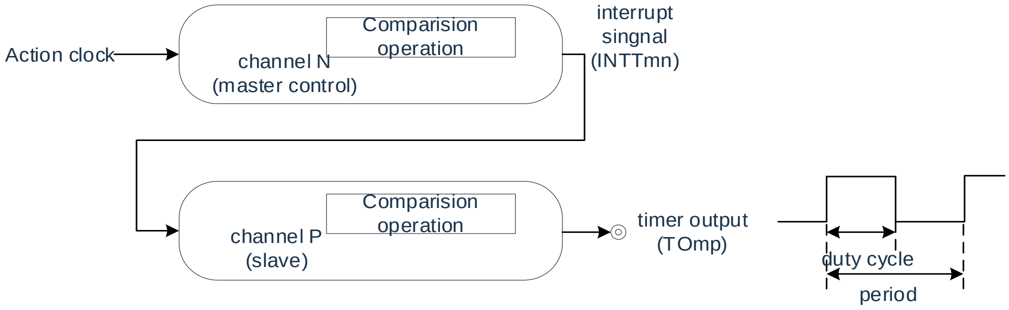
(1) Single trigger pulse output

Use 2 channels in pairs to generate a single-trigger pulse that arbitrarily sets the output timing and pulse width.



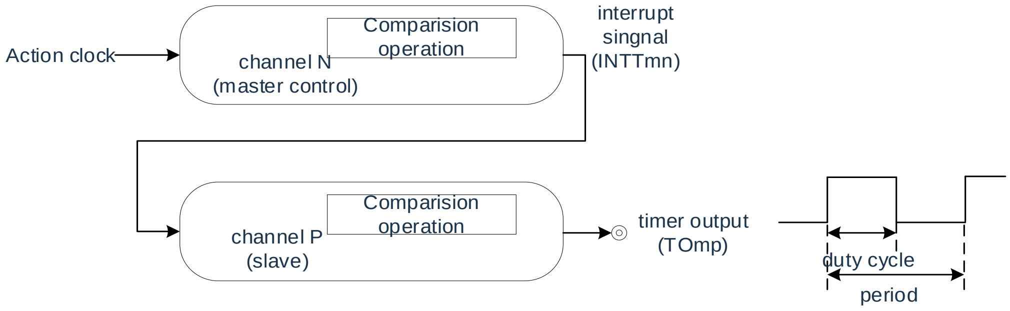
(2) PWM (Pulse Width Modulation) output

Use 2 channels in pairs to generate pulses that can set the period and duty cycle arbitrarily.



(3) Multiple PWM (Pulse Width Modulation) outputs

Up to 3 arbitrary duty cycle PWM signals can be generated in a fixed period by extending the PWM function and using 1 master channel and multiple slave channels.



Note For more information about the multi-channel linkage operation function rules, please refer to it “5.4.1 Basic rules of the multi-channel linkage operation function”.

Remark m: Unit number (m=0,1) n: Channel number (n=0 ~ 3) p, q: The slave channel number(n < p < q ≤ 3)

5.1.3 8-bit timer operation function (limited to Channel 1 and Channel 3 of Unit 0).

The 8-bit timer run function is the ability to use the 16-bit timer channel as two 8-bit timer channels. Only Channel 1 and Channel 3 can be used.

Note There are several rules when using the 8-bit timer to run the feature.

For details, please refer to “5.4.2 Basic Rules for 8-Bit Timer Operation Functions (Channel 1 and Channel 3 Only)”.

5.1.4 LIN-bus support functions (channel 3 of unit 0 only).

Check that the received signal in the LIN-bus communication is suitable for the LIN-bus communication form with the universal timer unit

(1) Detection of wake-up signals

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

(2) Detection of break filed

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

(3) Measurement of synchronous field pulse width

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

Note For operational settings for LIN-bus support functions, refer to “5.3.13 Input Switching Control Register (ISC)” and “5.8.5 Operation as Input Signal High and Low Width Measurements”.

5.2 Structure of the universal timer unit

The universal timer unit consists of the following hardware.

Table 5-1 Structure of the universal timer unit

Item	Structure
counter	Timer count register mn (TCRmn).
register	Timer data register mn (TDRmn).
The input to the timer	TI00~TI03 Note 1, TI10~TI13 Note 1
The output of the timer	TO00~TO03Note 1, TO 1 0~TO13Note 1, Output control circuitry
Control registers	< Unit setting register > • Peripheral enable register 0 (PER0). • Timer clock selection register m (TPSm). • Timer channel enable status register m (TEm). • Timer channel start register m (TSm). • Timer channel stop register m (TTm). • Timer input selects register 0 (TIOS0) Note 2 • Timer output enable register m (TOEm). • Timer output register m (TOm). • Timer output level register m (TOLm). • Timer output mode register m (TOMm).
	< Each channel register > • Timer mode register mn (TMRmn). • Timer status register mn (TSRmn). • Noise filter enable registers 1, 2 (NFEN1, NFEN2). • Port mode control register (PMCxx) Note 3 • Port mode register (PMxx) Note 3 • Port output multiplexing function configuration register (PxxCFG) Note 3 • Port input multiplexing function configuration register (TI1XPCFG) Note 3

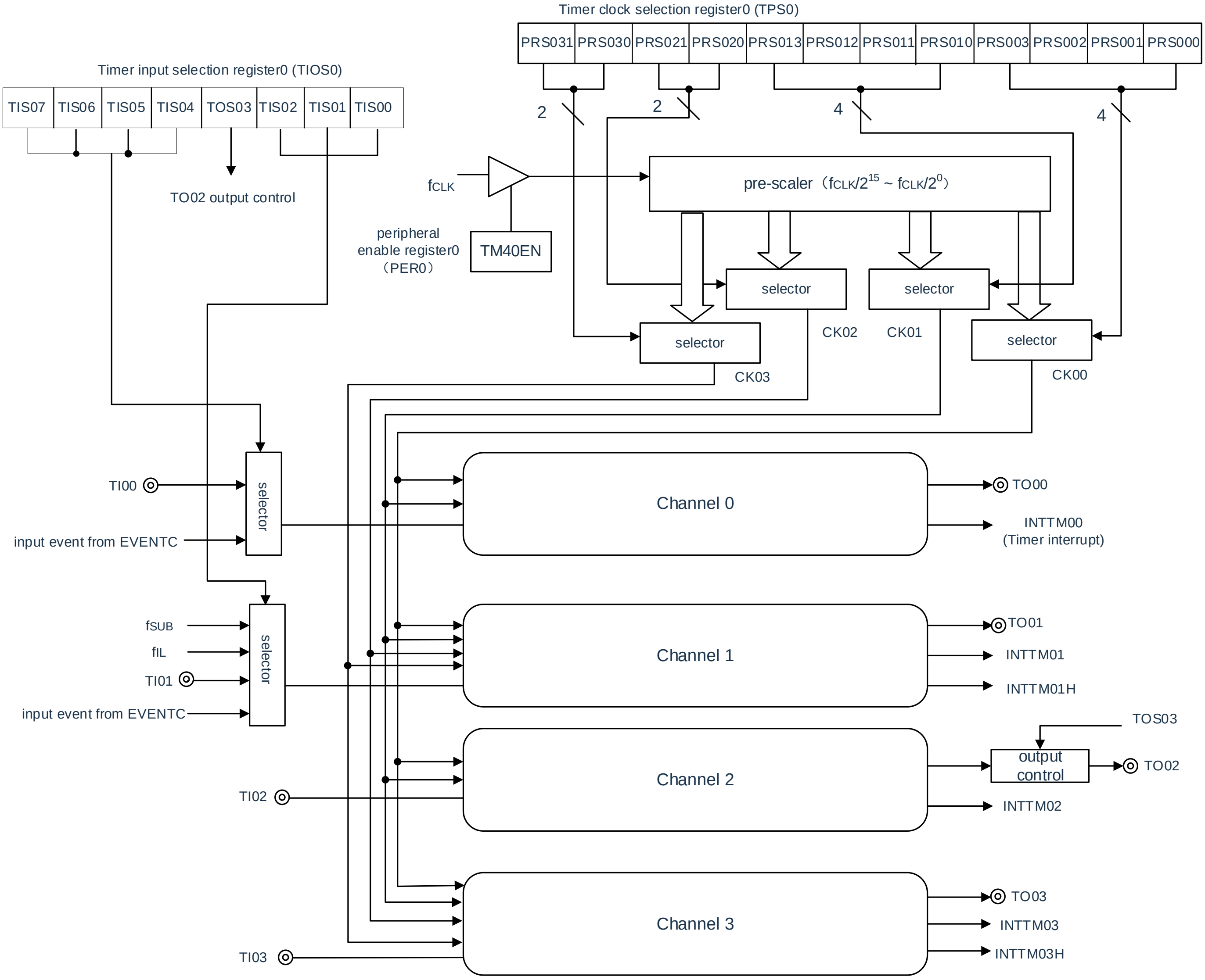
Note 1: The input/output pins of general-purpose timer unit 0 are multiplexed to fixed ports, and the timer input/output pins of channels 0 to 3 of general-purpose timer unit 1 can be arbitrarily configured to each port except RESETB. For details, please refer to “Chapter 2 Pin Functions”.

Note 2: Channel selection for unit 0 only

Note 3: Timer input/output pin configuration for channels 0 to 3. For details, please refer to “Chapter 2 Pin Functions”.
Remark m: unit number (m=0,1) n: channel number (n=0~ 3).

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

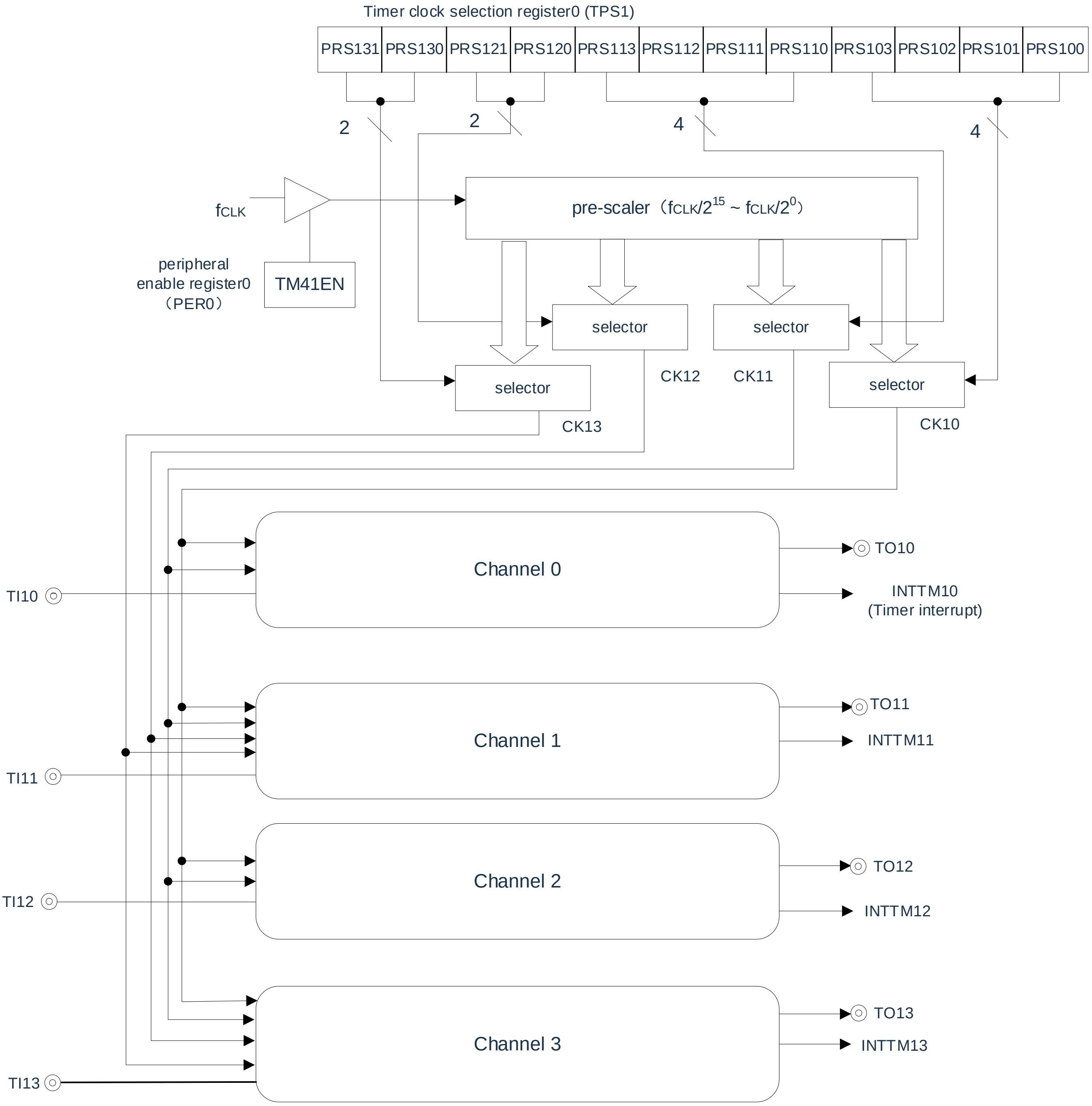
Figure 5-1 Block diagram of universal timer unit 0



Note f_{SUB} : Subsystem clock frequency

f_{IL} : Low speed internal oscillator clock frequency

Figure 5-2 Block diagram of universal timer unit 1



5.2.1 List of universal timer unit 0 registers

Register base address for unit 0: 0x40041C00

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

5.2.2 List of universal timer unit 1 registers

Register base address for unit 1: 0x400420 00

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

5.2.3 Timer count register mn (TCRmn)

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

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

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

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

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

The count value can be read by reading the timer count register mn (TCRmn).
In the following cases, the count value becomes “FFFFH”.

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

In the following cases, the count value becomes “0000H”.

- Enter Start when triggering in snap mode
- At the end of the capture in snap mode

Note that even if the TCRmn register is read0, the count value is not captured to the timer data register mn (TDRmn).

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

Table 5-2 Read values of the timer count register mn (TCRmn) in each operating mode

Run mode	Counting mode	Timer count register mn (TCRmn) read value note			
		The value when the operating mode is changed after the reset is released	Count pause (TTmn=1) value	Count pause (TTmn=1) after changing the value in run mode	Wait after single count The value at which the trigger begins
Interval timer mode	Decrement count	FFFFH	The value at stop	Indefinite value	—
Capture mode	Increment the count	0000H	The value at stop	Indefinite value	—
Event counter mode	Decrement count	FFFFH	The value at stop	Indefinite value	—
Single count mode	Decrement count	FFFFH	The value at stop	Indefinite value	FFFFH
Capture & Single Count Mode	Increment the count	0000H	The value at stop	Indefinite value	The capture value of the TDRmn register is +1

Note Indicates the read value of the TCRmn register when channel n is in the timer run stop state (TEmn=0) and the count enable state (TSmn=1). Keep this value in the TCRmn register until the count begins.

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

5.2.4 Timer data register mn (TDRmn)

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

TDRmn registers can be rewritten at any time.

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

The SPLIT bit in 8-bit timer mode (timer mode registers m1, m3 (TMRm1, TMRm3) is “1”), can read and write TDRm1 registers and TDRm3 registers in 8-bit units, where TDRm1H and TDRm3H Used as high 8 bits, 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 5-4 Format of timer data register mn(TDRmn) (n=0, 2)

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

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

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

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

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

The count is decremented from the config value of the TDRmn register, and when the count value becomes “0000H”, an interrupt signal (INTTMmn) is generated. The value of the TDRmn register is held until it is overwritten.

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

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

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

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

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

5.3 Registers for controlling general-purpose timer unit

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

- Peripheral enable register 0 (PER0).
- Timer clock selection register m (TPSm).
- Timer mode register mn (TMRmn).
- Timer status register mn (TSRmn).
- Timer channel enable status register m (TEm).
- Timer channel start register m (TSm).
- Timer channel stop register m (TTm).
- Timer Input and Output 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).
- Noise filter enable register 1 (NFEN1).
- Noise filter enable register 2 (NFEN2).
- Port Mode Control Register (PMCxx).
- Port Mode Register (PMxx).
- Port multiplexing function configuration register (PxxCFG).

Note The allocated registers and bits vary from product to product.

Unassigned bits must be initialized.

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

5.3.1 Peripheral enable register 0 (PER0)

The PER0 register is a register that sets the clock to be enabled or disenabled to be supplied to each peripheral hardware. Reduce power consumption and noise by stopping clocking unused hardware.

To use universal timer unit 0, bit0 (TM40EN) must be set to “1”. The PER0 register is set via an 8-bit memory operation command. After the reset signal is generated, the value of the PER0 register changes to “00H”.

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

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

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

TM40EN	Control of the input clock of the universal timer unit 0
0	Stop supplying the input clock. • Cannot write the SFR used by General Timer Unit 0. • General purpose timer unit 0 is in the reset state.
1	An input clock is provided. • Reads and writes SFR used in general-purpose timer unit 0.

TM41IN	Control of the input clock of the universal timer unit 1
0	Stop supplying the input clock. • Cannot write SFR used by universal timer unit 1. • General purpose timer unit 1 is in a reset state.
1	An input clock is provided. • SFR used in universal timer unit 1 can be read and written.

Note 1 To set the general-purpose timer unit, the following registers must first be set in the TM4mEN bit “1”. When the TM4mEN bit is “0”, the value of the control register of the timer array unit is the initial value, ignoring the write operation (timer input and output selection register 0 (TIOS0), noise filter enable register 1 (NFEN1), noise filter enable register 2 (NFEN2), port mode control register PMCx, Port mode register PMx and port multiplexing function configuration register PxxCFG).

- Timer status register mn (TSRmn).
- Timer channel enable status register m (TEm).
- Timer channel start register m (TSm).
- Timer channel stop register m (TTm).
- Timer output enable register m (TOEm).
- Timer output register m (TOm).
- Timer output level register m (TOLm).
- Timer output mode register m (TOMm).

5.3.2 Timer clock select register m (TPSm)

TPSm register is a 16-bit register that selects 2 or 4 common operating clocks (CKm0, CKm1, CKm2) available to each channel, CKm3). CKm0 is selected by bits 3 to 0 of the TPSm register, and CKm1 is selected by bits 7 to 4 of the TPSm register. In addition, only channel 1 and channel 3 can select CKm2 and CKm3. CKm2 is selected through bits 9 to 8 of the TPSm register, and CKm3 is selected through bits 13 and 12 of the TPSm register.

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

In the case where PRSm00~PRSm03 bits can be rewritten (n=0~3):

Select CKm0 as the running clock (CKSmn1, CKSmn0=0, 0) for all channels in the stopped state (TEmn=0).

In the case where PRSm10~PRSm13 bits can be rewritten (n=0~3):

Select CKm2 as the running clock (CKSmn1, CKSmn0=0, 1) for all channels in the stopped state (TEmn=0).

Can override the PRSm 20 bits and PRSm 21 bits (n=1, 3):

Select CKm1 as the channel running clock (CKSmn1, CKSmn0=1, 0) all in the stopped state (TEmn=0).

Can override PRSm 30 bits and PRSm 31 bits (n=1, 3):

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

The TPSm register is set via 16-bit memory operation instructions. After the reset signal is generated, the value of the TPSm register changes to “0000H”.

Figure 5-7 Format of the timer clock selection register m (TPSm)(1/2)

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

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

Note that in case of changing the clock selected as f_{CLK} (changing the value of the system clock control register (CKC)), the general-purpose timer unit (TTm=000FH) must be stopped. Even when selecting the operating clock (f_{MCK}) or the active edge of the input signal at the TImn pin, the general-purpose timer unit needs to be stopped.

Note 1 You must set bit15, 14, 11, and 10 to “0”.

2. If you select f_{CLK} (divided) as the running clock (CKmk) and set TDRmn to “0000H” (m=0 , 1, n = 0 ~ 3), you can not use the universal timer unit interrupt request.

Note 1. f_{CLK}: The clock frequency of the CPU/peripheral hardware.

2. The TPSm register selects a clock waveform that is only 1 fCLK cycle high from the rising edge. For details, please refer to “5.5.1 Counting Clock (fTCLK)”.

Figure 5-8 Format of the timer clock selection register m (TPSm)(2/2)

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

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

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

Note that in case of changing the clock selected as f_{CLK} (changing the value of the system clock control register (CKC)), the general-purpose timer unit (TTm=000FH) must be stopped. Even when selecting the operating clock (f_{MCK}) or the active edge of the input signal at the TImn pin, the general-purpose timer unit needs to be stopped.

Note: You must set bit15, 14, 11, and 10 to "0".

If you use channels 1 and 3 in 8-bit timer mode and use CKm2 and CKm3 as the operating clocks, the interval times shown in the following table can be achieved through the interval timer function.

Table 5-3 Configurable interval for running clocks CKSm2 and CKSm3

Clock		Interval Note (f_{CLK}=32MHz)			
		10us	100us	1ms	10ms
CKm2	$f_{CLK}/2$	○	—	—	—
	$f_{CLK}/2^{2}$	○	—	—	—
	$f_{CLK}/2^{4}$	○	○	—	—
	$f_{CLK}/2^{6}$	○	○	—	—
CKm3	$f_{CLK}/2^{8}$	—	○	○	—
	$f_{CLK}/2^{10}$	—	○	○	—
	$f_{CLK}/2^{12}$	—	—	○	○
	$f_{CLK}/2^{14}$	—	—	○	○

Note: ○ contains an error of less than 5%.

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

2. For details of the $f_{CLK}/2^{r}$ waveform selected for the TPSm register, refer to "5.5.1 Counting Clock(f_{TCLK})".

5.3.3 Timer mode register mn (TMRmn)

The TMRmn register is the register for setting the operation mode of channel n. It carries out the selection of the operation clock (f_{MCK}), the selection of the count clock, the selection of the master/slave, the selection of the 16-bit/8-bit timer (channel 1 and channel 3 of unit 0 only), the setting of the start trigger and the capture trigger, the selection of the valid edge of the timer input and the setting of the operation mode (interval, capture, event counter, single count, capture & single count).

It is forbidden to override the TMRmn register in operation (TEmn=1). However, bit7 and bit6 (CISmn1, CISmn0) can be overridden in some functions (TEmn=1) (for details, please refer to “ 5.8 Independent Channel Operation Function of General Timer Unit” and “Multi-channel Linkage Operation Function of 5.9 Timer Array Unit”).

The TMRmn register is set via a 16-bit memory operation command. After the reset signal is generated, the value of the TMRmn register changes to “0000H”.

Note: Bit11 of the TMRmn register varies from channel to channel.
TMRm2: MASTERmn bit (n=2)
TMRm1, TMRm3: SPLITmn bit (n=1, 3)
TMRm0: fixed as “0”.

Figure 5-9 Table of timer mode register mns (TMRmn) (1/4)

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

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

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

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

CCSmn	Channel n count clock (f_{TCLK}) selection
0	The CKSmn0 bit and CKSmn1 bit specify the running clock (f_{MCK}).
1	The active edge of the TImn pin input signal • Unit 0 in the case: Channel 0: The effective edge of the input signal selected by TIOS0 Channel 1: The effective edge of the input signal selected by TIOS0
The count clock (f_{TCLK}) is used for counters, output control circuitry, and interrupt control circuitry.	

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

Note 1 You must set bit13, 5, and 4 to “0”.

2. When you want to change the clock selected as f_{CLK} (change the value of the system clock control register (CKC)), even if the CKSmn0 bit and the CKSmn1 bit specify the running clock (f) is selected $_{MCK}$) or the active edge of the input signal at the TImn pin as the count clock (f_{TCLK}), the timer array unit must also be stopped TTm=00FFH).

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

Figure 5-10 Table of timer mode register mns (TMRmn) (2/4)

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

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

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

(bit11 of TMRmn(n=2))

MASTERmn	Choice of independent channel operation / multi-channel linkage operation (slave or master) of channel n
0	It is used as a slave channel for independent channel operation function or multi-channel linkage operation function.
1	It is used as the main control channel for multi-channel linkage operation function.
Only channel 2 can be set as the main control channel (MASTERmn=1). Channel 0 is fixed to “0” (because channel 0 is the channel at the highest bit, it is independent of the setting of this bit and is used as the master channel). For channels used as stand-alone channel operating functions, place the MASTERmn position “0”.	

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

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

STSmn2	STSmn1	STSmn0	Setting of the start trigger and the capture trigger of channel n
0	0	0	Only software triggers start to be valid (no other trigger source is selected).
0	0	1	Use the active edge of the TImn pin input for start trigger and snap trigger.
0	1	0	Use the bilateral edges of the TImn pin inputs for start triggering and snap triggering, respectively.
1	0	0	The interrupt signal of the master channel is used (in the case of slave channels with multi-channel linkage operation function).
Beyond the above			Disable settings.

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).

Figure 5-11 Table of timer mode register mns (TMRmn) (3/4)

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

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

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

CISmn1	CISmn0	Valid edge selection for the TImn pin
0	0	Falling edge
0	1	Rising edge
1	0	Bilateral edges (when measuring low widths) Start trigger: falling edge, snap trigger: rising edge
1	1	Bilateral edges (when measuring high widths) Start trigger: rising edge, snap trigger: falling edge
When the STSmn2~STSmn0 bits are not “010B” and are specified using a bilateral edge, the CISmn1~CISmn0 positions must be “10B”.		

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).

Figure 5-12 Table of timer mode register mns (TMRmn) (4/4)

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

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

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

MD mn3	MD mn2	MD mn1	The setting of the channel n operating mode	Corresponding functions	The count of TCR runs
	0	0	Interval timer mode	Interval timer/square wave output/ Crossover function / PWM output (master).	Decrement count
	1	0	Capture mode	Measurement of input pulse intervals	Increment the count
	1	1	Event counter mode	External event counters	Decrement count
	0	0	Single count mode	Delay counter/single trigger pulse output/PWM output (Subgenerated)	Decrement count
	1	0	Capture & Single Count Mode	Measurement of the high and low level width of the input signal	Increment the count
	Others		Settings are forbidden		
The operation of each mode varies depending on the MDmn0 bit (refer to the table below).					

Operating mode (MDmn3~MDmn1 bit setting (see table above)).	MD mn0	Start counting and interrupt settings
• Interval timer mode (0, 0, 0) • Capture mode (0, 1, 0).	0	No timer interrupt occurs at the start of the count (nor does the output of the timer change).
	1	A timer interrupt is generated at the start of the count (the output of the timer also changes).
• Event counter patterns (0, 1, 1).	0	No timer interrupt occurs at the start of the count (nor does the output of the timer change).
• Single count mode[Note 2] (1, 0, 0)	0	The start trigger in the count run is invalid. No interruption occurs at this time.
	1	The start of the count run triggers a valid [Note 3]. No interruption occurs at this time.
• Capture & Single Count Mode (1, 1, 0).	0	No timer interrupt occurs at the start of the count (nor does the output of the timer change). The start trigger in the count run is invalid. 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, the interrupt output (INTTMmn) and TOmn output at start count are not controlled.

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

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

5.3.4 Timer status register mn (TSRmn)

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

The TSRmn register is only valid in capture mode (MDmn3 to MDmn1=010B) and capture & single count mode (MDmn3 to MDmn1=110B). Refer to Table 5-4 for the change of OVF bits and the set/clear conditions in each operation mode.

The TSRmn register is read through 16-bit memory operation instructions.

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

Figure 5-13 Table of timer status register mn (TSRmn)

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

OVF	Counter overflow status of channel n
0	No overflow occurred.
1	Overflow occurs.
If the OVF bit is “1”, this flag is cleared the next time the count does not overflow and the count value is snapped (OVF=0).	

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

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

Timer operating mode	OVF bit	Set/clear conditions
• Capture mode • Capture & Single Count mode	Clear	No overflow occurred at the time of capture
	Set	An overflow occurs during capture
• Interval timer mode • Event counter mode • Single count mode	Clear	— (Cannot be used)
	Set	

Remark: Even if the counter overflows, the OVF bit does not change immediately and changes at capture thereafter.

5.3.5 Timer channel enable status register m (TEm)

TEm registers are registers that represent the enabled or stopped state of each channel timer operation.

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

The TEm register is read via 16-bit memory operation instructions.

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

Figure 5-14 Timer Channel Enable Status Registers m (TEm)

Symbol	15 0	14	13	12	11	10	9	8	7	6	5	4	3	2	1	
Has	0	0	0	0	TEH m3	0	TEH m1	0	0	0	0	0	TEm3	Has2	TEm1	Has0

TEHm3	Channel 3 is a representation of the operating enable or stop state of the high 8-bit timer in 8-bit timer mode
0	Running stop state
1	Run enabled status

TEHm1	Channel 1 is a representation of the operating enable or stop status of the high 8-bit timer in 8-bit timer mode
0	Running stop state
1	Run enabled status

TEmn	A representation of the running enabled or stopped status of channel n
0	Running stop state
1	Run enabled status
When channels 1 and 3 are in 8-bit timer mode, TEm1 and TEm3 indicate the operating enable or stop state of the low-8-bit timer.	

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

5.3.6 Timer channel start register m (TSm).

The TSm register is a trigger register that initializes the timer count register mn (TCRmn) and sets the start of each channel count operation. If each position is "1", the timer channel allows the corresponding bit of the status register m (TEm) to be set to "1". Because the TSmn bit, TSHm1 bit, and TSHm3 bit are the trigger bits, if they become operational enable (TEmn, TEHm1, TEHm3=1), the TSmn, TSHm1, and TSHm3 bits are immediately cleared.

The TSm register is set via 16-bit memory operation instructions.

The lower 8 bits of the TSm register can be set with TSmL and via 8-bit memory operation instructions. After the reset signal is generated, the value of the TSm register changes to "0000H".

Figure 5-15 Table of Timer Channel Start Register m (TSm)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1`	0
TSm	0	0	0	0	TSHm3	0	TSHm1	0	0	0	0	0	TSm3	TSm2	TSm1	TSm0

TSHm3	The operation of the high 8-bit timer in channel 3 for 8-bit timer mode enable (start) to trigger
0	No triggering.
1	Place the TEHm3 position "1" into the count allow state. If the counting of TCRm3 registers is started while counting is allowed, interval timer mode is entered (refer to Table 5-5 of "Start Timing of 5.5.2 Counters").

TSHm1	The operation of the high 8-bit timer in channel 1 for 8-bit timer mode enable (start) to trigger
0	No triggering.
1	Place the TEHm1 position "1" into the count allow state. If the counting of TCRm1 registers is started while counting is allowed, interval timer mode is entered (refer to Table 5-5 of "Start Timing of 5.5.2 Counters").

TSmn	The operation of channel n enable (start) to trigger
0	No triggering.
1	Place the TEmn position "1" into the count allow state. The start of counting of TCRmn registers in the enable state of counting varies by mode of operation (refer to Table 5- of "Start Timing of 5.5.2 Counters" 5). When channels 1 and 3 are in 8-bit timer mode, TSm1 and TSm3 are 8 low The operation of the bit timer enable (start) to trigger.

Note 1 You must set bit15~12, 10, 8~4 to "0".

2. When switching from the function of not using the TImn pin input to the function of using the TImn pin input, from setting the timer mode register mn (TMRmn) to the TSmn (TSHm1, TSHm3) position "1" until the following period of wait:

When the TImn pin noise filter is active (TNFENmn=1): 4 operating clocks (f_{MCK}).

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

Note: 1. The read value of the TSm register is always "0".

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

5.3.7 Timer channel stop register m (TTm)

The TTm register is the trigger register that sets the count stop for each channel.

If each position is “1”, the timer channel allows the corresponding bit of the status register m (TEm) to be cleared “0”. Because the TTmn bit, TTHm1 bit, and TTHm3 bit are the trigger bits, if they become run stop states (TEmn, TEHm1, TEHm3=0), the TTmn bit, the TTHm1 bit, and the TTHm3 bit are immediately cleared.

The TTm register is set via a 16-bit memory operation command.

The low 8 bits of the TTm register can be set with TTmL and via 8-bit memory operation instructions. After the reset signal is generated, the value of the TTm register changes to “0000H”.

Figure 5-16 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	TTHm 3	0	TtHm 1	0	0	0	0	0	TTm 3	TTm 2	TTm 1	TTm 0

TTHm3	Channel 3 triggers the operation stop of the high 8-bit timer in 8-bit timer mode
0	No triggering.
1	Clear the TEHm3 bit “0” and enter the counting stop state.

TTHm1	Channel 1 triggers the operation stop of the high 8-bit timer in 8-bit timer mode
0	No triggering.
1	Clear the TEHm1 bit “0” and enter the counting stop state.

TTmn	The operation of channel n is stopped triggered
0	No triggering.
1	Clear the TEmn bit to “0” and enter the count stop state. When channels 1 and 3 are in 8-bit timer mode, TTm1 and TTm3 are the operation stop triggers for the low 8-bit timer.

Note You must set bit15~12, 10, 8~4 to “0”.

Note: 1. The read value of the TTm register is always “0”.
2. m: unit number (m=0,1) n: channel number (n=0~3).

5.3.8 Timer input-output select register (TIOS0)

The TIOS0 register is used to select the input and output of unit 0. Select the timer inputs for Channel 0 and Channel 1 of Unit 0 and the timer output for Channel 2. The TIOS0 register is set via an 8-bit memory operation command. After the reset signal is generated, the value of the TIOS0 register changes to “00H”.

Figure 5-17 Table of timer input selection register 0 (TIOS0)

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

Symbol	7	6	5	4	3	2	1	0
TIOS0	TUES07	TUES06	TUES05	TUES04	TOS03	YOUWITH02	TIS01	TIS00

TIS07	TIS06	TIS05	Channel 0 uses the selection of timer inputs
0	0	0	The input signal of the timer input pin (TI00).
others			Settings are forbidden

TIS04	Channel 0 uses the selection of timer inputs
0	Input signal selected via TIS07~TIS05
1	The event input signal for ELC

TOS03	Enable of the timer output of channel 2
0	Enable output
1	Disable output (output fixed at 0).

TIS02	TIS01	TIS00	Channel 1 uses the selection of timer inputs
0	0	0	The input signal of the timer input pin (TI01).
0	0	1	The event input signal for ELC
0	1	0	The input signal of the timer input pin (TI01).
0	1	1	
1	0	0	Low-speed internal oscillator clock (f_{IL}).
1	0	1	Subsystem clock (f_{SUB}).
Others			Settings are forbidden

Note 1 The high or low level width of the selected timer input needs to be greater than or equal to 1/fMCK+10ns. Therefore, when selecting fSUB as fCLK (CSS=1 for CKC registers), TIS02 position “1” cannot be used.

2. When selecting the event input signal of ELC by timer input Register 0 (TIOS0), it must be selected by timer clock selection register 0 (TPS0). fCLK.

5.3.9 Timer output enable register m (TOEm)

The TOEm register is a register that sets the enable or disallow timer outputs for each channel.

For channel n that enable timer output, the value of the TOmn bit of the timer output register m (TOm) described later cannot be rewritten by software, and the value reflected by the timer output function of the counting operation is from the timer's output pin (TOmn) output.

The TOEm register is set via a 16-bit memory operation command.

The lower 8 bits of the TOEm register can be set with TOEmL and via 8-bit memory operation instructions. After the reset signal is generated, the value of the TOEm register changes to “0000H”.

Figure 5-18 Table of timer output enable register m (TOEm)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOEm	0	0	0	0	0	0	0	0	0	0	0	0	TOEm3	TOEm2	TOEm1	TOEm0

TOEmn	Enable/disable the timer output of channel n
0	Disable timer output. The operation of the timer is not reflected to the TOmn bit, fixed output. TOmn bits can be written and TOmn bits are output from the TOmn pin.
1	Enable timer output. The operation of the timer is reflected to the TOmn bit, resulting in an output waveform. Ignore the write operation of the TOmn bit.

Note: bit15~4 must be set to “0”.
Remark: m: unit number (m=0,1) n: channel number (n=0~3).

5.3.10 Timer output register m (TOm).

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

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

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

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

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

The lower 8 bits of the TOm register can be set with TOmL and via 8-bit memory operation instructions. After the reset signal is generated, the value of the TOm register changes to "0000H".

Figure 5-19 Table of timer output register m (TOm)

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

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

Note: bit15~4 must be placed to "0".

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

5.3.11 Timer output level register m (TOLm)

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

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

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

The lower 8 bits of the TOLm register can be set with TOLmL and via 8-bit memory operation instructions. After the reset signal is generated, the value of the TOLm register changes to “0000H”.

Figure 5-20 Table of timer output level register m (TOLm)

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

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

Note: bit15 to 4 and bit0 must be set to “0”.

Remark 1: If you override the value of this register while the timer is running, the output logic of the timer is reversed the next time the timer output signal changes, rather than immediately after the override.
2. m: unit number (m=0,1) n: channel number (n=0~3).

5.3.12 Timer output mode register m (TOMm)

The TOMm register is a register that controls the output mode of each channel timer. When used as a standalone channel operation function, the corresponding position of the channel used is “0”.

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

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

The TOMm register is set via a 16-bit memory operation command.

I can set the low 8 bits of the TOMm register with TOMmL and via an 8-bit memory operation command. After the reset signal is generated, the value of the TOMm register changes to “0000H”.

Figure 5-21 Table of timer output mode register m (TOMm)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOMm	0	0	0	0	0	0	0	0	0	0	0	0	TOMm3	TOMm2	TOMm1	0

TOMmn	Control of the timer output mode of channel n
0	The main channel output mode (alternately output via timer interrupt request signal (INTTMmn).
1	Slave channel output mode (the output is asserted via the timer interrupt request signal (INTTMmn) of the master channel and reset by the timer interrupt request signal (INTTMmp) of the slave channel).

Note: bit15 to 4 and bit0 must be set to “0”.

Remark: m: Unit number (m=0,1) n: channel number n=0~3 (for the main channel: n=0，2)
p: The slave channel number
n=0： p=1, 2, 3
n=2： p=3
(For more information on the relationship between master and slave channels, refer to “5 4.1 Basic rules for multi-channel linkage operation function“).

5.3.13 Noise filter enable register 1 (NFEN1)

The NFEN1 register sets whether the noise filter is used for the input signal of the timer input pins of each channel of Unit 0. For pins that need to be noise canceled, the corresponding position “1” must be placed for the noise filter to be effective. When the noise filter is active, detect whether the two clocks are consistent after synchronization through the running clock (fMCK) of the object channel; When the noise filter is invalid, the synchronization is only made through the running clock (fMCK) of the object channel.

The NFEN1 register is set via an 8-bit memory operation command. After the reset signal is generated, the value of the NFEN1 register changes to “00H”.

For details, please refer to “5.5.1(2) Selecting the Valid Edge of the TImn Pin Input Signal (CCSmn=1)” and “5” .5.2 Start Timing of Counters“ and “Control of 5.7 Timer Inputs (TImn)”.

Figure 5-22 Table of Noise Filter Enable Register 1 (NFEN1)

Address: 0x40040471

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

TNFEN03	Whether the input signal noise filter of the TI03 pin is used or not
0	Noise filter OFF
1	Noise filter ON

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

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

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

Note: The configuration of the timer input/output pins of channels 0 to 3 is described in Chapter 2 Pin Functions.

5.3.14 Noise filter enable register 2 (NFEN2)

The NFEN2 register sets whether the noise filter is used for the input signal of the timer input pins of each channel of Element 1. For pins that need to be noise canceled, the corresponding position “1” must be placed for the noise filter to be effective. When the noise filter is active, detect whether the two clocks are consistent after synchronization through the running clock (fMCK) of the object channel; When the noise filter is invalid, the synchronization is only made through the running clock (fMCK) of the object channel.

The NFEN 2 registers are set via 8-bit memory operation instructions. After the reset signal is generated, the value of the NFEN2 register changes to “00H”.

For details, please refer to “5.5.1(2) Selecting the Valid Edge of the TImn Pin Input Signal (CCSmn=1)” and “5” .5.2 Start Timing of Counters“ and “Control of 5.7 Timer Inputs (TImn)”.

Figure 5-23 Table of noise filter enable register 2 (NFEN2)

Address: 0x40040472

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

TNFEN13	TI1 3-pin input signal noise filter is used or not
0	Noise filter OFF
1	Noise filter ON

TNFEN12	TI12 pin input signal noise filter is used or not
0	Noise filter OFF
1	Noise filter ON

TNFEN11	TI1 pin 1 input signal noise filter is used or not
0	Noise filter OFF
1	Noise filter ON

TNFEN10	TI1pin 0 input signal noise filter is used or not
0	Noise filter OFF
1	Noise filter ON

Note: The configuration of the timer input/output pins of channels 0 to 3 is described in Chapter 2 Pin Functions.

5.3.15 Registers for controlling timer input/output pin port functions

When using a general-purpose timer unit, the input/output pins of timer 0 are multiplexed to a fixed port, and the input/output pins of timer 1 can be arbitrarily configured to ports except RESETB. For details, please refer to “Chapter 2 Pin Functions”.

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

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

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

Example: P21 is configured to TO10 as the timer output
Set the PMC21 bit of the port mode control register 2 to “0”.
Set the PM21 bit of the port mode register 2 to “0”.
Set the port output multiplexing function configuration register P21 CFG to “0x01”.

When using the multiplexed port of Timer 1 input pin as timer input, the bit of Port Mode Register (PMxx) corresponding to each port must be set to “1” and the bit of Port Mode Control Register (PMCxx) must be set to “0”. And set the port multiplexing function configuration register (TI10PCFG). At this point, the port register (Pxx) bit can be “0” or “1”.

Example: P20 / TI10 is used as the timer input
Set the PMC20 bit of Port Mode Control Register 2 to “0”.
Set the PM20 bit of Port Mode Register 2 to “1”.
Set the port input multiplexing function configuration register TI10PCFG to “0x0b”.

5.4 Basic rules of the universal timer unit

5.4.1 Basic rules of the multi-channel linkage operation function

The multi-channel linkage operation function is a combination of the master channel (the reference timer that mainly counts cycles) and the slave channel (the timer that obeys the operation of the master channel), and several rules need to be observed when used.

The basic rules of the multi-channel linkage operation function are as follows.

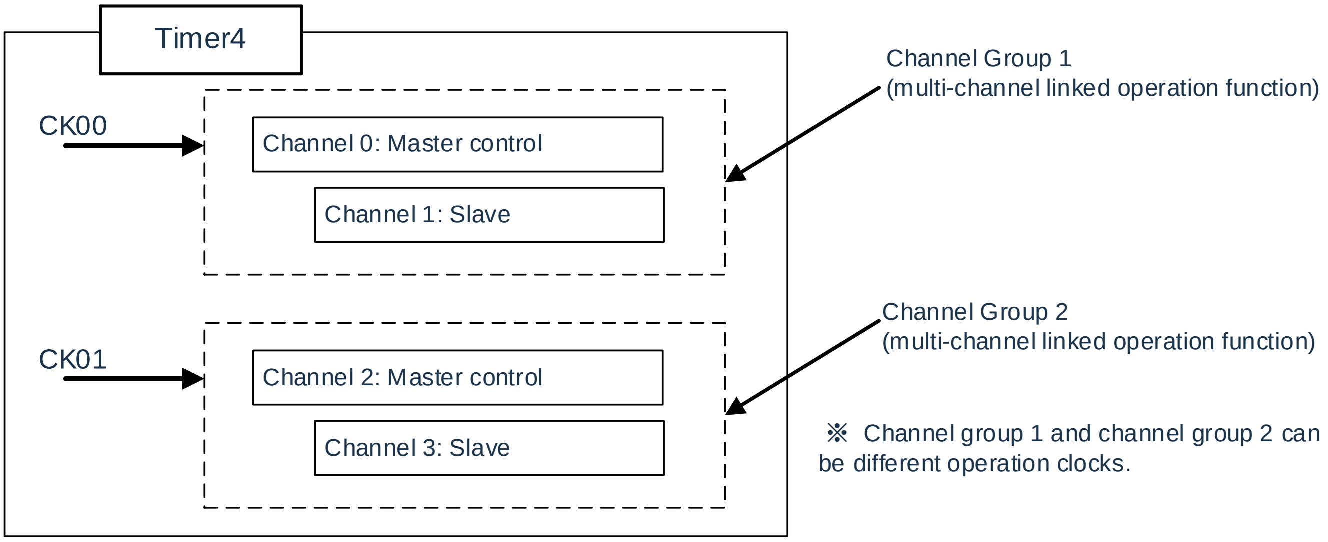
1) Only even channels (channel 0, channel 2) can be set as the main channel.
2) Any channel other than channel 0 can be set as a slave channel.
3) Only the low-level channel of the master channel can be set as a slave channel.
   For example, when channel 0 is set as the main control channel, the channel (channel 1, channel 2, channel 3) starting from channel 1 can be set as a subordinate channel.
4) Multiple slave channels can be set for one master channel.
5) When using multiple master channels, you cannot set slave channels that span the master channel.
   For example, when channel 0 and channel 2 are set as the main control channel, channel 1 can be set as the subordinate channel of the main control channel 0, and channel 3 cannot be set as the subordinate channel of the main control channel 0.
6) The slave channels linked to the master channel need to set the same operating clock. CKSmn0 bits and CKSmn1 bits of the slave channel linked to the master channel (bit15 and bit14 of the timer mode register mn (TMRmn).) value needs to be the same set value.
7) The master channel passes INTTMmn (interrupt), start software trigger, and count clock to the low channel.
8) The slave channel can use the INTTMmn (interrupt), start software trigger, and count clock of the master channel as the source clock, but cannot pass its own INTTMmn (interrupt), start software trigger, and count clock to the low-level channel.
9) The master channel cannot use the INTTMmn (interrupt), start software trigger, and count clocks of other high master channels as source clocks.
10) In order to start the channel to be linked at the same time, it is necessary to set the channel start trigger bit (TSmn) of the linkage channel at the same time.
11) Only all channels of the linkage or the master channel can use the setting of the TSmn bit in the counting operation. You cannot use only the setting of the TSmn bit of the slave channel.
12) In order to stop the channel to be linked at the same time, it is necessary to set the channel stop trigger bit (TTmn) of the linkage channel at the same time.
13) When the linkage is running, CKm2/CKm3 cannot be selected because the master and slave channels require the same operating clock.
14) Timer mode register m0 (TMRm0) is fixed to “0” without a master position. However, because channel 0 is the highest channel, channel 0 can be used as the master channel when the linkage is running.

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

If you set 2 or more non-interconnected channel groups, the above basic rules do not apply to each other.

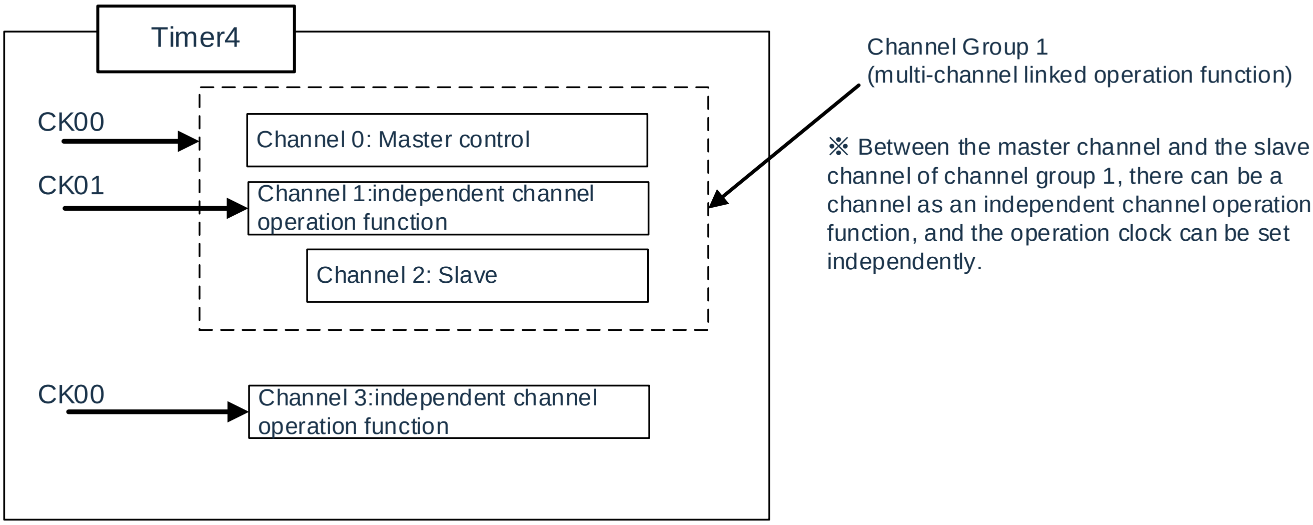
Remark: m: unit number (m=0,1) n: channel number (n=0~3).

Example 1


Timer4
CK00
CK01
Channel 0: Master control
Channel 1: Slave
Channel 2: Master control
Channel 3: Slave
Channel Group 1
(multi-channel linked operation function)
Channel Group 2
(multi-channel linked operation function)

※ Channel group 1 and channel group 2 can be different operation clocks.

Example 2


Timer4
CK00
CK01
CK00
Channel 0: Master control
Channel 1:independent channel operation function
Channel 2: Slave
Channel 3:independent channel operation function
Channel Group 1
(multi-channel linked operation function)

※ Between the master channel and the slave channel of channel group 1, there can be a channel as an independent channel operation function, and the operation clock can be set independently.

5.4.2 Basic rules for the 8-bit timer to operate the function (limited to Channel 1 and Channel 3 of Unit 0).

The 8-bit timer operation function is the function of using the channel of the 16-bit timer as the channel of two 8-bit timers.

Only Channel 1 and Channel 3 can use the 8-bit timer operation function, and there are several rules to be observed when using.

The basic rules for the 8-bit timer to run the function are as follows.

1) The 8-bit timer operation function is only available for Channel 1 and Channel 3.
2) When used as an 8-bit timer, the SPLIT position of the timer mode register mn (TMRmn) is “1”.
3) The high 8-bit timer can be used as an interval timer function.
4) At the beginning of operation, the high 8-bit timer outputs INTTMm1H/INTTMm3H (interrupt) (the same as the operation of MDmn0 bit “1”).
5) The choice of operating clock for a high 8-bit timer depends on the setting of the CKSmn1 bit and CKSmn0 bits of the low-bit TMRmn register.
6) For the high 8-bit timer, the operation of the channel is started by operating the TSHm1/TSHm3 bits, and the operation of the channel is stopped by operating the TTHm1/TTHm3 bits. The status of the channel can be confirmed by the TEHm1/TEHm3 bits.
7) The operation of the low 8-bit timer depends on the setting of the TMRmn register, and there are three functions that support the operation of the low 8-bit timer:
   - Interval timer function
   - External event counter function
   - Delay counting function
8) For the low 8-bit timer, the operation of the channel is started by operating the TSm1/TSm3 bits, and the operation of the channel is stopped by operating the TTm1/TTm3 bits. The status of the channel can be confirmed by the TEm1/TEm3 bits.
9) The operation of the TSHm1/TSHm3/TTHm1/TTHm3 bits is invalid when the 16-bit timer is running. Channel 1 and Channel 3 operate by operating the TSm1/TSm3 bits and the TTm1/TTm3 bits. TEHm3 bits and TEHm1 bits are unchanged.
10) The 8-bit timer function cannot use the linkage operation function (single trigger pulse, PWM, and multiple PWM).

Remarks m: Unit number (m=0) n: Channel number (n=1, 3).

5.5 Operation of the counter

5.5.1 Count clock (f_{TCLK})

The Count Clock of the General-Purpose Timer Unit (f_{TCLK}) can select any of the following clocks via the CCSmn bit of the timer mode register mn (TMRmn).

- The CKSmn0 bit and CKSmn1 bit specify the running clock (f_{MCK}).
- The effective edge of the TImn pin input signal.

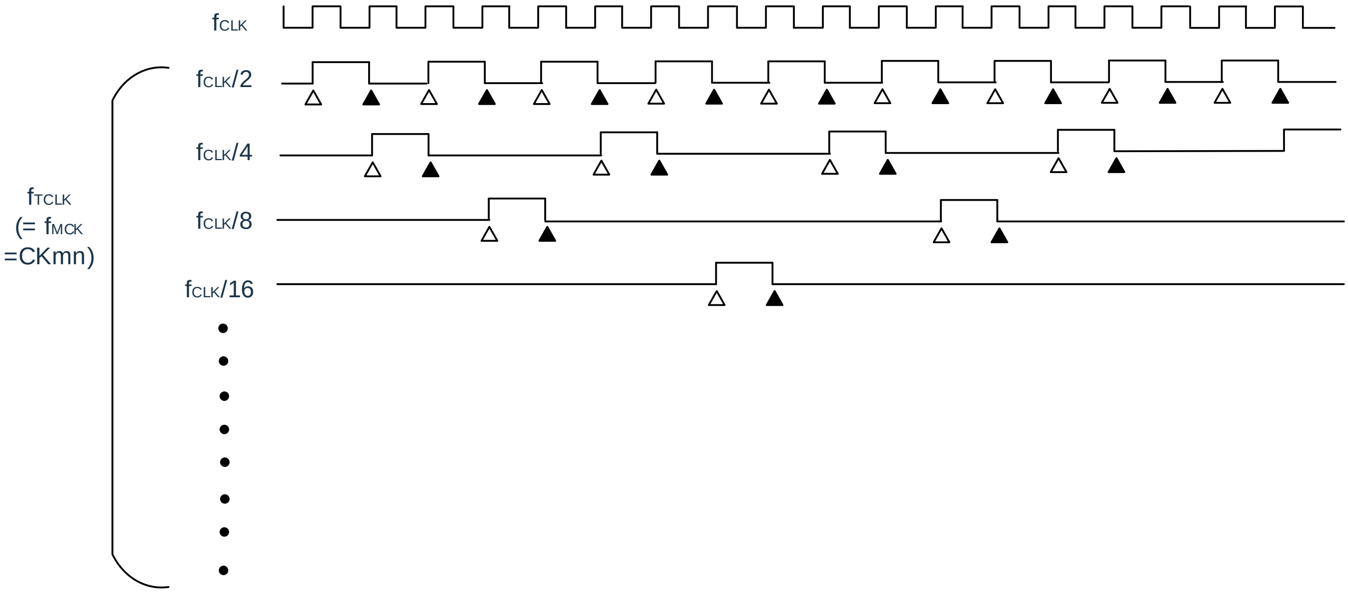
The general-purpose timer unit is designed to run synchronously with the f_{CLK}, so the timing of the count clock (f_{TCLK}) is as follows.

(1) Select the case where CKSmn0 bit and CKSmn1 bit specify the operating clock (f_{MCK}) (CCSmn=0).

According to the timer clock selection register m (TPSm) setting, the count clock (f_{TCLK}) is f_{CLK}~f_{CLK}/2 15. However, when selecting a crossover of f_{CLK}, the TPSm register selects a clock for a signal that is high with only 1 f_{CLK} cycle from the rising edge. When f_{CLK} is selected, it is fixed to high.

To synchronize with f_{CLK}, the timer count register mn (TCRmn) counts after delaying 1 f_{CLK} clock from the rising edge of the count clock. For convenience, it is called “counting on the rising edge of the counting clock.”

Figure 5-24 Timing of f_{CLK} and count clock (f_{TCLK}) (in the case of CCSmn=0)



Remark 1. △: Counts the rising edge of the clock

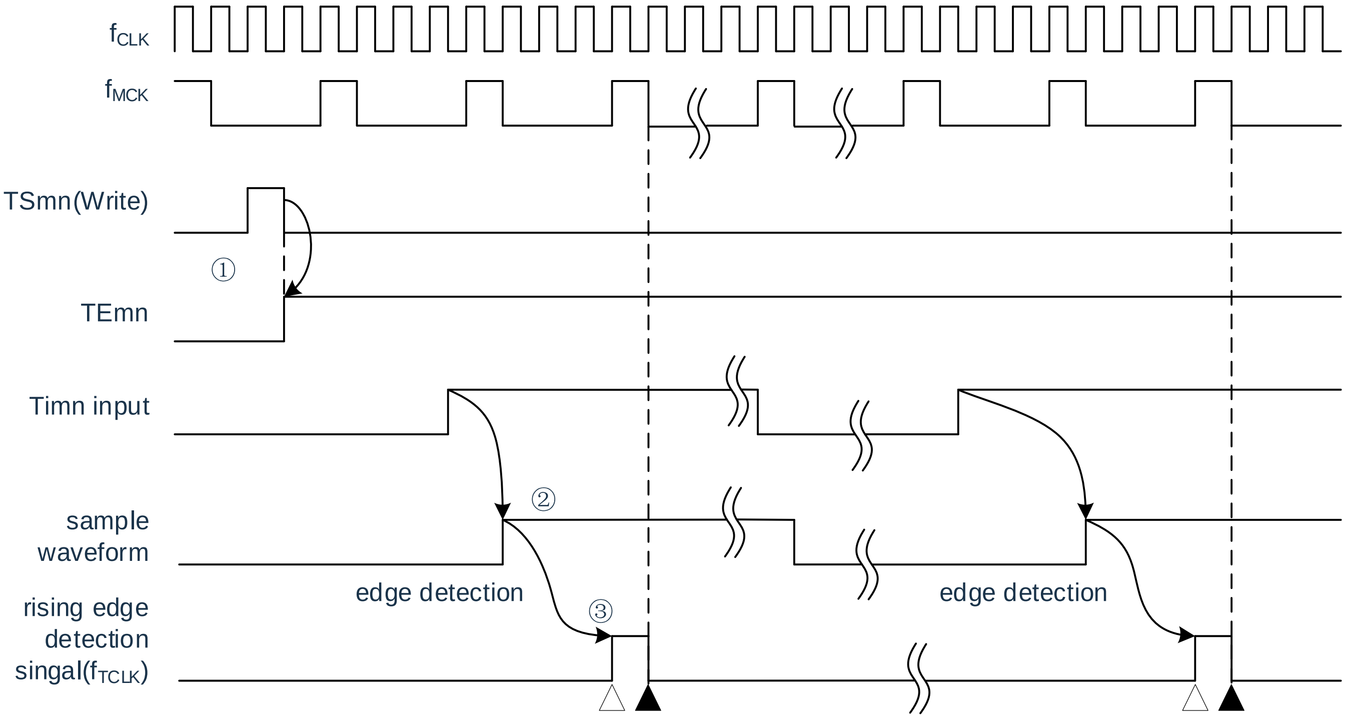
▲: Synchronization, increment/decrement of counters

2.f_{CLK}: The clock for CPU/peripheral hardware

(2) Selecting the active edge of the TImn pin input signal (CCSmn=1)

The Count Clock (fTCLK) is the signal that detects the active edge of the input signal of the TImn pin and synchronizes it with the rising edge of the next fMCK. In fact, this is a signal that is delayed by 1 to 2 fMCK clocks than the input signal at the TImn pin (3 to 4 when using noise filters). fMCK clocks). To obtain synchronization with fCLK, the timer count register mn (TCRmn) counts after delaying 1 fCLK from the rising edge of the count clock. For convenience, it is called “counting on the effective edge of the input signal at the TImn pin”.

Figure 5-25 Count Clock (f_{TCLK}) (CCSmn=1, without noise filter)



(1) Start the operation of the timer by placing the TSmn position bit, and wait for the valid edge of the TImn input.

(2) Sample the rising edge of the TImn input through f_{MCK}.

(3) The edge is detected on the rising edge of the sampled signal, and the detection signal (counting clock) is output.

Note 1 △: Counts the rising edge of the clock

▲: Synchronization, increment/decrement of counters

2.f_{CLK}: CPU peripheral hardware clock

f_{MCK}: The operating clock for channel n

3. The same waveform is used for the measurement of the input pulse interval, the measurement of the high and low levels of the input signal, the delay counter, and the TImn input for the single-trigger pulse output function.

5.5.2 Start timing of counter

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

The operation from the counting enabled state to the start of the timer count register mn (TCRmn) is shown in Table 5-5.

Table5-5 Operation from the counting enabled state to the start of the timer count register mn (TCRmn)

Operating mode of the timer	Operation after setting the TSmn bit to “1”
• Interval timer mode	No action is taken from the time the start trigger is detected (TSmn=1) until the count clock is generated. The value of the TDRmn register is loaded into the TCRmn register by the first count clock and the count is decremented by the subsequent count clock (see “Operation in 5.5.3(1) 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 the input edge of TImn is detected, the count is decremented by the subsequent count clock. (Refer to “5.5.3(2) Operation of Event Counter Mode“).
• Capture mode	No action is taken from the time the start of the trigger is detected until the count clock is generated. The “0000H” is loaded into the TCRmn register by the first count clock and the count is incremented by the subsequent count clock (refer to the operation of the capture mode “5.5.3(3) (interval measurement of the input pulse)”.
• Single count mode	By writing “1” to the TSmn bit in the state where the timer is stopped (TEmn=0), it enters the start trigger, etc Pending status. No action is taken from the time the start of the trigger is detected until the count clock is generated. The value of the TDRmn register is loaded into the TCRmn register by the first count clock and passed through subsequent meters The number of clocks is decremented (see “5.5.3(4) Single-Count Mode Operation”).
• Capture & Single Count Mode	By writing “1” to the TSmn bit in the state where the timer is stopped (TEmn=0), it enters the start trigger, etc Pending status. No action is taken from the time the start of the trigger is detected until the count clock is generated. “0000H” is loaded into the TCRmn register by the first count clock and proceeded via subsequent count clocks Increment count (see “5.5.3(5) Capture & Operation of Single Count Mode (Measurement of High Level Width)”).

5.5.3 Operation of counter

The following describes the counter operation for each mode.

(1) Operation of interval timer mode

(1) Enter the operating enabled state (TEmn=1) by writing "1" to the TSmn bit. The timer count register mn (TCRmn) holds the initial value until the count clock is generated.

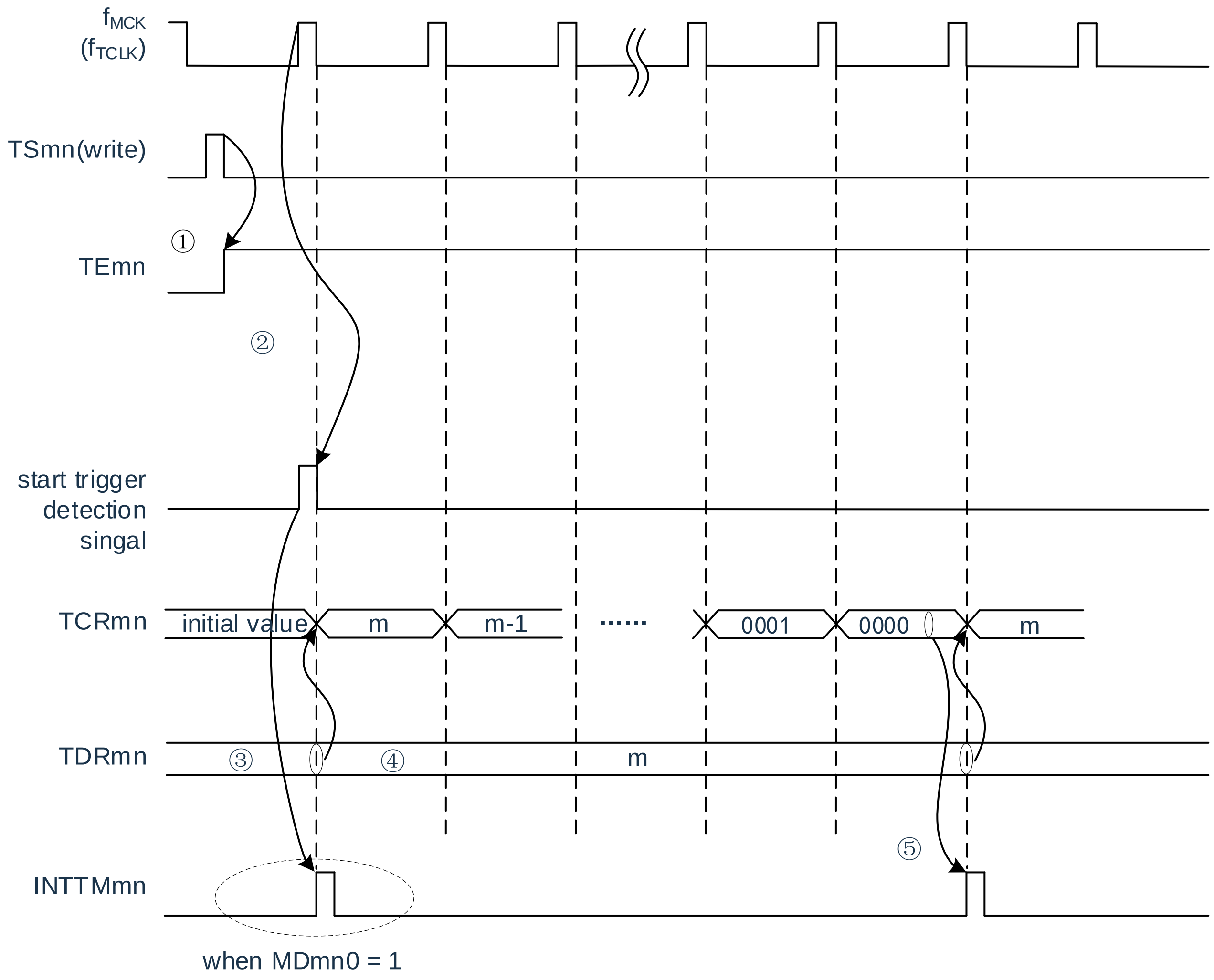
(2) Generate a start trigger signal by allowing the first counting clock (fMCK) after operation.

(3) When the MDmn0 bit is "1", INTTMmn is generated by starting the trigger signal.

(4) Load the value of the timer data register mn (TDRmn) into the TCRmn register by allowing the first count clock after operation, and start counting in interval timer mode.

(5) If the TCRmn register is decremented to count to "0000H", INTTMmn is generated by the next count clock (fMCK), and the timer data register mn (TDRmn) value continues to be counted after loading the TCRmn register.

Figure 5-26 Operation timing (interval timer mode)



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

Remark: f_{MCK}, the start trigger detection signal and INTTMmn are synchronized with fCLK and are valid for 1 clock.

(2) Operation of event counter mode

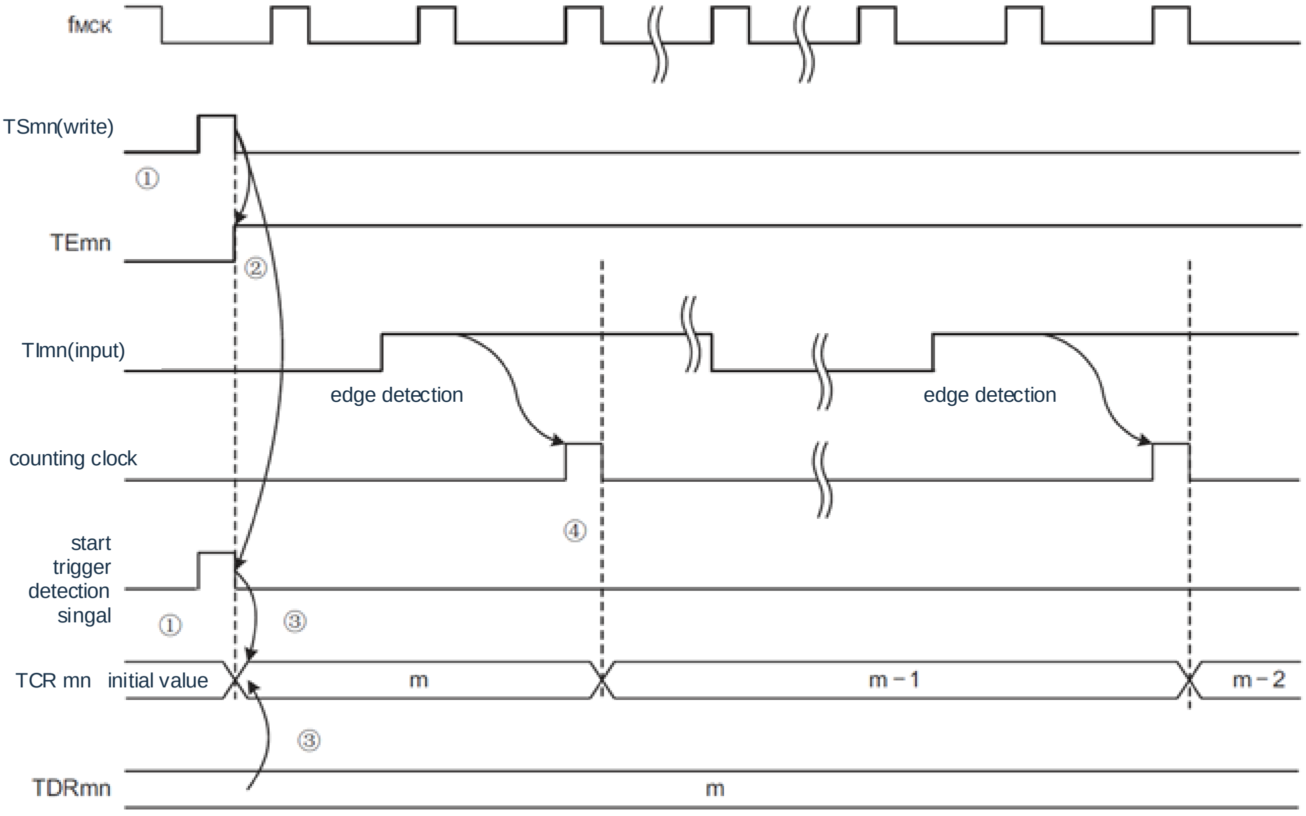
(1) During the operation stop state (TEmn=0), the timer count register mn (TCRmn) maintains the initial value.

(2) Enter the operating enabled state (TEmn=1) by writing “1” to the TSmn bit.

(3) Load the value of the timer data register mn (TDRmn) into the TCRmn register while both the TSmn bit and the TEmn bit become “1” and start counting.

(4) Thereafter, on the valid edge of the TImn input, the value of the TCRmn register is decremented by counting the clock.

Figure 5-27 Operation timing (event counter mode)

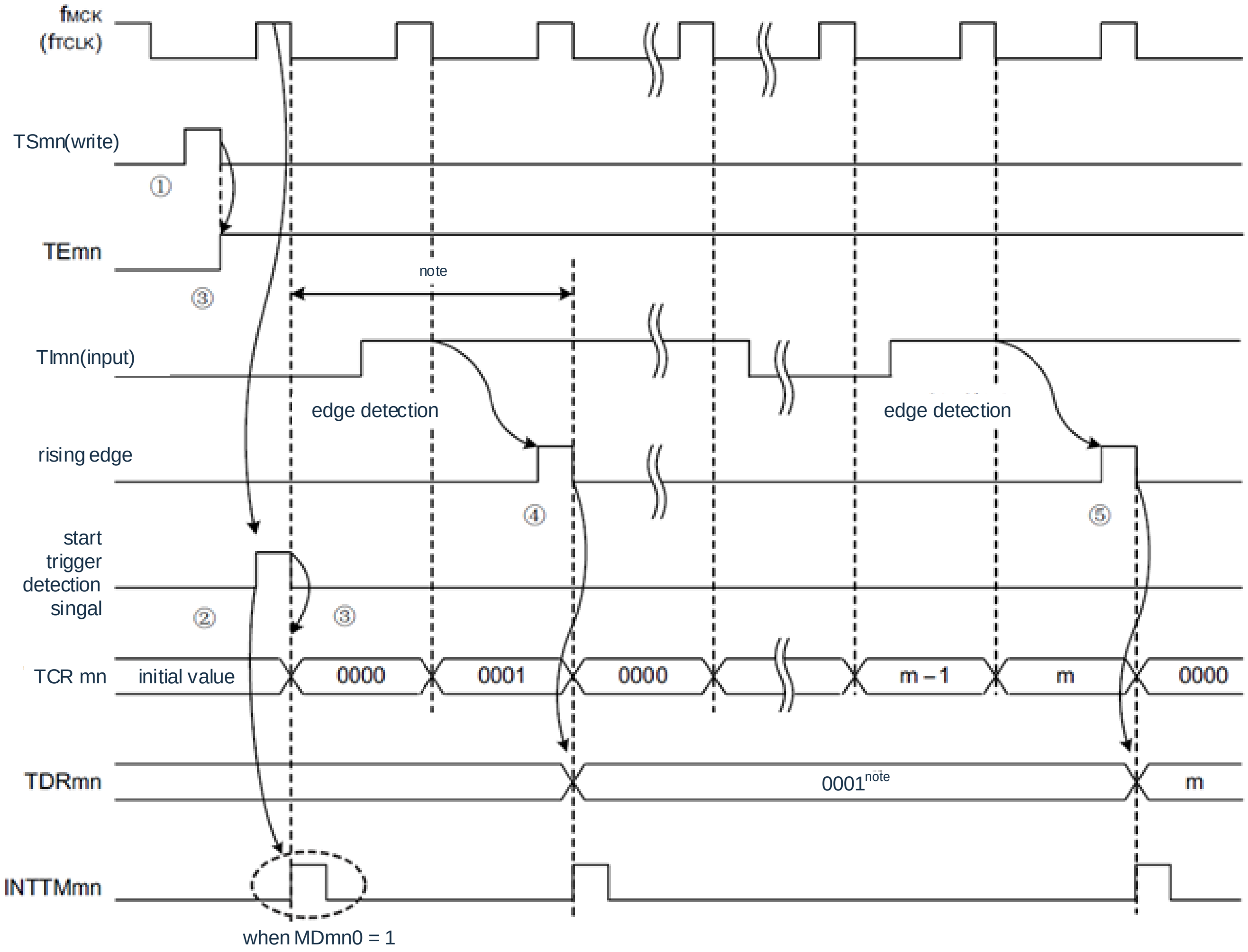


Remark This is the timing when no noise filter is used. If a noise filter is used, edge detection is delayed by another 2 f_{MCK} cycles (3 to 4 cycles total) from the TImn input. The 1-cycle error is due to the fact that the TImn input is out of sync with the count clock (f_{MCK}).

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

(1) Enter the operating enabled state (TEmn=1) by writing “1” to the TSmn bit.
(2) The timer count register mn (TCRmn) maintains the initial value until the count clock is generated.
(3) Generate a start trigger signal by allowing the first counting clock (fMCK) after operation. Then, “0000H” is loaded into the TCRmn register and counts start in capture mode (when MDmn0 bits are “1”, INTTMmn is generated by the start trigger signal).
(4) If a valid edge of the TImn input is detected, the value of the TCRmn register is captured to the TDRmn register, and an INTTMmn interrupt is generated. The snap value at this point is meaningless. The TCRmn register continues counting starting at “0000H”.
(5) If a valid edge of the next TImn input is detected, the value of the TCRmn register is snapped to the TDRmn register and an INTTMmn interrupt is generated.

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



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

Notice: Because the first count clock cycle runs after the TSmn bit is written and delays the start of the count before the count clock is generated, an error of up to 1 clock cycle is generated. In addition, if you need information about the start counting timing, set MDmn0 at “1” so that an interrupt can occur when the count starts.

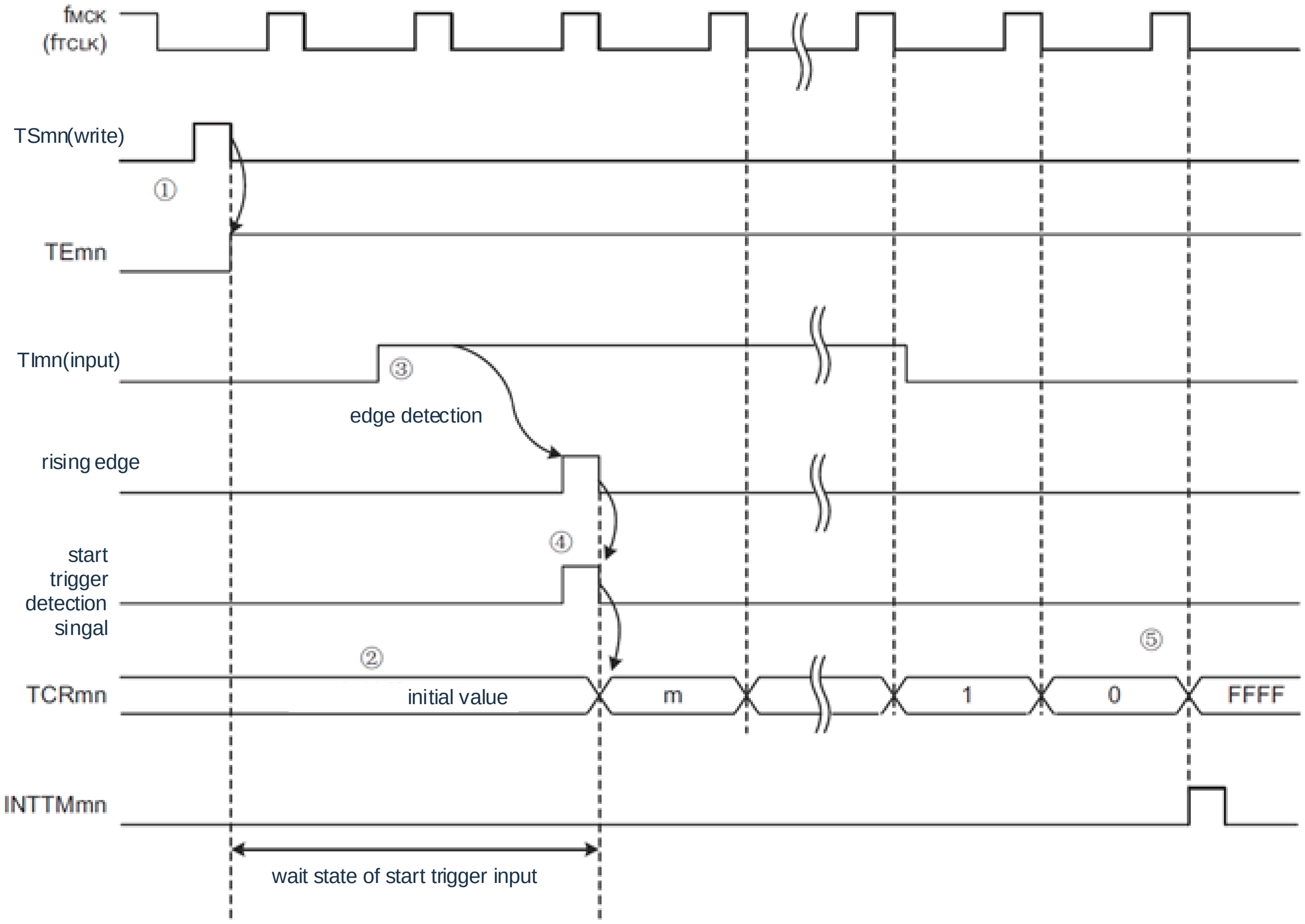
Remark This is the timing when no noise filter is used. If a noise filter is used, edge detection is delayed by 2 more f_{MCK} cycles from the TImn input (3~4 cycles in total). The 1-cycle error is due to the fact that the TImn input is out of sync with the count clock (f_{MCK}).

(4) Operation of single-count mode

(1) Enter the operating enabled state (TEmn=1) by writing “1” to the TSmn bit.
(2) The timer count register mn (TCRmn) holds the initial value until a 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 the count begins.
(5) When the TCRmn register decrements the count to “0000H”, an INTTMmn interrupt is generated, and the value of the TCRmn register becomes “FFFFH”, stopping the count.

.

Figure 5-29 Operation timing (single count mode)



Note: This is the timing when no noise filter is used. If a noise filter is used, edge detection is delayed by 2 more f_{MCK} cycles from the TImn input (3~4 cycles in total). The 1-cycle error is due to the fact that the TImn input is out of sync with the count clock (f_{MCK}).

(5) Capture & single count mode operation (measurement of high-level width)

(1) The TSmn bit of the start register m (TSm) is written “1” through the given timer channel to enter the operating enable state (TEmn=1).

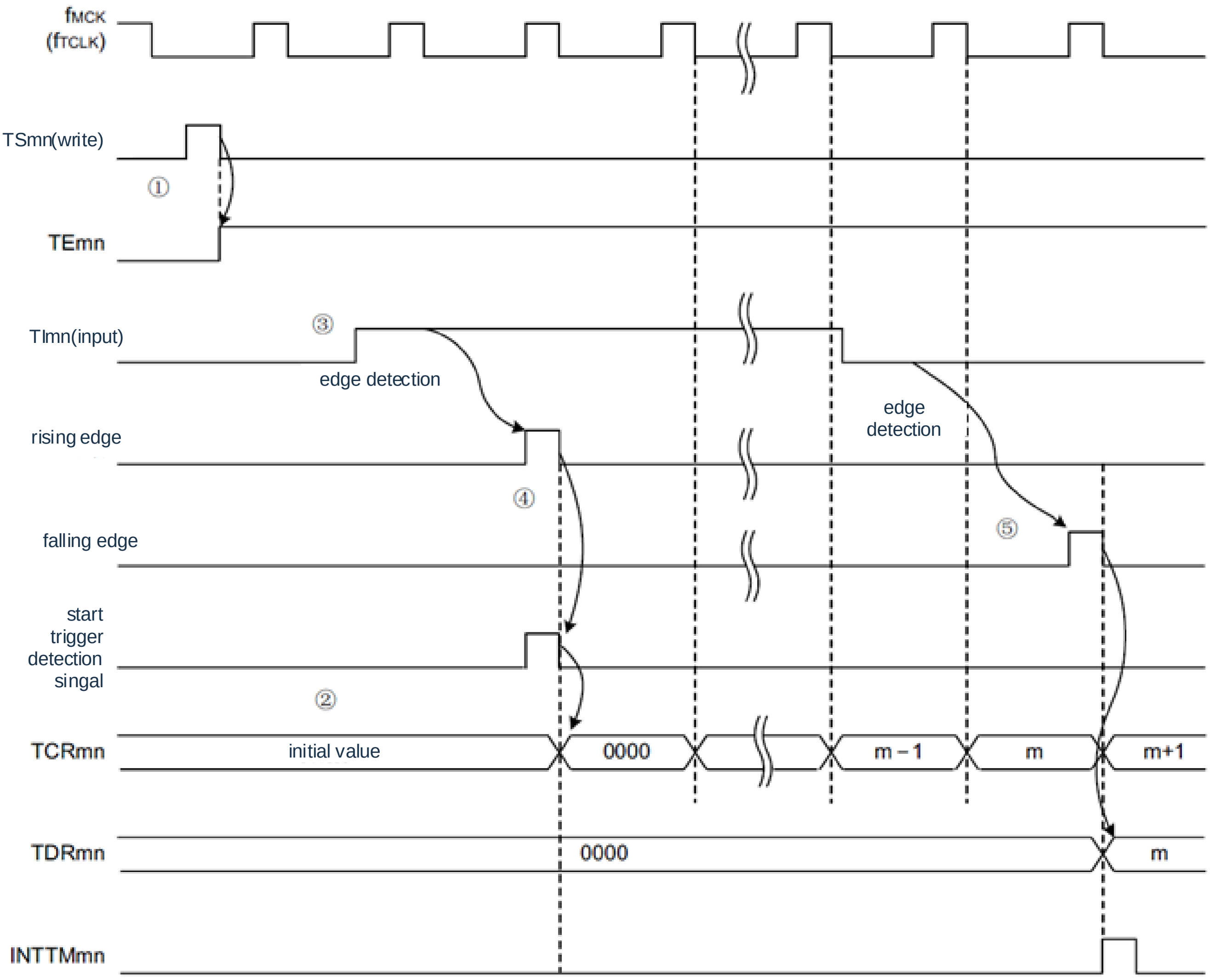
(2) The timer count register mn (TCRmn) holds the initial value until a start trigger signal is generated.

(3) Detect the rising edge of TImn input.

(4) After the start trigger signal is generated, “0000H” is loaded into the TCRmn register and the count begins.

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

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

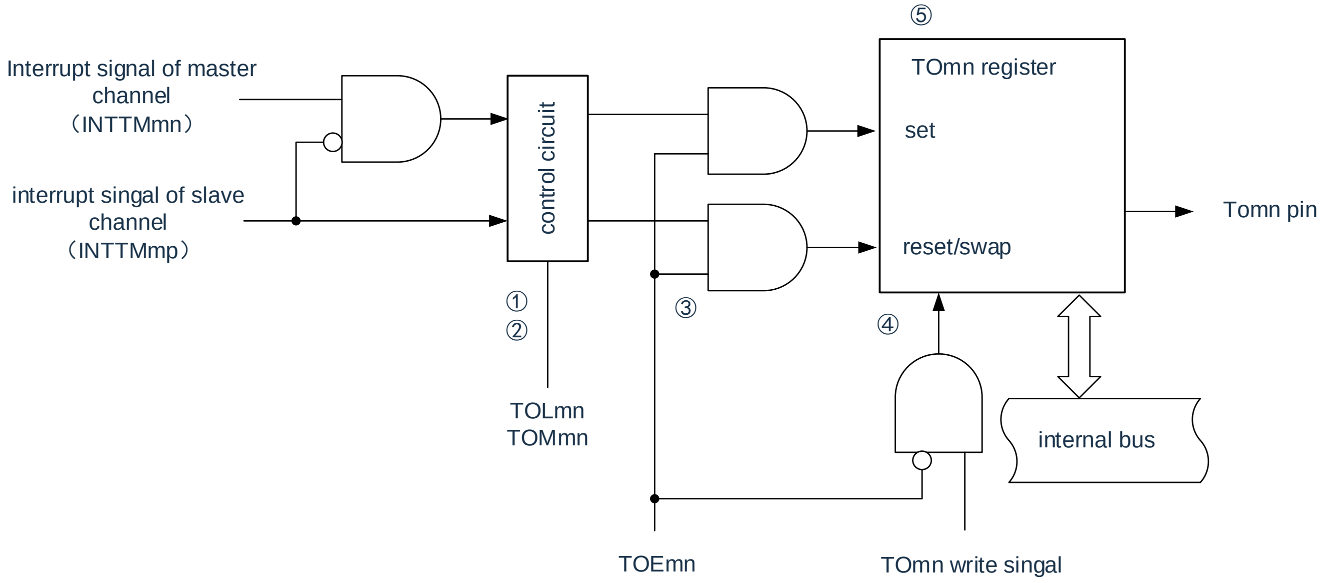


Note This is the timing when no noise filter is used. If a noise filter is used, edge detection is delayed by 2 more fMCK cycles from the TImn input (3~4 cycles in total). The 1 cycle error is because the TImn input and counting clock (fMCK) are not synchronized.

5.6 Control of the channel output (TOmn pin)

5.6.1 Structure of the TOmn pin output circuit

Figure 5-31 Structure of output circuitry



The following describes the output circuit of the Tomn pin.

(1) When the TOMmn bit is "0" (main channel output mode), ignore the timer output level register m (TOLm) setting, and only INTTMmp (Slave Channel Timer Interrupt) passes the given timer output register m (TOm).

(2) When the TOMmn bit is "1" (slave channel output mode), INTTMmn (master channel timer interrupt) and INTTMmp (slave channel timer interrupt) are passed to the TOm register.
At this point, the TOLm register is valid and the following
signals are controlled:
TOLmn=0: Normal-phase operation (INTTMmn emset, INTTMmp reset)
TOLmn=1: Inverting operation (INTTMmn reset, INTTMmp set).
When both INTTMmn and INTTMmp (0% of the PWM output) are generated simultaneously, the INTTMmn (reset signal) is preferred and the INTTMmn is masked (Set signal).

(3) In the state of allowing timer output (TOEmn=1), the INTTMmn (master channel timer interrupt) and INTTMmp (slave channel timer interrupt) are passed to the TOm register. The write operation of the TOm register (TOmn write signal) is invalid.
When the TOEmn bit is "1", the output of the TOmn pin is not changed except for the interrupt signal.

To initialize the output level of the TOmn pin, the TOm register needs to be written after setting the disable timer output (TOEmn=0).

(4) In the state of disabling the timer output (TOEmn=0), the write operation of the TOmn bit of the object channel (TOmn write signal) is valid. When the timer output is in a disabled state (TOEmn=0), INTTMmn (master channel timer interrupt) and INTTMmp (slave channel timer interrupt) are not passed to the TOm register.

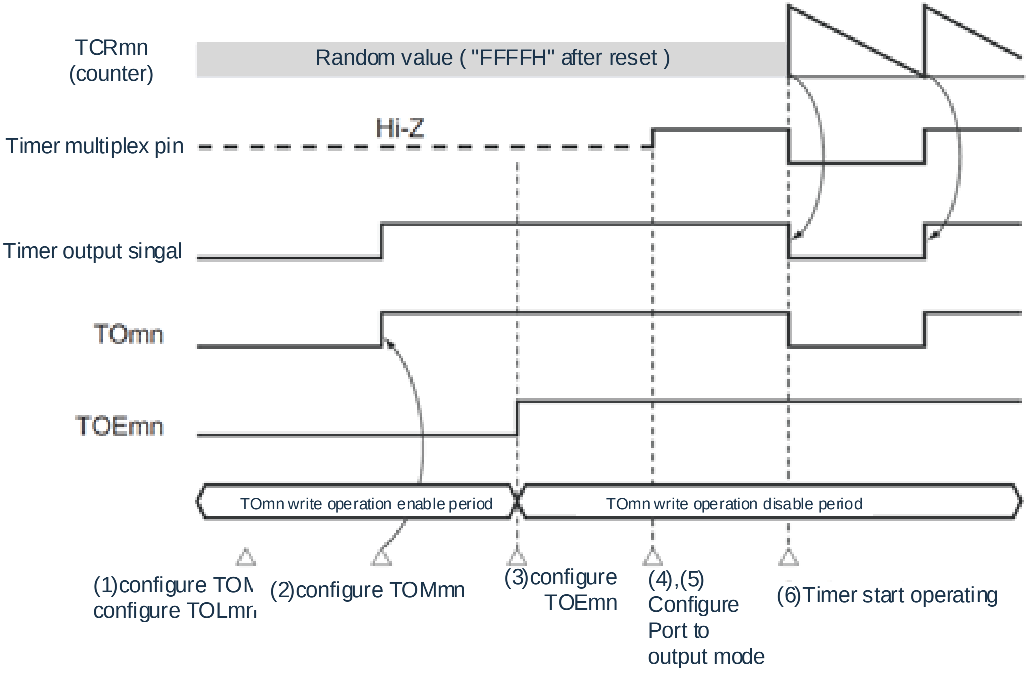
(5) The TOm register can be read at any time and the output level of the TOmn pin can be confirmed.

Remarks m: Unit number (m=0,1) n: channel number n=0~3 (master channel: n=0, 2)
p: The slave channel number
n=0: p=1, 2, 3
n=2: p=3

5.6.2 Output setting of the TOmn pin

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

Figure 5-32 State change from setting timer output to start of operation



(1) Set the operating mode of the timer output.

- TOMmn bit (0: master channel output mode, 1: slave channel output mode).
- TOLmn bit (0: positive logic output, 1: negative logic output).

(2) Set the timer output signal to the initial state by setting the timer output register m (TOm).
(3) Write “1” to the TOEmn bit, allowing the timer output (disable writing of TOm register).
(4) Set the port to a digital input/output through the Port Mode Control Register (PMCxx).
(5) Set the input/output of the port to output.
(6) Allow the timer to run (TSmn=1).

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

5.6.3 Cautions for channel output operation

(1) Change of TOm, TOEm, TOLm, and TOMm register settings during timer operation

The operation of the timer (the operation of the timer count register mn (TCRmn) and the timer data register mn (TDRmn)) and the TOmn output circuit are independent of each other. Thus, the timer output register m (TOm), the timer output enable register m (TOEm) and the timer output level register m (TOLm) config value changes do not affect the operation of the timer, you can change the config value while the timer is running. However, in order to output the expected waveform from the TOmn pin during the operation of each timer, it must be set to the values of the register settings shown in 5.8 and 5.9 for each run content example.

If you change the settings of the TOEm register and the TOLm register in addition to the TOm register before and after generating the timer interrupt (INTTMmn) signal for each channel, it is based on the timer interrupt (INTTMmn) being generated Whether the signal changes before or after generation, the waveform of the TOmn pin output may be different.

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

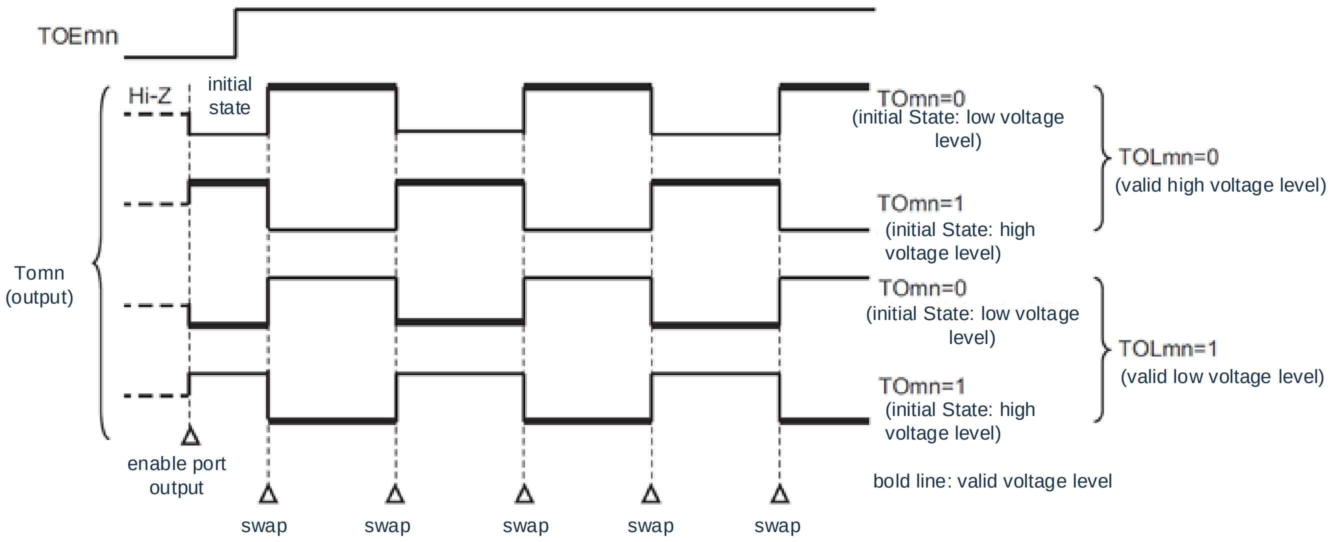
(2) The initial level of the TOmn pin and the output level after the timer starts operating

The timer output register m (TOm) is written before the port output is enabled and in the state of disabling the timer output (TOEmn=0), and set to the timer output enabled state (TOEmn=1) after changing the initial level. The change in the output level of the TOmn pin is shown below.

(a) When starting operation in the master channel output mode (TOMmn=0).

In the master channel output mode (TOMmn=0), the timer output level register m (TOLm) is not set. If the operation of the timer begins after setting the initial level, the output level of the inverting TOmn pin is reversed by generating an alternating signal.

Figure 5-33 Output state of TOmn pin at alternate output (TOMmn=0)

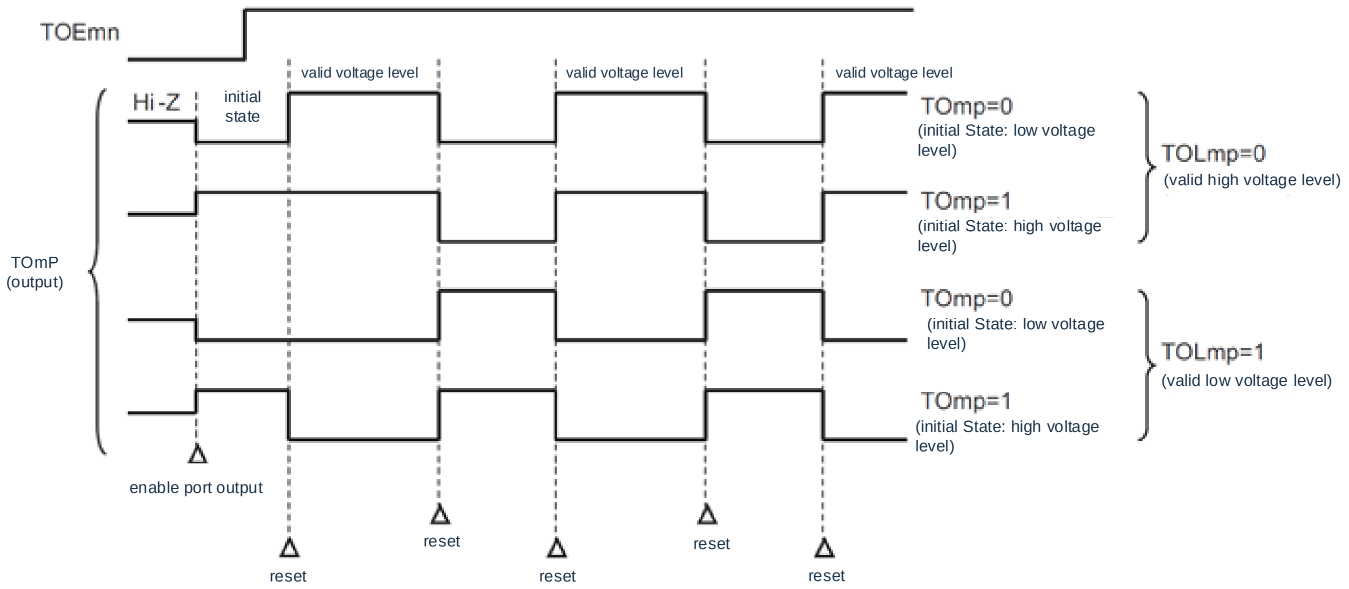

TOEmn
Hi-Z
initial state
Tomn (output)
TOmn=0 (initial State: low voltage level)
TOmn=1 (initial State: high voltage level)
TOLmn=0 (valid high voltage level)
TOmn=0 (initial State: low voltage level)
TOmn=1 (initial State: high voltage level)
TOLmn=1 (valid low voltage level)
enable port output
swap
swap
swap
swap
swap
bold line: valid voltage level

Note 1. Alternating: The output status of the inverting TOmn pin.
2. m: unit number (m=0,1) n: channel number (n=0~3).

(b) Start of operation in slave channel output mode (TOMmn=1) (PWM output)

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

Figure 5-34 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)
TOmp=0 (initial State: low voltage level)
TOmp=1 (initial State: high voltage level)
TOLmp=0 (valid high voltage level)
TOLmp=1 (valid low voltage level)
enable port output
reset
reset
reset
reset
reset

Note 1. Set: The output signal of the TOmp pin changes from an invalid level to a valid level.

Reset: The output signal of the TOmp pin changes from a valid level to an invalid level.

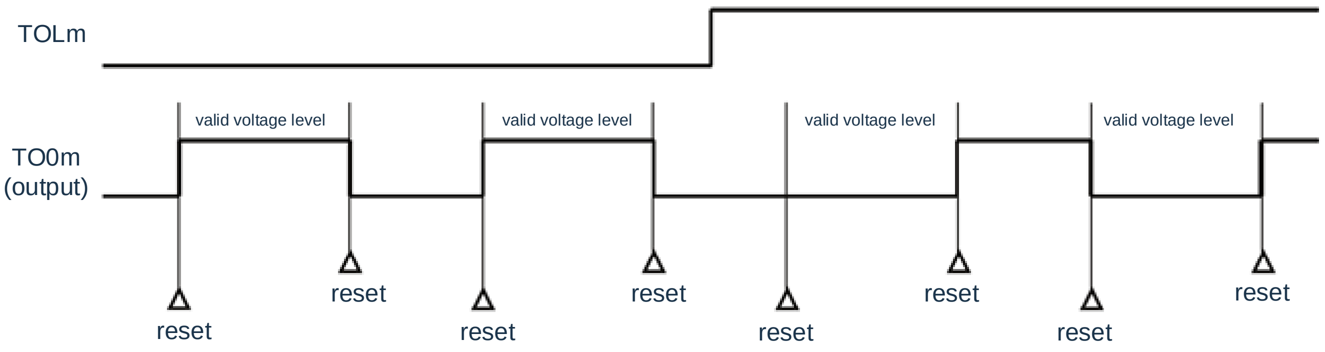
2. m: unit number (m=0,1) n: channel number (p=1~3).

(3) TOmn pin change regarding slave channel output mode (TOMmn=1)

(a) When the setting of the timer output level register m (TOLm) is changed during timer operation

If you change the setting of the TOLm register while the timer is running, the setting is valid when the TOmn pin change condition is generated. The output level of the TOmn pin cannot be changed by rewriting the TOLm register.
When the TOMmn bit is “1”, the operation when changing the value of the TOLm register in timer operation (TEmn=1) is shown below.

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



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

(b) Set/reset timing

To achieve 0% and 100% output at PWM output, the TOmn pin/TOmn at the time of the master channel timer interrupt (INTTMmn) is generated through the slave channel. The timing delay of the bits is 1 count clock.

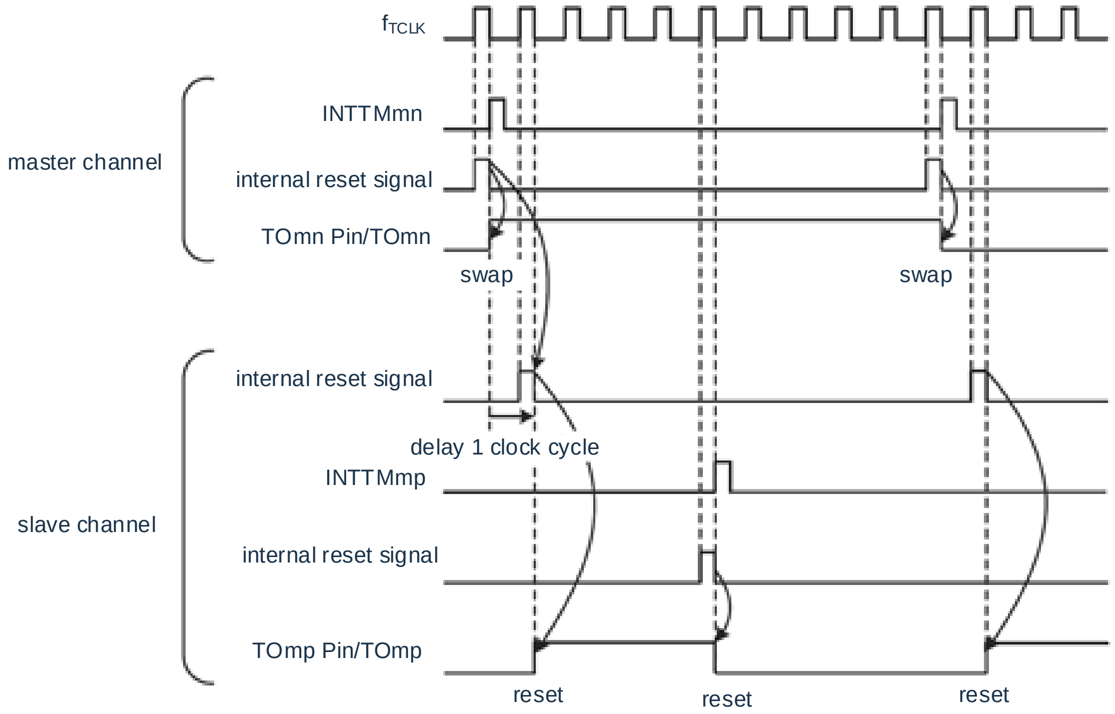
When the set condition and the reset condition arise at the same time, the reset condition takes precedence.

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

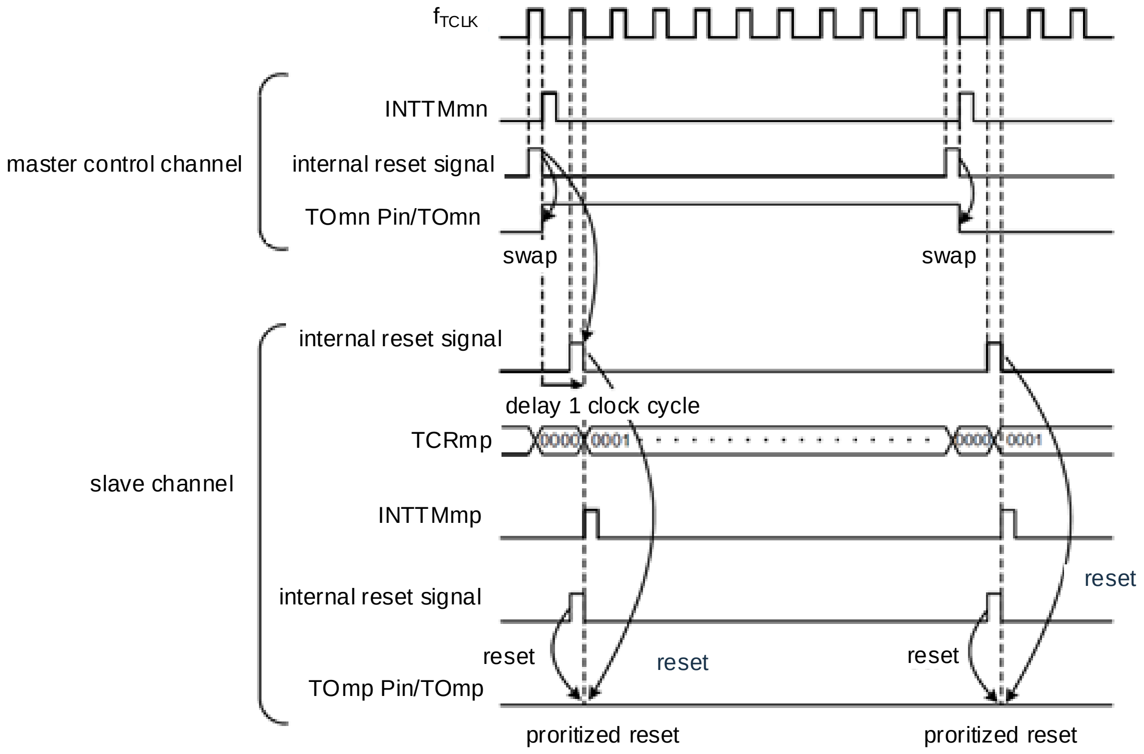
Master channel: TOEmn=1, TOMmn=0, TOLmn=0
Slave channel: TOEmp=1, TOMmp=1, TOLmp=0

Figure 5-36 Reset/set timing operation status

(1) Basic operation timing



(2) 0% duty cycle operation timing



Note 1 Internal reset signal: A reset/alternating signal on the TOmn pin

Internal set signal: The set signal of the TOmn pin

2.m: Unit number (m=0,1) n: channel number n=0~3(Master channel: n=0, 2)

p: The slave channel number

n=0: p=1, 2, 3

n=2: p=3

5.6.4 One-time operation of the TOmn bit

Like the timer channel start register m (TSm), the timer output register m (TOm) has the set bit (TOmn) for all channels. This allows the TOmn bit of all channels to be operated at once.

Figure 5-37 Example of a one-time operation with the TO0n bit

Before writing

TO0	0	0	0	0	0	0	0	0	0	0	0	0	TO03 1	TO02 0	TO01 1	TO00 0
TOE0	0	0	0	0	0	0	0	0	0	0	0	0	TOE03 0	TOE02 0	TOE01 0	TOE00 1

The data to write

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

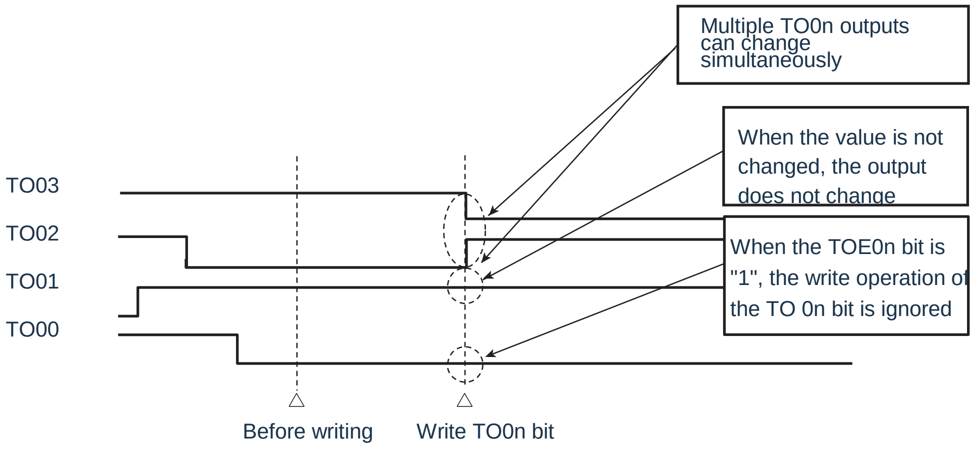
After writing

													↓○	↓○	↓○	↓×
TO0	0	0	0	0	0	0	0	0	0	0	0	0	TO03 0	TO02 1	TO01 1	TO00 0

Only the TOmn bit with TOEmn bit “0” can be written, ignoring the write operation of the TOmn bit with TOEmn bit “1”.

TOmn (channel output) with TOEmn bit “1” is not affected by write operations, even if the write TOmn bit is ignored, and output changes caused by timer operation occur normally.

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


Multiple TO0n outputs can change simultaneously
When the value is not changed, the output does not change
When the TOE0n bit is "1", the write operation of the TO 0n bit is ignored
TO03
TO02
TO01
TO00
Before writing
Write TO0n bit

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

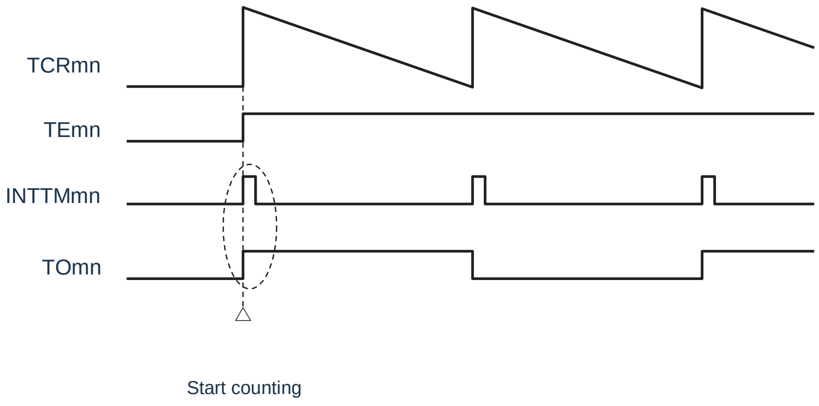
5.6.5 Timer interrupt and TOmn pin output when counting starts

In interval timer mode or capture mode, the MDmn0 bit of the timer mode register mn (TMRmn) is the bit that sets whether a timer interrupt is generated at the start of the count.

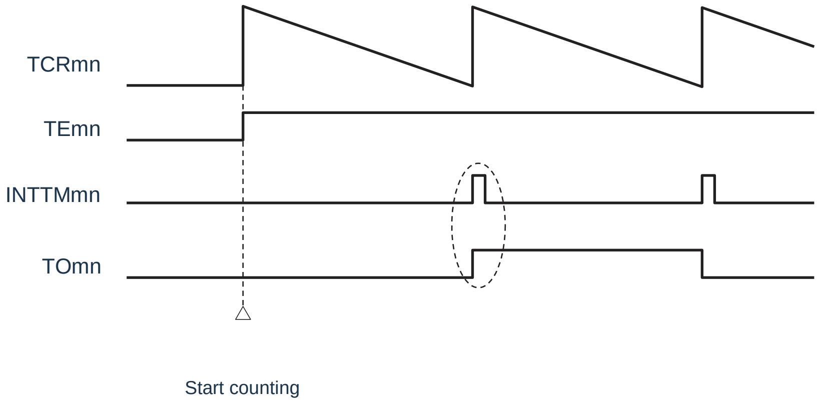
When the MDmn0 bit is “1”, the start timing of the count can be known by generating a timer interrupt (INTTMmn). In other modes, the timer interrupt at start count and the TOmn output are not controlled. An example of operation when set to interval timer mode (TOEmn=1, TOMmn=0) is shown below.

Figure 5-39 Running example of timer interrupt and TOmn output at the start of counting

(a) When the MDmn0 bit is “1”



(b) When the MDmn0 bit is “0”



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

When the MDmn0 bit is “0”, the timer interrupt (INTTMmn) is not output at the beginning of the count and the TOmn does not change, while the INTTMmn is output after counting 1 cycle and TOmn is output alternately.

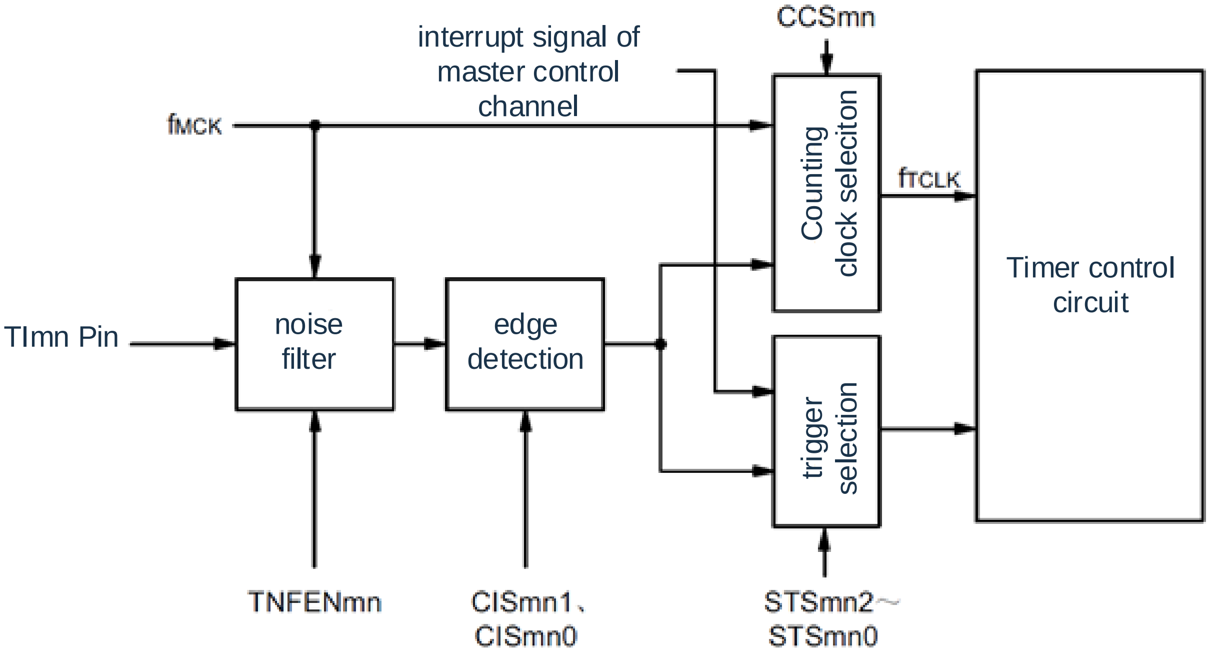
Note m: Unit number (m=0, 1) n: channel number (n=0~3).

5.7 Control of timer input (TImn)

5.7.1 Structure of TImn pin input circuit

The signal from the timer input pin is input to the timer control circuit through the noise filter and edge detection circuitry. For pins that need to be noise removed, the corresponding pin noise filter must be set to active. The structure of the input circuit is as follows.

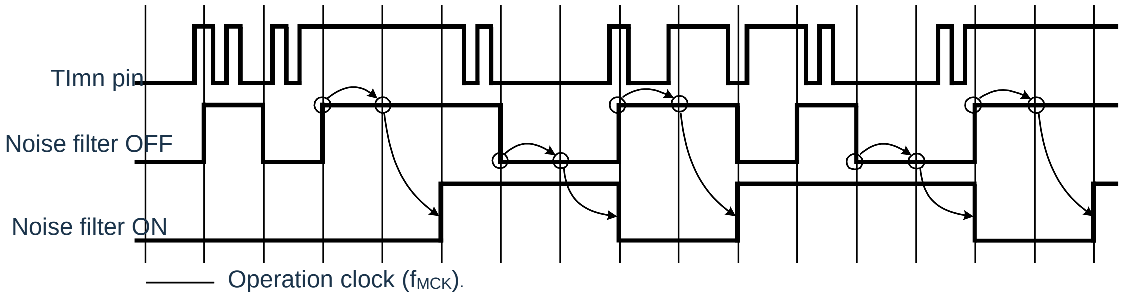
Figure 5-40: Structure of input circuit



5.7.2 Noise filter

When the noise filter is invalid, it is only synchronized by the running clock of channel n (f_{MCK}); When the noise filter is valid, the two clocks are detected to be consistent after synchronization through the operating clock of channel n (f_{MCK}). The waveform of the TM4lmn input pin after passing through the noise filter circuit in the case of noise filter ON or OFF is shown below.

Figure 5-41 Waveform of the TImn input pin in the case of noise filter ON or OFF



Notice: The input waveform of the Tlmn pin is used to illustrate the operation of the noise filter ON or OFF. In practice, the Tlmn input must be entered according to the high and low level widths of the Tlmn input shown in the AC characteristics of the data sheet.

5.7.3 Considerations when manipulating channel inputs

When set to not use the timer input pin, no operating clock is provided to the noise filter circuit. Therefore, the following wait times are required from the channel operation that is set to use the timer input pin to the channel operation corresponding to the set timer input pin.

(1) When the noise filter is OFF

If any bit is set in the state that bit12(CCSmn), bit9(STSmn1) and bit8(STSmn0) of the timer mode register mn (TMRmn) are all “0”, it is necessary to set the operation permission trigger of the timer channel start register (TSm) after at least two running clock (f_{MCK}) cycles.

(2) When the noise filter is ON

If any bit is set in the state that bit12(CCSmn), bit9(STSmn1) and bit8(STSmn0) of the timer mode register mn (TMRmn) are all “0”, the operation permission trigger of the timer channel start register (TSm) must be set at least after four running clock (f_{MCK}) cycles.

5.8 Independent channel operation function of the universal timer unit

5.8.1 Operates as an interval timer / square wave output

(1) Interval timer

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

INTTMmn (timer interrupt) generation period = Counting clock period X (Setting value of TDRmn+1)

(2) Operates as square wave output

TOmn produces INTTMmn while alternating the output, with an output duty cycle of 50% square wave.
The period and frequency of the TOmn output square wave can be calculated using the following formula:

- Square wave period of TOmn output = Counting clock period × (Setting value of TDRmn +1) × 2

- Square wave frequency of TOmn output = Counting clock frequency / {(Setting value of TDRmn +1) × 2}

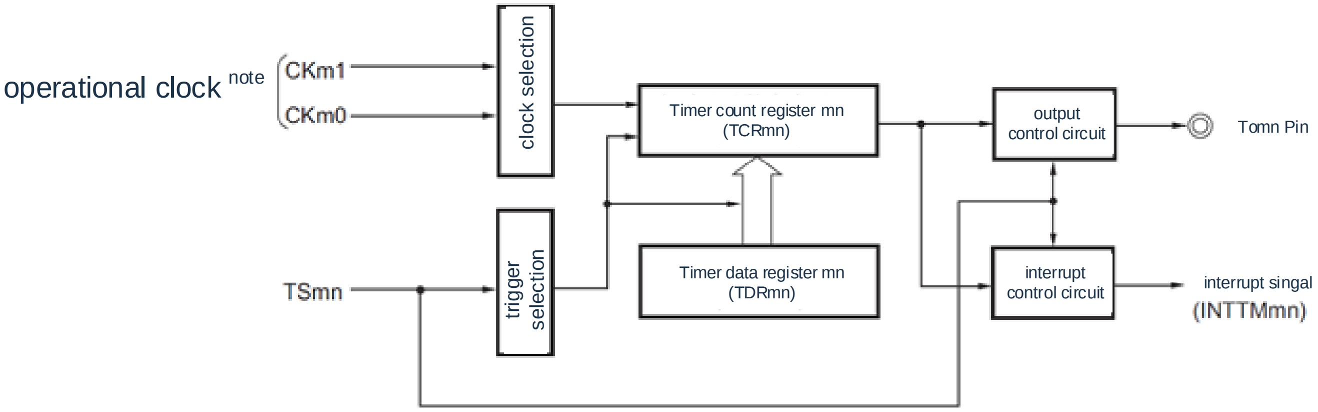
In interval timer mode, the timer count register mn (TCRmn) is used as a decreasing counter.

After setting the channel start trigger bit (TSmn, TSHm1, TSHm3) of the timer channel start register m (TSm) to “1”, pass through the first A count clock loads the value of the timer data register mn (TDRmn) into the TCRmn register. At this point, if the MDmn0 bit of timer mode register n (TMRmn) is “0”, INTTMmn is not output and TOmn also does not alternate outputs. If the MDmn0 bit of the TMRmn register is “1”, INTTMmn is output and TOmn is output alternately. The TCRmn register then decrements the count by counting the clock.

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

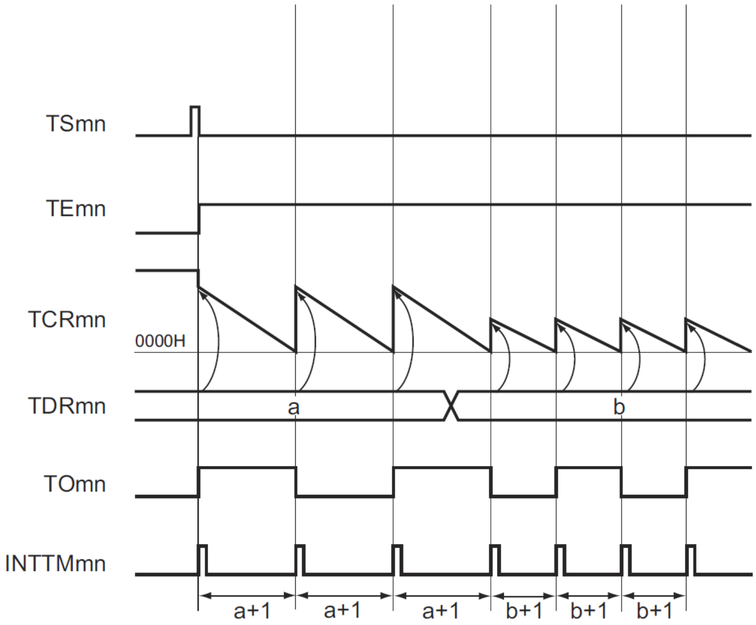
TDRmn registers can be overridden at any time, and the values of the overridden TDRmn registers are valid from the next cycle.

Figure 5-42 Basic timing example of operation as a spacer timer / square wave output (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
Tomn Pin
interrupt control circuit
interrupt singal (INTTMmn)

Note: On channels 1 and 3, clocks can be selected from CKm0, CKm1, CKm2, and CKm3.

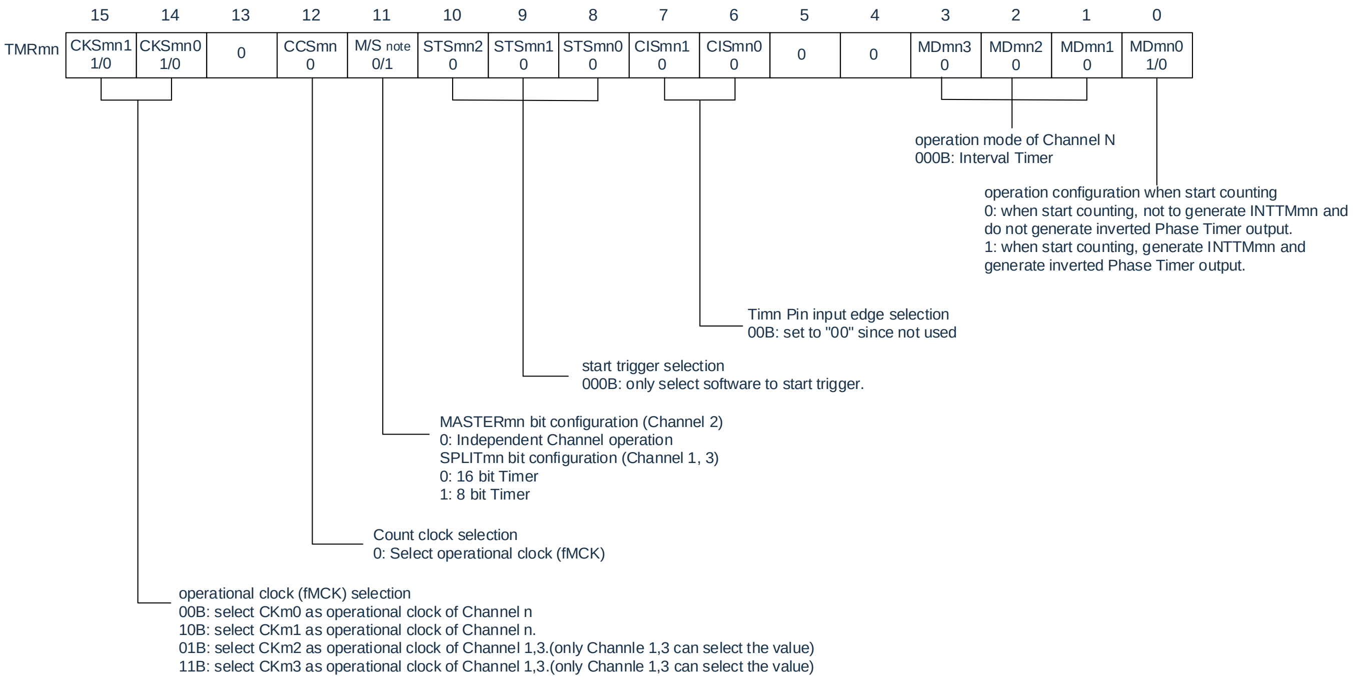
Figure 5-43 Basic timing example of operation as a spacer timer / square wave output (MDmn0=1)



Note 1. m: unit number (m= 0,1) n: channel number (n=0 ~ 3).
2. TSmn: bit n of timer channel start register m (TSm).
TEmn: timer channel enable bit n of status register m (TEm).
TCRmn: timer count register mn (TCRmn).
TDRmn: timer data register mn (TDRmn).
TOmn: TOmn pin output signal.

Figure 5-44 Example of register setting contents 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 n
TOm	TOmn 1/0

0: output “0” by TOmn.
1: output “1” by TOmn.

(c) Timer output enable register m (TOEm)

	bit n
TOEm	TOEmn 1/0

0: Stop the TOmn output by the count run.
1: Allow the TOmn output by the count run.

(d) Timer output level register m (TOLm)

	bit n
TOLm	TOLmn 0

0: “0” is set at TOMmn=0 (master channel output mode).

(e) Timer output mode register m (TOMm)

	bit n
TOMm	TOMmn 0

0: Set the master channel output mode.

Note: TMRm2: MASTERmn bit.
TMRm1, TMRm3: SPLITmn bit.
TMRm 0: fixed to “0”.

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

Figure 5-45 Operation 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 TM4mEN bit of 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 ~ 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: Set the TOMmn bit of the timer output mode register m (TOMm) to "0" (master channel output mode). Set the TOLmn bit to "0". Set the TOmn bit to determine the initial level of the TOmn output. Set TOEmn bit to "1" to enable 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 Run	(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 a 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.
Running	You can change the setting values of the TDRmn register at will. It can read TCRmn register at any time. TSRmn register cannot be used. It can change the TOm register and TOEm register settings. Prevents the setting of the TMRmn register, TOMmn bit, and TOLmn bit from being changed.	The counter (TCRmn) counts down. If the count goes to "0000H", the value of the TDRmn register is loaded again into the TCRmn register and counting continues. When TCRmn is detected as "0000H", INTTMmn is generated and TOmn is output interleaved. This run is repeated thereafter.
Stop Running	Set TTmn (TTHm1, TTHm3) to "1". Automatically returned to '0' because the TTmn (TTHm1, TTHm3) bit is a 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 to "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 the 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 TM4mEN 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

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

5.8.2 Operate as external event counter

It can be used as an event counter to count the detected valid edges (external events) of the TImn pin input, and if the specified count value is reached, an interrupt is generated. The specified count values can be calculated using the following calculation formula:

Specified counting value = set value of TDRmn + 1

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

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

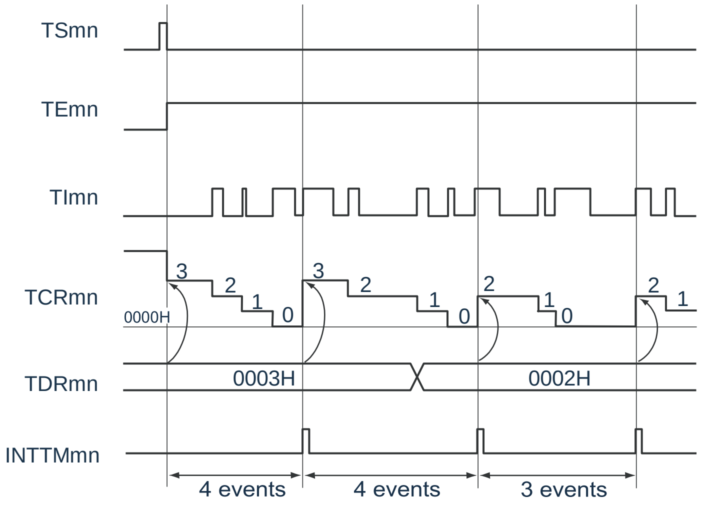
The TCRmn register decrements the count while detecting the valid edge of the TImn pin input. If the TCRmn changes to "0000H", the value of the TDRmn register is loaded again and INTTMmn is output.

After that, continue with the same run.

Because the TOmn pin outputs irregular waveforms based on external events, the TOEmn bit of timer output enable register m (TOEm) must be set to "0" to stop the output.

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

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



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

2. TSmn: Bit n of the timer channel start register m (TSm).
TEmn: Timer channel enable bit n of status register m (TEm).
TImn: Input signal of the TImn pin.
TCRmn: Timer counter register mn (TCRmn).
TDRmn: Timer data register mn (TDRmn).

Figure 5-47 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 n
TOm	TOmn 1/0

0: Output “0” by TOmn.

(c) The timer output enable register m (TOEm).

	bit n
TOEm	TOEmn 1/0

0: Stop the TOmn output by the count run.
1: Enable the TOmn output by the count run.

(d) Timer output level register m (TOLm).

	bit n
TOLm	TOLmn 0

0: “0” is set at TOMmn=0 (master channel output mode).

(e) Timer outputs mode register m (TOMm).

	bit n
TOMm	TOMmn 0

0: Set the master channel output mode.

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

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

Figure 5-48 Opeation 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 TM4mEN 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. Output timer to allow TOEmn location “0” for register m (TOEm).	The channel is in an operational stop state. (Provide clocks, consume a portion of the Power)
Start Run	Set TSmn 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.
Running	You can change the settings of the TDRmn register at will. Can read TCRmn register at any time. The TSRmn register is not used. Prevents the setting of TMRmn registers, TOMmn bits, TOLmn bits, TOmn bits, and TOEmn bits from being changed.	The counter (TCRmn) counts down each time an input edge of the TImn pin is detected, and if the count reaches '0 000H', loads the value of the TDRmn register again into the TCRmn register and continues counting. A INTTMmn is generated when TCRmn is detected as '0000H'. This run is repeated thereafter.
Stop Running	Set TTmn 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

5.8.3 Operates as frequency divider

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

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

- Selecting the case of rising or falling edge:
  Divided clock frequency = Input clock frequency / {(set value of TDR00+ 1×2}
- For selecting double edges:
  Divided clock frequency ≈Input clock frequency / (set value of TDR00 + 1)

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

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

The TCR00 register is then decremented by the active edge of the TI00 pin input. If TCR00 becomes “0000H”, TO00 alternates the output. At the same time, the value of the TDR00 register is loaded into the TCR00 register and the count continues.

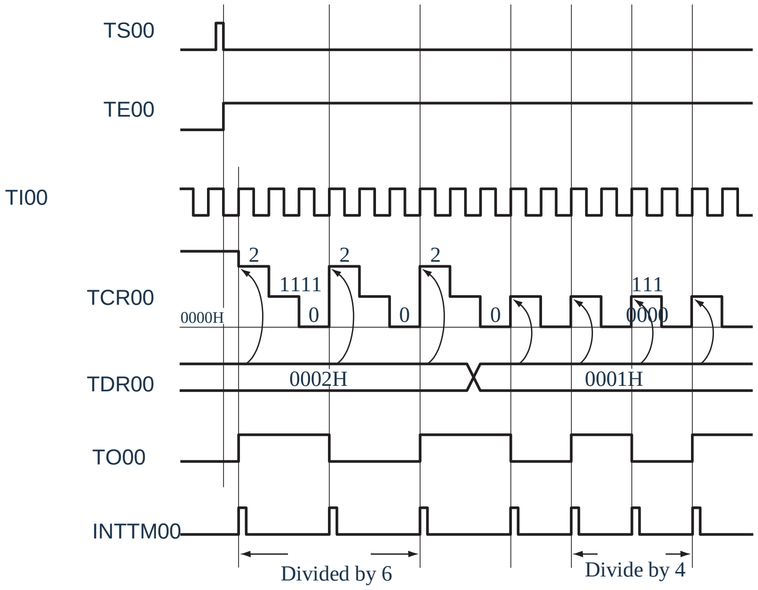
If you select the detection of the bilateral edge of the TI00 pin input, the duty cycle error of the input clock affects the divider clock period of the TO00 output.

The clock cycle of the TO00 output contains a sampling error of 1 running clock cycle.

TO00 output clock period = ideal TO00 output clock period ± run clock period (error)

The TDR00 register can be overridden at any time, and the value of the overridden TDR00 register is valid for the next count period.

Figure 5-49 Example of basic timing as a frequency divider operation (MD000=1)



Note TS00: Bit0 of timer channel start register 0 (TS0)

TE00: Bit0 of timer channel enable for status register 0 (TE0).

TI00: TI00 pin input signal.

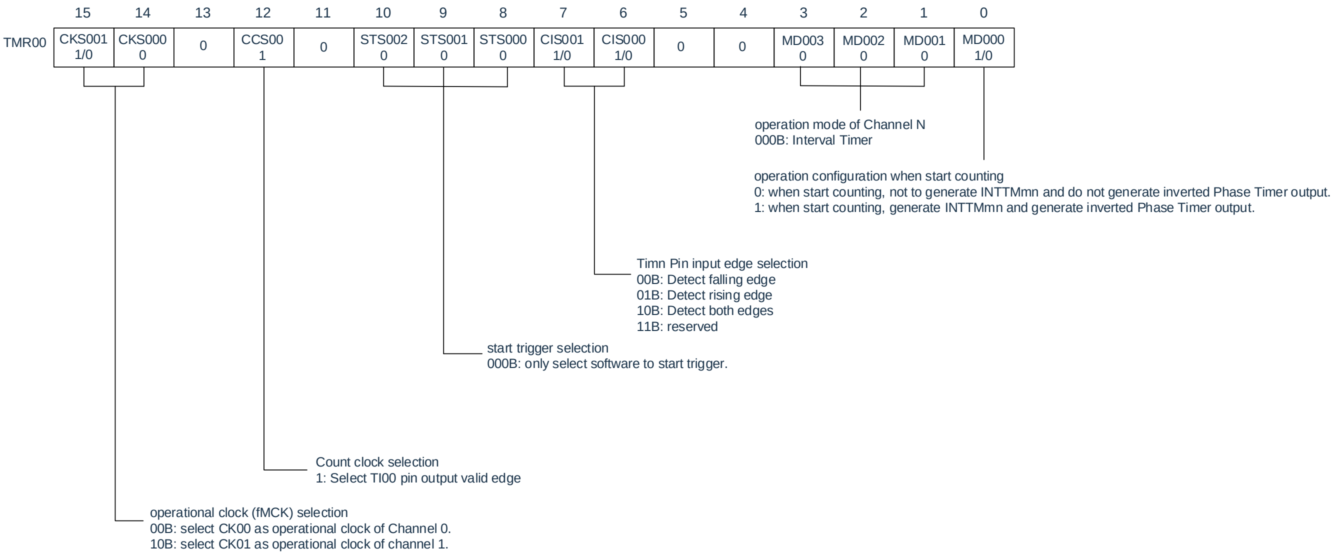
TCR00: Timer counter register 00 (TCR00).

TDR00: Timer data register 00 (TDR00).

TO00: TO00 pin output signal.

Figure 5-50 Example of register setting content for operation as a frequency divider

(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” by TO00.

(c) Timer output enable register 0 (TOE0).

	bit 0
TOE0	TOE00 1/0

0: Stop TO00 output by the count run.
1: EnableTO00 output by the count run.

(d) Timer output level register 0 (TOL0).

	bit 0
TOL0	TOLL00 0

0: Set “0” in the master channel output mode (TOM00=0).

(e) Timer output mode register 0 (TOM0).

	bit n
TAT0	TOM00 0

0: Set the master channel output mode.

Figure 5-51 Opeation procedure for frequency divider function

	Software operation	Hardware state
Timer 4 initial configuration		Timer Unit 0 input clock is in stopped state (stop providing clock, not able to write into registers)
	Set TM4mEN bit of peripheral enable register 0 (PER0) to '1'	Timer Unit 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

5.8.4 Operates as input pulse interval measurement

Counts can be captured at TImn effective edges and the interval between TImn input pulses can be measured. During the TEmn bit of “1”, the software operation (TSmn=1) can also be set to capture the trigger to capture the count value.

The pulse interval can be calculated using the following calculation formula:

TImn input pulse interval = cycles of the counting hour × ((100 00H × TSRmn: OVF) + (catch value of TDRmn +1)).

Note: Because the TImn pin input is sampled by the operating clock selected by the CKSmn bit of the timer mode register mn (TMRmn), an error of 1 run clock is generated.

In capture mode, the timer count register mn (TCRmn) is used as an increment counter.

If the channel start triggers bit (TSmn) of the timer channel start register m (TSm) is set to “1”, the TCRmn register is clocked from “0000H” by counting the clock Start incrementing the count.

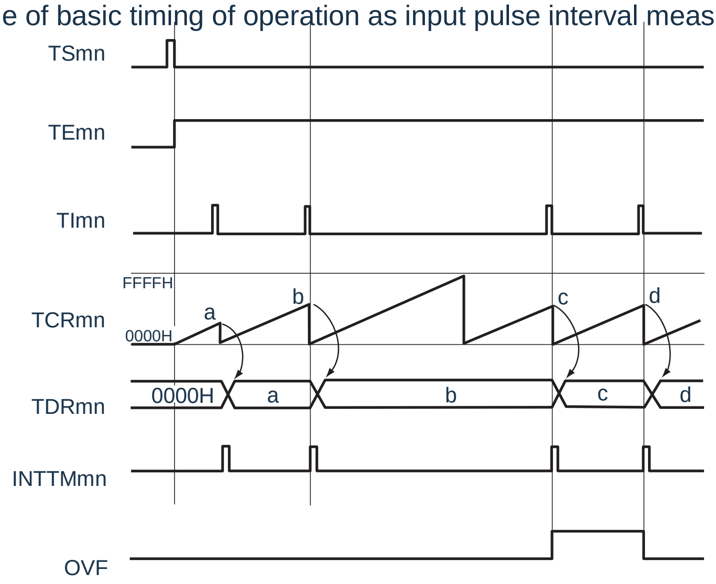
If a valid edge of the TImn pin input is detected, the count value of the TCRmn register is transferred (captured) to the timer data register mn (TDRmn) and the TCRmn register is cleared” 0000H”, and then output INTTMmn. At this point, if the counter overflows, the OVF position of the timer status register mn (TSRmn) is “1”. If the counter does not overflow, the OVF bit is cleared. After that, continue with the same run.

While snapping the count value to the TDRmn register, the OVF bit of the TSRmn register is updated depending on whether an overflow occurred during the measurement and the overflow state of the captured value can be confirmed.

Even if the counter performs a full count of 2 cycles or more, it is considered that an overflow occurs and the OVF position of the TSRmn register is “1”. However, in the event of 2 or more overflows, the interval value cannot be measured normally by the OVF bit.

Set the STSmn2 to STSmn0 bits of the TMRmn register to “001B” and use the valid edges of TImn for start trigger and capture trigger.

Figure 5-52 Example of basic timing of operation as input pulse interval measurement (MDmn0=0).



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

2. TSmn: Bit n of timer channel start register m (TSm)

TEmn: Bit n of timer channel enable status registerm (TEm).

TImn: TImn pin input signal.

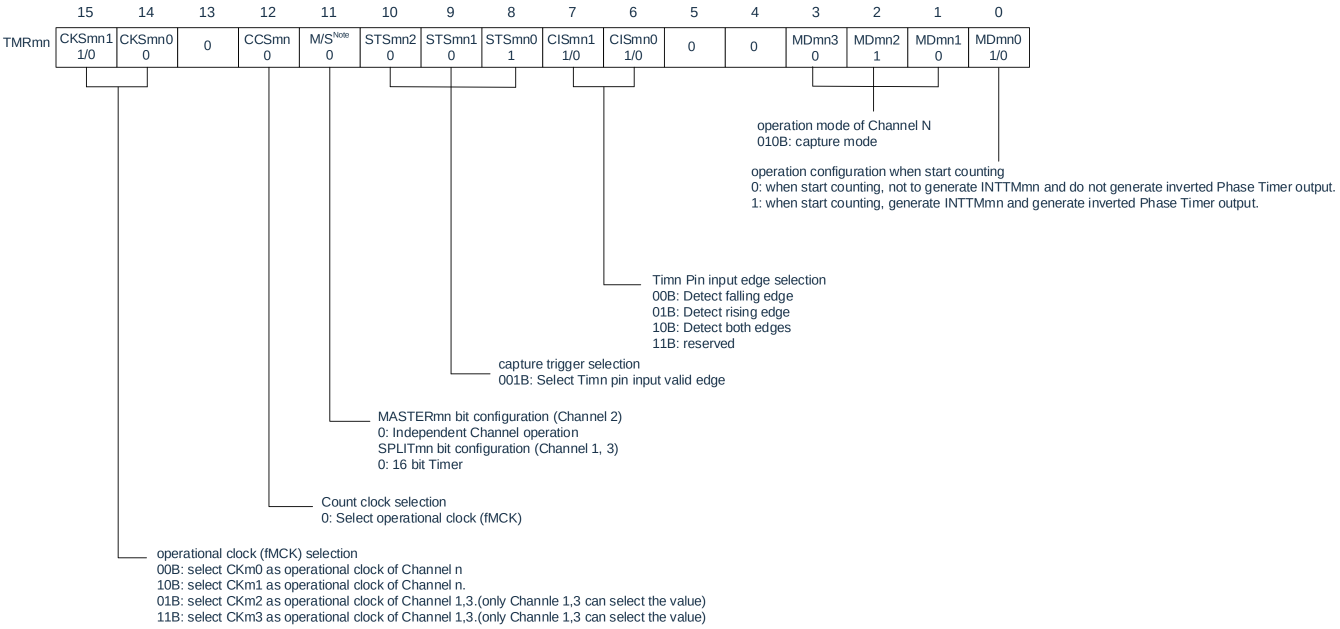
TCRmn: Timer counter register mn (TCRmn).

TDRmn: Timer data register mn (TDRmn).

OVF: Bit 0 of the timer status register mn (TSRmn).

Figure 5-53 Example of register settings when measuring input pulse intervals

(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^{Note} 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 n
TOm	TOmn 0

0: Output “0” by TOmn.

(c) The timer output enable register m (TOEm).

	bit n
TOEm	TOEmn 0

0: Stop the TOmn output by the count run.

(d) Timer output level register m (TOLm).

	bit n
TOLm	TOLmn 0

0: Set “0” in the master channel output mode (TOMmn=0).

(e) Timer outputs mode register m (TOMm).

	bit n
TOMm	TOMmn 0

0: Set the master channel output mode.

Note TMRm2: MASTERmn bit.
TMRm1, TMRm3: SPLITmn bit.
TMRm0: Fixed to “0”.
Notem: Unit number (m=0) n: Channel number (n=0 ~ 3).

Figure 5-54 Operation procedure when entering the pulse interval measurement 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.
	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)

restart operation

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

5.8.5 Operation as input signal high and low level width measurement

Notice: When used as a LIN-bus support function, bit1 (ISC1) of the input switch control register (ISC) must be set to "1", and in the following instructions, RxD0 must be used Instead of TImn.

TImn's signal width (high and low level widths) can be measured by starting counting on one edge input at the TImn pin and capturing the count value on the other edge. The signal width of TImn can be calculated using the following calculation formula.

TImn input signal width = period of the counting hour clock × ((100 00H × TSRmn: OVF) + (capture value of TDRmn +1)).

Notice: Because the TImn pin input is sampled by the operating clock selected by the CKSmn bit of the timer mode register mn (TMRmn), an error of 1 run clock is generated.

In Capture & Single Count mode, the timer count register mn (TCRmn) is used as an incremental counter. If the channel start trigger bit (TSmn) of the timer channel start register m (TSm) is set to "1", the TEmn bit becomes "1", and enters the start edge of the TImn pin to detect the wait state.

If the start edge of the TImn pin input (the rising edge of the TImn pin input when measuring the high width) is detected, the count is incremented from "0000H" onwards. Then, if a valid capture edge (the falling edge of the TImn pin input when measuring the width of the high) is detected, the INTTMmn is output at the same time as the count value is transferred to the timer data register mn (TDRmn). At this point, if the counter overflows, the OVF position bit of the timer status register mn (TSRmn) is placed. If the counter does not overflow, the OVF bit is cleared. The value of the TCRmn register becomes +1 and the value transferred to the TDRmn register stops counting and enters the start edge of the TImn pin to detect a wait state. After that, continue with the same run.

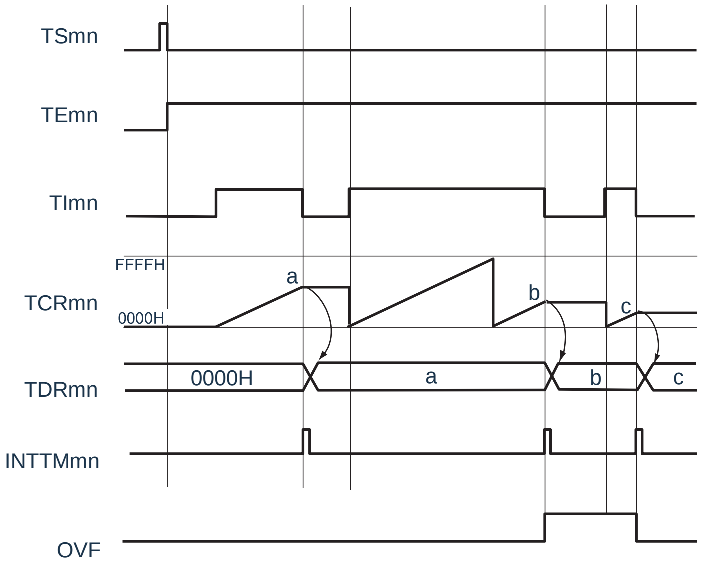
While snapping the count value to the TDRmn register, update the OVF bit of the TSRmn register based on whether an overflow occurred during the measurement, and confirm the overflow state of the captured value.

Even if the counter performs a full count of 2 cycles or more, it is considered that an overflow occurs and the OVF bit of the TSRmn register is "1". However, in the event of 2 or more overflows, the interval value cannot be measured normally by the OVF bit.

The CISmn1 and CISmn0 bits of the TMRmn register can be used to set whether the high or low width of the TImn pin is measured. This feature is intended to measure the input signal width of the TImn pin. Therefore, you cannot set the TSmn position to "1" during the period when the TEmn bit is "1".

CISmn1, CISmn0=10B of TMRmn register: Measure the width of the low level.
CISmn1, CISmn0=11B of TMRmn register: Measure the width of the high level.

Figure 5-55 Example of basic timing for operation as input signal high and low level width measurement



Note 1. m: unit number (m= 0) n: channel number (n=0 ~ 3).
2. TSmn: bit n of timer channel start register m (TSm).
TEmn: bit n of timer channel enable status register m (TEm).
TImn: TImn pin input signal.
TCRmn: Timer counter register mn (TCRmn).
TDRmn: Timer data register mn (TDRmn).
OVF: bit0 of timer status register mn (TSRmn).

Figure 5-56 Example of register settings when measuring the high and low level widths of an 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 Note 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
010B: 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 n
TOm	TOmn 0

0: Output “0” by TOmn.

(c) Timer output enable register m (TOEm).

	bit n
TOEm	TOEmn 0

0: Stop the TOmn output by the count run.

(d) Timer output level register m (TOLm).

	bit n
TOLm	TOLmn 0

0: “0” in the master channel output mode (TOMmn=0).

(e) Timer output mode register m (TOMm).

	bit n
TOMm	TOMmn 0

0: Set the master channel output mode.

Note: TMRm2: MASTERmn bit
TMRm1, TMRm3: SPLITmn bit
TMRm0: Fixed to “0”.
Remark: m: Unit number (m=0) n: Channel number (n=0 ~ 3).

Figure 5-57 Operation procedure for the input signal high and low level width measurement function

	software operation	hardware state
Timer 4 initial configuration		Timer Unit 0 input clock is in stopped state (stop providing clock, not able to write into registers)
	set TM4mEN bit of peripheral enable register 0 (PER0) to '1' →	Timer Unit m input clock is in active state, all channels in operation stopped state.
	configure Timer clock selection register m(TPSm), confirm CKm0~CKm3 clock frequency	
Channel Initial configuration	set corresponding bit of noise filter enable register 1 (NFEN1) to '0' (OFF) or '1' (ON). Configure Timer mode register mn (TMRmn) (confirm channel operation mode). Set T0Emn bit to '0', and stop T0mn operation.	channel in operation stopped state (providing clock, consume portion of power)
Start operation	set TSmn bit to '1'. → 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

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

5.8.6 Operation as delay counter

The decreasing count can be started by a valid edge detection (external event) at the TImn pin input and the INTTMmn is generated at any set interval (Timer interrupt).

During the period when the TEmn bit is “1”, it is possible to set the TSmn bit to “1” by software to start decreasing the count and generate INTTMmn (timer interrupt) at any set interval.

The interrupt generation period can be calculated using the following equation:

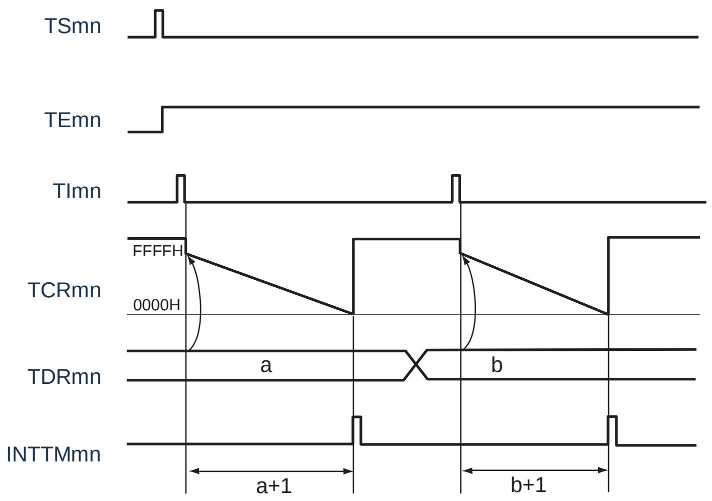
INTTMmn (timer interrupt) generation period = period of counting clock   (setting value of TDRmn + 1)

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

If the channel start trigger bits (TSmn, TSHm1, TSHm3) of the Timer Channel Start Register m (TSm) are set to “1”, the TEmn bit, TEHm1 bit and TEHm3 bit becomes “1” and enters the valid edge detection wait state of the TImn pin. The valid edge detection by the TImn pin input starts the operation of the TCRmn register and loads the value of the Timer Data Register mn (TDRmn). The TCRmn register starts counting decreasingly from the value of the loaded TDRmn register by counting the clock. If TCRmn changes to “0000H”, INTTMmn is output and counting stops before a valid edge is detected for the next TImn pin input.

TDRmn registers can be overridden at any time, and the values of the overridden TDRmn registers are valid from the next cycle.

Figure 5-58 Example of basic timing of operation as a delay counter

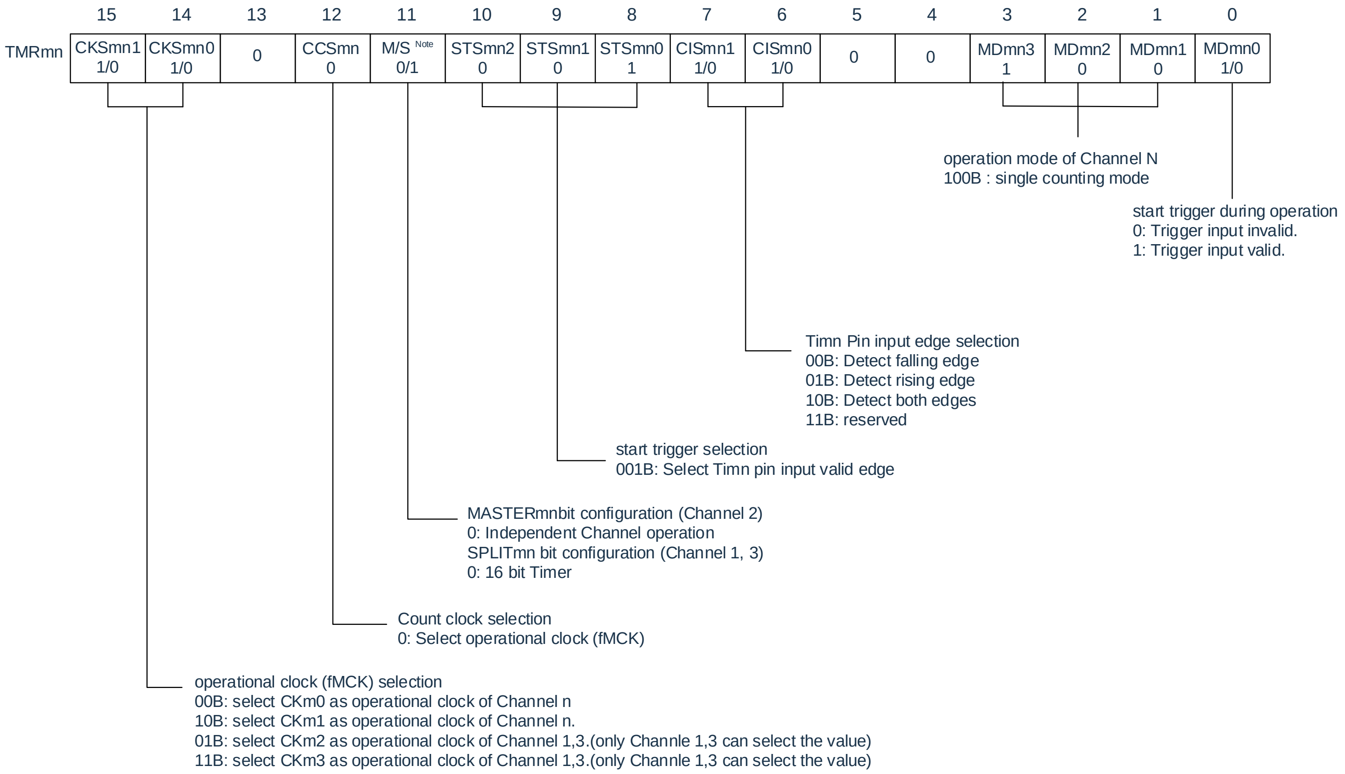


Note 1. m: unit number (m= 0,1) n: channel number (n=0 ~ 3).

2. TSmn: Bit n of the timer channel start register m (TSm).
TEmn : Bit n of timer channel enable status register m (TEm).
TImn : TImn pin input signal
TCRmn: Timer counter register mn (TCRmn).
TDRmn: Timer data register mn (TDRmn).

Figure 5-59 Example of register setting contents 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 Note 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 n
TOm	TOmn 0

0: Output “0” by TOmn.

(c) The timer output enable register m (TOEm).

	bit n
TOEm	TOEmn 0

0: Stop the TOmn output by the count run.

(d) Timer output level register m (TOLm).

	bit n
TOLm	TOLmn 0

0: “0” in the master channel output mode (TOMmn=0).

(e) Timer outputs mode register m (TOMm).

	bit n
TOMm	TOMmn 0

0: Set the master channel output mode.

Note TMRm2: MASTERmn bit.
TMRm1, TMRm3: SPLITmn bit.
TMRm0: Fixed to “0”.
Notem: Unit number (m=0) n: Channel number (n=0 ~ 3).

Figure 5-60 Operation procedure for 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 (TCR00) performs decremental counting. When TCRmn count reaches '0000H', then generate INTTMmn and before detecting the next start trigger (detect TImn pin input valid edge or set TSmn bit to '1'), TCRmn is "0000H" and stop counting.
stop operation	set TTmn bit to '1'. Because TTmn bit is trigger bit, thus automatically return to '0'.	TEmn bit turns into '0' and stop counting. TCRmn register hold counted value and stop counting.
Timer 4 stop	set TM4mEN bit of peripheral enable register 0 (PER0) to '0'	Timer Unit m input clock is not been provided.Perform initialization to all circuit and SFR of all channels.

restart operation

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

5.9 Multi-channel linkage operation of the universal timer unit

5.9.1 Operates as single-trigger pulse output function

Using 2 channels in pairs, a single-trigger pulse with any delay pulse width can be generated from the input of the TImn pin. Delay and pulse width can be calculated using the following formula:

Delay = {set value of TDRmn (master) +2} × Counting clock period
Pulse width = {set value of TDRmp (slave)} × Counting clock period

In single-count mode, the master channel runs and counts the latency. By detecting the start trigger, the timer count register mn (TCRmn) of the master channel starts to run and loads the value of the timer data register mn (TDRmn). The TCRmn register decrements the count from the value of the loaded TDRmn register by counting the clock. If the TCRmn becomes “0000H”, INTTMmn is output and the count is stopped before the next start trigger is detected.

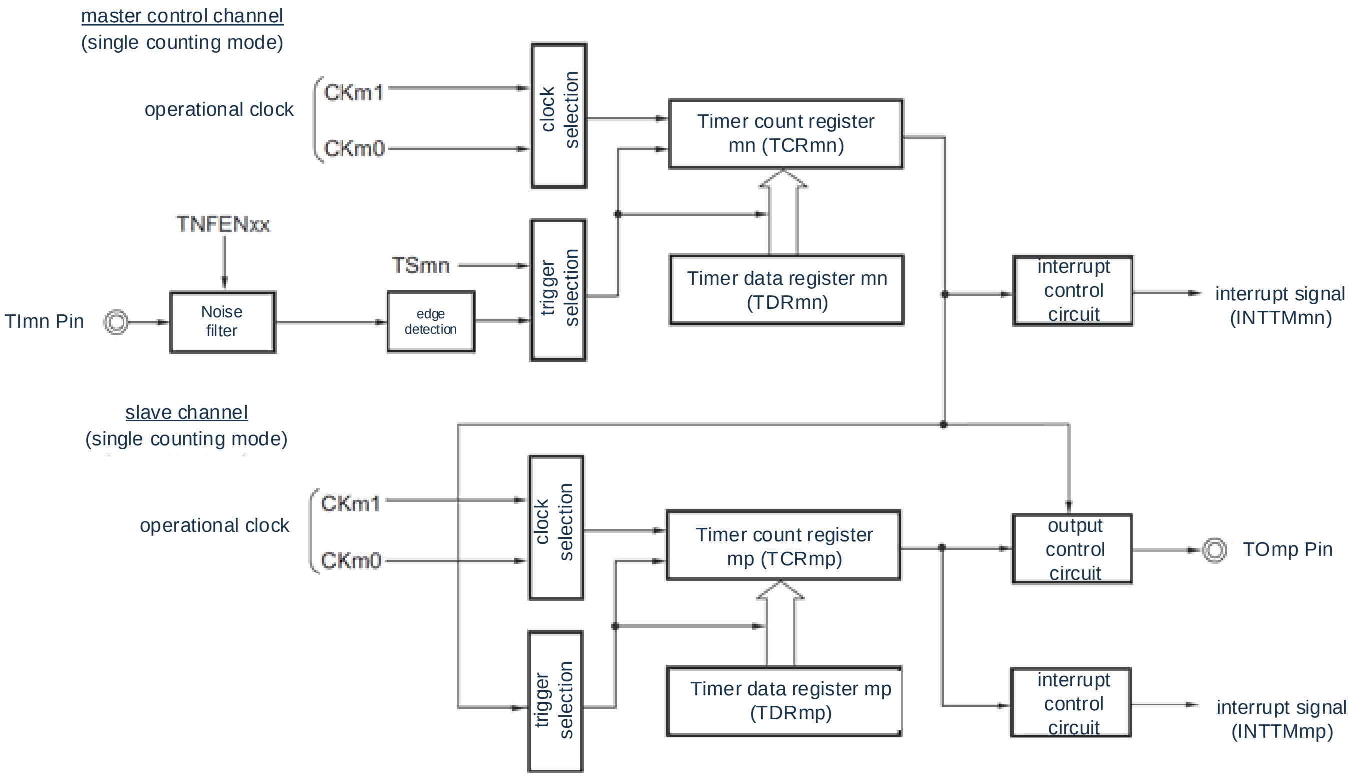
In single-count mode, the slave channel runs and the pulse width is counted. With the INTTMmn of the master channel as the start trigger, the TCRmp register of the slave channel starts running and the value of the TDRmp register is loaded. The TCRmp register decrements the count from the loaded TDRmp register value by counting the clock. If the count value changes to “0000H”, INTTMmp is output and the count is stopped before the next start trigger (INTTMmn of the master channel) is detected. After the INTTMmn is generated from the master channel and 1 count clock has passed, the output level of TOmp becomes effective if the TCRmp becomes “0000H”, it becomes an invalid level.

Software operation (TSmn=1) can be used as a start trigger to output a single trigger without using the TImn pin input.

Notice: Because the TDRmn registers of the master channel and the TDRmp registers of the slave channels have different loading timings, if you overwrite the TDRmn registers and TDRmp registers during the counting process, you may compete with the load timing and output abnormal waveforms. The TDRmn registers must be overwritten after INTTMmn is generated and TDRmp register after INTTMmp is generated.

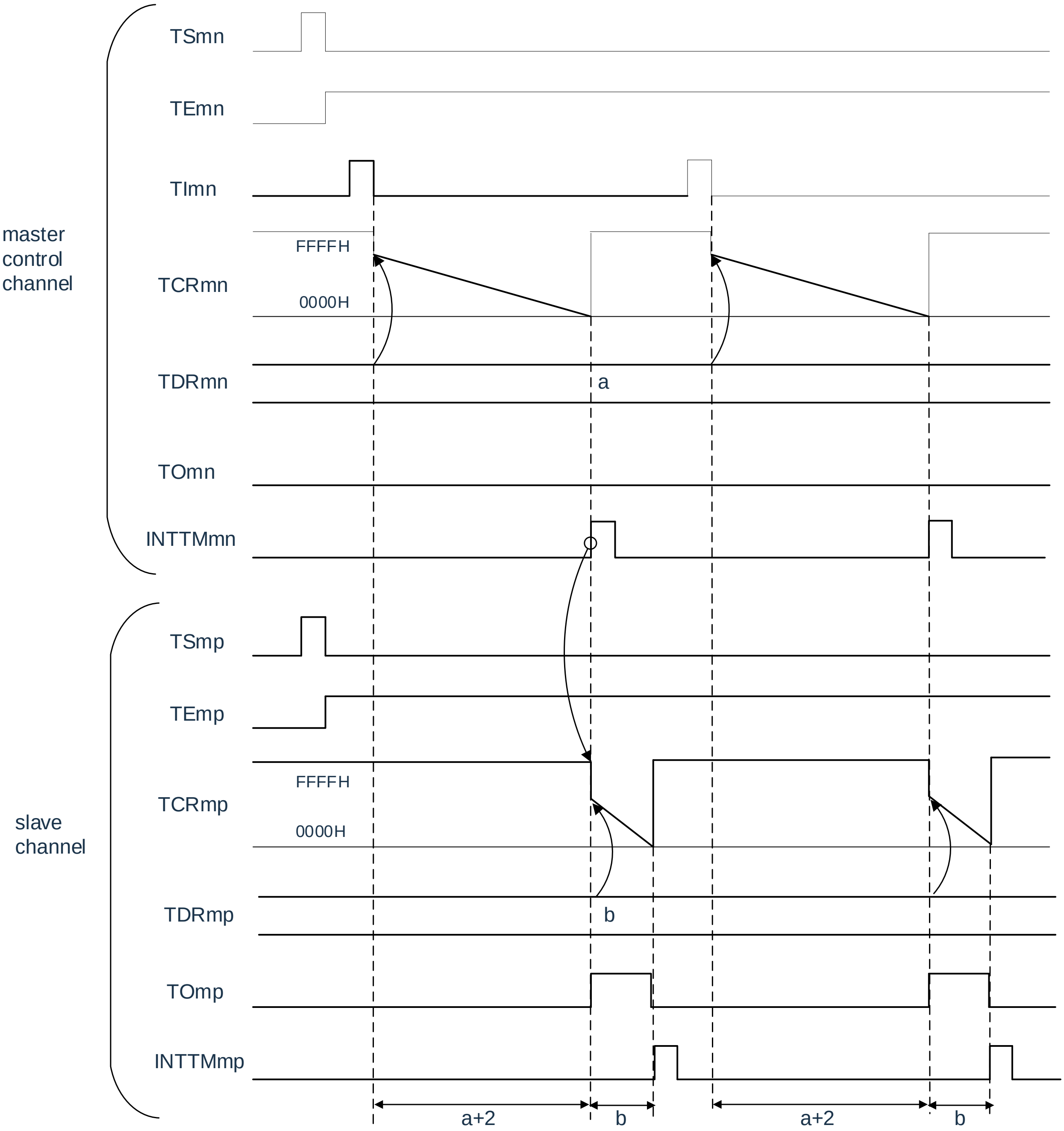
Notem: Unit number (m= 0,1) n: master channel number (n=0, 2) p: slave channel number (n= 0： p=1, 2, 3, n=2: p=3)

Figure 5-61 Block diagram of operation as a single-trigger pulse output function



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

Figure 5-62 Example of basic timing for operation as a single trigger pulse output function

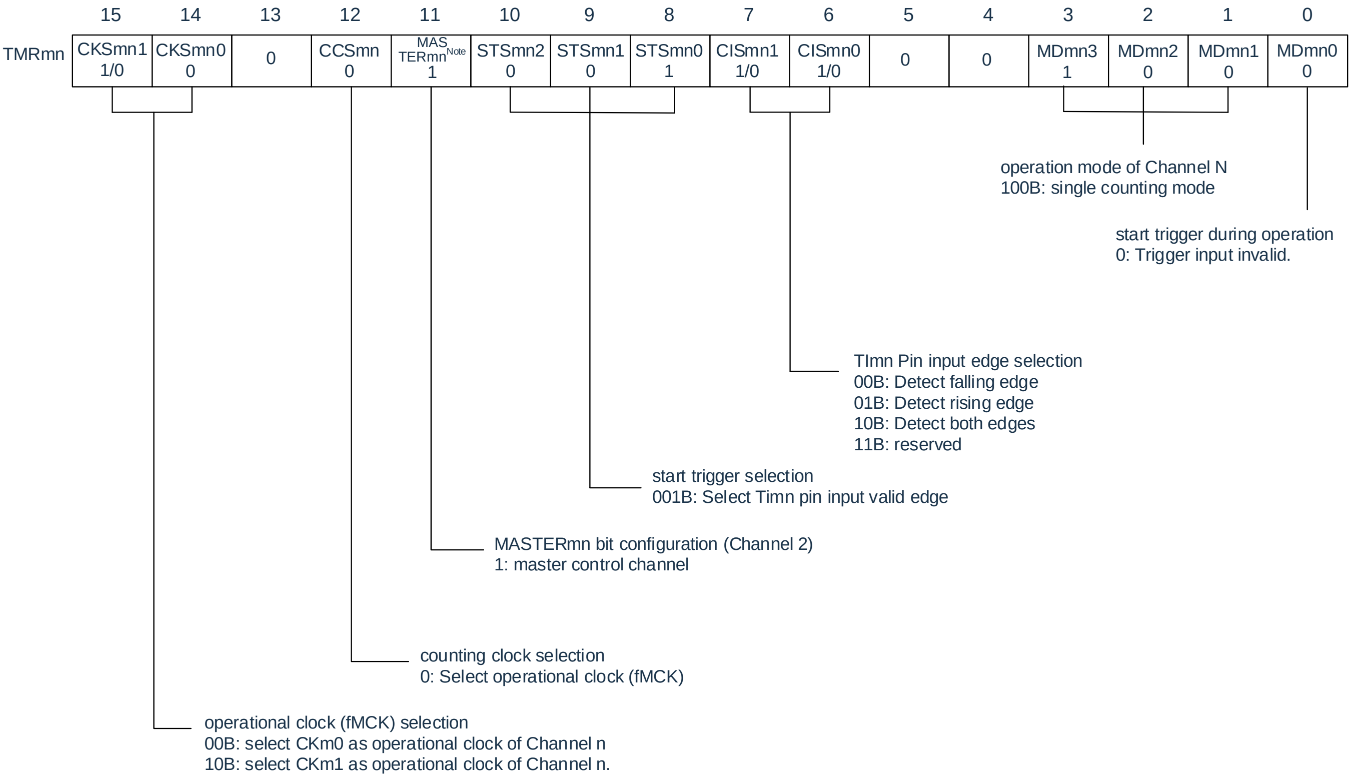


Note 1. m: Unit number (m= 0,1) n: master channel number (n=0, 2) p: slave channel number (n=0: p=1, 2, 3, n=2: p=3)

2. TSmn, TSmp: Bit n, p of timer channel start register m (TSm)
TEmn, TEmp: Bit n, p of timer channel enable status registers m (TEm)
TImn, TImp: TImn and TImp pin input signals
TCRmn, TCRmp: Timer count register mn, mp (TCRmn, TCRmp).
TDRmn, TDRmp: Timer data register mn, mp (TDRmn, TDRmp).
TOmn, TOmp: TOmn and TOmp pin output signals.

Figure 5-63 Example of register settings for 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	MASTERmn[Note] 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).

TOm — bit n: TOmn 0 — 0: Output “0” by TOmn.

(c) The timer output enable register m (TOEm).

TOEm — bit n: TOEmn 0 — 0: Stop TOmn output by the count run.

(d) Timer output level register m (TOLm).

TOLm — bit n: TOLmn 0 — 0: “0” is set at TOMmn=0 (master channel output mode).

(e) Timer outputs mode register m (TOMm).

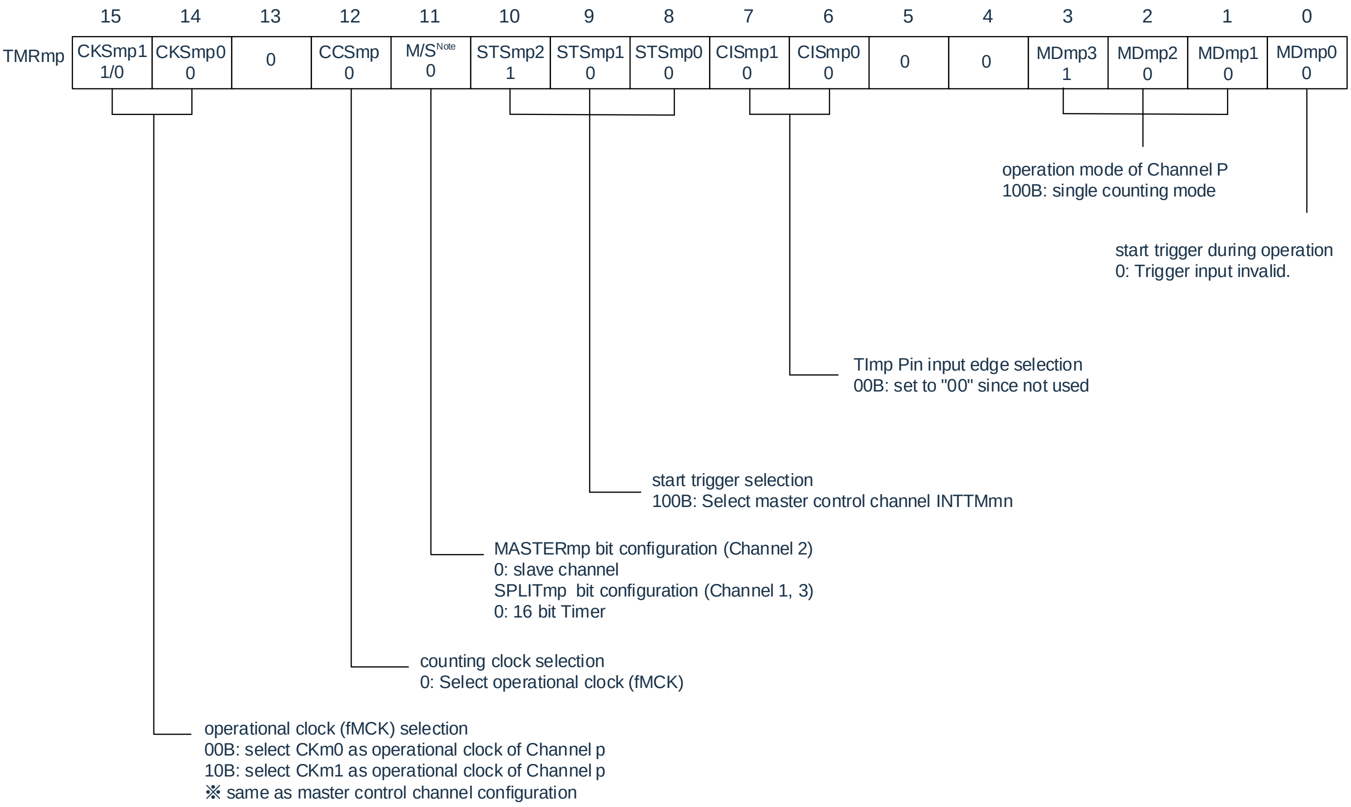
TOMm — bit n: TOMmn 0 — 0: Set the master channel output mode.

concentrate TMRm2 : MASTERmn=1
TMRm0 : Fixed to “0”.

Note m: Unit number (m= 0,1) n: master channel number (n=0, 2).

Figure 5-64 Example of register settings (slave channels) for single-trigger pulse output functions

(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[Note] 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” by TOmp.
1: Output “1” by TOmp.

(c) The timer output enable register m (TOEm).

	bit p
TOEm	TOEmp 1/0

0: Stop TOmp output from counting runs.
1: Enable TOmp output by counting runs.

(d) Timer output level register m (TOLm).

	bit p
TOLm	TOLmp 1/0

0: Positive logic output (active high).
1: Negative logic output (active-low).

(e) Timer outputs mode register m (TOMm).

	bit p
TOMm	TOMmp 1

1: Set the slave channel output mode.

Note TMRm2: MASTERmp bit
TMRm1, TMRm3: SPLITmp bit

Note:m: Unit number (m= 0,1) n: master control channel number (n=0, 2) p: slave channel number (n= 0: p=1, 2, 3, n=2: p=3)

Figure 5-65 Operation procedure for single trigger pulse output function (1/2)

	software operation	hardware state
Timer 4 initial configuration		Timer Unit m input clock is in stopped state (stop providing clock, not able to write into registers)
	set TM4mEN bit of peripheral enable register 0 (PER0) to '1'	Timer Unit m input clock is in active state, all channels in operation stopped state. (start providing clock, Start to provide clock, can write to each register)
	configure Timer clock selection register m(TPSm), confirm CKm0~CKm3 clock frequency	
Channel Initial configuration	set corresponding bit of noise filter enable register 1 (NFEN1) to 1'. Configure Timer mode registers mn,mp of 2 channels (TMRmn, TMRmp) (confirm channel operation mode). Set master control channel Timer data register mn (TDRmn) configure output delay time, and set slave channel TDRmp register pulse width.	channel in operation stopped state (providing clock, consume portion of power)
	slave channel configuration set TOMmp bit of timer output mode register m(TOMm) to '1' (slave channel output mode). Configure TOLmp bit. Configure TOmp bit and confirm TOmp otuput initial voltage. Set TOEmp bit to '1', enable TOmp output. Set port regsiter and port mode regsiter to '0'.	T0mp pin in Hi-Z output state. When port mode register set to output mode and port register as '0', output T0mp initial configured voltage level. Because channel is in operation stopped state, thus T0mp remains unchange. T0mp pin output T0mp configured voltage level.

Figure 5-66 Operation procedure for 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'.	

Note m: unit number (m= 0,1) n: master channel number (n=0).
p: slave channel number q: slave channel number
n < p < q ≤3 (p and q are integers greater than n)

5.9.2 Operates as PWM function

Using 2 channels in pairs, pulses of any period and duty cycle can be generated. The period and duty cycle of the output pulse can be calculated using the following calculation formula:

Pulse Pulse Cycle = {set value of TDRmn (master) +1} ×counting clock period
Duty cycle [%] = {set value of TDRmp (slave)} / {set value of TDRmn (master) +1} ×100 % input: set value of TDRmp (slave) = 0000H
100% input: set value of TDRmp (slave). ≥ {set value of TDRmn (master) +1}

Note When the config value of TDRmp (slave) > the config value of {TDRmn (master) +1}, the duty cycle exceeds 100%, but it is 100% output.

The master channel is used as an interval timer mode. If the channel starts triggerring bit (TSmn) of the timer channel start register m (TSm) is set to “1”, the output interrupt (INTTMmn) is then loaded into the timer count register mn by the config value of the timer data register mn (TDRmn), and the count is decremented by counting the clock. When counted to “0000H”, the value of the TDRmn register is loaded into the TCRmn register again after the INTTMmn is output, and the count is decremented. Thereafter, repeat this operation before setting the channel stop touch bit (TTmn) of the timer channel stop register m (TTm) to “1”.

When used as a PWM function, the master channel is decremented in count for PWM output (TOmp) cycles during the period counted to “0000H”. The slave channel is used as a single-count mode. Starting with the INTTMmn of the master channel, the value of the TDRmp register is loaded into the TCRmp register and decremented counting until “0000H”. When counted to “0000H”, INTTMmp is output and waits for the next ON

Initial trigger (INTTMmn for the master channel).

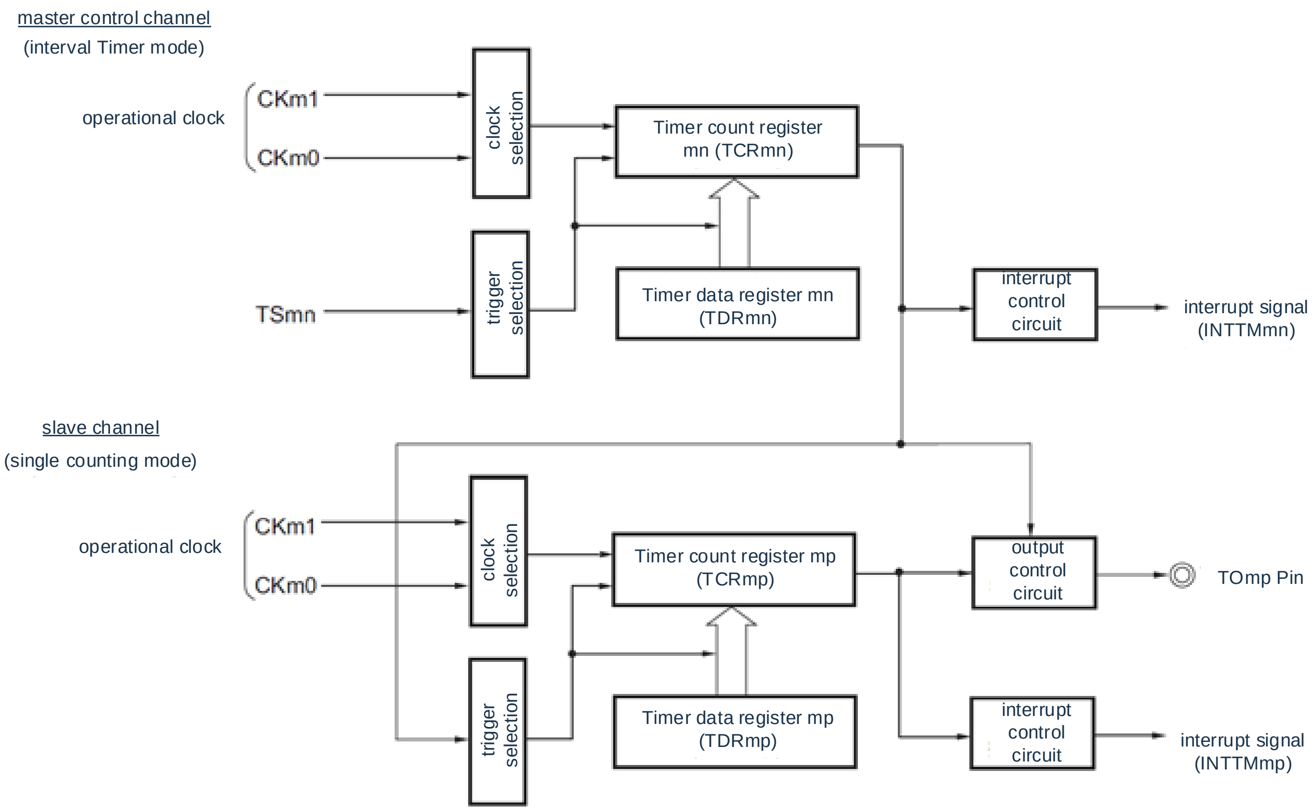
When used as a PWM function, the slave channel is decremented and counted as the duty cycle of the PWM output (TOmp) for the period counted to “0000H”.

After generating INTTMmn from the master channel and passing 1 clock, the PWM output (TOmp) becomes effective and the value of the TCRmp register on the slave channel is “0000H” becomes an invalid level.

Notice: Two write accesses are required to overwrite both the timer data register mn (TDRmn) of the master channel and the TDRmp register of the slave channel. Because the values of the TDRmn register and TDRmp register are loaded into the TCRmn register and the TCRmp register when the master channel generates INTTMmn. In this case, if INTTMmn is rewritten before and after the main channel is generated, the TOmp pin cannot output the expected waveform. Therefore, to overwrite both the TDRmn registers of the master and the TDRmp registers of the slave, the two registers must be overwritten immediately after the INTTMmn is generated on the master channel.

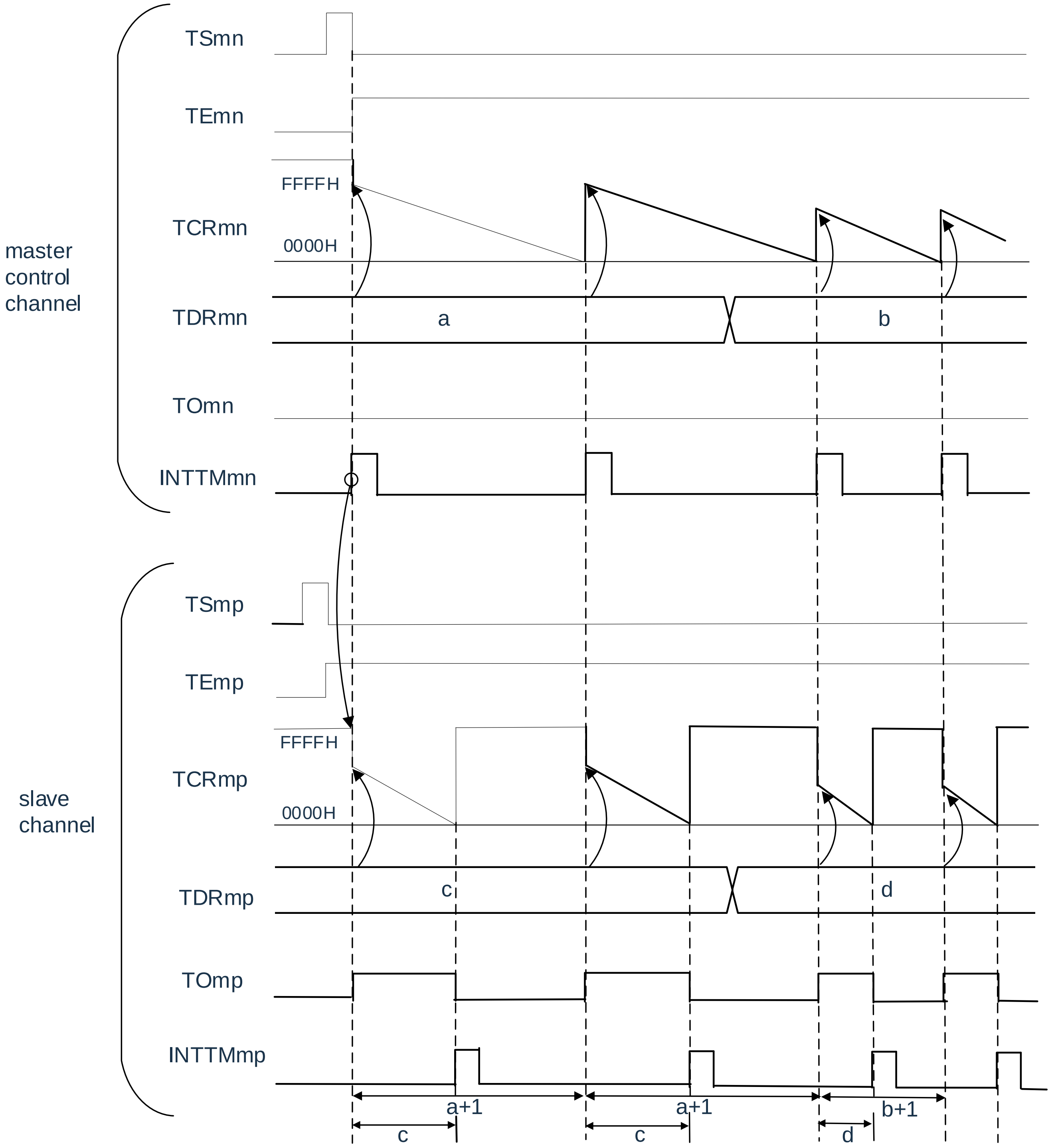
Note m: Unit number (m= 0,1) n: master channel number (n=0, 2) p: slave channel number (n= 0: p=1, 2, 3, n=2: p=3)

Figure 5-67 Block diagram of operation as a PWM function



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

Figure 5-68 Example of basic timing for operation as a PWM function



Note 1. m: Unit number (m= 0,1) n: master channel number (n=0, 2) p: slave channel number (n=0 : p=1, 2, 3, n=2: p=3)

2. TSmn, TSmp: Bit n, p of timer channel start register m (TSm)

TEmn, TEmp: Bit n, p of timer channel enable status register m (TEm)

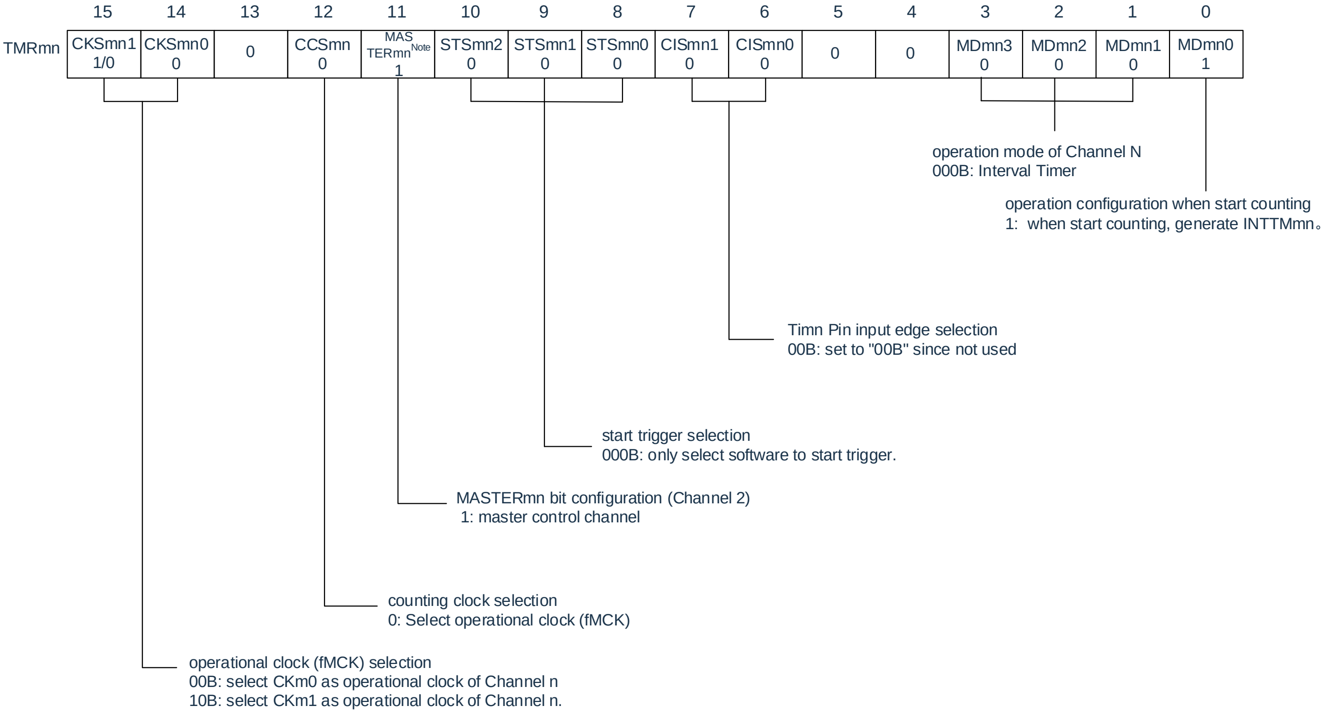
TCRmn, TCRmp: Timer count register mn, mp (TCRmn, TCRmp)

TDRmn, TDRmp: Timer data register mn, mp (TDRmn, TDRmp)

TOmn, TOmp: TOmn and TOmp pin output signals

Figure 5-69 Example of register setting contents 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	MASTERmn Note 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 n
TOm	TOmn 0

0: Output “0” by TOmn.

(c) Timer output enable register m (TOEm).

	bit n
TOEm	TOEmn 0

0: Stop TOmn output by the count run.

(d) Timer output level register m (TOLm).

	bit n
TOLm	TOLmn 0

0: “0” is set at TOMmn=0 (master channel output mode).

(e) Timer outputs mode register m (TOMm).

	bit n
TOMm	TOMmn 0

0: Set the master channel output mode.

Note TMRm2 : MASTERmn=1
TMRm0 : Fixed to “0”.

Note:m: Unit number (m= 0,1) n: master channel number (n=0, 2).

Figure 5-70 Example of register setting contents 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[Note] 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” by TOmp.
1: output “1” by TOmp.

(c) Timer output enable register m (TOEm).

	bit p
TOEm	TOEmp 1/0

0: Stop TOmp output from counting runs.
1: Enable TOmp output by counting runs.

(d) Timer output level register m (TOLm).

	bit p
TOLm	TOLmp 1/0

0: Positive logic output (active high).
1: Negative logic output (active-low).

(e) Timer outputs mode register m (TOMm).

	bit p
TOMm	TOMmp 1

1: Set slave channel output mode.

Note TMRm2: MASTERmp bit
TMRm1, TMRm3: SPLITmp bit

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

Figure 5-71 Operation procedure for the PWM function (1/2)

	Softeware operation	Hardware status
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	
	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)

Figure 5-72 Operation procedure for the PWM 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 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)

Note m: unit number (m= 0,1) n: master channel number (n=0).
p: slave channel number q: slave channel number
n < p < q≤3 (p and q are integers greater than n)

5.9.3 Operates as multiplex PWM output function

This is the ability to perform multiple PWM outputs by extending the PWM functionality and using multiple slave channels for different duty cycles. For example, when 2 slave channels are used in pairs, the period and duty cycle of the output pulse can be calculated using the following formula:

Pulse period = { set value of TDRmn (master) +1} × counting clock period
Duty cycle 1[%] = set value of {TDRmp (slave 1)} / {set value of TDRmn (master control) +1} ×100
Duty cycle 2[%] = set value of {TDRmq (slave 2)} / {set value of TDRmn (master control) +1} ×100

Note When the set 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, the timer count register mn (TCRmn) of the master channel runs and counts cycles. In single-count mode, the TCRmp register of slave channel 1 runs and counts the duty cycle and outputs the PWM waveform from the TOmp pin. Starting with the INTTMmn of the master channel, the value of the timer data register mp (TDRmp) is loaded into the TCRmp register and decremented in the count. If TCRmp becomes “0000H”, 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 master channel and 1 count clock has passed, the output level of TOmp becomes effective if TCRmp becomes “0000H”, it becomes an invalid level.

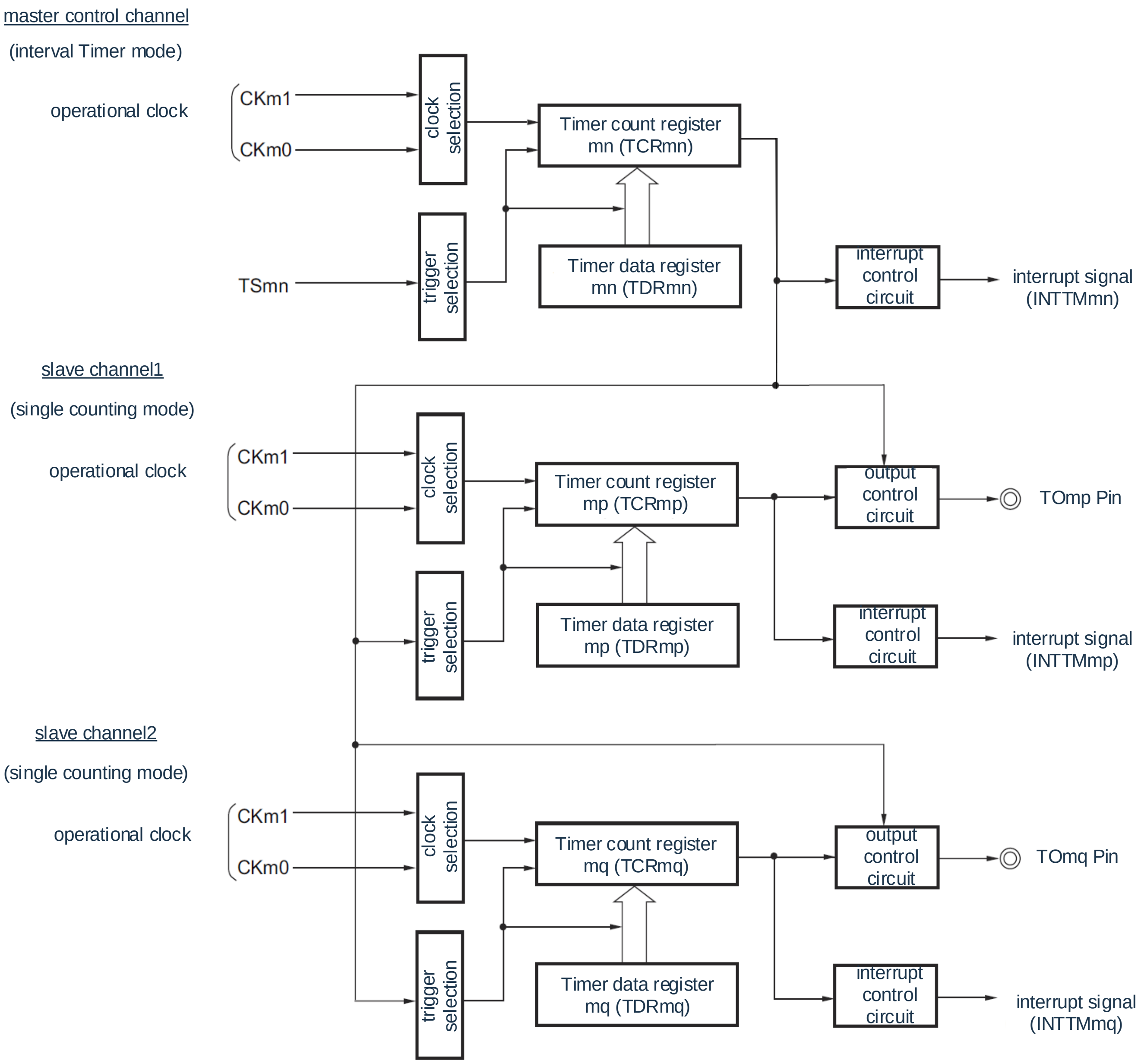
As with the TCRmp register of slave channel 1, in single-count mode, the TCRmq register of slave channel 2 runs and counts the duty cycle and outputs the PWM from the TOmq pin waveform. Starting with the INTTMmn of the master channel, the value of the TDRmq register is loaded into the TCRmq register and decremented in the count. If TCRmq becomes “0000H”, 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 master channel and 1 count clock has passed, the output level of TOmq becomes effective if the TCRmq becomes “0000H”, it becomes an invalid level.

When channel 0 is used as the master channel in this operation, up to 3 PWM signals can be output simultaneously.

Note To overwrite the timer data register mn (TDRmn) of the master channel and the TDRmp register of slave channel 1 at the same time, at least 2 write accesses are required. Because the values of the TDRmn register and TDRmp register are loaded into the TCRmn register and the TCRmp register when the INTTMmn is generated on the master channel, Therefore, if the INTTMmn is rewritten before and after the intake is generated on the master channel, the TOmp pin cannot output the expected waveform. Therefore, to override both the TDRmn registers of the master and the TDRmp registers of the slave, the 2 registers must be overwritten immediately after the INTTMmn is generated on the master channel (the same applies to the slave channels). TDRmq register of 2).

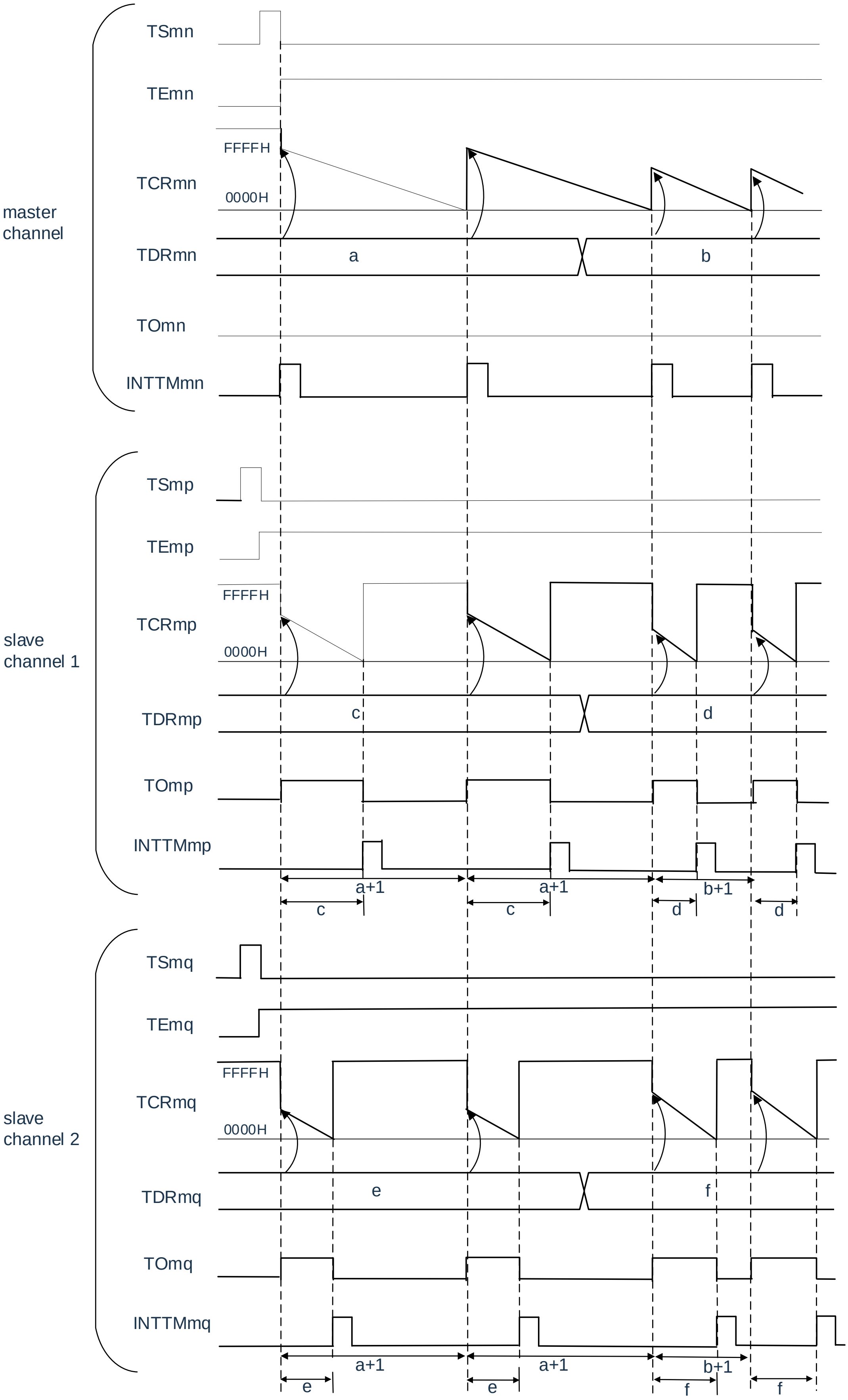
Note m: Unit number (m= 0,1) n: Master channel number (n=0).
p: slave channel number q: slave channel number
n < p < q≤ 3 (p and q are integer greater than n)

Figure 5-73 Block diagram of operation as multiple PWM output function (when outputting 2 kinds of PWM)



Remark m: Unit number (m= 0,1) n: master channel number (n=0).
p: slave channel number q: slave channel number
n < p < q≤ 3 (p and q are integers greater than n)

Figure 5-74 Block diagram of operation as multiple PWM output function (when outputting 2 kinds of PWM)

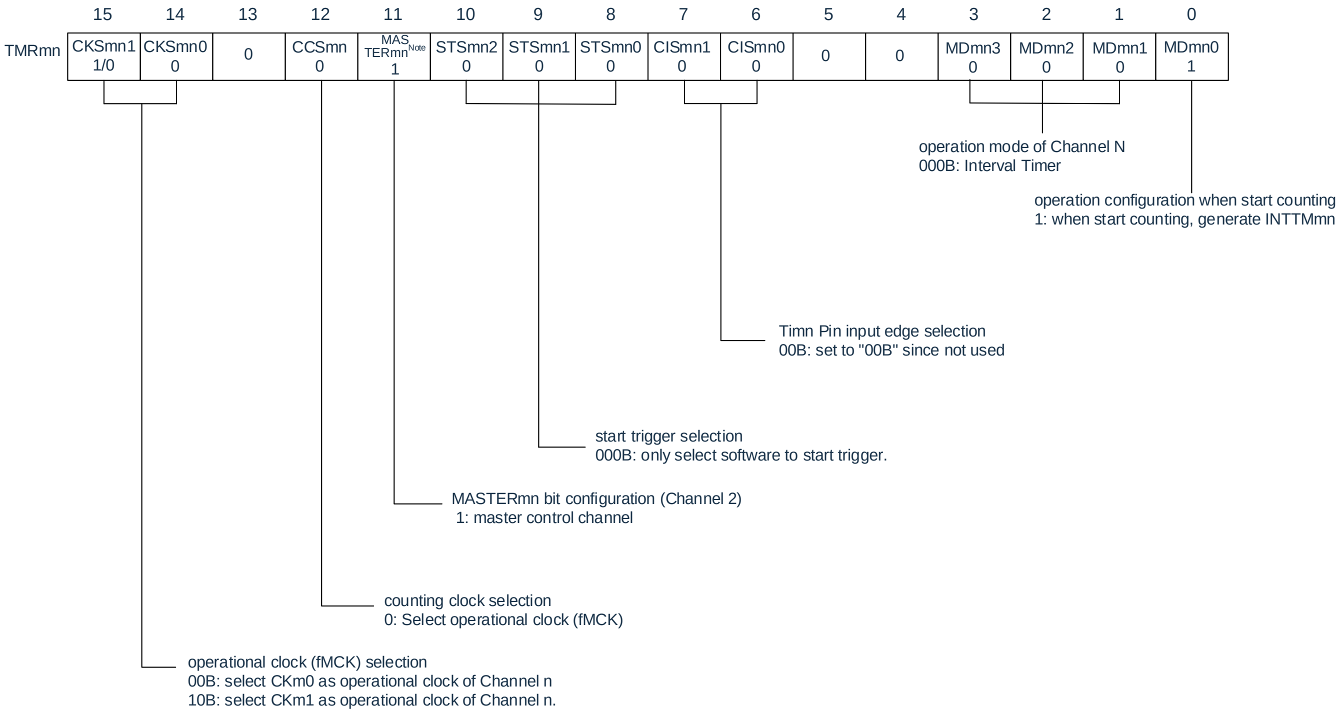


Note 1. m: Unit number (m= 0,1) n: master channel number (n=0).
p: slave channel number q: slave channel number
n ＜ p ＜ q ≤ 3 (p and q are integers greater than n)

2. TSmn, TSmp, TSmq: Bit n, p, 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 register mn, mp, mq (TCRmn, TCRmp, TCRmq)
TDRmn, TDRmp, TDRmq: Timer data register mn, mp, mq (TDRmn, TDRmp, TDRmq)
TOmn, TOmp, TOmq: TOmn, TOmp, TOmq pin output signals

Figure 5-75 Example of register setting contents 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	MASTERmn^{Note} 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 n	
TOm	TOmn 0	0: Output “0” by TOmn.

(c) The timer output enable register m (TOEm).

	bit n	
TOEm	TOEmn 0	0: Stop TOmn output by the count run.

(d) Timer output level register m (TOLm).

	bit n	
TOLm	TOLmn 0	0: “0” is set at TOMmn=0 (master channel output mode).

(e) Timer outputs mode register m (TOMm).

	bit n	
TOMm	TOMmn 0	0: Set the main channel output mode.

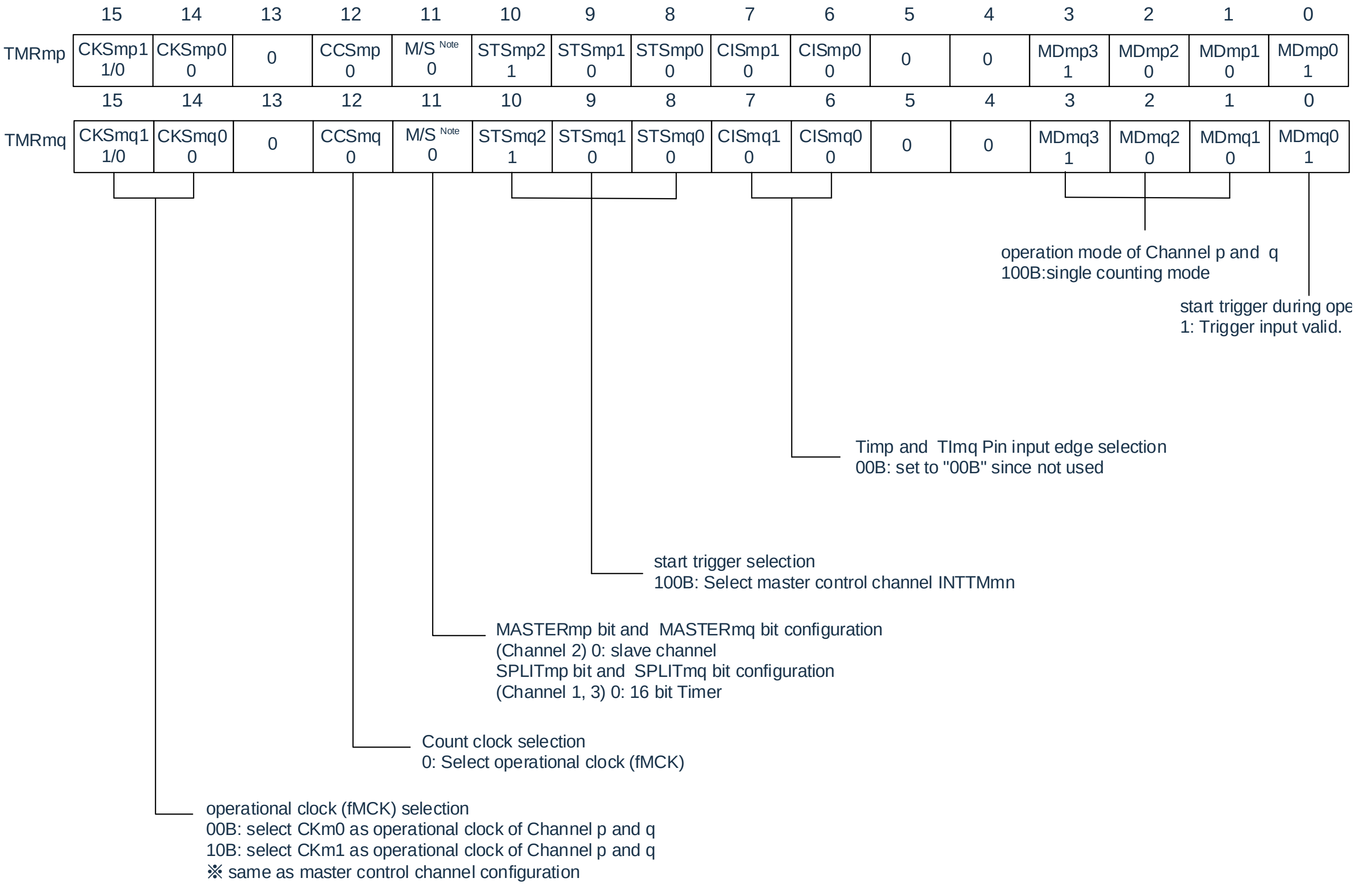
conce ntrate TMRm2 : MASTERmn=1

TMRm0 : Fixed to “0”.

Remarks m: Unit number (m= 0,1) n: master channel number (n=0).

Figure 5-76 Example of register setting contents for multiple PWM output function (slave channel) (when outputting 2 types of PWM)

(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 Note 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 Note 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) The timer output enable register m (TOEm).

	bit q	bit p
TOEm	TOEmq 1/0	TOEmp 1/0

0: Stop TOmp and TOmq outputs by counting runs.
1: Enable TOmp and TOmq outputs by counting runs.

(d) Timer output level register m (TOLm).

	bit q	bit p
TOLm	TOELq 1/0	TOELp 1/0

0: Positive logic output (active high).
1: Negative logic output (active-low).

(e) Timer outputs mode register m (TOMm).

	bit q	bit p
TOMm	TOMLq 1	TOMLp 1

1: Set slave channel output mode.

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

Remark m: unit number (m= 0,1) n: master channel number (n=0).
p: slave channel number q: slave channel number
n < p < q≤ 3 (p and q are integers greater than n)

Figure 5-77 Operation procedure for multiple PWM output function (in case of 2 PWM outputs) (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 5-78 Operation procedure for multiple PWM output function (in case of 2 PWM outputs) (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 pin 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).
p: slave channel number q: slave channel number
n < p < q≤3 (p and q are integers greater than n)

Chapter 6 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.

6.1 Structure of the output control circuit

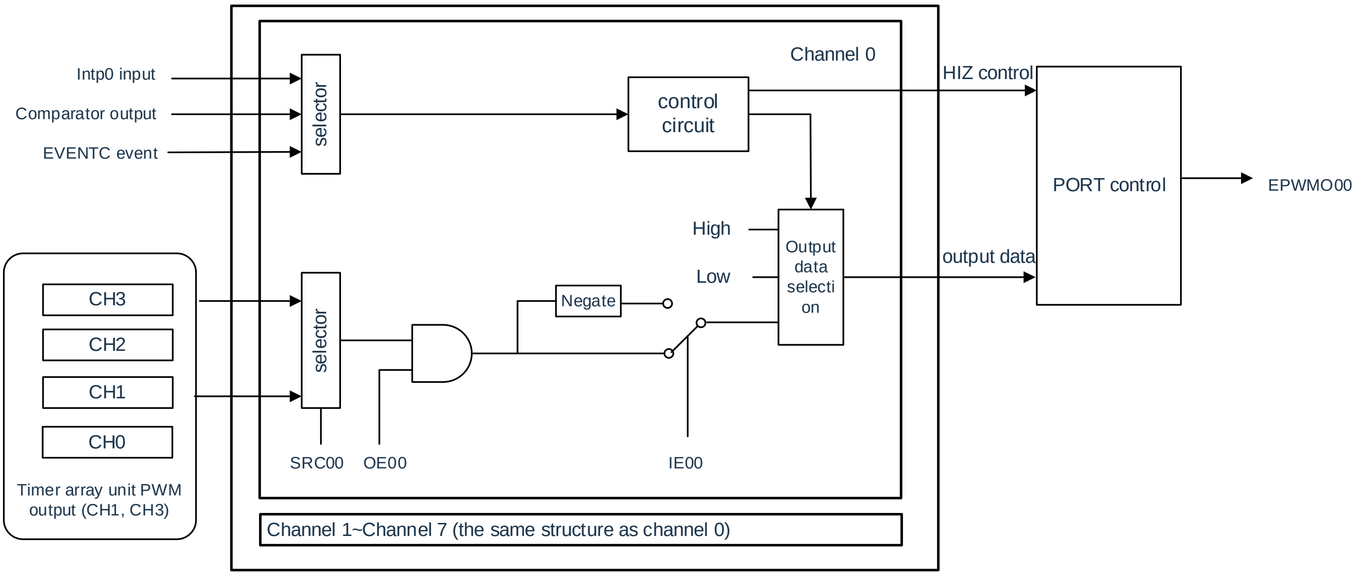
The EPWM output control circuit consists of the following hardware.

Table 6-1 Structure of the output control circuit of EPWM

Item	Structure
Control registers	EPWM input source selection register (EPWMSRC).
	EPWM output control register (EPWMCTL).
	EPWM force truncated input selection register (EPWMSTC).
	EPWM force truncated output selection register (EPWMSTL).
	EPWM Status Register (EPWMSTR).
output	EPWM output (EPWMO00~EPWMOP07)

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

Figure 6-1 Block diagram of EPWM output control circuit



6.2 Registers for controlling EPWM output control circuit

The real-time output control circuit is controlled by the following registers.

- Peripheral enable register 0 (PER1).
- EPWM input source selection register (EPWMSRC).
- EPWM output control register (EPWMCTL).
- EPWM force truncated input select register (EPWMSTC).
- EPWM force truncated output select register (EPWMSTL).
- EPWM status register (EPWMSTR).
- Port mode register (PMxx).
- Port mode control register (PMCxx).
- Port register (Pxx).

6.2.1 Peripheral enable register 1 (PER1).

The PER1 register is a register that sets the clock that 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 Permissible Registers 0, 1 (PER0, PER1)” for details

6.2.2 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 6-2 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

Note n: Channel number (n=0~7).

6.2.3 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 6-3 Format of EPWM Output Control Register (EPWMCTL).

Address: 0x40044408 After reset: 0000H R/W

symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EPWMCTL	IE07	IE06	IE05	IE04	IE03	IE02	IE01	IE00	OE07	OE06	OE05	OE04	OE03	OE02	OE01	OE00

OE0n	Control of EPWMO0n output
0	Disable output
1	Enable output

Note n: Channel number (n=0~7).

IE0n	Reverse control of EPWMO0n output
0	Not reversed
1	Reversed

Note n: Channel number (n=0~7).

6.2.4 EPWM force truncated input selection 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 6-4 Format of EPWM force truncated input selection 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 theN_EG at least three clocks apart after the I SC_SEL1 setting, and then set the C_SEL0 and S.

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.

6.2.5 EPWM force truncated output selection 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 6-5 Format of EPWM force truncated output selection 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

IONo. 1	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

Note n: Channel number (n=0~7).

6.2.6 EPWM status register (EPWMSTR)

The EPWMSTR register clears the forced truncation signal and displays the truncation status. If the clear trigger bit HZCLR is set to “1”, the truancy state is released. When the truncation status indicates that the signal of the SHTFLG is high, it enters the forced truncation state. bit0 is write-only bit, and the read value is always “0”. bit7~1 is read-only.

The EPWMSTR registers are set via 8-bit memory operation instructions.

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

Figure 6-6 Format of EPWM status register (EPWMSTR)

Address: 0x4004410 After reset: 0000H R/W

symbol	7	6	5	4	3	2	1	0
EPWMSTR	0	0	0	0	0	0	SHTFLG	HZCLR

SHTFLG	Force truncation status flag
0	Normal output state
1	Force truncation state

HZCLR	Software clearance to force truncation signals
0	-
1	The software dismisses the truncation state

Note: 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.

6.2.7 Control registers for the port function of the EPWM output pins

When using the EPWM output, the control register (Port Mode Register (PMxx, PMCxx)) for the port function multiplexed with the EPWM output pin (EPWMOn pin) must be set. For details, refer to “2.3.1 Port Mode Register (PMxx)”.

When using the multiplexed ports of the EPWM pins as outputs of EPWMO, the bits of the port mode registers (PMxx, PMCxx) corresponding to each port must be set to “0”. In this case, the bit of the port register (Pxx) can be “0” or “1“.

For details, please refer to “2.5 Register Settings When Using the Multiplexing Function“.

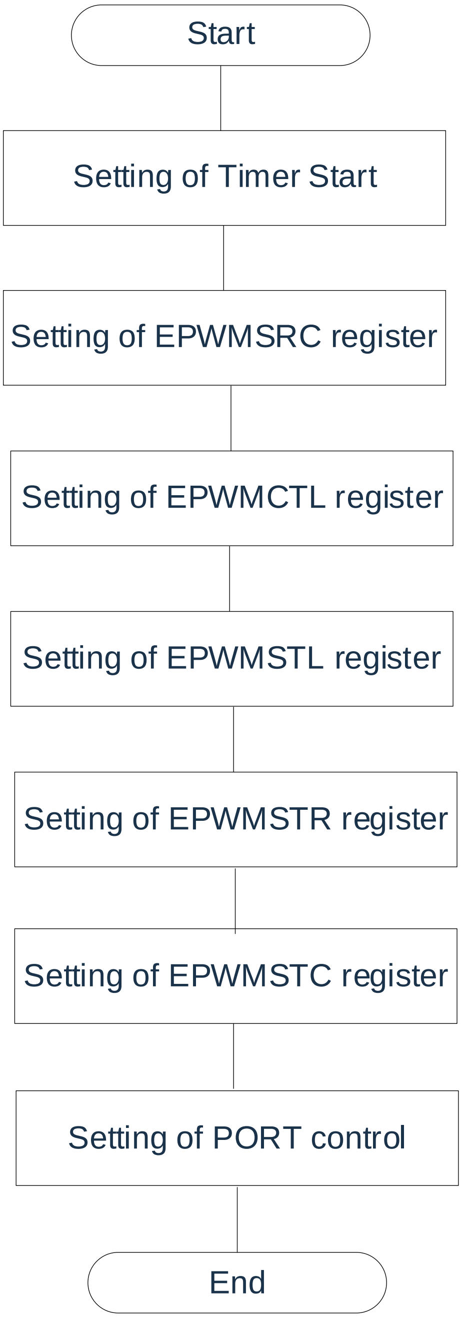
6.3 Operation of EPWM output control circuit

6.3.1 Initial setup

The timer waveform selects the TAU output (TO01, TO03) as the source clock through the EPWSRC register. The positive or inverting phase of the timer waveform can be fixed by setting the EPWMCTL register.

In the event of forced truncation, the Hi-Z output, low output, high output, or disable cut-off output can be selected through the setting of the EPWMSTL register.

Figure 6-7 Initial configuration flow of registers

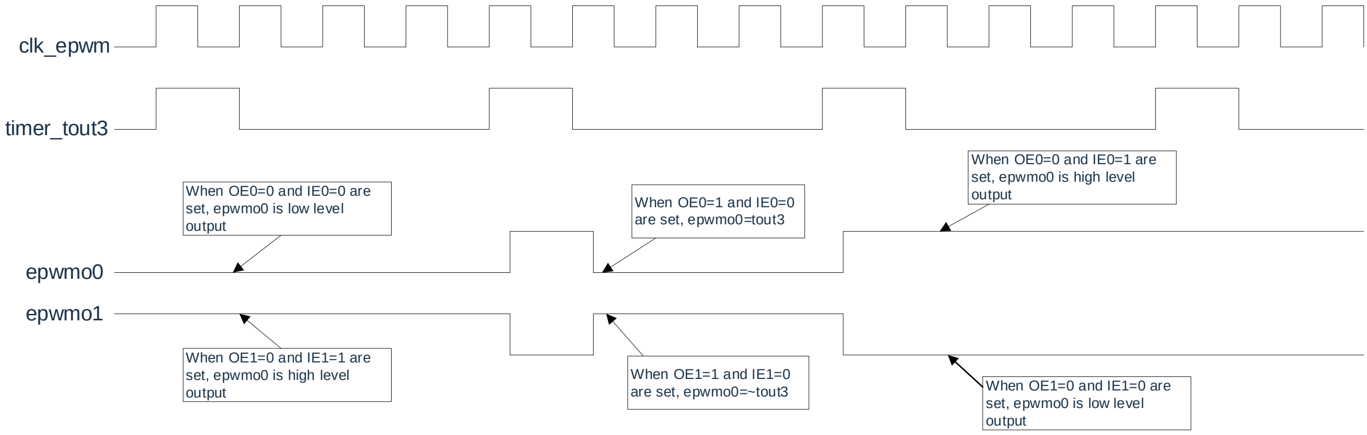

Start
Setting of Timer Start
Setting of EPWMSRC register
Setting of EPWMCTL register
Setting of EPWMSTL register
Setting of EPWMSTR register
Setting of EPWMSTC register
Setting of PORT control
End

6.3.2 Normal operation

Depending on the register settings, four output data can be selected, namely forward waveform output, inverted waveform output, low level output, and high-level output. The EPWMCTL registers can be changed at runtime. Both OE0n bits and IE0n bits must be written at the same time.

For details, please refer to “Table 6-2 Operation Instructions for truncation signals”.

Figure 6-8 Output timing diagram


clk_epwm
timer_tout3
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
epwmo0
epwmo1
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

6.3.3 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 6-9.

(2) Release of forced truncation

a) Software release: When bit3 (HS_SEL) of EPWMSTC register is 0, the software release mode is used. 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.

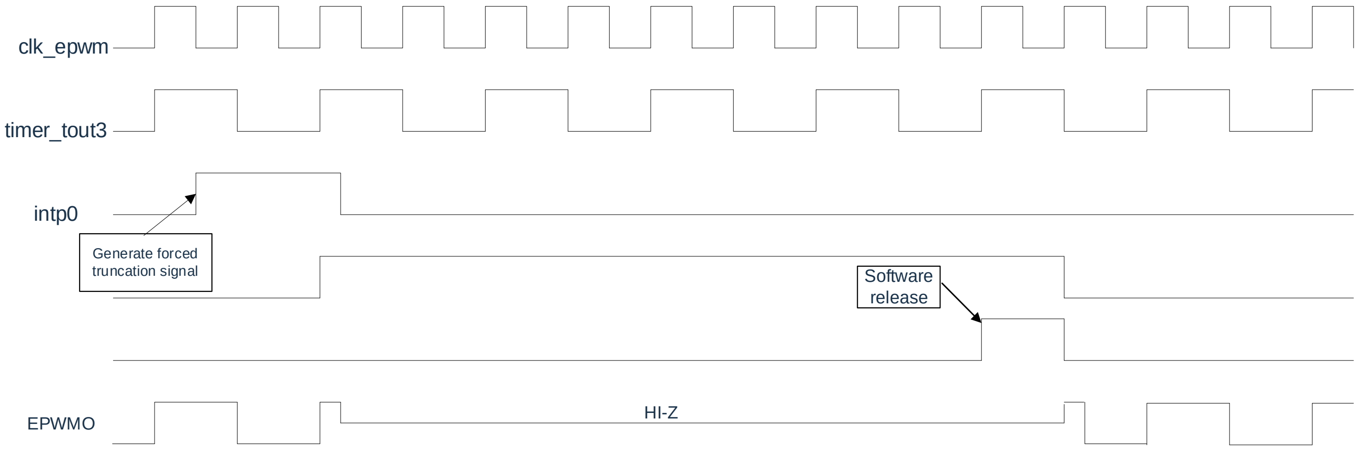
b) Hardware release: When bit3 (HS_SEL) of EPWMSTC register is 1, the hardware release mode is used. The forced truncation state is released by the edge of CMP0 output or INTP0 input.

Table 6-2 Table of operation Instructions for truncation signals

bit	IOn1-0	OE0n	IE0n	SHTFLG	EPWM output pin
The setting 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

Notes n=0~7

Figure 6-9 Timing diagram for generation and release of INTP0 truncation (HS_SEL=0, REL_SEL=0)



Note: Short pulses may be generated when switching from “normal operation” to “Hi-Z”, “fixed low” or “fixed high” during forced cutoff caused by the cutoff signal INTP0, or when returning to the forced cutoff state by immediate release.

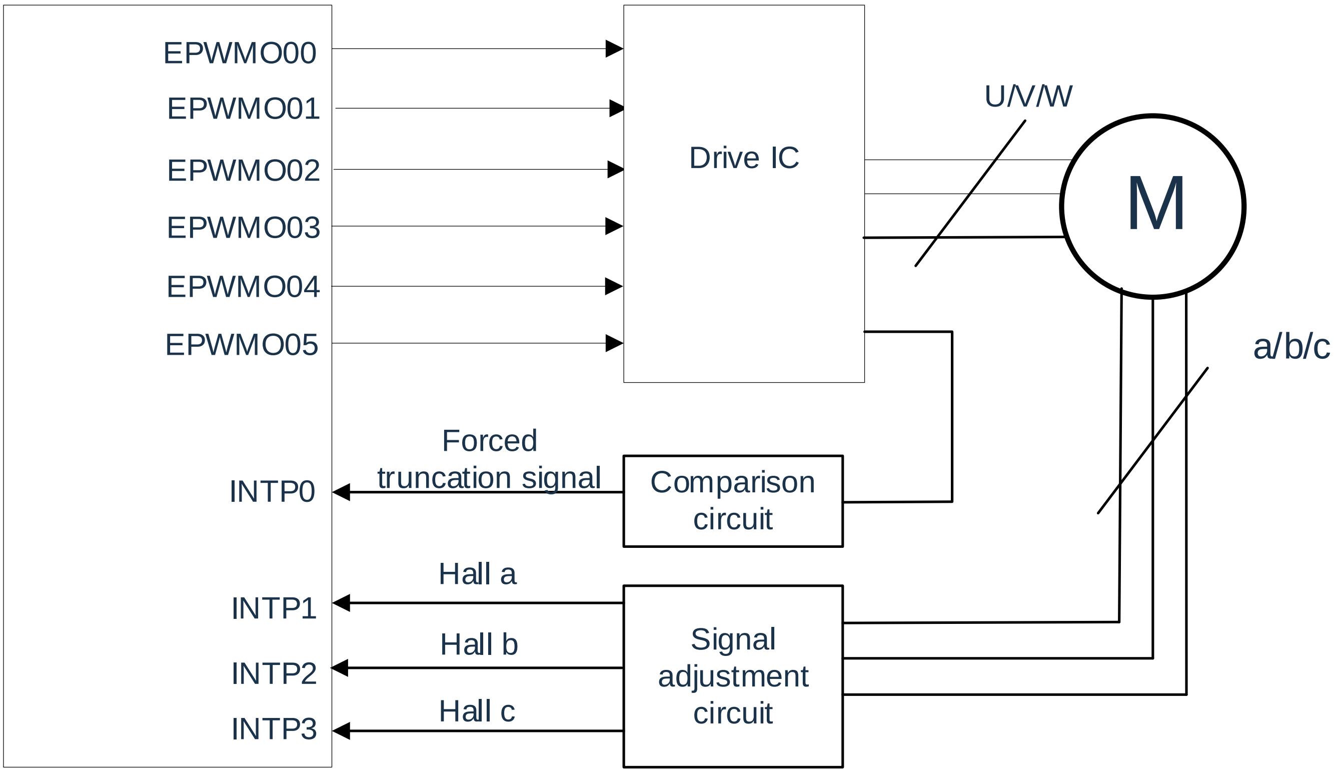
6.4 Control example of brushless DC motor

The following is an example of using the EPWM control function to control a brushless DC motor (hereinafter referred to as a BLDC motor).

6.4.1 Example of a hardware connection

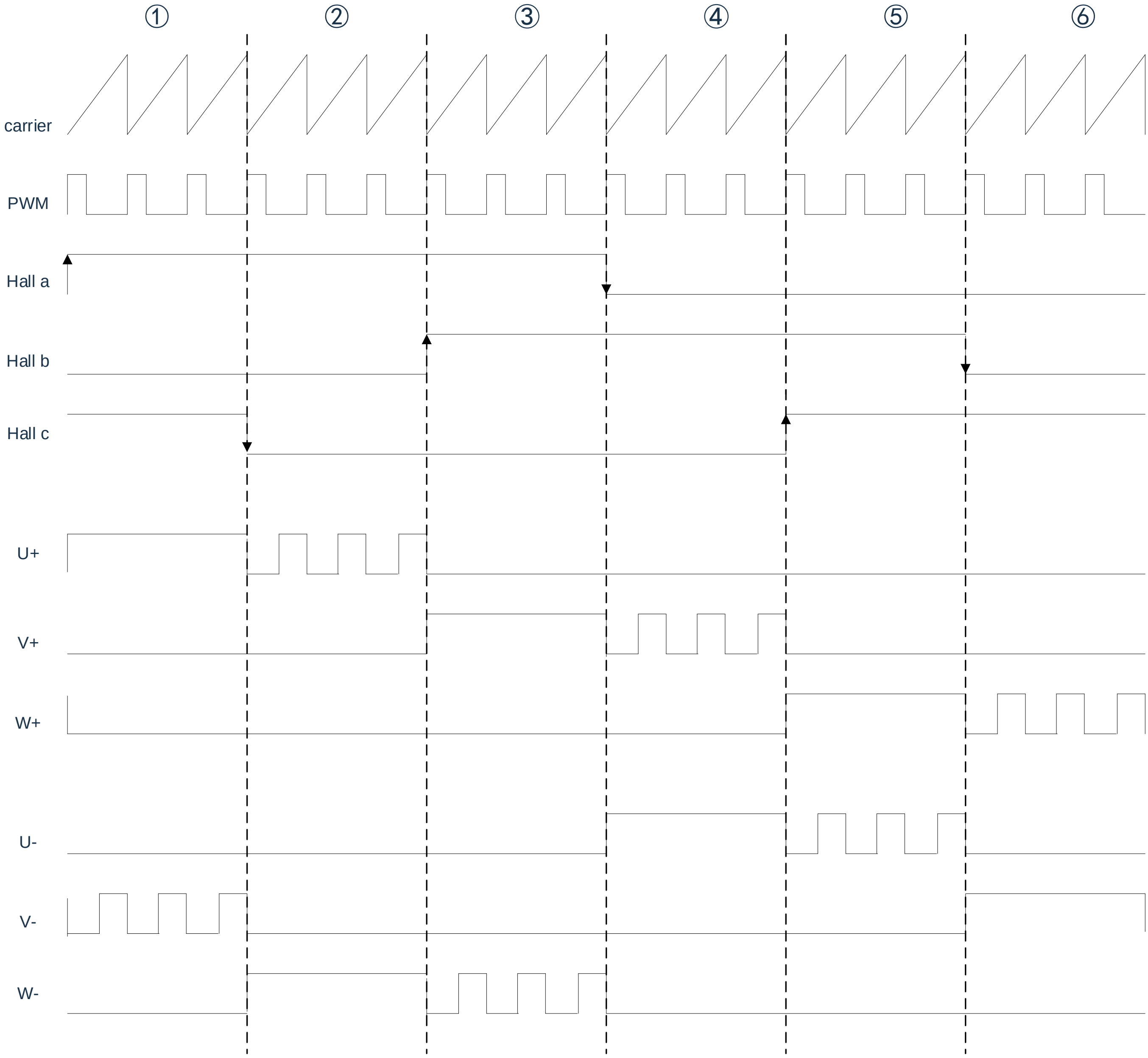
An example of a hardware connection for a brushless DC motor is shown in Figure 6-10. In this example, EPWMO00~EPWMO05 (output) is used for output control of BLDC motors, INTP1~INTP3(input) for the output signal of the Hall sensor, and INTP0 (input) is used to force a truncated signal.

Figure 6-10 Example of hardware connections



6.4.2 Control timing of three-phase brushless DC motors

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



6.4.3 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 6-4 Example of setting the EPWMCTL0 register

Description	Setting value of the EPWMCTL
State (1): The rising edge of Hall a disables U+, U+ reverse outputs, allowing V–, V–forward outputs.	0x0110
State (2): The Hall c falling edge allows U+, U+ forward output, and disables W–, W– reverse outputs.	0x2001
State (3): The Hall b rising edge disables V+, V+ reverse outputs, allowing W–, W– forward outputs.	0x0220
State (4): The falling edge of Hall A allows V+, V+ forward output, disables U–, U–reverse output.	0x0802
State (5): The Hall c rising edge disables W+, W+ reverse outputs, allowing U–, U–forward outputs.	0x0408
State (6): The Hallb falling edge allows W+, W+ forward outputs, and V–, V–reverse outputs are disabled.	0x1004

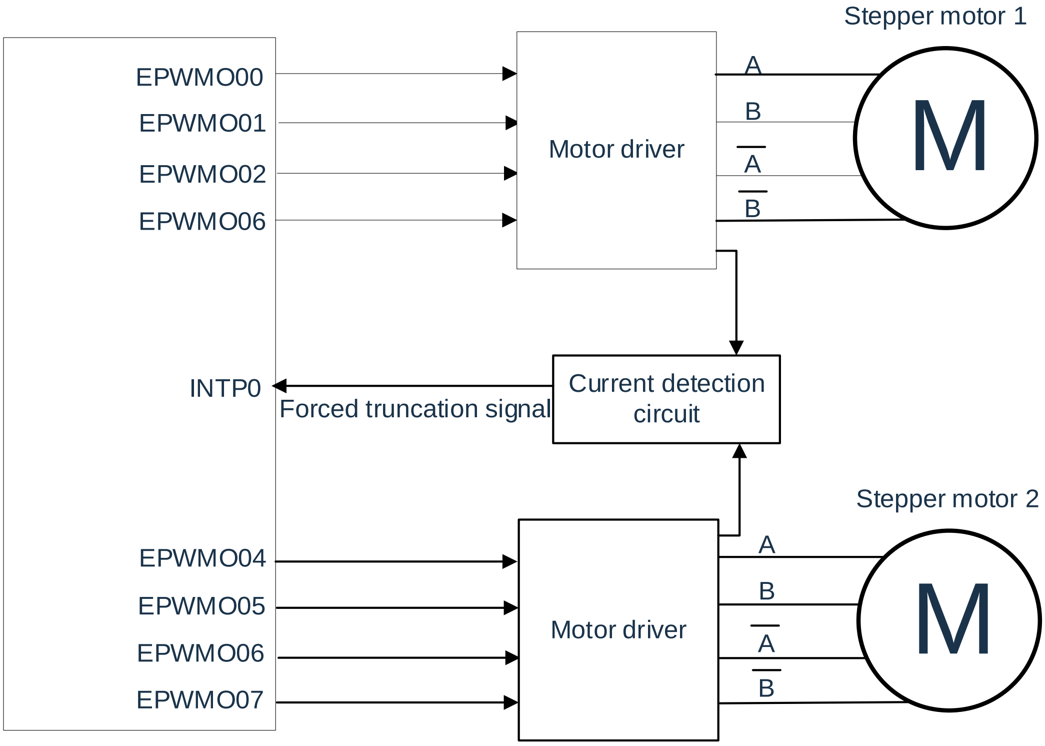
6.5 Example of stepper motor control

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

6.5.1 Example of a hardware connection

An example of a hardware connection to control two stepper motors is shown in Fig. 6-12.

Figure 6-12 Example of a hardware connection



6.5.2 Control method

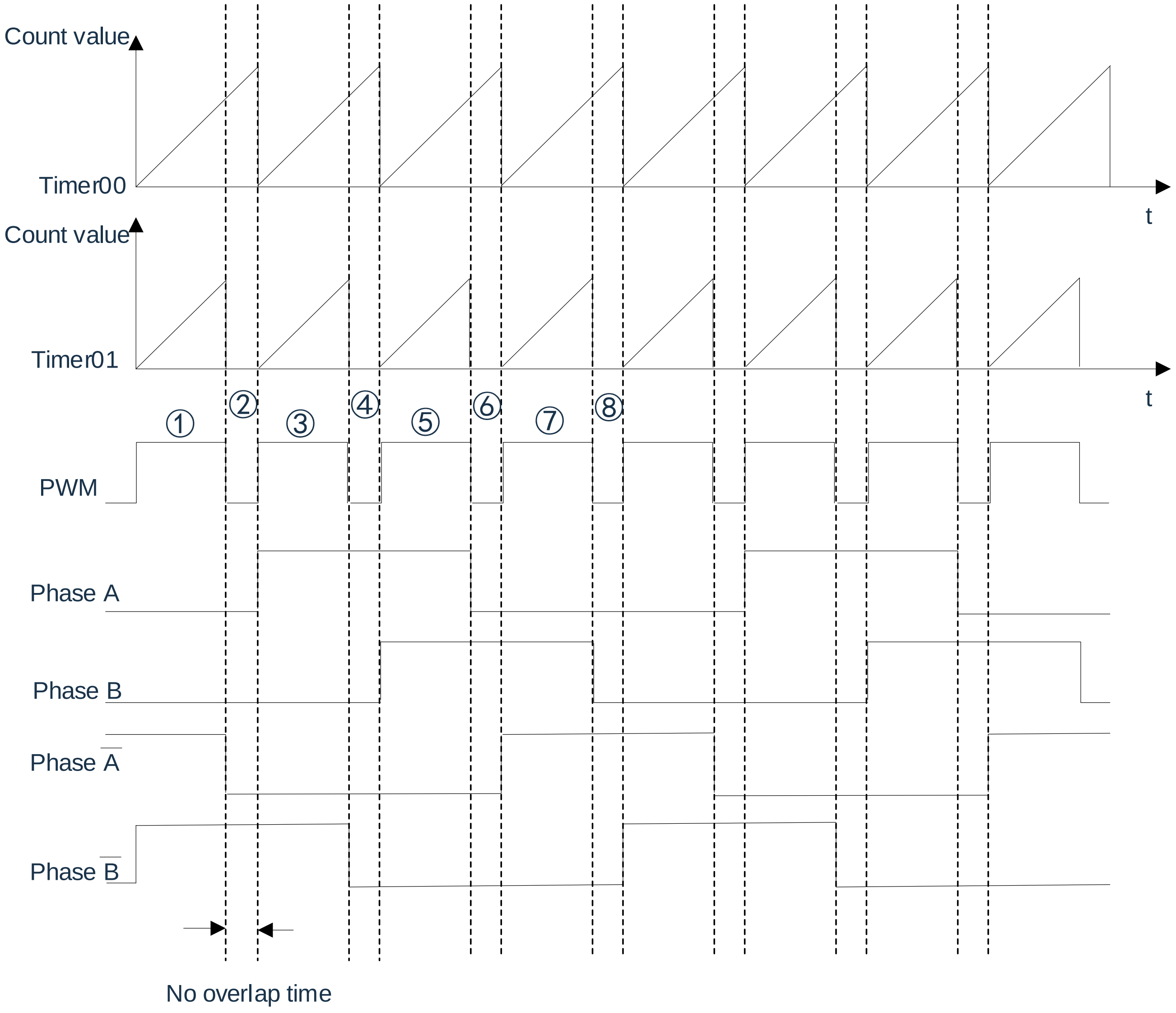
The stepper motor is rotated, reversed or stopped in two-phase excitation mode by using eight EPWMOs. Control the rotation speed via Timer's PWM mode.

In this example, Timer's CH0 and CH1 are used for the control of stepper motor 1, CH2 and CH3 are used for the control of stepper motor 2. If you combine 2 Timer channels, you can generate pulses of any period and duty cycle. CH0 and CH2 are the main control channels and operate as interval timer mode. CH1 and CH3 are slave channels and operate as single-count mode.

In addition, the cross-current prevention time (no overlapping time) is inserted when switching the output type.

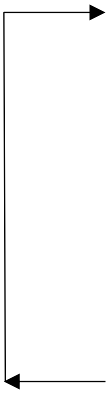
An example of a waveform for stepper motor control is shown in Figure 6-13.

Figure 6-13 Waveform example of step motor control



6.5.3 Example of register setting

Table 6-5 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 7 Real-Time Clock

7.1 Function of real-time clock

The real-time clock has the following functions.

- Hold counters for years, months, weeks, days, hours, minutes, and seconds, up to 99 years.
- Fixed cycle interrupt function (period: 0.5 seconds, 1 second, 1 minute, 1 hour, 1 day, 1 month).
- Alarm interrupt function (alarm: week, hour, minute).
- 1Hz pin output function

7.2 Structure of real-time clock

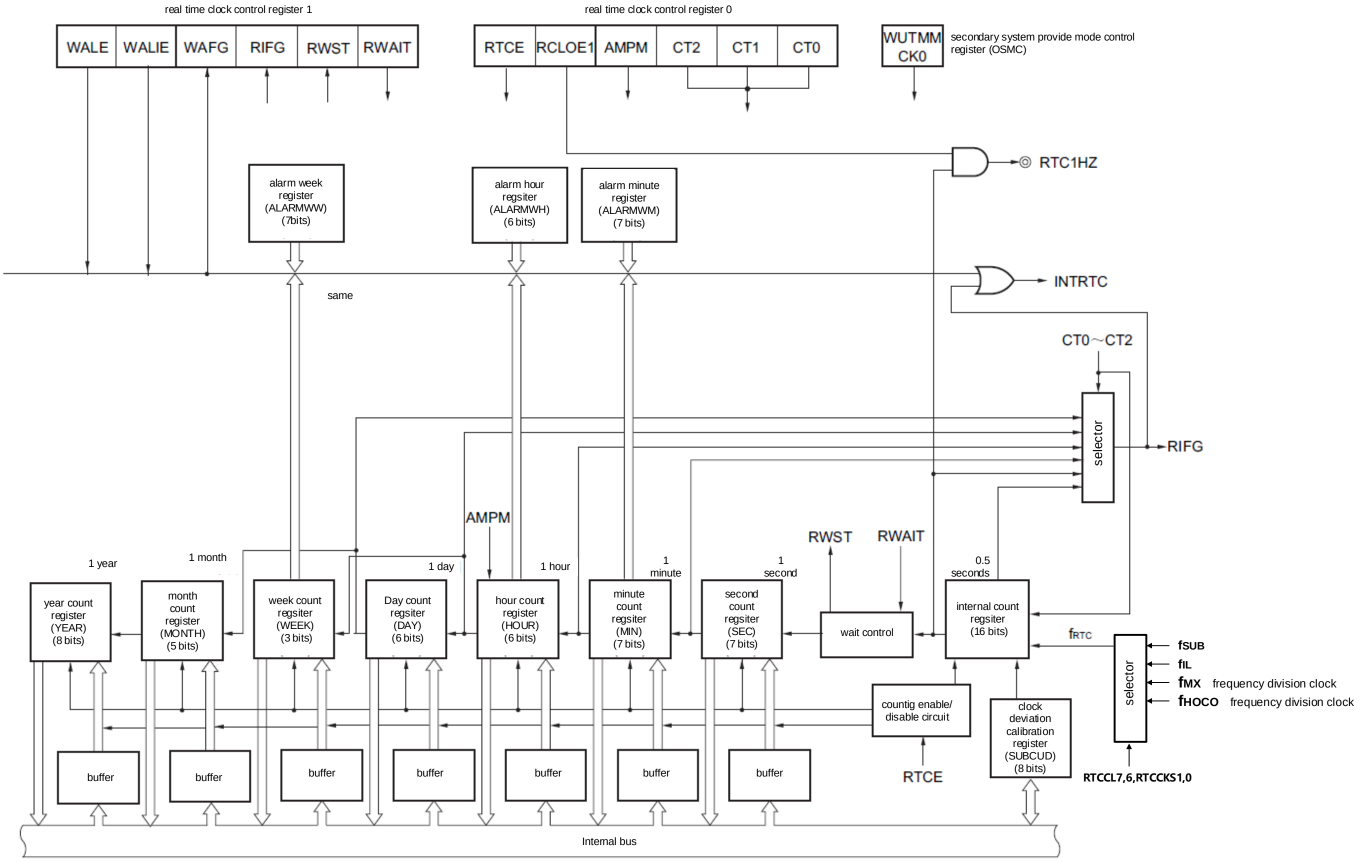
The real-time clock consists of the following hardware.

Table 7-1 Structure of real-time clock

project	structure
counter	Internal counter (16-bit).
Control registers	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).
	The hour count register (HOUR).
	Day Count Register (DAY).
	Week count register (WEEK).
	Month count register (MONTH).
	Year Count Register (YEAR).
	Clock error correction register (SUBCUD).
	Alarm clock minute register (ALARMWM).
	Alarm clock hour register (ALARMWH).
	Alarm clock week register (ALARMWW).

Note: The reset of the above RTC control registers is only controlled by POR reset.

Figure 7-1 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
INTRTC
alarm week register (ALARMWW) (7bits)
alarm hour register (ALARMWH) (6 bits)
alarm minute register (ALARMWM) (7 bits)
same
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 register (WEEK) (3 bits)
Day count register (DAY) (6 bits)
hour count register (HOUR) (6 bits)
minute count register (MIN) (7 bits)
second count register (SEC) (7 bits)
wait control
internal count register (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 Only the fmx/fhoco is selected for the weekly clock (≈ 32,768 KHZ) or the sub-system clock (f_{SUB}=32.768kHz) after the selection of fmx/f hoco. As a real-time clock running clock, the counting of years, months, weeks, days, hours, minutes, and seconds can be performed. When selecting a low-speed internal oscillator clock (f_{IL}=15kHz), only a fixed-cycle interrupt function can be used.
The fixed-period interrupt interval when selecting f_{IL} is calculated using the following formula:

Fixed period (value selected by the RTCC0 register) f_{SUB}/f_{IL}

7.3 Registers for controlling real-time clock

The real-time clock is controlled by the following registers.

- Peripheral enable register 0 (PER0).
- Real-time clock selection register (RTCCL).
- Real-time clock control register0 (RTCC0).
- Real-time clock control register1 (RTCC1).
- Second count register (SEC).
- Minute count register (MIN).
- Hour count register (HOUR).
- Day count register (DAY).
- Week count register (WEEK).
- Month count register (MONTH).
- Year count register (YEAR).
- Clock error correction register (SUBCUD).
- Alarm clock minute register (ALARMWM).
- Alarm clock hour register (ALARMWH).
- Alarm clock week register (ALARMWW).
- Port mode register (PMxx).
- Port mode control register (PMCxx).
- Port multiplexing function configuration register (PxxCFG).

7.3.1 Peripheral enable register 0 (PER0).

The PER0 register is the register that sets whether to enable or disable the supply of clocks to each peripheral hardware. Reduce power consumption and noise by stopping clocking unused hardware.

To use a real-time clock, bit7 (RTCEN) must be set to “1”. The PER0 register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register becomes “00H”.

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

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

symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	GODAEN	ADCEN	IICA0EN	SAU1EN	SAU0EN	TM41IN	TM40EN

RTCEN	Provides control of the real-time clock (RTC) and the input clock of a 15-bit interval timer
0	Stop supplying the input clock. • Cannot write the SFR used by the real-time clock (RTC) and 15-bit interval timer. • The real-time clock (RTC) and 15-bit interval timer are in the reset state.
1	An input clock is provided. • Reads and writes to the SFR used by real-time clocks (RTCs) and 15-bit interval timers.

Note 1 If you want to use a real-time clock, you must first set the RTCEN bit to “1” in the stable oscillation of the count clock (fRTC) and then set the following registers. When the RTCEN bit is “0”, the write operation of the real-time clock control register is ignored, and the read value is the initial value (except for the real-time clock selection register (RTCCL), the port mode register, and the 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 of 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 on the subsystem clock.

7.3.2 Real-time clock selection register (RTCCL)

The real-time clock and the count clock (fRTC) of the 15-bit interval timer can be selected via RTCCL.

Figure 7-3 Format of real-time clock selection register (RTCCL)

Address: 0x4004047C After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
RTCCL	RTCCL7	RTCCL6	RTCCL5	0	0	0	RTCCKS1	RTCCKS0

RTCCL7	Selection of clock source for real time clock, counting clock of 15-bit interval timer
0	Select the high-speed system clock (fMX).
1	Select the 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 be set to 1).
1	0	0	1	Master clock fmax/fhoco (selected via RTCCL7)/1952
1	0	0	0	Master clock fmax/fhoco (selected via RTCCL7)/1464
1	0	1	0	Master clock fmax/fhoco (selected via RTCCL7)/976
1	1	0	0	Master clock fmax/fhoco (selected via RTCCL7)/488
1	1	1	0	Master clock fmax/fhoco (selected via RTCCL7)/244

7.3.3 Real-time clock control register0 (RTCC0)

This is an 8-bit register that sets the start or stop of operation of the real-time clock, control of the RTC1HZ pin, 12/24-hour system, and fixed-cycle interrupt function.

The RTCC0 register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register becomes “00H”.

Figure 7-4 Format of real-time clock control register 0 (RTCC0)

Address: 0x40044F5D After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
RTCC0	RTCE	0	RCLOE1[Note]	0	AMPM	CT2	CT1	CT0

RTCE	Operational control of real-time clock
0	Stop the operation of the counter.
1	Start the operation of the counter.

RCLOE1	Output control of RTC1HZ pin
0	Disable the output of the RTC1HZ pin (1Hz).
1	Enable the output of the RTC1HZ pin (1Hz).

AMPM	Selection of 12-hour system/24-hour system
0	12-hour system (for morning or afternoon).
1	24-hour system
• To change the value of the AMPM bit, the RWAIT bit (bit0 of the real-time clock control register 1 (RTCC1)) must be set to “1” and rewrite later. If you change the value of the AMPM bit, the value of the hour count register (HOUR) becomes the corresponding value of the time system in which you set it. • The representation of the time bit is shown in Table 11-2.	

CT2	CT1	CT0	Selection of fixed period interruption (INTRTC)
0	0	0	Do not use the fixed-cycle interrupt feature.
0	0	1	0.5 seconds once (synchronized with second accumulation).
0	1	0	Once every 1 second (at the same time as the second accumulation).
0	1	1	Once every 1 minute (00 seconds per minute).
1	0	0	Once every 1 hour (00 minutes and 00 seconds per hour).
1	0	1	Once a day (00:00:00 every day).
1	1	×	Once every 1 month (1 of each month at 00:00:00 AM).
To change the value of the CT2~CT0 bits while the counter is running (RTCE=1), it must be overridden after setting the INTRTC to disable interrupt handling through the interrupt mask flag register, and the RIFG must be cleared after the override flags and RTCIF flags, and then set to allow interrupt handling.			

Note 1 When the RCE bit is “1”, the RCLOE1 bit cannot be changed.

2. When the RTCE bit is “0”, even if the RCLOE1 set to “1” is not output 1Hz.

Remark ×: Ignore

7.3.4 Real-time clock control register1 (RTCC1)

This is the 8-bit register that controls the alarm interrupt function and the counter waits. The RTCC1 register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register becomes “00H”.

Figure 7-5 Format of real-time clock control register 1 (RTCC1) (1/2)

Address: 0x40044F5E After reset: 00H R/W

symbol	7	6	5	4	3	2	1	0
RTCC1	INBut	WALIE	0	WAFG	RIFG	0	RWST	RWHas

WHALES	Alarm clock operation control
0	Consistent operation is invalid.
1	Consistent operation is valid.
To set the WALE bit while the counter is running (RTCE=1) and the NULLE bit is “1”, it must be overridden after setting the INTRTC to disable interrupt handling through the interrupt mask flag register. And the WAFG flag and the RTCIF flag must be cleared after rewriting. To set each alarm register (AL flag for RTCC1 register, alarm clock minute register (ALARMWM), alarm hour register (ALARMWH) and alarm clock day registers (ALARMWW)), the WALE set to “0” (consistent operation is invalid).	

WALIE	Operation control of alarm interrupt (INTRTC) function
0	No alarm clock consistent interruptions are generated.
1	Generates an alarm clock consistent interrupt.

WAFG	Alarm detection status flag
0	Alarm clock is inconsistent.
1	Alarm clock consistency detected.
This is a status flag that indicates that the alarm clock is detected consistently. Valid only when the WALE bit is “1” and becomes “1” when the alarm clock is detected to be consistent and 1 f_{RTC} clock has passed. Clear this flag by writing “0” to this flag. The action to write “1” is invalid.	

Figure 7-5 Format of real-time clock control register 1 (RTCC1) (2/2)

RIFG	Fixed-period interrupt status flag
0	No fixed-cycle interruptions are generated.
1	A fixed-cycle interruption is generated.
This is a status flag that indicates a fixed-cycle interruption. This flag is “1” when a fixed-period interrupt occurs. Clear this flag by writing “0” to this flag. The action to write “1” is invalid.	

RWST	Wait status flag for the real-time clock
0	The counter is running.
1	It is in the counter read/write mode.
This is the state that indicates whether the setting of the RWAIT bit is valid or not. The read and write count value must be read and written after confirming that this flag is “1”.	

RWAIT	Wait control of the real-time clock
0	Set to run as a counter.
1	Set the SEC~YEAR counter to stop running and enter the read and write mode of the counter.
This bit controls the operation of the counter. To read and write the 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”. It takes up to 1 f_{RTC} clock from placing the RWAIT set to “1” to being able to read and write the count value (RWSt=1). If an internal counter (16-bit) overflow occurs when the RWAIT bit is “1”, the overflow is maintained and the count is incremented after the RWAIT bit becomes “0”. However, when the seconds count register is written, the overflow is not kept in the state.	

Remark 1. Fixed-cycle interrupts and alarm clock consistent interrupts use the same interrupt source (INTRTC). In the case of using these two interrupts at the same time, INTRTC can occur to determine which interrupt occurred by acknowledging the fixed-period interrupt status flag (RIFG) and the alarm detection status flag (WAFG).

2. If the seconds count register (SEC) is written, the internal counter (16-bit) is cleared.

7.3.5 Clock error correction register (SUBCUD)

This is a register that can correct clock speed with high accuracy by changing the overflow value (reference value: 7FFFH) from the internal counter (16 bits) to the second count register (SEC). The SUBCUD register is set via a 16-bit memory operation command. After the reset signal is generated, the value of this register becomes “00 00H”.

Figure 7-6 Format of clock error correction register (SUBCUD)

Address: 0x40044F34H After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8
SUBCUD	THEV	0	0	F12	F11	F10	F9	F8

	7	6	5	4	3	2	1	0
	F7	F6	F5	F4	F3	F2	F1	F0

DEV	Timing setting for correction of clock errors
0	Correction of clock error is performed when the seconds are “00”, “20”, “40” (every 20 seconds).
1	Correction of clock error is performed only when the second bit is “00” (every 60 seconds).
Writing of the SUBCUD register is prohibited during the following period: • DVA=0:SEC=00H, 20H, 40H period • DVE = 1: SEC=00H period	

F12	Setting of clock error correction value
0	{(F11,F10,F9,F8,F7,F6,F5,F4,F3,F2,F1,F0)–1} ×2 added
1	{(/F11,/F10,/F9,/F8,/F7,/F6,/F5,/F4,/F3,/F2,/F1,/F0)+1} ×2 reduced
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, 0, 0, 0) or (1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1), no correction of clock error is performed. Range of correction values: (F12=0)2, 4, 6, 8, ... , 8186, 8188 (F12=1)–2, –4, –6, –8, …… , –8186, –8188	

Note: The “/” indicates the value after the inverse of each bit.

The range of corrections that can be made by clock error correction registers (SUBCUDs) is shown below.

	DEV = 0 (correction every 20 seconds)	DEV = 1 (correction every 60 seconds)
Correctable range	-12496.9ppm~12496.9ppm	-4165.6ppmto4165.6ppm
Max. quantization error	±1.53ppm	±0.51ppm
Min. resolution	±3.05ppm	±1.02ppm

Note When the correction range is outside the range of –4165.6ppm to 4165.6ppm, the DEV bit must be “0”.

7.3.6 Second count register (SEC)

This is an 8-bit register that represents the second count value from 0 to 59 (decimal). Increment counting is performed by overflow of the internal counter (16-bit).

At write time, data is first written to the buffer and to the counter after up to 2 FRTC clocks. The decimal 00 to 59 must be set in BCD code.

The SEC registers are set via 8-bit memory operation instructions. After the reset signal is generated, the value of this register becomes “00H”.

Figure 7-7 Format of second count register (SEC)

Address: 0x40044F52 After reset: 00H R/W

Symbol	0	7	6	5	4	3	2	1
SEC	0	SEC40	SEC20	SEC10	SEC8	SEC4	SEC2	SEC1

Note To read and write this register while the counter is running (RTCE=1), you must follow the steps described in “7.4.3”.

Note If you write the seconds count register (SEC), the internal counter (16 bits) is cleared.

7.3.7 Minute count register (MIN)

This is an 8-bit register that represents the minute count value from 0 to 59 (decimal). The count is incremented by the overflow of the seconds counter.

During writing, data is first written to the buffer and to the counter after up to 2 f_{RTC} clocks. Overflow of the seconds count register is ignored during a write operation and set to a write value. The decimal 00 to 59 must be set in BCD code.

MIN registers are set via 8-bit memory operation instructions. After the reset signal is generated, the value of this register becomes “00H”.

Figure 7-8 Format of minute count register (MIN)

Address: 0x40044F53 After reset: 00HR/W

Symbol	7	6	5	4	3	2	1	0
MIN	0	MANDN40	MIN20	MANDN10	MIN8	MIN4	MIN2	MIN1

Note To read and write this register while the counter is running (RTCE=1), you must follow the steps described in “7.4.3 Read and write to the real-time clock counter.

7.3.8 Hour count register (HOUR).

This is an 8-bit register that represents the hour count value in 00 ~ 23 or 01 ~ 12, 21 ~ 32 (decimal). Incremental counting is performed by means of the overflow of the minute counter.

At write time, data is first written to the buffer and to the counter after up to 2 f_{RTC} clocks. Overflow of the minute count register is ignored during a write operation and set to a write value.

The time system must be set at bit3 (AMPM) of register 0 (RTCC0) according to the real-time clock control register 0 (RTCC0), and the decimal 00 ~ in BCD code 23 or 01~12, 21~32.

If you change the value of the AMPM bit, the value of the HOUR register becomes the corresponding value of the time system being set. The Hour register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register changes to “12H”.

However, if the AMPM bit “1” is removed after the reset, the value of this register becomes “00H”.

Figure 7-9 Hour count register (HOUR)

Address: 0x40044F54 After reset: 12HR/W

Symbol	7	6	5	4	3	2	1	0
HOUR	0	0	HOUR20	HOUR10	HOUR8	HOUR4	HOUR2	HOUR1

Note 1 When the AMPM bit is selected as “0” (12-hour system), 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 7.4.3 “Read and write to the real-time clock counter“.

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 7-2.

Table 7-2 Representation of the time bits

24-hour (AMPM=1)		12-hour (AMPM=0)	
Time	HOUR register	Time	HOUR register
0 o'clock	00H	AM 12 o'clock	12H
1 o'clock	01H	AM 1 o'clock	01H
2 o'clock	02H	AM 2 o'clock	02H
3 o'clock	03H	AM 3 o'clock	03H
4 o'clock	04H	AM 4 o'clock	04H
5 o'clock	05H	AM 5 o'clock	05H
6 o'clock	06H	AM 6 o'clock	06H
7 o'clock	07H	AM 7 o'clock	07H
8 o'clock	08H	AM 8 o'clock	08H
9 o'clock	09H	AM 9 o'clock	09H
10 o'clock	10H	AM 10 o'clock	10H
11 o'clock	11H	AM 11 o'clock	11H
12 o'clock	12H	PM 12 o'clock	32H
13 o'clock	13H	PM 1 o'clock	21H
14 o'clock	14H	PM 2 o'clock	22H
15 o'clock	15H	PM 3 o'clock	23H
16 o'clock	16H	PM 4 o'clock	24H
17 o'clock	17H	PM 5 o'clock	25H
18 o'clock	18H	PM 6 o'clock	26H
19 o'clock	19H	PM 7 o'clock	27H
20 o'clock	20H	PM 8 o'clock	28H
21 o'clock	21H	PM 9 o'clock	29H
22 o'clock	22H	PM 10 o'clock	30H
23 o'clock	23H	PM 11 o'clock	31H

When the AMPM bit is “0”, the value of the HOUR register is 12 hours; When the AMPM bit is “1”, the value of the HOUR register is 24 hours.

At 12 hours, bit5 of the HOUR register indicates morning/afternoon. The morning (AM) is “0” and the afternoon (PM) is “1”.

7.3.9 Day count register (DAY)

This is an 8-bit register that represents the daily count value from 1 to 31 (decimal). The count is incremented by the overflow of the hour counter. The counter performs the following counts.

- 01 to 31 (January, March, May, July, August, October, December)
- 01 to 30 (April, June, September, November)
- 01 to 29 (February, leap year)
- 01 to 28 (February, normal year)

When data is written to this register, it is written to a buffer and then to the counter up to two cycles of F_{RTC} later. Even if the hour count register overflows while this register is being written, this register ignores the overflow and is set to the value written. Set a decimal value of 01 to 31 to this register in BCD code.

The DAY register is set by an 8-bit memory manipulation instruction. After a reset signal is generated, the value of this register becomes "01H".

Figure 7-10 Format of day count register (DAY)

Address: 0x40044F56H After reset: 01H R/W

Symbol	0	7	6	5	4	3	2	1
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 "7.4.3 Read and write to the real-time clock counter.

7.3.10 Week count Register (WEEK)

This is an 8-bit register that represents the day of the week count value from 0 to 6 (decimal). Increment counting synchronized with the day counter.

At write time, data is first written to the buffer and to the counter after up to 2 f_{RTC} clocks. The decimal 00 to 06 must be set in BCD code.

The WEEK register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register becomes “00H”.

Figure 7-11 Format of week count register (WEEK)

Address: 0x40044F55H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
WEEK	0	0	0	0	0	WEEK4	WEEK2	WEEK1

Note 1 The corresponding values of the month count register (MONTH) and the day count register (DAY) are not automatically saved to the week register (WEEK). The following settings must be made after the reset is released:

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

2. To read and write this register while the counter is running (RTCE=1), you must follow the steps described in “7.4.3 Read and write to the real-time clock counter“.

7.3.11 Month count register (MONTH)

This is an 8-bit register that represents the monthly count value from 1 to 12 (decimal). The count is incremented by the overflow of the day counter.

At write time, data is first written to the buffer and to the counter after up to 2 fRTC clocks. Overflow of the day count register is ignored during the write operation and set to a write value. The decimal 01~12 must be set in BCD code.

The MONTH register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register changes to “01H”.

Figure 7-12 Format of month count register (MONTH)

Address: 0x40044F57H After reset: 01H R/W

Symbol	7	6	5	4	3	2	1	0
MONTH	0	0	0	MONTH10	MONTH8	MONTH4	MYTH2	MONTH1

Note To read and write this register while the counter is running (RTCE=1), you must follow the steps described in “7.4.3Read and write to the real-time clock counter“.

7.3.12 Year Count Register (YEAR)

This is an 8-bit register that represents the annual count value from 0 to 99 (decimal). Increment counts by overflow of the month counter (MONTH).

00, 04, 08, ... 92, 96 are leap years.

At write time, data is first written to the buffer and to the counter after up to 2 fRTC clocks. Overflow of the MONTH register is ignored during the write operation and set to the write value. The decimal 00 to 99 must be set in BCD code. The YEAR register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register becomes “00H”.

Figure 7-13 Format of year count register (YEAR)

Address: 0x40044F58H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
YEAR	YEAR80	YEAR40	ANDEAR20	YEAR10	YEAR8	ANDEAR4	YEAR2	YEAR1

Note To read and write this register while the counter is running (RTCE=1), you must follow the steps described in “7.4.3 Read and write to the real-time clock counter“.

7.3.13 Alarm minute register (ALARMWM)

This is the register that sets the alarm for minutes.

The ALARMWM register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register becomes “00H”.

Note The decimal 00 to 59 must be set in BCD code. If you set a value outside the range, the alarm is not detected.

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

Address: 0x40044F5AH After reset: 00H R/W

Symbol	0	7	6	5	4	3	2	1
ALARMWM	0	M40	WM20	WM10	WM8	InM4	WM2	InM1

7.3.14 Alarm hour register (ALARMWH)

This is the register that sets the alarm for hours.

The ALARMWH register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register changes to “12H”.

However, the value of this register is 00H if the AMPM bit is set to 1 after reset.

Note The decimal 00~23 or 01~12, 21~32 must be set in BCD code. If you set a value outside the range, the alarm is not detected.

Figure 7-15 Format of Alarm Hour Register (ALARMWH)

Address: 0x40044F5BH After reset: 12H R/W

Symbol	0	7	6	5	4	3	2	1
ALARMWH	0	0	WH20	InH10	InH8	WH4	InH2	WH1

Note When the AMPM bit is selected as “0” (12-hour system), bit5 (WH20) of the ALARMWH register represents AM (0) /PM (1).

7.3.15 Alarm week register (ALARMWW)

This is the register that sets the alarm for the week.

The ALARMWW register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register becomes “00H”.

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

Address: 0x40044F5CH after reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ALARMWW	0	InW6	InW5	InW4	InW3	InW2	WW1	InW0

An example of setting the alarm time is as follows.

Time of Alarm	Day							12 hours				24-hour			
	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Hour 10	Hour 1	Minute 10	Minute 1	Hour 10	Hour 1	Minute 10	Minute 1
Every day 0:00 a.m				1				1							0
Every day 1.30 a.m															0
Every day 11.59 a.m															9
Mon ~ Fri 0:00 p.m															0
Sunday 1:30 p.m															0
Mon, Wed, Fri 11:59 p.m															9

7.3.16 Port mode register and port register

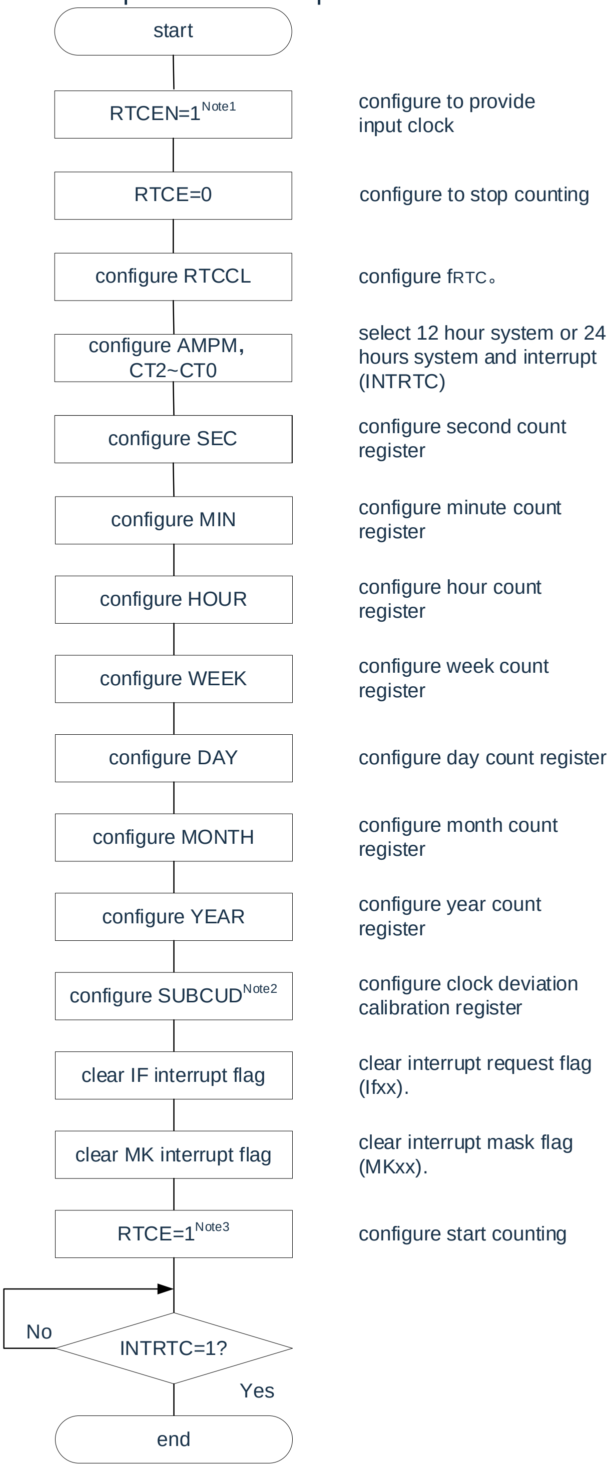
When the multiplexed port of the RTC1HZ output pin is output at 1Hz, the bits of the port-mode control register (PMCxx), the bits of the port-mode register (PMxx), and the port register (Pxx) corresponding to each port must be used) set to “0”.

The Port Mode Register (PMxx), Port Register (Pxx), and Port Mode Control Register (PMCxx) set vary by product. For details, please refer to “2.5 Register Settings When Using the Multiplexing Function”.

7.4 Operation of real-time clock

7.4.1 Start of real-time clock operation

Figure 7-17 Steps to start the operation of the real time clock


start
RTCEN=1[Note1]
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 SUBCUD[Note2]
configure clock deviation calibration register
clear IF interrupt flag
clear interrupt request flag (Ifxx).
clear MK interrupt flag
clear interrupt mask flag (MKxx).
RTCE=1[Note3]
configure start counting
No
INTRTC=1?
Yes
end

Note 1 The RTCEN set to “1” must first be set in a stable state of the Count Clock (f_{RTC}) oscillation.

2. This is only the case when the clock error needs to be corrected. For the calculation of the correction value, please refer to “7.4.6 Clock Error Correction Example for Real-Time Clock”.

3. If you do not wait for the INTRTC bit to change to “1” after the RTCE bit is “1” and then transfer to the sleep mode, please check the procedure in “7.4.2 Shifting to sleep mode after starting operation”.

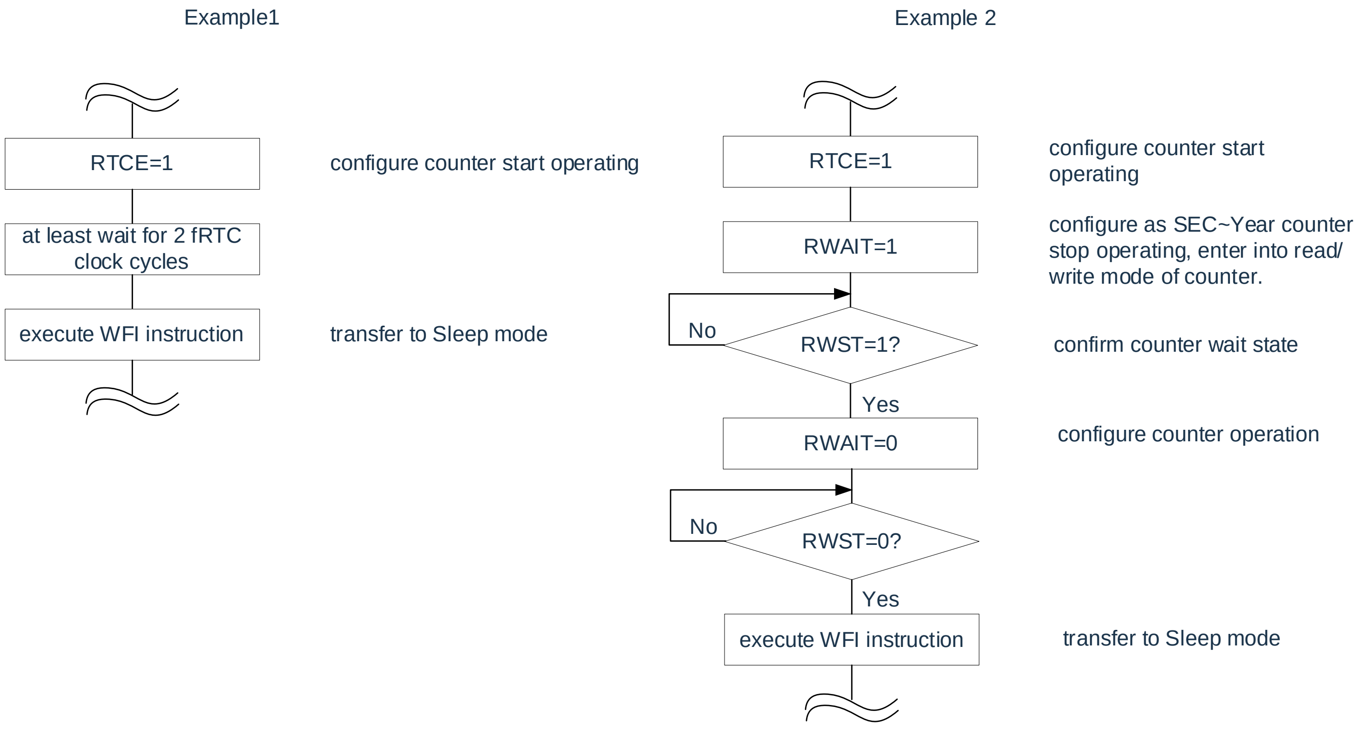
7.4.2 Shifting to sleep mode after starting operation

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 (f_{RTC}) elapsed after the RTCE set to “1” (see
- Figure 7-18).
- 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 7-18).

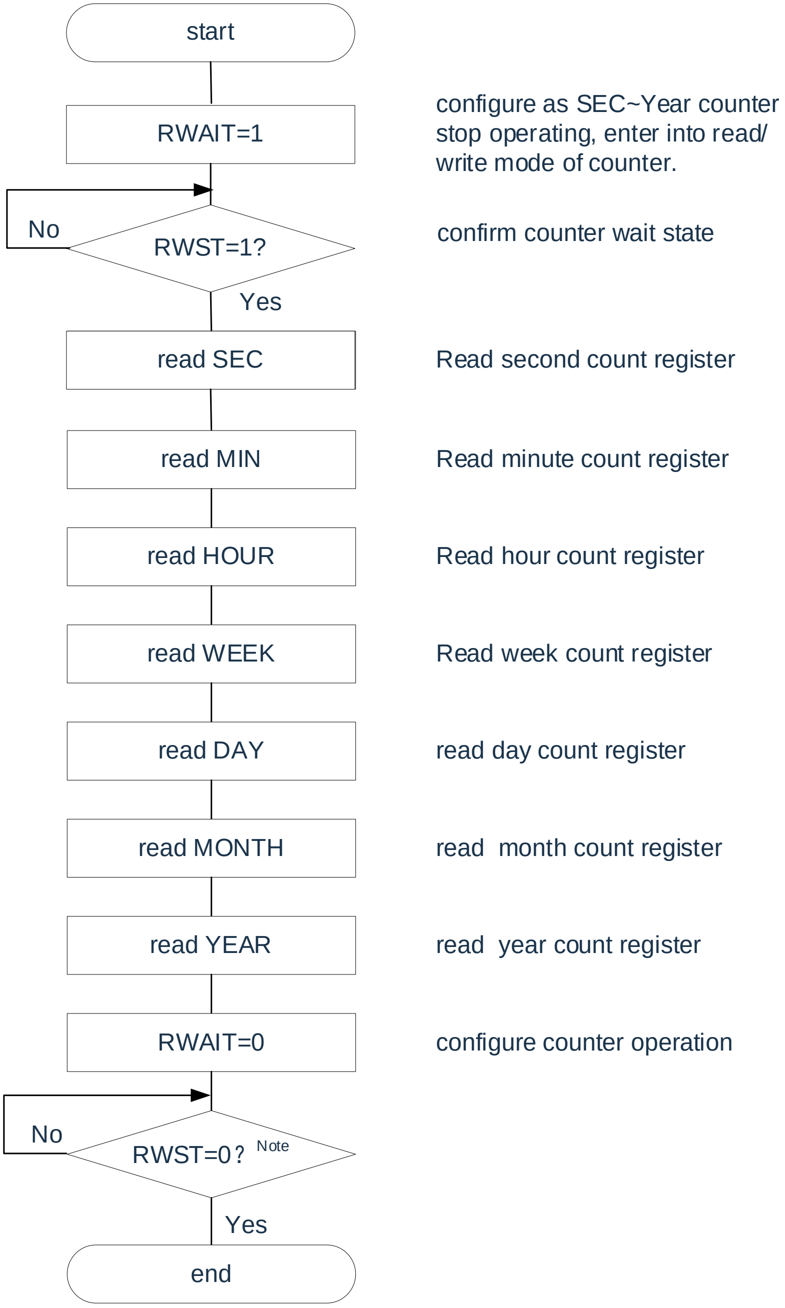
Figure 7-18 Procedure for shifting to sleep/deep sleep mode after setting RTCE bit to 1


Example1
RTCE=1
configure counter start operating
at least wait for 2 fRTC clock cycles
execute WFI instruction
transfer to Sleep mode
Example 2
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

7.4.3 Read and write to the real-time clock counter

Read or write the counter after setting “1” to RWAIT first. Set RWAIT to “0” after completion of reading or writing the counter.

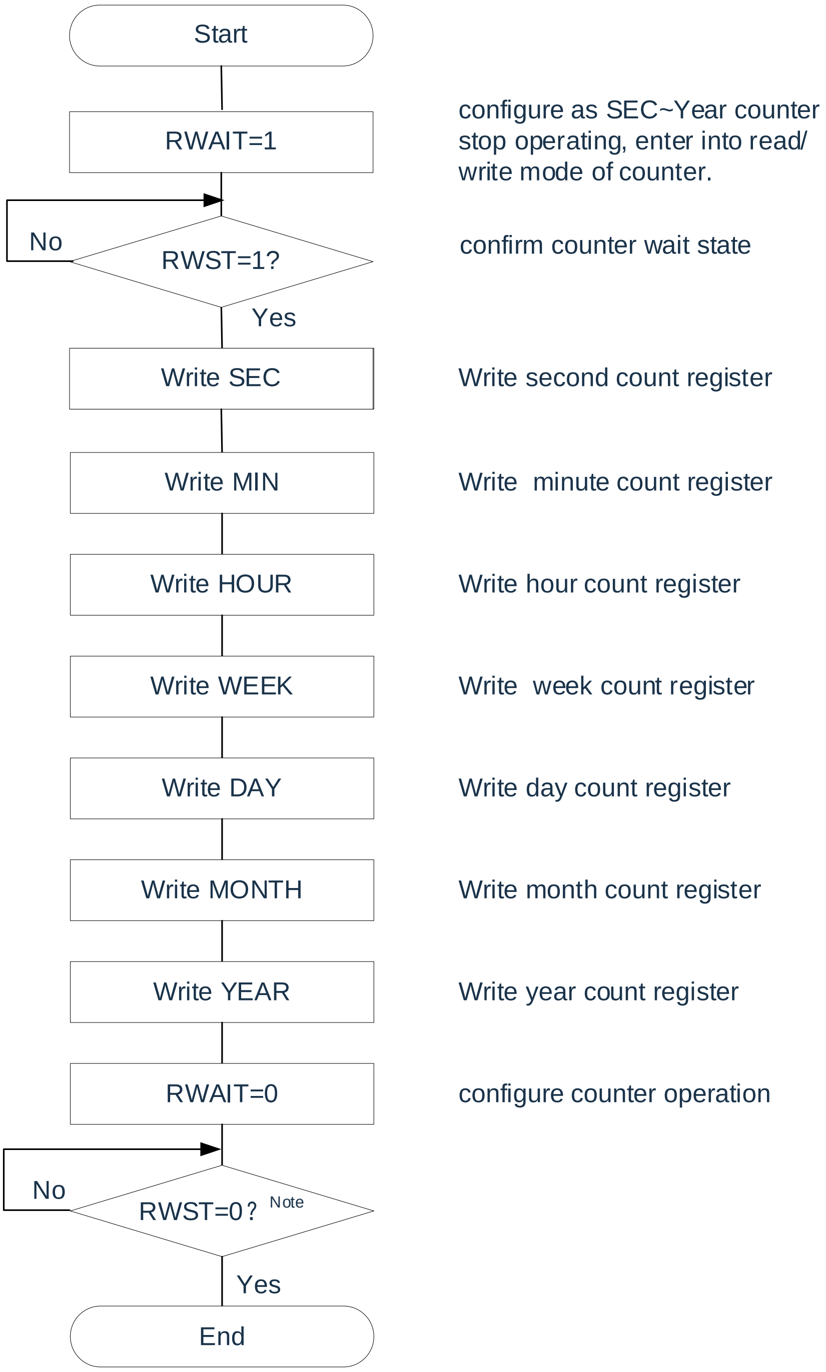
Figure 7-19 Read operation steps of the real-time clock counter


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 RWST bit must be confirmed to be “0” before shifting to sleep mode.

Note RWAIT set to “1” to the RWAIT set to “0” must be processed within 1 second.
Note Do not limit the order of read operations for seconds/minutes/hours/weeks/days/months/and year count register. You can read only some registers without reading all registers.

Figure 7-20 Read operation steps of the 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 Be sure to confirm that the RWST bit is “0” before moving to SLEEP mode.

Note 1 RWAIT set to “1” to the RWAIT set to “0” must be processed within 1 second.

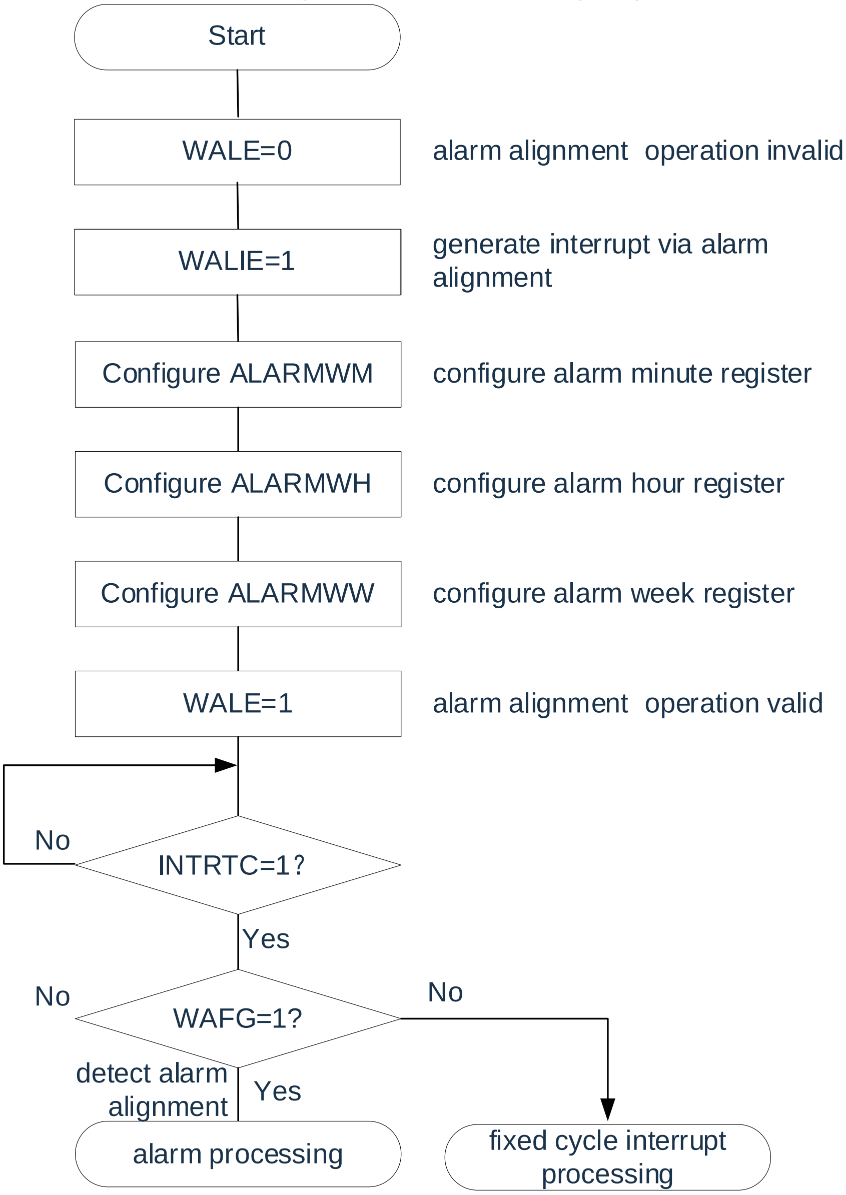
2. To overwrite the SEC, MIN, HOUR, WEEK, DAY, in the counter running (RTCE=1). When the MONTH and YEAR registers are registered, the INTRTC must be overridden after the interrupt mask flag register is set to disable interrupt handling, and the WAFG flag and RIFG must be cleared after the rewriting Logo and RTCIF Mark.

Note Do not limit the order of read operations for seconds/minutes/hours/weeks/days/months/and year count register. You can read only some registers without reading all registers.

7.4.4 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 7-21 Alarm setting steps

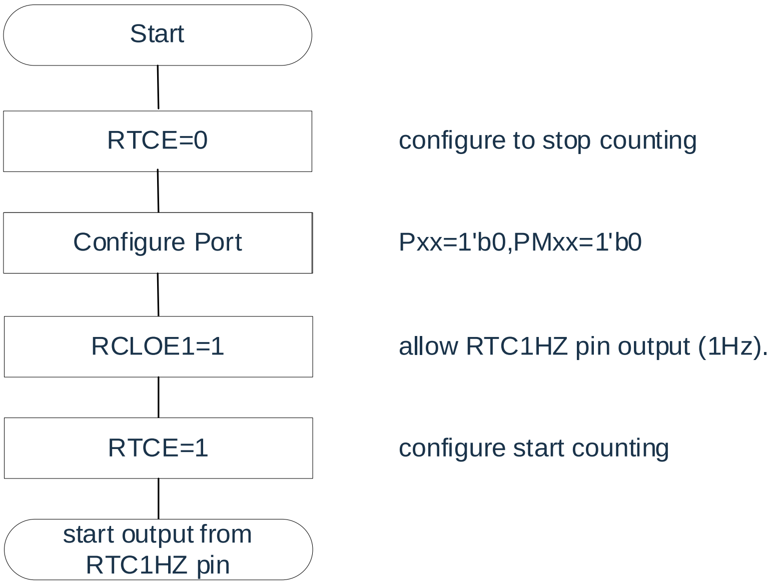

Start
WALE=0
alarm alignment operation invalid
WALIE=1
generate interrupt via alarm alignment
Configure ALARMWM
configure alarm minute register
Configure ALARMWH
configure alarm hour register
Configure ALARMWW
configure alarm week register
WALE=1
alarm alignment operation valid
No
INTRTC=1?
Yes
No
WAFG=1?
No
detect alarm alignment
Yes
alarm processing
fixed cycle interrupt processing

Note 1 There is no restriction on the order in which alarm clock minutes registers (ALARMWMs), alarm hour registers (ALARMWH), and alarm clock day registers (ALARMWWs) are written.

2. Fixed-cycle interrupts and alarm clock consistent interrupts use the same interrupt source (INTRTC). When using both interrupts at the same time, you can determine which interrupt occurred by acknowledging the fixed-cycle interrupt status flag (RIFG) and the alarm detection status flag (WAFG) when INTRTC occurs.

7.4.5 1Hz output of the real-time clock

Figure 7-22 Steps for setting 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 set to “1” must first be set in a stable state with the Count Clock (fSUB) oscillation.

7.4.6 Example of clock error correction for a real-time clock

Clock fast and slow correction can be performed with high accuracy by setting the clock error correction register.

An example of the calculation method for correcting values

The correction value when correcting the count value of the internal counter (16-bit) can be calculated using the following calculation formula. When the correction range is outside the range of –4165.6ppm to 4165.6ppm, set 0 to DEV.

(In the case of DEV=0).

Correction value[Note] = 1 minute correction count value÷3 = (oscillation frequency÷target frequency –1) ×32768×60÷3

(In the case of DEV=1).

Correction value[Note] = 1 minute correction count value = (oscillation frequency÷target frequency –1) ×32768×60

Note Correction is the clock error correction value calculated from the value of bit12~0 of the clock error correction register (SUBCUD).

(F12=0) Correction value = {(F11, F10, F9, F8, F7, F6, F5, F4, F3, F2, F1, F0) –1}×2

(F12=1) Correction value =–{(/F11,/F10,/F9,/F8,/F7,/F6,/F5,/F4,/F3,/F2,/F1,/F0)+1}×2

When (F12~F0) = (*, 0,0,0,0,0,0,0,0,0,0,0,0,*), no correction for clock error is performed. * is “0” or “1”.
/F12~/F0 is the value after you take the opposite (“000000000011”, “111111111100”).

Remark: 1 The correction values are 2, 4, 6, 8, ... , 8186, 8188 or –2, –4, –6, –8, ... , –8186, –8188.

2. The oscillation frequency is the value of the count clock (f_{RTC}) and can be calculated with the following Formula:

The output frequency of the RTC1HZ pin at clock error correction register is 0 32768 at the initial value (“00H”).

3. The frequency of interest is the frequency that is corrected using the clock error correction register.

Correction examples

Example of correcting from 32767.4Hz to 32768Hz (32767.4Hz+18.3ppm).

[Measurement of oscillation frequency]

When the clock error correction register (SUBCUD) is the initial value (“0000H”), the oscillation frequency of each product is measured by outputting a signal of approximately 1Hz from the RTC1HZ pin.

Note For the setting steps for the RTC1Hz output, refer to “1Hz Output of the 10.4.5 Real-Time Clock”.

[Calculation of correction value]

(The output frequency of the RTC1HZ pin is 0.9999817Hz).

Oscillation frequency = 327680.9999817×≈32767.4Hz

Suppose the frequency of interest is 32768Hz (32767.4Hz + 18.3ppm) and DEV=1.
The formula for calculating the correction value when the DEV bit is “1” applies.

Correction value =1 minute correction count value = (oscillation frequency÷target frequency –1) ×32768×60
= (32767.4÷32768–1) ×32768×60
=-36

[Calculation of the config value of (F12~F0)].

(In the case of correction value = –36).

Because the correction value is less than 0 (in case of speeding up), F12=1. Calculated from the correction value (F11~F0).

-{(/F11~/F0)–1}2=-36 ×
(/F11~/F0)=17
(/F11~/F0)=(0,0,0,0,0,0,0,1,0,0,0,1)
(F11~F0)=(1,1,1,1,1,1,1,0,1,1,1,0)

Therefore, the correction from 32767.4Hz to 32768Hz (32767.4Hz+18.3ppm) is as follows:

If you pass DEV=1 and correct values =-36 (bit12~0: 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0) to set the correction register, which can be corrected to 32768Hz (0ppm).

Chapter 8 15-Bit Interval Timer

8.1 Function of 15-bit interval timer

Interrupts (INTITs) are generated at any pre-set interval and can be used to wake up from deep sleep mode.

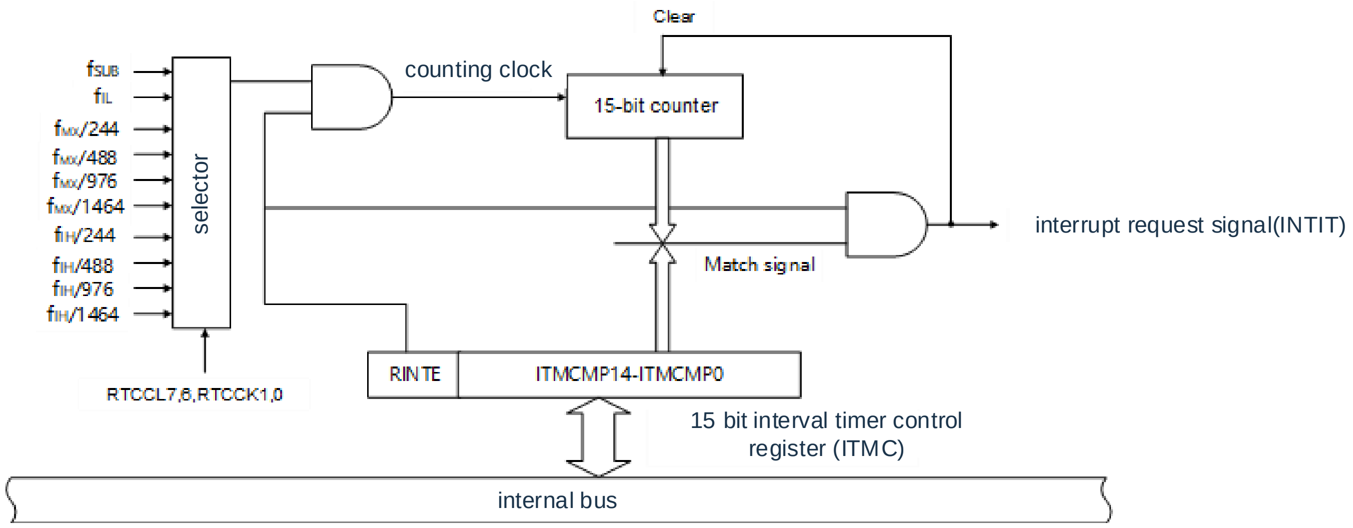
8.2 Structure of 15-bit interval timer

The 15-bit interval timer consists of the following hardware.

Table 8-1 Structure of 15-bit interval timer

Item	Structure
Counter	15-bit counter
Control registers	Peripheral enable register 0 (PER0).
	Real-time clock selection register (RTCCL).
	Control register (ITMC) for a 15-bit interval timer.

Figure 8-1 Diagram of 15-bit interval timer



8.3 Registers for controlling 15-bit interval timer

The 15-bit interval timer is controlled by the following registers.

- Peripheral enable register 0 (PER0).
- Real-time clock selection register (RTCCL).
- 15-bit interval timer control register (ITMC)

8.3.1 Peripheral enable register 0 (PER0).

The PER0 register is a register that sets the clock to be allowed or disallowed to be supplied to each peripheral hardware. Reduce power consumption and noise by stopping clocking unused hardware.

To use a 15-bit interval timer, bit7 (RTCEN) must be set to “1”. The PER0 register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register becomes “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	GODAEN	ADCEN	IICA0EN	SCI1EN	SCI0EN	TM41IN	TM40EN

RTCEN	Provides control of the real-time clock (RTC) and the input clock of a 15-bit interval timer
0	Stop supplying the input clock. • Cannot write the SFR used by the real-time clock (RTC) and 15-bit interval timer. • The real-time clock (RTC) and 15-bit interval timer are in a reset state.
1	An input clock is provided. • SFR can read and write to the real-time clock (RTC) and 15-bit interval timer.

8.3.2 Real-time clock selection register (RTCCL)

The real-time clock and the count clock (f RTC) of the 15-bit interval timer can be selected via RTCCL.

Figure 8-3 Format of real-rime 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	Clock source selection for real-time clocks, counting clocks for 15-bit interval timers
0	Select the high-speed System Clock (f_{MX}).
1	Select the high-speed internal oscillator (f_{hoco}).

RTCCKS1	RTCCKS0	RTCCL6	RTCCL5	Selection of operating clocks for real-time clocks, counting clocks for 15-bit interval timer
0	0	x	x	Subsystem clock (f_{SUB}).
0	1			Low-speed internal oscillator clock (f_{IL}) (WUTMMCK0 must be set to 1).
1	0	0	1	Main clock f_{max}/f_{hoco} (selected via RTCCL7)/1952
1	0	0	0	Main clock f_{max}/f_{hoco} (selected via RTCCL7)/1464
1	0	1	0	Main clock f_{max}/f_{hoco} (selected via RTCCL7)/976
1	1	0	0	Main clock f_{max}/f_{hoco} (selected via RTCCL7)/488
1	1	1	0	Main clock f_{max}/f_{hoco} (selected via RTCCL7)/244

8.3.3 15-bit interval timer control register (ITMC)

This is the register that sets the start and stop of operation of the 15-bit interval timer and the comparison value.

The ITMC registers are set via 16-bit memory operation instructions.

After the reset signal is generated, the value of this register changes to “7FFFH”.

Figure 8-4 Format of 15-bit interval timer control register (ITMC)

Address: 0x40044F50 After reset: 7FFFHR/W

symbol	15	14~0
ITMC	RINTE	ITCMP14 ~ ITCMP0

RINTE	15-bit interval timer operational control
0	Stop the run of the counter (clear the count).
1	Start the run of the counter.

ITCMP14~ITCMP0	15-bit interval timer comparison value setting
0001H	These bits produce a fixed-period interrupt of “counting clock cycles (ITCMP config value +1)”.
• • •	
7FFFH	
0,000H	Disable settings.
Example of an interrupt period when ITCMP1 4 to ITCMP0 is “0 001H” or “7FFFH” • ITCMP14 ~ ITCMP0=0001H, Count clock: f_{SUB}=32.768kHz 1/32.768[kHz] ×(1+1) =0.06103515625[ms]≈61.03[us] • ITCMP14~ITCMP0=7FFFH, Counting clock: f_{SUB}=32.768kHz1/32.768[kHz] ×(32767+1) =1000[ms]	

Note:

1. To change the RINTE bit from “1” to “0”, it must be overridden after setting INTIT to disable interrupt handling through the interrupt mask flag register. To restart the run (change from “0” to “1”), it must be set to allow interrupt processing after clearing the ITIF flag.
2. The read value of the RINTE bit is reflected after 1 count clock after the RINTE bit is set.
3. After moving from sleep mode to normal operating mode, if you want to set the ITMC register and move to sleep mode again, you must at least pass after confirming that the write value of the ITMC register is reflected or after setting the ITMC register 1 count clock is then transferred to sleep mode.
4. Only change the setting of the ITCMP14 to ITCMP0 bits when RINTE = 0. However, it is possible to change the settings of the ITCMP14 to ITCMP0 bits at the same time as when changing RINTE from “0” to “1” or “1” to “0”.

8.4 15-bit interval timer operation

8.4.1 15-bit interval timer operation timing

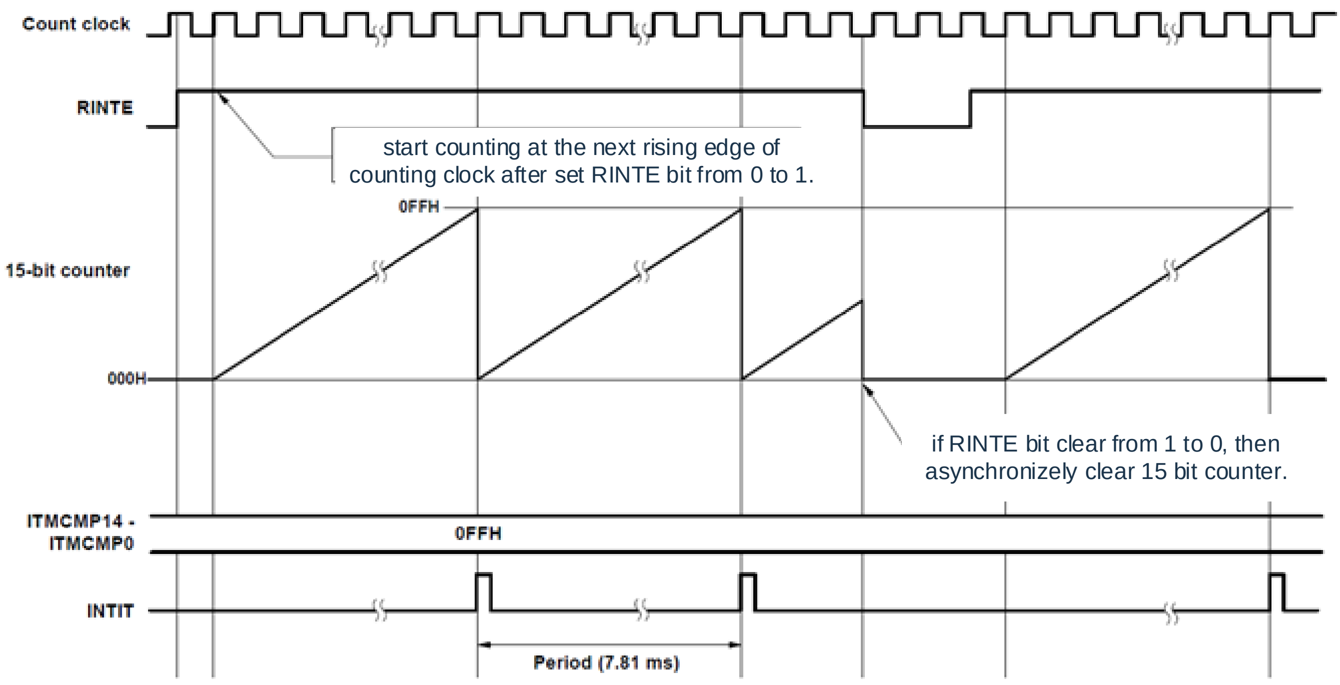
Runs as a 15-bit interval timer for repeated interrupt requests (INTITs) at intervals set by ITCMP14 to ITCMP0 bits. If you put the RINTE set to “1”, the 15-bit counter starts counting.

When the 15-bit count value and the ITCMP14~ITCMP0 config value are the same, clear the 15-bit count value to “0” and continue counting, while generating an interrupt request signal (INTIT).

The basic operation of the 15-bit interval timer is shown in Figure 8-5.

Figure 8-5 15-bit interval timer operation timing

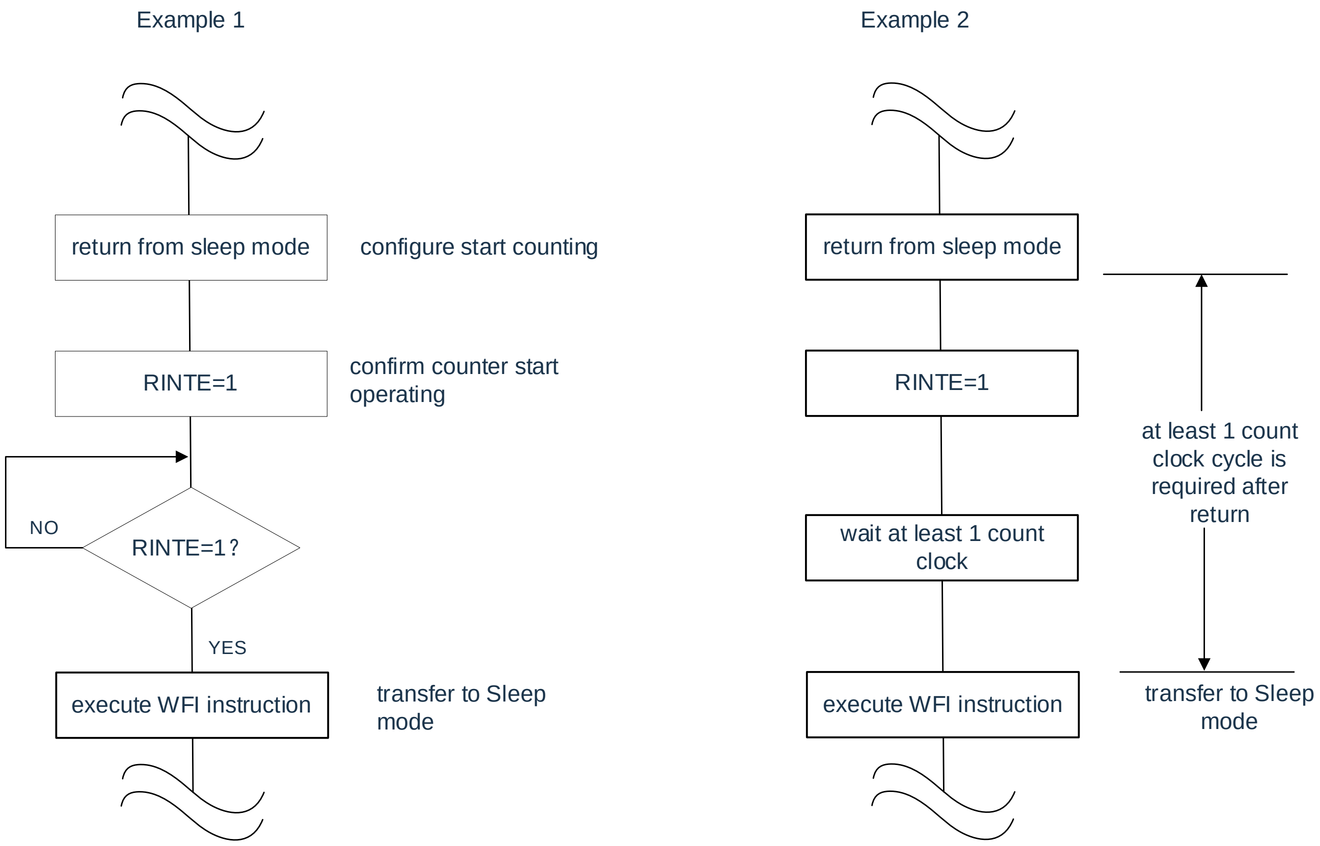
(ITCMP14~ITCMP0=0FFH, count clock: f_{SUB}=32.768kHz).)


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.
ITMCMP14 - ITMCMP0
0FFH
INTIT
Period (7.81 ms)

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

After returning from sleep mode, if you want to transfer the RINTE set to “1” and transfer it to sleep mode again, you must confirm that the RENTE bit is reflected after the RINTE set to “1”, or at least pass after the return After 1 count clock of time, it is transferred to sleep mode.

- After setting the RINTE to “1”, confirm that the RINTE bit changes to “1” by polling and then transfer to sleep mode (see the example in the figure below 1).
- After setting RINTE to 1, wait for at least one cycle of the count clock and then enter sleep mode (see Figure, Example 2).



Chapter 9 Clock output/Buzzer Output Controller

Note: The following sections in this chapter are mainly for 48-pin products.

9.1 Functions of clock output/buzzer output controller

The clock output is the function of the output to the clock of the peripheral IC, and the buzzer output is the function of the square wave of the output buzzer frequency.

This product has two clock output/buzzer output pins, where CLKBUZ0 can select any pin other than RESETB as a clock output or buzzer output, and CLKBUZ1 can be used with P15 as a clock output or buzzer output.

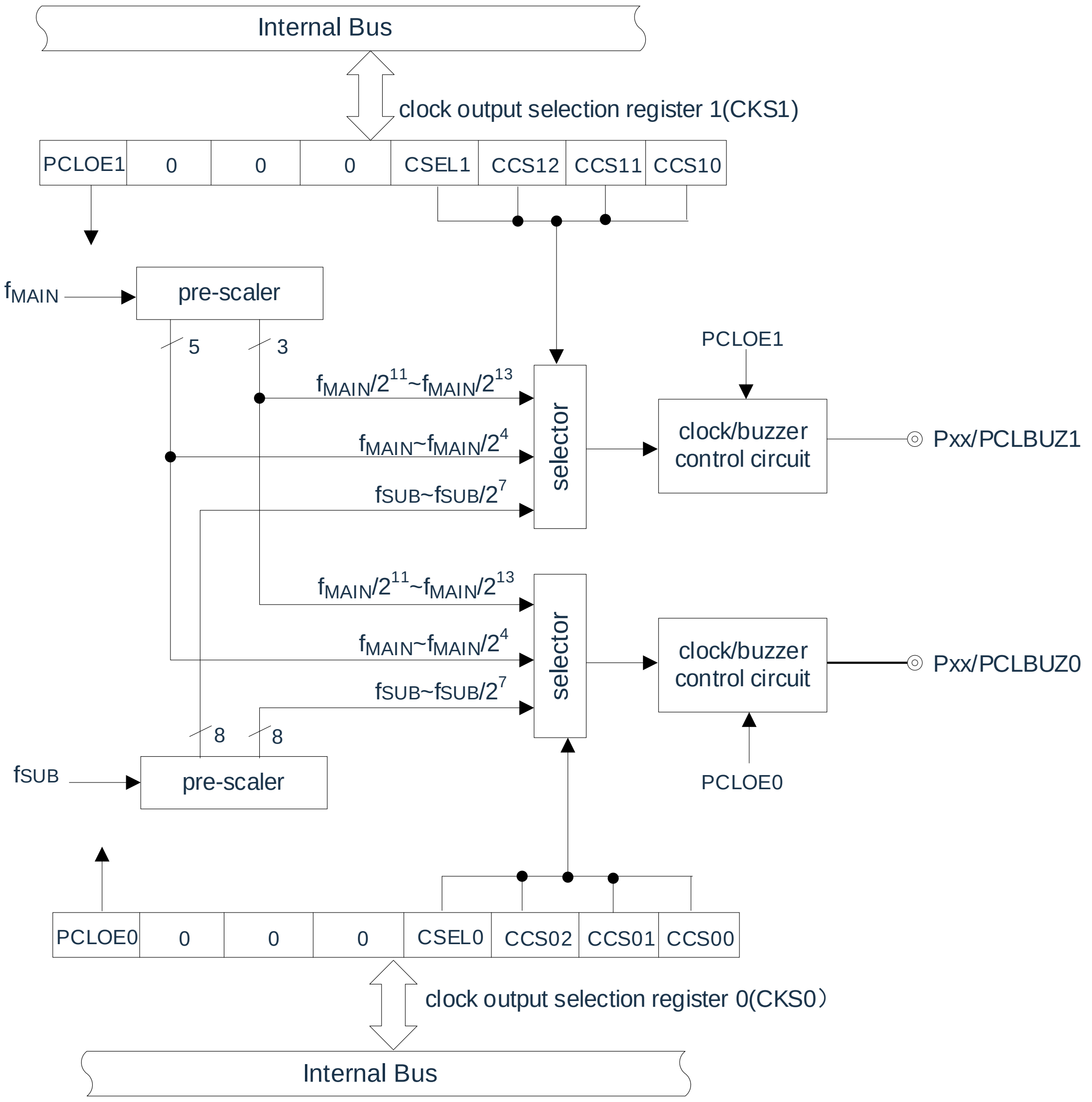
The CLKBUZn pin outputs the clock selected by clock output select register n (CKSn).

A block diagram of the clock output/buzzer output control circuit is shown in Figure 9-1.

Note: The subsystem clock (f_{SUB}) cannot be output from the CLKBUZn pin when the RTCLPC bit of the subsystem clock supply mode control register (OSMC) is “1” and in the SLEEP mode where the CPU is running with the subsystem clock (f_{SUB}).

Note: n=0, 1

Figure 9-1 Block diagram of clock output/buzzer output controller



Note For the frequency at which it can be output from the CLKBUZ0 pin and the CLKBUZ1 pin, please refer to “AC Features” in the Data Sheet.

9.2 Structure of clock output/buzzer output controller

The clock output/buzzer output controller consist of the following hardware.

Table 9-1 Structure of clock output/buzzer output controller

Item	Structure
Control registers	Clock output select register n (CKSn). Port mode control register (PMCxx), port mode register (PMxx), port multiplexing control register (PxxCFG).

9.3 Registers for controlling clock output/buzzer output controller

9.3.1 Clock output select register n (CKSn)

This is the output that allows or disables the clock output pin or the buzzer frequency output pin (CLKBUZn) and the register that sets the output clock.

The clock output of the CLKBUZn pin is selected via the CKSn register. The CKSn register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register becomes “00H”.

Figure 9-2 Format of clock output selection register n (CKSn)

Address: 0x40040FA5 (CKS0), 0x40040FA6 (CKS1) After reset: 00HR/W

Symbol	7	6	5	4	3	2	1	0
CKSn	PCRead	0	0	0	CHerselfLn	CCSn2	CCSn1	CCSn0

PCLOEn	CLKBUZn pin output enable/disable specification
0	Disable output (default).
1	Enable output.

CSELn	CCSn2	CCSn1	CCSn0	CLKBUZn pin output clock selection
0	0	0	0	f_{MAIN}
0	0	0	1	$f_{MAIN}/2$
0	0	1	0	$f_{MAIN}/2^2$
0	0	1	1	$f_{MAIN}/2^3$
0	1	0	0	$f_{MAIN}/2^4$
0	1	0	1	$f_{MAIN}/2^{11}$
0	1	1	0	$f_{MAIN}/2^{12}$
0	1	1	1	$f_{MAIN}/2^{13}$
1	0	0	0	f_{SUB}
1	0	0	1	$f_{SUB}/2$
1	0	1	0	$f_{SUB}/2^2$
1	0	1	1	$f_{SUB}/2^3$
1	1	0	0	$f_{SUB}/2^4$
1	1	0	1	$f_{SUB}/2^5$
1	1	1	0	$f_{SUB}/2^6$
1	1	1	1	$f_{SUB}/2^7$

Note The output clock must be used within the range of 16MHz. For details, please refer to “AC characteristics” of the data sheet.

Note 1 The switching of the output clock must be made after the output is set to disable (PCLOEn=0).

2. When selecting the master system clock (CSELn=0), if you want to transfer to deep sleep mode, you must set PCLOEn to “0” before executing WFI instructions; When the secondary system clock (CSELn=1) is selected, because the RTCLPC bit of the mode control register (OSMC) can be supplied on the secondary system clock with “0” and in deep sleep The clock is output in mode, so PCLOEn can be set to “1”.

3. When the RTCLPC bit of the Mode Control Register (OSMC) of the subsystem clock is “1” and the CPU is running at the secondary system clock (f_{SUB}). In sleep mode, the secondary system clock (f_{SUB}) cannot be output from the CLKBUZn pin.

Note 1. n =0, 1

2. $f_{MAIN:}$ Main system clock frequency

$f_{SUB:}$ Subsystem clock frequency

9.3.2 Registers for controlling the function of the clock output/buzzer output pin port

This product can multiplex the clock output/buzzer output function CLKBUZ0 to any port except RESETB, and CLKBUZ1 can be multiplexed to P15. When using the clock output/buzzer output function, the port multiplexing function configuration register (Pxx CFG), port register (Pxx), port mode register (PMxx), and port mode control register (PMCxx) must be set. For details, please refer to “Chapter 2 Pin Functions”.

The multiplexed port, which is configured as a clock output/buzzer output pin, must have “0” in the corresponding port register (Pxx), the bits of the port mode register (PMxx), and the port mode control register (PMCxx).

(Example) P20 as clock output/buzzer output (CLKBUZ0):

Set the P20 bit of Port Register 2 to “0”.

Set the PM20 bit of Port Mode Register 2 to “0”.

Set the PMC20 bit of Port Mode Control Register 2 to “0”.

Set P20CFG of Port Multiplex Function Configuration Register to “0x07”.

If P15 is used as clock output/buzzer output (CLKBUZ1):

Set the P15 bit of Port Register 1 to “0”.

Set the PM15 bit of Port Mode Register 1 to “0”.

Set the PMC15 bit of Port Mode Control Register 1 to “0”.

9.4 Operation of clock output/buzzer controller

It can be selected with 1 pin as clock output or buzzer output.

The CLKBUZ0 pin outputs the clock/buzzer selected by clock output select register 0 (CKS0).

The CLKBUZ1 pin outputs the clock/buzzer selected by clock output select register 1 (CKS1).

9.4.1 Operation of output pin

The CLKBUZn pin is output as follows:

1) Set the port multiplexing function configuration register (Pxx CFG), the port register (Pxx) corresponding to the port that will be used as the CLKBUZ0 pin, and the port mode register (PMxx). and the port mode control register (PMCxx) set to “0”.

2) Select bit0~3 (CCSn0~CCSn2) of register (CKSn) through the clock output of CLKBUZn pin CSELn) Select the output frequency (output is disabled).

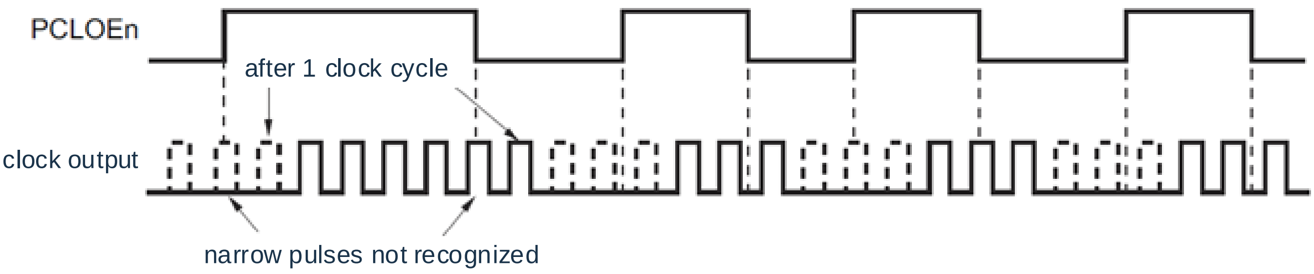
3) Set bit7 (PCLOEn) of the CKSn register to “1” to allow clock/buzzer output.

Note 1 CLKBUZ1 fixed multiplexing to P15 port, with CLKBUZ1, there is no need to set the port multiplexing function configuration register (Pxx CFG).

2. The control circuit used as the clock output starts or stops the clock output after one clock after the clock output (PCLOEn bit) is allowed or disable. Pulses with narrow widths are not output at this time. The timing of the output and clock output allowed or stopped by the PCLOEn bit is shown in Figure 9-3.

3.n=0, 1

Figure 9-3 Output timing of CLKBUZn pin



9.5 Cautions for clock output/buzzer output control circuitry

When the main system clock is selected as the CLKBUZn output (CSELn=0), the output width of CLKBUZn becomes narrower if it is shifted to deep sleep mode within 1.5 output clocks of the CLKBUZn pin after setting the stop output (PCLOEn=0).

Chapter 10 Watchdog Timer

10.1 Function of watchdog timer

The counting operation of the watchdog timer is set by the option byte (000C0H). The watchdog timer operates with a low-speed internal oscillator clock (f_{IL}).

A watchdog timer is used to detect a program that is out of control. When a program runaway is detected, an internal reset signal is generated.

The following situation is judged to be out of control of the program.

- When the watchdog timer's counter overflows
- When performing bit operation instructions on the Enable Register (WDTE) of the watchdog timer
- When writing data other than “ACH” to the WDTE register
- When writing data to the WDTE register during window closing

When a reset occurs due to a watchdog timer, set bit4 (WDTRF) of the reset control flag register (RESF) to “1”. For more information on RESF registers, refer to Chapter 21 Reset Functions. When 75% of the overflow time is reached +1/2f_{IL}, an interval interrupt can be generated.

10.2 Structure of watchdog timer

The watchdog timer consists of the following hardware.

Table 10-1 Structure of watchdog timer

Item	Structure
Counter	Internal counter (17 bits).
Control registers	Watchdog timer enable register (WDTE)

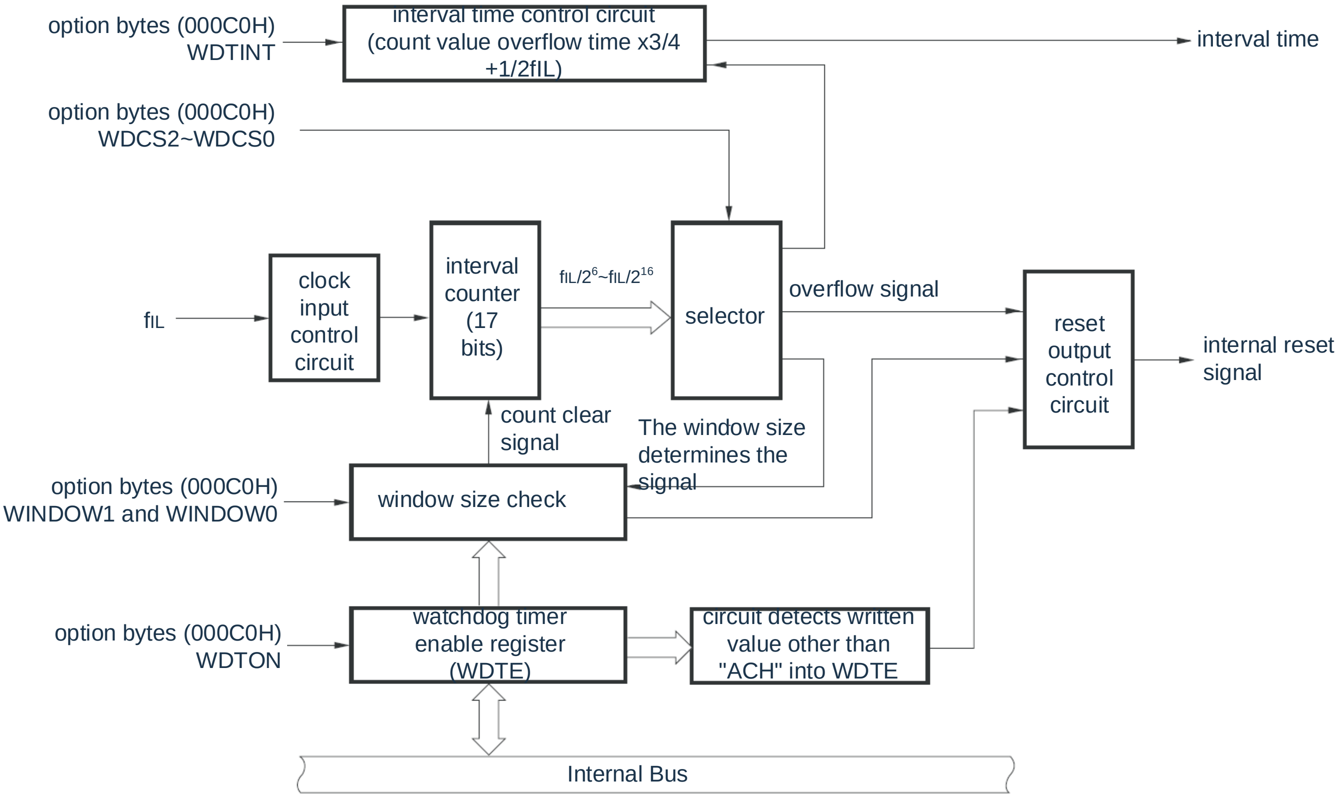
Control the operation of the counter and set the overflow time, window open period, and interval interruption through option bytes.

Table 10-2 Option bytes and watchdog timer settings

Setting content of the watchdog timer	Option byte (000C0H)
The setting of the interval interrupt of the watchdog timer	bit7(WDTINT)
The setting during the window opening	bit6 and bit5(WINDOW1, WINDOW0)
Counter run control of the watchdog timer	bit4(WDTON)
The setting of the overflow time of the watchdog timer	bit3~1(WDCS2~WDCS0)
The counter of the watchdog timer runs under control (while sleeping).	bit0(WDSTBYON)

Note For option bytes, refer to Chapter 26 Option Bytes.

Figure 10-1 Block diagram of watchdog timer


option bytes (000C0H)
WDTINT
interval time control circuit
(count value overflow time x3/4 +1/2fIL)
interval time
option bytes (000C0H)
WDCS2~WDCS0
fIL
clock input control circuit
interval counter (17 bits)
fIL/2^6~fIL/2^16
selector
overflow signal
reset output control circuit
internal reset signal
count clear signal
The window size determines the signal
option bytes (000C0H)
WINDOW1 and WINDOW0
window size check
option bytes (000C0H)
WDTON
watchdog timer enable register (WDTE)
circuit detects written value other than "ACH" into WDTE
Internal Bus

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

10.3 Registers for controlling watchdog timer

Control the watchdog timer through the watchdog timer enable register (WDTE).

10.3.1 Watchdog timer enable register (WDTE)

By writing “ACH” to the WDTE register, clear the counter for the watchdog timer and restart the count. The WDTE registers are set via 8-bit memory operation instructions. After the reset signal is generated, the value of this register changes to “9AH” or “1AH” Note.

Figure 10-2 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 setting value of the WDTON bit of the option byte (000C0H). To make the watchdog timer run, you must set the WDTON bit set to “1”.

Config value of the WDTON bit	Reset value of the WDTE register
0 (disable the counting run of the watchdog timer).	1AH
1 (enable the counting of watchdog timers to run).	9AH

Note 1 When a value other than “ACH” is written to the WDTE register, an internal reset signal is generated.

2. When performing bit operation instructions on the WDTE registers, an internal reset signal is generated.
3. The read value of the WDTE register is “9AH/1AH” (different from the write value (“ACH”)).

10.3.2 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 10-3 Format of LOCKUP control register (LOCKCTL)and its protection register (PRCR) (1/2)

Address: 40020405H After reset: 00HR/W

Symbol	7	6	5	4	3	2	1	0
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 10-3 Format of LOCKUP control register (LOCKCTL)and its protection register (PRCR) (2/2)

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

Symbol	7	6	5	4	3	2	1	0
PRCR	PRTKEY[7:1]							PRCR

PRCR	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
other	• PRCR is not writable

10.4 Operation of the watchdog timer

10.4.1 Operational control of the watchdog timer

1. When using the watchdog timer, set the following via option bytes (000C0H):
   - The bit4 (WDTON) of option byte (000C0H) must be set to "1" to enable the watchdog timer's count to run (after the reset is released, the counter starts running) (see Section 1 Chapter 26 Option Byte for details).

WDTON	Watchdog timer counter
0	Disable the counter to run (stop counting after the reset is released).
1	Enable the counter to run (start counting after the reset is released).

   - The overflow time must be set by bit3~1 (WDCS2~WDCS0) of option bytes (000C0H) (refer to Section 10.4.2 and Chapter 26).
   - You must set the window opening period by option bytes (000C0H) bit6 and bit5 (WINDOW1, WINDOW0) (see for details Section 10.4.2 and Chapter 26).

2. After the reset is released, the watchdog timer starts counting.
3. After starting the count and before the overflow time set by the option byte, if you write "ACH" to the allowed register (WDTE) of the watchdog timer, clear the watchdog timer and restart the count.
4. After that, the write operation of the WDTE register after the second time after the reset is released must be performed during the window open. If the WDTE register is written during window shutdown, an internal reset signal is generated.
5. If the overflow time is exceeded by not writing "ACH" to the WDTE register, an internal reset signal is generated. An internal reset signal is generated in the following cases:
   - When performing bit manipulation instructions on WDTE registers
   - When writing data other than "ACH" to the WDTE register

Note 1 Only when the allowed register (WDTE) of the watchdog timer is written for the first time after the reset is released, regardless of the window opening period, as long as the WDTE is written at any time before the overflow time, the watchdog timer is cleared and the count is restarted.

2. From writing "ACH" to the WDTE register to clearing the counter of the watchdog timer, it is possible to generate an error of up to 2 f-IL clocks.

3. Before the overflow of the count value, the watchdog timer can be cleared.

4. As shown below, the operation of the watchdog timer in sleep or deep sleep mode varies depending on the setting value of bit0 (WDSTBYON) of the option byte (000C0H).

	WDSTBYON=0	WDSTBYON=1
Sleep mode	Stop the watchdog timer from running.	Continue watchdog timer operation.
Deep sleep mode		

When the WDSTBYON bit is "0", the watchdog timer counts are restarted after the sleep or deep sleep mode is released. At this point, clear the counter to "0" and start counting.

When the X1 oscillation clock is run after the deep sleep mode is released, the CPU starts running after the oscillation stabilization time.

If the time from the release of the deep sleep mode to the overflow of the watchdog timer is short, the watchdog overflow occurs within the oscillation stabilization time and a reset occurs. Therefore, if you want to run with the X1 oscillation clock and clear the watchdog timer after the interval interrupt is released from the deep sleep mode, since the watchdog timer is not cleared after the oscillation settling time, this situation must be considered for the setting of the overflow time.

10.4.2 Watchdog timer overflow time setting

Set the overflow time of the watchdog timer by option bytes (000C0H) bit3~1 (WDCS2~WDCS0).

In the event of an overflow, an internal reset signal is generated. If the window opens before the overflow time, the allowed register for the watchdog timer is given

(WDTE) writes “ACH”, clears the count and restarts the count. The overflow times that can be set are shown below.

Table 10-3 Watchdog timer overflow time settings

WDCS2	WDCS1	WDCS0	Overflow time of the watchdog timer (f_{IL}=20kHz (MAX.))
0	0	0	$2^6/f_{IL}$(3.2ms)
0	0	1	$2^7/f_{IL}$(6.4ms)
0	1	0	$2^8/f_{IL}$(12.8ms)
0	1	1	$2^9/f_{IL}$(25.6ms)
1	0	0	$2^{11}/f_{IL}$(102.4ms)
1	0	1	$2^{13}/f_{IL}$(409.6ms)
1	1	0	$2^{14}/f_{IL}$(819.2ms)
1	1	1	$2^{16}/f_{IL}$(3276.8ms)

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

10.4.3 Setting window open period of watchdog timer

Set the watchdog timer window open by option bytes (000C0H) bit6 and bit5 (WINDOW1, WINDOW0). The window is summarized as follows:

- If you write "ACH" to the enable register (WDTE) of the watchdog timer while the window is open, the watchdog timer is cleared and the count is restarted.
- During window shutdown, even if "ACH" is written to the WDTE register, an exception is detected and an internal reset signal is generated.

Note Only when the WDTE register is written for the first time after the reset is released, regardless of the window open, as long as the WDTE is written at any time before the overflow time, the watchdog timer is cleared and the count is restarted.

The window opening period that can be set is shown below.

Table 10-4 Watchdog Timer Settings During Window Open

WINDOW1	WINDOW0	Window open period of watchdog timer
0	-	Disable settings
1	0	75%
1	1	100%

Note When bit0 (WDSTBYON) of option byte (000C0H) is "0", it is independent of the values of WINDOW1 and WINDOW0 bits. 100% during window open.

Note When setting the overflow time to $2^9/f_{IL}$, the window closing time and the open time are as follows.

	Setting of window open period	
	75%	100%
Window close time	0~12.8ms	None
Window open time	12.8~25.6ms	0~25.6ms

< When window open period is 75%>

- Overflow time:
  $2^9/f_{IL}$(MAX.)=2^9/20kHz(MAX.)=25.6ms
- Window closing time:
  0~$2^9/f_{IL}$(MIN.) ×(1–0. 75)=0~2^9/10kHz×0. 25=0~12.8ms
- Window open time:
  $2^9/f_{IL}$(MIN.) ×(1–0. 75)~$2^9/f_{IL}$(MAX.)=12.8~25.6ms

10.4.4 Setting watchdog timer interval interruption

Interval interrupt (INTWDTI) can be generated when 75% +1/2f_{IL} of the overflow time is reached by setting bit7 (WDTINT) of option byte (000C0H).

Table 10-5 Setting of watchdog timer interval interrupt

WDTINT	Watchdog timer interval interruption use/not used
0	Interval interrupts are not used.
1	Interval interrupts occur when 75% of the overflow time is reached +1/2f_{IL}.

Note when the X1 oscillation clock is run after the deep sleep mode is released, the CPU starts running after the oscillation stabilization time.

If the time from the release of the deep sleep mode to the overflow of the watchdog timer is short, the watchdog overflow occurs within the oscillation stabilization time and a reset occurs. Therefore, if you want to run with the X1 oscillation clock and clear the watchdog timer after the interval interrupt is released from the deep sleep mode, since the watchdog timer is not cleared after the oscillation settling time, this situation must be considered for the setting of the overflow time.

Note Continue counting even after the INTWDTI is generated (continue until “ACH” is written to the Allowed Register (WDTE) of the watchdog timer). If “ACH” is not written to the WDTE register before the overflow time, an internal reset signal is generated.

10.4.5 Operation of the 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 vibrate, 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 11 A/D Converter

The number of analog input channels for A/D converters varies by product, and detailed pins refer to the corresponding product data sheet.

Number of pins	32 pins	32-pin (-A) Note 1	40 pins	40 pins (-A) Note 1	44-pin (-A) Note 1	48 pins	48 pins (-A) Note 1	48-pin (-B) Note 1
Analog input channels	25ch	22ch	28ch	28ch	31ch	35ch	35ch	37ch
	(ANI0~ANI 3, YEARS8~ANI14, ANI16~ANI24, ANI29, ANI31~ANI33, ANI36)	(ANI0~ANI 3, YEARS8~ANI14, ANI16~ANI24, ANI27,ANI29)	(ANI0~ANI5, ANI8~ANI14, ANI16~ANI24, ANI29, ANI31~ANI34, ANI36)	(ANI0~ANI6, ANI8~ANI14, ANI16~ANI24, ANI27, ANI29~ANI32)	(ANI0~AN24, ANI27~AN32)	(ANI0~AN24, ANI27~AN36)	(ANI0~AN24, ANI27~AN36)	(ANI0~AN36)

Note 1. (-A) is limited to BJHH502Axxx-A series products. (-B) indicates that it is limited to BJHH502Axxx-B series products.

11.1 Function of A/D converter

An A/D converter is a converter that converts an analog input to a digital value, and an A/D converter has the following functions.

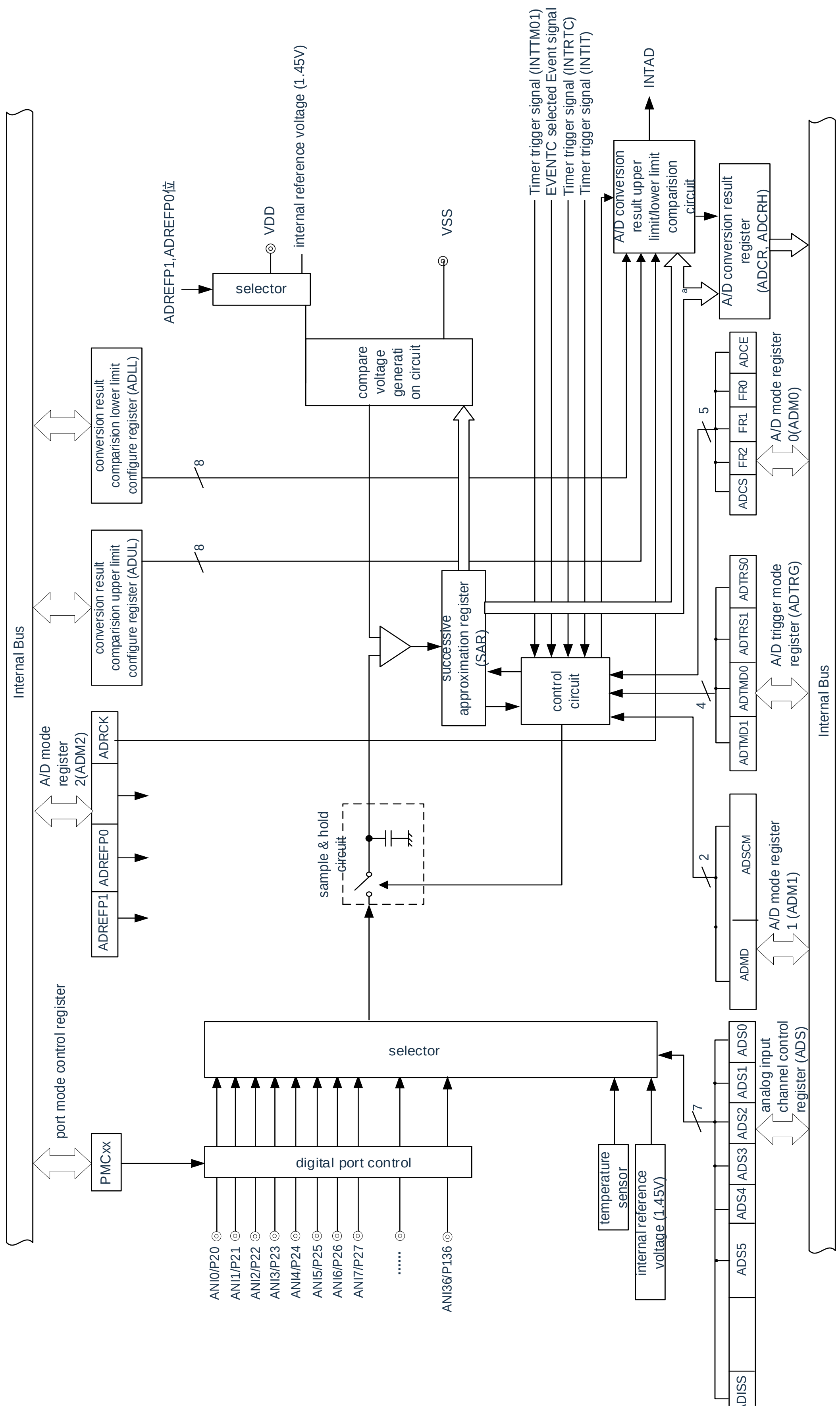
- A/D conversion with 12-bit resolution

Select a channel of analog inputs from ANI0 to ANI36 and a temperature sensor and repeat the A/D conversion with 12-bit resolution. For every A/D conversion that ends, an interrupt request (INTAD) is generated (the case of a select mode).

Various A/D conversion modes can be set through the following combination of modes.

Trigger mode	Software triggered	Start the conversion with software operation.
	Hardware trigger no-wait mode	Start the conversion by detecting a hardware trigger.
	Hardware trigger wait mode	In the transition standby state when the A/D power supply is cut off, the power supply is turned on by detecting the hardware trigger, and the conversion automatically begins after the A/D power supply stabilization waiting time.
Channel selection mode	Select mode	Select 1 channel of analog inputs for A/D conversion.
	Scan mode	A/D conversion of analog inputs for 4 channels is performed sequentially. Four consecutive channels from ANI0 to ANI1 5 can be selected as analog inputs.
Conversion mode	Single conversion mode	Performs 1 A/D conversion on the selected channel.
	Continuous conversion mode	Continuous A/D conversion of the selected channel until it is stopped by the software.
Sample time	Sample clock 4/8 ADCLK	The sampling time can be selected via the ADSMPWAIT register, which uses four conversion clocks (f_{AD}) by default.

Figure11-1 Block diagram of A/D converter


Internal Bus
conversion result comparison lower limit configure register (ADLL)
conversion result comparison upper limit configure register (ADUL)
A/D mode register 2(ADM2)
ADRCK
ADREFP0
ADREFP1
port mode control register
PMCxx
ADREFP1,ADREFP0位
selector
VDD
internal reference voltage (1.45V)
VSS
compare voltage generati on circuit
8
8
successive approximation register (SAR)
control circuit
sample & hold circuit
selector
digital port control
ANI0/P20
ANI1/P21
ANI2/P22
ANI3/P23
ANI4/P24
ANI5/P25
ANI6/P26
ANI7/P27
......
ANI36/P136
temperature sensor
internal reference voltage (1.45V)
Timer trigger signal (INTTM01)
EVENTC selected Event signal
Timer trigger signal (INTRTC)
Timer trigger signal (INTIT)
INTAD
A/D conversion result upper limit/lower limit comparision circuit
A/D conversion result register (ADCR, ADCRH)
ADCE
FR0
FR1
FR2
ADCS
5
A/D mode register 0(ADM0)
ADTRS0
ADTRS1
ADTMD0
ADTMD1
4
A/D trigger mode register (ADTRG)
ADSCM
ADMD
2
A/D mode register 1 (ADM1)
ADS0
ADS1
ADS2
ADS3
ADS4
ADS5
DISS
7
analog input channel control register (ADS)
Internal Bus

Note: Please refer to 0 for the selection of analog input channel ANIx

11.2 Control registers of A/D converter

The registers that control the A/D converter are as follows:

Register base address: CSC_BASE=4002_0420H; ADC_BASE=4004_5000H; PORT_BASE=4004_0000H

Register name	Register description	R/W	Reset value	Register address
PER0	Peripheral enable register 0	R/W	00H	CSC_BASE+20H
ADM0	A/D converter mode register 0	R/W	00H	ADC_BASE+00H
ADM1	A/D converter mode register 1	R/W	00H	ADC_BASE+02H
ADM2	A/D converter mode register 2	R/W	00H	ADC_BASE+04H
ADTRG	A/D converter trigger mode register	R/W	00H	ADC_BASE+06AM
ADS	Analog input channel specification register	R/W	00H	ADC_BASE+08AM
ADLL	Conversion result comparison lower limit setting register	R/W	00H	ADC_BASE+0AH
ADUL	Conversion result comparison upper limit setting register	R/W	00H	ADC_BASE+0BH
ADCR	12-bit A/D conversion result register	R	0,000H	ADC_BASE+0EH
ADCRH	8-bit A/D conversion result register	R	00H	ADC_BASE+0FH
ADSMPWAIT	A/D converter sampling time extension control register	R/W	00H	ADC_BASE+15H
PMCn	Port mode control register	R/W	Note 1	PORT_BASE+$^{\text{Note1}}$

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

Note 1: When selecting a channel through the ADS registers, the PMC register of the channel pin needs to be configured as an analog channel.

11.2.1 Peripheral enable register 0 (PER0)

The PER0 register is a register that sets the clock to be enable or disable to be supplied to each peripheral hardware. Reduce power consumption and noise by stopping clocking unused hardware.

To use an A/D converter, bit5 (ADCEN) must be set to “1”.

The PER0 register is set via an 8-bit memory operation command.

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

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

Reset value: 00H
R/W

	7	6	5	4	3	2	1	0
PER0	RTCEN	IRDAEN	ADCEN	IICA0EN	SCI1EN	SCI0IN	TM41IN	TM40EN

ADCEN	Control of the input clock of the A/D converter
0	Stop supplying the input clock. • Cannot write A/D converters using SFR. • The A/D converter is in a reset state.
1	An input clock is provided. • SFR can read and write to A/D converters used.

Note 1 To set up an A/D converter, you must first read and write the following registers in the ADCEN bit “1”. When the ADCEN bit is “0”, the value of the control register of the A/D converter is the initial value, ignoring the write operation (port mode control register (PMCxx except).

- 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)
- 12-bit A/D conversion result register (ADCR)
- 8-bit A/D conversion result register (ADCRH)
- A/D converter sampling time extension control register (ADSMPWAIT)

11.2.2 A/D converter mode register 0 (ADM0)

A register for setting the A/D conversion clock, conversion start, or stop. The ADM0 register is set with 8-bit memory operation instructions.

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

Figure11-3 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	Control of A/D conversion runs
0	Stop the conversion run. [while reading]. Stop the transition run/standby
1	Allow the conversion to run. [while reading]. When the software triggers the mode: Transitions the running state When the hardware triggers the wait mode: the A/D power supply waits for a steady state + transitions to run the state

ADCE	Operation control of the A/D voltage comparator$^{\text{Note 2}}$
0	Stop operation of the A/D voltage comparator.
1	Allow operation of the A/D voltage comparator.

Note 1 For details on FR2~FR0 bits and A/D conversion, please refer to “Table 11-3 Selection of A/D Conversion Time”.

2. The A/D converter needs 2us stabilization time to start operation. In software-triggered mode or hardware-triggered no-wait mode, if at least 2us elapses after setting the ADCE bit to "1" and then setting the ADCS bit to "1", the conversion result is valid. If the waiting time is less than 2us and the ADCS bit is set to "1", the conversion result must be ignored. In hardware-triggered wait mode, the 2us wait time is guaranteed by design.

Note 1 The FR2~FR0 bits must be changed in the transition stop state ADCS=0.

2. Disable the setting of ADCS=1 and ADCE=0.

3. It is forbidden to set the status of ADCS=0 and ADCE=0 to ADCS=1 and ADCE=1 through 8-bit operation instructions.

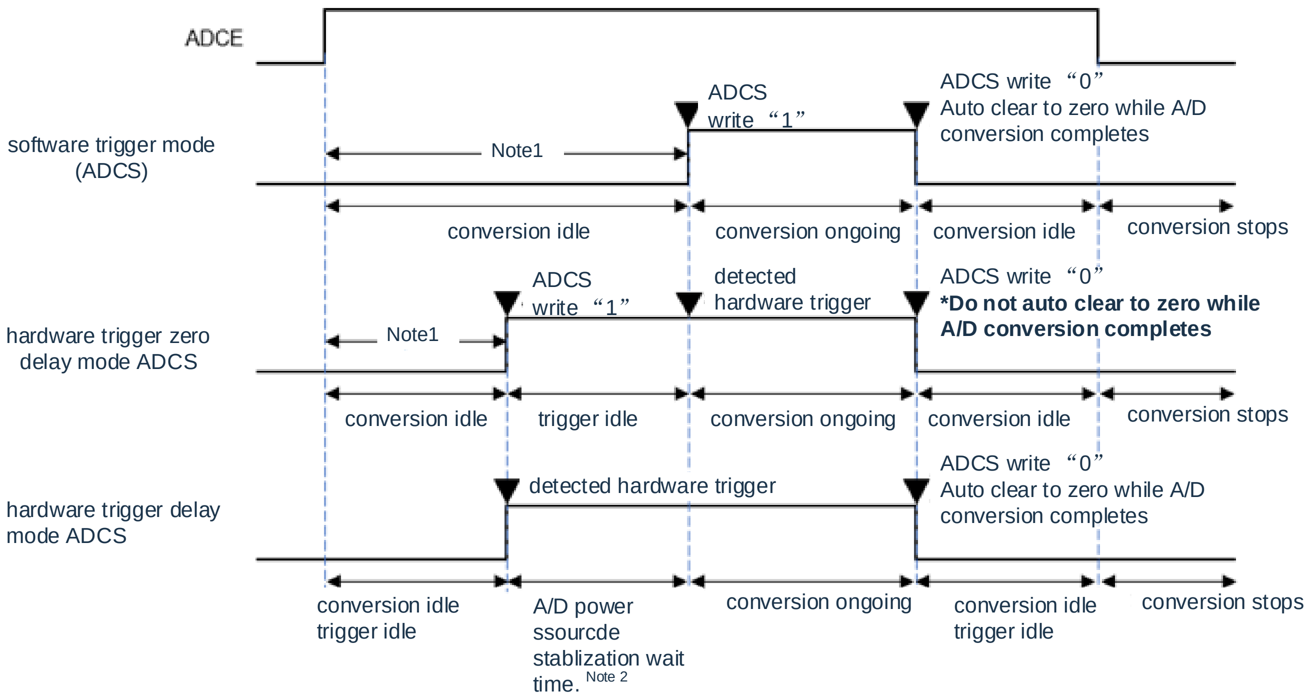
Table11-1 Configuration of ADCS and ADCE bits

ADCS	ADCE	A/D conversion operation
0	0	Transition stop
0	1	Transition standby
1	0	Disable settings.
1	1	Transition run

Table11-2 Setting and clearing conditions for ADCS bits

A/D conversion mode			Set condition	Clear the condition
Software triggered	Select the mode	Continuous conversion mode	When giving ADCS bits to write “1” time	When writing “0” to the ADCS bit
		Single conversion mode		• When writing “0” to the ADCS bit • Automatically clear “0” at the end of A/D conversion.
	Scan mode	Continuous conversion mode		When writing “0” to the ADCS bit
		Single conversion mode		• When writing “0” to the ADCS bit • When the set 4 channels are converted at the end, the “0” is automatically cleared.
Hardware trigger no-wait mode	Select the mode	Continuous conversion mode	When giving ADCS bits to write “1” time	When writing “0” to the ADCS bit
		Single conversion mode		• When writing “0” to the ADCS bit
	Scan mode	Continuous conversion mode		When writing “0” to the ADCS bit
		Single conversion mode		• When writing “0” to the ADCS bit
Hardware trigger wait mode	Select the mode	Continuous conversion mode	When the input hardware triggers	When writing “0” to the ADCS bit
		Single conversion mode		• When writing “0” to the ADCS bit • Automatically clear “0” at the end of A/D conversion.
	Scan mode	Continuous conversion mode		When writing “0” to the ADCS bit
		Single conversion mode		• When writing “0” to the ADCS bit • When the set 4 channels are converted at the end, the “0” is automatically cleared.

Figure11-4 Diagram of using various modes of A/D


ADCE
software trigger mode (ADCS)
Note1
ADCS write "1"
ADCS write "0"
Auto clear to zero while A/D conversion completes
conversion idle
conversion ongoing
conversion idle
conversion stops
hardware trigger zero delay mode ADCS
Note1
ADCS write "1"
detected hardware trigger
ADCS write "0"
*Do not auto clear to zero while A/D conversion completes
conversion idle
trigger idle
conversion ongoing
conversion idle
conversion stops
hardware trigger delay mode ADCS
detected hardware trigger
ADCS write "0"
Auto clear to zero while A/D conversion completes
conversion idle trigger idle
A/D power ssourcde stablization wait time. Note 2
conversion ongoing
conversion idle trigger idle
conversion stops

Note 1. In software-triggered mode or hardware-triggered wait-free mode, it takes at least 2us (TBD) to rise from the ADCE bit to the ADCS bit to stabilize the internal circuitry.

2. In hardware-triggered wait mode, the A/D power supply settling time of 1uis guaranteed by design.

Notice 1. To use the hardware trigger wait mode, setting the ADCS bit to "1" is prohibited (it is automatically switched to "1" when a hardware trigger signal is detected). However, the ADCS bit can be set to "0" in order to set the standby state for A/D conversion.

2. The ADCE bit must be overridden when the ADCS bit is "0" (Stop Transition/Transition Standby).

3. In order to end the A/D conversion, the hardware trigger interval must be set at least to the following time:

When hardware triggers no-wait mode: 2 f-CLK clocks + A/D conversion time

When the hardware triggers the wait mode: 2 f_{CLK} clock + A/D power supply stable wait time + A/D transition time

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

Table 11-3 A/D conversion time (1/2)

(1) No A/D power stabilization wait time (software trigger mode/hardware trigger no wait mode).

A/D converter mode register 0 (ADM0)			A/D sampling time extension register (ADSMPWAIT)	Convert frequency of clock ADCLK (f_{AD})	12-bit resolution conversion time $^{\text{Note 2}}$ ADC conversion time = (number of sample clocks + number of successive comparison clocks)/ fAD	
FR2	FR1	FR0	ADSMPWAIT		Number of ADC conversion clocks	ADC conversion time
0	0	0	0	fCLK/32	16 ADCLK (4 sample clocks +12 successive comparison clocks).	16/ fAD
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	fCLK/32	20 ADCLK (8 sample clocks +12 successive comparison clocks).	20/ fAD
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: To override the FR2~FR0 bits and ADSMPWAIT bits into different data, it must be done in the transition stop state (ADCS=0).

Note 2. Time required for an ADC conversion = (number of sample clocks + number of successive comparison clocks)/ fAD

The number of sample clocks can be adjusted via the ADSMPWAIT register, which defaults to four ADCLK.
The fastest clock supported by ADCLK is 8MHz.

Note fCLK: The clock frequency of the CPU/peripheral hardware
fAD: The ADC converts the clock frequency up to 8MHz.

Table11-4 A/D conversion times (2/2)

(2) There is an A/D power stabilization wait time (hardware triggered wait mode Note 1).

A/D converter mode register 0 (ADM0)			A/D sampling time extension register (ADSMPWAIT)	Convert frequency of clock ADCLK (f_{AD})	A/D power supply stabilization Time	Number of ADC conversion clocks	A/D power supply stabilization time +ADC conversion time Note 2
FR2	FR1	FR0	ADSMPWAIT				
0	0	0	0	fCLK/32	2us	16 ADCLK (4 sample clocks + 12 successive comparison clocks)	2us +16/fAD
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	fCLK/32	2us	20 ADCLK (8 sample clocks + 16 successive comparison clocks)	2us +20/fAD
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			

Notice 1. When the hardware trigger wait mode, the power supply settling time is guaranteed by the hardware design and does not need to be set. And in continuous conversion mode, the A/D power stabilization wait time occurs only after the hardware trigger is detected for the first time.

Notice 2. Time required for ADC conversion after hardware triggering = 2us + (number of sample clocks + number of successive comparison clocks)/fAD
The number of sample clocks can be adjusted via the ADSMPWAIT register, which defaults to four ADCLK.
The fastest clock supported by ADCLK is 8MHz.

Note 1. To override the FR2~FR0 bits and ADSMPWAIT bits into different data, it must be done in the transition stop state (ADCS=0).
2. The transition time in the hardware-triggered wait mode includes the A/D power stabilization wait time after the hardware trigger is detected.

Remark fCLK: Clock frequency of the CPU/peripheral hardware

11.2.3 A/D converter mode register 1 (ADM1)

This is the register that sets the A/D conversion mode.

The ADM1 register is set via 8-bit memory operation instructions.

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

Figure11-5 Format of A/D converter mode register 1 (ADM1)

Reset value: 00H
R/W

	7	6	5	4	3	2	1	0
ADM1	ADMD	0	0	0	ADSCM	0	0	0

ADMD	Setting of the A/D conversion channel selection mode
0	Select mode
1	Scan mode

ADSCM	Setting of the A/D conversion mode
0	Continuous conversion mode
1	Single conversion mode

Note: You must set bit6~4,2 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 the A/D conversion normally, the hardware trigger interval must be set at least to the following time:

When hardware triggers no-wait mode: 2 f-CLK clocks + A/D conversion time

When the hardware triggers the wait mode: 2 fCLK clock + A/D power supply stable wait time + A/D transition time

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

11.2.4 A/D converter mode register 2 (ADM2)

The ADM2 register is set by 8-bit memory operation instructions.

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

Figure11-6 Format of A/D converter mode register 2 (ADM2) (1/3)

Reset value: 00H
R/W

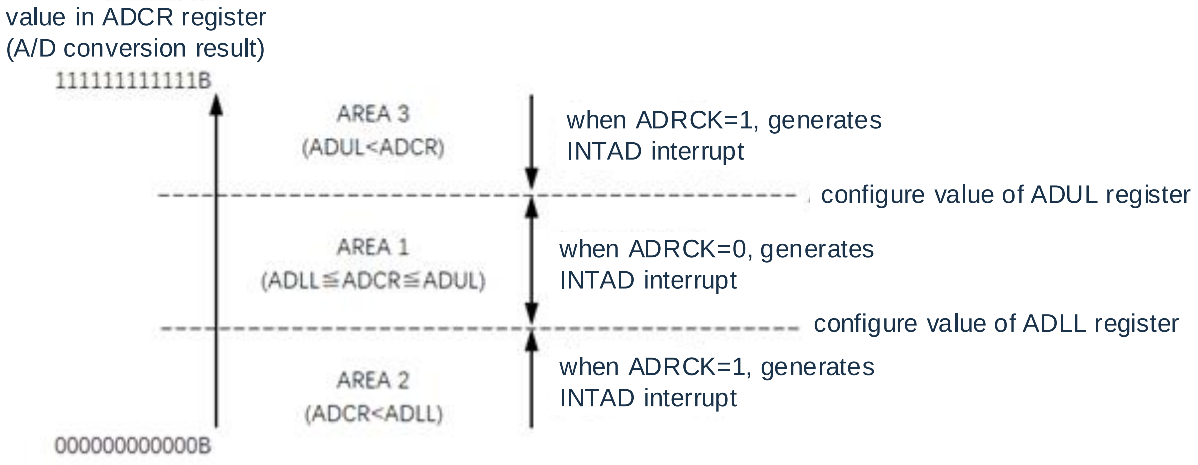
	7	6	5	4	3	2	1	0
ADM2	ADREFP1	HOMEP0	0	0	ADRCK	0	CHRDE	0

ADREFP1	ADREFP0	Selection of positive (+) voltage references for A/D converters
0	0	Provided by V_{DD}.
1	0	Provided by internal reference voltage (1.45V).
other		Set Prohibited

ADRCK	Check the upper and lower values of the conversion result
0	When the ADLL register ≤ the ADCR register ≤ ADUL register (AREA1), an interrupt signal (INTAD) is generated.
1	When ADCRRegister <ADLLRegisters (AREA2OrADULTSRegister <ADCRRegisters (AREA3), an interrupt signal is generated(INTAD).
The range of interrupt signal (INTAD) generated from AREA1 to AREA3 is shown in Figure 15-8.	

CHRDE	The output of the channel identification is enabled when the A/D converter scans mode
0	When scanning mode, the channel number is not identified in the conversion results
1	When scanning mode, the high four bits of the converted result ([15:12] of the ADCR register) are the channel numbers for this result

Figure11-7 Range of interrupt signal generation for the ADRCK bit



Note 1 To override the ADM2 register, it must be done in the transition stop state (ADCS=0).

Note When INTAD does not occur, the A/D conversion results are not saved to the ADCR register and the ADCRH register.

11.2.5 A/D converter trigger mode register (ADTRG)

This is the register that sets the A/D conversion trigger mode and the hardware trigger signal.

The AD TRG register is set via an 8-bit memory operation command.

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

Figure11-8 Format of A/D converter trigger mode register (ADTRG)

Reset value: 00H
R/W

	7	6	5	4	3	2	1	0
ADTRG	ADTMD1	ADTMD0	0	0	0	0	ADTRS1	ADTRS0

ADTMD1	ADTMD0	Selection of A/D conversion trigger modes
0	0	Software triggering mode
0	1	
1	0	Hardware triggers no-wait mode
1	1	The hardware triggers the wait mode

ADTRS1	ADTRS0	Selection of hardware trigger signals
0	0	The counting end of timer channel 1 or the capture of the end interrupt signal (INTTM01).
0	1	The event signal selected by ELC
1	0	Real-time clock interrupt signal (INTRTC).
1	1	Interval timer interrupt signal (INTIT).

Note 1 To override the ADTRG register, it must be done in the transition stop state (ADCS=0, ADCE=0).

2. In order to end the A/D conversion normally, the hardware trigger interval must be set at least to the following time:
When hardware triggers no-wait mode: 2 f_{CLK} clocks + A/D conversion time
When the hardware triggers the wait mode: 2 f_{CLK} clock + A/D power supply stable wait time + A/D transition time

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

11.2.6 Analog input channel specification register (ADS)

This is the register that specifies the analog voltage input channel to be A/D converted. The ADS registers are set with 8-bit memory operating instructions.

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

Figure11-9 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

○ Selection mode (ADM1. ADMD=0)

ADS register setting value		Analog input channel
ADISS	ADS[5:0]	
0	6'h00	ANI0(P20)
0	6'h01	ANI1(P21)
0	6'h02	ANI2(P22)
0	6'h03	ANI3(P23)
0	6'h04	ANI4(P24)
0	6'h05	ANI5(P25)
0	6'h06	ANI6(P26)
0	6'h07	ANI7(P27)
0	6'h08	ANI8(P11)
0	6'h09	ANI9(P10)
0	6'h0a	ANI10(P01)
0	6'h0b	ANI11(P00)
0	6'h0c	ANI12(P147)
0	6'h0d	ANI13(P12)
0	6'h0e	ANI14(P120)
0	6'h0f	ANI15(P146)
0	6'h10	ANI16(P13)
0	6'h11	ANI17(P14)
0	6'h12	ANI18(P15)
0	6'h13	ANI19(P16)
0	6'h14	ANI20(P17)
0	6'h15	ANI21(P30)
0	6'h16	ANI22(P31)
0	6'h17	ANI23(P50)
0	6'h18	ANI24(P51)
0	6'h19	ANI25(P60)
0	6'h1a	ANI26(P61)
0	6'h1b	ANI27(P62)
0	6'h1c	ANI28(P63)
0	6'h1d	ANI29(P70)
0	6'h1e	ANI30(P71)
0	6'h1f	ANI31(P72)
0	6'h20	ANI32(P73)
0	6'h21	ANI33(P74)
0	6'h22	ANI34(P75)
0	6'h23	ANI35(P130)
0	6'h24	ANI36(P136)
0	6'h3f	SW ALL OFF
1	6'h00	BGR (temperature sensor0)
1	6'h01	BGR (1.45V)
Beyond the above		Disable settings.

Note 1 The analog input channels of A/D converters vary from product to product. Detailed channel assignment information can be found in the data sheet.

○ Scan mode (ADM1. ADMD=1)

ADISS	ADS3	ADS2	ADS1	ADS0	Analog input channels			
					Scan 0	Scan 1	Scan 2	Scan 3
0	0	0	0	0	ANI0	ANI1	ANI2	ANI3
0	0	0	0	1	ANI1	ANI2	ANI3	ANI4
0	0	0	1	0	ANI2	ANI3	ANI4	ANI5
0	0	0	1	1	ANI3	ANI4	ANI5	ANI6
0	0	1	0	0	ANI4	ANI5	ANI6	ANI7
0	0	1	0	1	ANI5	ANI6	ANI7	ANI8
0	0	1	1	0	ANI6	ANI7	ANI8	ANI9
0	0	1	1	1	ANI7	ANI8	ANI9	ANI10
0	1	0	0	0	ANI8	ANI9	ANI10	ANI11
0	1	0	0	1	ANI9	ANI10	ANI11	ANI12
0	1	0	1	0	ANI10	ANI11	ANI12	ANI13
0	1	0	1	1	ANI11	ANI12	ANI13	ANI14
0	1	1	0	0	ANI12	ANI13	ANI14	ANI15
Beyond the above					Disable settings.			

Note 1. When scanning mode, bit4, bit5 and bit6 must be set to “0”.

2. For ports that are set as analog inputs by the PMCx register, A/D conversion can only be specified as analog inputs by ADS.

3. Pins set to digital input/output by the port-mode control register (PMC xx) cannot be set through the ADS register.

4. To override the ADISS bit, it must be done in the transition stop state (ADCS=0, ADCE=0).

5. After setting the ADISS bit to “1”, the result of the first conversion cannot be used.

6. The ADISS bit cannot be set to “1” when shifting to deep sleep mode or when shifting to sleep mode while the CPU is running on the subsystem clock.

11.2.7 12-bit A/D conversion result register (ADCR)

This is a 16-bit register that holds the results of the A/D conversion, and this register is readable only. Whenever the A/D conversion ends, the conversion result notes are loaded from the successive approximation register (SAR).

The high 4 bits of this register are fixed to “0” when mode is selected, and ADM2.CHRDE=1 can be configured as the channel number of this conversion result when scanning mode.

The ADCR register is read through 16-bit memory operation instructions.

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

Note If the value of the A/D conversion result is not within the set value of the A/D conversion result comparison function (set by ADRCK bits and ADUL/ADLL registers (refer to Figure 11-7)), it is not saved A/D conversion result.

Figure11-10 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 upper 8 bits of the conversion result can be read through the ADCRH register.

2. When 16-bit access to the ADCR register, the high 12 bits of the conversion result can be read sequentially from bit11.

○ Selection mode (ADM1. ADMD=0)

The readout value of AD CH0~3 is fixed at 4'b0000

○ Scan mode (ADM1. ADMD=1) and ADM2 CHRDE=1, ADCH0~3 readout value and conversion channel relationship are as follows:

ADCH3	ADCH2	ADCH1	ADCH0	Transform the channel identity
0	0	0	0	ANI0
0	0	0	1	ANI1
0	0	1	0	ANI2
0	0	1	1	ANI3
0	1	0	0	ANI4
0	1	0	1	ANI5
0	1	1	0	ANI6
0	1	1	1	ANI7
1	0	0	0	ANI8
1	0	0	1	ANI9
1	0	1	0	ANI10
1	0	1	1	ANI11
1	1	0	0	ANI12
1	1	0	1	ANI13
1	1	1	0	ANI14
1	1	1	1	ANI15

11.2.8 8-bit A/D conversion result register (ADCRI)

This is an 8-bit register that holds the results of the A/D conversion, holding a high 8-bit note with 12-bit resolution.

The ADCRH register is read through an 8-bit memory operation instruction.

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

Note If the value of the A/D conversion result is not in the A/D conversion result comparison function (set via the ADRCK bit and the ADUL/ADLL register (see Figure 11-8)) within the set value range, the A/D conversion results are not saved.

Figure11-11 Format of 8-bit A/D Conversion Result Register (ADCRH)

Reset value: 00H R

	7	6	5	4	3	2	1	0
ADCRH								

Note The conversion results must be read after the conversion is complete and before configuring the ADM0 and ADS registers. Otherwise, you may not read the correct conversion results.

11.2.9 Conversion result comparison upper limit setting register (ADUL)

This is the set register used to check the upper limit of the A/D conversion result.

The A/D conversion result is compared to the value of the ADUL register, and the ADRCK in the mode register 2 (ADM2) of the A/D converter

The set range of bits (see Figure11-7 Range of interrupt signal generation for the ADRCK bit) controls the generation of the interrupt signal (INTAD). The ADUL register is set via 8-bit memory operation instructions.

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

Note 1 Only the 12-bit A/D is converted to the high 8-bit and ADUL registers of the result register (ADCR) and the ADLL Registers are compared.

2. To override the ADUL register and the ADLL register, it must be done in the transition stop state (ADCS=0).

3. When setting the ADUL register and the ADLL register, the ADUL must be > ADLL.

Figure11-12 Format of conversion result comparison upper limit setting register (ADUL)

Reset value: FFH R/W

	7	6	5	4	3	2	1	0
ADULTS	ADUL7	AOFL6	ADUL5	ADUL4	ADUL3	ADUL2	ADUL1	ADUL0

11.2.10 Conversion result comparison lower limit setting register (ADLL)

This is the set register used to check the lower limit of the A/D conversion result.

The A/D conversion result is compared to the value of the ADLL register, and the ADRCK in the mode register 2 (ADM2) of the A/D converter

The set range of bits (see FigureFigure11-7 Range of interrupt signal generation for the ADRCK bit) controls the generation of the interrupt signal (INTAD). The ADLL register is set via an 8-bit memory operation command.

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

Figure11-13 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 12-bit A/D is converted to the high 8-bit and ADUL registers of the result register (ADCR) and the ADLL Registers are compared.

2. To override the ADUL register and the ADLL register, it must be done in the transition stop state (ADCS=0).

3. When setting the ADUL register and the ADLL register, the ADUL must be > ADLL.

11.2.11 A/D converter sampling time extension control register (ADSMPWAIT)

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

The ADSMPWAIT register is set via an 8-bit memory operation command.

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

Figure11-14 Format of A/D converter sampling time extension control register (ADSMPWAIT)

Reset value: 00H
R/W

	7	6	5	4	3	2	1	0
ADSMPWAIT								ADSMPWAIT

ADSMPWAIT	A/D conversion objects
0	When “0”, the A/D sampling time is 4 ADCLKs
1	When “1”, the A/D sampling time is 8 ADCLK

Note: Set ADSMPWAIT in the transition stop state (ADCS=0).

11.2.12 Registers for controlling the function of the analog input pin port

When using the ANIx pin as the analog input to an A/D converter, the port must be configured as an analog channel by setting the corresponding Port Mode Control Register (PMCxx) bit to “1”. For details, please refer to “Chapter 2 Pin Functions”.

11.3 Input voltage and conversion results

The analog input voltage at the analog input pin (ANIx) and the theoretical A/D conversion result (12-bit A/D Conversion Result Register (ADCR)) are related by the following expressions.

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

or

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

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

V_{AN} : Analog input voltage

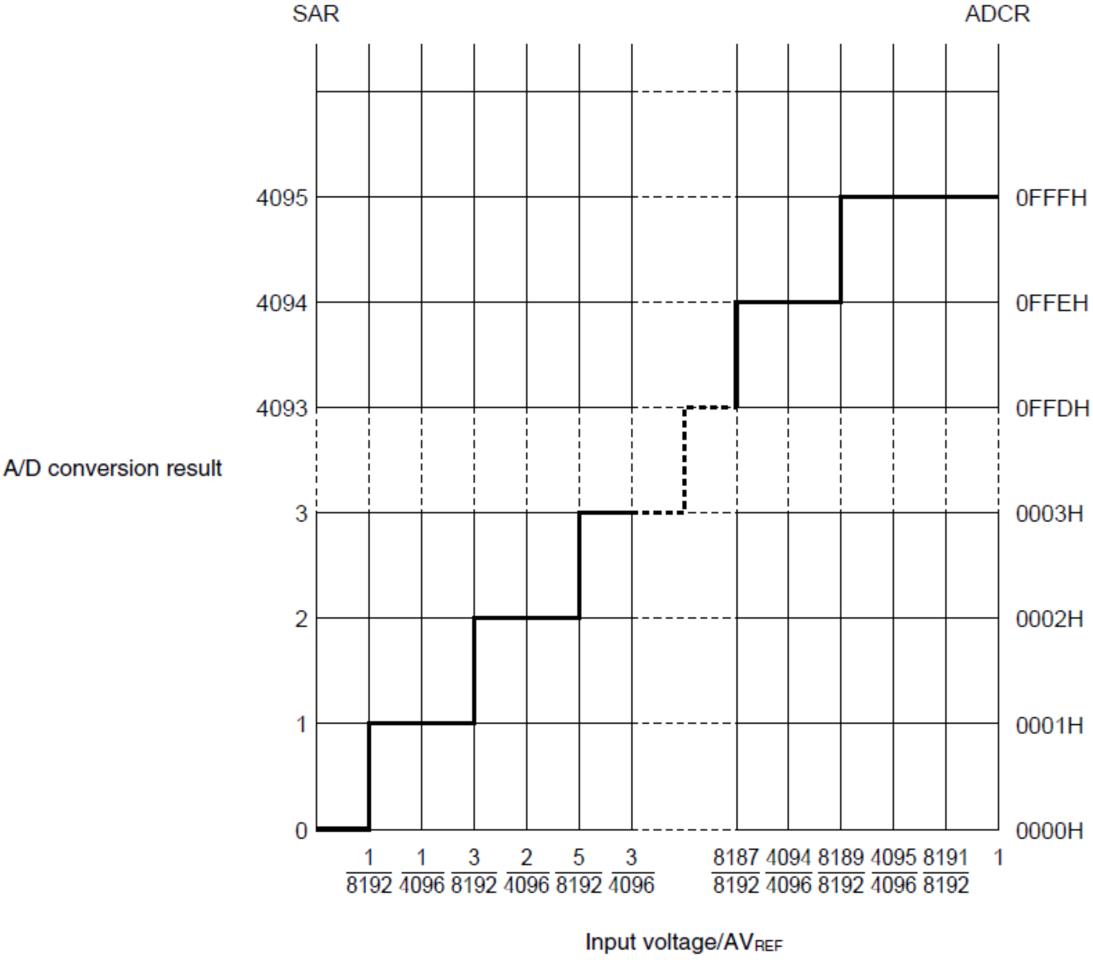
AV REF : AV_{REF} pin voltage

ADCR : The value of the A/D conversion result register (ADCR).

SAR : Successive approximation registers

The relationship between the analog input voltage and the A/D conversion results is shown in the following figure.

Figure11-15 Analog input voltage vs. A/D conversion results



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

11.4 Operation mode of A/D converter

The operation of each mode of the A/D converter is as follows. For the setting steps for each mode, refer to “Setup Flow Diagram of 11.5 A/D Converter“.

11.4.1 Software trigger mode (select mode, continuous conversion mode)

(1) In the stop state, the ADCE bit of the mode register 0 (ADM0) of the A/D converter is “1” and enters the A/D transition standby state.

(2) After counting the stable wait time (1 us) by software, the ADCS bit of the ADM0 register is “1” for the register specified by the analog input channel (ADS) specifies the analog input for A/D conversion.

(3) If the A/D conversion ends, the conversion results are saved to the A/D conversion result register (ADCR, ADCRH) and an A/D conversion end interrupt request signal (INTAD) is generated). Start the next A/D conversion immediately after the A/D conversion ends.

(4) If you override the “1” to the ADCS bit during the conversion, the current A/D conversion is aborted immediately and the conversion begins again.

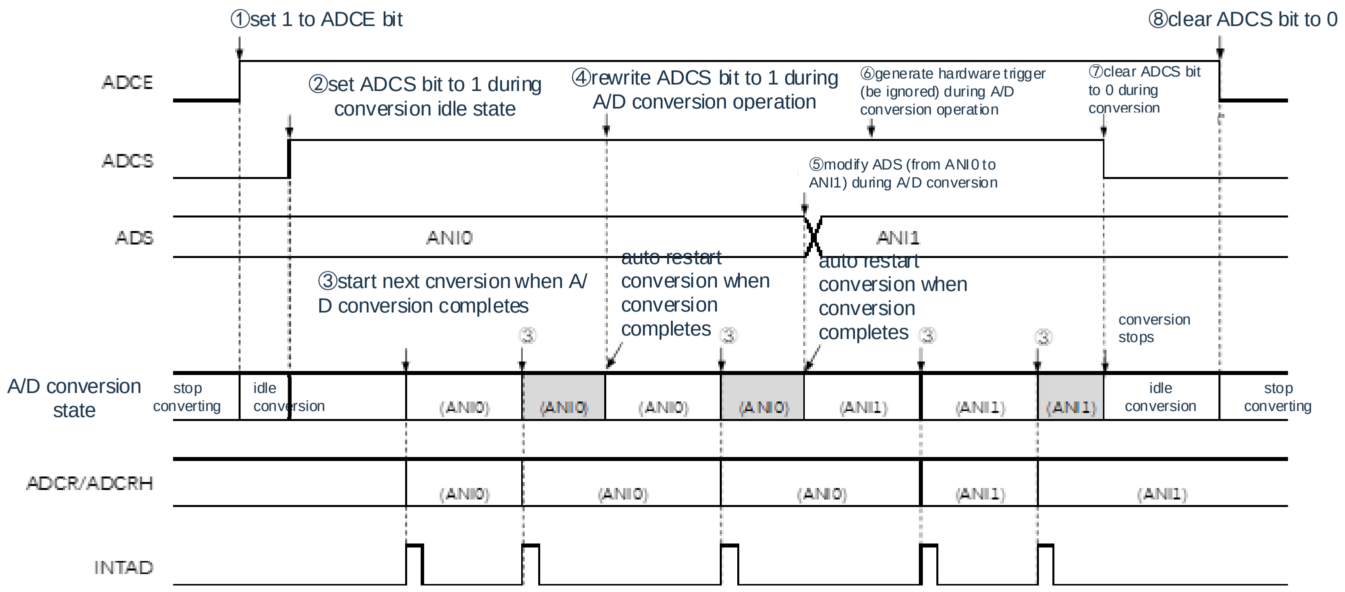
(5) If the ADS registers are overwritten or rewritten during the conversion, the current A/D conversion is aborted immediately, and then the analog inputs respecified by the ADS registers are A/D.

(6) The A/D conversion does not start even if the input hardware triggers during the conversion.

(7) If the ADCS bit is “0” during the conversion, the current A/D conversion is aborted immediately and then enters the A/D transition standby.

(8) If the ADCE bit is “0” in the A/D transition standby state, the A/D converter enters a stopped state. When the ADCE bit is “0”, even the ADCS set to “1” is ignored and the A/D conversion is not started.

Figure11-16 Timing example of software trigger mode (select mode, continuous conversion mode)


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
auto restart conversion when conversion completes
③
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

11.4.2 Software trigger mode (select mode, single conversion mode)

(1) In the stop state, the ADCE bit of the mode register 0 (ADM0) of the A/D converter is “1” and enters the A/D transition standby state.

(2) After counting the stable wait time (1 us) by software, the ADCS bit of the ADM0 register is “1” for the register specified by the analog input channel (ADS) specifies the analog input for A/D conversion.

(3) If the A/D conversion ends, the conversion results are saved to the A/D conversion result register (ADCR, ADCRH) and an A/D conversion end interrupt request signal (INTAD) is generated).

(4) After the A/D conversion is completed, the ADCS bit automatically clears “0” and enters the A/D transition standby state.

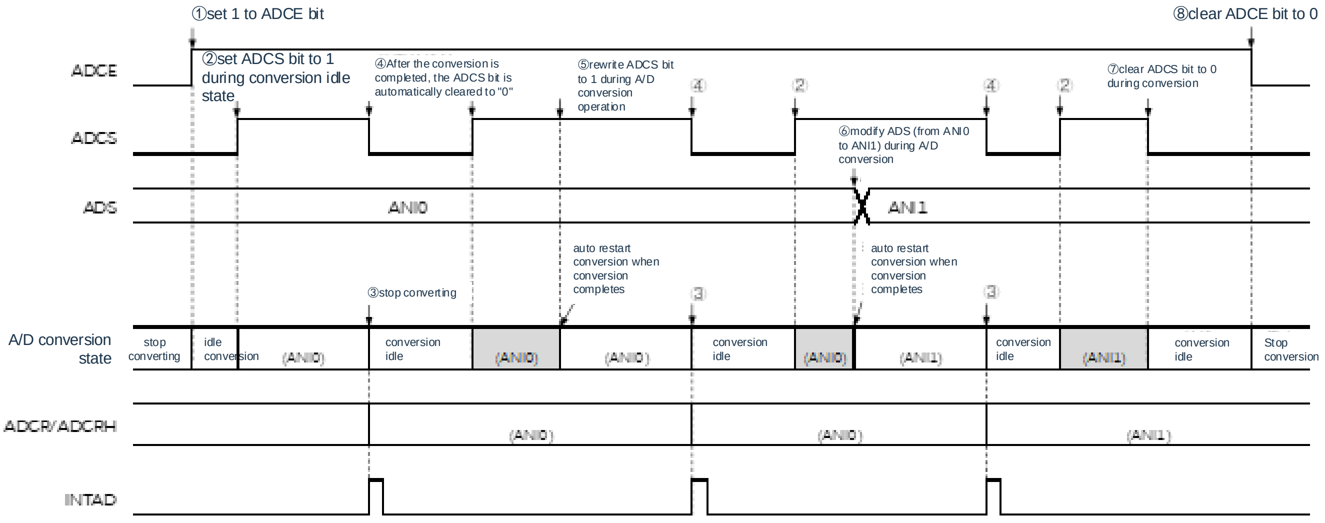
(5) If you override the “1” to the ADCS bit during the conversion, the current A/D conversion is aborted immediately and the conversion begins again.

(6) If the ADS registers are overwritten or rewritten during the conversion, the current A/D conversion is aborted immediately, and then the analog inputs respecified by the ADS registers are A/D.

(7) If the ADCS bit is “0” during the conversion, the current A/D conversion is aborted immediately and then enters the A/D transition standby.

(8) If the ADCE bit is “0” in the A/D transition standby state, the A/D converter enters a stopped state. When the ADCE bit is “0”, even the ADCS set to “1” is ignored and the A/D conversion is not started. The A/D transition does not start even when the input hardware triggers in the A/D transition standby state.

Figure11-17 Timing example of software trigger mode (select mode, single conversion mode)


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
auto restart conversion when conversion completes
③stop converting
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

11.4.3 Software trigger mode (scan mode, continuous conversion mode)

(1) In the stop state, the ADCE bit of the mode register 0 (ADM0) of the A/D converter is "1" and enters the A/D transition standby state.

(2) After counting the stable wait time (1 us) by software, the ADCS bit of the ADM0 register is "1" for the register specified by the analog input channel (ADS The four analog input channels specified from scan 0 to scan 3 are converted to A/D. A/D conversion is performed sequentially from the analog input channels specified by Scan 0.

(3) A/D conversion of 4 analog input channels in succession. Whenever the A/D conversion ends, the conversion results are saved to the A/D conversion result register (ADCR, ADCRH) and an A/D conversion end interrupt request signal is generated (INTAD). Immediately after the A/D conversion of the 4 channels is completed, the next A/D conversion (4 channels) is automatically started from the set channel.

(4) If you override the "1" to the ADCS bit during the conversion, the current A/D conversion is aborted immediately and the conversion begins again.

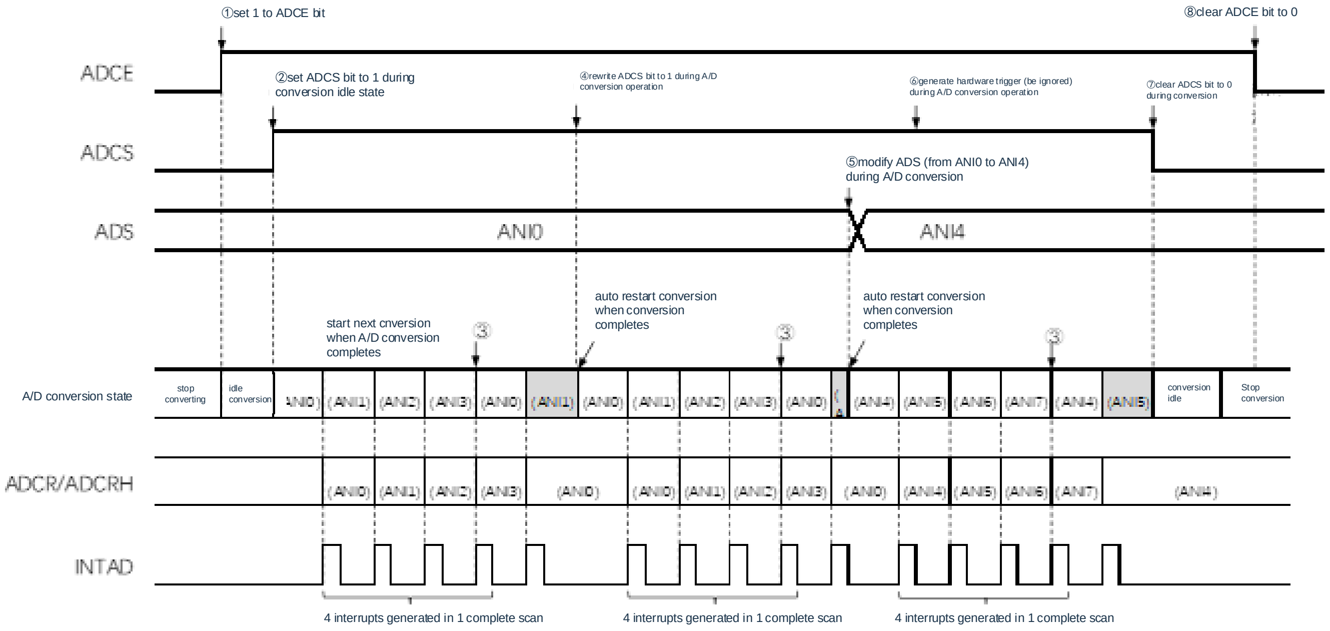
(5) If the ADS registers are overwritten or overwritten during the conversion, the current A/D conversion is aborted immediately and then A/D converted from the original channel respecified by the ADS registers.

(6) The A/D conversion does not start even if the input hardware triggers during the conversion.

(7) If the ADCS bit is "0" during the conversion, the current A/D conversion is aborted immediately and then enters the A/D transition standby.

(8) If the ADCE bit is "0" in the A/D transition standby state, the A/D converter enters a stopped state. When the ADCE bit is "0", even the ADCS set to "1" is ignored and the A/D conversion is not started.

Figure11-18 Timing example of software trigger mode (scan mode, continuous conversion mode)


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
auto restart conversion when conversion completes
auto restart conversion when conversion completes
③
③
③
A/D conversion state
stop converting
idle conversion
(ANI0)
(ANI1)
(ANI2)
(ANI3)
(ANI4)
(ANI5)
(ANI6)
(ANI7)
conversion idle
Stop conversion
ADCR/ADCRH
INTAD
4 interrupts generated in 1 complete scan
4 interrupts generated in 1 complete scan
4 interrupts generated in 1 complete scan

11.4.4 Software trigger mode (scan mode, single conversion mode)

(1) In the stop state, the ADCE bit of the mode register 0 (ADM0) of the A/D converter is “1” and enters the A/D transition standby state.

(2) After counting the stable wait time (1 us) by software, the ADCS bit of the ADM0 register is “1” for the register specified by the analog input channel (ADS The four analog input channels specified from scan 0 to scan 3 are converted to A/D. A/D conversion is performed sequentially from the analog input channels specified by Scan 0.

(3) A/D conversion of 4 analog input channels in succession. Whenever the A/D conversion ends, the conversion results are saved to the A/D conversion result register (ADCR, ADCRH) and an A/D conversion end interrupt request signal is generated (INTAD).

(4) After the A/D conversion of the four channels is completed, the ADCS bit automatically clears “0” and enters the A/D transition standby state.

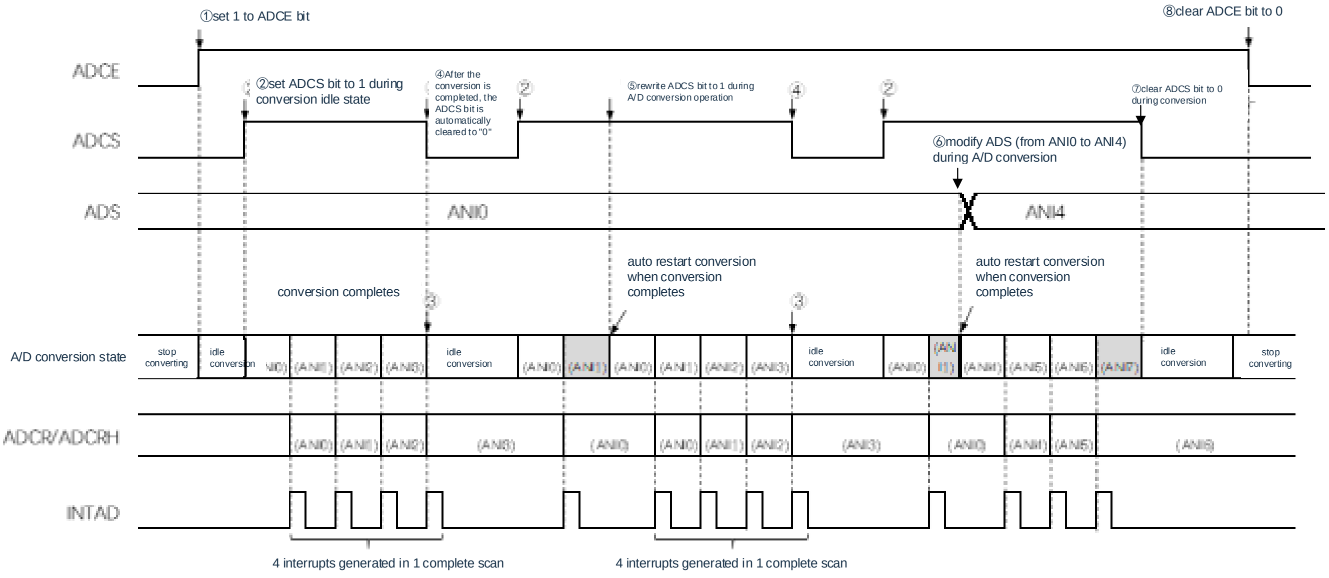
(5) If you override the “1” to the ADCS bit during the conversion, the current A/D conversion is aborted immediately and the conversion begins again.

(6) If the ADS registers are overwritten or overwritten during the conversion, the current A/D conversion is aborted immediately and then A/D converted from the original channel respecified by the ADS registers.

(7) If the ADCS bit is “0” during the conversion, the current A/D conversion is aborted immediately and then enters the A/D transition standby.

(8) If the ADCE bit is “0” in the A/D transition standby state, the A/D converter enters a stopped state. When the ADCE bit is “0”, even the ADCS set to “1” is ignored and the A/D conversion is not started. The A/D transition does not start even when the input hardware triggers in the A/D transition standby state.

Figure11-19 Timing example of software trigger mode (scan mode, single conversion mode).


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"
②
⑤rewrite ADCS bit to 1 during A/D conversion operation
④
②
⑦clear ADCS bit to 0 during conversion
ADCS
⑥modify ADS (from ANI0 to ANI4) during A/D conversion
ADS
ANI0
ANI4
conversion completes
③
auto restart conversion when conversion completes
③
auto restart conversion when conversion completes
A/D conversion state
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
ADCR/ADCRH
(ANI0)
(ANI1)
(ANI2)
(ANI3)
(ANI0)
(ANI0)
(ANI1)
(ANI2)
(ANI3)
(ANI0)
(ANI4)
(ANI5)
(ANI6)
INTAD
4 interrupts generated in 1 complete scan
4 interrupts generated in 1 complete scan

11.4.5 Hardware triggered no-wait mode (select mode, continuous conversion mode)

(1) In the stop state, the ADCE bit of the mode register 0 (ADM0) of the A/D converter is “1” and enters the A/D transition standby state.

(2) After counting the steady wait time (1 us) by software, the ADCS bit of the ADM0 register is “1” into a hardware-triggered standby state (this phase does not begin the transition). When the hardware triggers standby, the A/D transition does not start even when the ADCS bit “1” is applied.

(3) If the input hardware triggers in the state where the ADCS bit is “1”, the analog input specified by the analog input channel specified register (ADS) is A/D converted.

(4) If the A/D conversion ends, the conversion results are saved to the A/D conversion result register (ADCR, ADCRH) and an A/D conversion end interrupt request signal (INTAD) is generated). Start the next A/D conversion immediately after the A/D conversion ends.

(5) If the input hardware triggers during the conversion, the current A/D conversion is aborted immediately and then restarted.

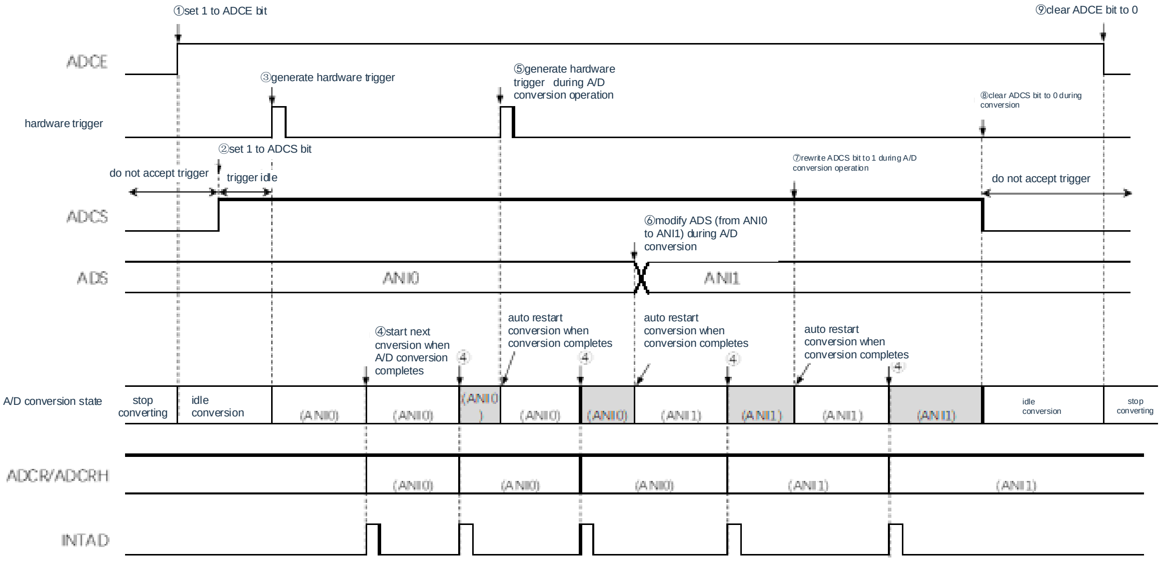
(6) If the ADS registers are overwritten or overwritten during the conversion, the current A/D conversion is aborted immediately, and then the analog input respecified by the ADS register is A/D converted.

(7) If you override the “1” to the ADCS bit during the conversion, the current A/D conversion is aborted immediately and the conversion begins again.

(8) If the ADCS bit is “0” during the conversion, the current A/D conversion is aborted immediately and then enters the A/D transition standby. However, in this state, the A/D converter does not enter the stopped state.

(9) If the ADCE bit is “0” in the A/D transition standby state, the A/D converter enters a stopped state. When the ADCS bit is “0”, even the input hardware trigger is ignored and the A/D conversion does not begin.

Figure11-20 Timing example of hardware trigger no-wait mode (select mode, continuous conversion mode)


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
⑦rewrite ADCS bit to 1 during A/D conversion operation
do not accept trigger
trigger idle
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)
(ANI0)
(ANI0)
(ANI0)
(ANI0)
(ANI1)
(ANI1)
(ANI1)
(ANI1)
idle conversion
stop converting
ADCR/ADCRH
(ANI0)
(ANI0)
(ANI0)
(ANI1)
(ANI1)
INTAD

11.4.6 Hardware trigger no-wait mode (select mode, single conversion mode)

(1) In the stop state, the ADCE bit of the mode register 0 (ADM0) of the A/D converter is “1” and enters the A/D transition standby state.

(2) After counting the steady wait time (1 us) by software, the ADCS bit of the ADM0 register is “1” into a hardware-triggered standby state (this phase does not begin the transition). When the hardware triggers standby, the A/D transition does not start even when the ADCS bit “1” is applied.

(3) If the input hardware triggers in the state where the ADCS bit is “1”, the analog input specified by the analog input channel specified register (ADS) is A/D converted.

(4) If the A/D conversion ends, the conversion results are saved to the A/D conversion result register (ADCR, ADCRH) and an A/D conversion end interrupt request signal (INTAD) is generated).

(5) After the A/D conversion ends, the ADCS bit remains in the state of “1” and enters the A/D transition standby state.

(6) If the input hardware triggers during the conversion, the current A/D conversion is aborted immediately and then restarted.

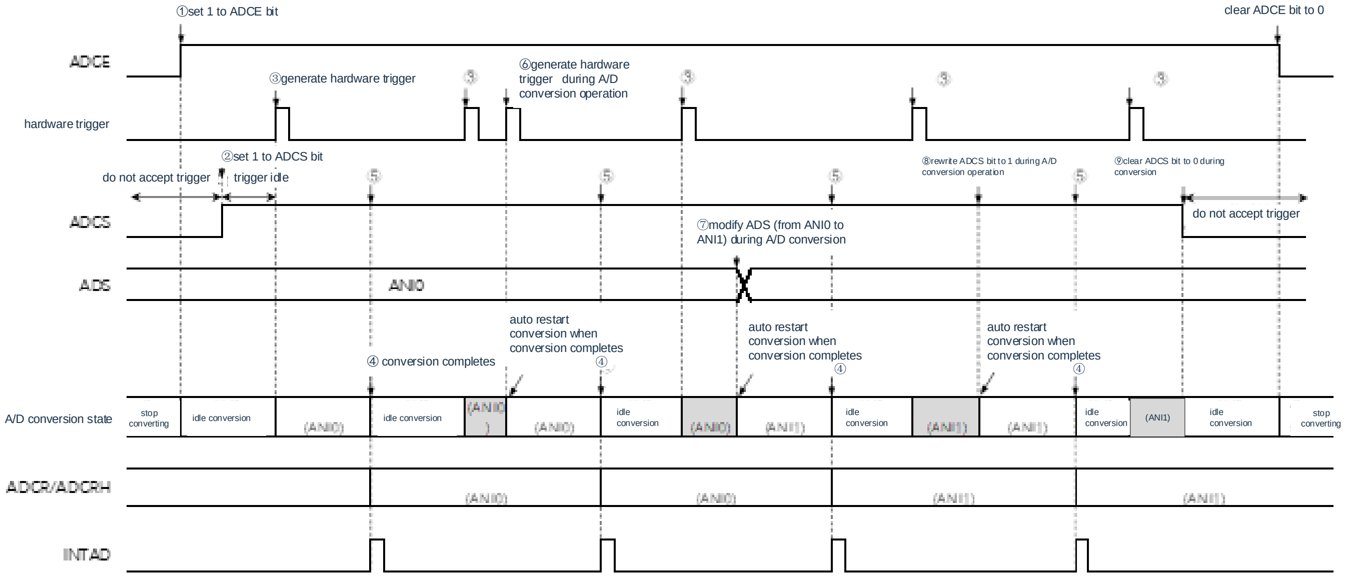
(7) If the ADS registers are overwritten or overwritten during the conversion, the current A/D conversion is aborted immediately, and then the analog input respecified by the ADS register is A/D converted.

(8) If you override the “1” to the ADCS bit during the conversion, the current A/D conversion is aborted immediately and the conversion begins again.

(9) If the ADCS bit is “0” during the conversion process, the current A/D conversion stops immediately and then enters the A/D transition standby state. However, in this state, the A/D converter does not enter the stopped state.

(10) If the ADCE bit is “0” in the A/D transition standby state, the A/D converter enters a stopped state. When the ADCS bit is “0”, even the input hardware trigger is ignored and the A/D conversion does not begin.

Figure11-21 Timing example of hardware trigger no-wait mode (select mode, sequential conversion mode)



11.4.7 Hardware trigger no-wait mode (scan mode, continuous conversion mode)

(1) In the stop state, the ADCE bit of the mode register 0 (ADM0) of the A/D converter is “1” and enters the A/D transition standby state.

(2) After counting the steady wait time (1 us) by software, the ADCS bit of the ADM0 register is “1” into a hardware-triggered standby state (this phase does not begin the transition). When the hardware triggers standby, the A/D transition does not start even when the ADCS bit “1” is applied.

(3) If the input hardware triggers in the state where the ADCS bit is “1”, scan 0 to 4 of 3 is specified by the analog input channel specified register (ADS). analog input channels for A/D conversion. A/D conversion is performed sequentially from the analog input channels specified by Scan 0.

(4) A/D conversion of 4 analog input channels in succession. Whenever the A/D conversion ends, the conversion results are saved to the A/D conversion result register (ADCR, ADCRH) and an A/D conversion end interrupt request signal is generated (INTAD). Immediately after the A/D conversion of the 4 channels is completed, the next A/D conversion is automatically started from the set channel.

(5) If the input hardware triggers during the conversion, the current A/D conversion is aborted immediately and then restarted from the original channel.

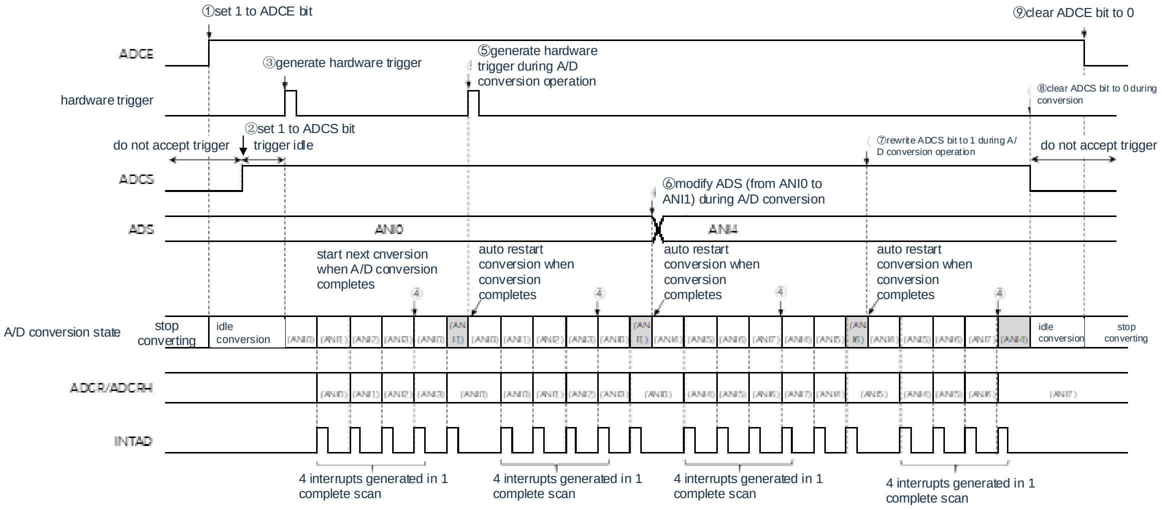
(6) If the ADS registers are overwritten or overwritten during the conversion, the current A/D conversion is aborted immediately and then A/D converted from the channel respecified by the ADS registers.

(7) If you override the ADCS bit “1” during the conversion process, the current A/D conversion is aborted immediately and the conversion is restarted from the original channel.

(8) If the ADCS bit is “0” during the conversion, the current A/D conversion is aborted immediately and then enters the A/D transition standby. However, in this state, the A/D converter does not enter the stopped state.

(9) If the ADCE bit is “0” in the A/D transition standby state, the A/D converter enters a stopped state. When the ADCE bit is “0”, even the ADCS set to “1” is ignored and the A/D conversion is not started.

Figure11-22 Timing example of hardware trigger no-wait mode (scan mode, continuous conversion mode)


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

11.4.8 Hardware trigger no-wait mode (scan mode, single conversion mode)

(1) In the stop state, the ADCE bit of the mode register 0 (ADM0) of the A/D converter is “1” and enters the A/D transition standby state.

(2) After counting the steady wait time (1us) by software, the ADCS bit of the ADM0 register is “1” and enters the hardware-triggered standby state (this phase does not start the transition). When the hardware triggers standby, the A/D transition does not start even when the ADCS bit “1” is applied.

(3) If the input hardware triggers in the state where the ADCS bit is “1”, scan 0 to 4 of 3 is specified by the analog input channel specified register (ADS). analog input channels for A/D conversion. A/D conversion is performed sequentially from the analog input channels specified by Scan 0.

(4) A/D conversion of 4 analog input channels in succession. Whenever the A/D conversion ends, the conversion results are saved to the A/D conversion result register (ADCR, ADCRH) and an A/D conversion end interrupt request signal is generated (INTAD).

(5) After the A/D conversion of the four channels is completed, the ADCS bit remains in the state of “1” and enters the A/D transition standby state.

(6) If the input hardware triggers during the conversion, the current A/D conversion is aborted immediately and then restarted from the original channel.

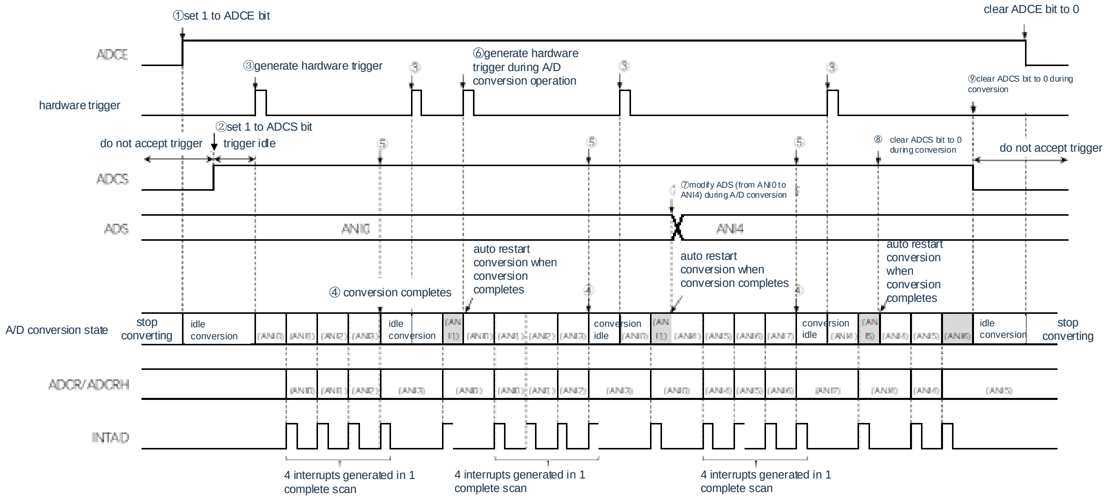
(7) If the ADS registers are overwritten or overwritten during the conversion, the current A/D conversion is aborted immediately and then A/D converted from the original channel respecified by the ADS registers.

(8) If you rewrite the ADCS bit “1” during the conversion process, the current A/D conversion is aborted immediately and then the conversion starts again from the original channel.

(9) If the ADCS bit is “0” during the conversion, the current A/D conversion is aborted immediately and then enters the A/D transition standby. However, in this state, the A/D converter does not enter the stopped state.

(10) If the ADCE bit is “0” in the A/D transition standby state, the A/D converter enters a stopped state. When the ADCS bit is “0”, even the input hardware trigger is ignored and the A/D conversion does not begin.

Figure11-23 Timing example of hardware trigger no-wait mode (scan mode, single conversion mode)


①set 1 to ADCE bit
clear ADCE bit to 0
ADCE
③generate hardware trigger
⑥generate hardware trigger during A/D conversion operation
③
⑨clear ADCS bit to 0 during conversion
hardware trigger
②set 1 to ADCS bit
do not accept trigger
trigger idle
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
④ conversion completes
auto restart conversion when conversion completes
auto restart conversion when conversion completes
A/D conversion state
stop converting
idle conversion
(ANI0)
(ANI1)
(ANI2)
(ANI3)
idle conversion
conversion idle
(ANI4)
(ANI5)
(ANI6)
(ANI7)
conversion idle
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

11.4.9 Hardware trigger wait mode (select mode, continuous conversion mode)

(1) In the stop state, the ADCE bit of the mode register 0 (ADM0) of the A/D converter is “1” into a hardware-triggered standby state.

(2) If hardware triggers are entered in hardware-triggered standby, the analog inputs specified by the analog input channel specified registers (ADS) are A/D converted. The ADCS bit of the ADM0 register is automatically “1” while the input hardware triggers.

(3) If the A/D conversion ends, the conversion results are saved to the A/D conversion result register (ADCR, ADCRH) and an A/D conversion end interrupt request signal (INTAD) is generated). Start the next A/D conversion immediately after the A/D conversion is over (at this point, no hardware triggering is required).

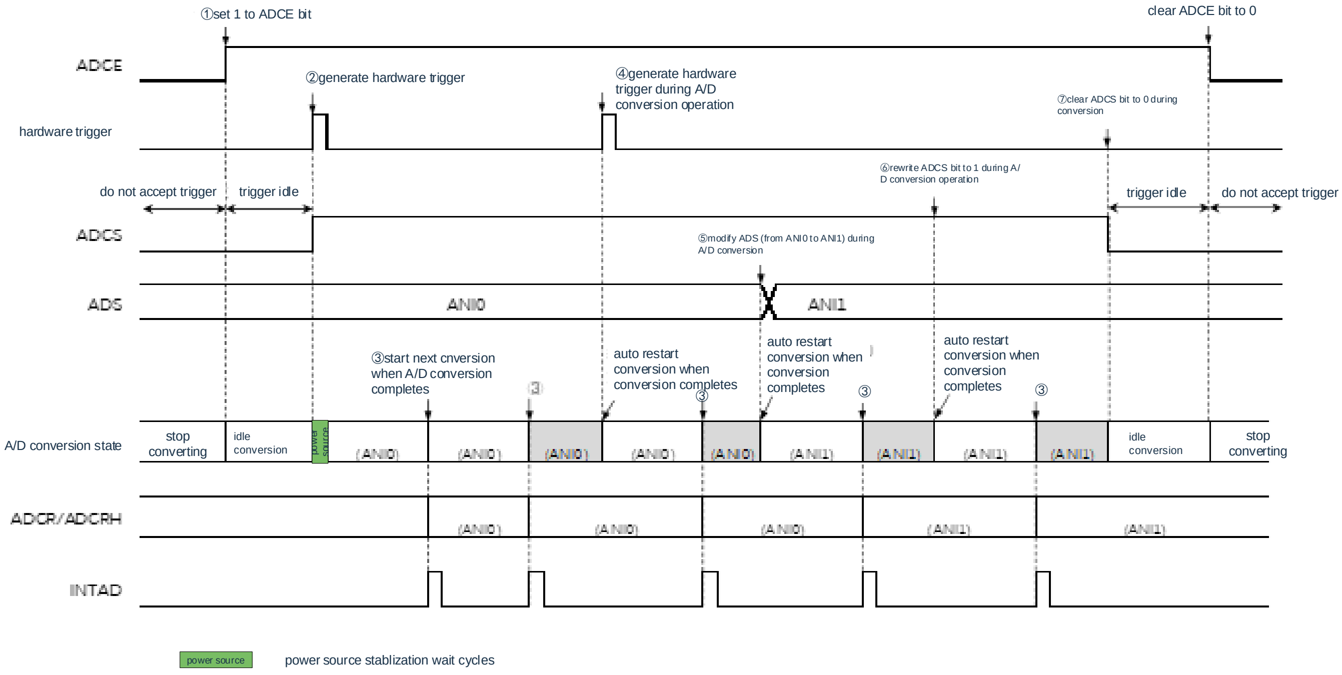
(4) If the input hardware triggers during the conversion, the current A/D conversion is aborted immediately and then restarted.

(5) If the ADS registers are overwritten or overwritten during the conversion, the current A/D conversion is aborted immediately, and then the analog input respecified by the ADS register is A/D converted.

(6) If you override the “1” to the ADCS bit during the conversion, the current A/D conversion is aborted immediately and the conversion begins again.

(7) If the ADCS bit is “0” during the transition, the current A/D transition is aborted immediately, then enters a hardware-triggered standby state, and the A/D converter enters a stopped state. When the ADCE bit is “0”, even the input hardware trigger is ignored and the A/D conversion does not begin.

Figure11-24 Timing example of hardware trigger wait mode (select mode, continuous conversion mode)


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

11.4.10 Hardware trigger wait mode (select mode, single conversion mode)

(1) In the stop state, the ADCE bit of the mode register 0 (ADM0) of the A/D converter is “1” into a hardware-triggered standby state.

(2) If hardware triggers are entered in hardware-triggered standby, the analog inputs specified by the analog input channel specified registers (ADS) are A/D converted. The ADCS bit of the ADM0 register is automatically “1” while the input hardware triggers.

(3) If the A/D conversion ends, the conversion results are saved to the A/D conversion result register (ADCR, ADCRH) and an A/D conversion end interrupt request signal (INTAD) is generated).

(4) After the A/D conversion is complete, the ADCS bit automatically clears “0” and the A/D converter enters a stopped state.

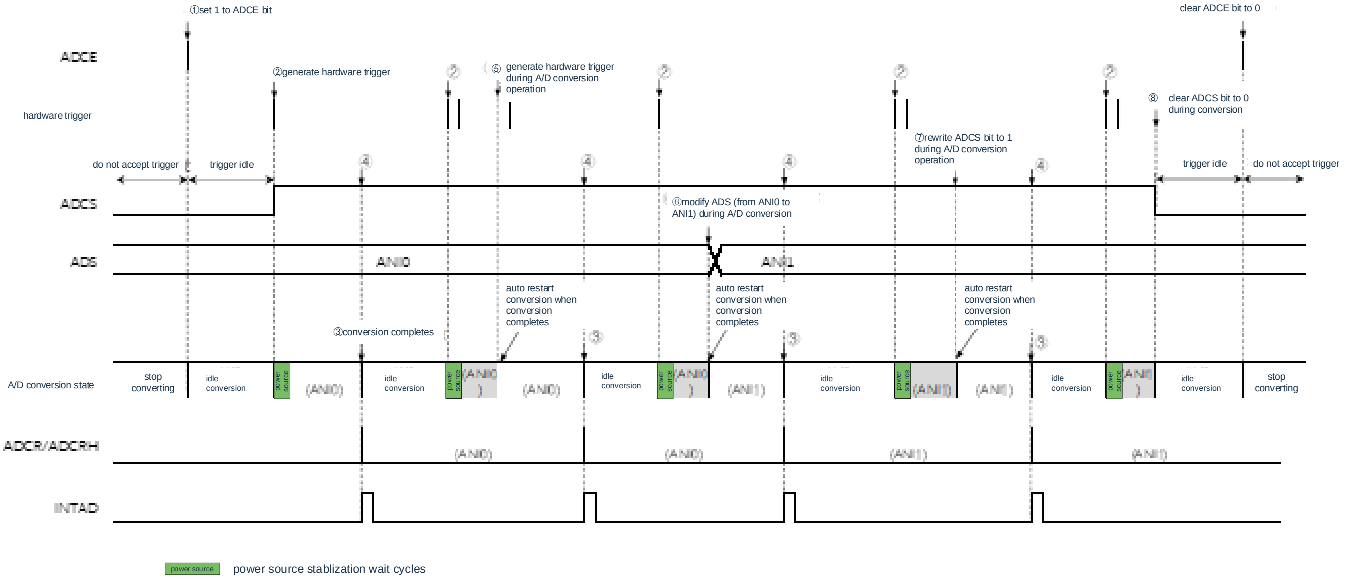
(5) If the input hardware triggers during the conversion, the current A/D conversion is aborted immediately and then restarted.

(6) If the ADS registers are overwritten or overwritten during the conversion, the current A/D conversion is aborted immediately, and then the analog input respecified by the ADS register is A/D converted.

(7) If you override the “1” to the ADCS bit during the conversion, the current A/D conversion is aborted immediately and the conversion begins again.

(8) If the ADCS bit is “0” during the transition, the current A/D transition is aborted immediately, then enters a hardware-triggered standby state, and the A/D converter enters a stopped state. When the ADCE bit is “0”, even the input hardware trigger is ignored and the A/D conversion does not begin.

Figure11-25 Timing example of hardware trigger wait mode (select mode, single conversion mode)



11.4.11 Hardware trigger wait mode (scan mode, continuous conversion mode)

(1) In the stop state, the ADCE bit of the mode register 0 (ADM0) of the A/D converter is “1” into a hardware-triggered standby state.

(2) If hardware triggers are entered in hardware-triggered standby, A/D conversion is performed on the four analog input channels specified by the analog input channel specified registers (ADS) from scan 0 to scan 3. The ADCS bit of the ADM0 register is automatically “1” while the input hardware triggers. A/D conversion is performed sequentially from the analog input channels specified by Scan 0.

(3) A/D conversion of 4 analog input channels in succession. Whenever the A/D conversion ends, the conversion results are saved to the A/D conversion result register (ADCR, ADCRH) and an A/D conversion end interrupt request signal is generated (INTAD). Immediately after the A/D conversion of the 4 channels is completed, the next A/D conversion is automatically started from the set channel.

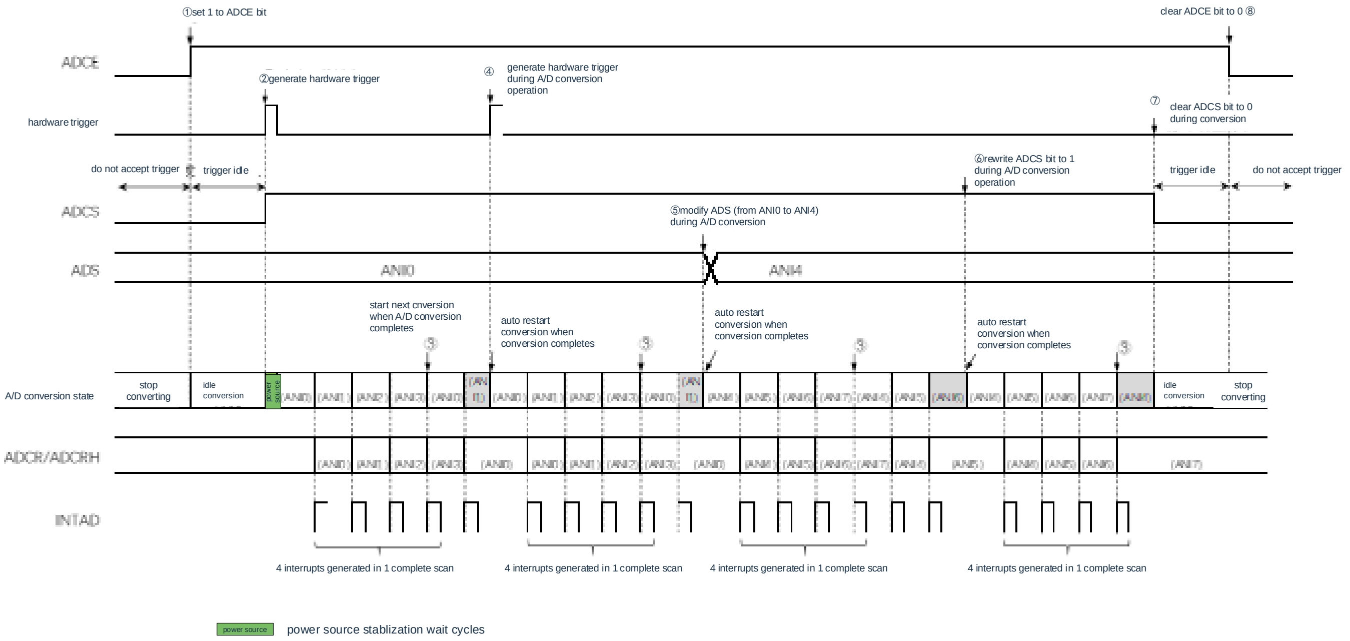
(4) If the input hardware triggers during the conversion, the current A/D conversion is aborted immediately and then restarted from the original channel.

(5) If the ADS registers are overwritten or overwritten during the conversion, the current A/D conversion is aborted immediately and the scan conversion begins with the channel respecified by the ADS registers.

(6) If you override the ADCS bit “1” during the conversion process, the current A/D conversion is aborted immediately and the conversion is restarted from the original channel.

(7) If the ADCS bit is “0” during the transition, the current A/D transition is aborted immediately, then enters a hardware-triggered standby state, and the A/D converter enters a stopped state. When the ADCE bit is “0”, even the input hardware trigger is ignored and the A/D conversion does not begin.

Figure11-26 Timing example of hardware trigger wait mode (scan mode, continuous conversion mode)


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
③
③
③
A/D conversion state
stop converting
idle conversion
power source
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

11.4.12 Hardware trigger wait mode (scan mode, single conversion mode)

(1) In the stop state, the ADCE bit of the mode register 0 (ADM0) of the A/D converter is “1” into a hardware-triggered standby state.

(2) If hardware triggers are entered in hardware-triggered standby, A/D conversion is performed on the four analog input channels specified by the analog input channel specified registers (ADS) from scan 0 to scan 3. Automatically puts the ADCS bit of the ADM0 register “1” after the input hardware triggers. A/D conversion is performed sequentially from the analog input channels specified by Scan 0.

(3) A/D conversion of 4 analog input channels in succession. Whenever the A/D conversion ends, the conversion results are saved to the A/D conversion result register (ADCR, ADCRH) and an A/D conversion end interrupt request signal is generated (INTAD).

(4) After the A/D conversion is complete, the ADCS bit automatically clears “0” and the A/D converter enters a stopped state.

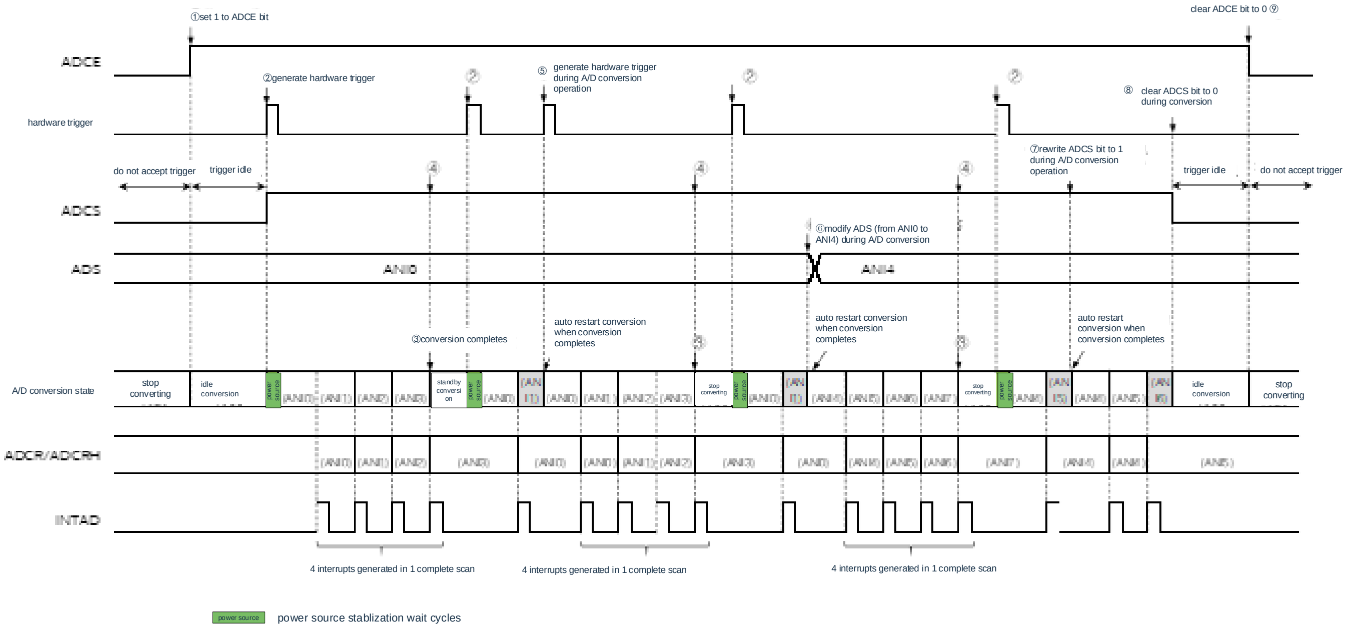
(5) If the input hardware triggers during the conversion, the current A/D conversion is aborted immediately and then the scan conversion is re-started from the original channel.

(6) If the ADS registers are overwritten or overridden during the conversion, the current A/D conversion is aborted immediately and the conversion is then scanned from the channel respecified by the ADS registers.

(7) If you rewrite the ADCS bit “1” during the conversion process, the current A/D conversion is aborted immediately, and the conversion is scanned from the initial channel.

(8) If the ADCS bit is “0” during the transition, the current A/D transition is aborted immediately, then enters a hardware-triggered standby state, and the A/D converter enters a stopped state. When the ADCE bit is “0”, even the input hardware trigger is ignored and the A/D conversion does not begin.

Figure11-27 Timing example of hardware trigger wait mode (scan mode, single conversion mode)


①set 1 to ADCE bit
clear ADCE bit to 0 ⑨
ADCE
②generate hardware trigger
⑤ generate hardware trigger during A/D conversion operation
⑧ clear ADCS bit to 0 during conversion
hardware trigger
do not accept trigger
trigger idle
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
A/D conversion state
stop converting
idle conversion
power source
standby conversi on
stop converting
stop converting
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

Chapter 12 Universal Serial Communication Unit

Unit 0 of the universal serial communication unit has 4 serial channels and unit 1 has 2 serial channels, each channel can achieve 3-wire serial (SSPI), UART, and simple I^{2}C communication function.

The functions of each channel supported by this product are assigned as follows:

Unit	Channel	Used as SSPI	Used as a UART	Used as a simple I^{2}C
0	0	SSPI00 (Support slave select input function)	UART0	IIC00
	1	SSPI01		IIC01
	2	SSPI10	UART1	IIC10
	3	SSPI11		IIC11
1	0	SSPI20	UART2	IIC20
	1	SSPI21		IIC21

When UART0 is used for Channel 0 and Channel 1 of Unit 0, SSPI00 and SSPI01 cannot be used, but SSPI10, UART1 and IIC10 on channel 2 and channel 3 can be used.

Note The following sections of this chapter describe the cell and channel structure of the 48-pin product.

12.1 Function of universal serial communication unit

The characteristics of each serial interface supported by this product are as follows.

12.1.1 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)

Data is transmitted and received synchronously with the serial clock (SCLK) output by the master device.

This is a clock synchronous communication function that uses a serial clock (SCLK), a transmit serial data (SDO), and a receive serial data (SDI) for communication on a total of three communication lines.

For specific setup examples, see “12.5 3-Wire Serial I/O (SSPI00, SSPI01, SSPI10, SSPI10, SSPI11, SSPI20, SSPI21) communication operation”.

[Transmit and receive data]

- 7-bit or 8-bit data length
- Phase control of transmitting and receiving data
- MSB/LSB preferred choice

[Clock control]

- Master or slave selection
- Phase control of input/output clocks
- Sets the transfer period generated by the prescaler and the in-channel counter.
- Maximum transfer rate Note

Master communication: Max.f CLK/2 (SSPI00 only) Max.f_{CLK}/4

Slave communication: Max.f_{MCK}/6

[Interrupt function]

- End of transfer interrupt, buffer null interrupt

[Error detection flag]

- Overflow error

Note It must be used within the range that satisfies the SCLK cycle time (t_{KCY}) characteristic. Please refer to the data sheet for details.

12.1.2 UART (UART0~UART2)

This is a function that communicates asynchronously through a total of two lines: serial data transmission (TxD) and serial data reception (RxD). Using these two communication lines, data is sent and received asynchronously (using the internal baud rate) with other communicating parties in a data frame (consisting of a start bit, data, parity bit, and stop bit). Full-duplex UART communication can be achieved by using two channels, send private (even channel) and receive private (odd channel).

For specific setting examples, see “12.7 Operation of UART (UART0~UART2) communication“.

[Transmit and receive data]

- 7-bit, 8-bit or 9-bit data length Note
- MSB/LSB preferred choice
- Level settings for sending and receiving data, selection of inversions
- Appending parity function for parity bits
- Appending stop bits

[Interrupt function].

- End of transfer interrupt, buffer empty interrupt
- Error interrupts caused by frame errors, parity errors, or overflow errors

[Error detection flag].

- Frame errors, parity errors, overflow errors

12.1.3 Simple I^{2}C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)

This is a function that synchronizes clock communication with multiple devices through a total of 2 lines of serial clock (SCL) and serial data (SDA). Because this simple I2C is designed for single communication with EEPROM, flash memory, A/D converters, etc., it is only used as a master device.

For start and stop conditions, AC specifications must be adhered to and processed by software while operating the control registers. For specific setup examples, refer to “12.9 Simplified I2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication operation.

[Transmit and receive data]

- Master sending, master receiving (limited to single master control functions).
- ACK output function Note, ACK detection function
- 8 bits of data length (when sending the address, specify the address with a high 7 bits, and use the lowest bit for R/W control).
- Manual generation of start and stop conditions

[Interrupt function]

- End of transfer interruption

[Error detection flag]

- ACK error, overflow error.

※[Features not supported by Simplified I^{2}C]

- Slave send, slave receive
- Quorum failure detection function
- Wait for detection function

Note When receiving the last data, if you write “0” to the SOEmn bit (serial output enable register m (SOEm)) to stop the output of the serial communication data, the ACK is not output. For details, please refer to “12.9.3 Processing Flow”.

Note When using the full-featured I^{2}C-bus, refer to Chapter 14, Serial Interface IICA.

12.2 Structure of universal serial communication unit

The universal serial communication unit consists of the following hardware.

Table 12-1 Structure of universal serial communication unit

Project	Structure
Shift register	8-bit or 9-bit Note 1
Buffer register	The serial data register mn (SDRmn) is 8 bits low or 9 bits Note 1 and 2
Serial clock input/output	SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, SCLK21 pins (for 3-wire serial I/O). SCL00, SCL01, SCL10, SCL11, SCL20, SCL21 pins (for Easy I^{2}C)
Serial data input	SDI00, SDI01, SDI10, SDI11, SDI20, SDI21 pins (for 3-wire serial I/O), RxD0, RxD1, RxD2 pins (for UART).
Serial data output	SDO00, SDO01, SDO10, SDO11, SDO20, SDO21 pins (for 3-wire serial I/O), TxD0, TxD1, TxD2 pins (for UART).
Serial data input/output	SDA00, SDA01, SDA10, SDA11, SDA20, SDA21 pins (for Simplified I^{2}C)
Slave Select Input	SS00 pin (for slave select input function).
Control registers	< Register of Unit Setting Section > • Peripheral Enable register 0 (PER0). • Serial clock selection register m (SPSm). • Serial channel enable status register m (SEm). • Serial channel start register m (SSm). • Serial channel stop register m (STm). • Serial output allows register m (SOEm). • Serial output register m (SOm). • Serial output level register m (SOLm). • Input switch control register (ISC). • Noise filter enable register 0 (NFEN0).
	< register for each channel > • Serial data register mn (SDRmn). • Serial mode register mn (SMRmn). • Serial communication operation setting register mn (SCRmn). • Serial status register mn (SSRmn). • Serial flag clear trigger register mn (SIRmn).
	• Port multiplexing function configuration register (PxxCFG). • Port output mode register (POMxx). • Port mode register (PMxx). • Port register (Pxx).

Note 1 The number of bits used as shift registers and buffer registers varies by unit and channel.

- mn = 00, 01: low 9-bit
- Others: low 8-bit

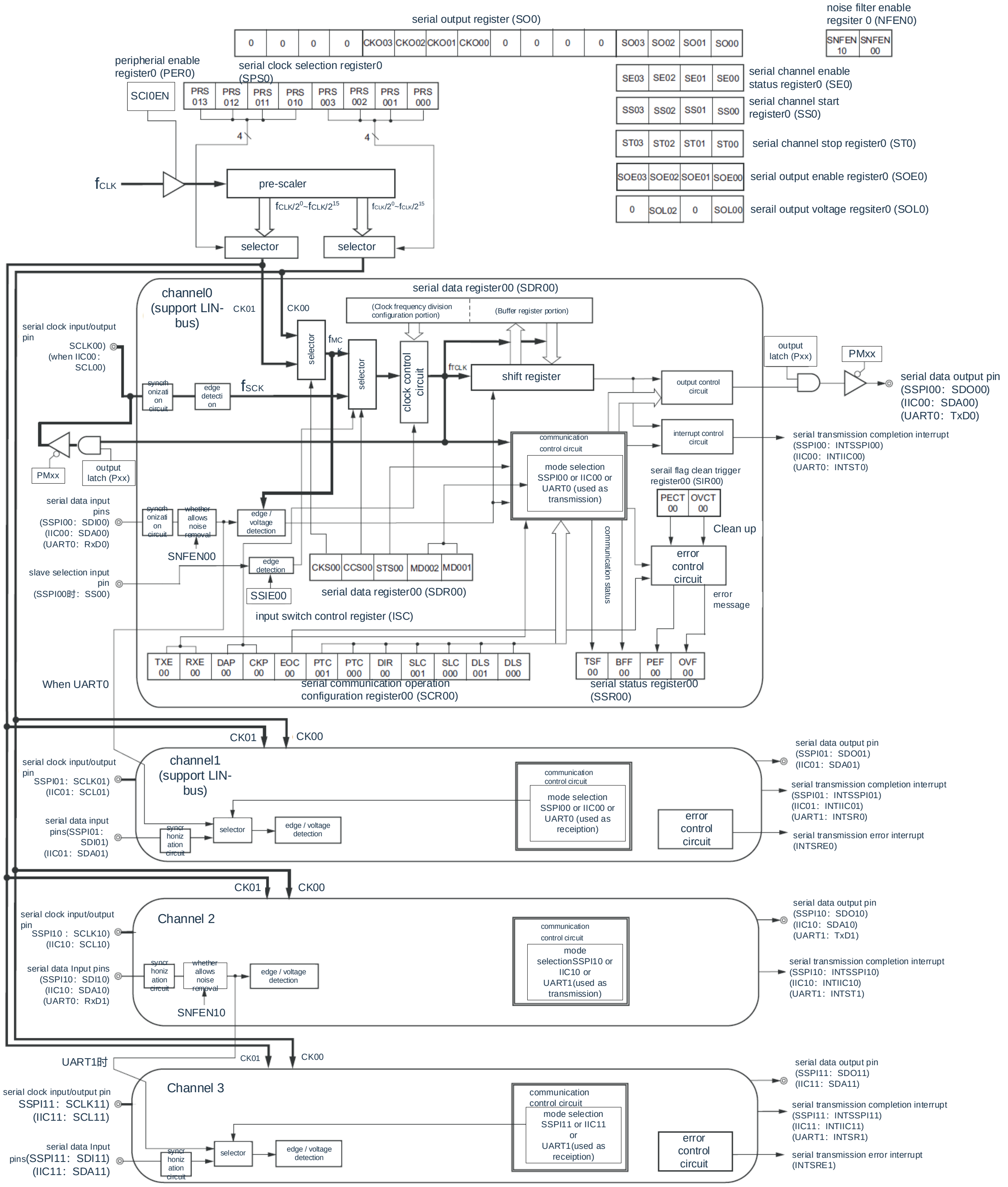
2. Depending on the communication mode, the lower 8 bits of the serial data register mn (SDRmn) can be read and written with the following SFR name.

- SSPIp communication... SIOp (SSPIp Data Register).
- UARTq receives... RXDq (UARTq Receive Data Register).
- UARTq sends... TXDq (UARTq Transmit Data Register).
- IICr Communications... SIOr (IICr Data Register).

Remark m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
q: UART number (q=0~2) r: IIC number (r=00, 01, 10, 11, 20, 21)

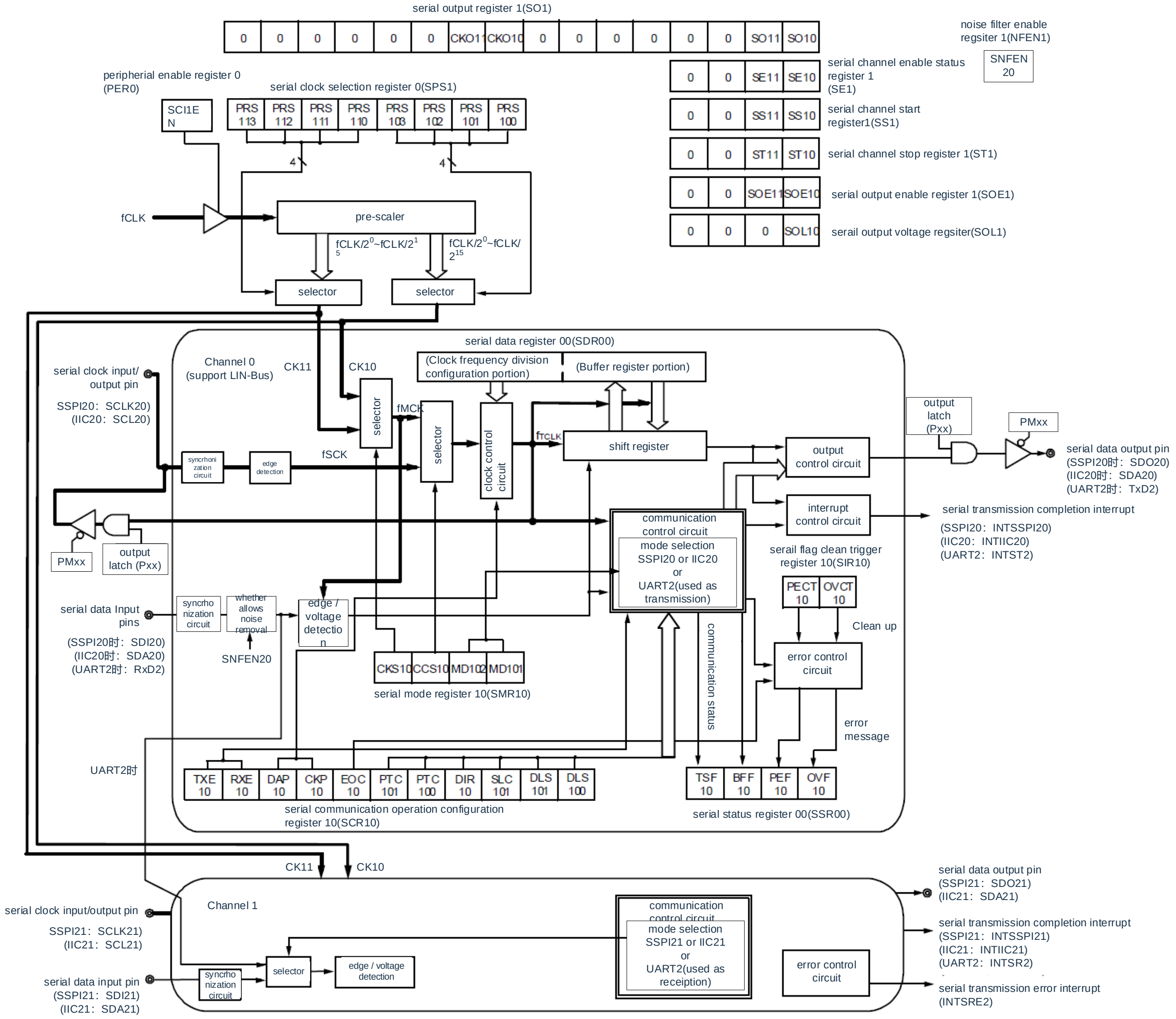
A block diagram of universal serial communication unit 0 is shown in Figure 12-1.

Figure 12-1 Diagram of universal serial communication unit 0


serial output register (SO0)
0
0
0
0
CKO03
CKO02
CKO01
CKO00
0
0
0
0
SO03
SO02
SO01
SO00
noise filter enable regsiter 0 (NFEN0)
SNFEN 10
SNFEN 00
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
SE03
SE02
SE01
SE00
serial channel enable status register0 (SE0)
SS03
SS02
SS01
SS00
serial channel start register0 (SS0)
ST03
ST02
ST01
ST00
serial channel stop register0 (ST0)
SOE03
SOE02
SOE01
SOE00
serial output enable register0 (SOE0)
0
SOL02
0
SOL00
serail output voltage regsiter0 (SOL0)
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
SCLK00) (when IIC00: SCL00)
selector
fMCK
selector
clock control circuit
fTCLK
shift register
output latch (Pxx)
PMxx
serial data output pin (SSPI00: SDO00) (IIC00: SDA00) (UART0: TxD0)
synchronization circuit
edge detection
fSCK
output control circuit
interrupt control circuit
serial transmission completion interrupt (SSPI00: INTSSPI00) (IIC00: INTIIC00) (UART0: INTST0)
PMxx
output latch (Pxx)
communication control circuit
mode selection SSPI00 or IIC00 or UART0 (used as transmission)
serial flag clean trigger register00 (SIR00)
PECT 00
OVCT 00
Clean up
serial data input pins (SSPI00: SDI00) (IIC00: SDA00) (UART0: RxD0)
synchronization circuit
whether allows noise removal
edge / voltage detection
SNFEN00
edge detection
CKS00
CCS00
STS00
MD002
MD001
communication status
error control circuit
error message
slave selection input pin (SSPI00时: SS00)
SSIE00
serial data register00 (SDR00)
input switch control register (ISC)
TXE 00
RXE 00
DAP 00
CKP 00
EOC 00
PTC 001
PTC 000
DIR 00
SLC 001
SLC 000
DLS 001
DLS 000
serial communication operation configuration register00 (SCR00)
TSF 00
BFF 00
PEF 00
OVF 00
serial status register00 (SSR00)
When UART0
CK01
CK00
channel1 (support LIN-bus)
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 SSPI00 or IIC00 or UART0 (used as receiption)
error control circuit
serial data output pin (SSPI01: SDO01) (IIC01: SDA01)
serial transmission completion interrupt (SSPI01: INTSSPI01) (IIC01: INTIIC01) (UART1: INTSR0)
serial transmission error interrupt (INTSRE0)
CK01
CK00
Channel 2
serial clock input/output pin SSPI10 : SCLK10) (IIC10: SCL10)
serial data Input pins (SSPI10: SDI10) (IIC10: SDA10) (UART0: RxD1)
synchonization circuit
whether allows noise removal
edge / voltage detection
SNFEN10
communication control circuit
mode selectionSSPI10 or IIC10 or UART1(used as transmission)
serial data output pin (SSPI10: SDO10) (IIC10: SDA10) (UART1: TxD1)
serial transmission completion interrupt (SSPI10: INTSSPI10) (IIC10: INTIIC10) (UART1: INTST1)
UART1时
CK01
CK00
Channel 3
serial clock input/output pin SSPI11: SCLK11) (IIC11: SCL11)
serial data Input pins(SSPI11: SDI11) (IIC11: SDA11)
synchonization circuit
selector
edge / voltage detection
communication control circuit
mode selection SSPI11 or IIC11 or UART1(used as receiption)
error control circuit
serial data output pin (SSPI11: SDO11) (IIC11: SDA11)
serial transmission completion interrupt (SSPI11: INTSSPI11) (IIC11: INTIIC11) (UART1: INTSR1)
serial transmission error interrupt (INTSRE1)

A block diagram of a universal serial communication unit 1 is shown in Figure 12-2.

Figure 12-2 Diagram of universal serial communication unit 1


serial output register 1(SO1)
0 0 0 0 0 0 CKO11 CKO10 0 0 0 0 0 0 SO11 SO10
noise filter enable regsiter 1(NFEN1)
SNFEN 20
serial channel enable status register 1 (SE1)
0 0 SE11 SE10
serial channel start register1(SS1)
0 0 SS11 SS10
serial channel stop register 1(ST1)
0 0 ST11 ST10
serial output enable register 1(SOE1)
0 0 SOE11 SOE10
serail output voltage regsiter(SOL1)
0 0 0 SOL10
peripheral enable register 0 (PER0)
SCI1EN
serial clock selection register 0(SPS1)
PRS 113 PRS 112 PRS 111 PRS 110 PRS 103 PRS 102 PRS 101 PRS 100
4
4
fCLK
pre-scaler
fCLK/2^0~fCLK/2^15
fCLK/2^0~fCLK/2^15
selector
selector
serial data register 00(SDR00)
(Clock frequency division configuration portion)
(Buffer register portion)
Channel 0 (support LIN-Bus)
CK11
CK10
serial clock input/ output pin
SSPI20：SCLK20)
(IIC20：SCL20)
fMCK
fTCLK
fSCK
selector
selector
clock control circuit
shift register
synchronization circuit
edge detection
output latch (Pxx)
PMxx
output control circuit
serial data output pin
(SSPI20时：SDO20)
(IIC20时：SDA20)
(UART2时：TxD2)
communication control circuit
mode selection SSPI20 or IIC20 or UART2(used as transmission)
interrupt control circuit
serial transmission completion interrupt
(SSPI20：INTSSPI20)
(IIC20：INTIIC20)
(UART2：INTST2)
PMxx
output latch (Pxx)
serail flag clean trigger register 10(SIR10)
PECT 10
OVCT 10
Clean up
serial data Input pins
(SSPI20时：SDI20)
(IIC20时：SDA20)
(UART2时：RxD2)
synchronization circuit
whether allows noise removal
SNFEN20
edge / voltage detection
CKS10 CCS10 MD102 MD101
serial mode register 10(SMR10)
communication status
error control circuit
error message
UART2时
TXE 10 RXE 10 DAP 10 CKP 10 EOC 10 PTC 101 PTC 100 DIR 10 SLC 101 DLS 101 DLS 100
serial communication operation configuration register 10(SCR10)
TSF 10 BFF 10 PEF 10 OVF 10
serial status register 00(SSR00)
CK11
CK10
Channel 1
serial clock input/output pin
SSPI21：SCLK21)
(IIC21：SCL21)
serial data input pin
(SSPI21：SDI21)
(IIC21：SDA21)
synchronization circuit
selector
edge / voltage detection
communication control circuit
mode selection SSPI21 or IIC21 or UART2(used as receiption)
error control circuit
serial data output pin
(SSPI21：SDO21)
(IIC21：SDA21)
serial transmission completion interrupt
(SSPI21：INTSSPI21)
(IIC21：INTIIC21)
(UART2：INTSR2)
serial transmission error interrupt
(INTSRE2)

12.2.1 Shift register

This is a 9-bit register that converts parallel and serial to and from each other.

For UART communication at 9 bits of data length, use 9 bits (bit0 to 8) Note 1. Converts the input data of the serial input pin into parallel data when receiving data; When data is sent, the value to this register will be transferred as serial data from the serial output pin output Note 1. Shift registers cannot be manipulated directly through the program.

To read and write data from shift registers, use the low 8 or 9 bits of the serial data register mn (SDRmn).

	8	7	6	5	4	3	2	1	0
Shift register									

12.2.2 Low 8 bits or low 9 bits of serial data register mn (SDRmn)

The SDRmn register is the transmit and receive data registers (16-bit) of channel n.

Bit8~0 (low 9 bits) Note or bit7~0 (low 8 bits) is used as the transmit and receive buffer registers bit15~9 is used as a crossover setting register for the running clock (f_{MCK}).

When receiving data, save the parallel data converted by the shift register to the lower 8 bits or the lower 9 bits; When transmitting data, the transmitted data to the shift register is set to a lower 8 bits or a low 9 bits.

Regardless of the output order of the data, set registers mn (SCRmn) bit0 and bit1 (DLSmn0, DLSmn1) are run according to serial communication) settings saved to the lowest 8 bits or lower 9 bits of data as follows:

- 7-bit data length (saved in bit0~6 of the SDRmn register).
- 8 bits of data length (saved in bit0~7 of the SDRmn register).
- 9 bits of data length (saved in bit0~8 of the SDRmn register) Note 1

SDRmn registers can be read and written in 16-bit increments.

Depending on the communication mode, the low 8 bits of the SDRmn register or the low 9 bits of the SDRmn register can be read and written in 8-bit units with the following SFR nameNote 2.

- SSPIp communication... SDIOp (SSPIp Data Register).
- UARTq receives... RXDq (UARTq Receive Data Register).
- UARTq sends... TXDq (UARTq Transmit Data Register).
- IICr Communications... SDIOr (IICr Data Register).

After the reset signal is generated, the value of the SDRmn register changes to “0000H”.

Note 1 Only UART0 supports 9-bit data length.

2. When the operation is stopped (SEmn=0), it is forbidden to override SDRmn [7:0] via 8-bit memory operation instructions (otherwise, SDRmn [15:9] is cleared).

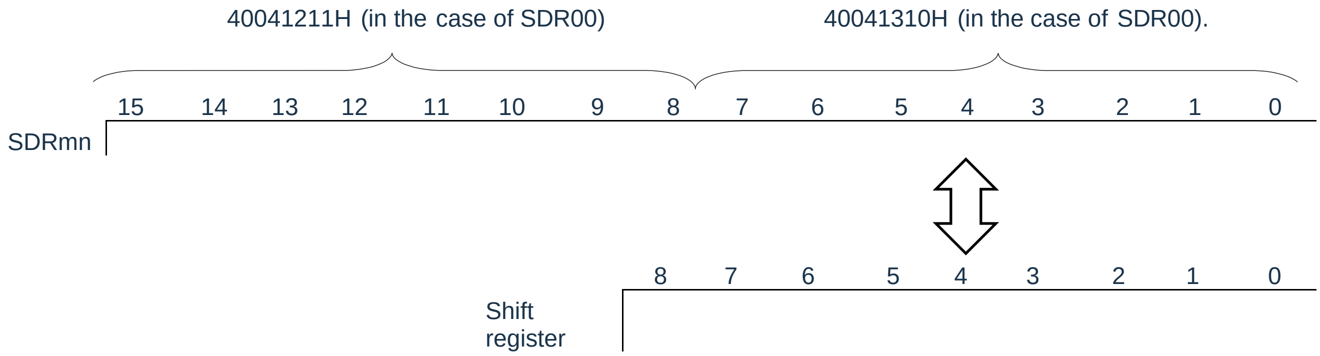
Note 1 After the reception ends, bits from bit0 to 8 that exceed the length of the data are “0”.

2.m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)

q: UART number (q=0~2) r: IIC number (r=00, 01, 10, 11, 20, 21)

Figure 12-3 Format of serial data register mn (SDRmn) (mn=00, 01, 10, 11)

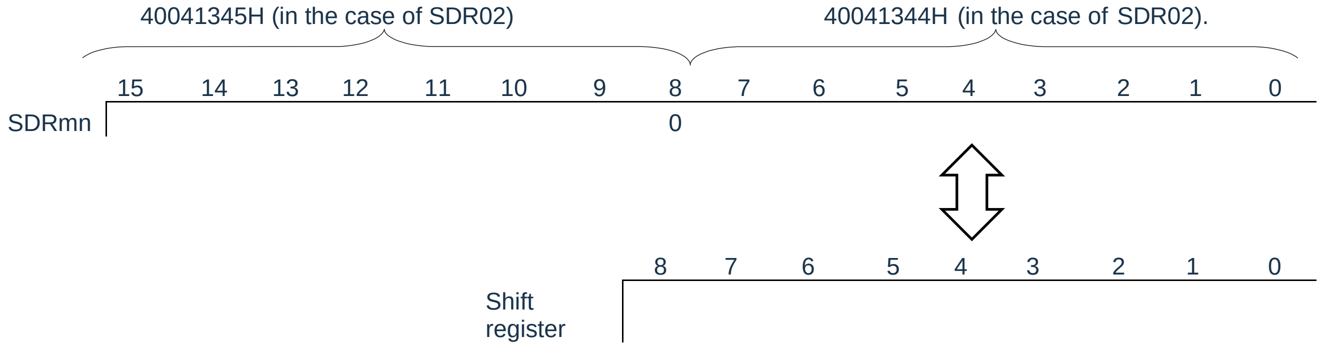
Address: 40041310H (SDR00), 40041312H (SDR01) after reset: 0000HR/W
40041748H(SDR10), 4004174AH(SDR11)



Note For the function of the SDRmn register with a high 7 bit, refer to“12.3 Registers for controlling universal serial communication unit”

Figure 12-4 Format of serial data register mn (SDRmn) (mn=02, 03, 10, 11, 12, 13)

Address: 4004134 4H (SDR02), 40041346H (SDR03) after reset: 0000HR/W



Note Bit8 must be set to “0”.

For the function of the SDRmn register with a high 7 bit, refer to “12.3 Registers for controlling universal serial communication unit”.

12.3 Registers for controlling universal serial communication unit

The registers that control the universal serial communication unit are as follows:

- Peripheral enable register 0 (PER0).
- Serial clock selection register m (SPSm).
- Serial mode register mn (SMRmn).
- Serial communication operation setting register mn (SCRmn).
- Serial data register mn (SDRmn).
- Serial flag clears trigger register mn (SDIRmn).
- Serial status register mn (SSRmn).
- Serial channel start register m (SSm).
- Serial channel stop register m (STm).
- Serial channel allows status register m (SEm).
- Serial output allows register m (SOEm).
- Serial output level register m (SOLm).
- Serial output register m (SOm).
- Input Switch Control Register (ISC).
- Noise filter allows register 0 (NFEN0).
- Port multiplexing function configuration register (PxxCFG).
- Port output mode register (POMx).
- Port mode register (PMx).
- Port register (Px).

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

12.3.1 Peripheral enable register 0 (PER0).

The PER0 register is a register that sets the clock to be allowed or disallowed to be supplied to each peripheral hardware. Reduce power consumption and noise by stopping clocking unused hardware.

To use universal serial communication unit 0, bit2 (SCI0EN) must be set to “1”.

To use universal serial communication unit 1, bit3 (SCI1EN) must be set to “1”.

The PER0 register is set via an 8-bit memory operation command.

After the reset signal is generated, the value of the PER0 register changes to “00H”.

Figure 12-5 Format of peripheral enable register 0 (PER0)

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

symbol	7	6	5	4	3	2	1	0
PER0	RTCEN	GODAEN	ADCEN	IICA0EN	SCI1EN	SCI0EN	TM41IN	TM40EN

SCImEN	Provides control of the input clock of the universal serial communication unit m
0	Stop supplying the input clock. • You cannot write a generic serial communication unit m using SFR. • Universal serial communication unit m is in a reset state.
1	Allows the input clock to be provided. • SFR can read and write to the general serial communication unit m used.

Note 1 To set the universal serial communication unit m, the following registers must first be set in the S CImEN 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 value is the initial value (input switch control register (ISC), The noise filter allows register 0 (NFEN0), the port multiplexing function configuration register (Px xCFG), and the port output mode register (POMx), port mode registers (PMx), port mode control registers (PMCx), and port registers (Px).

- Serial clock selection register m (SPSm).
- Serial mode register mn (SMRmn).
- Serial communication operation setting register mn (SCRmn).
- Serial data register mn (SDRmn).
- Serial flag clear trigger register mn (SIRmn).
- Serial status register mn (SSRmn).
- Serial channel start register m (SSm).
- Serial channel stop register m (STm).
- Serial channel enable status register m (SEm).
- Serial output enable register m (SOEm).
- Serial output level register m (SOLm).
- Serial output register m (SOm).

12.3.2 Serial clock select register m (SPSm)

The SPSm register is a 16-bit register that selects two common operating clocks (CKm0, CKm1) available to each channel. CKm1 is selected by bit7~4 of the SPSm register, and by bit3~0 CKm0.

It is forbidden to overwrite the SPSm register during operation (SEmn=1).

The SPSm register is set via 16-bit memory operation instructions.

I can set the low 8 bits of the SPSm register with SPSmL and via 8-bit memory operation instructions.

After the reset signal is generated, the value of the SPSm register changes to “0000H”.

Figure 12-6 Format of serial clock selection register m (SPSm)

Address: 40041126H (SPS0), 40041566H (SPS1) after reset: 0000HR/W

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

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

Note To change the clock selected as f_{CLK} (change the value of the system clock control register (CKC)) during the operation of the Universal Serial Communication Unit (SCI), the operation of the SCI must be stopped (serial channel stop register m (STm)=000FF) and then make the change.

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).

3. k=0, 1

12.3.3 Serial mode register mn (SMRmn)

The SMRmn register is a register that sets the channel n operating mode, selects the operating clock (f_{MCK}), specifies whether the serial clock (f_{SCLK}) input can be used, sets the start trigger, and operates the mode (SSPI, UART, Simplified I^{2}C) settings and interrupt source selection. In addition, the inverting level of the received data is set only in UART mode.

Overwriting the SMRmn register during operation (SEmn=1) is prohibited, but the MDmn0 bit can be overridden during operation.

The SMRmn register is set via a 16-bit memory operation command.

After the reset signal is generated, the value of the SMRmn register changes to “0020H”.

Figure 12-7 Format of serial mode register mn (SMRmn) (1/2)

Address: 40041110H(SMR00)~40041116H(SMR03) After reset: 0020HR/W
40041550H(SMR10)~40041552H(SMR11)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn	CCSmn	0					STSmn Note1		SISmn0 Note1		0		MDmn2	MDmn1	MDmn0

CKSmn	Selection of channel n operating clock (f_{MCK})
0	The SPSm register sets the operating clock CKm0
1	The SPSm register sets the operating clock CKm1
The operating clock (f_{MCK}) is used for edge detection circuitry. The transmit clock (f_{TCLK}) is generated by setting the CCSmn bit and the SDRmn register high 7 bits.	

CCSmn	Selection of channel n transmit clock (f_{TCLK})
0	The CKSmn bit specifies the crossover clock of the running clock f_{MCK}
1	The input clock f_{SCLK} from the SCLKp pin (slave transfer in SSPI mode).
The transmit clock f_{TCLK} is used for shift registers, communication control circuits, output controllers, interrupt control circuits, and error control circuits. When the CCSmn bit is “0”, the runtime clock (f_{MCK}) is divided by the high 7 bits of the SDRmn register.	

STSmn Note1	Selection of start triggering sources
0	Only software triggers are valid (selected in SSPI, UART Send, Simplified I2C).
1	The effective edge of the RxDq pin (selected when received by the UART).
Transmission begins when the above conditions are met after setting the SSm register to “1”.	

Note 1 SMR01, SMR03, SMR11 registers only.

Note You must set bit13~9, 7, 4, 3 (SMR00, SMR02, SMR10 registers are bit13~6, 4, 3) to “0”, and set bit5 to “1”.

Remark m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
q: UART number (q=0~2) r: IIC number (r=00, 01, 10, 11, 20, 21)

Figure 12-7 Format of serial mode register mn (SMRmn) (2/2)

Address: 40041110H(SMR00)~40041116H(SMR03) After reset: 0020HR/W
40041550H(SMR10)~40041552H(SMR11)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKS mn	CCS mn	0					STS mn Note1		SISmn0 Note1		0		MD mn2	MD mn1	MD mn0

SISmn0[Note1]	Level inversion control of received data for channel n in UART mode
0	Detect the falling edge as the starting bit. The input communication data is not reversed.
1	Detect the rising edge as the starting point. The input communication data is reversed.

MDmn2	MDmn1	Setting of the channel n operating mode
0	0	SSPI mode
0	1	UART mode
1	0	Simplified I^{2}C mode
1	1	Disable settings.

MDmn0	Channel n interrupt source selection
0	The end of the transfer is interrupted
1	Buffer empty interrupt (Occurs when data is transferred from the SDRmn register to the shift register).
In continuous transmission, if the MDmn0 bit is “1” and the data for SDRmn is empty, the next sent data is made.	

Note 1 SMR01, SMR03, SMR11, registers only.

Note You must set bit13 to 9, 7, 4, 3 (SMR00, SMR02, SMR10 registers are bit13~6, 4, 3) to “0”, and set bit5 to 1”.

Remark m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
q: UART number (q=0~2) r: IIC number (r=00, 01, 10, 11, 20, 21)

12.3.4 Serial communication operation setting register mn (SCRmn)

The SCRmn register is the communication operation setting register for channel n, which sets the data transmit and receive modes, data and clock phases, whether to mask error signals, parity bits, start bits, stop bits, and data length.

It is forbidden to overwrite the SCRmn register during operation (SEmn=1).

The SCRmn register is set via a 16-bit memory operation command.

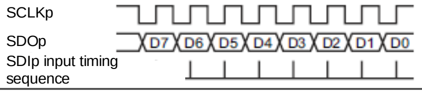
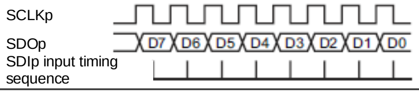
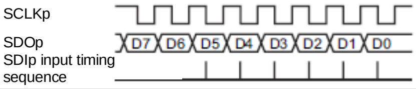
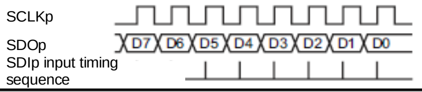
After the reset signal is generated, the value of the SCRmn register changes to “0087H”.

Figure 12-8 Format of serial communication operation setting register mn (SCRmn) (1/2)

Address: 40041118H(SCR00)~4004111EH(SCR03) After reset: 0087HR/W
40041558H(SCR10)~4004155AH(SCR13)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXE mn	RXE mn	Dap mn	CKP mn		EOC mn	PTC mn1	PTC mn0	You mn		SLC mn1 Note1	SLC mn0			DLS mn1 Note2	DLS mn0

TXEmn	RXEmn	Setting of the channel n operating mode
0	0	Prohibited communication.
0	1	Receive only.
1	0	Transmit only.
1	1	Enables transmit and receive.

DAPmn	CKPmn	data and clock phase selection in SSPI mode	Type
0	0	 SCLKp 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	Mask control of error interrupt signal (INTSREx (x=0 to 3))
0	Disable the generation of error interrupts INTSREx (generate INTSRx).
1	Enable error interrupt INTSREx (no INTSRx is generated when an error occurs).
The EOCmn bit must be set to "0" in SSPI mode and Simplified I^{2}C mode or when sending from UART Note 3.	

Note 1 Limited to SCR00, SCR02, SCR10 registers only.

2. Limited to SCR00 register and SCR01 register, the others are fixed as “1”.

3. When the EOCmn bit is “0” and SSPImn is not used, it is possible to generate an error interrupt INTSREn.

Notice Bit 3, 6, and 11 must be set to "0" (also bit 5 of SCR01, SCR03, and SCR11 registers must be set to "0"), and bit 2 must be set to "1".

Remark m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)

Figure 12-8 Format of serial communication operation setting register mn (SCRmn) (2/2)

Address: 40041118H(SCR00)~4004111EH(SCR03) After reset: 0087HR/W
40041558H(SCR10)~4004155AH(SCR13)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXEmn	RXEmn	Dapmn	CKPmn		EOCmn	PTCmn1	PTCmn0	Youmn		SLCmn1 Note1	SLCmn0			DISmn1 Note2	DLSmn0

PTCmn1	PTCmn0	Setting of parity bits in UART mode	
		Transmission	Reception
0	0	Parity bits are not output.	There is no parity at the time of reception.
0	1	Output parity[Note 3].	Parity is not judged.
1	0	Output parity.	Judgment even.
1	1	Output odd check.	Judgment odd check.
In SSPI mode and Simplified I²C mode, both the PTCmn1 bit and the PTCmn0 bit must be set to “0”.			

DIRmn	Selection of data transfer order in SSPI and UART modes
0	Performs MSB-priority input/output.
1	Perform LSB-priority input/output.
In Simplified I²C mode, the DIRmn bit must be “0”.	

SLCmn1[Note1]	SLCmn0	Setting of the stop bit in UART mode
0	0	No stop bits
0	1	Stop bit length = 1 bit
1	0	Stop bit length = 2 bits (limited to mn=00, 02, 10).
1	1	Disable settings.
If an end-of-transfer interrupt is selected, an interrupt occurs after all stop bits have been transferred. When UART is received or in Simplified I²C mode, it must be set to 1 stop bit (SLCmn1, SLCmn0=0, 1). In SSPI mode, it must be set to no stop (SLCmn1, SLCmn0=0, 0). When UART is sent, it must be set to 1 bit (SLCmn1, SLCmn0=0, 1) or 2 bits (SLCmn1, SLCmn0=1, 0).		

DLSmn1[Note2]	DLSmn0	Setting of data length in SSPI and UART modes
0	1	9 bits of data length (saved in bit0~8 of the SDRmn register) (selectable only in UART mode).
1	0	7 bits of data length (saved in bit0 to 6 of the SDRmn register).
1	1	8 bits of data length (bit0 to 7 in the SDRmn register).
other		Disable settings.
In Simplified I²C mode, both the DLSmn1 bit and the DLSmn0 bit must be set to “1”.		

Note 1 Limited to SCR00, SCR02, SCR10 registers only.

2. Limited to SCR00 register and SCR01 register, the others are fixed as “1”.

3. Nothing to do with the content of the data, always appended “0”.

Notice Bit 3, 6, and 11 must be set to "0" (also bit 5 of the SCR01, SCR03, and SCR11 registers must be set to "0"), and bit 2 must be set to "1".

Remark m: Unit number (m=0, 1) n: channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)

12.3.5 Serial data register mn (SDRmn).

The SDRmn register is the data register (16 bits) that channel n transmits and receives.

The bits 8 to 0 (low 9 bits) of SDR00 and SDR01 or bits 7 to 0 (low 8 bits) of SDR02, SDR03, SDR10 and SDR11 are used as transmit and receive buffer registers, and bits 15 to 9 (high 7 bits) are used as operating clock (f_{MCK}) divider setting registers.

If the CCSmn bit of the serial mode register mn (SMRmn) is set to "0", the divider clock of the operation clock set by bits 15 to 9 (high 7 bits) of the SDRmn register is used as the transmit clock.

If the CCSmn bit is set to "1", bit15~9 (high 7 bits) of SDRmn must be set to “0000000B”. The input clock f_{SCLK} (slave transfer in SSPI mode) of the SCLKp pin is the transmit clock.

The low 8 or 9 bits of the SDRmn register are used as transmit and receive buffer registers. When receiving data, shift registers are converted in parallel data is saved to the lowest 8 bits or the lower 9 bits; When transmitting data, the transmitted data to the shift register is set to a lower 8 bits or a low 9 bits.

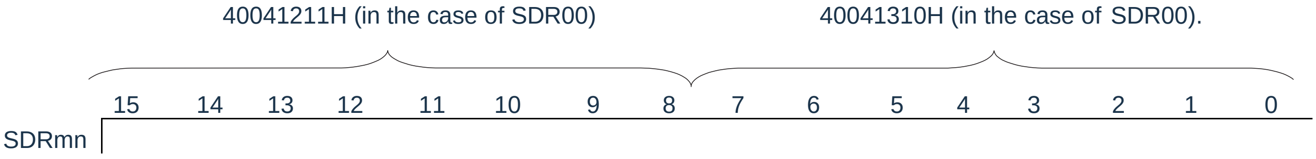
SDRmn registers can be read and written in 16-bit increments. However, high 7 bits can only be read and written when the operation is stopped (SEmn=0). In operation (SEmn=1) only the low 8 bits or 9 bits of the SDRmn register can be written, and the high 7 bits of the SDRmn register are always read as "0".

After the reset signal is generated, the value of the SDRmn register changes to “0000H”.

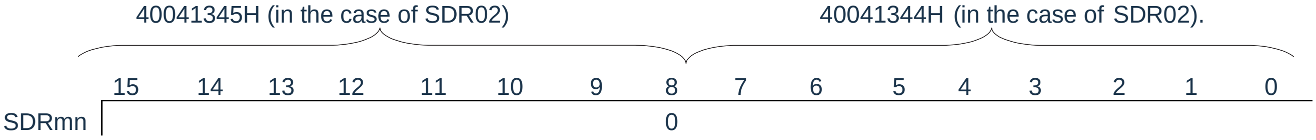
Figure 12-9 Format of serial data register mn (SDRmn)

Address: 40041310 H (SDR00), 40041312H (SDR01) After reset: 0000HR/W

40041748H(SDR10), 4004174AH(SDR11)


40041211H (in the case of SDR00)
40041310H (in the case of SDR00).
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
SDRmn

Address: 4004134 4H (SDR02), 40041346H (SDR03) after reset: 0000HR/W


40041345H (in the case of SDR02)
40041344H (in the case of SDR02).
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
SDRmn
0

SDRmn[15:9]							Run the transmit clock setting for clock division
		0				0	$f_{MCK}/2$
		0				1	$f_{MCK}/4$
		0				0	$f_{MCK}/6$
		0				1	$f_{MCK}/8$
			• • •			• • •	• • •
		1				0	$f_{MCK}/254$
		1				1	$f_{MCK}/256$

Note 1. Bit8 of the SDR02, SDR03, SDR10, and SDR11 registers must be set to “0”.

2. When using UART, it is forbidden to set SDRmn [15:9] to “0000000B” and “0000001B”.

3. When using Simplified I2C, it is forbidden to set SDRmn[15:9] to “0000000B”, and the SDRmn [15:9] setting value must

be greater than or equal to “0000001B”.

4. When the operation is stopped (SEmn=0), it is forbidden to override SDRmn [7:0] via 8-bit memory operation instructions (otherwise, SDRmn [15:9] is all cleared “0”).

Note 1 For the function of the SDRmn register with the low 8 or 9 bits, refer to “12.2 Structure of universal serial communication unit.

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

12.3.6 Serial flag clear trigger register mn (SIRmn)

This is the trigger register used to clear each error flag of channel n.

If you set the various (FECTmn, PECTmn, OVCTmn) to “1”, the corresponding bits (FEFmn, PEFmn, OVFmn) clear “0”. Because the SDIRmn register is a trigger register, if the corresponding bit of the SSRmn register is cleared, the SDIRmn register is also cleared immediately.

The SIRmn register is set via 16-bit memory operation instructions.

The low 8 bits of the SIRmn register can be set with SIRmnL and via 8-bit memory operation instructions.

After the reset signal is generated, the value of the SIRmn register changes to “0000H”.

Figure 12-10 Format of serial flag clear trigger register mn (SIRmn)

Address: 40041108H(SIR00)~4004110EH(SIR03) After reset: 0000HR/W
40041548H(SIR10)~4004154AH(SIR11)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIRmn			0											FECT Mn[Note1]	PEC Tmn	OVC Tmn

FECTmn[Note1]	Channel n-frame error flag clear trigger
0	No clearance.
1	Clear the FEFMN bit of the SSRmn register to “0”.

PECTmn	Channel n parity error flag clear trigger
0	No clearance.
1	Clear the PEFmn bit of the SSRmn register to “0”.

OVCTmn	Channel n overflow error flag clear trigger
0	No clearance.
1	Clear the OVFmn bit of the SSRmn register to “0”.

Note 1: Restricted to SIR01, SIR03, SIR11 registers only.

Notice Bit 15 to 3 (bit 15 to 2 for SIR00, SIR02 and SIR10 registers) must be set to "0".

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

2. Read value of the SIRmn register is always “0000H”.

12.3.7 Serial status register mn (SSRmn)

The SSRmn register indicates the communication status of channel n and the occurrence of errors. The errors represented are frame errors, parity errors, and overflow errors. The SSRmn register is read via 16-bit memory operation instructions.

The lower 8 bits of the SSRmn register can be read with SSRmnL and via 8-bit memory operation instructions.

After the reset signal is generated, the value of the SSRmn register changes to “0000H”.

Figure 12-11 Format of serial status register mn (SSRmn) (1/2)

Address: 40041100H (SSR00) ~ 40041106H (SSR03) After reset: 0000HR
40041540H(SSR10)~40041542H(SSR11)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSRmn			0							TSF mn	BFF mn	0		FEFMn[Note1]	PEF mn	OVF mn

TSFmn	Indication flag for channel n communication status
0	Communication stop state or communication standby state
1	Communication operation status
[Clear condition]. • When STmn of STm register is set to "1" (communication stopped state) or SSmn bit of SSm register is set to "1" (communication standby state) • When communication ends [Set condition]. • When communication begins	

BFFmn	Indication flag for channel n buffer register
0	The SDRmn register does not hold valid data.
1	The SDRmn register holds valid data.
[Clear condition]. •When the transmitted data from the SDRmn register to the shift register is transferred during the transmit process • When the received data is finished reading from the SDRmn register during the receive process • When the STmn bit of STmn register is set to "1" (communication stopped state) or the SSm bit of the SSm register is set to "1" (Communication enable state) [Set condition]. •When writing data to the SDRmn register in the state where the TXEmn bit of the SCRmn register is “1” (transmit, transmit, and receive modes in each communication mode). •When the received data is saved to the SDRmn register in the state where the RXEmn bit of the SCRmn register is “1” (receive mode, transmit and receive modes in each communication mode). •When a receive error occurs	

Note 1 For SSR01, SSR03, SSR11 registers only.

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

Figure 12-11 Format of serial status register mn (SSRmn) (2/2)

Address: 40041100H (SSR00) ~ 40041106H (SSR03) After reset: 0000HR
40041540H(SSR10)~40041542H(SSR11)

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSRmn			0							TSF mn	BFF mn	0		FEFmn[Note1]	PEF mn	OVF mn

FEFmn[Note1]	Detection flag for channel n frame errors
0	No errors occurred.
1	An error occurred (when the UART was received).
[Clear condition]. • When writing "1" to the FECTmn bit of the SIRmn register [Accent condition]. • When no stop bit is detected at the end of UART reception	

PEFmn	Detection flag for channel n parity errors
0	No errors occurred.
1	An error occurred (when the UART was received) or the ACK was not detected (when the I2C was sent).
[Clear condition]. • When writing "1" to the PECTmn bit of the SIRmn register [Set condition]. • When the parity and parity bits of the data sent at the end of the UART reception are different (parity errors). • When it is sent by I2C and when the ACK receiving timing slave does not return an ACK signal (no ACK detected).	

OVFmn	Detection flag for channel n overflow error
0	No errors occurred.
1	An error has occurred.
[Clear condition]. • When writing "1" to the OVCTmn bit of the SIRmn register [Set condition]. • In the state where the RXEmn bit of the SCRmn register is "1" (receive mode, transmit and receive modes in each communication mode), although the received data is saved in the SDRmn register, but when there is no read receive data and write send data or write down a received data • When data is not ready to be sent during slave send or slave send and receive in SSPI mode.	

Note 1 For SSR01, SSR03, SSR11 registers only.

Notice 1 If you write the SDRmn register when the BFFmn bit is "1", the saved transmit or receive data is corrupted and an overflow error is detected (OVEmn=1).

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

12.3.8 Serial channel start register m (SSm)

The SSm register is a trigger register that sets the communication/start count enabled for each channel.

If you write “1” to you (SSmn), set the corresponding bit (SEmn) of the serial channel enable status register m (SEmn) to “1” (operation enable status). Because the SSmn bit is the trigger bit, the SSmn bit is cleared immediately if the SEmn bit is “1”.

The SSm register is set via a 16-bit memory operation command.

I can set the lower 8 bits of the SSm register with SSmL and via 8-bit memory operation instructions.

After the reset signal is generated, the value of the SSm register changes to “0000H”.

Figure 12 12-12 Format of serial channel start register m (SSm)

Address: 40041122H (SS0) After reset: 0000HR/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS0			0										SS03	SS02	SS01	SS00

Address: 40041562H (SS1) After reset: 0000HR/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS1			0										0	0	SS11	SS10

SSm	Trigger at the beginning of channel n operation
	No triggering.
	Set the SEmn bit "1" and shift to communication standby state Note.

Note If the SSmn bit is set to “1” during communication, communication is stopped and enters standby. At this point, the values of the control register and shift register, the SCLKmn pin and the SDOmn pin, the FEFmn flag, the PEFmn flag, and the OVFmn flag remain in state.

Notice 1 Bit 15 to 4 of SS0 register and bit 15 to 2 of SS1 register must be set to "0".

2. For UART receive, at least 4 f_{MCK} clocks must be set to "1" after setting RXEmn in SCRmn register to "1", and then setting SSmn to "1".

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

2. The read value of the SSm register is always “0000H”.

12.3.9 Serial channel stop register m (STm)

The STm register is a trigger register that sets the communication/stop count allowed for each channel.

If a "1" is written to each bit (STmn), the corresponding bit (SEmn) in the serial channel enable status register m (SEm) is cleared to "0" (stop status). Since the STmn bit is a trigger bit, if the SEmn bit is "0", the STmn bit is cleared immediately.

The STm register is set via a 16-bit memory operation command.

The low 8 bits of the STm register can be set with STmL and via 8-bit memory operation instructions.

After the reset signal is generated, the value of the STm register changes to “0000H”.

Figure 12-13 Format of serial channel stop register m (STm)

Address: 40041124H (ST0) After reset: 0000HR/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST0													ST03	ST02	ST01	ST00

Address: 40041564H (ST1) After reset: 0000HR/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST1													0	0	ST11	ST10

STmn	Stop trigger for channel n operation
	No triggering.
	Clear the SEmn bit “0” to stop the communication from running.

Note The control register and shift register values, the SCLKmn pin and SDOmn pin, and the FEFmn flag, PEFmn flag, and OVFmn flag hold status.

Note Bit15~4 of the ST0 register and bit15~2 of the ST1 register must be set to “0”.

Note 1.m: unit number (m=0, 1) n: channel number (n=0~3).
2. The read value of the STm register is always “0000H”.

12.3.10 Serial channel enable status register m (SEm)

SEm registers are used to confirm the allowed or stopped status of serial transmits and receivings for each channel.

If "1" is written to each of the serial start allow register m (SSm), the corresponding bit is set to "1". If you write "1" to each bit of the serial channel stop register m (STm), the corresponding bit is cleared to "0".

For channel n that is allowed to run, the value of the CKOmn bit (the serial clock output of channel n) of the serial output register m (SOm) described later cannot be rewritten by software, and the value reflected by the communication operation is output from the serial clock pin.

For a stopped channel n, the value of the CKOmn bit of the SOm register can be set by software and output from the serial clock pin. Thus, an arbitrary waveform such as a start condition or a stop condition can be generated by software.

The SEm register is read via 16-bit memory operation instructions.

The lower 8 bits of the SEm register can be read with SEmL and via 8-bit memory operation instructions.

After the reset signal is generated, the value of the SEm register changes to “0000H”.

Figure 12-14 Format of serial channel enable status register m (SEm)

Address: 40041120H (SE0) After reset: 0000H R

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SE0													SE03	SE02	SE01	SE00

Address: 40041560H (SE1) After reset: 0000H R

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SE1			0										0	0	Herself11	SE10

Sign	Indication of the enable or stop state of channel n operation
	Running stop state
	Run allowed status

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

12.3.11 Serial output enable register m(SOEm)

SOEm register settings allow or stop the output of serial communication for each channel.

For channel n that allows serial output, the value of the SOmn bit of the serial output register m (SOm) described later cannot be rewritten by software, and the value reflected by the communication operation is output from the serial data output pin.

For channel n that stops the serial output, the value of the SOmn bit of the SOm register can be set by software and output from the serial data output pin. Thus, an arbitrary waveform such as a start condition or a stop condition can be generated by software.

The SOEm register is set via a 16-bit memory operation command.

The lower 8 bits of the SOEm register can be set with SOEmL and via 8-bit memory operation instructions.

After the reset signal is generated, the value of the SOEm register changes to “0000H”.

Figure 12-15 Format of serial output enable register m (SOEm)

Address: 4004112AH After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE0			0										SOE 03	SOE 02	SOE 01	SOE 00

Address: 4004156AH After reset: 0000HR/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOE1			0										0	0	SOE 11	SOE 10

SOE mn	Allow or stop of channel n serial output
	Stops the output of serial communication.
	An output that allows serial communication.

Note Bit15~4 of the SOE0 register and bit15~2 of the SOE1 register must be placed “0” .

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

12.3.12 Serial output register m (SOm)

The SOm register is a buffer register for the serial output of each channel.

The value of the SOmn bit of this register is output from the serial data output pin of channel n.

The value of the CKOmn bit of this register is output from the serial clock output pin of channel n.

The SOmn bit of this register can only be rewritten by software when serial output is disabled (SOEmn=0). When serial output (SOEmn=1) is allowed, the value of the SOmn bit of this register can only be changed by serial communication by software overwriting.

The CKOmn bit of this register can only be rewritten by software only when the channel is stopped (SEmn=0). When allowing the channel to run (SEmn=1), the value of the CKOmn bit of this register can only be changed by serial communication by overriding the software.

To use the serial interface pins for non-serial interface functions such as port functions, the corresponding CKOmn bit and SOmn bit must be set to "1".

The SOm register is set via a 16-bit memory operation command.

After the reset signal is generated, the value of the SOm register changes to “0F0FH”.

Figure 12-16 Format of serial output register m (SOm)

Address: 40041128H After reset: 0F0FHR/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO0			0		CKO 03	CKO 02	CKO 01	CKO 00				0	SO 03	SO 02	SO 01	SO 00

Address: 40041568H After reset: 0303HR/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO1		0	0	0	0	0	CKO 11	CKO 10	0			0	0	0	SO 11	SO 10

CKO mn	Serial clock output for channel n
	The output value of the serial clock is “0”.
	The output value of the serial clock is “1”.

SO mn	Serial data output for channel n
	The output value of the serial data is “0”.
	The output value of the serial data is “1”.

Note Bit15~12 and bit7~4 of the SO0 register must be set to “0”.
Bit15~10 and bit7~2 of the SO1 register must be set to “0”.

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

12.3.13 Serial output level register m (SOLm)

The SOLm register is a register that sets the inverting of the data output level of each channel.

This register can be set only in UART mode. In SSPI mode and Simplified I^{2}C mode, the corresponding bit must be set to "0". Only when serial output is allowed (SOEmn=1), the n-inverting setting of each channel of this register is reflected to the pin output. When serial output is disabled (SOEmn=0), the value of the SOmn bit is output directly. It is forbidden to override the SOLm register during operation (SEmn=1).

The SOLm register is set via a 16-bit memory operation command.

The low 8 bits of the SDOLm register can be set with SOLmL and via 8-bit memory operation instructions.

After the reset signal is generated, the value of the SOLm register changes to “0000H”.

Figure 12-17 Format of serial output level register m (SOLm)

Address: 40041134H After reset: 0000HR/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOL0			0											SOL 02		SOL 00

Address: 40041574H After reset: 0000HR/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOL1			0													SOL 10

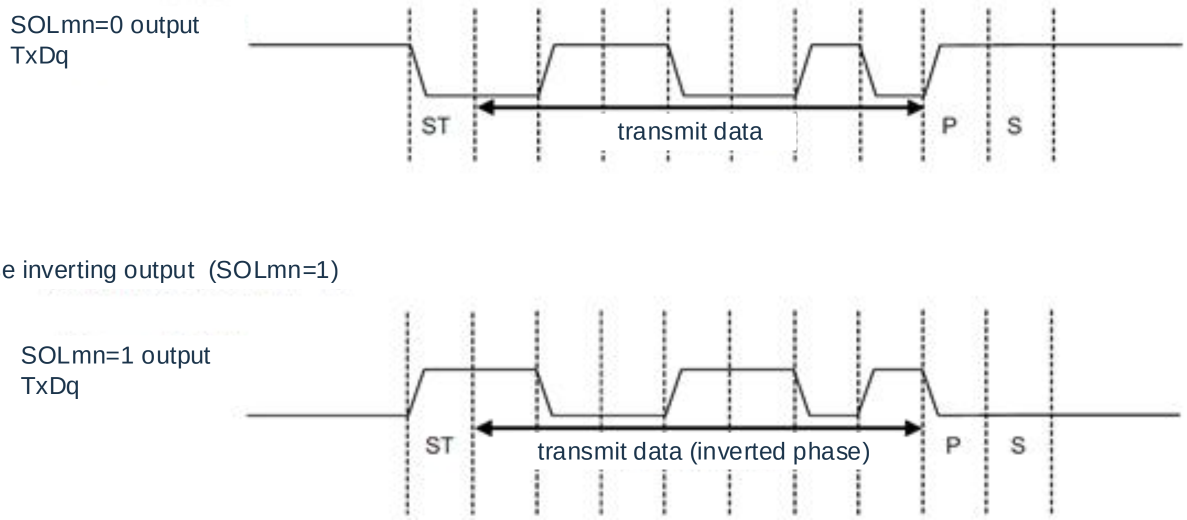
Ground	Selection of channel n transmit data level inversion in UART mode
	The communication data is output directly.
	Inverts the output of communication data.

Notice The bit15 to 3 and bit1 of the SOL0 register and the bit15 to 1 of the SOL1 register must be set to "0".

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

When UART is transmitted, an example of the level inversion of the transmitted data is shown in Figure 12-18.

Figure 12-18 Example of level inversion of transmitted data



(a)Non-inverting output (SOLmn=0)

(b)phase inverting output (SOLmn=1)

Note m: unit number (m=0, 1) n: channel number (n=0, 2).

12.3.14 Input switching control register (ISC)

When implementing LIN-bus communication via UART0, the ISC1 and ISC0 bits of the ISC register are used for the coordination of external interrupts and timer array units. If bit0 is set to “1”, the input signal from the serial data input (RxD0) pin is selected as the input for the external interrupt (INTP0) and therefore passes The INTP0 interrupt detects the wake-up signal.

If bit1 is set to “1”, the input signal from the serial data input (RxD0) pin is selected as the input to the timer, so that the wake-up signal can be detected by the timer and the low-level width of the interval segment and the pulse width of the synchronization segment can be measured.

The SS1E00 bit controls the SS00 pin input of channel 0 in slave mode of SSPI00 communication. During the period when the SS00 pin is input high, no transmission and reception occurs even if the serial clock is input; During the low input level to the SS00 pin, if a serial clock is entered, it is transmitted and received according to the settings of each mode.

The ISC register is set via an 8-bit memory operation command.

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

Figure 12-19 Format of input switching control register (ISC)

Address: 40040473H After reset: 00HR/W

Symbol	7	6	5	4	3	2	1	0
ISC	SYESE00	0	0	0	0	0	ISC1	ISC0

SSDIE00	Setting of SS00 input of channel 0 in slave mode for SSPI00 communication
0	The SS00 pin input is invalid.
1	The SS00 pin input is valid.

ISC1	Input switching of channel 3 of timer Timer4
0	Use the input signal from the TI03 pin as the input to the timer (usually running).
1	Use the input signal from the RxD0 pin as the input to the timer (detects the wake-up signal and measures the low-level width of the interval and the pulse width of the sync field).

ISC0	Input switching for external interrupt (INTP0)
0	Use the input signal from the INTP0 pin as the input for an external interrupt (usually in operation).
1	Use the input signal from the RxD0 pin as the input for the external interrupt (detect wake-up signal).

Note Bit6~0 must be set to “0”.

12.3.15 Noise filter enable register 0 (NFEN0)

The NFEN0 register sets whether the noise filter is used for the input signal of the serial data input pins of each channel.

For pins used for SSPI or simple I^{2}C communication, the corresponding bit must be “0” to invalidate the noise filter. For the pins used for UART communication, the corresponding bit must be set to "1" to make the noise filter active.

When the noise filter is active, detect whether the two clocks are consistent after synchronization through the running clock (f_{MCK}) of the object channel; When the noise filter is invalid, synchronization is performed only through the running clock (f_{MCK}) of the object channel.

The NFEN0 register is set via an 8-bit memory operation command.

After the reset signal is generated, the value of the NFEN0 register changes to “00H”.

Figure 12-20 Format of noise filter enable register 0 (NFEN0)

Address: 40040470H After reset: 00HR/W

Symbol	7	6	5	4	3	2	1	0
NFEN0	0	0	0	SNFEN20	0	SNFEN10	0	SNFEN00

SNFEN20	RxD2 pin noise filter is used or not
0	Noise filter OFF
1	Noise filter ON
When used as the RxD2 pin, SNFEN20 must be set to "1". When used as a function other than the RxD2 pin, the SNFEN20 must be set to "0".	

SNFEN10	RxD1 pin noise filter is used or not
0	Noise filter OFF
1	Noise filter ON
When used as the RxD1 pin, SNFEN10 must be set to "1". When used as a function other than the RxD1 pin, the SNFEN10 must be set to "0".	

SNFEN00	RxD0 pin noise filter is used or not
0	Noise filter OFF
1	Noise filter ON
When used as the RxD0 pin, SNFEN00 must be set to "1". When used as a function other than the RxD0 pin, the SNFEN00 must be set to "0".	

Note Bit7~5, 3, and 1 must be set to “0”.

12.3.16 Registers controlling the function of the serial input/output pin port

When using a general-purpose serial communication unit, the control registers for the multiplexed port function (Port Mode Register (PMxx), Port Multiplexing Function Configuration Register (PxxCFG), Port Output Mode Register (POMxx), and Port Mode Control Register must be set (PMCxx)).

For details, please refer to “Chapter 2 Pin Functions”.

When using the multiplexed port of the serial data output pin or the serial clock output pin as the serial data output or serial clock output, the bits of the corresponding port mode control register (PMCxx) and the bit of the port mode register (PMxx) corresponding to each port are “0”. In this case, the bit of the port register (Pxx) can be “0” or “1”.

In addition, when used for N-channel open-drain output mode, the bit of the port output mode register (POMxx) corresponding to each port must be “1”.

When using the multiplexed port of the serial data input pin or serial clock input pin as serial data input or serial clock input, you must set the bit of the Port Mode Register (PMxx) corresponding to each port to "1" and set the bit of the Port Mode Control Register (PMCxx) to "0". In this case, the Port Register (Pxx) bits can be "0" or "1".

12.4 Operation stop mode

Each serial interface of the universal serial communication unit has a stop-and-run mode. Serial communication is not possible in run-stop mode, so power consumption can be reduced. In addition, pins for the serial interface can be used as port functions in run-stop mode.

12.4.1 Stopping the operation by units

Peripheral allows register 0 (PER0) to set stop operation in units.

The PER0 register is a register that sets the clock to be allowed or disallowed to be supplied to each peripheral hardware. Reduce power consumption and noise by providing a clock to hardware that is not in use.

To stop universal serial communication unit 0, bit2 (SCI0EN) must be set to “0”; To stop universal serial communication unit 1, bit3 (SCI1EN) must be set to “0”.

Figure 12-21 Setting of peripheral allowable register 0 (PER0) when stopping operation by unit

(a) Peripheral Enable Register 0 (PER0).... Only the corresponding bit of SCIm to be stopped is set to "0".

7 6 5 4 3 2 1
0

PER0	RTCEN ×	IRDAEN ×	ADCEN ×	IICA0EN ×	SCI1EN 0/1	SCI0EN 0/1	TM41EN ×	TM40EN ×

SCIm input clock control
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 all initial. However, the following registers are excluded:

- Input switch control register (ISC).
- Noise filter enable register 0 (NFEN0).
- Port multiplexing function configuration register (PxxCFG).
- Port output mode register (POMx).
- Port mode register (PMx).
- Port register (Px).

Note ×: This is an unused bit in the universal serial communication unit (depending on the settings of other peripheral functions).

0/1: Set “0” or “1” according to the user's purpose.

12.4.2 Stopping the operation by channels

Stop-operation by channel is set by each of the following registers.

Figure 12-22 Setting of each register when stopping the operation by channels

(a) Serial channel stop register m (STm)... This is the register that sets the communication/stop count allowed 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			STm1 0/1	STm0 0/1

1: Clear the SEmn bit "0" and stop the communication operation

※Because the STmn bit is the trigger bit, the STmn bit is immediately cleared if the SEmn bit is "0".

(b) The serial channel allows status register m (SEm)... This register represents the running or stopped state of data transmission and reception for each channel.

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

STm	0	0	0	0	0	0	0	0	0	0	0	0	SEm3Note 0/1	SEm2Note 0/1	Herselfm1 0/1	SEm0 0/1

0: Running stop state

※ The SEm register is a read-only status register that stops operation through the STm register. For channels that have stopped running, the value of the CKOmn bit of the SOm register can be set by software.

(c) The serial output allows register m (SOEm)... This is the register that sets the serial communication output that allows or stops each channel.

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

SOEm	0	0	0	0	0	0	0	0	0	0	0	0	SOEm3Note 0/1	SOEm2Note 0/1	SOEm1 0/1	SOEm0 0/1

0: Stops the output by running through serial communication

※For channels that have stopped the 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

As	0	0	0	0	CKOm3Note 0/1	CKOm2Note 0/1	CKOm1 0/1	CKOm0 0/1	0	0	0	0	SOm3Note 0/1	SOm2Note 0/1	SOm1 0/1	SOm0 0/1

1: The output value of the serial clock is "1"1: the output value of the serial data is "1"

※When using the corresponding pin for each channel as a port function, the corresponding CKOmn bit and SOmn bit must be set to "1".

Note Limited to universal serial communication unit 0 only.

Note 1.m: Unit number (m=0, 1) n: channel number (n=0~3).
2. ▨ : Cannot be set (set initial value). 0/1: Set "0" or "1" according to the user's purpose.

12.5 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) communication

This is a clock synchronization communication function implemented by a total of 3 wires of serial clock (SCLK) and serial data (SDI and SDO).

[Transmit and receive data]

- 7-bit or 8-bit data length
- Phase control of sending and receiving data
- MSB/LSB preferred choice

[Clock control]

- Master or slave selection
- Phase control of input/output clocks
- Sets the transfer period generated by the prescaler and the in-channel counter.
- Maximum transfer rate Note

Master communication: Max.f_{CLK}/2 (SSPI00 only).

Master communication: Max.f_{CLK}/4

Slave communication: Max.f_{MCK}/6

[Interrupt function]

- End of transfer interrupt, buffer empty interrupt

[Error detection flag]

- Overflow error

Note It must be used within the scope of the SCLK Cycle Time (t_{KCY}) characteristics. Please refer to the data sheet for details.

Channels 0 to 3 of SCI0 and channels 0 to 1 of SCI1 support 3-wire serial I/O SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) channels.

3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) has the following 6 types of communication operation:

- Master transmission (see 12.5.1).
- Master reception (see 12.5.2).
- Master transmission and reception (refer to 12.5.3).
- Slave transmission (see 12.5.4).
- Slave reception (see 12.5.5).
- Slave transmission and reception (see 12.5.6).

12.5.1 Master transmission

Master transmission refers to the operation of this product output transmission clock and sending data to other devices.

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channel	Channel 0 of SCI0	Channel 1 of SCI0	Channel 2 of SCI0	Channel 3 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1
The pin used	SCLK00, SDO00	SCLK01, SDO01	SCLK10, SDO10	SCLK11, SDO11	SCLK20, SDO20	SCLK21, SDO21
interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21
	You can select either a transmit-end interrupt (single-pass mode) or a buffer null interrupt (continuous transfer mode).					
Error detection flag	not					
The length of the transmitted data	7 or 8 bits					
Transfer Rate Note	Max.f_{CLK}/2[Hz] (SSPI00 only), f_{CLK}/4[Hz] Min.f CLK/(22^{15}×128) [Hz]f×$_{CLK}$: System clock frequency					
Data phase	It can be selected via the DAPmn bit of the SCRmn register. • DAPmn=0: The data output starts when the serial clock starts running. • DAPmn=1: Starts data output half a clock before the serial clock starts running.					
Clock phase	It can be selected via the CKPmn bit of the SCRmn register. • CKPmn=0: Normal • CKPmn=1: Inverted					
Data direction	MSB first or LSB first					

Note It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remarks m: Unit number (m=0, 1) n: Channel number (n=0~3)mn=00~ 03, 10~11

(1) Register setting

Figure 12-23 3 wire serial I/O(SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)
Example of register settings when the master is transmitted

(a) serial mode register mn (SMRmn)

	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 0	DAPmn 0/1	CKPmn 0/1	0	EOCmn 0	PTCmn1 0	PTCmn0 0	DIRmn 0/1	0	SLCmn1 0	SLCmn0 0	0	1	DLSmn1 1 注	DLSmn0 0/1

data and clock phase selection (details refer to "19.3 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
0: 7 bit data length
1: 8 bit data length

(c) serial data registermn mn(SDRmn)

	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	transmit data (configuration of transmit data)							
									SIOp							

(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 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 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, SCR01 register, others fixed as “1”

Note 1.m: Unit number (m=0, 1) n: channel number (n=0~3)mn=00~ 03, 10~11

2. ▨ : Cannot be set (set initial value). 0/1: Set “0” or “1” according to the user's purpose.

(2) Operation Steps

Figure 12-24 Initial setup step of the master transmission

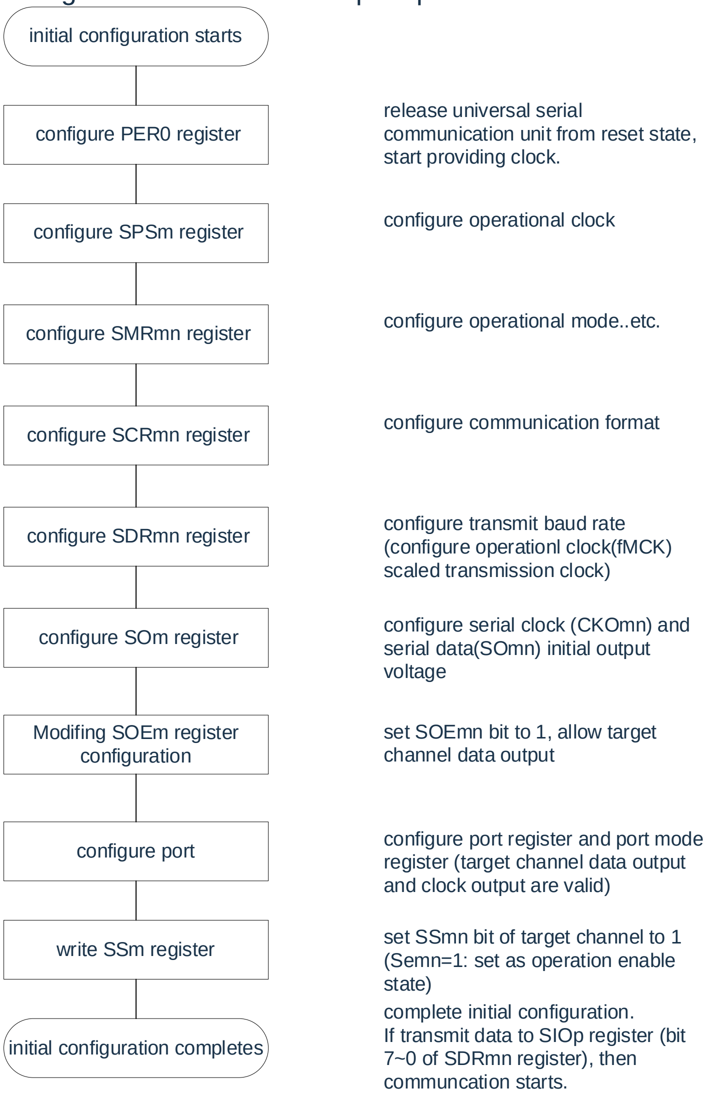
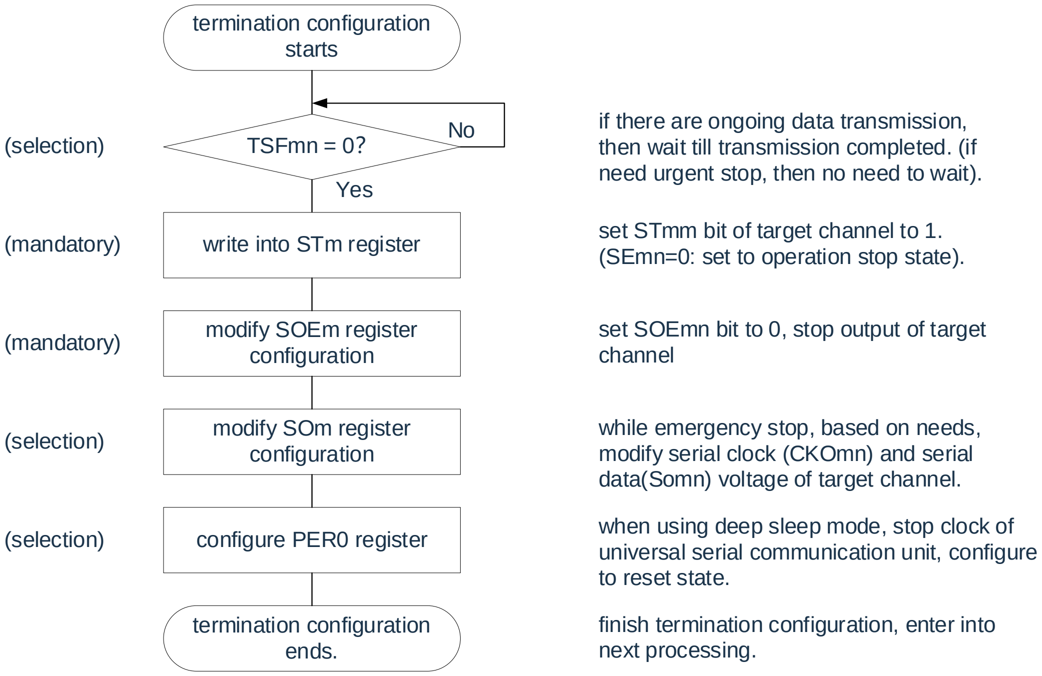

initial configuration starts
configure PER0 register
release universal serial communication unit from reset state, start providing clock.
configure SPSm register
configure operational clock
configure SMRmn register
configure operational mode..etc.
configure SCRmn register
configure communication format
configure SDRmn register
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 SIOp register (bit 7~0 of SDRmn register), then communcation starts.

Figure 12-25 Master send abort step


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).

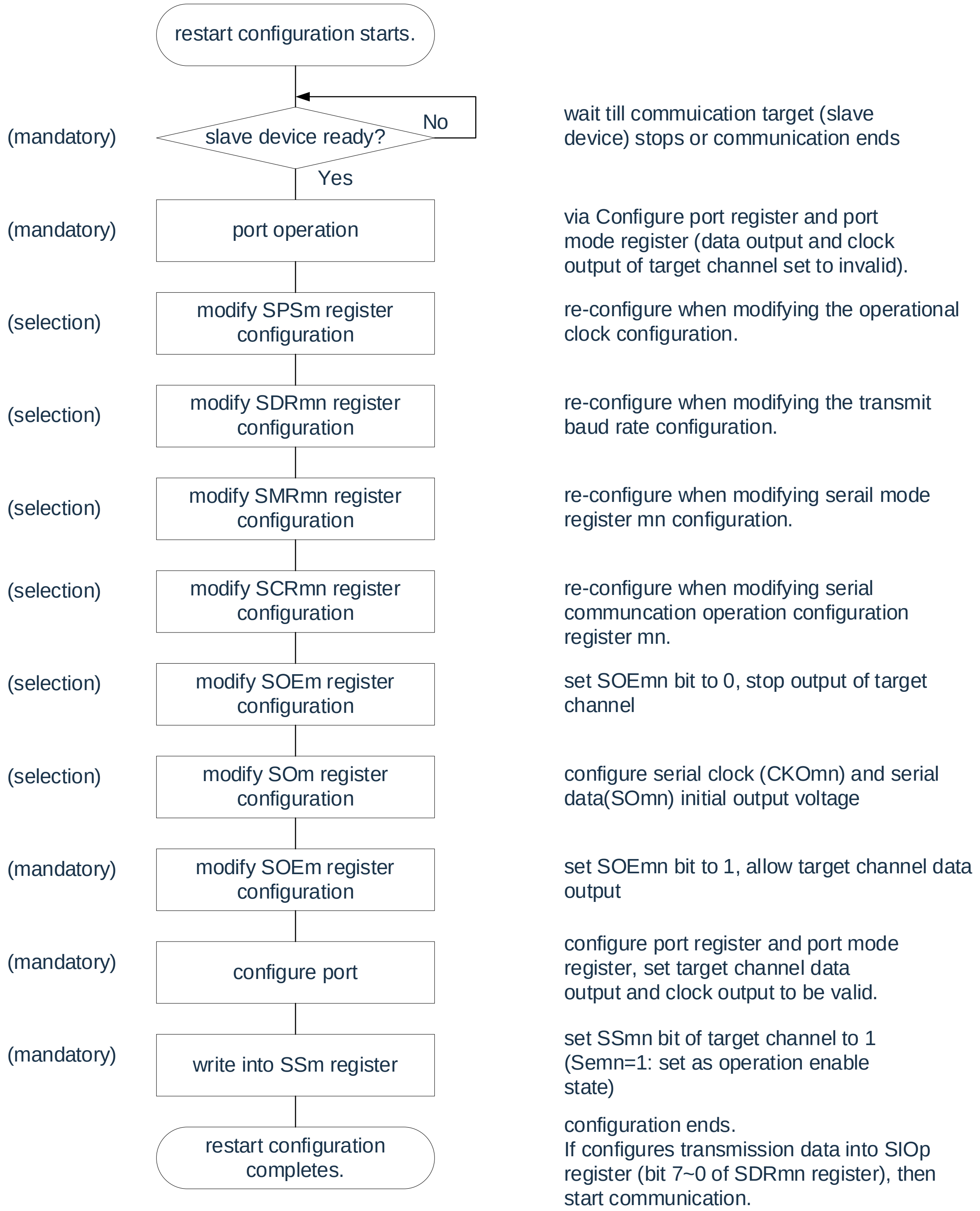
set SOEmn bit to 0, stop output of target channel

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 12-26 Restart step of the master transmission


restart configuration starts.
(mandatory)
slide device ready?
No
Yes
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)
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
(mandatory)
modify SOEm register configuration
set SOEmn bit to 1, allow target channel data output
(mandatory)
configure port
configure port register and port mode register, set target channel data output and clock output to be valid.
(mandatory)
write into SSm register
set SSmn bit of target channel to 1 (Semn=1: set as operation enable state)
restart configuration completes.
configuration ends.
If configures transmission data into SIOp register (bit 7~0 of SDRmn register), then start communication.

Note If you override PER0 in the abort setting to stop the clock, you must wait until the communication object (slave device) stops or the communication is over to make the initial setting instead of starting the setting again.

(3) Process flow (single-transmit mode).

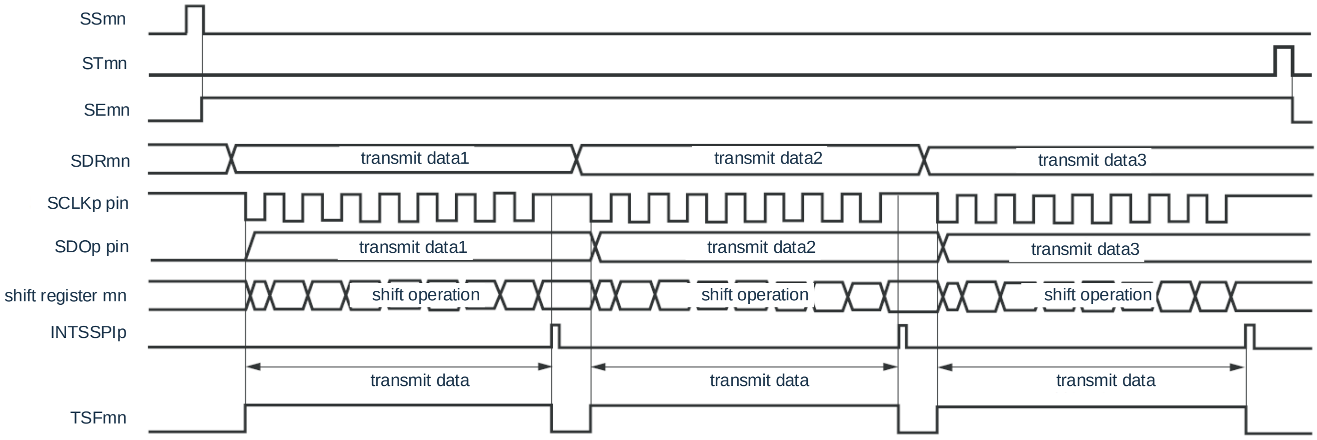
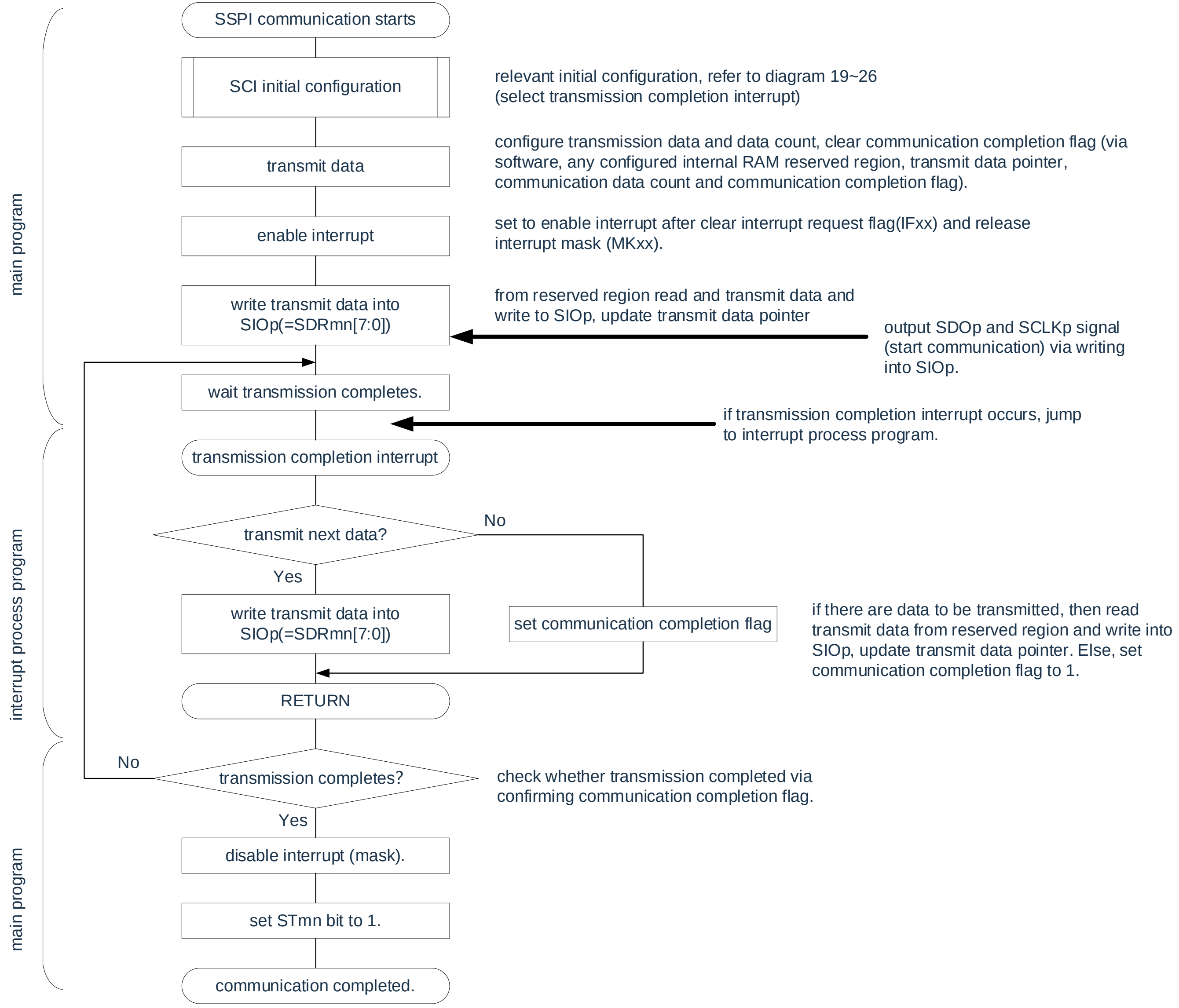


Figure 12-27 Timing diagram of the main transmission (single-pass transmission mode) (type 1: DAPmn=0, CKPmn=0).

Remark m: Unit number (m=0, 1) n: Channel number (n=0~3)p: SSPI number (p=00, 01, 10, 11, 20, 21.))

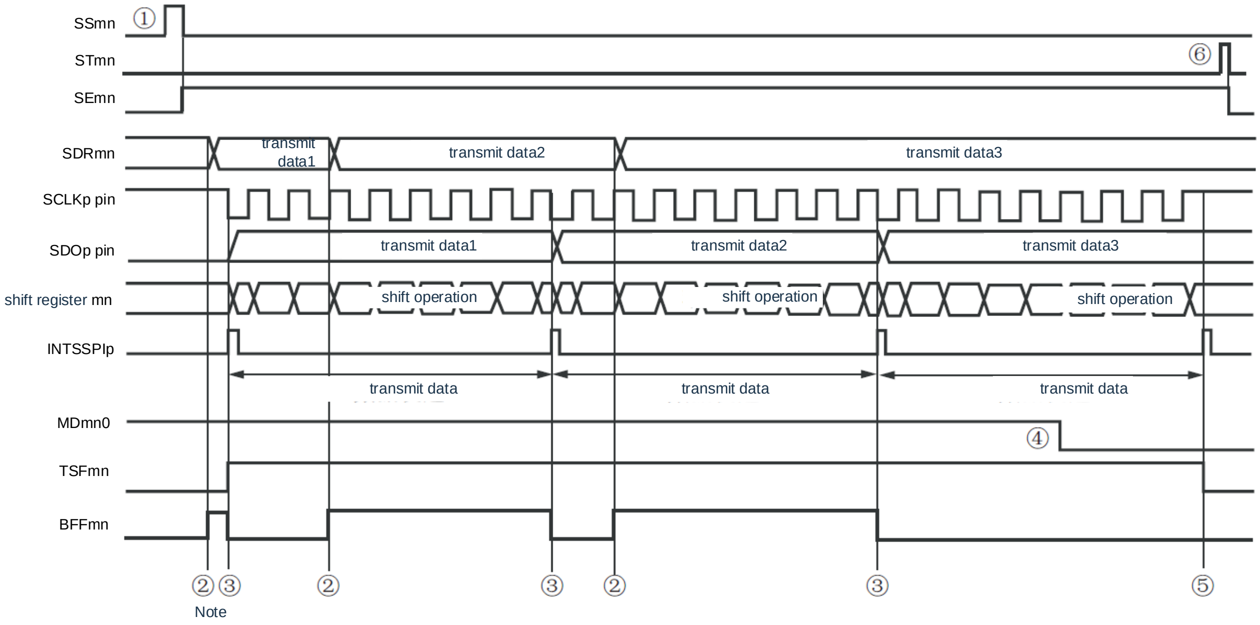
mn=00~03, 10~11

Figure 12-28 Flow of master transmission (single transmit mode)


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.
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
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 12-29 Timing diagram of the master transmission (continuous transmit mode) (type 1: DAPmn=0, CKPmn=0)

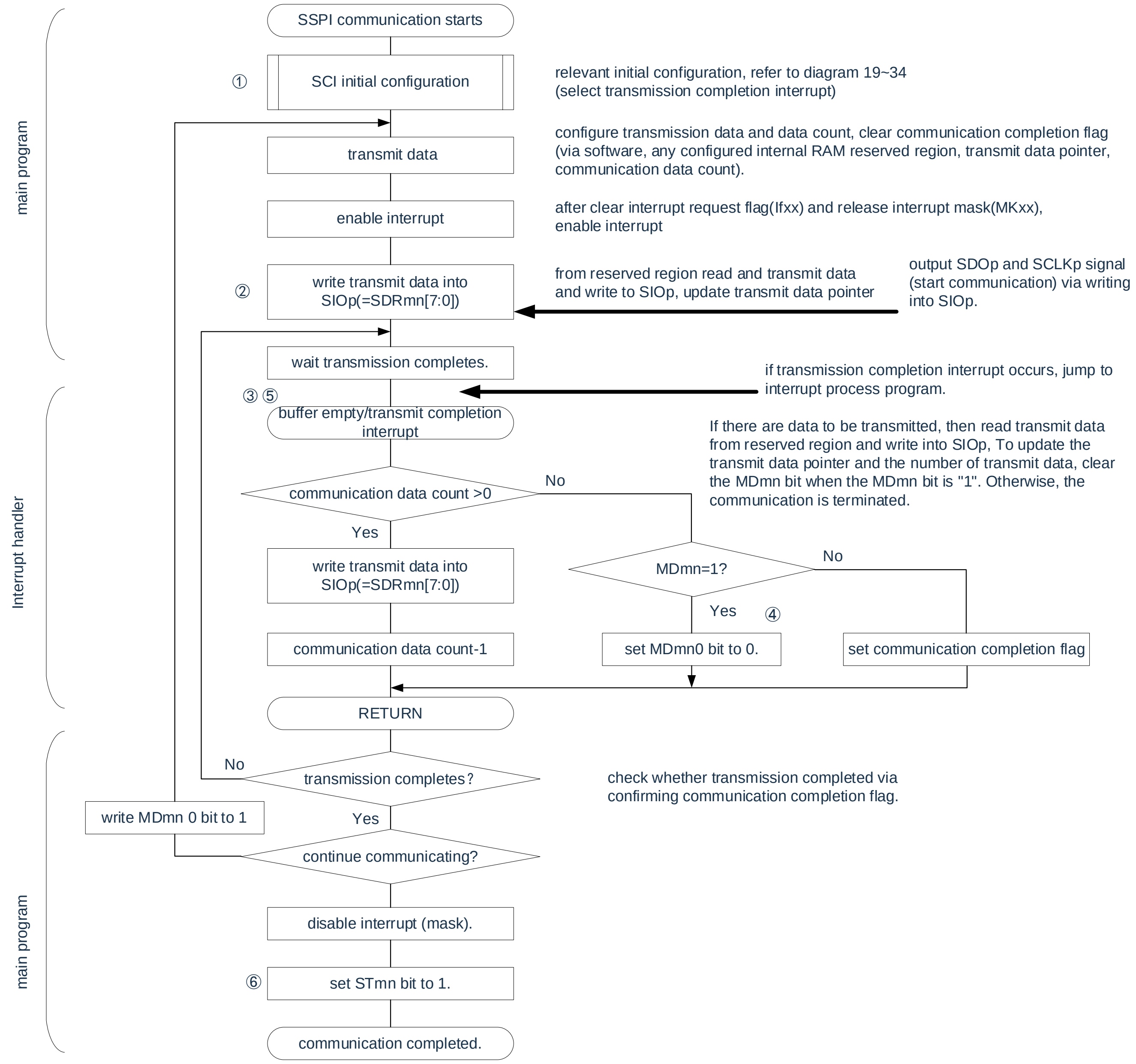


Note If the BFFmn bit of the serial status register mn (SSRmn) is “1” (when valid data is saved in the serial data register mn (SDRmn)) is given The SDRmn register writes the transmitted data and overrides the transmitted data.

Note that the MDmn0 bit of the serial mode register mn (SMRmn) can be overridden even during operation. However, in order to catch up with the end of the transmission interruption of the last transmitted data, it must be overwritten before the last bit of transmission begins.

Remarks m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21.))
mn=00~03, 10~11

Figure 12-30 Flowchart of the master transmission (continuous transmit mode)


main program
Interrupt handler
main program
SSPI communication starts
①
SCI initial configuration
transmit data
enable interrupt
②
write transmit data into SIOp(=SDRmn[7:0])
wait transmission completes.
③ ⑤
buffer empty/transmit completion interrupt
communication data count >0
No
Yes
write transmit data into SIOp(=SDRmn[7:0])
communication data count-1
MDmn=1?
No
Yes
④
set MDmn0 bit to 0.
set communication completion flag
RETURN
No
transmission completes?
Yes
write MDmn 0 bit to 1
continue communicating?
disable interrupt (mask).
⑥
set STmn bit to 1.
communication completed.

relevant initial configuration, refer to diagram 19~34 (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

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.

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 SIOp, 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) to (6) in the note figure corresponds to (1) to (6) in the “Figure 12-29“.

12.5.2 Master reception

Master reception refers to the operation of this product output transmission clock and receiving data from other devices.

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channel	Channel 0 of SCI0	Channel 1 of SCI0	Channel 2 of SCI0	Channel 3 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1
The pin used	SCLK00, SDO00	SCLK01, SDO01	SCLK10, SDO10	SCLK11, SDO11	SCLK20, SDO20	SCLK21, SDO21
interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21
	You can choose between an end-of-the-transmission interrupt (single-pass mode) or a buffer null interrupt (continuous transfer mode).					
Error detection flag	Only the overflow error detection flag (OVFmn)					
The length of the transmitted data	7 or 8 bits					
Transmission rate note	Max. f_{CLK}/2[Hz](Only.)SSPI00)， f_{CLK}/4[Hz] Mi n.$f_{CLK}/(2\times2^{15}\times128)$[Hz]$f_{CLK}$: System clock frequency					
Data phase	It can be selected via the DAPmn bit of the SCRmn register. • DAPmn=0: Data output starts when the serial clock starts running. • DAPmn=1: Starts data output half a clock before the serial clock starts running.					
Clock phase	It can be selected by the CKPmn bit of the SCRmn register. • CKPmn=0: Positive phase • CKPmn=1: Inverted phase					
Data direction	MSB preferred or LSB preferred					

Note It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11.

(1) Register setting

Figure 12-31 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)
Example of register setting content when the master receives

(a) serial mode register mn(SMRmn)

	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 0	RXEmn 1	DAPmn 0/1	CKPmn 0/1	0	EOCmn 0	PTCmn1 0	PTCmn0 0	DIRmn 0/1	0	SLCmn1 0	SLCmn0 0	0	1	DLSmn1 1 注	DLSmn0 0/1

data and clock phase selection (details refer to "19.3 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
0: 7 bit data length
1: 8 bit data length

(c) serial data register mn(SDRmn) (low 8 bit: SIOp)

	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	received data write virtual data "FFH"							

SIOp

(d) serial output registerm (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 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 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

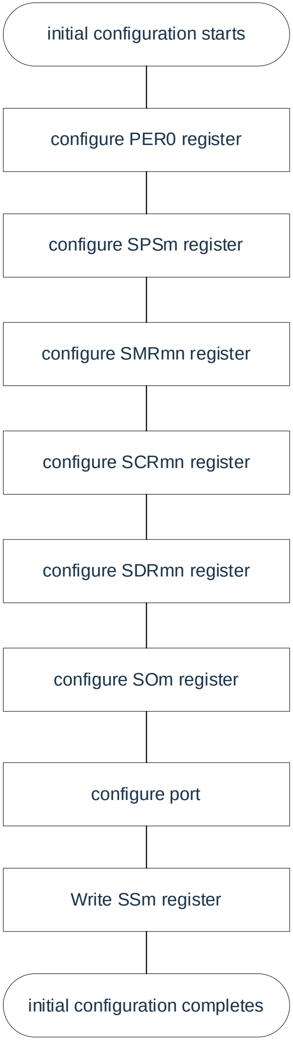
Note Limited to SCR00 register and SCR01 register, the others are fixed as "1".

Note 1.m: Unit number (m=0, 1) n: Channel number (n=0~3)p: SSPI number (p=00, 01, 10, 11, 20, 21.))
mn=00~03, 10~11

2. □ : Fixed in SSPI master receive mode. ▨ : Cannot be set (initial value).
×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).
0/1: Set "0" or "1" according to the user's purpose.

(2) Operation Steps

Figure 12-32 Initial setup step of 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)

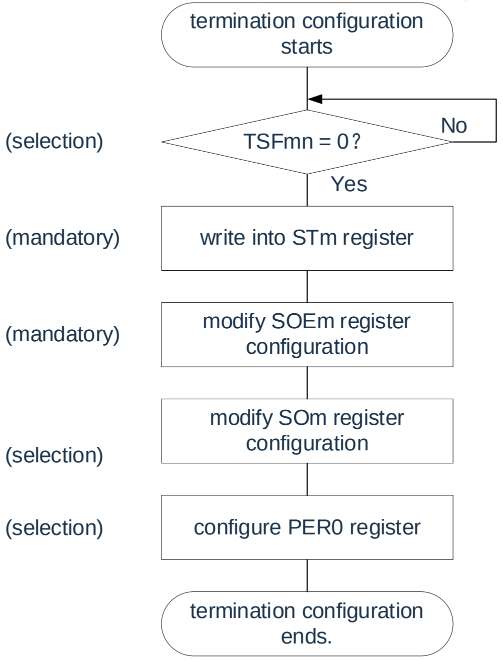
configure serial clock (CKOmn) initial output voltage

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 SIOp register (bit7~0 of the SDRmn register).

Figure 12-33 Stop steps of master 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).

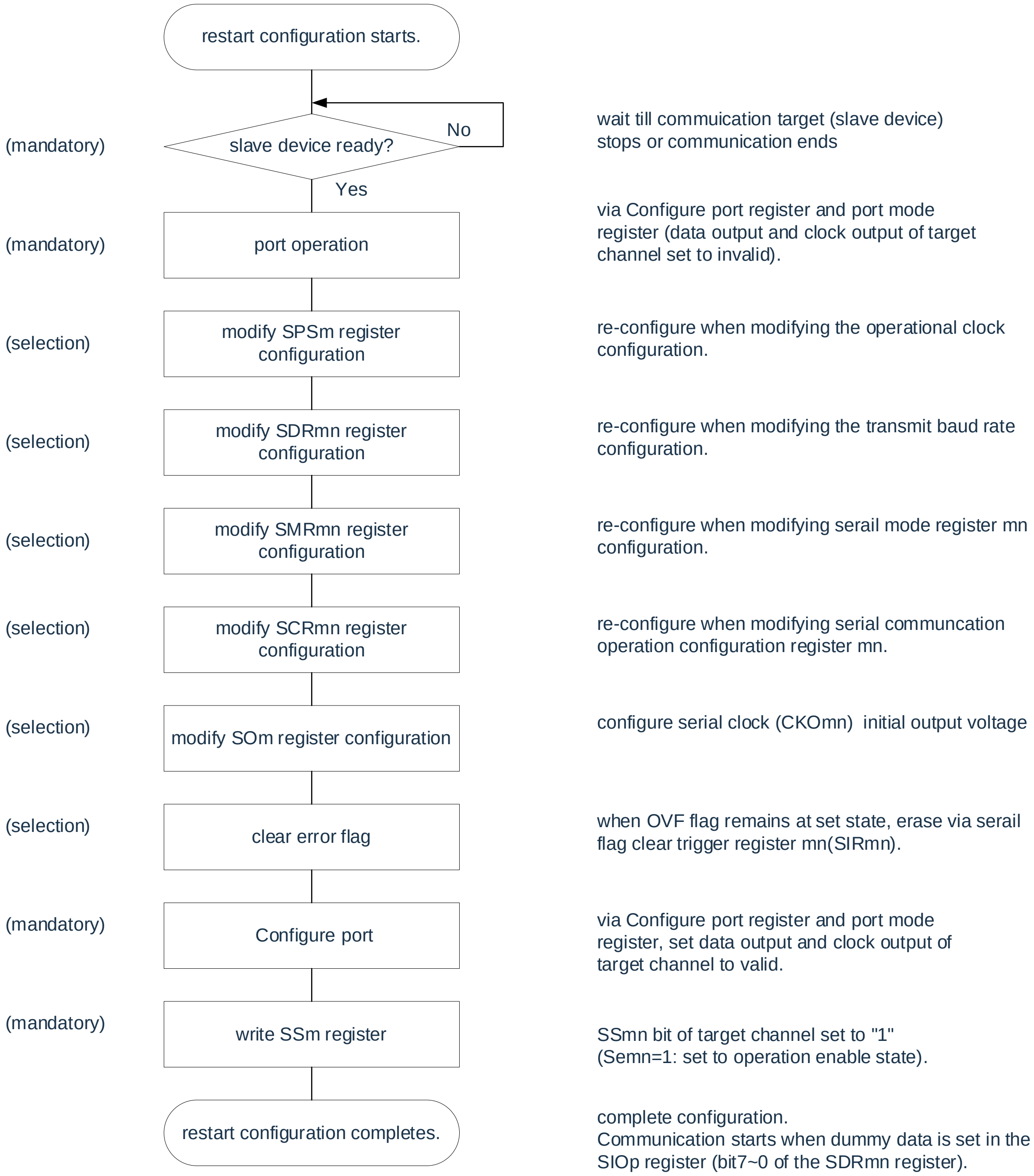
set SOEmn bit to 0, stop output of target channel

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.

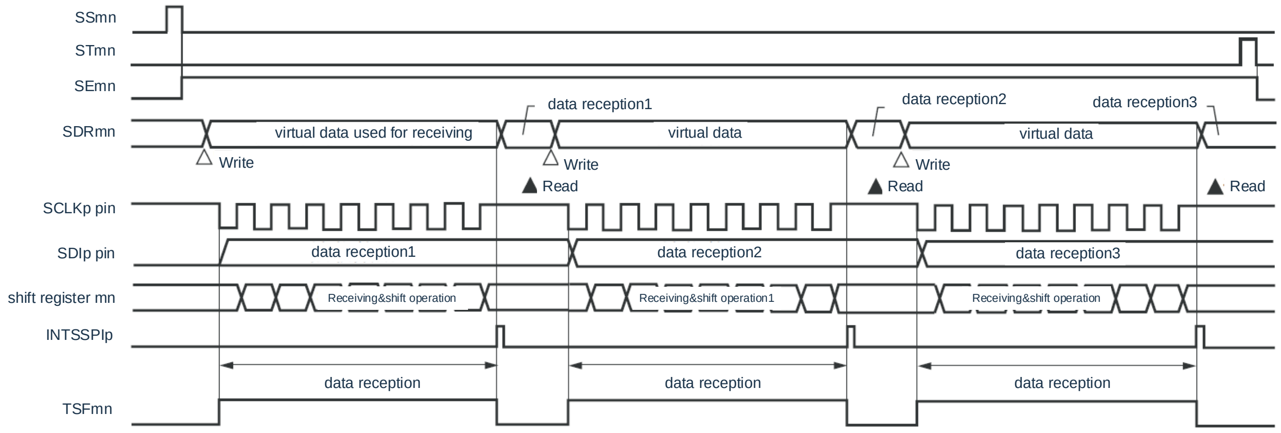
Figure 12-34 Restart step of master reception


restart configuration starts.
(mandatory)
slave device ready?
No
Yes
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)
modify SOm register configuration
configure serial clock (CKOmn) initial output voltage
(selection)
clear error flag
when OVF flag remains at set state, erase via serail flag clear trigger register mn(SIRmn).
(mandatory)
Configure port
via Configure port register and port mode register, set data output and clock output of target channel to valid.
(mandatory)
write SSm register
SSmn bit of target channel set to "1" (Semn=1: set to operation enable state).
restart configuration completes.
complete configuration. Communication starts when dummy data is set in the SIOp register (bit7~0 of the SDRmn register).

Note: If you override PER0 in the abort setting to stop the clock, you must wait until the communication object (slave device) stops or the communication is over to make the initial setting instead of starting the setting again.

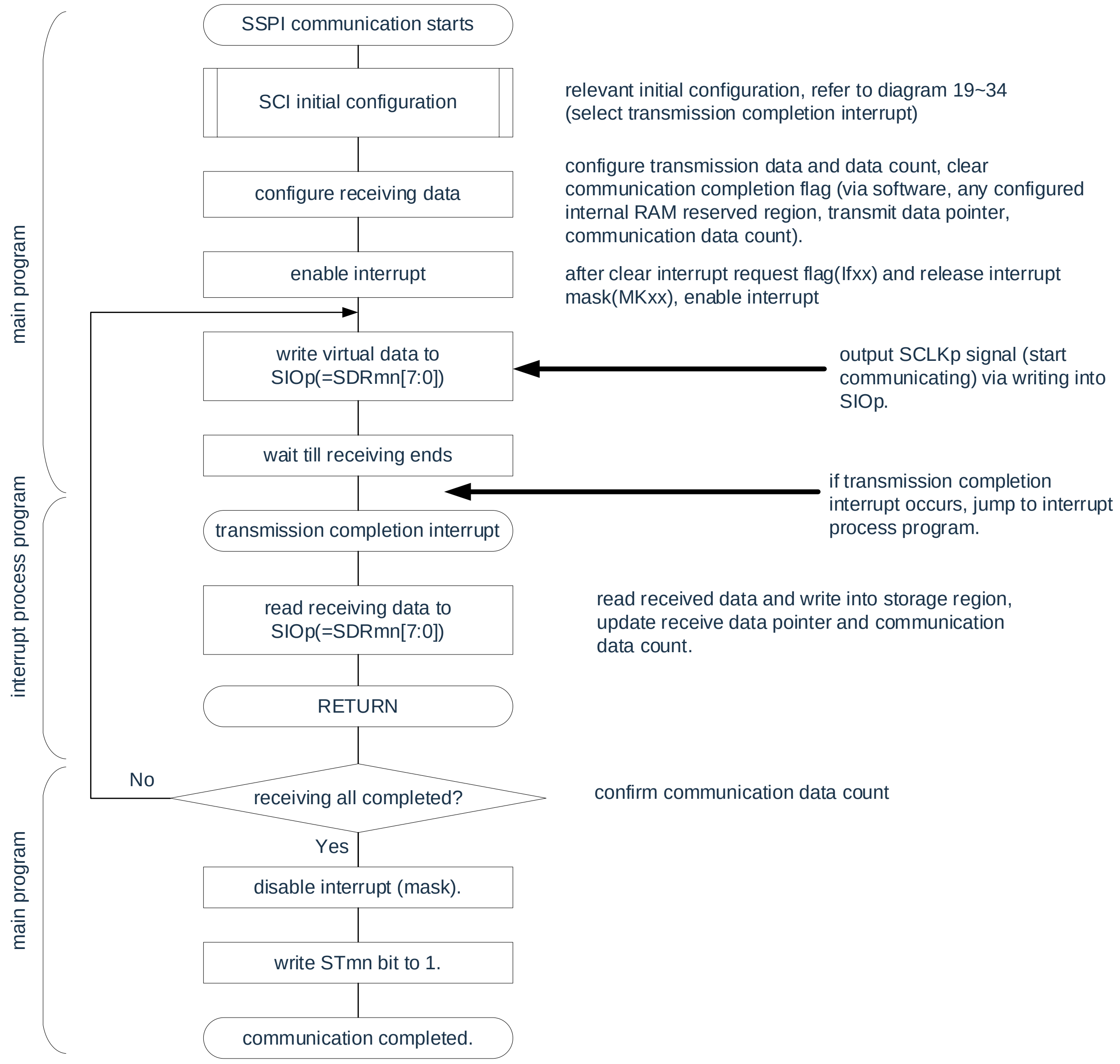
(3) Process flow (single receive mode).

Figure 12-35 Timing diagram of the master receive (single receive mode) (type 1: DAPmn=0, CKPmn=0).



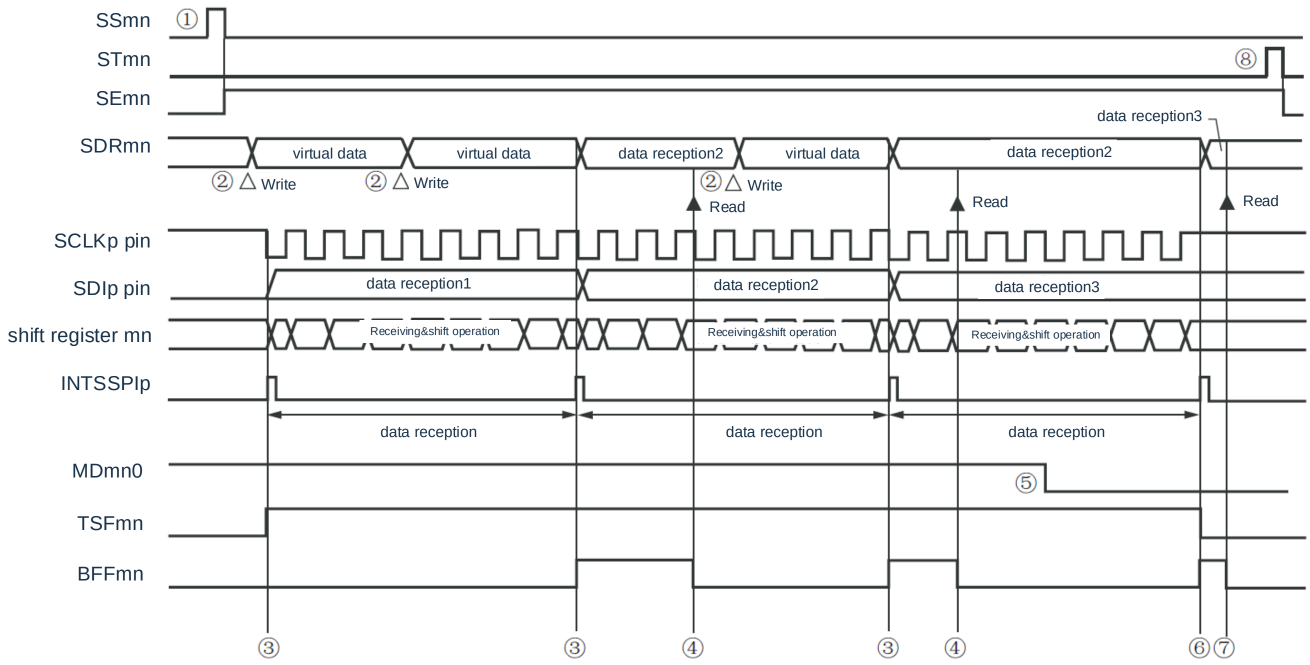
Remark m: Unit number (m=0, 1) n: Channel number (n=0~3)p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11

Figure 12-36 Flowchart of the master receive (single receive mode)


main program
SSPI communication starts
SCI initial configuration
relevant initial configuration, refer to diagram 19~34
(select transmission completion interrupt)
configure receiving 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).
enable interrupt
after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt
write virtual data to SIOp(=SDRmn[7:0])
output SCLKp signal (start communicating) via writing into SIOp.
wait till receiving ends
if transmission completion interrupt occurs, jump to interrupt process program.
interrupt process program
transmission completion interrupt
read receiving data to SIOp(=SDRmn[7:0])
read received data and write into storage region, update receive data pointer and communication data count.
RETURN
No
receiving all completed?
confirm communication data count
Yes
main program
disable interrupt (mask).
write STmn bit to 1.
communication completed.

(4) Process flow (continuous receive mode).

Figure 12-37 Timing diagram of the master receive (continuous receive mode) (type 1: DAPmn=0, CKPmn=0).



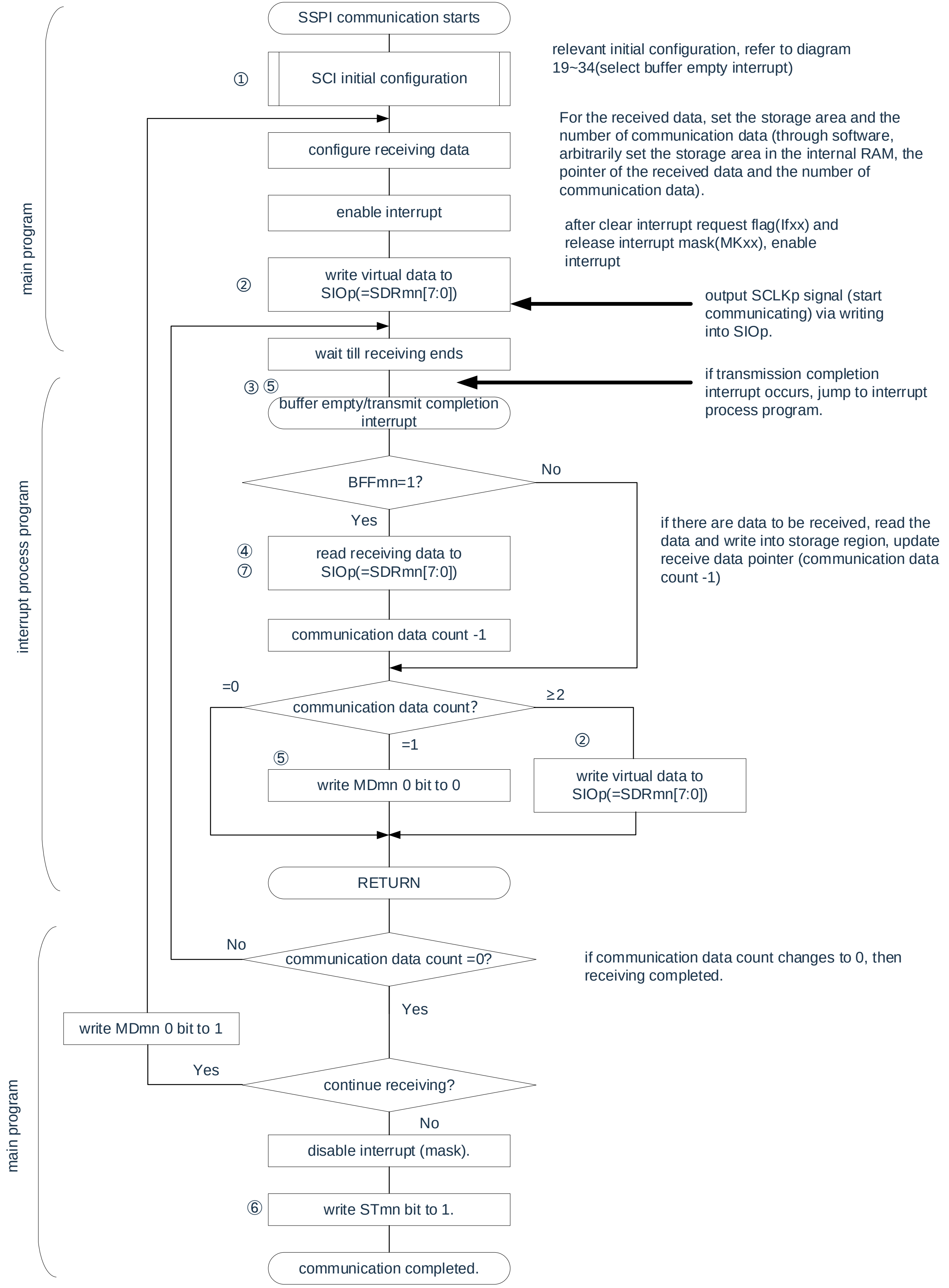
Note The MDmn0 bit can be overridden even during operation. However, in order to catch up with the end of transmission interruption of the last received data, it must be overridden before the last bit of reception begins.

Note 1 (1) ~ (8) in the figure corresponds to (1) ~ (8) in the “flowchart of theFigure 12-381238”.

2.m: Unit number (m=0, 1)n: Channel number (n=0~3)p: SSPI number (p=00, 01, 10, 11, 20, 21)

mn=00~03, 10~11

Figure 12-38 Flowchart of the master receive (continuous receive mode)



main program
SSPI communication starts
①
SCI initial configuration
relevant initial configuration, refer to diagram 19~34(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 SIOp(=SDRmn[7:0])
output SCLKp signal (start communicating) via writing into SIOp.
wait till receiving ends
interrupt process program
③ ⑤
buffer empty/transmit completion interrupt
if transmission completion interrupt occurs, jump to interrupt process program.
BFFmn=1?
No
Yes
④
⑦
read receiving data to SIOp(=SDRmn[7:0])
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 SIOp(=SDRmn[7:0])
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.

(1) to (8) in the note figure corresponds to (1) to (8) in the “Figure 12-37“.

12.5.3 Master transmission and reception

The transmission and reception of the master refers to the operation of the output transmission clock of this product and the transmission and reception of data with other devices.

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channel	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1
The 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
	You can choose between an end-of-the-transmission interrupt (single-pass mode) or a buffer null interrupt (continuous transfer mode).					
Error detection flag	Only the overflow error detection flag (OVFmn)					
The length of the transmitted data	7 or 8 bits					
Transfer Rate Note	Max. f_{CLK}/2[Hz](Only.)SSPI00)f_{CLK}/4[Hz] Mi n.f_{CLK}/(2×2^{15}×128)[Hz]f_{CLK}: System clock frequency					
Data phase	It can be selected via the DAPmn bit of the SCRmn register. • DAPmn=0: Data output starts when the serial clock starts running. • DAPmn=1: Starts data output half a clock before the serial clock starts running.					
Clock phase	It can be selected by the CKPmn bit of the SCRmn register. • CKPmn=0: Positive phase • CKPmn=1: Inverted phase					
Data direction	MSB preferred or LSB preferred					

Note It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark m: Unit number (m=0, 1) n: Channel number (n=0~3)p: SSPI number (p=00, 01, 10, 11, 20, 21.))
mn=00~03, 10~11

(1) Register setting

Figure 12-39 3-wire serial I/O(SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)
Example of register settings when the master transmits and receives

(a) serial mode register mn (SMRmn)

	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	0	1	DLSmn1 1 Note	DLSmn0 0/1

data and clock phase selection (details refer to "19.3 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
0: 7 bit data length
1: 8 bit data length

(c) serial data regsiter mn (SDRmn) (low 8 bit: SIOp)

	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	configuration of transmit data/received data register							
									SIOp							

(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 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 Limited to SCR00 register and SCR01 register, the others are fixed as "1".

Note 1.m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11
2. □: Fixed in SSPI master T&R modes. ▨ : Cannot be set (initial value).
0/1: Set "0" or "1" according to the user's purpose.

(2) Operation Steps

Figure 12-40 Initial setup step of master transmit and receive

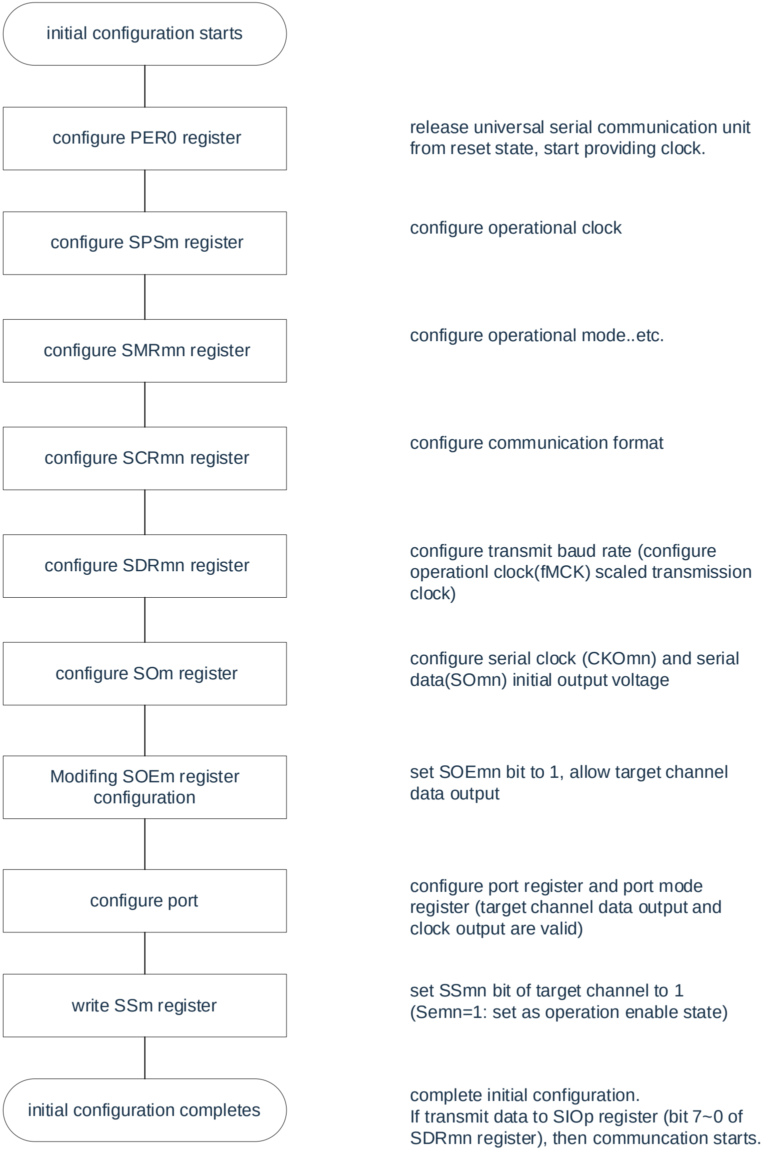

initial configuration starts
configure PER0 register
release universal serial communication unit from reset state, start providing clock.
configure SPSm register
configure operational clock
configure SMRmn register
configure operational mode..etc.
configure SCRmn register
configure communication format
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 SIOp register (bit 7~0 of SDRmn register), then communcation starts.

Figure 12-41 Stop step of the master transmit and receive

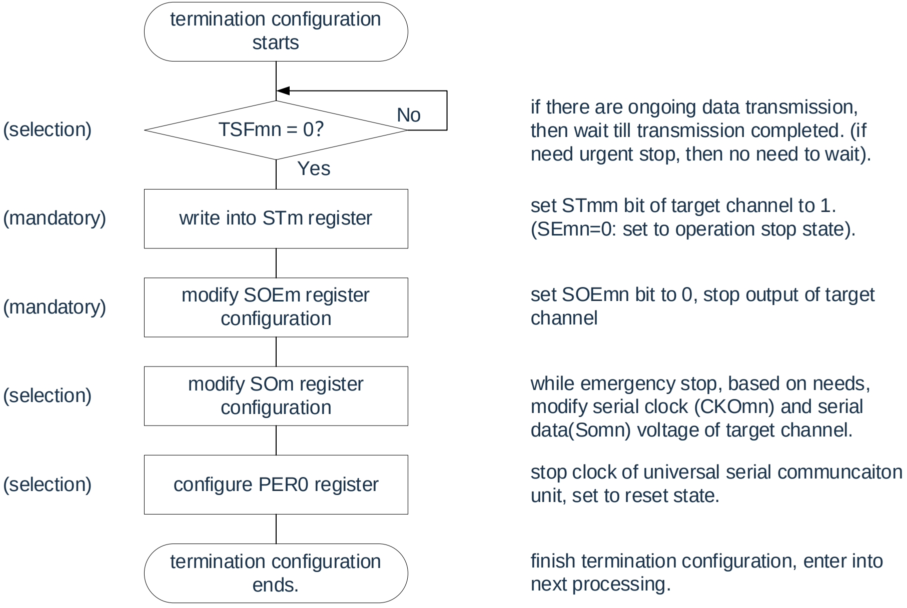
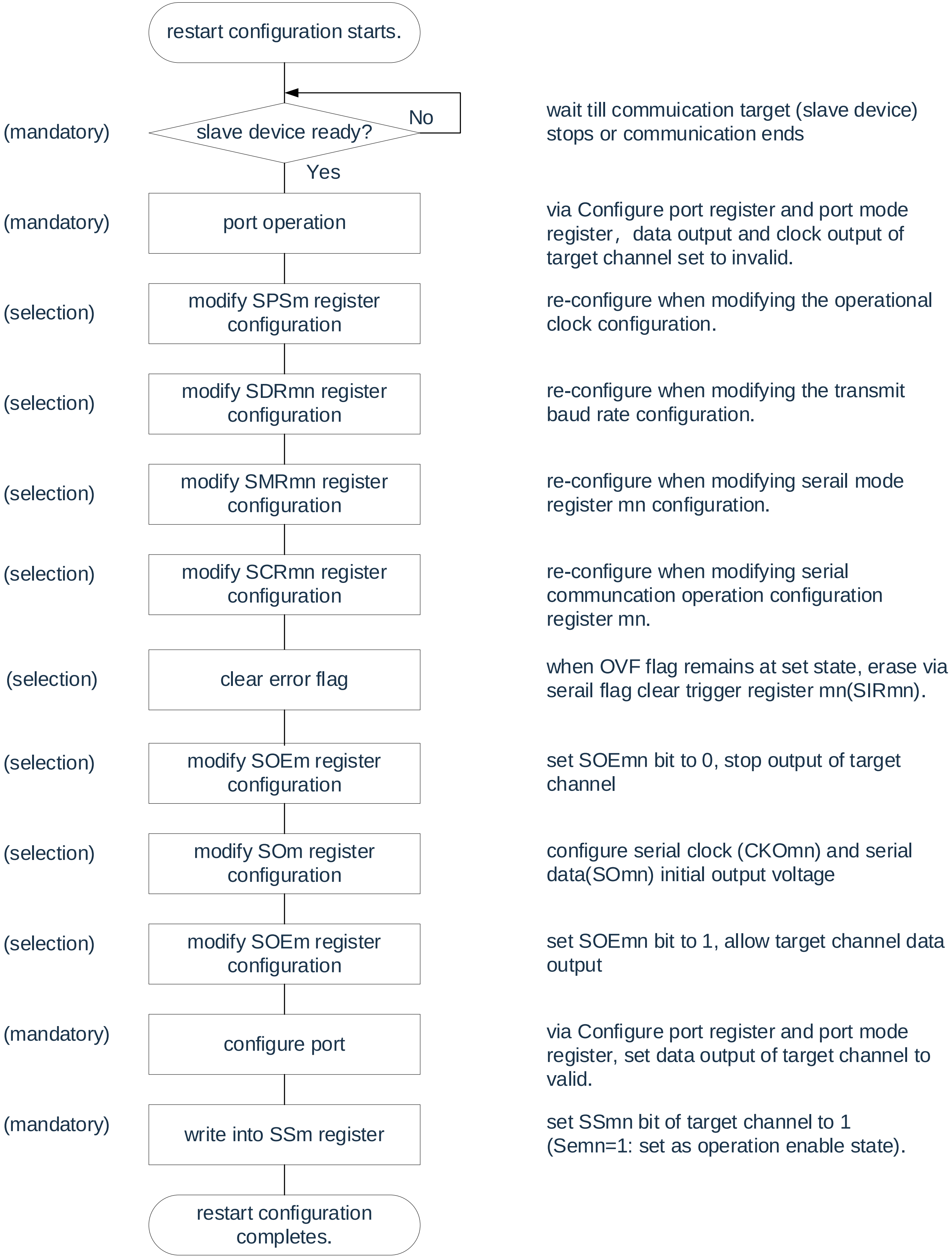
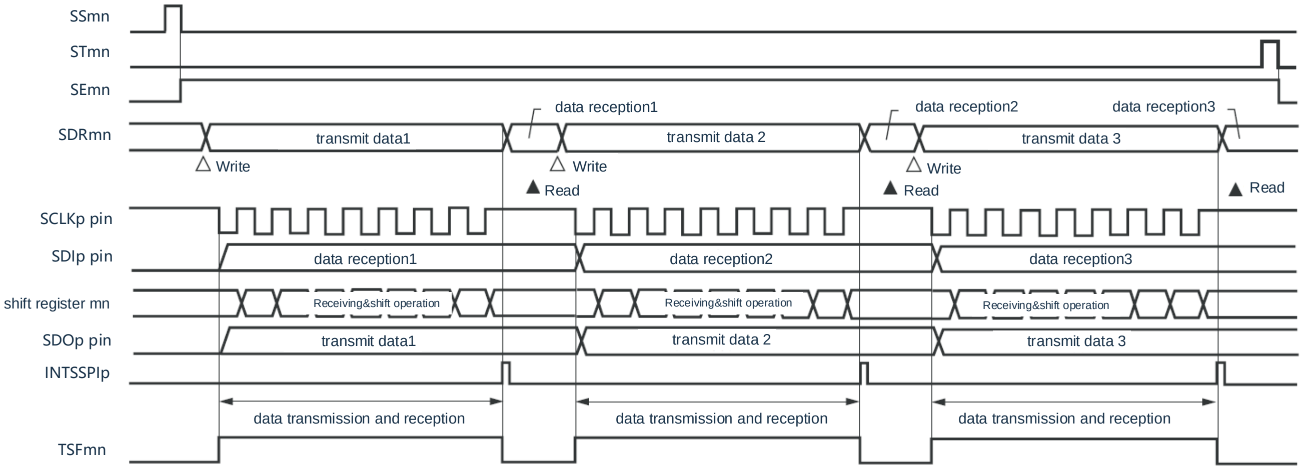

termination configuration starts
(selection)
TSFmn = 0?
No
Yes
if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).
(mandatory)
write into STm register
set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).
(mandatory)
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 12-42 Restart steps of the master transmit and receive


restart configuration starts.
(mandatory)
slave device ready?
No
Yes
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).
restart configuration completes.

(3) Processing flow (single send and receive mode).

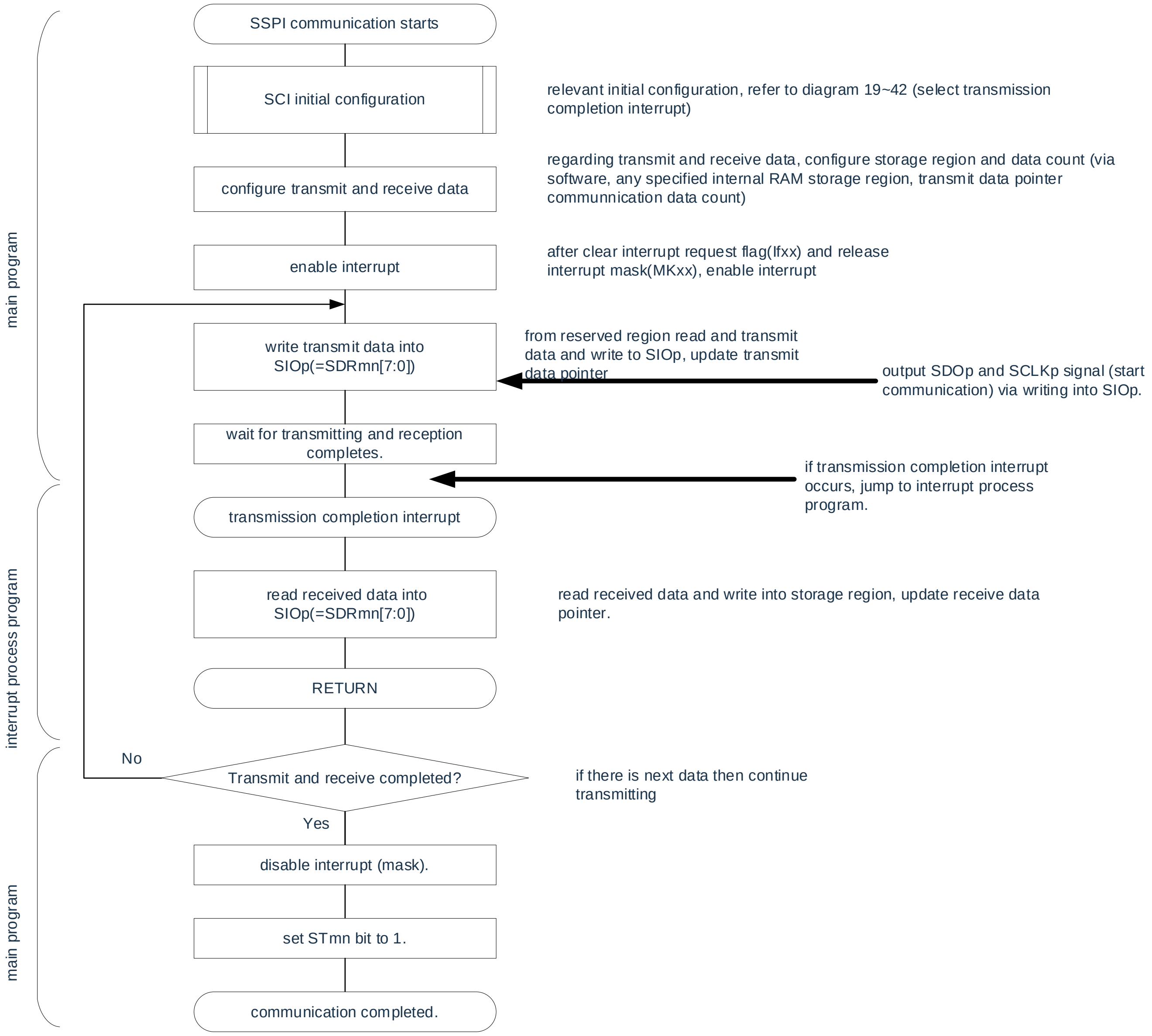
Figure 12-43 Timing diagram of the master transmit and receive (single-pass transmit and receive mode) (Type 1: DAPmn=0, CKPmn=0).



Remark m: Unit number (m=0, 1) n: Channel number (n=0~3)p: SSPI number (p=00，01, 10，11，20, 21.))

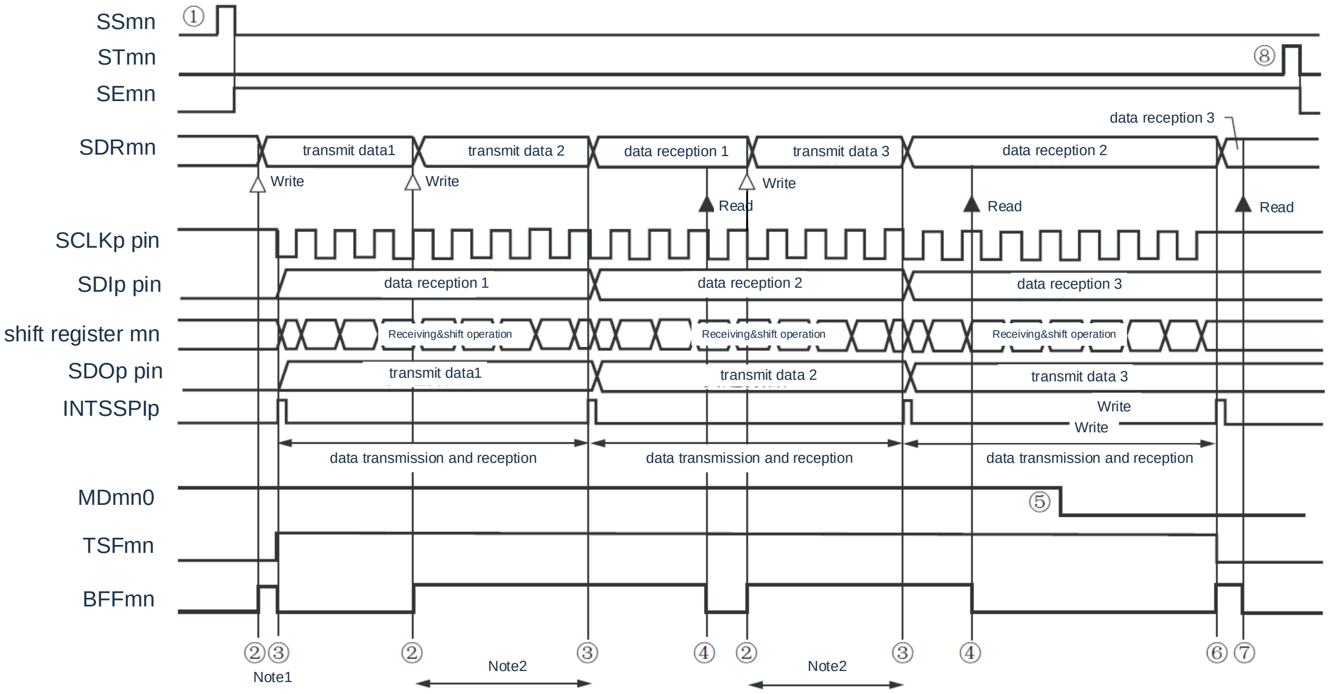
mn=00~03, 10~11

Figure 12-44 Flowchart of the master transmit and receive (single transmit and receive mode)


main program
SSPI communication starts
SCI initial configuration
relevant initial configuration, refer to diagram 19~42 (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 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 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 SIOp(=SDRmn[7:0])
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) Process flow (continuous send and receive mode).

Figure 12-45 Timing diagram of the main transmit and receive (continuous transmit and receive mode) (type 1: DAPmn=0, CKPmn=0).



Note 1 If the BFFmn bit of the serial status register mn (SSRmn) is “1” (valid data is saved in the serial data register mn (SDRmn) to write the send data to the SDRmn register, and rewrite the sent data.

2. If the SDRmn register is read during this period, the transmitted data can be read. At this point, the transfer run is not affected.

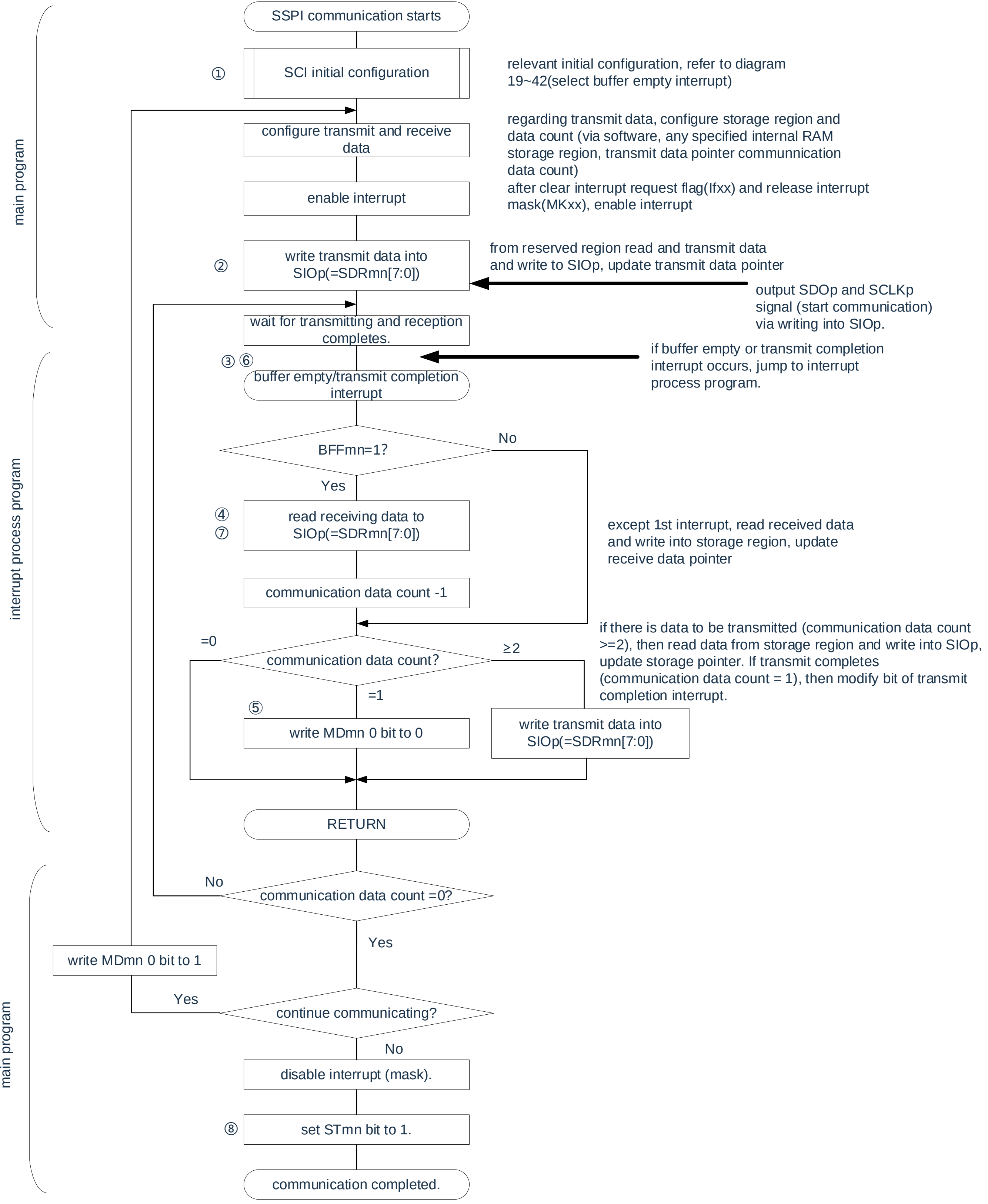
Notice The MDmn0 bit of the serial mode register mn (SMRmn) can be overridden even during operation. However, in order to catch up with the end of the transmission interruption of the last transmitted data, it must be overwritten before the last bit of transmission begins.

Remark 1. (1) to (8) in the figure corresponds to (1) to (8) in the “Flowchart of Figure 12-46)”.

2.m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)

mn=00~03, 10~11

Figure 12-46 Flowchart of the master transmit and receive (continuous transmit and receive mode)


main program
SSPI communication starts
①
SCI initial configuration
relevant initial configuration, refer to diagram 19~42(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 communnication data count)
enable interrupt
after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt
②
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 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 SIOp(=SDRmn[7:0])
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 SIOp, 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 SIOp(=SDRmn[7:0])
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.

(1) to (8) in the note figure corresponds to (1) to (8) in the “Timing diagram of Figure 12-45 Main Transmit and Receive (Continuous Send and Receive Mode)”.

12.5.4 Slave transmission

Slave transmission refers to the operation of the CMS32L051 microcontroller to send data to other devices in the state of transmitting the clock from the input of other devices.

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channel	Channel 0 of SCI0	Channel 1 of SCI0	Channel 2 of SCI0	Channel 3 of SCI0	Channel 0 of SCI1	Channel 1 of SCI1
The pin used	SCLK00, SDO00	SCLK01, SDO01	SCLK10, SDO10	SCLK11, SDO11	SCLK20, SDO20	SCLK21, SDO21
Interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21
	You can select either a transmit-end interrupt (single-pass mode) or a buffer null interrupt (continuous transfer mode).					
Error detection flag	Only the Overflow Error Detection Flag (OVFmn).					
The length of the transmitted data	7 or 8 bits					
Transfer rate	Max.f_{MCK}/6[Hz]Note1, 2					
Data phase	It can be selected via the DAPmn bit of the SCRmn register. • DAPmn=0: The data output starts when the serial clock starts running. • DAPmn=1: Starts data output half a clock before the serial clock starts running.					
Clock phase	It can be selected via the CKPmn bit of the SCRmn register. • CKPmn=0: Normal • CKPmn=1: Inverted					
Data direction	MSB first or LSB first					

Note 1. The maximum transfer rate is fMCK/6 [Hz] because the external serial clocks input from pins SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, and SCLK21 are sampled internally and then used.

2. it must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Note 1. f_{MCK}: The operating clock frequency of the object channel

2.m: Unit number (m=0, 1) n: channel number (n=0~3)mn=00~03 , 10~11.

(1) Register setting

Figure 12-47 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)
Example of register settings at the time of slave transmission

(a) serial mode register mn (SMRmn)

	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	0	1	DLSmn1 1 Note	DLSmn0 0/1

data and clock phase selection (details refer to "19.3 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
0: 7 bit data length
1: 8 bit data length

(c) serial data regsiter mn (SDRmn) (low 8 bit: SIOp)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	0000000 baud rate configuration							0	configuration of transmit data							

SIOp

(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

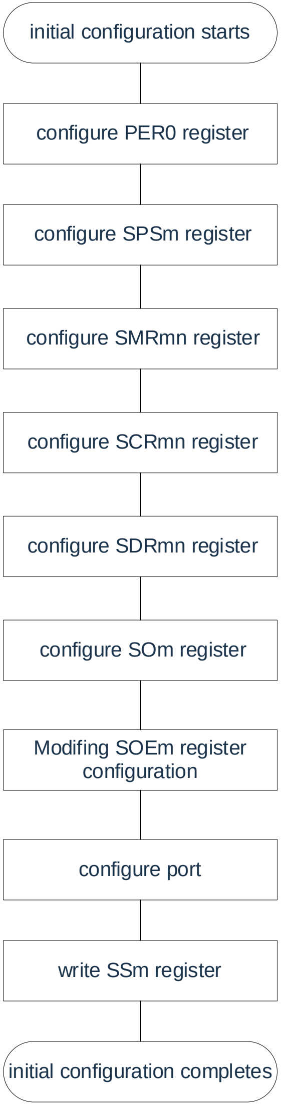
Note Limited to SCR00 register and SCR01 register, the others are fixed as “1”.

Note 1.m: Unit number (m=0, 1) n: Channel number (n=0~3)p: SSPI number (p=00, 01, 10, 11, 20, 21.))
mn=00~03, 10~11

2. □ : Fixed in SSPI slave send mode. ▨ : Cannot be set (initial value).
×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).
0/1: Set “0” or “1” according to the user's purpose.

(2) Operation Steps

Figure 12-48 Initial setup steps for slave sending



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

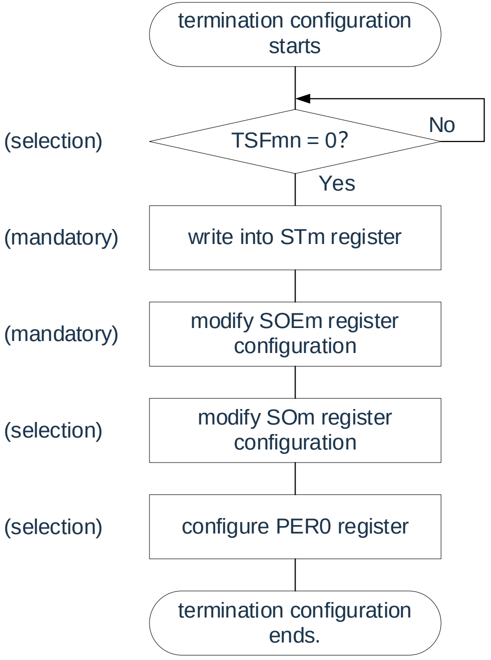
set SOEmn bit to 1, enable data output of target channel

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 SIOp register (bit 7~0 of SDRmn register), wait for master device clock.

Figure 12-49 Abort step of slave sending


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).

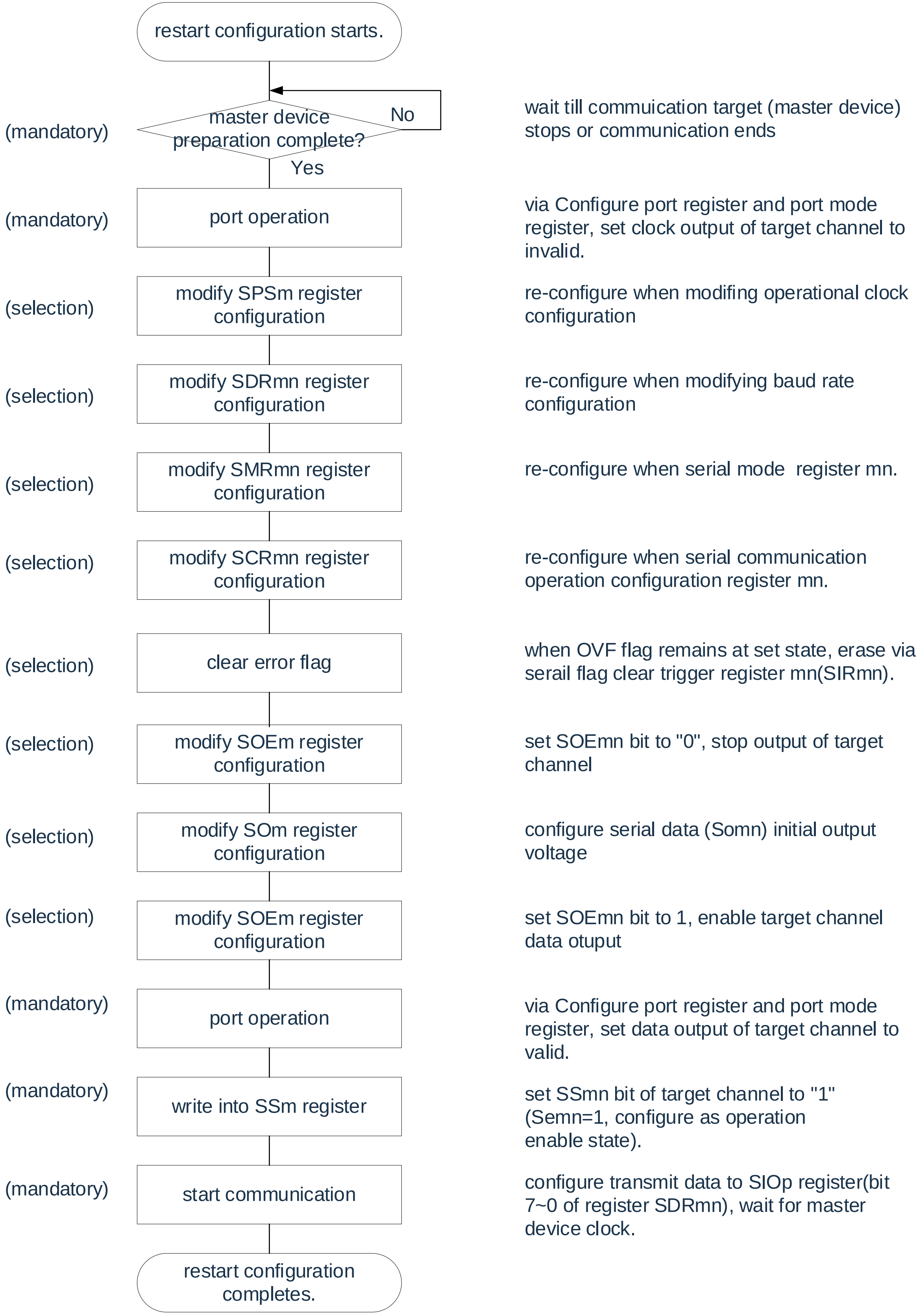
set SOEmn bit to 0, stop output of target channel

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.

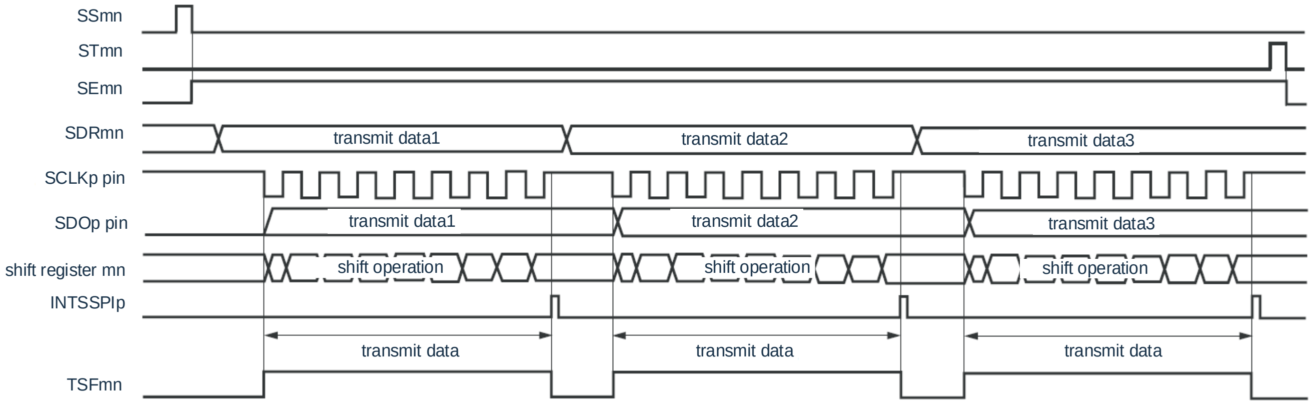
Figure 12-50 Restart setup step of slave sending


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 SIOp register(bit 7~0 of register SDRmn), wait for master device clock.
restart configuration completes.

Note If you override PER0 in the abort setting to stop the clock, you must wait until the communication object (the master device) stops or the communication is over to make the initial setting instead of the restart setting.

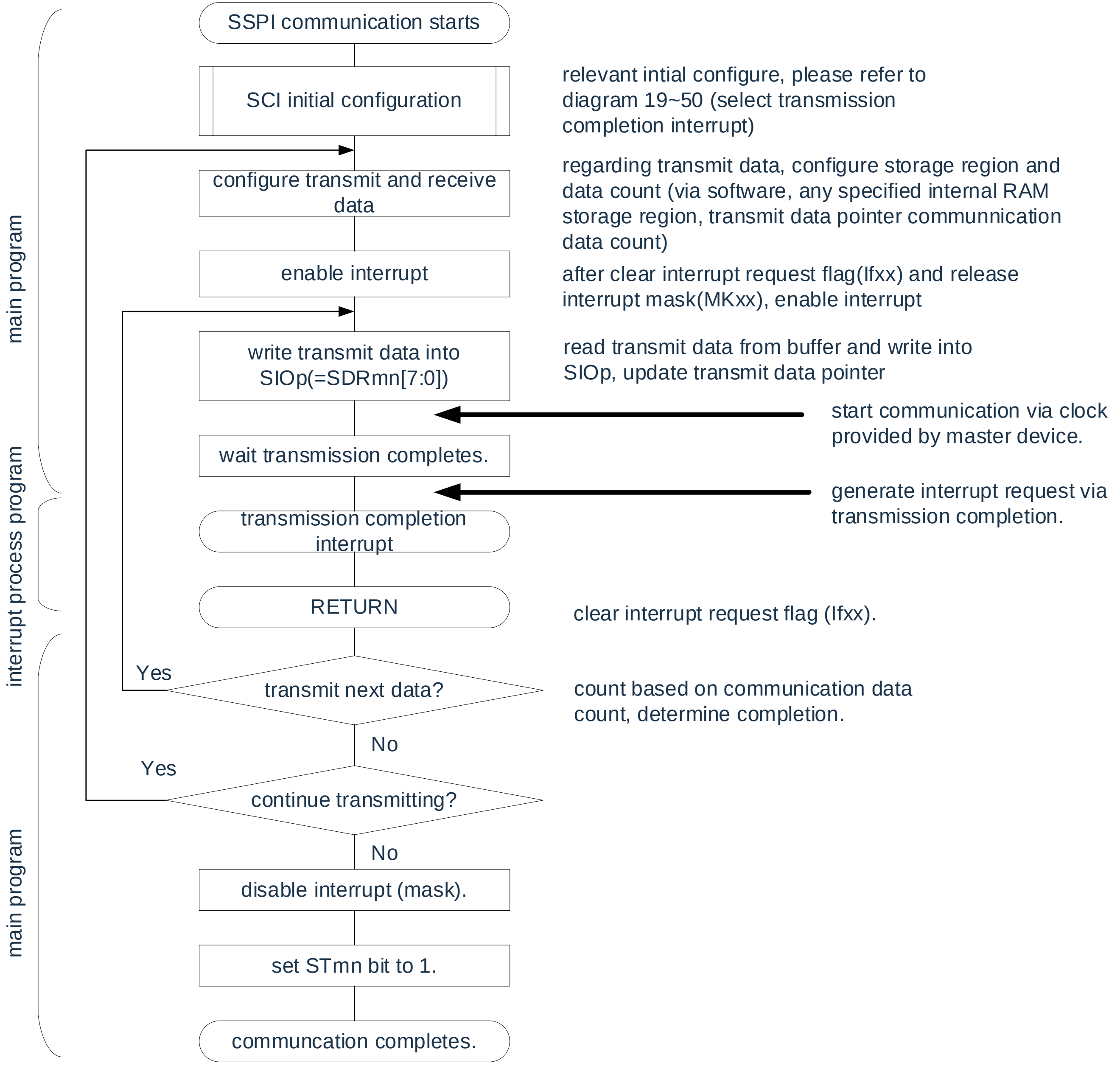
(3) Process flow (single-send mode).

Figure 12-51 Timing diagram of slave send (single-send mode) (type 1: DAPmn=0, CKPmn=0)



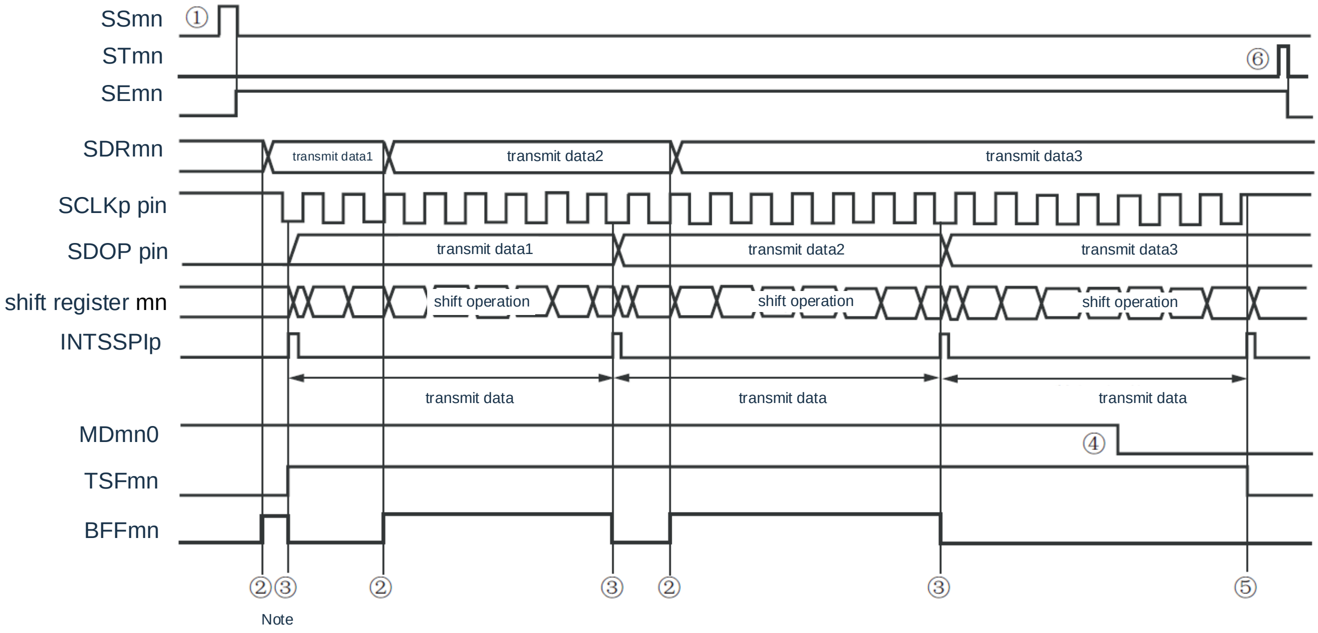
Remark m: Unit number (m=0, 1) n: Channel number (n=0~3)p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11

Figure 12-52 Flowchart of slave send (single send mode)


main program
interrupt process program
main program
SSPI communication starts
SCI initial configuration
relevant intial configure, please refer to diagram 19~50 (select transmission completion 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 communnication data count)
enable interrupt
after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt
write transmit data into SIOp(=SDRmn[7:0])
read transmit data from buffer and write into SIOp, 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?
count based on communication data count, determine completion.
No
Yes
continue transmitting?
No
disable interrupt (mask).
set STmn bit to 1.
communcation completes.

(4) Process flow (continuous send mode)

Figure 12-53 Timing diagram of slave transmit (continuous transmit mode) (type 1: DAPmn=0, CKPmn=0)

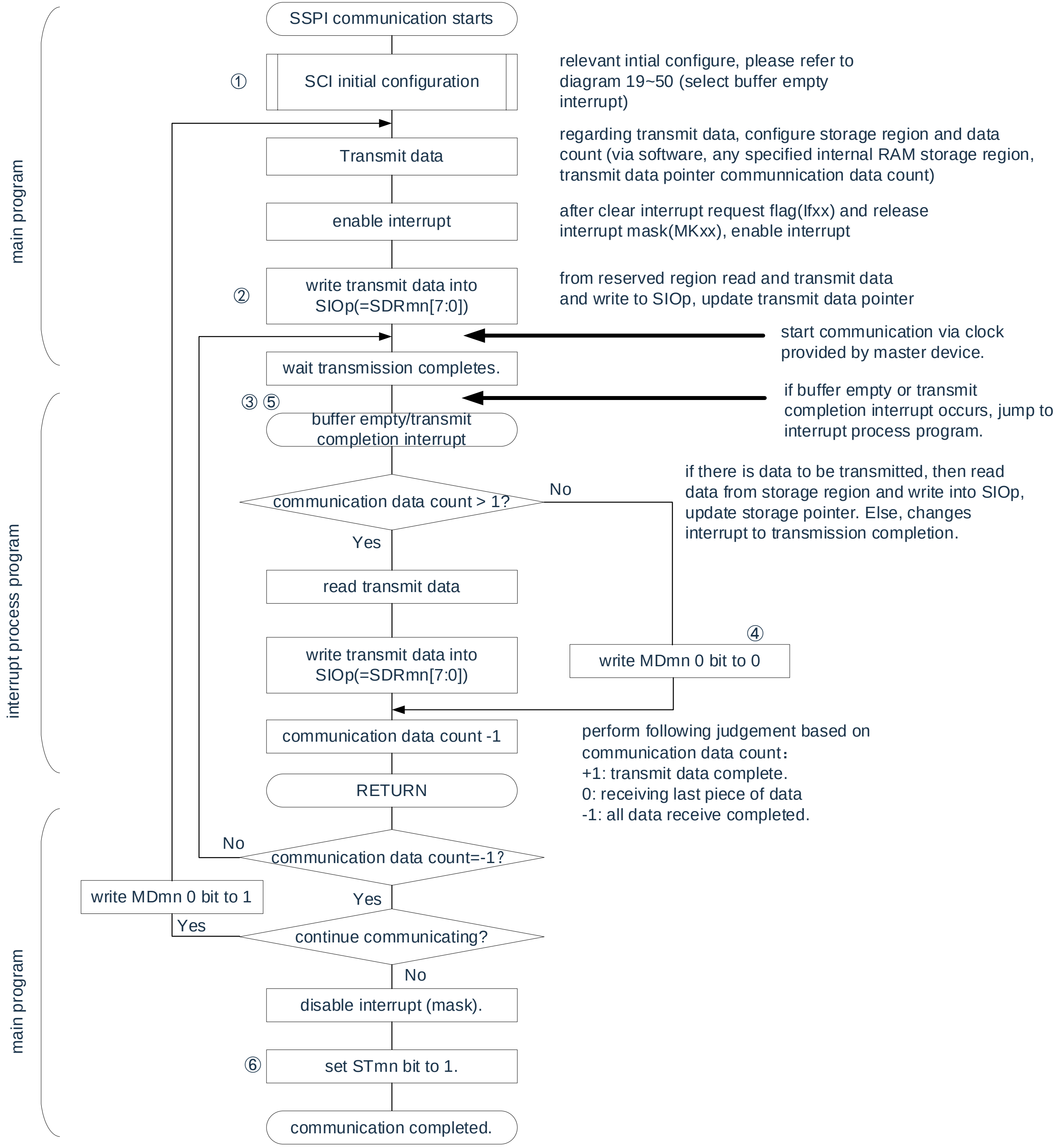


Note If the BFFmn bit of the serial status register mn (SSRmn) is “1” (when valid data is saved in the serial data register mn (SDRmn)) is given The SDRmn register writes the transmitted data and overrides the transmitted data.

Notice The MDmn0 bit of the serial mode register mn (SMRmn) can be overridden even during operation. However, it must be overridden before the last bit can be transferred.

Remark m: Unit number (m=0, 1) n: Channel number (n=0~3)p: SSPI number (p=00，01, 10，11，20，21)
mn=00~03, 10~11.

Figure 12-54 Flowchart of slave sending (continuous transmission mode).


main program
interrupt process program
main program
SSPI communication starts
①
SCI initial configuration
relevant intial configure, please refer to diagram 19~50 (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 SIOp(=SDRmn[7:0])
from reserved region read and transmit data and write to SIOp, 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.
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 SIOp, update storage pointer. Else, changes interrupt to transmission completion.
Yes
read transmit data
④
write transmit data into SIOp(=SDRmn[7:0])
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
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.

(1) to (6) in the note figure corresponds to (1) to (6) in the “Figure 12-53”

12.5.5 Slave receiving

Slave reception refers to the operation of this product to receive data from other devices in the state of transmitting clocks from other devices.

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channel	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1
The pin used	SCLK00, SDI00	SCLK01, SDI01	SCLK10, SDI10	SCLK11, SDI11	SCLK20, SDI20	SCLK21, SDI21
interrupt	INTSSPI00	INTSSPI01	INTSSPI10	INTSSPI11	INTSSPI20	INTSSPI21
	Limited to end-of-transfer interrupts (disable setting buffer null interrupts).					
Error detection flag	Only the Overflow Error Detection Flag (OVFmn).					
The length of the transmitted data	7 or 8 bits					
Transfer rate	Max.f_{MCK}/6[Hz]$^{Note1, 2}$					
Data phase	It can be selected via the DAPmn bit of the SCRmn register. • DAPmn=0: The data output starts when the serial clock starts running. • DAPmn=1: Starts data output half a clock before the serial clock starts running.					
Clock phase	It can be selected via the CKPmn bit of the SCRmn register. • CKPmn=0: Normal • CKPmn=1: Inverted					
Data direction	MSB first or LSB first					

Note 1. The maximum transfer rate is f_{MCK}/6 [Hz] because the external serial clocks input from pins SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, and SCLK21 are sampled internally and then used.

2. It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark 1. f_{MCK}: The operating clock frequency of the object channel

2.m: Unit number (m=0, 1) n: channel number (n=0~3)mn=00~03 , 10~11.

(1) Register setting

Figure 12-55 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) Example of register settings at slave receive

(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 registermn 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	0	1	DLSmn1 1 Note	DLSmn0 0/1

data and clock phase selection (details refer to "19.3 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
0: 7 bit data length
1: 8 bit data length

(c) serial data regsiter mn (SDRmn) (low 8 bit: SIOp)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	0000000 baud rate configuration							0	received data							
									SIOp							

(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 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

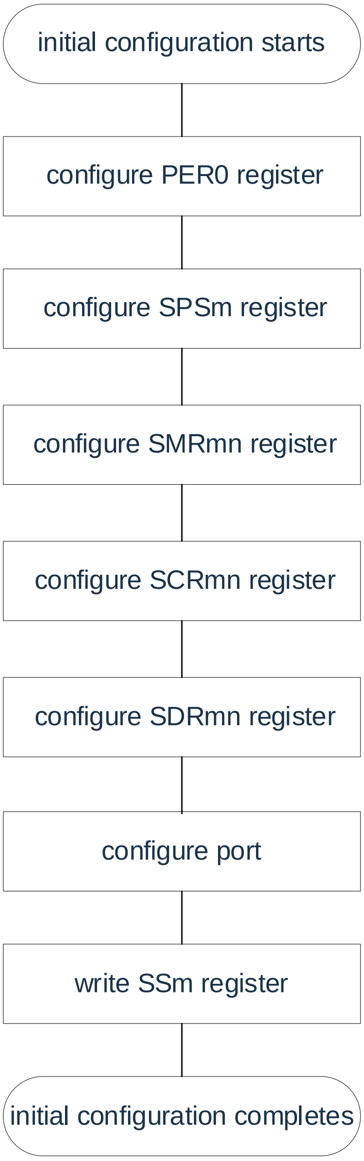
Note Limited to SCR00 register and SCR01 register, the others are fixed as "1".

Note 1.m: Unit number (m=0, 1)n: Channel number (n=0~3)p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)
mn=00~03, 10~11

2. □ : Fixed in Slave Receive mode. ▨ : Cannot be set (initial value).
×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).
0/1: Set "0" or "1" according to the user's purpose.

2) Operation steps

Figure 12-56 Initial setup step of slave reception



release universal serial communication unit from reset state, start providing clock.

configure operational clock

configure operational mode..etc.

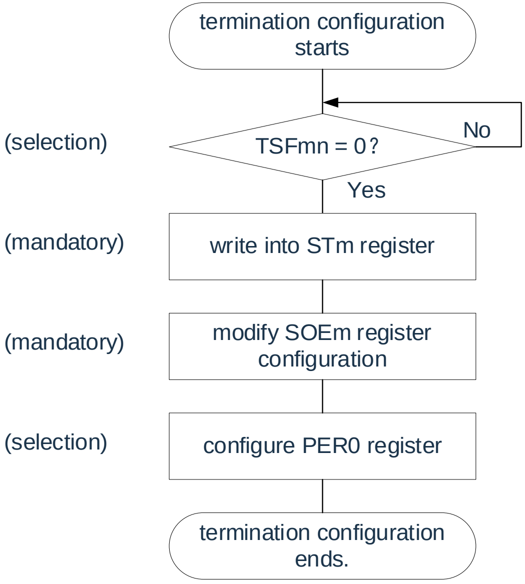
configure communication format

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 12-57 Abort step of Slave receiving



if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).

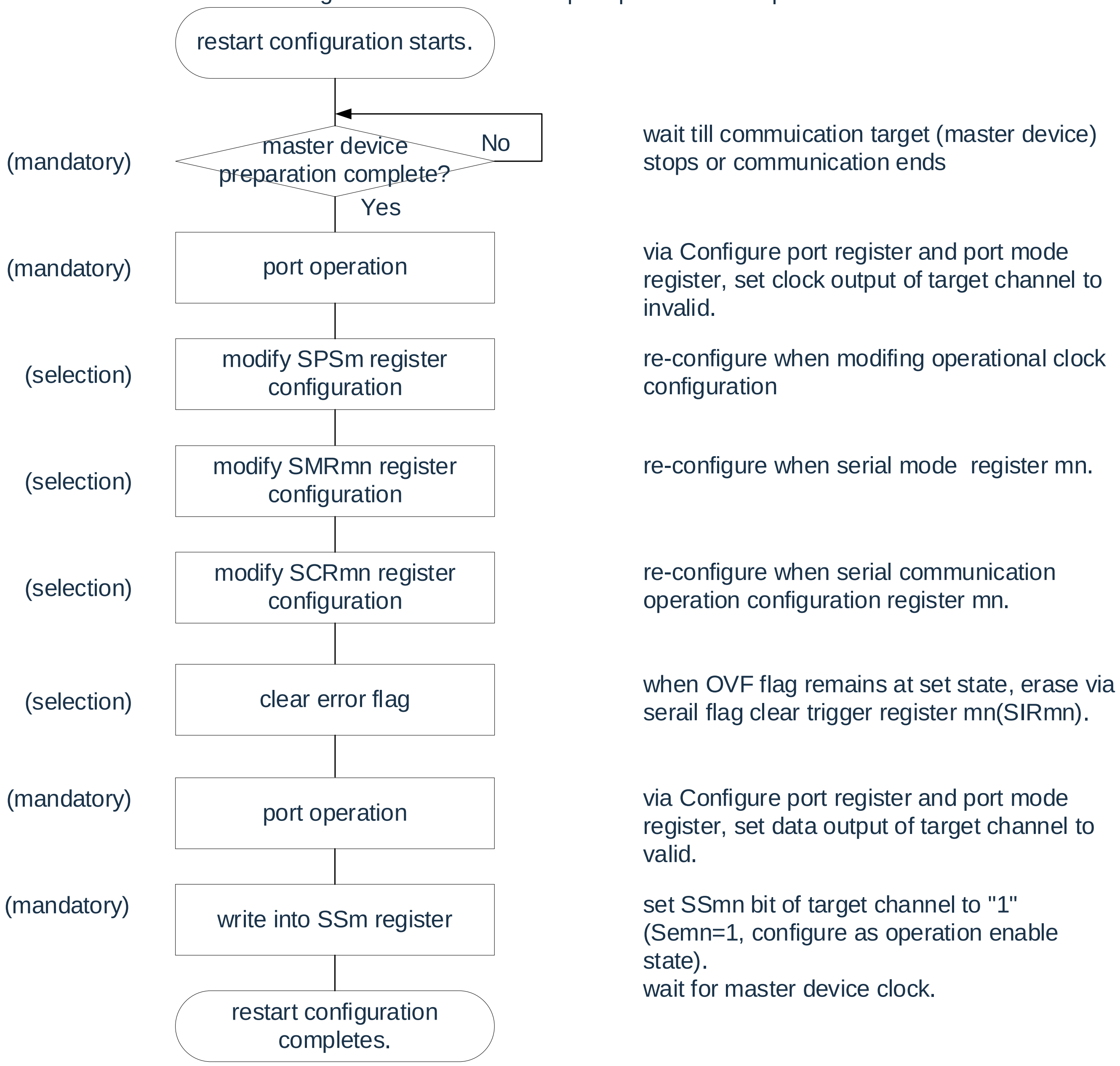
set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).

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.

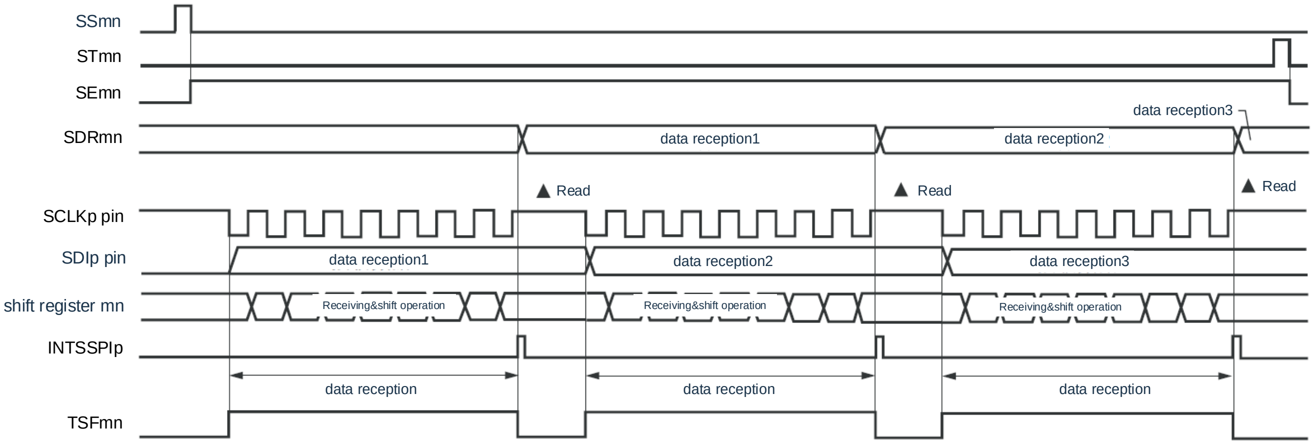
Figure 12-58 Restart setup step of slave reception


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).
(mandatory)
port operation
via Configure port register and port mode register, set data output of target channel to valid.
(mandatory)
write into SSm register
set SSmn bit of target channel to "1" (Semn=1, configure as operation enable state).
wait for master device clock.
restart configuration completes.

Note If you override PER0 in the abort setting to stop the clock, you must wait until the communication object (the master device) stops or the communication is over to make the initial setting instead of the restart setting.

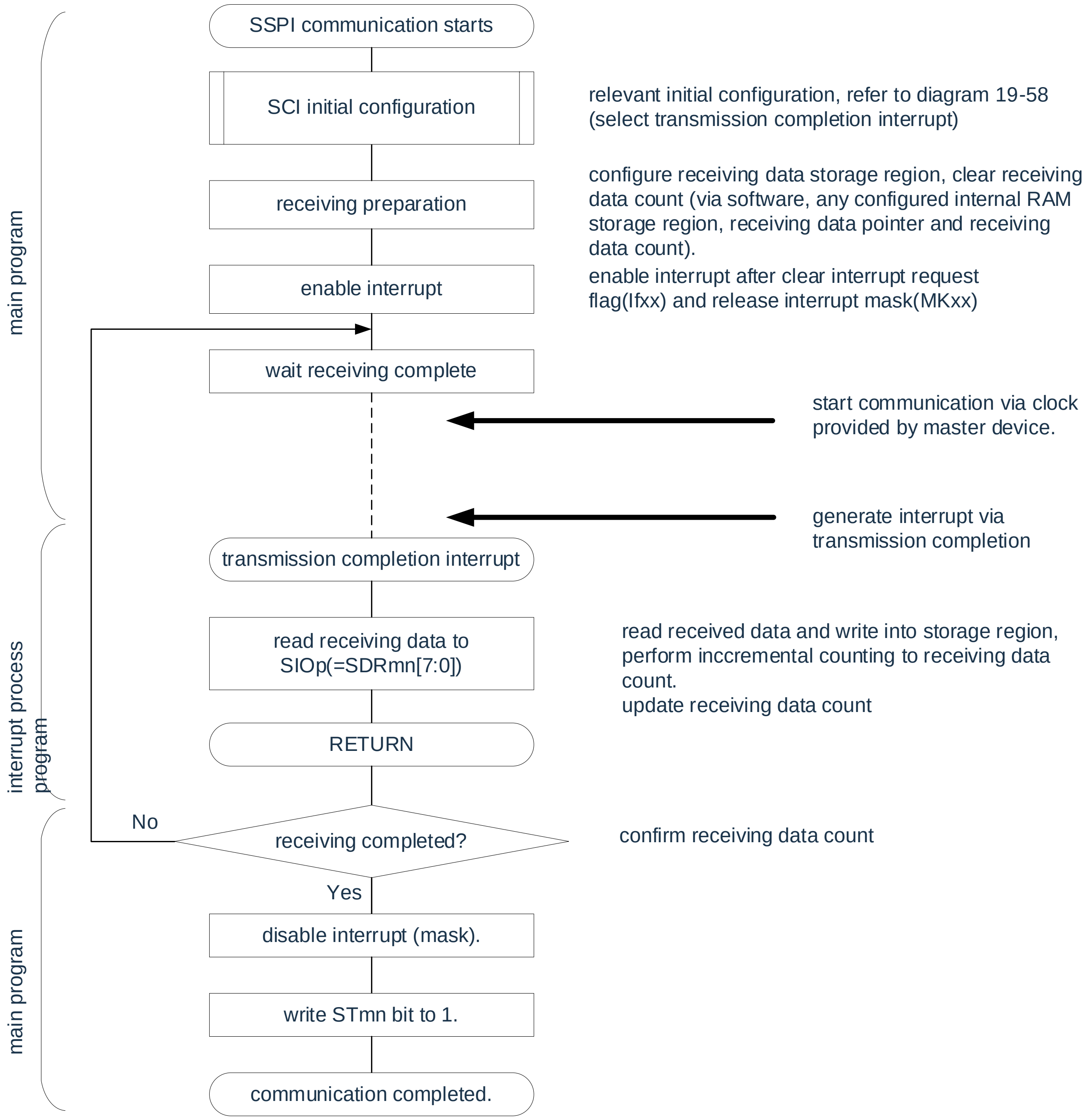
(3) Processing flow (single-receive mode).

Figure 12-59 Timing diagram of slave receive (single-receive mode) (type 1: DAPmn=0, CKPmn=0).



Note m: Unit number (m=0, 1)n: Channel number (n=0~3)p: SSPI number (p=00, 01, 10, 11, 20, 21, 30, 31)
mn=00~03, 10~11

Figure 12-60 Flowchart of slave receive (single-receive mode)


main program
SSPI communication starts
SCI initial configuration
relevant initial configuration, refer to diagram 19-58
(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
interrupt process program
transmission completion interrupt
read receiving data to SIOp(=SDRmn[7:0])
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
main program
disable interrupt (mask).
write STmn bit to 1.
communication completed.

12.5.6 Slave send and receive

Slave transmit and receive refers to the operation of the microcontroller and other devices of this product to transmit and receive data in the state of transmitting clocks from other devices.

3-wire serial I/O	SSPI00	SSPI01	SSPI10	SSPI11	SSPI20	SSPI21
Object channel	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1
The 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
	You can select either a transmit-end interrupt (single-pass mode) or a buffer null interrupt (continuous transfer mode).					
Error detection flag	Only the Overflow Error Detection Flag (OVFmn).					
The length of the transmitted data	7 or 8 bits					
Transfer rate	Max.f_{MCK}/6[Hz]Note[1, 2]					
Data phase	It can be selected via the DAPmn bit of the SCRmn register. • DAPmn=0: Data input/output starts when the serial clock starts running. • DAPmn=1: Starts data input/output half a clock before the serial clock starts running.					
Clock phase	It can be selected via the CKPmn bit of the SCRmn register. • CKPmn=0: Normal • CKPmn=1: Inverted					
Data direction	MSB first or LSB first					

Note 1. The maximum transfer rate fMCK/6 [Hz] is used because the external serial clock input from SCLK00, SCLK01, SCLK10, SCLK11, SCLK20, and SCLK21 pins is sampled internally and then used.

2. It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark 1. f_{MCK}: The operating clock frequency of the object channel

2.m: Unit number (m=0, 1) n: channel number (n=0~3)mn=00~03 , 10~11

(1) Register setting

Figure 12-61 3 wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) Example of register settings when slave transmit and receive

(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	0	1	DLSmn1 1 Note	DLSmn0 0/1

data and clock phase selection (details refer to "19.3 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
0: 7 bit data length
1: 8 bit data length

(c) serial data regsiter mn (SDRmn) (low 8 bit: SIOp)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	0000000 baud rate configuration							0	configuration of transmit data/received data register							

SIOp

(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 Limited to SCR00 register and SCR01 register, the others are fixed as “1”.

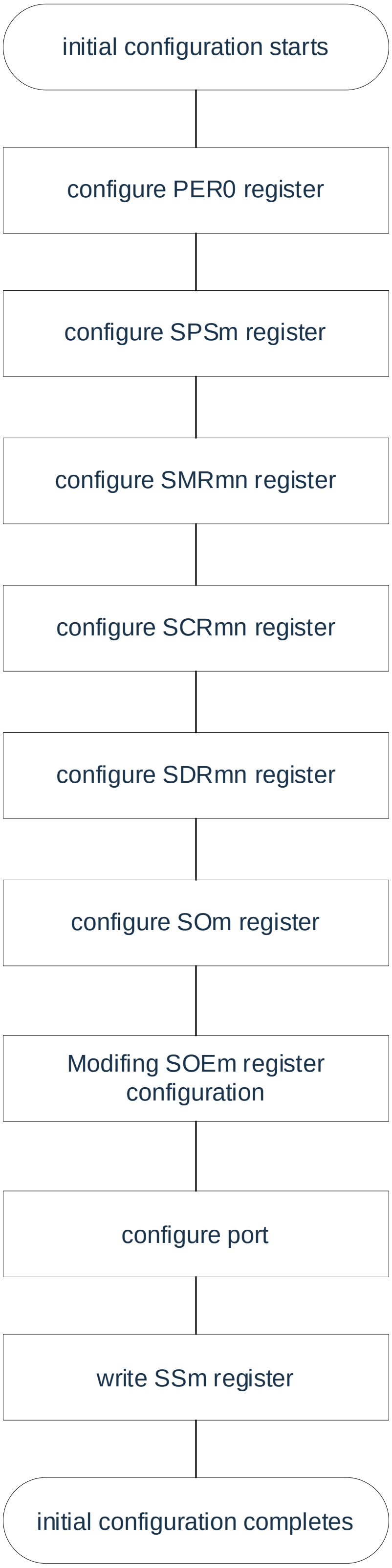
Notice Data must be sent to the SIOp register settings before the master device starts the output clock.

Note 1.m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11

2. □ : Fixed in SSPI slave T&R mode. ▨ : Cannot be set (set initial value)
×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).
0/1: Set “0” or “1” according to the user's purpose.

(2) Operation Steps

Figure 12-62 Initial setup steps for slave send and receive



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 SIOp register (bit 7~0 of SDRmn register), wait for master device clock.

Note that data must be sent to the SIOp register settings before the master device starts the output clock.

Figure 12-63 Stop steps for slave send and receive

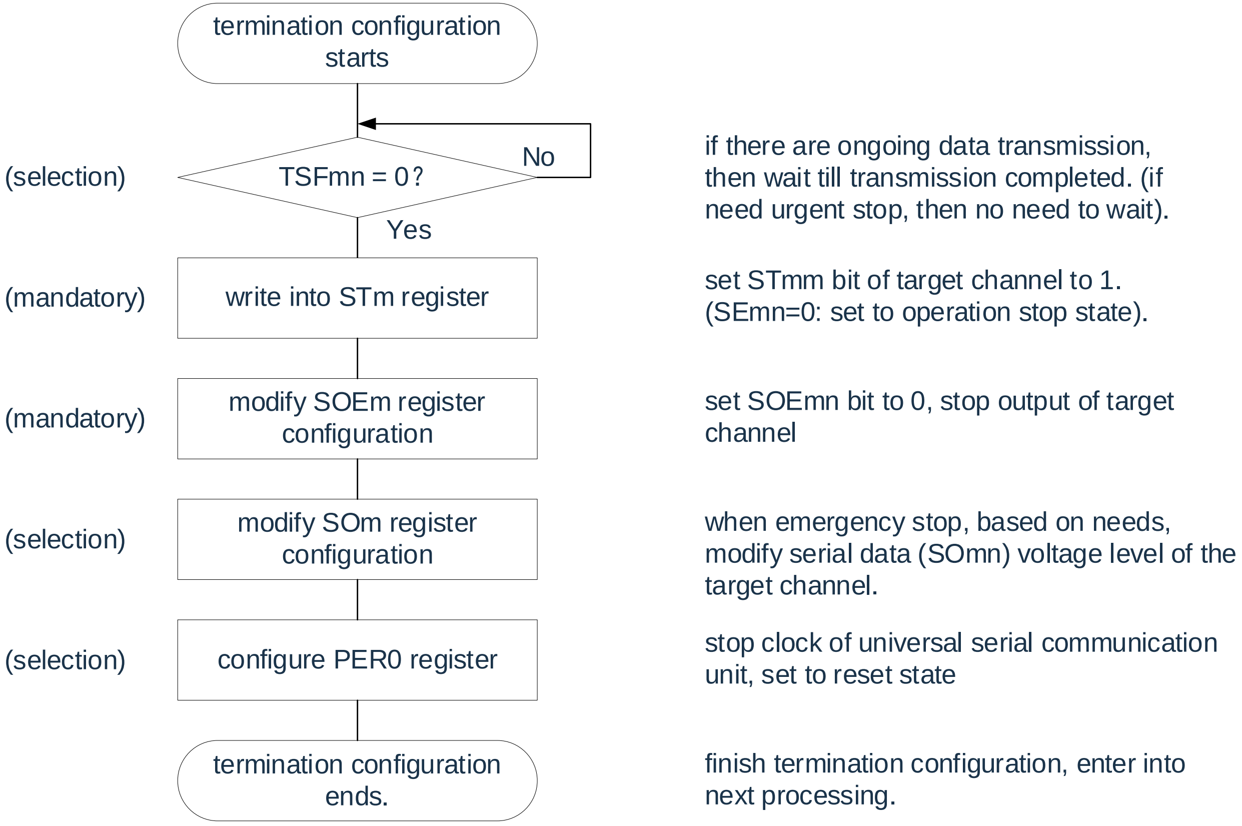
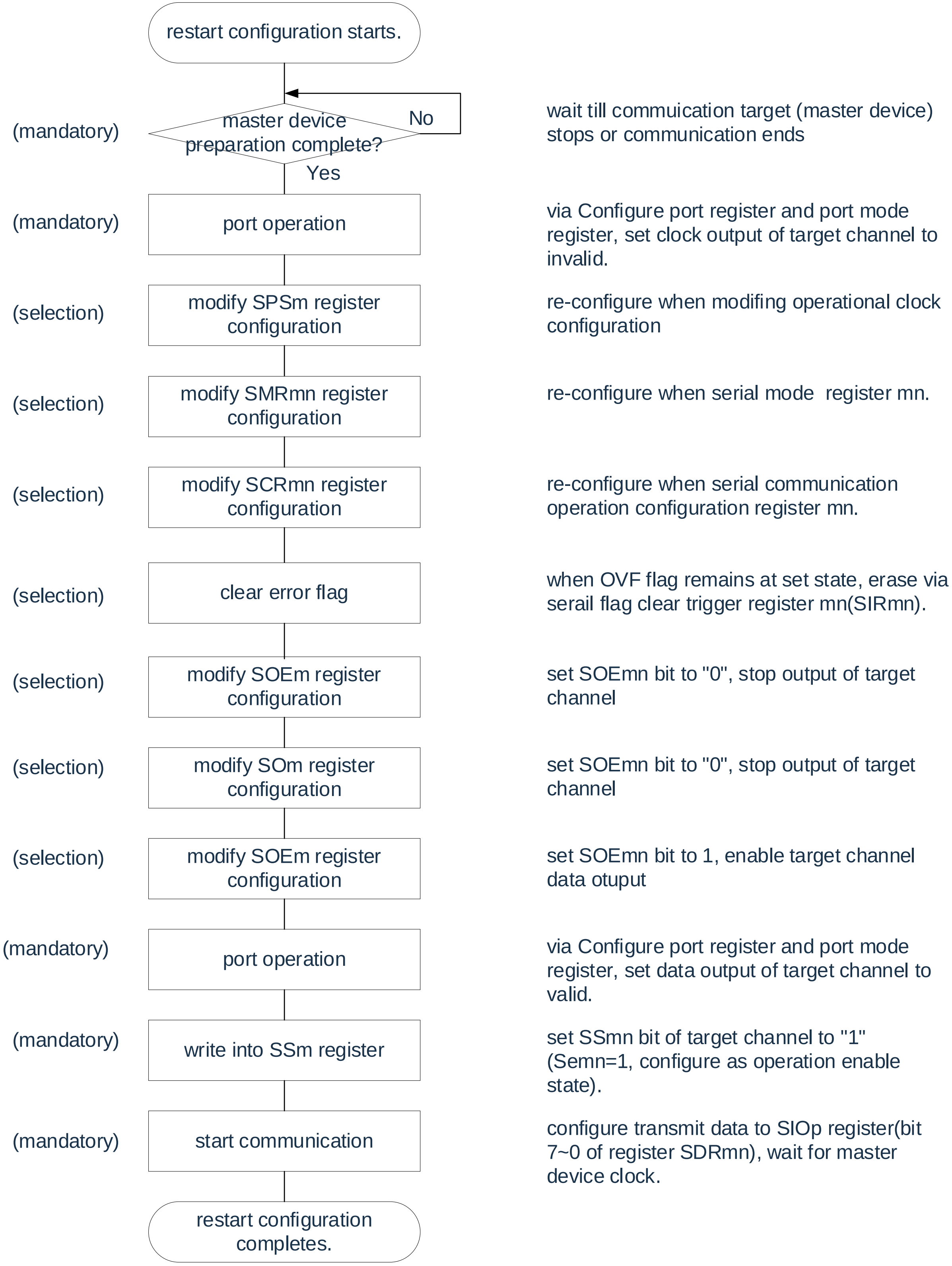

termination configuration starts
(selection)
TSFmn = 0 ?
No
Yes
if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).
(mandatory)
write into STm register
set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).
(mandatory)
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 12-64 Restarts setup steps of slave send and receive

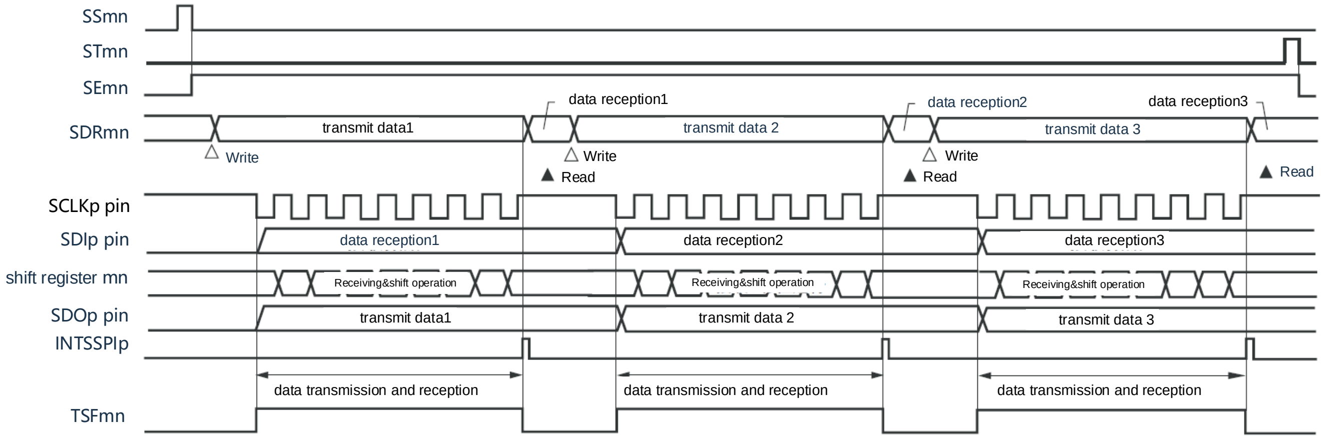

restart configuration starts.
(mandatory)
master device preparation complete?
No
Yes
wait till commuication target (master device) stops or communication ends
(mandatory)
port operation
via Configure port register and port mode register, set clock output of target channel to invalid.
(selection)
modify 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 SIOp register(bit 7~0 of register SDRmn), wait for master device clock.
restart configuration completes.

Note 1 Before the master device starts to output the clock, data must be sent to the SIOp register settings.

2. If you override PER0 in the abort setting to stop the clock, you must wait until the communication object (the master device) stops or the communication is over to make the initial setting instead of the restart setting.

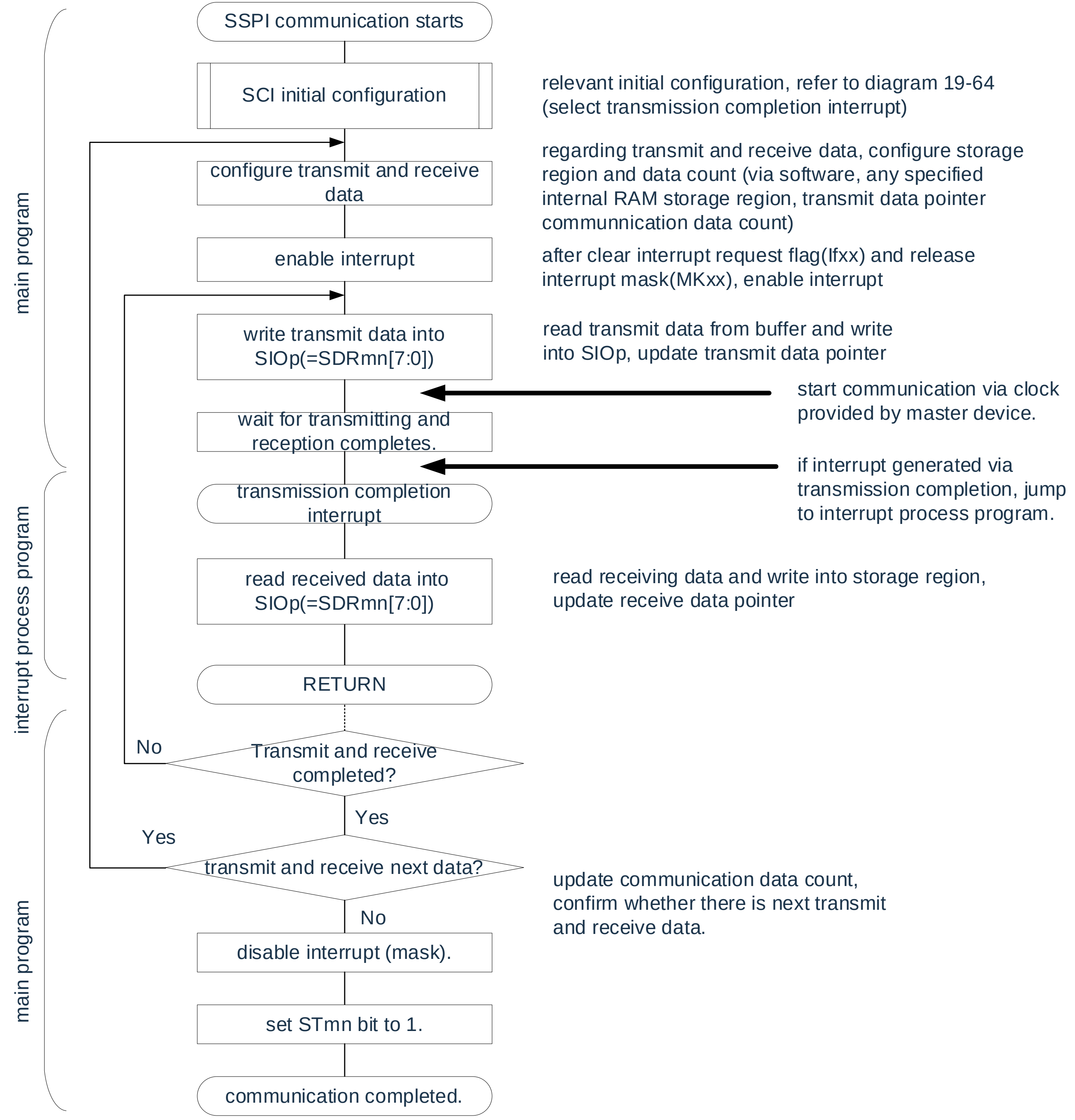
(3) Processing flow (single send and receive mode)

Figure 12-65 Timing diagram of slave transmit and receive (single transmit and receive mode) (type 1: DAPmn=0, CKPmn=0).



Remark m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11.

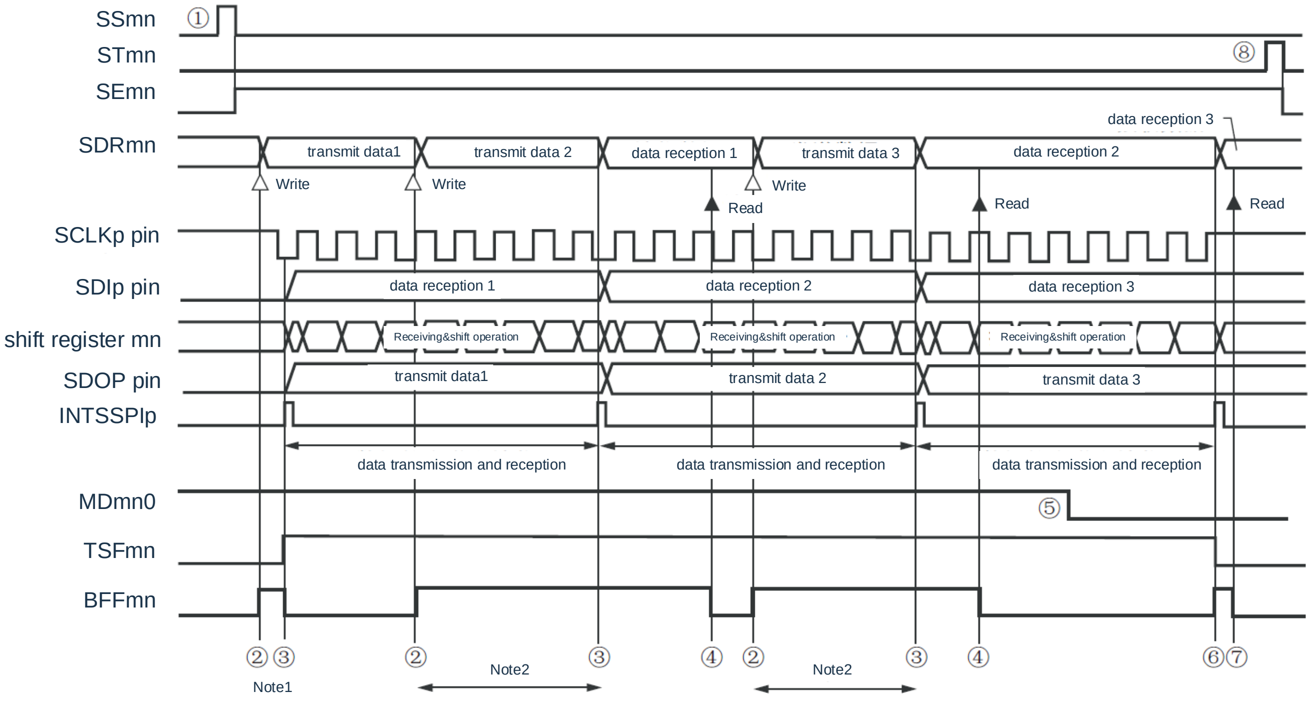
Figure 12-66 Flowchart of slave send and receive (single send and receive mode).


main program
interrupt process program
main program
SSPI communication starts
SCI initial configuration
relevant initial configuration, refer to diagram 19-64 (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 SIOp(=SDRmn[7:0])
read transmit data from buffer and write into SIOp, 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 SIOp(=SDRmn[7:0])
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.

Note Data must be sent to the SIOp register settings before the master device starts the output clock.

(4) Process flow (continuous send and receive mode).

Figure 12-67 Timing diagram of slave transmit and receive (continuous transmit and receive mode) (type 1: DAPmn=0, CKPmn=0).



Note 1 If the BFFmn bit of the serial status register mn (SSRmn) is “1” (valid data is saved in the serial data register mn (SDRmn) to write the send data to the SDRmn memory, and override the sent data.

2. If the SDRmn register is read during this period, the transmitted data can be read. At this point, the transfer run is not affected.

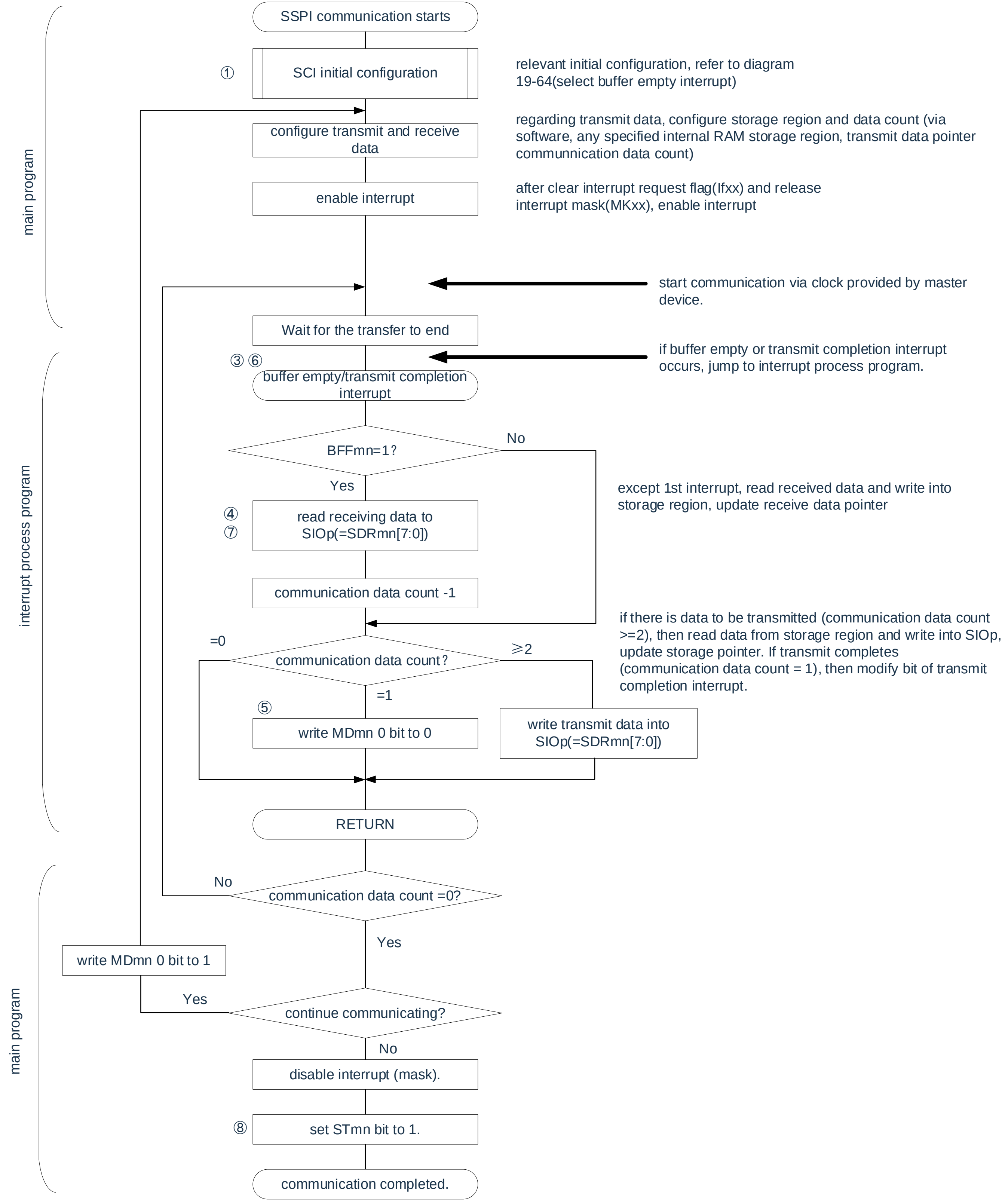
Notice The MDmn0 bit of the serial mode register mn (SMRmn) can be overridden even during operation. However, in order to catch up with the end of the transmission interruption of the last transmitted data, it must be overwritten before the last bit of transmission begins.

Remark 1. (1) to (8) in the figure corresponds to (1) to (8) in “Figure 12-68 of Slave Send and Receive (Continuous Send and Receive Mode)”.

2.m: Unit number (m=0, 1) n: Channel number (n=0~3) p: SSPI number (p=00, 01, 10, 11, 20, 21)

mn=00~03, 10~11

Figure 12-68 Flowchart of Slave transmit and receive (continuous transmit and receive mode)


main program
SSPI communication starts
①
SCI initial configuration
relevant initial configuration, refer to diagram 19-64(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 communnication data count)
enable interrupt
after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt
start communication via clock provided by master device.
Wait for the transfer to end
if buffer empty or transmit completion interrupt occurs, jump to interrupt process program.
interrupt process program
③ ⑥
buffer empty/transmit completion interrupt
BFFmn=1?
No
Yes
④
⑦
read receiving data to SIOp(=SDRmn[7:0])
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 SIOp, 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 SIOp(=SDRmn[7:0])
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.

Note Data must be sent to the SIOp register settings before the master device starts the output clock.

Remark (1) to (8) in the note figure corresponds to (1) to (8) in the “Figure 12-67”.

12.5.7 Calculation of transmit clock frequency

3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21) communication transmission clock frequency can be calculated using the following calculation equations.

(1) Master device

(transmit clock frequency) = {the operating clock of the object channel (f_{MCK}) frequency}   (SDRmn[15:9]+ 1)   2[Hz]

(2) Slave device

(transmit clock frequency) = {Serial clock (SCLK) frequency provided by the master device} Note [Hz].

Note The maximum enable transmit clock frequency is f_{MCK}/6.

Note Because the value of SDRmn [15:9] is the value of bit15~9 of the serial data register mn (SDRmn) (0000000B~). 1111111B), so it is 0~127.

The operating clock(F_{MCK}) depends on bit15 (CKSmn) of the serial clock selection register m (SPSm) and the serial mode register mn (SMRmn).

Table 12-2 Selection of 3-wire serial I/O running clocks

SMRmn register	SPSm register								Runing clock (f_{MCK}) Note	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		f_{CLK}=32MHz operation
0		X	X	X				0	f_{CLK}	32MHz
		X	X	X				1	$f_{CLK}/2$	16MHz
		X	X	X				0	$f_{CLK}/2^{2}$	8MHz
		X	X	X				1	$f_{CLK}/2^{3}$	4MHz
		X	X	X				0	$f_{CLK}/2^{4}$	2MHz
		X	X	X				1	$f_{CLK}/2^{5}$	1MHz
		X	X	X				0	$f_{CLK}/2^{6}$	500kHz
		X	X	X				1	$f_{CLK}/2^{7}$	250kHz
		X	X	X				0	$f_{CLK}/2^{8}$	125kHz
		X	X	X				1	$f_{CLK}/2^{9}$	62.5kHz
		X	X	X				0	$f_{CLK}/2^{10}$	31.25kHz
		X	X	X				1	$f_{CLK}/2^{11}$	15.63kHz
		X	X	X				0	$f_{CLK}/2^{12}$	7.81kHz
		X	X	X				1	$f_{CLK}/2^{13}$	3.91kHz
		X	X	X				0	$f_{CLK}/2^{14}$	1.95kHz
		X	X	X				1	$f_{CLK}/2^{15}$	977Hz
1					X	X	X	X	f_{CLK}	32MHz
					X	X	X	X	$f_{CLK}/2$	16MHz
					X	X	X	X	$f_{CLK}/2^{2}$	8MHz
					X	X	X	X	$f_{CLK}/2^{3}$	4MHz
					X	X	X	X	$f_{CLK}/2^{4}$	2MHz
					X	X	X	X	$f_{CLK}/2^{5}$	1MHz
					X	X	X	X	$f_{CLK}/2^{6}$	500kHz
					X	X	X	X	$f_{CLK}/2^{7}$	250kHz
					X	X	X	X	$f_{CLK}/2^{8}$	125kHz
					X	X	X	X	$f_{CLK}/2^{9}$	62.5kHz
					X	X	X	X	$f_{CLK}/2^{10}$	31.25kHz
					X	X	X	X	$f_{CLK}/2^{11}$	15.63kHz
					X	X	X	X	$f_{CLK}/2^{12}$	7.81kHz
					X	X	X	X	$f_{CLK}/2^{13}$	3.91kHz
					X	X	X	X	$f_{CLK}/2^{14}$	1.95kHz
					X	X	X	X	$f_{CLK}/2^{15}$	977Hz

Note To change the clock selected as fCLK (change the value of the system clock control register (CKC)), you must stop the operation of the universal serial communication unit (SCI) (serial channel stop register m(STm)=000FH) after making the change.

Remark 1.X: Ignore
2. m: Unit number (m=0, 1) n: channel number (n=0~3)mn=00~03 , 10~11.

12.5.8 Procedure for handling errors during 3-wire serial I/O communication (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20, SSPI21)

In 3-wire serial I/O (SSPI00, SSPI01, SSPI10, SSPI11, SSPI20 SSPI21), the processing steps when an error occurs during communication are shown in Figure 12-69.

Figure 12-69 Steps to handle when an overflow error occurs

Software operation	Hardware status	Remark
Read the serial data register	The BFFmn bit of the SSRmn register is “0” and the channel n is in acceptable.	This is to prevent overflow errors from ending the next reception during mishandling.
Read the serial status register mn (SSRmn)		The type of error is judged, and the reading value is used to clear the error flag.
Clear trigger register mn to the serial flag	Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors during read operations can only be cleared.

Remark m: Unit number (m=0, 1) n: Channel number (n=0~3)mn=00~ 03, 10~11.

12.6 Operation of clock-synchronous serial communication with slave selection input function

Channel 0 of SCI0 is the channel that supports clock-synchronous serial communication with the slave select input function.

[Transmit and receive data]

- 7-bit or 8-bit data length
- Phase control of sending and receiving data
- MSB/LSB preferred choice
- Level settings for sending and receiving data

[Clock control].

- Phase control of input/output clocks
- Sets the transfer period generated by the prescaler and the in-channel counter.
- Maximum transfer rate Note for slave communication: Max.f_{MCK}/6

[Interrupt function].

- End of transfer interrupt, buffer empty interrupt

[Error detection flag].

- Overflow error

Note it must be used within the scope of the SCLK Cycle Time (t_{KCY}) characteristics. Please refer to the data sheet for details.

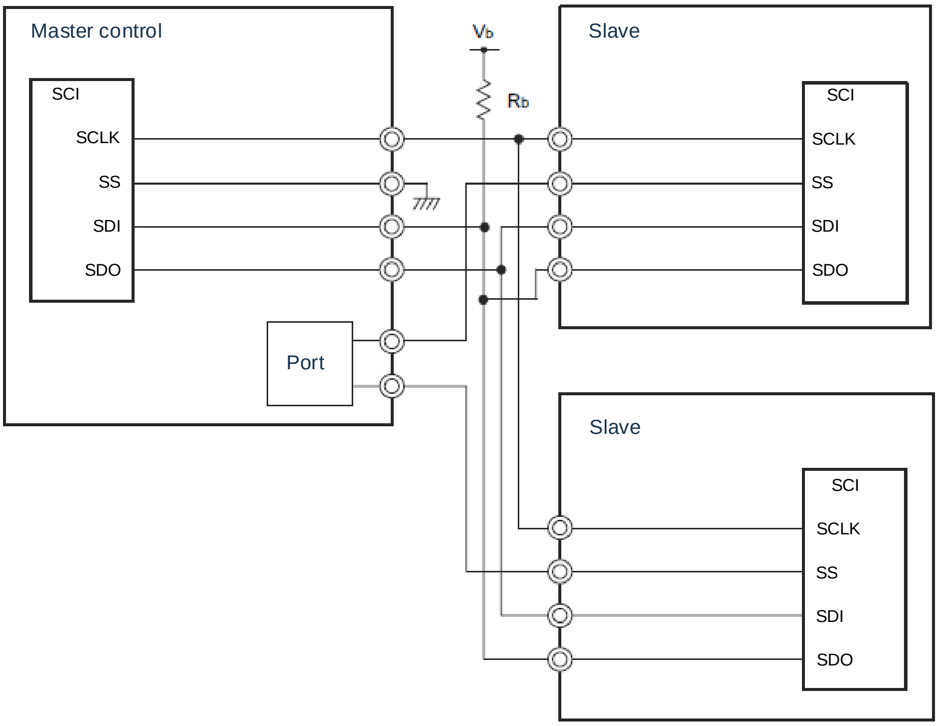
The slave select input function operates in three types of communication:

- Slave transmission (see 12.6.1).
- Slave reception (see 12.6.2).
- Slave transmission and reception (see 12.6.3).

By using the slave select input function, one master device can be connected to multiple slave devices for communication. The master device outputs a slave selection signal to the slave device (1) of the communication object, and each slave device determines whether it is selected as a communication object and controls the output of the SDO pin. When selected as a slave device for communication object, the SDO pin can communicate with the master device to send data; When a slave device is not selected as a communication object, the SDO pin becomes a high-level output, so in the environment where multiple slaves are connected, the SDO pin needs to be set to Nch-O.D and the node pulled up. In addition, even the serial clock of the input master device is not transmitted and received.

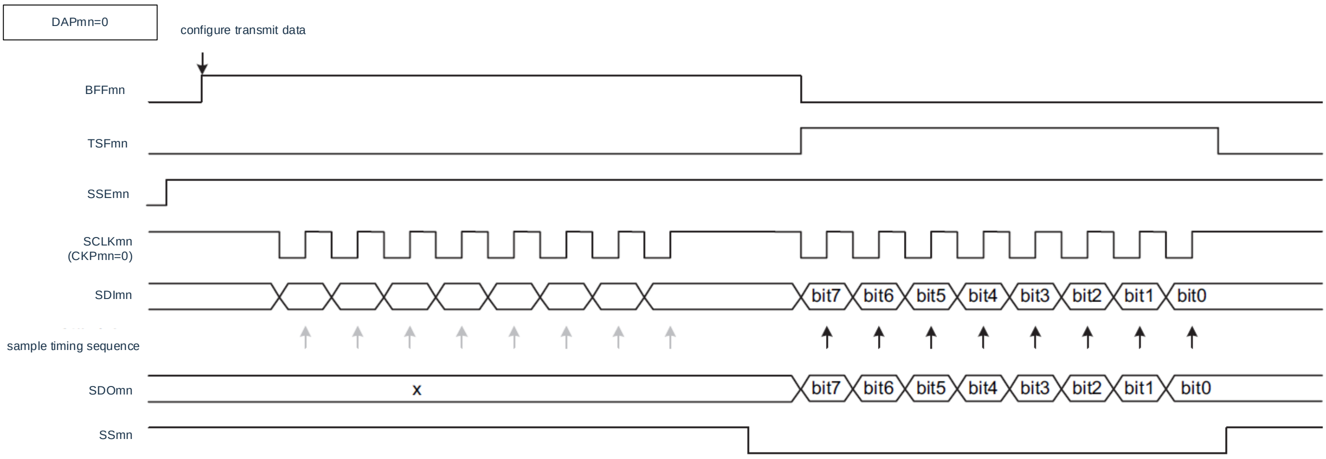
Note The slave select signal must be output through the operation of the port.

Figure 12-70 Structural example of the Slave select input function



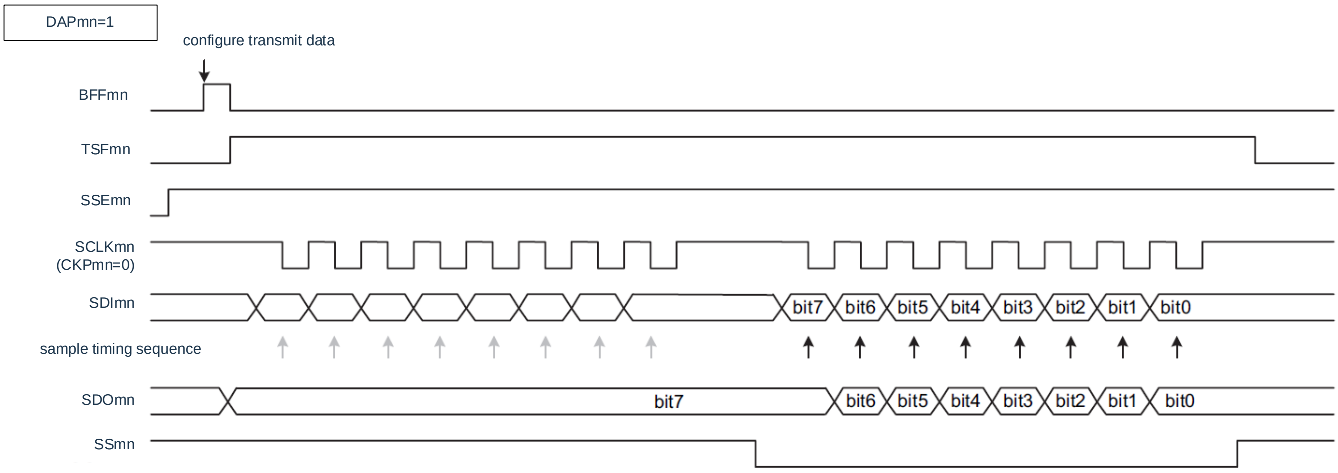
Note The SDO00 pin is selected for N-channel open-drain output mode.

Figure 12-71 Slave timing diagram of the select input function



During SSmn is high, even on the falling edge of the SCKmn (serial clock), no transmission occurs, and no sampling of received data synchronized with the rising edge is taken.

During SSmn low, the output data is synchronized (shifted) with the falling edge of the serial clock and received synchronously with the rising edge.



When the DAPmn bit is “1”, the initial data (bit7) is supplied to the data output if the transmit data is set during when SSmn is high. However, even the rising edge of the SCLK mn (serial clock) is not shifted, and the accepted data synchronized with the falling edge is not sampled. If SSmn goes low, the output data is synchronized (shifted) with the next rising edge and received synchronously with the falling edge.

Remark m: Unit number (m=0) n: Channel number (n=0).

12.6.1 Slave transmission

Slave transmission refers to the operation of this product to send data to other devices in the state of input transmission clock from other devices.

Slave Select Input function	SSPI00
Object channel	Channel 0 for SCI0
The pin used	SCLK00, SDO00, SS00
interrupt	INTSSPI00
	You can select either a transmit-end interrupt (single-pass mode) or a buffer null interrupt (continuous transfer mode).
Error detection flag	Only the Overflow Error Detection Flag (OVFmn).
The length of the transmitted data	7 or 8 bits
Transfer rate	Max.f_{MCK}/6[Hz]Note1, 2
Data phase	It can be selected via the DAPmn bit of the SCRmn register. • DAPmn=0: The data output starts when the serial clock starts running. • DAPmn=1: Starts data output half a clock before the serial clock starts running.
Clock phase	It can be selected via the CKPmn bit of the SCRmn register. • CKPmn=0: Normal • CKPmn=1: Inverted
Data direction	MSB first or LSB first
Slave Select Input function	You can choose to run the slave selection function.

Note 1 Because the external serial clock of the SCLK00 pin input is used internally, the maximum transfer rate is f_{MCK}/6[Hz].

2. Must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Note 1. f_{MCK}: The operating clock frequency of the object channel

2.m: Unit number (m=0) n: Channel number (n=0).

(1) Register setting

Figure 12-72 Example of register settings when slave select input function (SSPI00) slave transmits (1/2)

(a) serial mode register mn (SMRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn	CCSmn						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 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			DLSmn1	DLSmn0
	1	0	0/1	0/1	0	0	0	0	0/1	0	0	0	0	1	1	0/1

data and clock phase selection (details refer to "19.3 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
0: 7 bit data length
1: 8 bit data length

(c) serial data regsiter mn (SDRmn) (low 8 bit: SIOp)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	0000000 baud rate configuration							0	transmit data configuration							

SIOp (bits 7–0)

(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

(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													SOEm3	SOEm2	SOEm1	SOEm0
	0	0	0	0	0	0	0	0	0	0	0	0	×	×	×	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 be set (initial value).
×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).
0/1: Set "0" or "1" according to the user's purpose.

Figure 12-72 Example of register settings when slave select input function (SSPI00) slave transmits (2/2)

(f) serial channel start register m (SSm) …. Only set bit of target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	SSm3 ×	SSm2 ×	SSm1 ×	SSm0 0/1

(g) input switch control register (ISC)…. This is controlled by SS00 pin of SSPI00 slave channel (channel 0 of unit 0).

	7	6	5	4	3	2	1	0
ISC	SSIE00 0/1	0	0	0	0	0	ISC1 0/1	ISC0 0/1

0: SS00 pin input invalid
1: SS00 pin input valid

Note 1.m: Unit number (m=0) n: Channel number (n=0) p: SSPI number (p=00)

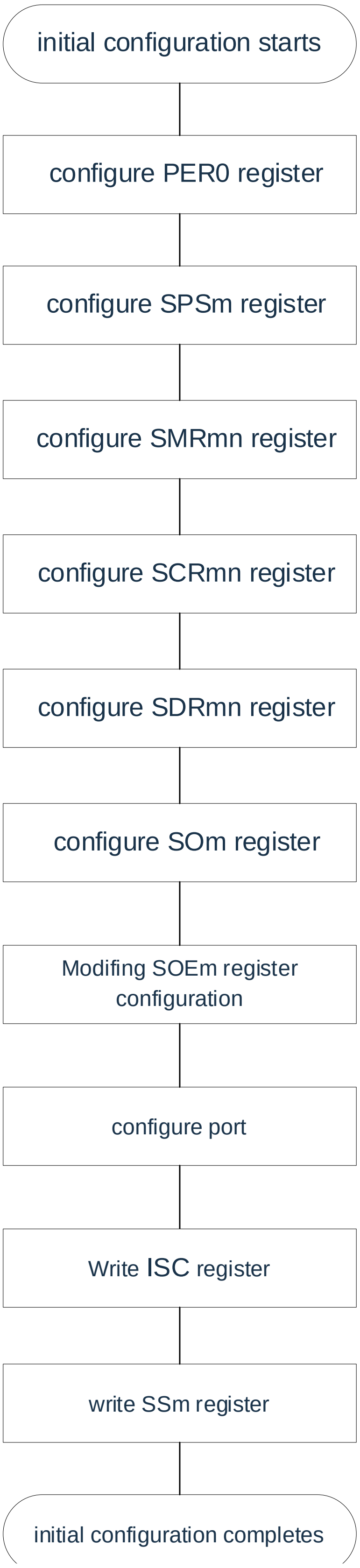
2. □ : Fixed in SSPI slave send mode. ▨ : Cannot be set (initial value).

×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).

0/1: Set “0” or “1” according to the user's purpose.

(2) Operation Steps

Figure 12-73 Initial setup steps for slave sending



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.

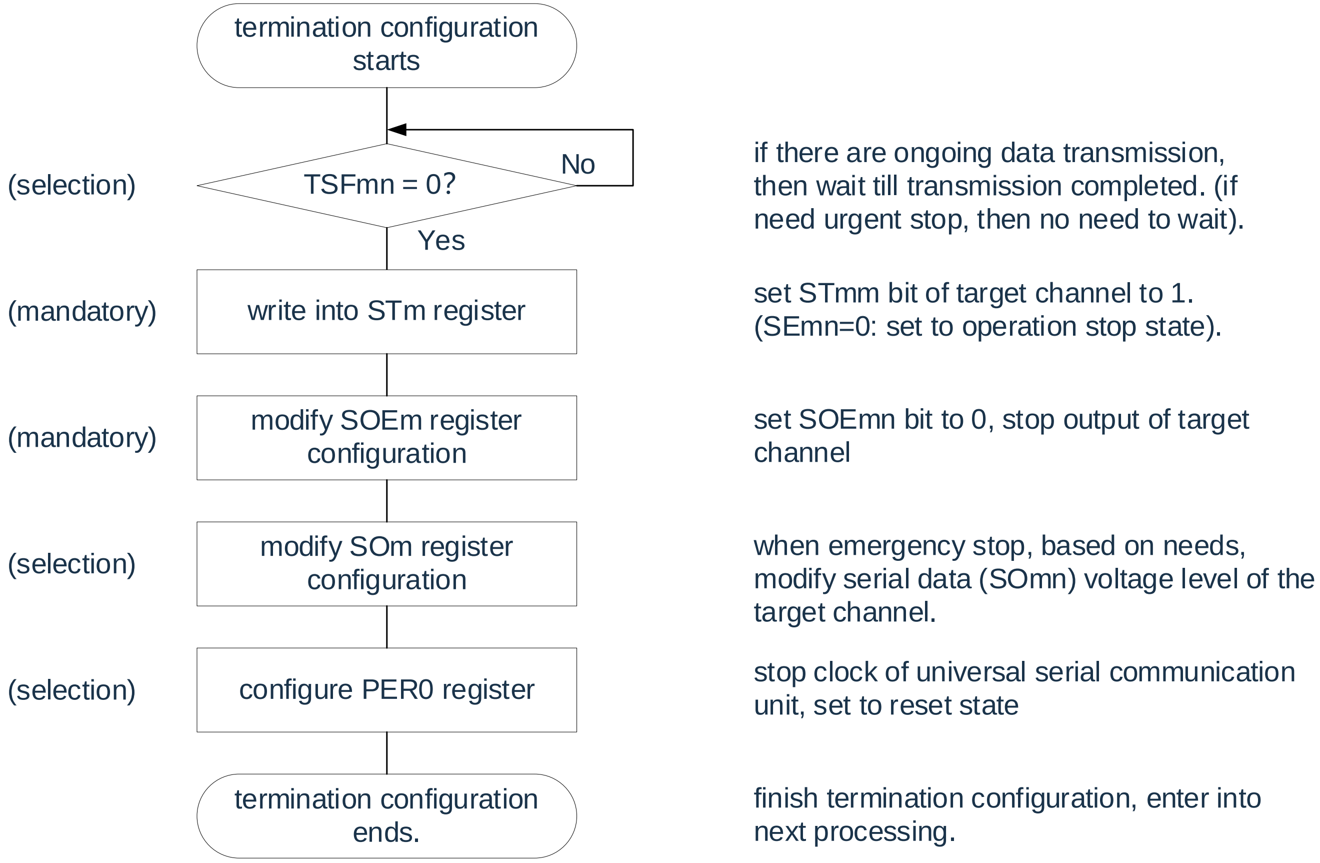
set SSIE00 bit to 1, enable channel 0 slave selection function operates.

set SSmn bit of target channel to "1": set to enable operation state).

complete initial configuration.
If transmit data to SIOp register (bit 7~0 of SDRmn register), wait for master device clock.

Remark m: Unit number (m=0) n: Channel number (n=0) p: SSPI number (p=00)

Figure 12-74 Stop step of slave send

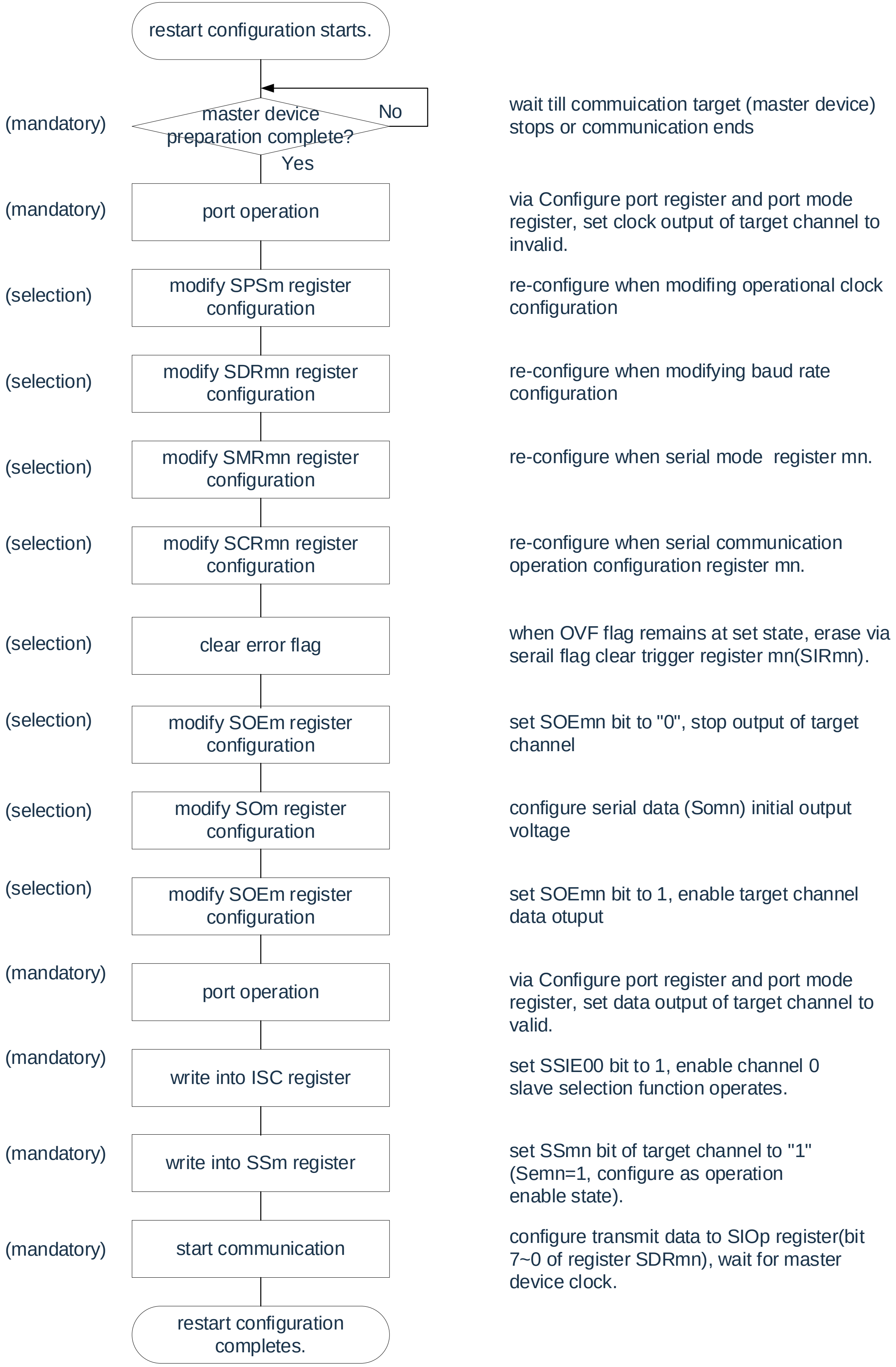

termination configuration starts
(selection)
TSFmn = 0?
No
Yes
if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).
(mandatory)
write into STm register
set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).
(mandatory)
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 the clock, you must wait until the communication object (the master device) stops or the communication is over to make the initial setting instead of starting the setting again.

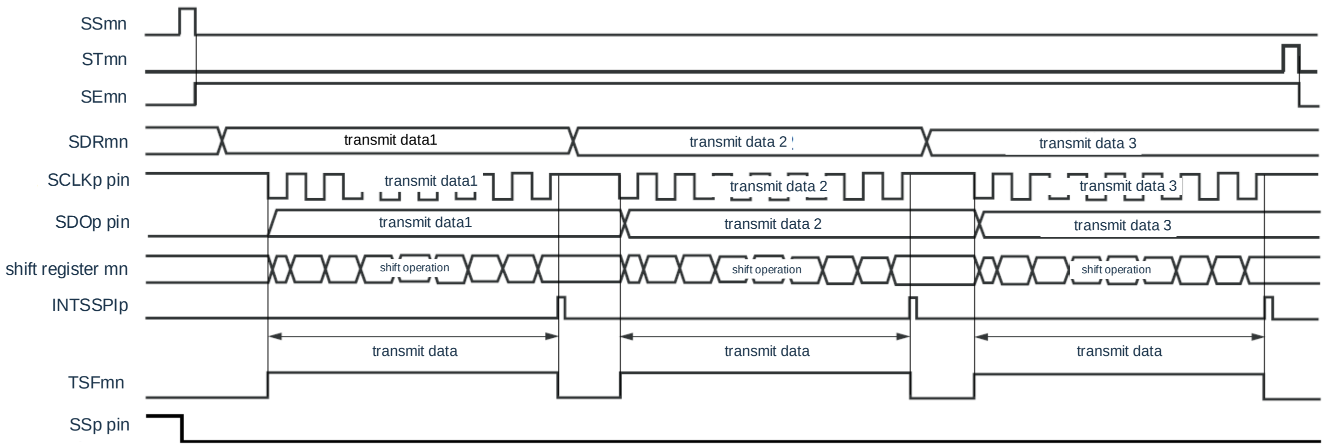
2.m: Unit number (m=0) n: Channel number (n=0) p: SSPI number (p=00).

Figure 12-75 Restart setup step of slave sending


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 SIOp register(bit 7~0 of register SDRmn), wait for master device clock.
restart configuration completes.

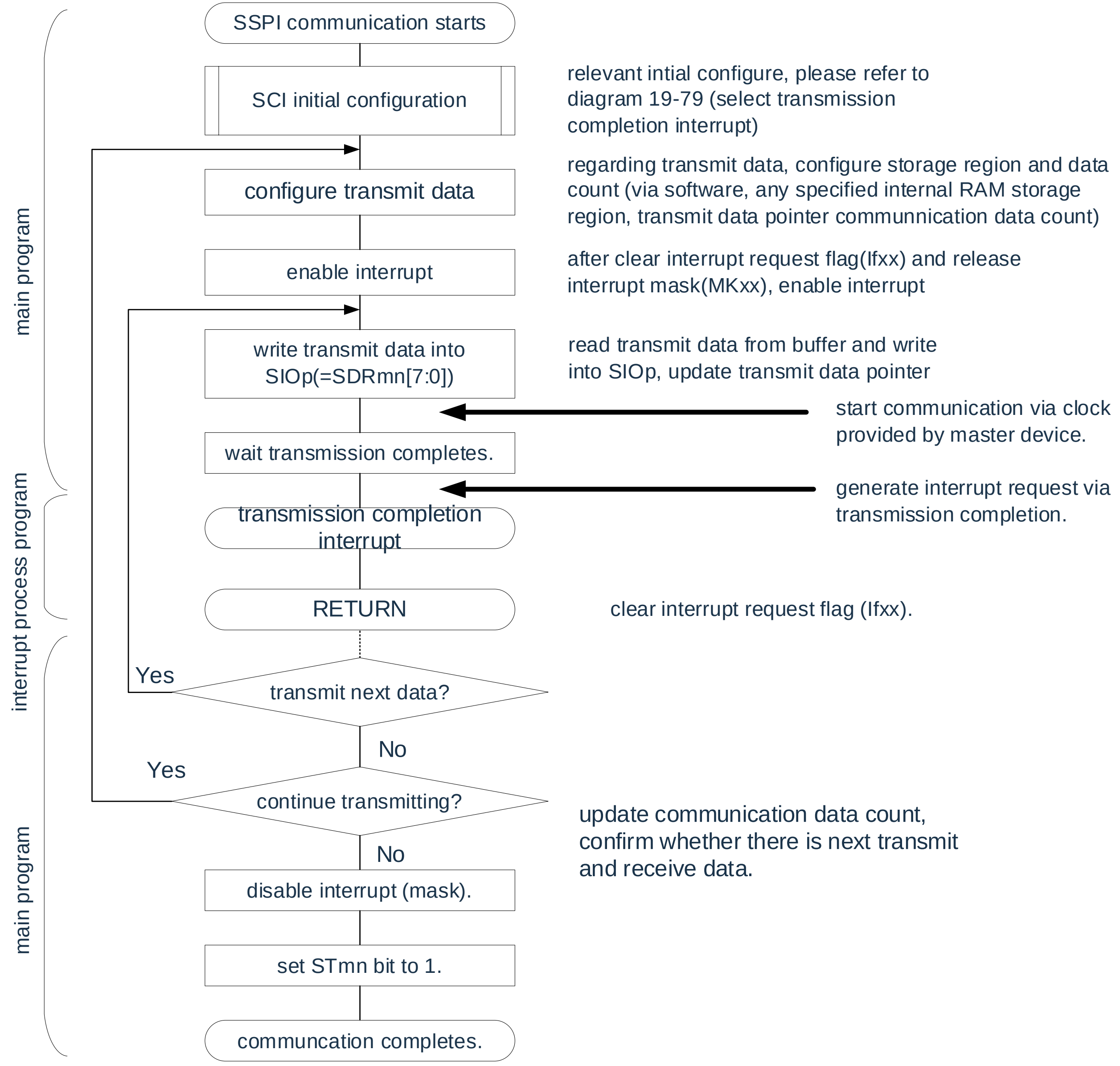
(3) Process flow (single send mode)

Figure 12-76 Timing diagram of slave send (single send mode) (Type 1: DAPmn=0, CKPmn=0).



Remark m: Unit number (m=0) n: Channel number (n=0) p: SSPI number (p=00)

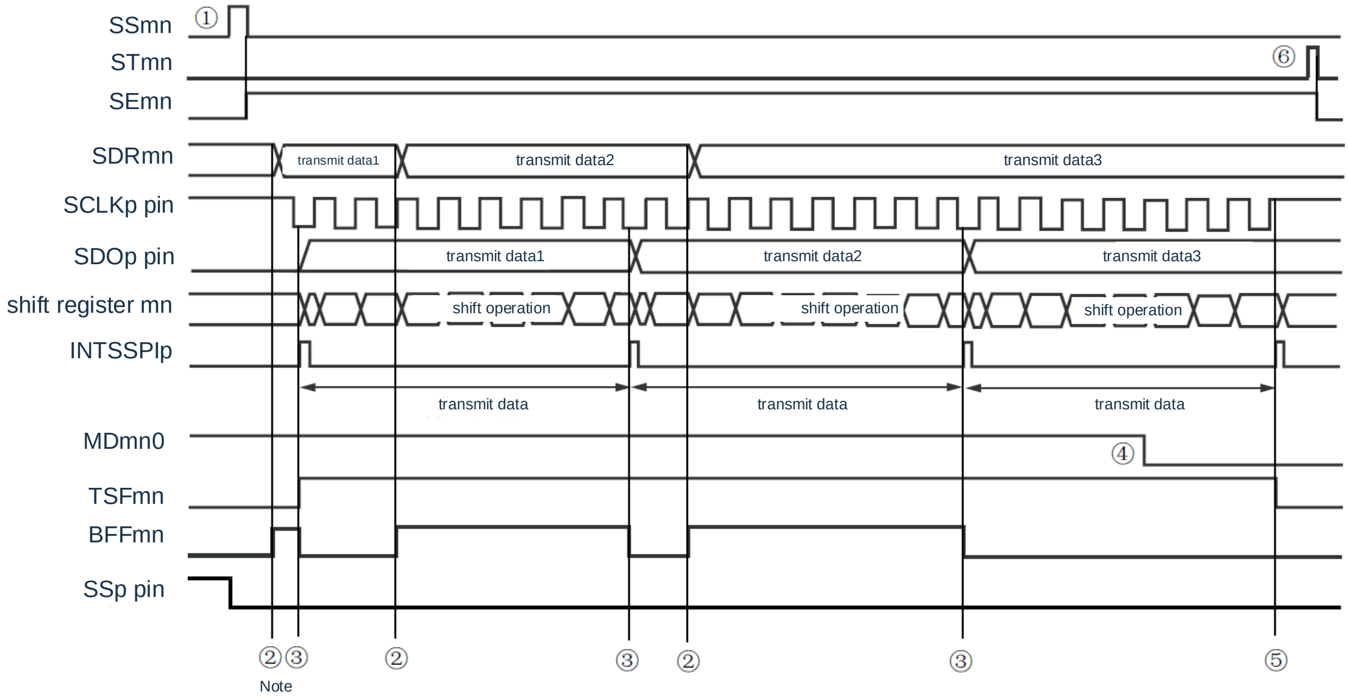
Figure 12-77 Flowchart of slave send (single send mode).


main program
interrupt process program
main program
SSPI communication starts
SCI initial configuration
configure transmit data
enable interrupt
write transmit data into SIOp(=SDRmn[7:0])
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.
relevant intial configure, please refer to diagram 19-79 (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 SIOp, update transmit data pointer
start communication via clock provided by master device.
generate interrupt request via transmission completion.
clear interrupt request flag (Ifxx).
update communication data count, confirm whether there is next transmit and receive data.

Remarks m: Unit number (m=0)n: Channel number (n=0)p:SSPI number (p=00)

(4) Process flow (continuous send mode)

Figure 12-78 Timing diagram of slave transmit (continuous transmit mode) (type 1: DAPmn=0, CKPmn=0).

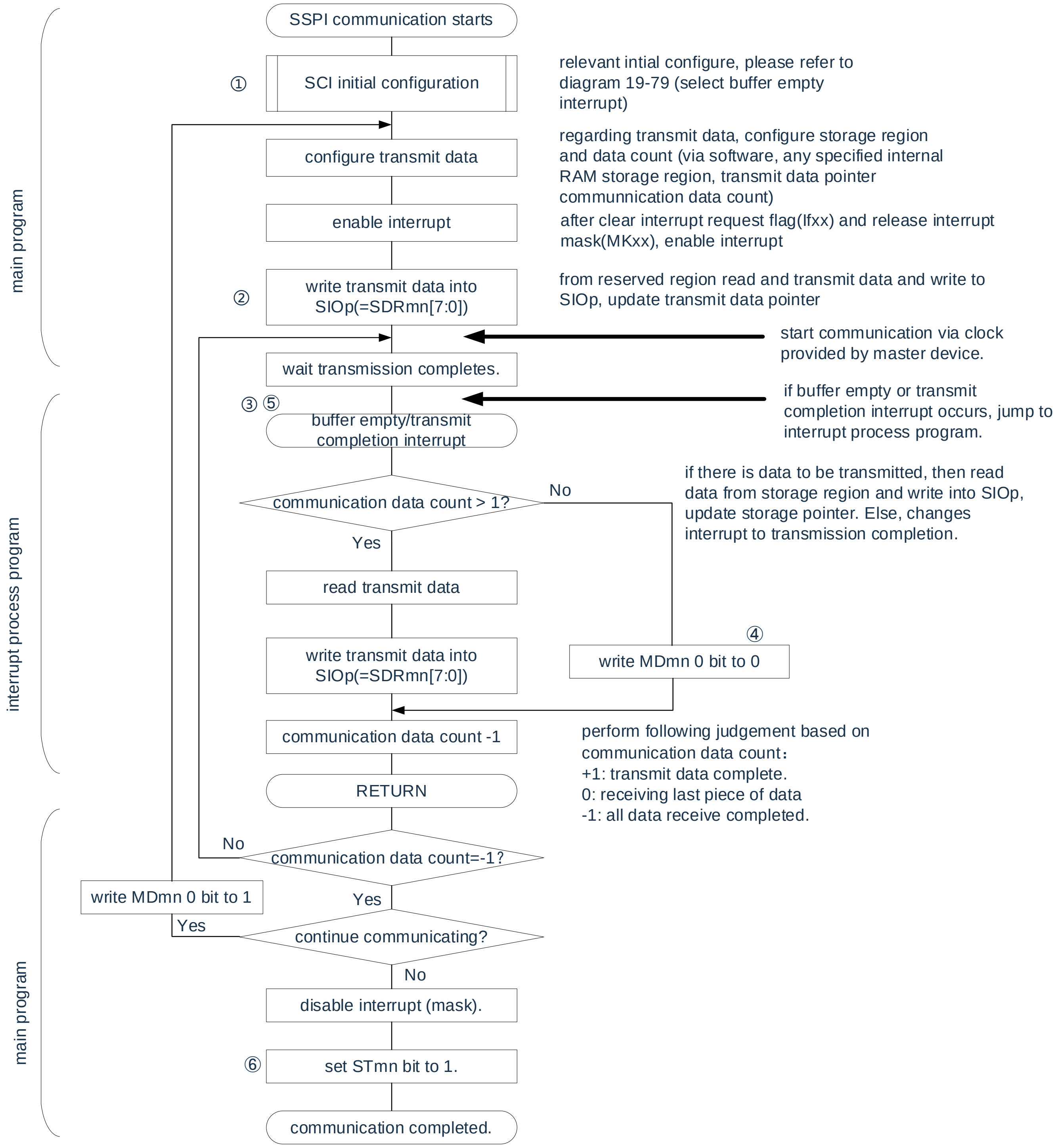


Note If the BFFmn bit of the serial status register mn (SSRmn) is “1” (when valid data is saved in the serial data register mn (SDRmn)) is given The SDRmn register writes the transmitted data and overrides the transmitted data.

Notice The MDmn0 bit of the serial mode register mn (SMRmn) can be overridden even during operation. However, it must be overridden before the last bit can be transferred.

Remark m: Unit number (m=0) n: Channel number (n=0) p: SSPI number (p=00)

Figure 12-79 Flowchart of slave transmission (continuous send mode)


main program
SSPI communication starts
①
SCI initial configuration
relevant intial configure, please refer to diagram 19-79 (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 SIOp(=SDRmn[7:0])
from reserved region read and transmit data and write to SIOp, update transmit data pointer
start communication via clock provided by master device.
wait transmission completes.
interrupt process program
③ ⑤
buffer empty/transmit completion interrupt
if buffer empty or transmit completion interrupt occurs, jump to interrupt process program.
communication data count > 1?
No
Yes
if there is data to be transmitted, then read data from storage region and write into SIOp, update storage pointer. Else, changes interrupt to transmission completion.
read transmit data
④
write transmit data into SIOp(=SDRmn[7:0])
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 12-78.

2.m: Unit number (m=0) n: Channel number (n=0) p: SSPI number (p=00).

12.6.2 Slave receiving

Slave reception refers to the operation of this product to receive data from other devices in the state of transmitting clocks from other devices.

Slave Select Input function	SSPI00
Object channel	Channel 0 for SCI0
The pin used	SCLK00, SDI00, SS00
interrupt	INTSSPI00
	Limited to end-of-transfer interrupts (disable setting buffer null interrupts).
Error detection flag	Only the Overflow Error Detection Flag (OVFmn).
The length of the transmitted data	7 or 8 bits
Transfer rate	Maxf$_{MCK}$/6[Hz]$^{Note1, 2}$
Data phase	It can be selected via the DAPmn bit of the SCRmn register. • DAPmn=0: The data output starts when the serial clock starts running. • DAPmn=1: Starts data output half a clock before the serial clock starts running.
Clock phase	It can be selected via the CKPmn bit of the SCRmn register. • CKPmn=0: Normal • CKPmn=1: Inverted
Data direction	MSB first or LSB first
Slave Select Input function	You can select the operation of the Slave select input function.

Note 1. The maximum transfer rate is fMCK/6 [Hz] because the external serial clock input to the SCLK00 pin is sampled internally and then used.

2. It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Note 1. f$_{MCK}$: The operating clock frequency of the object channel
2.m: Unit number (m=0) n: Channel number (n=0).

(1) Register setting

Figure 12-80 Select Input Function (SSPI00) Example of register setting content when slave receive (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 registermn 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	0	1	DLSmn1 1	DLSmn0 0/1

data and clock phase selection (details refer to "19.3 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
0: 7 bit data length
1: 8 bit data length

(c) serial data regsiter mn (SDRmn) (low 8 bit: SIOp)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	0000000 baud rate configuration							0	received data							
									SIOp							

(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. □ : Fixed in slave receive mode. ▨: Cannot be set (initial value).
×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).
0/1: Set "0" or "1" according to the user's purpose.

Figure 12-81 Select Input Function (SSPI00) Example of register settings when slave receive (2/2).

(f) serial channel start register m (SSm) …. Only set bit of target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm													SSm3	SSm2	SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	×	×	×	0/1

(g) input switch control register (ISC)…. This is controlled by SS00 pin of SSPI00 slave channel (channel 0 of unit 0).

	7	6	5	4	3	2	1	0
ISC	SSIE00						ISC1	ISC0
	0/1	0	0	0	0	0	0/1	0/1

0: SS00 pin input invalid
1: SS00 pin input valid

Note 1.m: Unit number (m=0)n: Channel number (n=0)p:SSPI number (p=00)

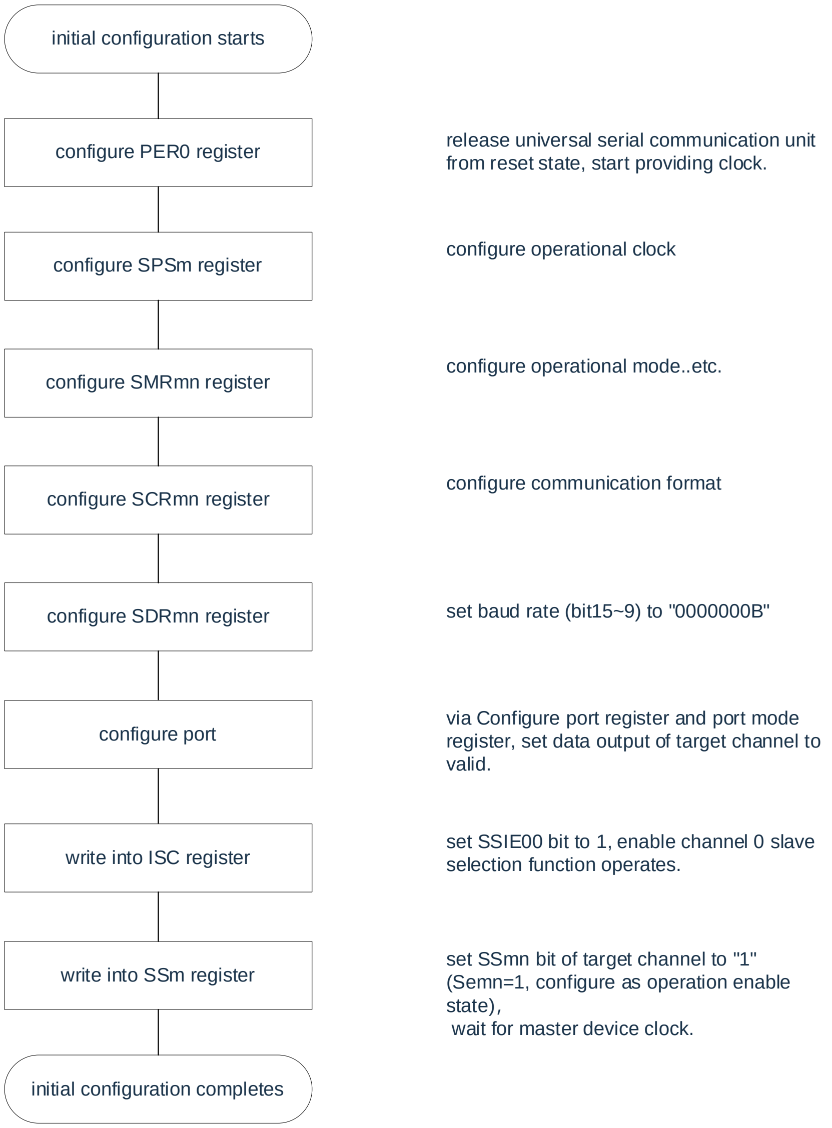
2. □ : Fixed in Slave Receive mode. ▨ : Cannot be set (initial value).

×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).

0/1: Set “0” or “1” according to the user's purpose.

(2) Operation Steps

Figure 12-82 Initial setup step of slave reception



release universal serial communication unit from reset state, start providing clock.

configure operational clock

configure operational mode..etc.

configure communication format

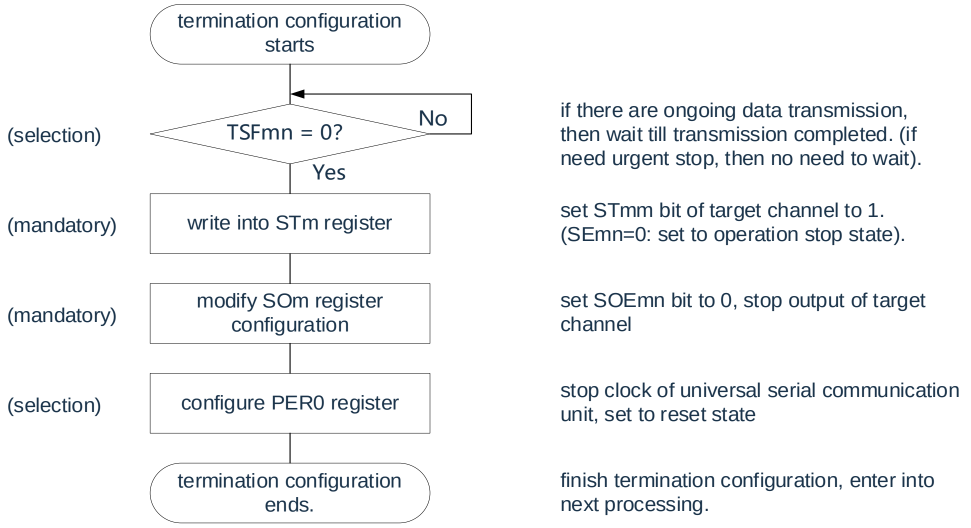
set baud rate (bit15~9) to "0000000B"

via Configure port register and port mode register, 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" (Semn=1, configure as operation enable state),
wait for master device clock.

Figure 12-83 Stop step of slave receiving



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).

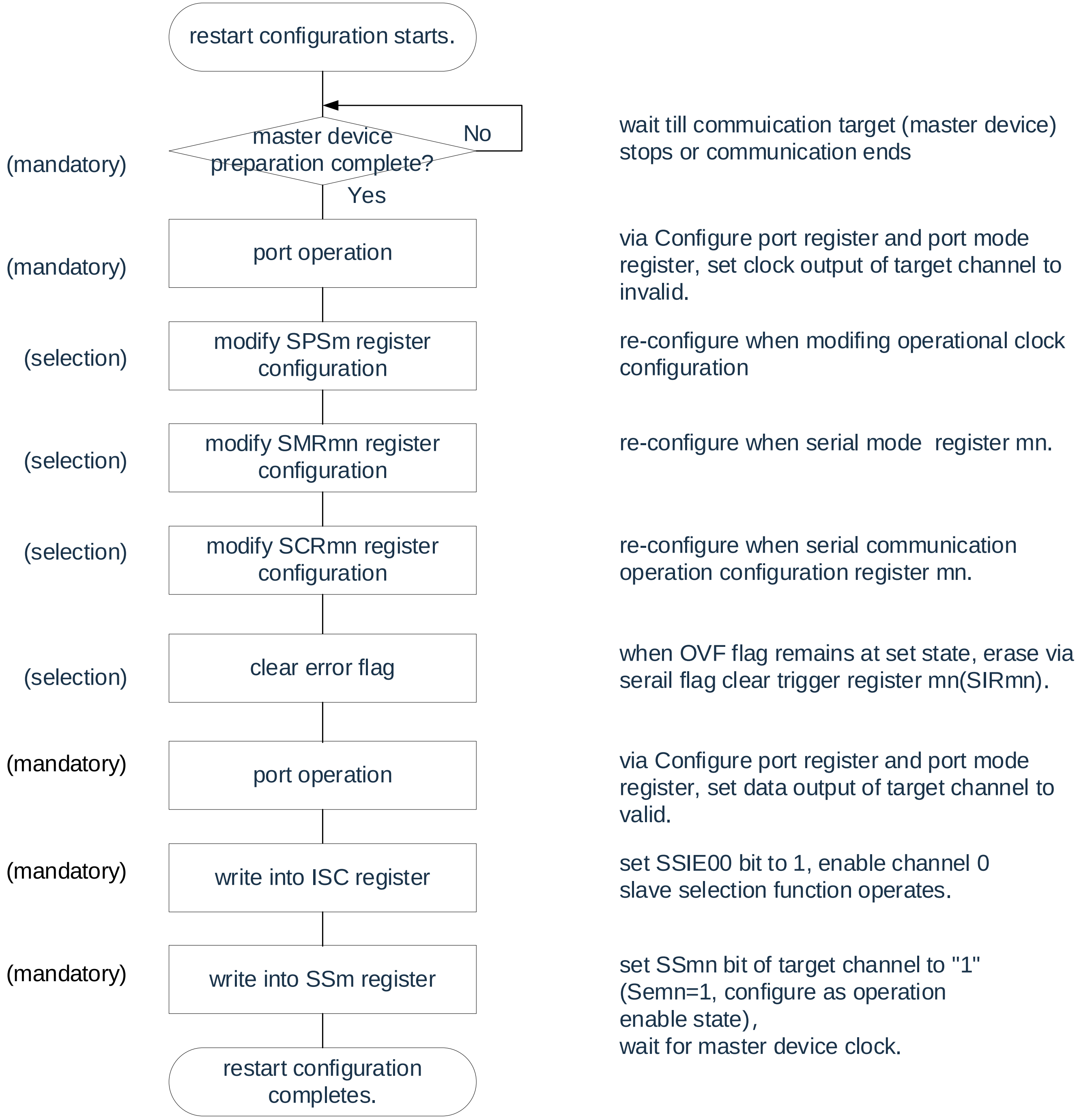
set SOEmn bit to 0, stop output of target channel

stop clock of universal serial communication unit, set to reset state

finish termination configuration, enter into next processing.

Remark m: Unit number (m=0) n: Channel number (n=0) p: SSPI number (p=00)

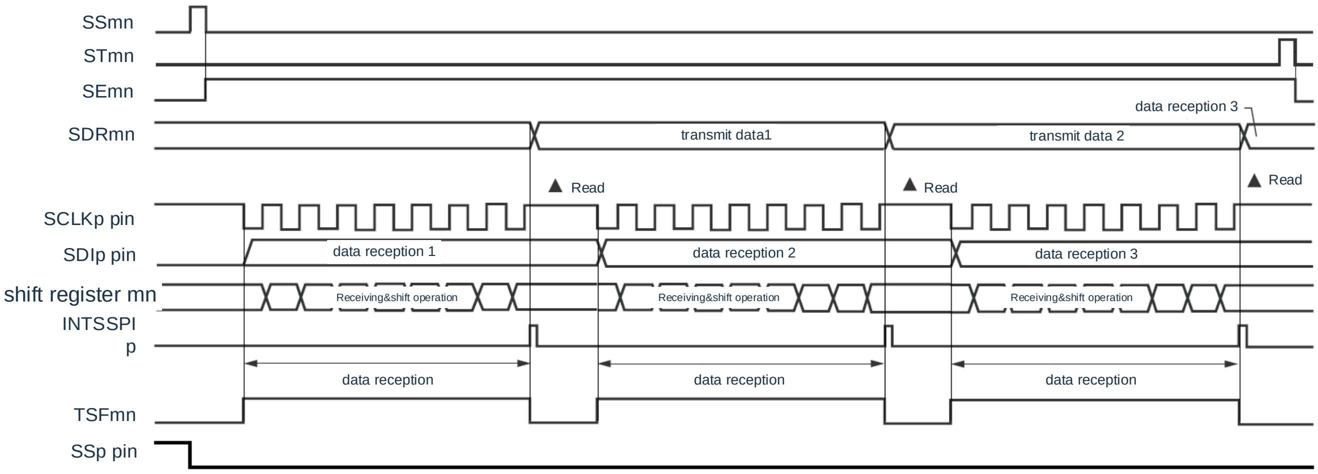
Figure 12-84 Restart setup step of slave reception


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).
(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.
restart configuration completes.

Remark m: Unit number (m=0) n: Channel number (n=0) p: SSPI number (p=00)

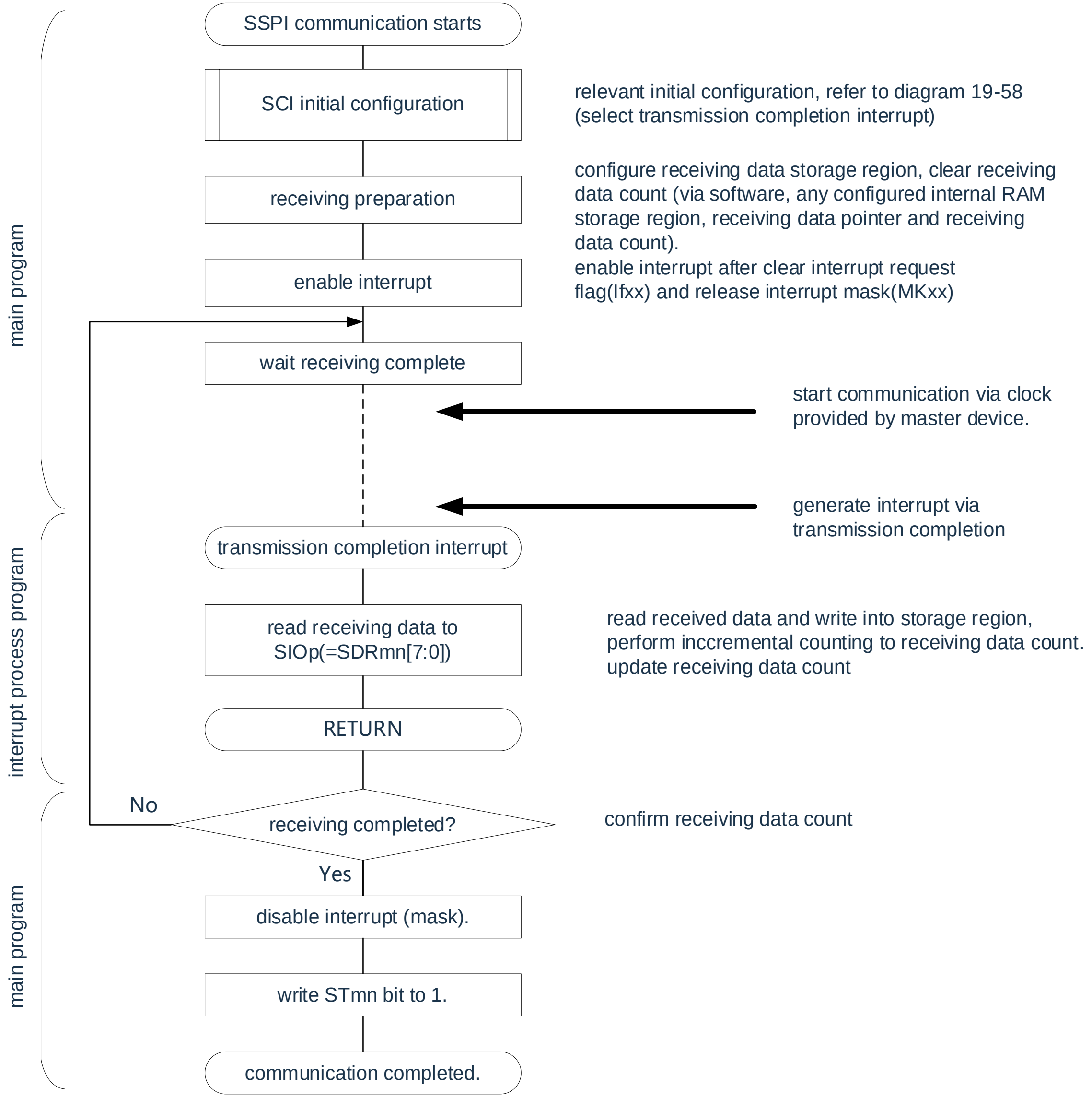
(3) Processing flow (single receive mode).

Figure 12-85 Timing diagram of slave receive (single receive mode) (Type 1: DAPmn=0, CKPmn=0)



Remark m: Unit number (m=0) n: Channel number (n=0) p: SSPI number (p=00)

Figure 12-86 Flowchart of slave receive (single receive mode)


SSPI communication starts
SCI initial configuration
receiving preparation
enable interrupt
wait receiving complete
transmission completion interrupt
read receiving data to SIOp(=SDRmn[7:0])
RETURN
receiving completed?
No
Yes
disable interrupt (mask).
write STmn bit to 1.
communication completed.
main program
interrupt process program
main program
relevant initial configuration, refer to diagram 19-58 (select transmission completion interrupt)
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 after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx)
start communication via clock provided by master device.
generate interrupt via transmission completion
read received data and write into storage region, perform inccremental counting to receiving data count. update receiving data count
confirm receiving data count

12.6.3 Slave transmission and reception

Slave transmit and receive refers to the operation of this product and other devices for data transmission and reception in the state of transmitting clocks from other devices.

Slave Select Input function	SSPI00
Object channel	Channel 0 for SCI0
The pin used	SCLK00, SDI00, SDO00, SS00
interrupt	INTSSPI00
	You can select either a transmit-end interrupt (single-pass mode) or a buffer null interrupt (continuous transfer mode).
Error detection flag	Only the Overflow Error Detection Flag (OVFmn).
The length of the transmitted data	7 or 8 bits
Transfer rate	Max.fMCK/6[Hz]Note1, 2
Data phase	It can be selected via the DAPmn bit of the SCRmn register. • DAPmn=0: The data output starts when the serial clock starts running. • DAPmn=1: Starts data output half a clock before the serial clock starts running.
Clock phase	It can be selected via the CKPmn bit of the SCRmn register. • CKPmn=0: Normal • CKPmn=1: Inverted
Data direction	MSB first or LSB first
Slave Select Input function	You can select the operation of the Slave select input function.

Note 1. The maximum transfer rate is fMCK/6 [Hz] because the external serial clock input to the SCLK00 pin is sampled internally and then used.

2. It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Note 1. f_{MCK}: The operating clock frequency of the object channel

2. m: Unit number (m=0) n: Channel number (n=0).

(1) Register setting

Figure 12-87 Slave select input function (SSPI00) Example of register setting content when slave send and receive (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	0	1	DLSmn1 1	DLSmn0 0/1

data and clock phase selection (details refer to "19.3 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
0: 7 bit data length
1: 8 bit data length

(c) serial data regsiter mn (SDRmn) (low 8 bit: SIOp)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	0000000 baud rate configuration							0	configuration of transmit data/received data register							

SIOp

(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 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 ×	SOEm2 ×	SOEm1 ×	SOEm0 0/1

Note Data must be sent to the SIOp register settings before the master device starts the output clock.

Notice 1.m: Unit number (m=0)n: Channel number (n=0)p:SSPI number (p=00)

2. □ : Fixed in Slave Receive mode. ▨ : Cannot be set (initial value).
×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).
0/1: Set “0” or “1” according to the user's purpose.

Figure 12-87 Slave select input function (SSPI00) Example of register setting content when slave send and receive (2/2)

(f) serial channel start register m (SSm) …. Only set bit of target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm													SSm3	SSm2	SSm1	SSm0
	0	0	0	0	0	0	0	0	0	0	0	0	×	×	×	0/1

(g) input switch control register (ISC)…. This is controlled by SS00 pin of SSPI00 slave channel (channel 0 of unit 0).

	7	6	5	4	3	2	1	0
ISC	SSIE00						ISC1	ISC0
	0/1	0	0	0	0	0	0/1	0/1

0: SS00 pin input invalid
1: SS00 pin input valid

Note Data must be sent to the SIOp register settings before the master device starts the output clock.

Notice 1.m: Unit number (m=0) n: Channel number (n=0)p:SSPI number (p=00)

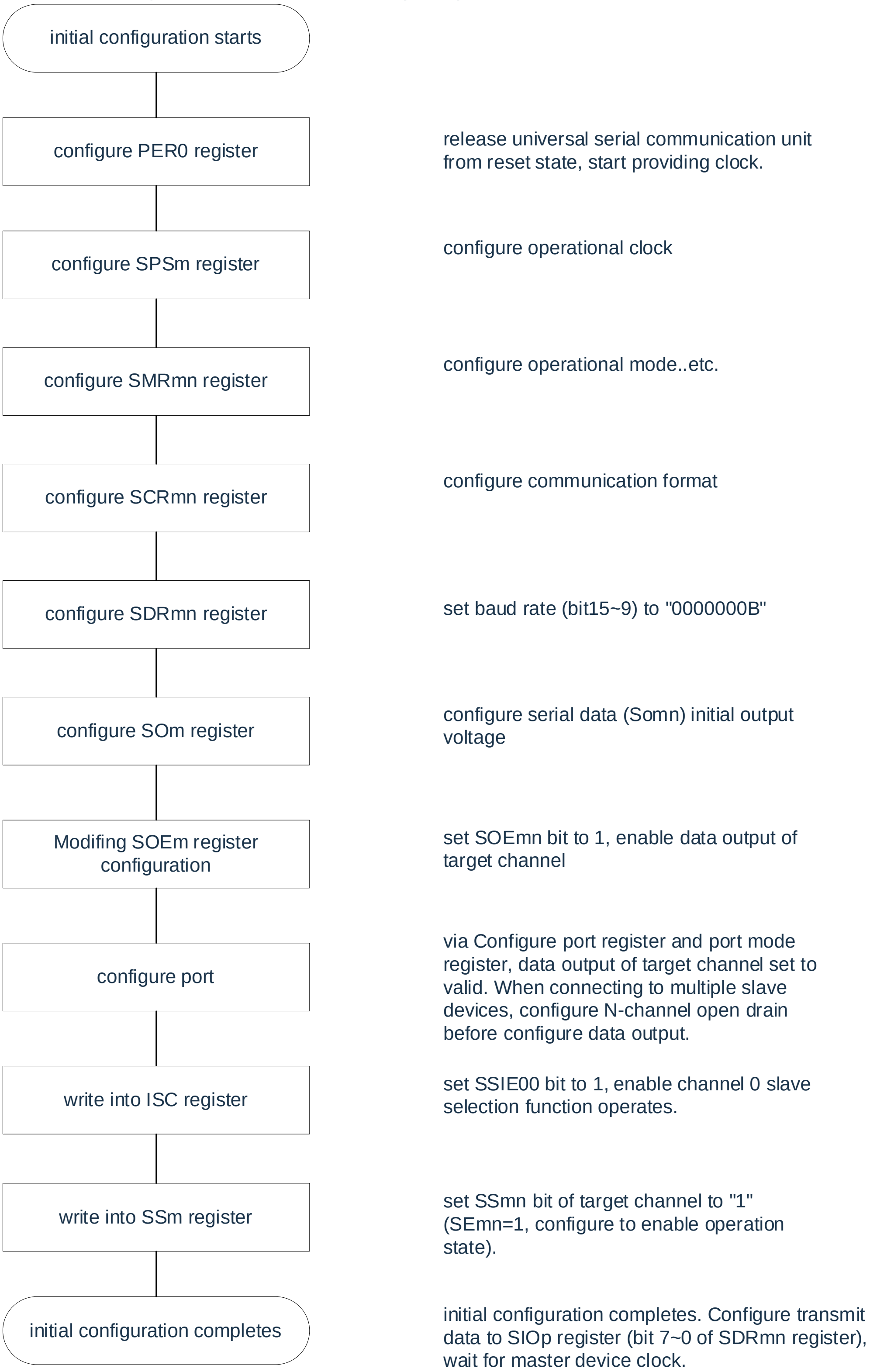
2. □ : Fixed in Slave Receive mode. ▨ : Cannot be set (initial value).

×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).

0/1: Set “0” or “1” according to the user's purpose.

(2) Operation Steps

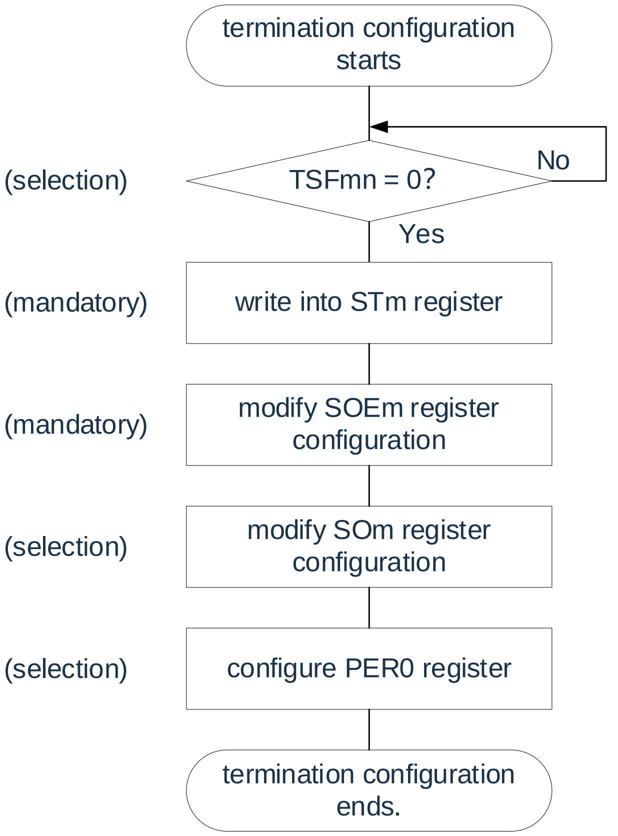
Figure 12-88 Initial setup steps for slave send and receive


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, data output of target channel set to valid. When connecting to multiple slave devices, configure N-channel open drain before configure data output.
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 to enable operation state).
initial configuration completes
initial configuration completes. Configure transmit data to SIOp register (bit 7~0 of SDRmn register), wait for master device clock.

Note Data must be sent to the SIOp register settings before the master device starts the output clock.

Remark m: Unit number (m=0) n: Channel number (n=0) p: SSPI number (p=00)

Figure 12-89 Stop steps of slave send and receive


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

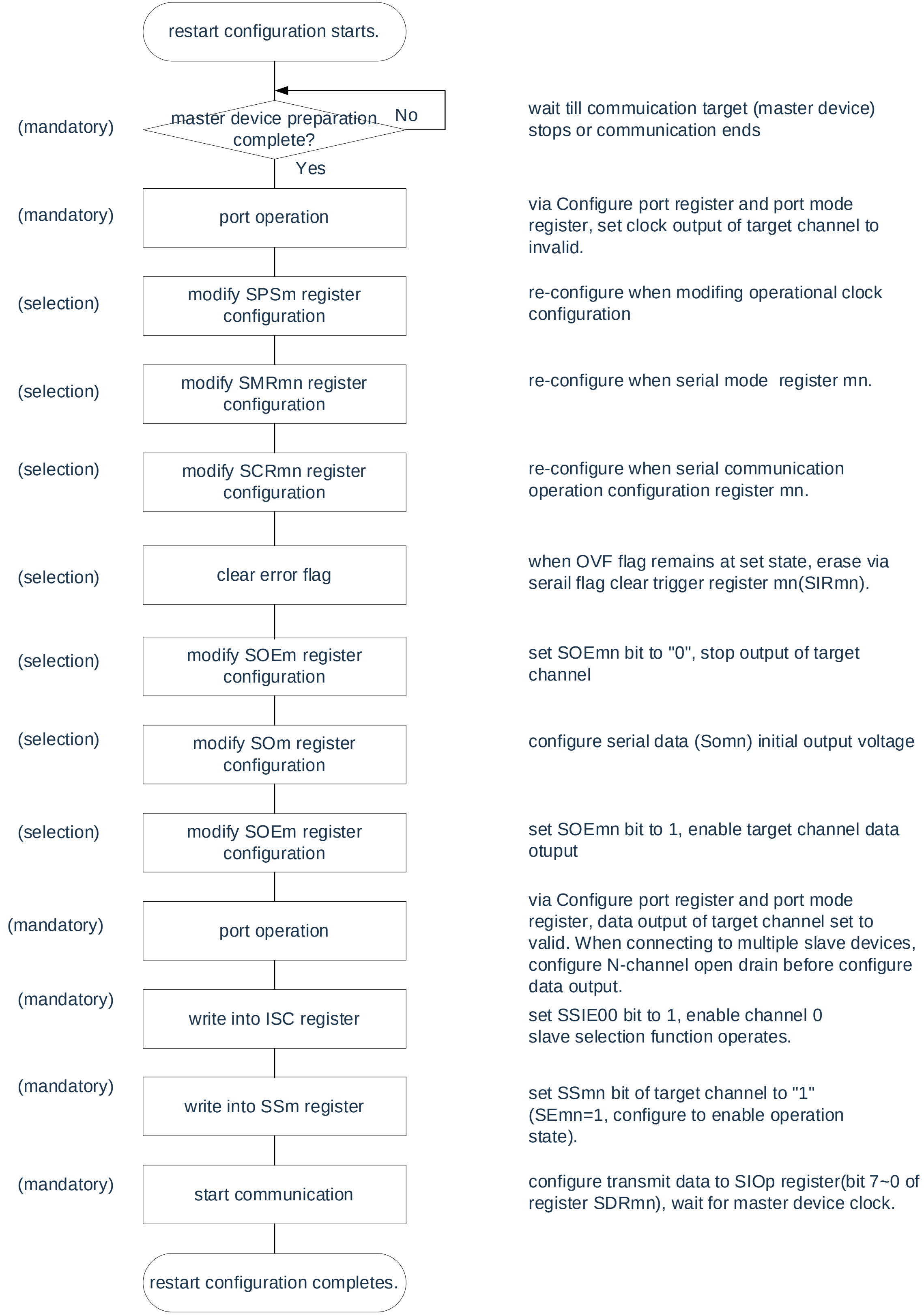
when emergency stop, based on needs, modify serial data (SOmn) voltage level of the target channel.

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 12-90 Restart setup steps of Slave send and receive


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
(selection)
modify SOEm register configuration
(selection)
modify SOm register configuration
(selection)
modify SOEm register configuration
(mandatory)
port operation
(mandatory)
write into ISC register
(mandatory)
write into SSm register
(mandatory)
start communication
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).

set SOEmn bit to "0", stop output of target channel

configure serial data (Somn) initial output voltage

set SOEmn bit to 1, enable target channel data otuput

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).

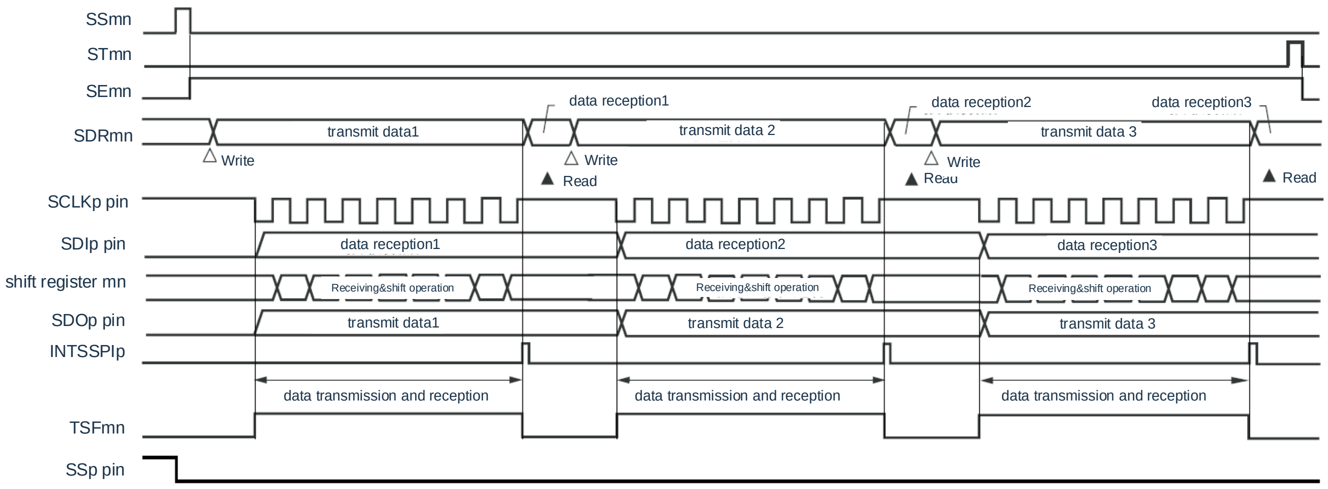
configure transmit data to SIOp register(bit 7~0 of register SDRmn), wait for master device clock.

Note 1 Before the master device starts to output the clock, data must be sent to the SIOp register settings.

2. If you override PER0 in the abort setting to stop the clock, you must wait until the communication object (the master device) stops or the communication is over to make the initial setting instead of starting the setting again.

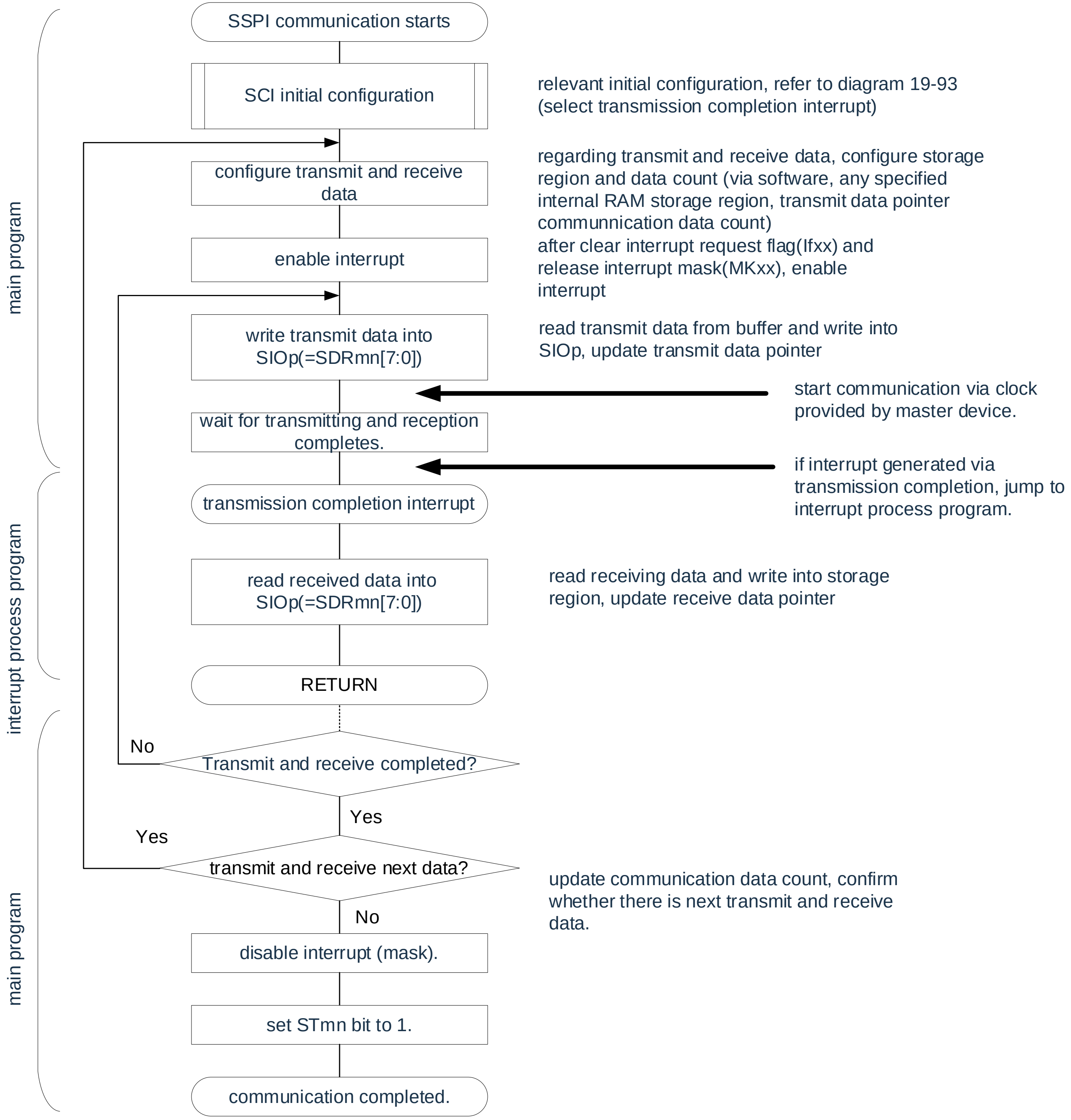
(3) Processing flow (single send and receive mode)

Figure 12-91 Timing diagram of slave transmit and receive (single-send and receive mode) (Type 1: DAPmn=0, CKPmn=0).



Remark m: Unit number (m=0) n: Channel number (n=0) p: SSPI number (p=00)

Figure 12-92 Flowchart of slave send and receive (single send and receive mode)

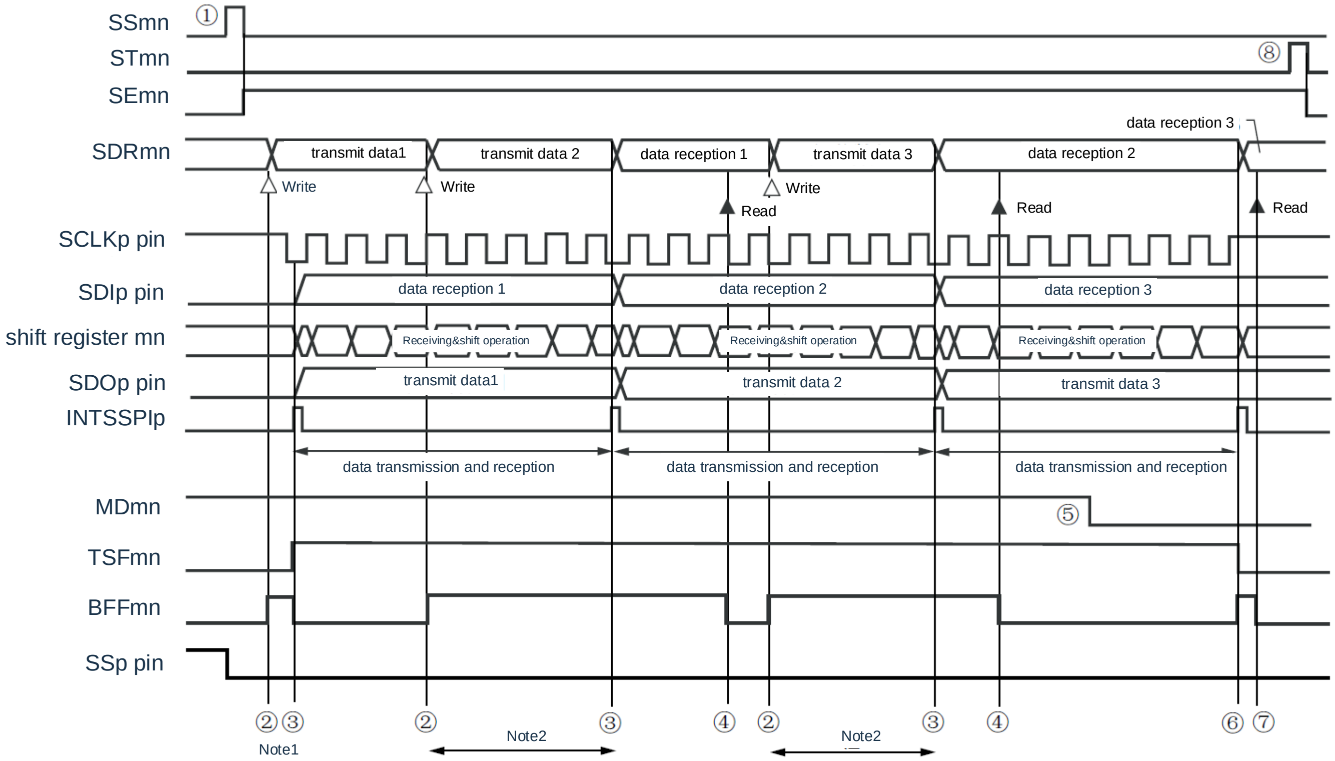

main program
interrupt process program
main program
SSPI communication starts
SCI initial configuration
relevant initial configuration, refer to diagram 19-93 (select transmission completion interrupt)
configure transmit and receive data
regarding transmit and receive data, configure storage region and 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 SIOp(=SDRmn[7:0])
read transmit data from buffer and write into SIOp, update transmit data pointer
wait for transmitting and reception completes.
start communication via clock provided by master device.
if interrupt generated via transmission completion, jump to interrupt process program.
transmission completion interrupt
read received data into SIOp(=SDRmn[7:0])
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.

Note Data must be sent to the SIOp register settings before the master device starts the output clock.

Remark m: Unit number (m=0) n: Channel number (n=0) p: SSPI number (p=00)

(4) Process flow (continuous send and receive mode)

Figure 12-93 Timing diagram of slave transmit and receive (continuous transmit and receive mode) (type 1: DAPmn=0, CKPmn=0)

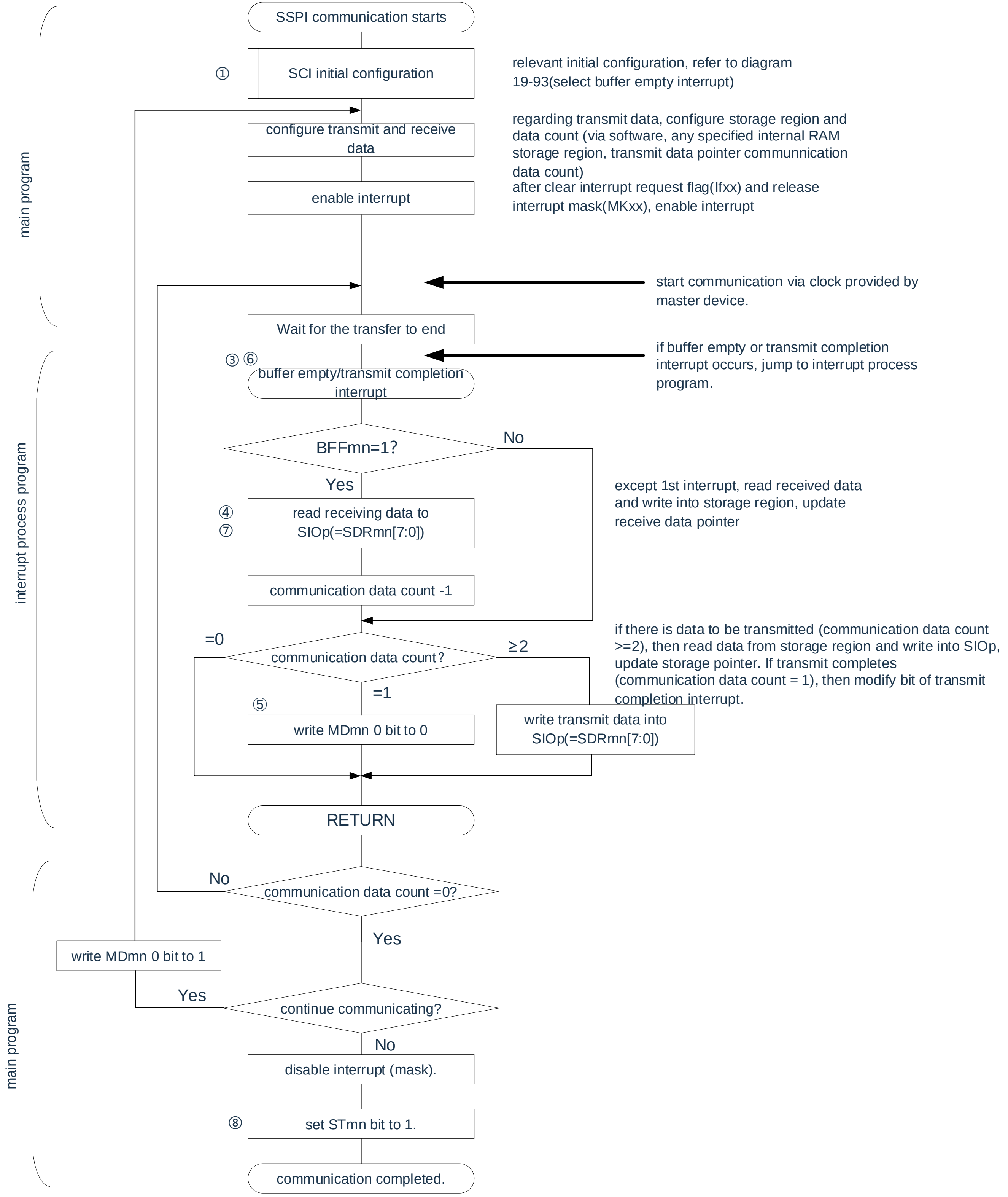


Note 1. If the BFFmn bit of the serial status register mn (SSRmn) is "1" (valid data is saved in the serial data register mn (SDRmn) to write the send data to the SDRmn register, and rewrite the sent data.
2. If the SDRmn register is read during this period, the transmitted data can be read. At this point, the transfer run is not affected.

Note The MDmn0 bit of the serial mode register mn (SMRmn) can be overridden even during operation. However, in order to catch up with the end of the transmission interruption of the last transmitted data, it must be overwritten before the last bit of transmission begins.

Note 1. (1) to (8) in the figure corresponds to (1) to (8) in the "Figure 12-94".
2.m: Unit number (m=0) n: Channel number (n=0) p: SSPI number (p=00).

Figure 12-94 Flowchart of slave send and receive (continuous send and receive mode)


main program
interrupt process program
main program
SSPI communication starts
①
SCI initial configuration
relevant initial configuration, refer to diagram 19-93(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 communnication data count)
enable interrupt
after clear interrupt request flag(Ifxx) and release interrupt mask(MKxx), enable interrupt
start communication via clock provided by master device.
Wait for the transfer to end
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 SIOp(=SDRmn[7:0])
except 1st interrupt, read received data and write into storage region, update receive data pointer
communication data count -1
=0
communication data count?
≥2
if there is data to be transmitted (communication data count >=2), then read data from storage region and write into SIOp, update storage pointer. If transmit completes (communication data count = 1), then modify bit of transmit completion interrupt.
=1
⑤
write MDmn 0 bit to 0
write transmit data into SIOp(=SDRmn[7:0])
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.

Note Data must be sent to the SIOp register settings before the master device starts the output clock.

Remark 1. (1) to (8) in the figure corresponds to (1) to (8) in the “Timing diagram of Figure 12-93”.

2.m: Unit number (m=0) n: Channel number (n=0) p: SSPI number (p=00).

12.6.4 Calculation of the transmit clock frequency

The transmit clock frequency of the slave select input function (SSPI00) communication can be calculated using the following calculation formula.

(1) Slave device

(transmit clock frequency) = {Serial clock (SCLK) frequency provided by the master device} Note [Hz].

Note The maximum enable transmit clock frequency is $f_{MCK}/6$.

Remark m: Unit number (m=0) n: Channel number (n=0) p: SSPI number (p=00)

Table 12-3 Slave select input function run clock selection

SMRmn register	SPSm register								Run clock (f_{MCK}) Note	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		f_{CLK}=32MHz operation
0		X	X	X				0	f_{CLK}	32MHz
		X	X	X				1	$f_{CLK}/2$	16MHz
		X	X	X				0	$f_{CLK}/2^2$	8MHz
		X	X	X				1	$f_{CLK}/2^3$	4MHz
		X	X	X				0	$f_{CLK}/2^4$	2MHz
		X	X	X				1	$f_{CLK}/2^5$	1kHz
		X	X	X				0	$f_{CLK}/2^6$	500kHz
		X	X	X				1	$f_{CLK}/2^7$	250kHz
		X	X	X				0	$f_{CLK}/2^8$	125kHz
		X	X	X				1	$f_{CLK}/2^9$	62.5kHz
		X	X	X				0	$f_{CLK}/2^{10}$	31.25kHz
		X	X	X				1	$f_{CLK}/2^{11}$	15.63kHz
		X	X	X				0	$f_{CLK}/2^{12}$	7.81kHz
		X	X	X				1	$f_{CLK}/2^{13}$	3.91kHz
		X	X	X				0	$f_{CLK}/2^{14}$	1.95kHz
		X	X	X				1	$f_{CLK}/2^{15}$	977Hz

Note To change the clock selected as f_{CLK} (change the value of the system clock control register (CKC)), you must stop the operation of the universal serial communication unit (SCI) (serial channel stop register m(STm)=000FH) after making the change.

Remark 1.X: Ignore
2.m: Unit number (m=0) n: Channel number (n=0).

12.6.5 Procedure for handling errors during clock-synchronous serial communication with the slave selection input function

The processing steps when an error occurs during clock-synchronous serial communication that is subordinate to the select input function are shown in Figure 12-95.

Figure 12-95 Handling steps when an overflow error occurs

Software operation	Hardware status	Comments
Read the serial data register mn(SDRMN). →	The BFF m n bit of the SSRm n register is “0” and channel n is acceptable.	This is to prevent overflow errors from ending the next reception during mishandling.
Read the serial status register mn (SSRmn).		The type of error is judged, and the reading value is used to clear the error flag.
Clear the trigger register mn to the serial flag →	Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors during read operations can only be cleared.

Remark m: Unit number (m=0) n: Channel number (n=0).

12.7 Operation of UART (UART0~UART2) communication

This is a function that communicates asynchronously through a total of two lines: serial data transmission (TxD) and serial data reception (RxD). Using these two communication lines, data that are transmitted and received asynchronously (using internal baud rate) with other communicating parties in data frames (consisting of start bits, data, parity bits, and stop bits) are used to send and receive data. Full-duplex asynchronous UART communication can be achieved by using two channels, send private (even channels) and receive private (odd channels).

[Transmit and receive data]

- 7-bit, 8-bit or 9-bit data length Note
- MSB/LSB preferred choice
- Level settings for sending and receiving data (choose whether the level is inverted).
- Parity bit appending and parity check functions
- Stop bit appending, and stop bit detection function

[Interrupt function]

- End of transfer interrupt, buffer empty interrupt
- Error interrupts caused by frame errors, parity errors, and overflow errors

[Error detection flag]

- Frame errors, parity errors, overflow errors

Note that only UART0 supports 9-bit data length.
UART0 uses channel 0 and channel 1 of SCI0.
UART1 uses channels 2 and 3 of SCI0.
UART2 uses Channel 0 and Channel 1 of SCI1.

Each channel can choose a function to use, except for the selected function, other functions can not operate.

For example, when UART0 is used for channel 0 and channel 1 of unit 0, SSPI00 and IIC01 cannot be used. However, while using UART0, channels 2 and 3 of different channels can use SSPI10, UART1, or SUDs IIC10.

Note When used as a UART, the sender (even channels) and receivers (odd channels) can only be used for the UART.

UART has the following 4 types of communication operations:

- UART send (see 12.7.1).
- UART reception (see 12.7.2).

12.7.1 UART transmission

UART transmission is the operation of the microcontroller of this product to send data asynchronously to other devices.

An even number of the 2 channels used by the UART are used for UART sending.

UART	UART0	UART1	UART2
Object channel	Channel 0 of SCI0	Channel 2 of SCI0	Channel 0 of SCI1
The pin used	TxD0	TxD1	TxD2
Interrupt	INTST0	INTST1	INTST2
	You can select either a transmit-end interrupt (single-pass mode) or a buffer null interrupt (continuous transfer mode).		
Error detection flag	not		
Length of transmitted data	7-bit, 8-bit or 9-digit Note 1		
Transfer rate	Max.f_{MCK}/6[bps](SDRmn[15:9]≥2), Min.f_{CLK}/($2\times2^{15}\times128$)[bps]Note2		
Data phase	Normal-phase output (default: high). Inverting output (default: low).		
Parity bits	You can choose from the following: • No parity bits. • Appending zero check. • Appending parity. • Appending odd parity.		
Stop bit	You can choose from the following: • Appending 1 bit. • Appending 2 bits.		
Data direction	MSB first or LSB first		

Note 1 Only UART0 supports 9-bit data length.

2. It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Note 1.f_{MCK}: Operating clock frequency of the object channel

f_{CLK}: System clock frequency

2.m: Unit number (m=0, 1) n: channel number (n=0, 2) mn=00, 02, 10.

(1) Register setting

Figure 12-96 Example of register settings when UART is sent by UART (UART0~UART 2) (1/2)

(a) serial mode register mn (SMRmn)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKSmn	CCSmn												MDmn2	MDmn1	MDmn0
	0/1	0	0	0	0	0	0	0	0	0	1	0	0	0	1	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	RXEmn	DAPmn	CKPmn		EOCmn	PTCmn1	PTCmn0	DIRmn		SLCmn1	SLCmn0			DLSmn1	DLSmn0
	1	0	0	0	0	0	0/1	0/1	0/1	0	0/1	0/1	0	1	0/1 注 1	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: add 1 bit
10B: add 2 bits

(c) serial data regsiter mn (SDRmn) (low 8 bit:TXDq)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration							0 Note2	transmit data configuration							
									TXDq							

(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														SOLm2		SOLm0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0/1	0	0/1

0: positive phase (normal) transmit
1: inverted phase transmit

Note 1 Limited to SCR00 registers, other fixed as “1”.

2. When communicating with a length of 9 bits of data, bit0 to 8 of the SDRm0 register is the setting area for sending data. Only UART0 can communicate with 9-bit data lengths.

Note 1.m: Unit number (m=0, 1) n: Channel number (n=0, 2)q: UART numbers (q=0~2)mn=00, 02, 10

2.□ : Fixed in UART send mode. ▨ : Cannot be set (initial value).

×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).

0/1: Set “0” or “1” according to the user's purpose.

Figure 12-96 Example of register settings when UART is sent by UART (UART0~UART 2) (2/2)

(e) serial output register m (SOm)…. Only configure bit of target channel

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm	0	0	0	0	CKOm3 ×	CKOm2 ×	CKOm1 ×	CKOm0 ×	0	0	0	0	SOm3 ×	SOm2 0/1 Note	SOm1 ×	SOm0 0/1 Note

0: serial data output value as "0"
1: serial data output value as "1"

(f) serial output enable register m (SOEm)…. Only set bit of target channel to "1".

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	SOEm3 ×	SOEm2 0/1	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	SSm3 ×	SSm2 0/1	SSm1 ×	SSm0 0/1

Note that before starting to send, when the SOLmn bit of the corresponding channel is “0”, it must be set to “1”; When the SOLmn bit of the corresponding channel is “1”, “0” must be set. During communication, the value varies from data to communication.

Note 1.m: Unit number (m=0, 1) n: Channel number (n=0, 2)q: UART numbers (q=0~2)mn=00, 02, 10

2.□ : Fixed in UART send mode. ▨ : Cannot be set (initial value).

×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).

0/1: Set “0” or “1” according to the user's purpose.

(2) Operation Steps

Figure 12-97 Initial setup steps for UART sending

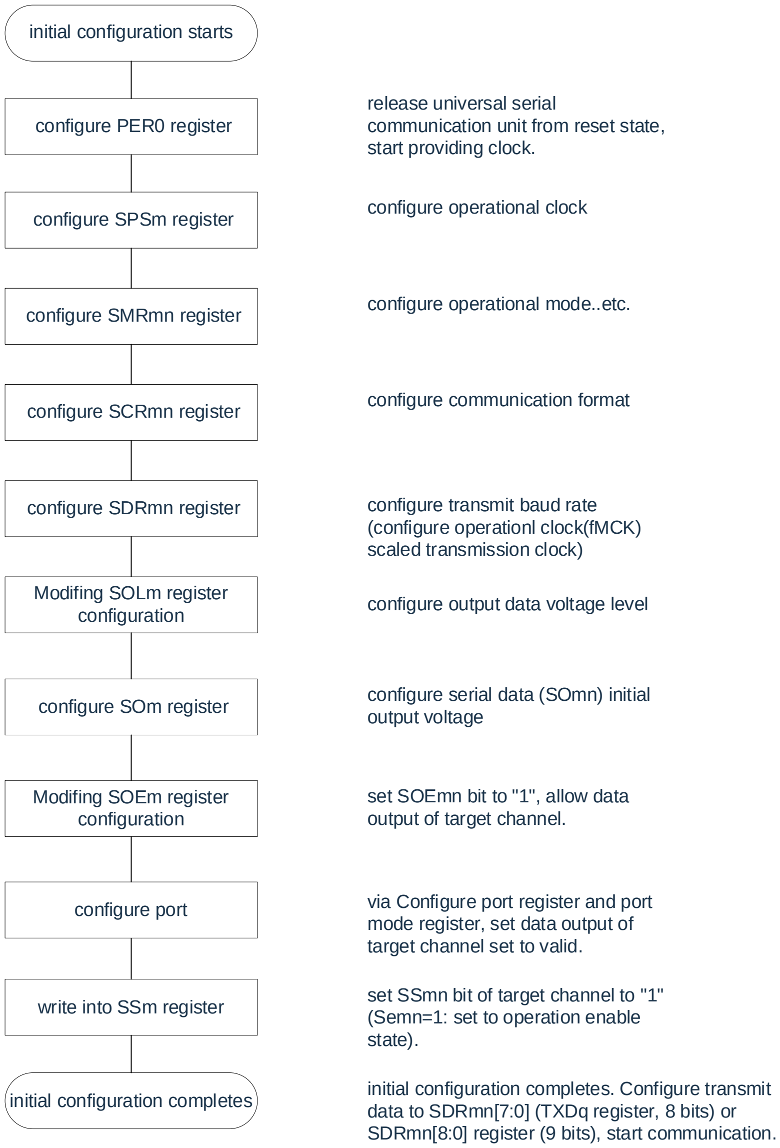

initial configuration starts
configure PER0 register
release universal serial communication unit from reset state, start providing clock.
configure SPSm register
configure operational clock
configure SMRmn register
configure operational mode..etc.
configure SCRmn register
configure communication format
configure SDRmn register
configure transmit baud rate (configure operationl clock(fMCK) scaled transmission clock)
Modifing SOLm register configuration
configure output data voltage level
configure SOm register
configure serial data (SOmn) initial output voltage
Modifing SOEm register configuration
set SOEmn bit to "1", allow data output of target channel.
configure port
via Configure port register and port mode register, set data output of target channel set to valid.
write into SSm register
set SSmn bit of target channel to "1" (Semn=1: set to operation enable state).
initial configuration completes
initial configuration completes. Configure transmit data to SDRmn[7:0] (TXDq register, 8 bits) or SDRmn[8:0] register (9 bits), start communication.

Figure 12-98 Stop step for UART sending

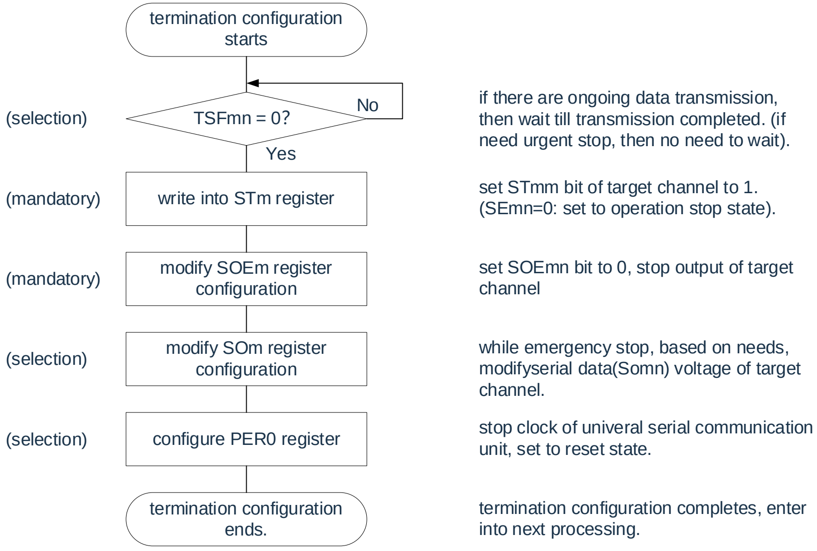
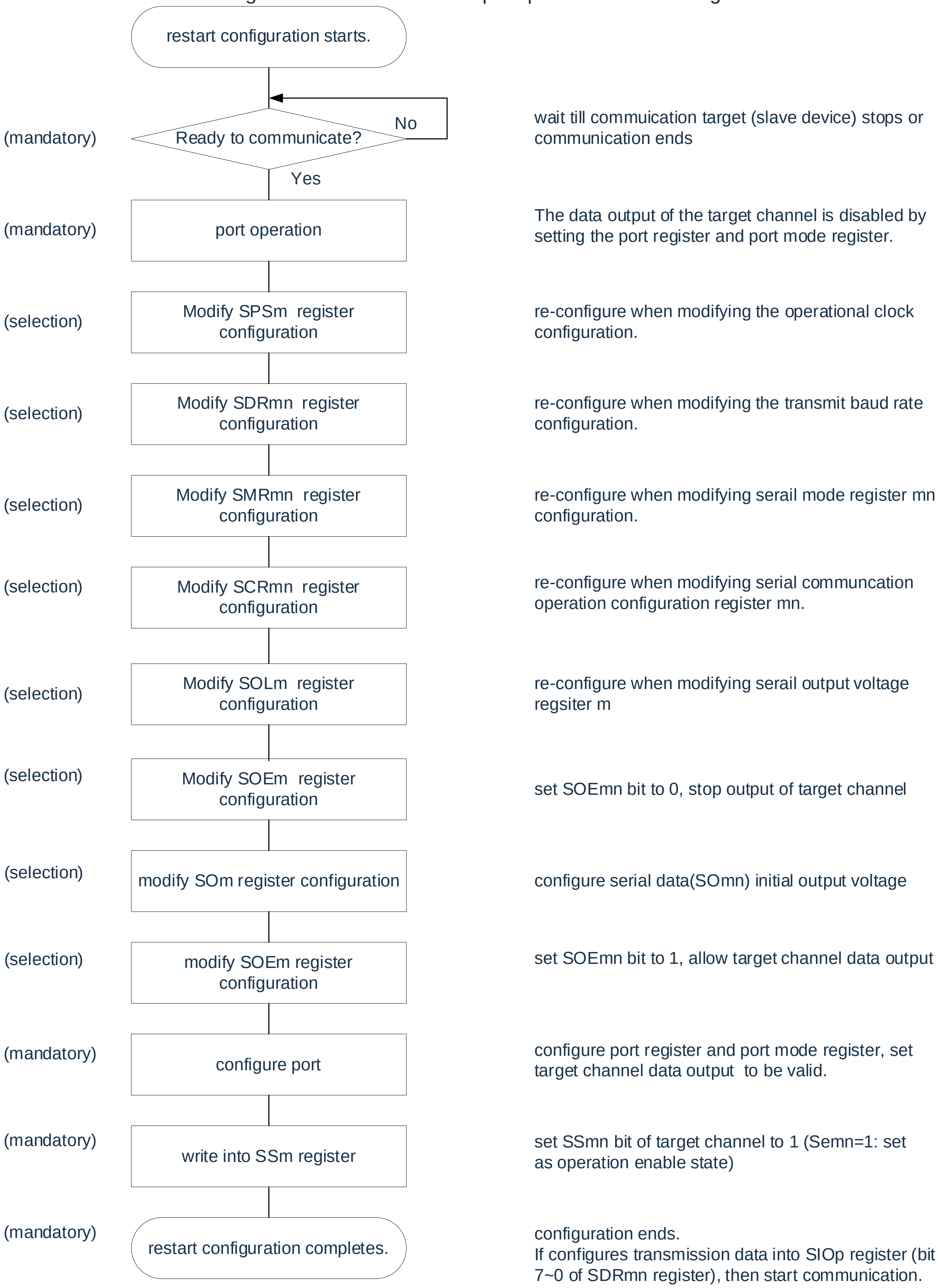

termination configuration starts
(selection)
TSFmn = 0?
No
Yes
if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).
(mandatory)
write into STm register
set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).
(mandatory)
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, modifyserial data(Somn) voltage of target channel.
(selection)
configure PER0 register
stop clock of univeral serial communication unit, set to reset state.
termination configuration ends.
termination configuration completes, enter into next processing.

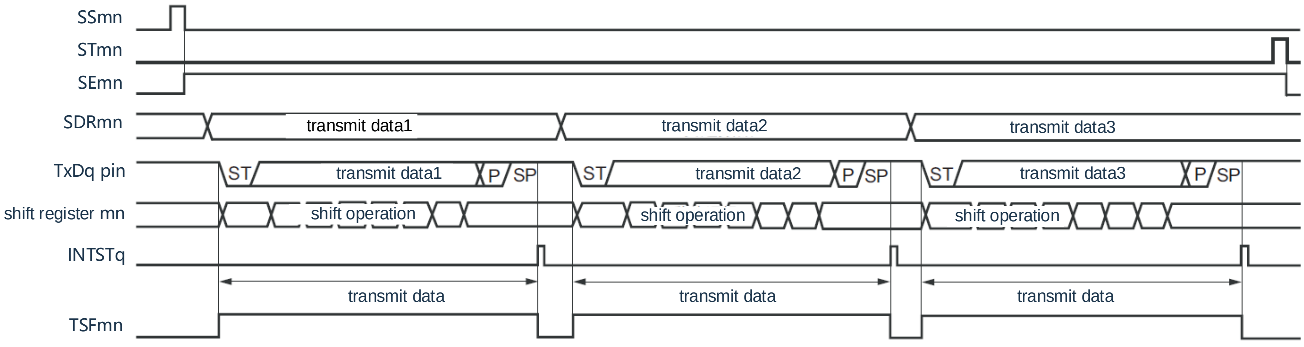
Figure 12-99 Restart setup steps for UART sending


restart configuration starts.
(mandatory)
Ready to communicate?
No
Yes
wait till commuication target (slave device) stops or communication ends
(mandatory)
port operation
The data output of the target channel is disabled by setting the port register and port mode register.
(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)
Modify SOLm register configuration
re-configure when modifying serail output voltage regsiter m
(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, allow target channel data output
(mandatory)
configure port
configure port register and port mode register, set target channel data output to be valid.
(mandatory)
write into SSm register
set SSmn bit of target channel to 1 (Semn=1: set as operation enable state)
(mandatory)
restart configuration completes.
configuration ends.
If configures transmission data into SIOp register (bit 7~0 of SDRmn register), then start communication.

Note If you override PER0 in the abort setting to stop the clock, you must wait until the communication object stops or the communication ends, instead of starting the setting again.

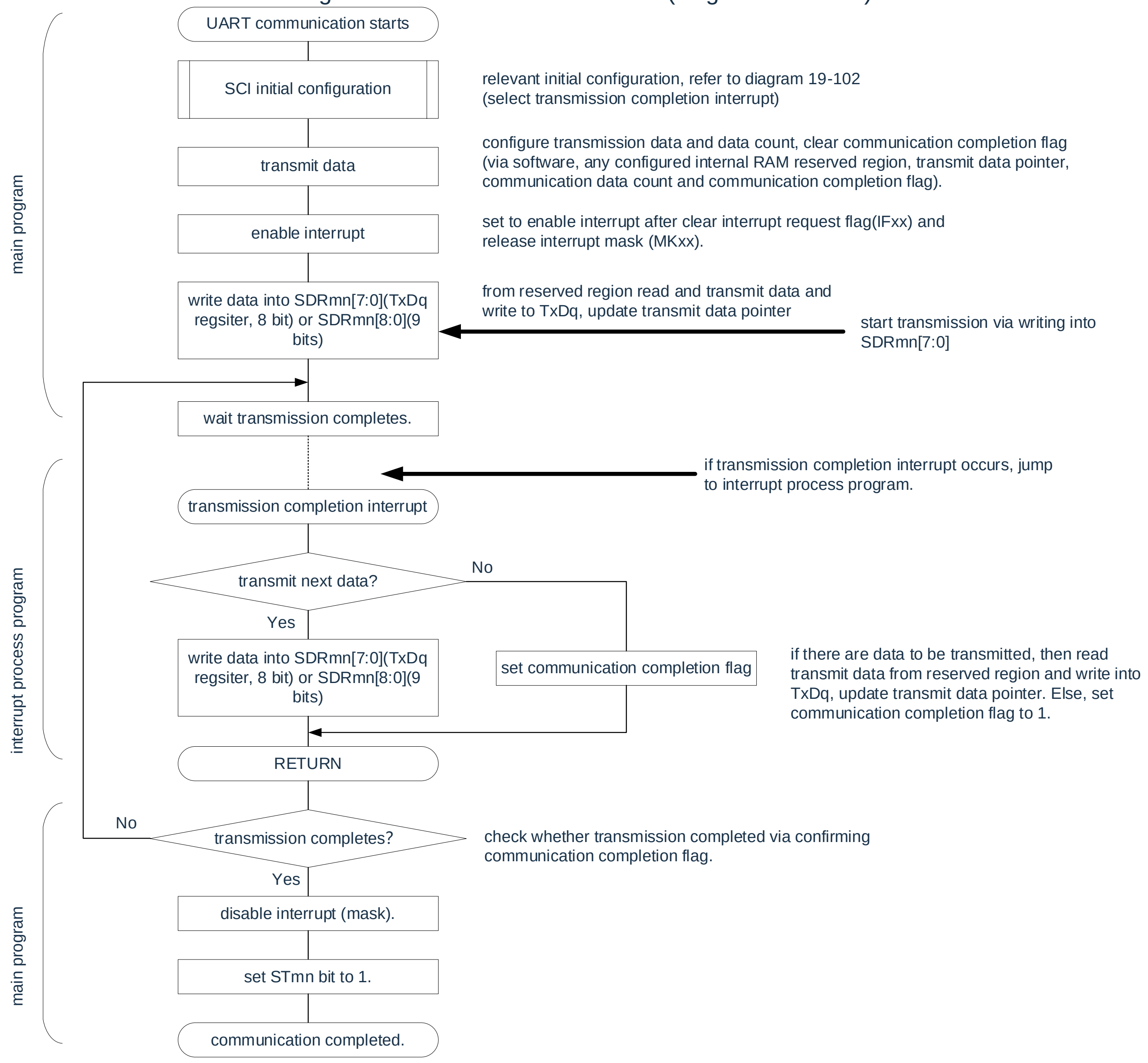
(3) Process flow (single-send mode).

Figure 12-100 UART transmission (single-send mode)



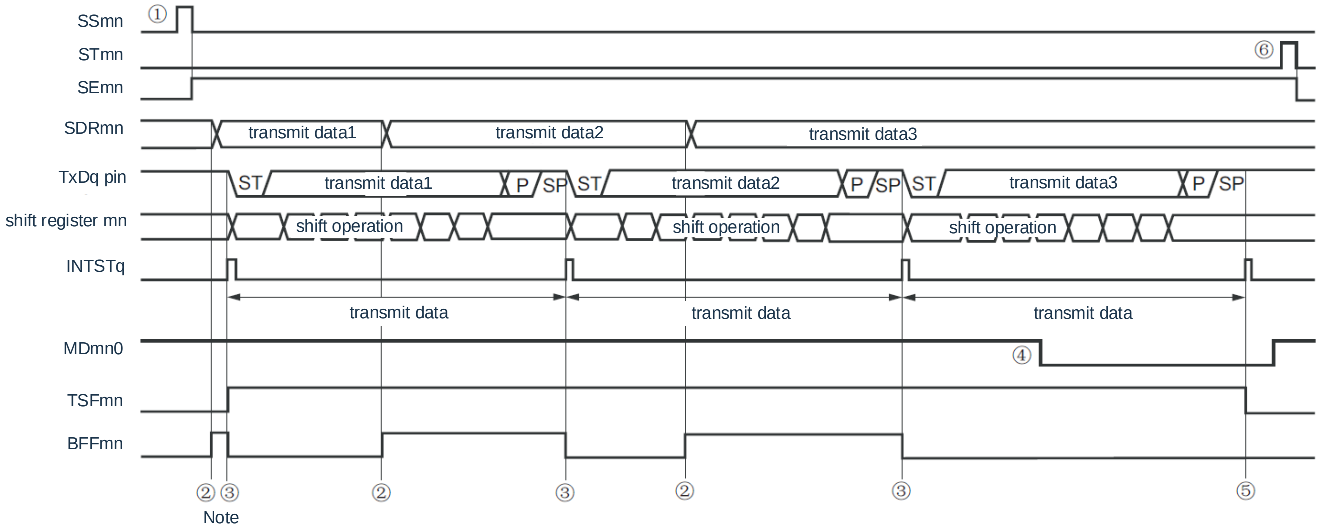
Remarks m: Unit number (m=0, 1) n: Channel number (n=0, 2) q: UART numbers (q=0~2) mn=00, 02, 10

Figure 12-101 UART transmission (single-send mode)


main program
UART communication starts
SCI initial configuration
relevant initial configuration, refer to diagram 19-102
(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[7:0](TxDq regsiter, 8 bit) or SDRmn[8:0](9 bits)
from reserved region read and transmit data and
write to TxDq, update transmit data pointer
start transmission via writing into
SDRmn[7:0]
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[7:0](TxDq regsiter, 8 bit) or SDRmn[8:0](9 bits)
set communication completion flag
if there are data to be transmitted, then read
transmit data from reserved region and write into
TxDq, 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 send mode)

Figure 12 12-102 UART transmission (continuous send mode)

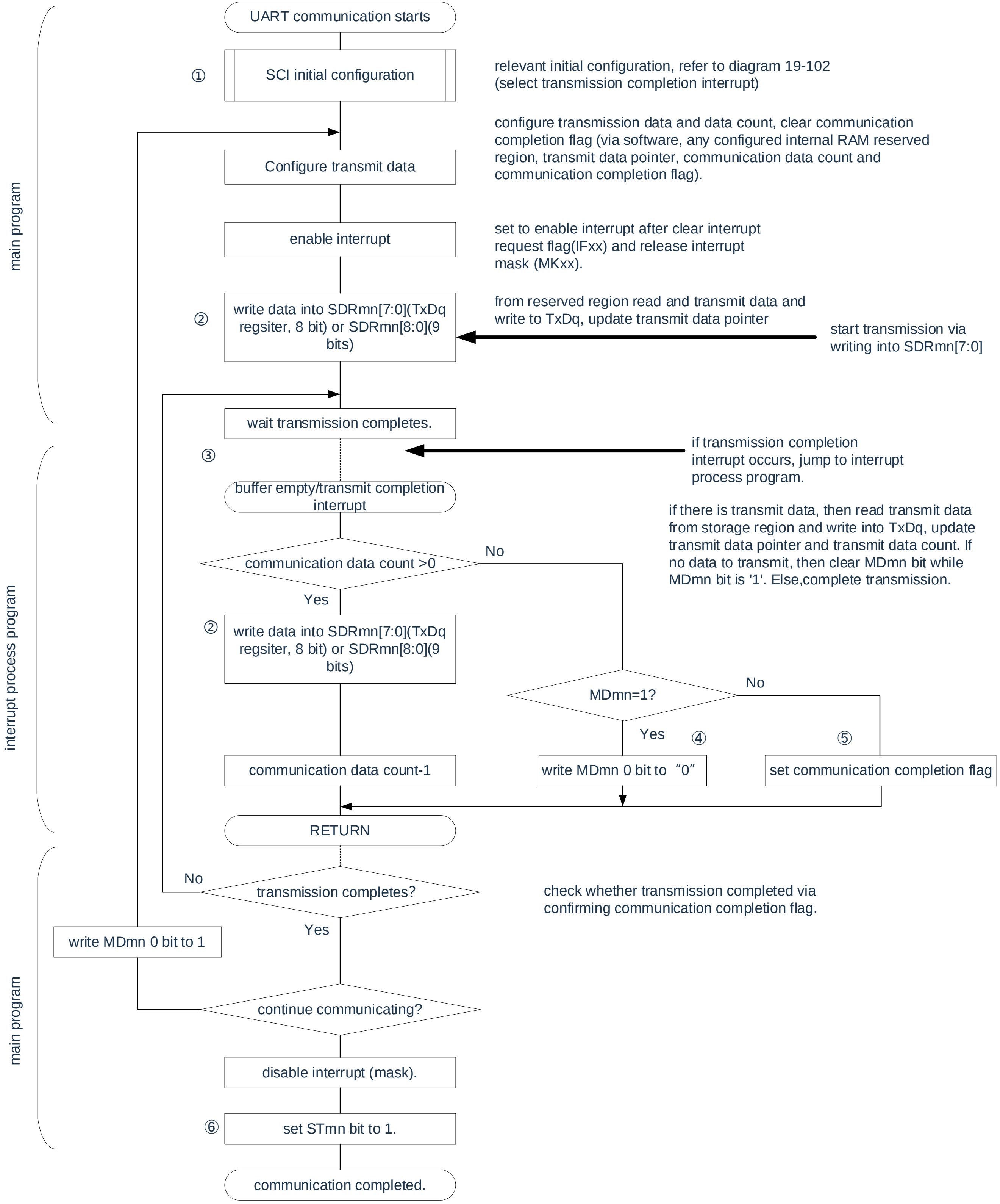


Note If the BFFmn bit of the serial status register mn (SSRmn) is “1” (when valid data is saved in the serial data register mn (SDRmn)) is given The SDRmn register writes the transmitted data and overrides the transmitted data.

Notice The MDmn0 bit of the serial mode register mn (SMRmn) can be overridden even during operation. However, in order to catch up with the end of the transmission interruption of the last transmitted data, it must be overwritten before the last bit of transmission begins.

Remark m: Unit number (m=0, 1) n: Channel number (n=0, 2) q: UART numbers (q=0~2) mn=00, 02, 10

Figure 12-103 UART transmission (continuous send mode)


UART communication starts
①
SCI initial configuration
relevant initial configuration, refer to diagram 19-102
(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 and communication completion flag).
Configure transmit data
enable interrupt
set to enable interrupt after clear interrupt request flag(IFxx) and release interrupt mask (MKxx).
②
write data into SDRmn[7:0](TxDq regsiter, 8 bit) or SDRmn[8:0](9 bits)
from reserved region read and transmit data and write to TxDq, update transmit data pointer
start transmission via writing into SDRmn[7:0]
main program
wait transmission completes.
③
if transmission completion interrupt occurs, jump to interrupt process program.
buffer empty/transmit completion interrupt
if there is transmit data, then read transmit data from storage region and write into TxDq, 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.
communication data count >0
No
Yes
②
write data into SDRmn[7:0](TxDq regsiter, 8 bit) or SDRmn[8:0](9 bits)
MDmn=1?
No
Yes
④
⑤
communication data count-1
write MDmn 0 bit to “0”
set communication completion flag
interrupt process program
RETURN
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.
main program
main program

Note (1) to (6) in the figure corresponds to (1) to (6) in the “Figure 12 12-102 UART Transmission (Continuous Send Mode)”.

12.7.2 UART reception

UART reception is the operation of other devices of this product's microcontroller to receive data asynchronously.

An odd number of the 2 channels used by the UART are used for UART reception. However, the SMR registers for both odd and even channels need to be set.

UART	UART0	UART1	UART2
Object channel	Channel 1 of SCI0	Channel 3 of SCI0	Channel 1 of SCI1
The pin used	RxD0	RxD1	RxD2
Interrupt	INTSR0	INTSR1	INTSR2
	Limited to end-of-transfer interrupts (disable setting buffer null interrupts).		
Error interrupt	INTSRE0	INTSRE1	INTSRE2
Error detection flag	• Frame Error Detection Flag (FEFmn). • Parity Error Detection Flag (PEFmn). • Overflow Error Detection Flag (OVFmn).		
The length of the transmitted data	7-bit, 8-bit or 9-digit Note 1		
Transfer rate	Max.f_{MCK}/6[bps](SDRmn[15:9]≥2), Min.f_{CLK}/(2×2^{15}×128)[bps]		
Data phase	Normal-phase output (default: high). Inverting output (default: low).		
Parity bits	You can choose from the following: • No parity bits (no parity). • Appending zero check (no parity). • Even-check • Odd check		
Stop bit	Appending 1 bit.		
Data direction	MSB first or LSB first		

Note 1 Only UART0 supports 9-bit data length.

2. Must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

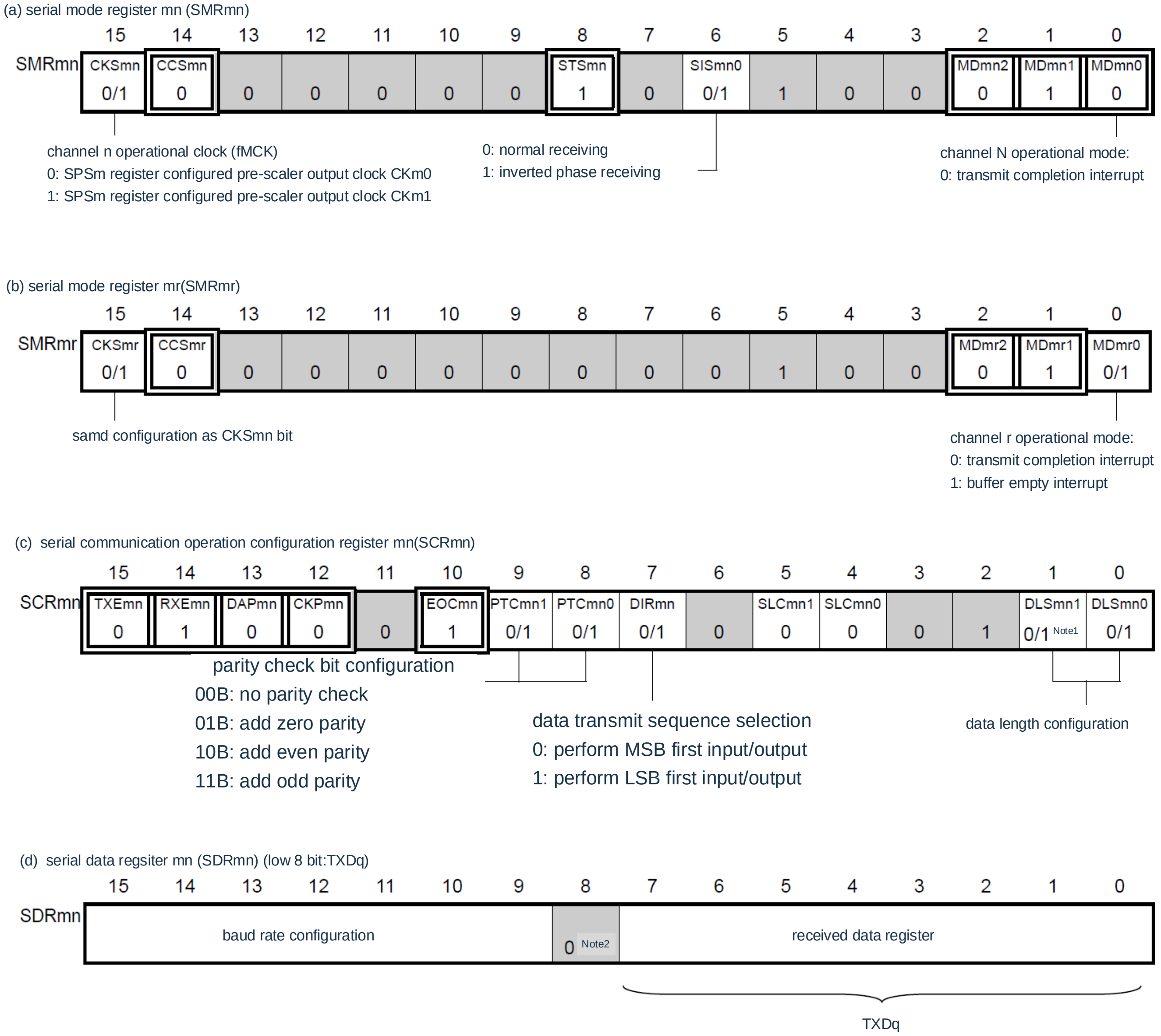
Note 1. f_{MCK}: The operating clock frequency of the object channel

f_{CLK}: System clock frequency

2.m: Unit number (m=0, 1) n: channel number (n=1, 3) mn=01, 03, 11.

(1) Register setting

Figure 12-104 Example of register settings 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

samd configuration as CKSmn bit

channel r operational mode:
0: transmit completion interrupt
1: buffer empty interrupt

(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	0	1	DLSmn1 0/1 Note1	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

data length configuration

(d) serial data regsiter mn (SDRmn) (low 8 bit:TXDq)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration							0 Note2	received data register							

TXDq

Note 1 Limited to SCR01 registers, other fixed as “1”.

2. When communicating with a length of 9 bits of data, bit0 to 8 of the SDRm1 register is the setting area for sending data. Only UART0 can communicate with 9-bit data lengths.

Notice When the UART is received, the SMRmr register of channel r paired with channel n must also be set.

Note 1. m: unit number (m=0, 1) n: channel number (n=1, 3) mn=01, 03, 11.
r: Channel number (r=n–1) q: UART number (q=0~2)

2. □ : Fixed in UART receive mode. ▨ : Cannot be set (initial value).
×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).
0/1: Set “0” or “1” according to the user's purpose.

Figure 12-104 Example of register settings when UART is received by UART (UART0~UART 2) (2/2)

(e) serial output register m (SOm)…. Not used in this mode.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm	0	0	0	0	CKOm3 ×	CKOm2 ×	CKOm1 ×	CKOm0 ×	0	0	0	0	SOm3 ×	SOm2 ×	SOm1 ×	SOm0 ×

(f) serial output enable register m (SOEm)…. Not used in this mode.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	SOEm3 ×	SOEm2 ×	SOEm1 ×	SOEm0 ×

(g) serial channel start register m (SSm) …. Only set bit of target channel to "1".

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	SSm3 0/1	SSm2 ×	SSm1 0/1	SSm0 ×

Note 1.m: Unit number (m=0, 1).

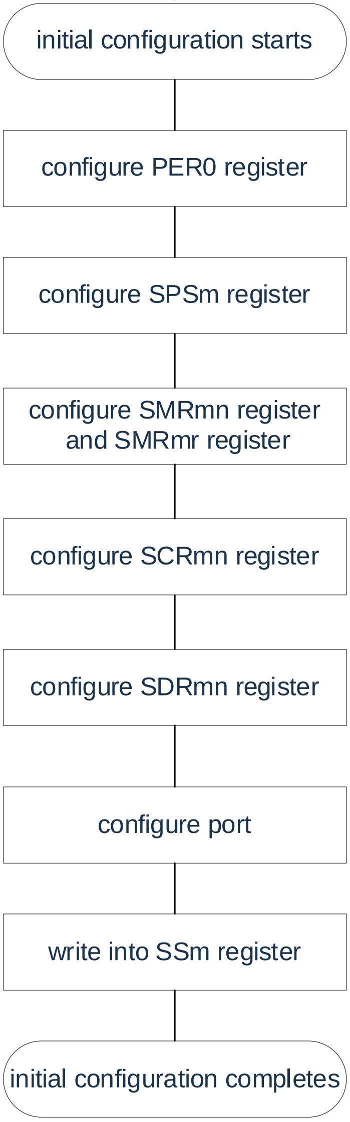
2. □ : Fixed in UART receive mode. ▨ : Cannot be set (initial value).

×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).

0/1: Set “0” or “1” according to the user's purpose.

(2) Operation Steps

Figure 12-105 Initial setup steps for UART reception



release universal serial communication unit from reset state, start providing clock.

configure operational clock

configure operational mode..etc.

configure communication format

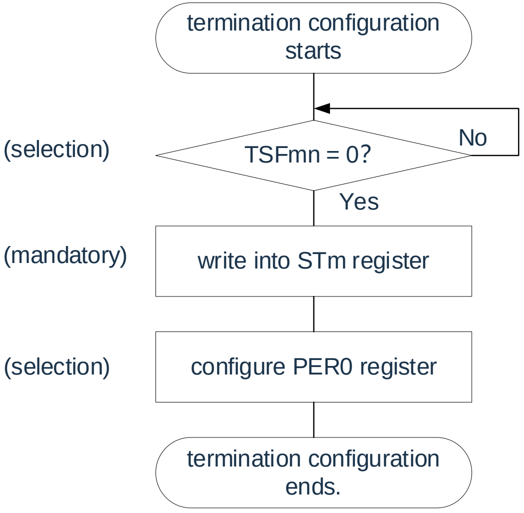
configure transmit baud rate (configure operationl clock(fMCK) scaled transmission clock)

via Configure port register and port mode register, set data output of target channel to valid.

set SSmn bit of target channel to "1", make Semn to "1" (operation enable state), and wait for start bit detection.

Note At least 4 FMCK clocks must be spaced after setting the RXEmn bit of the SCRmn register to "1" and then set the SSmn bit to “1”.

Figure 12-106 Stop steps for UART reception



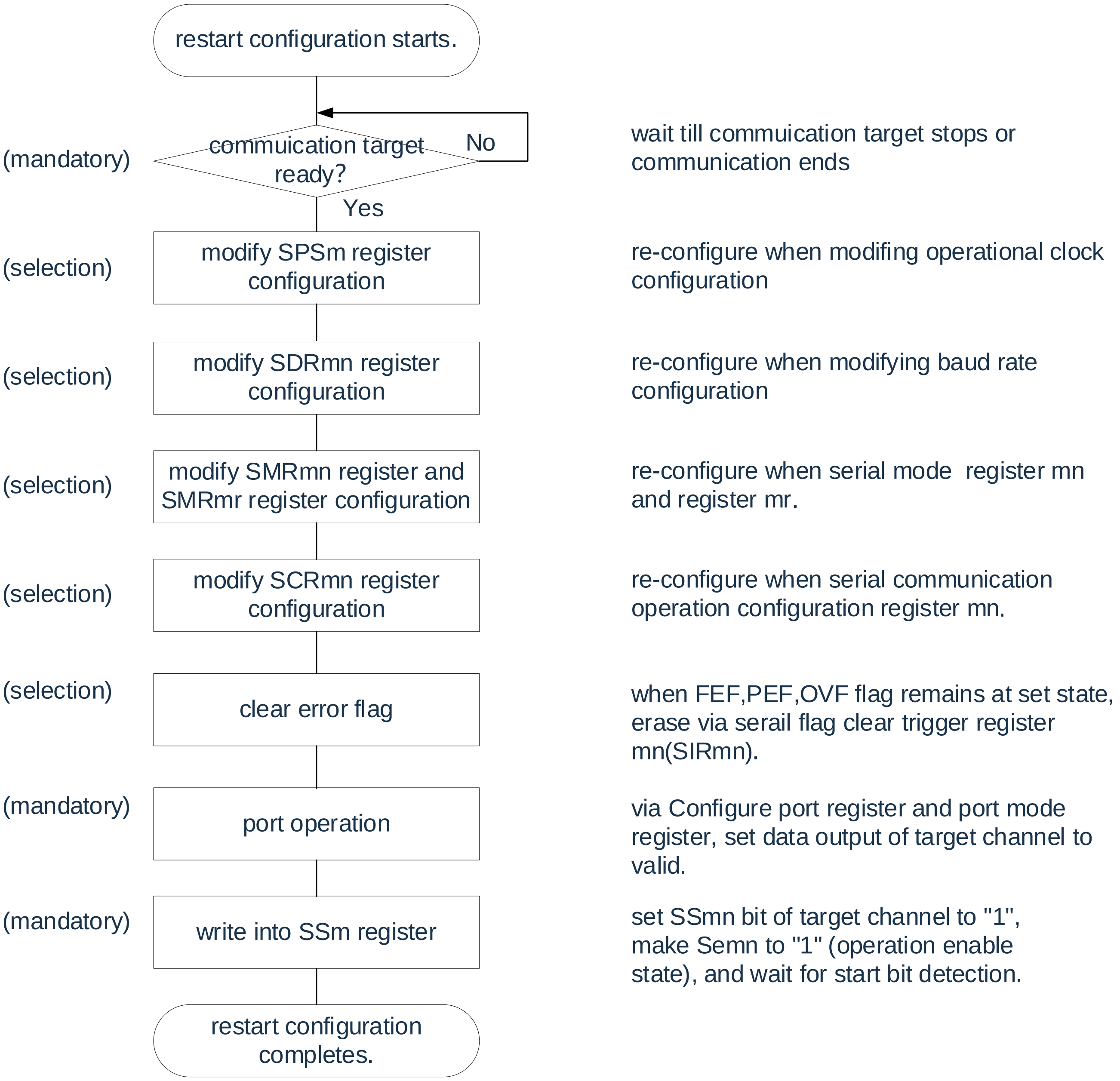
if there are ongoing data transmission, then wait till transmission completed. (if need urgent stop, then no need to wait).

set STmm bit of target channel to 1. (SEmn=0: set to operation stop state).

stop clock of universal serial communication unit, set to reset state

finish termination configuration, enter into next processing.

Figure 12-107 Restart setup step for UART reception

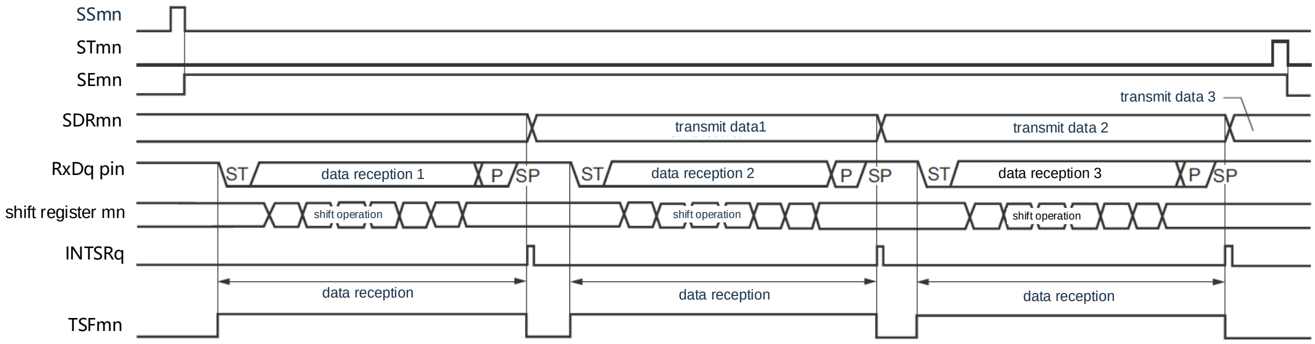

restart configuration starts.
(mandatory)
commuication target ready?
No
Yes
wait till commuication target stops or communication ends
(selection)
modify SPSm register configuration
re-configure when modifing operational clock configuration
(selection)
modify SDRmn register configuration
re-configure 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.

Note At least 4 FMCK clocks must be spaced after setting the RXEmn bit of the SCRmn register to "1" and then set the SSmn bit to “1”.

Note If you override PER0 in the abort setting to stop the clock, you must wait until the communication object stops or the communication ends, instead of starting the setting again.

(3) Process flow

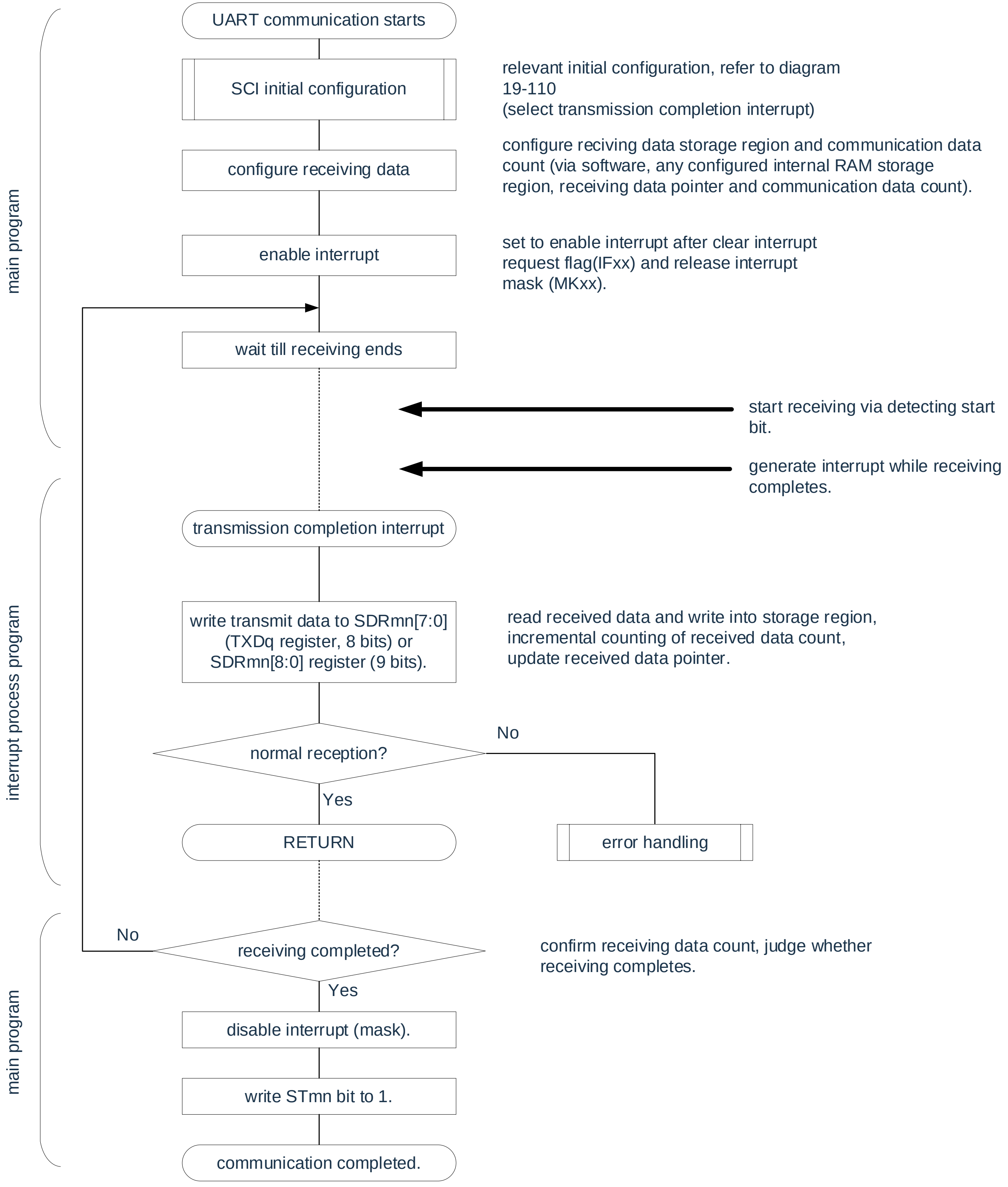
Figure 12-108 diagram of UART reception



Remark m: Unit number (m=0, 1) n: Channel number (n=1, 3) mn=01, 03, 11.

r: Channel number (r=n–1) q: UART number (q=0~2)

Figure 12-109 UART reception


main program
UART communication starts
SCI initial configuration
configure receiving data
enable interrupt
wait till receiving ends
interrupt process program
transmission completion interrupt
write transmit data to SDRmn[7:0] (TXDq register, 8 bits) or SDRmn[8:0] register (9 bits).
normal reception?
No
Yes
RETURN
error handling
main program
No
receiving completed?
Yes
disable interrupt (mask).
write STmn bit to 1.
communication completed.

relevant initial configuration, refer to diagram 19-110 (select transmission completion interrupt)

configure reciving data storage region and communication data count (via software, any configured internal RAM storage region, receiving data pointer and communication data count).

set to enable interrupt after clear interrupt request flag(IFxx) and release interrupt mask (MKxx).

start receiving via detecting start bit.

generate interrupt while receiving completes.

read received data and write into storage region, incremental counting of received data count, update received data pointer.

confirm receiving data count, judge whether receiving completes.

12.7.3 Calculation of the baud rate

(1) Formula for calculating the baud rate

The baud rate of UART (UART0~UART2) communication can be calculated using the following formula:

(baud rate) = {Clock of the object channel (f_{MCK}) frequency}× (SDRmn[15:9]+ 1) ÷2[bps]

Note The SDRmn [15:9] of the serial data register mn (SDRmn) is disabled from being set to “0000000B” and “0000001B”.

Notice 1 Because the value of SDRmn [15:9] when using UART is the value of bit15~9 of the SDRmn register (0000010B ~1111111B), so 2~127.
2.m: Unit number (m=0, 1) n: channel number (n=0~2). mn=00~03, 10~11.

The operating clock (f_{MCK}) depends on the serial clock select register m (SPSm) and bit 15 (CKSmn bit) of the serial mode register mn (SMRmn).

Table 12-4 Selection of the UART operating clock

SMRmn register	SPSm register								Run clock (f_{MCK}) Note	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		f_{CLK}=32MHz 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 stop the operation of the universal serial communication unit (SCI) (serial channel stop register m(STm)=000FH) after making the change.

Notice 1.X: Ignore
2.m: Unit number (m=0, 1) n: channel number (n=0~2)mn=00~03 , 10~11.

(2) Baud rate error at the time of sending

The baud rate error of UART (UART0~UART2) communication can be calculated using the following calculation formula, and the baud rate of the sender must be set within the Enable range of the receiver's baud rate.

(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}=32MHz is shown below.

UART baud rate (Target baud rate)	f_{CLK}=32MHz			
	Running clock (f_{MCK})	SDRmn[15:9]	Calculated value of the 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) n: Channel number (n=0, 2) mn=00, 02, 10.

(3) Enable range of the baud rate at reception

The baud rate tolerance range of UART (UART0~UART2) communication reception can be calculated using the following calculation formula, and the baud rate of the sender must be set within the acceptor's baud rate tolerance.

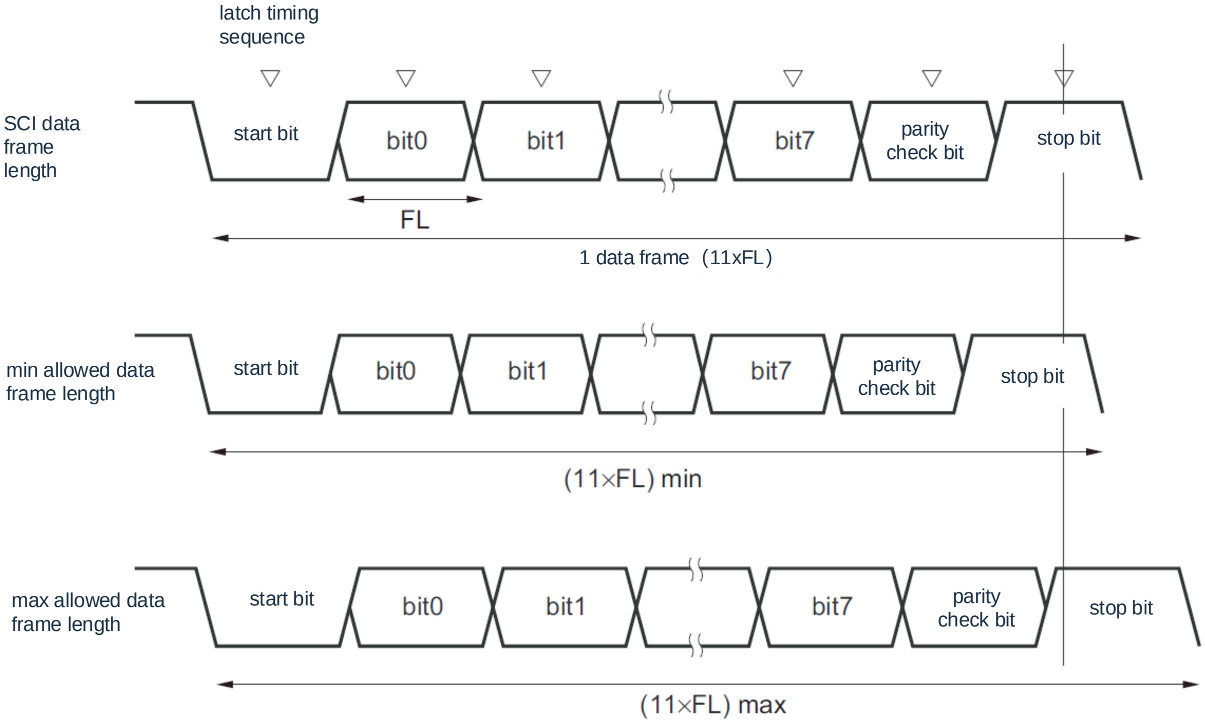
$$\text{(Maximum baud rate that can} \quad \frac{2\times k\times Nfr}{2\times k\times Nfr-k+2} \quad \times \text{ Brother}$$

$$\text{(Minimum baud rate that can} \quad \frac{2\times k\times (Nfr-1)}{2\times k\times Nfr-k-2} \quad \times \text{ Brother}$$

Brate: Calculated value of the baud rate of the receiver (see “12.7.4 baud rate calculation”).
k: SDRmn[15:9]+1
Nfr: 1 frame length of data [bit].
= (start bit) + (data length) + (parity bit) + (stop bit).

Remark m: unit number (m=0, 1) n: channel number (n=1, 3) mn=01, 03, 11

Figure 12-110 Enable range of baud rate at reception (frame length of 1 data = 11 bits)



As shown in Figure 12-110 after the start bit is detected, the latch timing of the received data depends on the divider ratio set by bit15 to 9 of the serial data register mn (SDRmn). If the last data (stop bit) can catch up with this latch timing, it can be received normally.

12.7.4 Handling steps when an error occurs during UART (UART0~UART 2) communication

The handling steps when an error occurs during UART (UART0~UART 2) communication are shown in Figure 12-111and Figure 12-112.

Figure 12-111 Processing steps when a parity error or overflow error occurs

Software operation	Hardware status	Remark
Read the serial data register mn (SDRmn).	The BFFmn bit of the SSRmn register is “0” and channel n is receiverable.	This is to prevent overflow errors from occurring when the next receive ends during error handling.
Read the serial status register mn (SSRmn).		Determine the type of error, and read the value to clear the error marker.
Clear the trigger register mn to the serial flag (SDIRmn) writes “1”.	Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors during read operations can only be cleared.

Figure 12-112 Processing steps when a frame error occurs

Software operation	Hardware status	Remark
Read the serial data register mn(SDRMN).	The BFF m n bit of the SSRm n register is “0” and channel n is acceptable.	This is to prevent overflow errors from ending the next reception during mishandling.
Read the serial status register mn(SSRmn).		Determine the error category, and read the value to remove the error marker.
Write the serial flag to clear the trigger register mn (SIRmn).	Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors during read operations can only be cleared.
Set the STmn bit of the serial channel stop register m (STm) to "1".	The serial channel allows the Without n bit of status register m (Herself m) to be “0” and channel n is the running stop state.	
Synchronize processing with the communicating party.		Because the start bit is offset, a frame error can be considered to have occurred. Therefore, it is necessary to re-synchronize with the communicating party and restart the communication.
Set the SSmn bit of the serial channel start register m (SSm) to "1".	The serial channel allows the SE m n bit of status register m (Herself m) to be “1” and channel n to be operational.	

Remarks m: Unit number (m=0, 1) n: Channel number (n=0~3) mn=00~ 03, 10~11.

12.8 Operation of LIN communication

12.8.1 LIN transmission

In UART sending, UART0 supports LIN communication.

LIN sends channel 0 using unit 0.

UART	UART0	UART1	UART2
LIN communication support	Yes	No	No
Object channel	Channel 0 for SCI0	—	—
The pin used	TxD0	—	—
Interrupt	INTST0	—	—
	Selectable transmission end interrupt (single transmission mode) or buffer air interrupt (continuous transmission mode).		
Error detection flag	Not		
The length of the transmitted data	8 bits		
Transfer Rate [Note]	Max.f_{MCK}/6[bps](SDR00[15:9]≥2), Min.$f_{CLK}/(2\times2^{15}\times128)$[bps]		
Data phase	Normal-phase output (default: high). Inverting output (default: low).		
Parity bits	No parity bits.		
Stop bit	Appending 1 bit.		
Data direction	LSB takes precedence		

Note It must be used within the range of peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet), and 2.4/9.6/19.2kbps are often used in LIN communication.

Notice f_{MCK}: Operating clock frequency for object channels
f_{CLK}: System clock frequency

LIN is short for Local Interconnect Network and is a low-speed (1 to 20kbps) serial communication protocol to reduce the cost of automotive networks. LIN communication is a single master communication, a master device can connect up to 15 slave devices.

LIN slave devices are used for the control of switches, transmissions, sensors, etc., which are connected to the master control device via LIN.

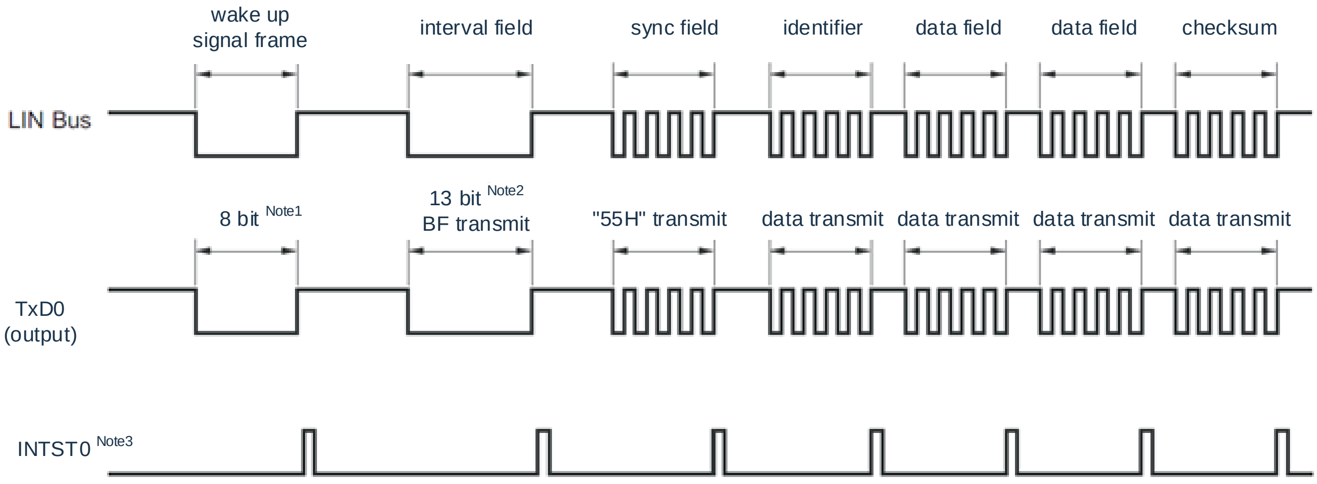
LIN masters are generally connected to networks such as the Controller Area Network.

The LIN bus is a single-wire bus that connects nodes through an ISDO9141-compliant transceiver.

According to the LIN protocol, the master device sends a frame with additional baud rate information, and the slave device receives this frame and corrects the baud rate error with the master control device. Therefore, if the baud rate error of the slave device is not greater than 15% ±, communication can be made.

A summary of the LIN's send operation is shown in Figure 14-113.

Figure 12-113 LIN Transmission Operation



Note 1 In order to meet the requirements of the wake-up signal, the baud rate is set and the corresponding data is sent by “80H”.

2. The interval segment is specified as a 13-bit wide low-level output, so assuming that the baud rate used for the main transmission is N[bps], the baud rate used in the break field is as follows:

(baud rate for break field) =9/13×N

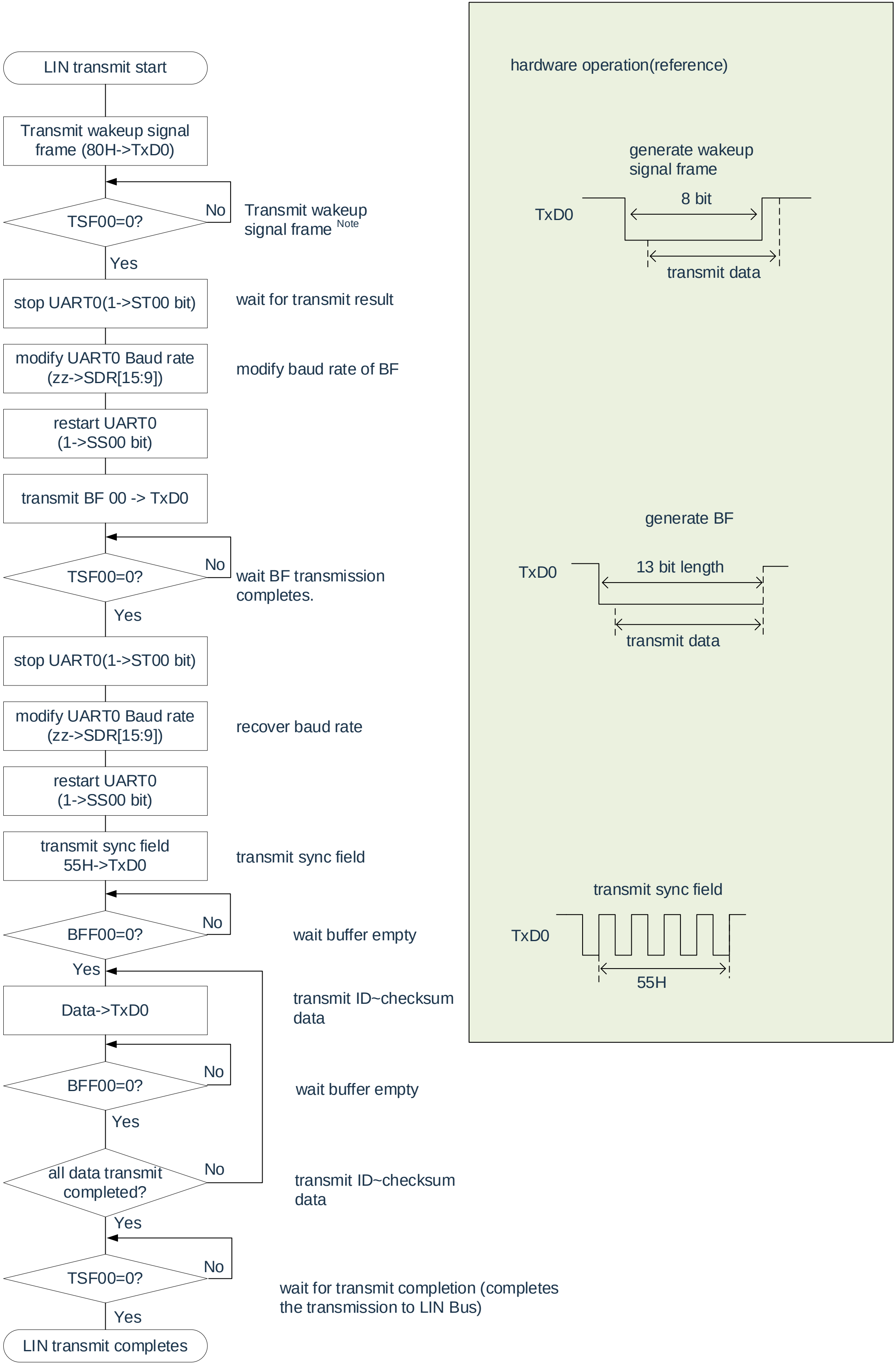
Send data of “00H” through this baud rate, generating interval segments.

3. Outputs INTST0 at the end of each data transmission, and also outputs

INTST0 when BF is transmitted.

Remarks The software controls the spacing between segments.

Figure 12-114 LIN sending


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 is limited to cases where the LIN-bus sleep state is initiated.
Remark This is the process that begins by ending the initial setting of the UART and allowing slave sending.

12.8.2 LIN reception

In UART reception, UART0 supports LIN communication.

The LIN receives the channel 1 of the Unit0.

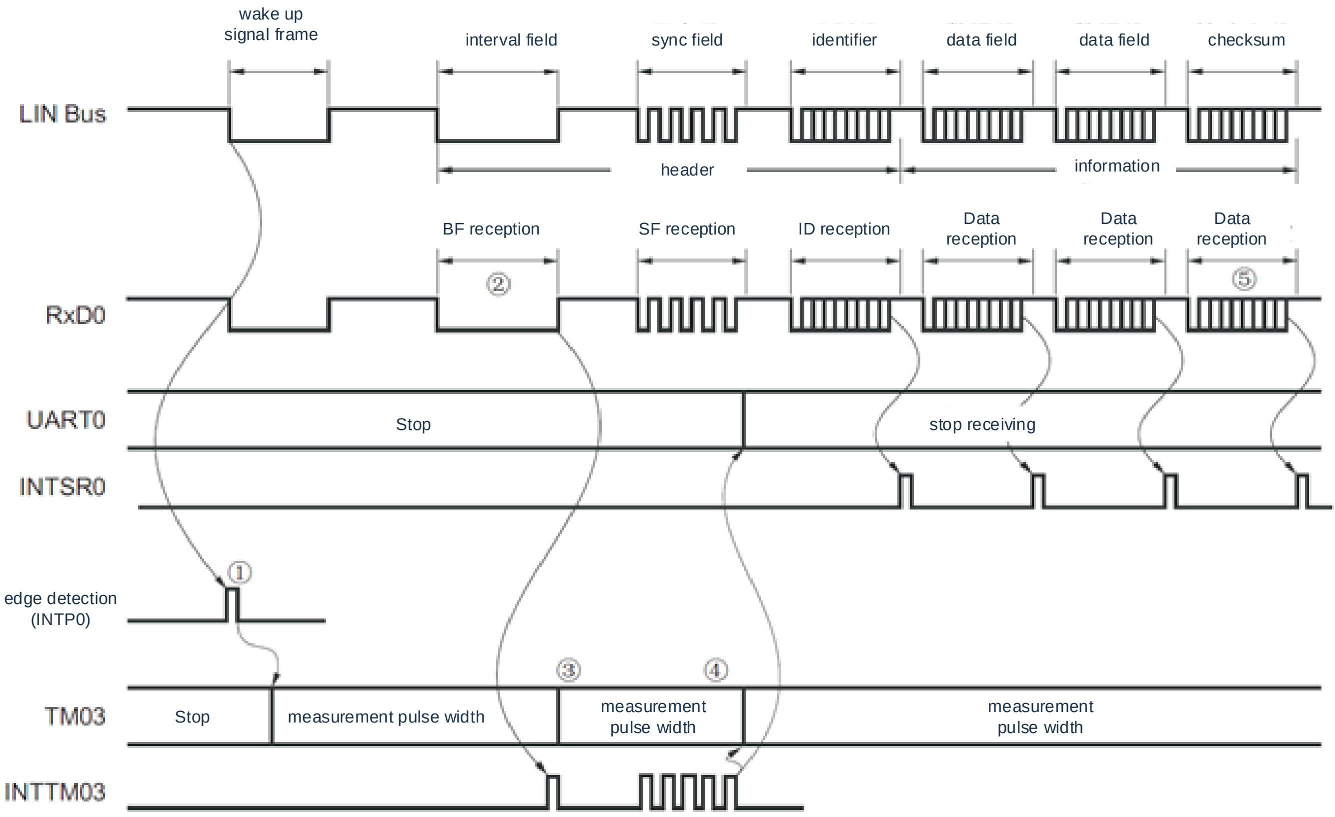
UART	UART0	UART1	UART2	UART3
LIN communication support	Yes	No	No	No
Object channel	Channel 1 of SCI0	—	—	—
The pin used	RxD0	—	—	—
interrupt	INTSR0	—	—	—
	Limited to end-of-transfer interrupts (disable setting buffer null interrupts).			
Error interrupt	INTSRE0	—	—	—
Error detection flag	• Frame error detection flag (FEF01). • Overflow error detection flag (OVF01).			
The length of the transmitted data	8 bits			
Transfer Rate Note	Max.f_{MCK}/6[bps](SDR01[15:9]≥2), Min.$f_{CLK}/(2\times2^{15}\times128)$[bps]			
Data phase	Normal-phase output (default: high). Inverting output (default: low).			
Parity bits	No parity bits (no parity).			
Stop bit	Appending 1 bit.			
Data direction	LSB takes precedence			

Note It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark f_{MCK}: Operating clock frequency for object channels
f_{CLK}: System clock frequency

A summary of the receive operation of LIN is shown in Figure 12-115.

Figure 12-115 Receive operation of LIN



The flow of signal processing is as follows:

(1) The wake-up signal is detected by detecting the interrupt edge (INTP0) of the pin. When a wake-up signal is detected, in order to measure the low-level width of BF, TM03 is set to measure the pulse width and then enters the BF receive wait state.

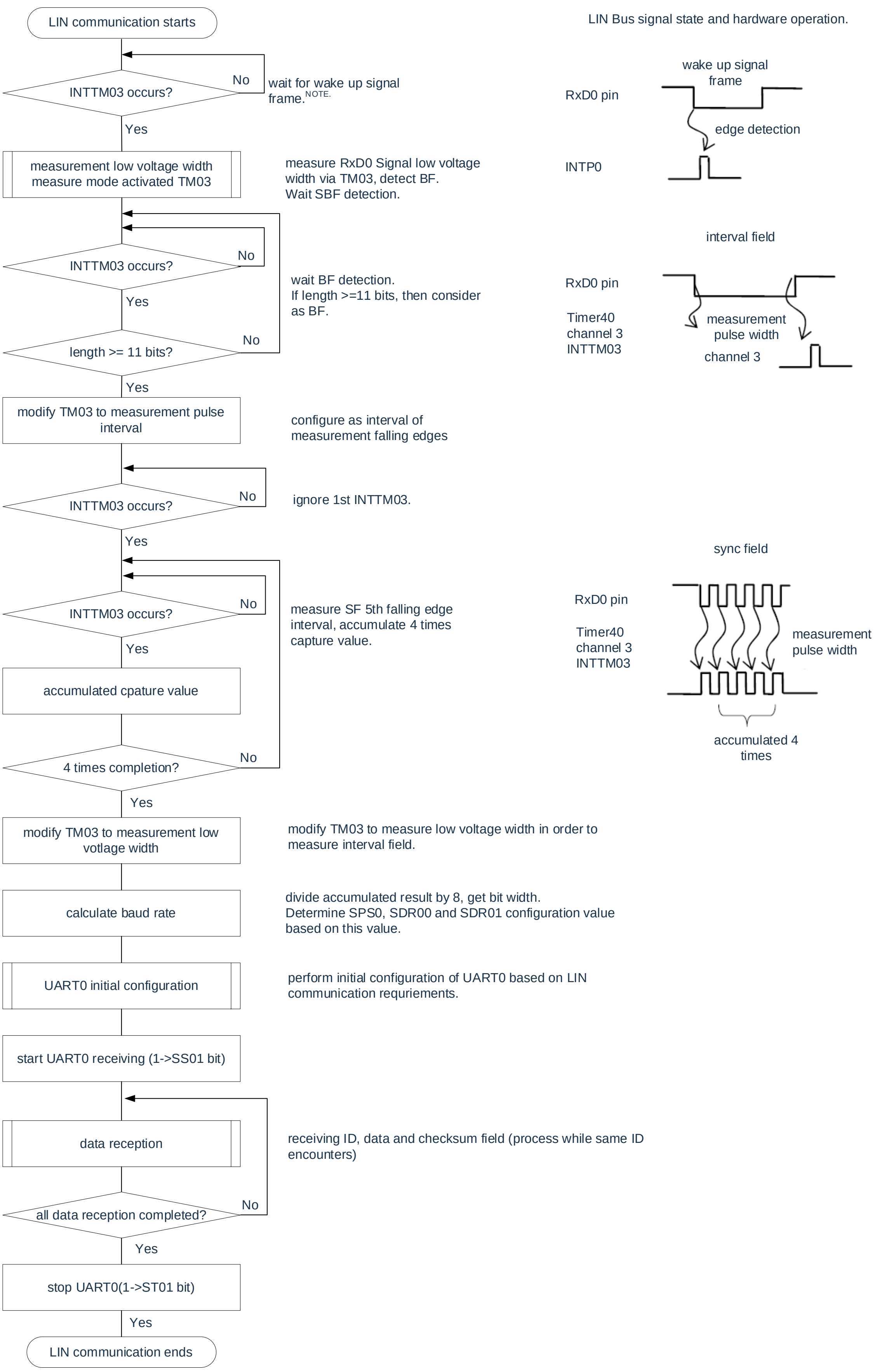
(2) If a falling edge of BF is detected, TM03 begins measuring the width of the low level and snaps at the rising edge of BF. Determine whether it is a BF signal based on the captured value.

(3) When BF reception ends normally, TM03 must be set to measure pulse intervals, and the interval between the falling edges of the RxD0 signal in the 4th sync field must be measured (see “5.8.4 Operation as input pulse interval measurements”).

(4) The baud rate error is calculated based on the bit interval of the sync field (SF). The baud rate must then be adjusted (reset) after pausing UART0 operation.

(5) The checksum field must be distinguished by software. UART0 must also be initialized by software after receiving the checksum field and set again to the BF receive wait state.

Figure 12-116 LIN reception


LIN communication starts
INTTM03 occurs?
No
wait for wake up signal frame.NOTE.
Yes
measurement low voltage width measure mode activated TM03
measure RxD0 Signal low voltage width via TM03, detect BF.
Wait SBF detection.
INTTM03 occurs?
No
Yes
wait BF detection.
If length >=11 bits, then consider as BF.
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.
INTTM03 occurs?
No
Yes
measure SF 5th falling edge interval, accumulate 4 times capture value.
accumulated cpature value
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
LIN Bus signal state and hardware operation.
wake up signal frame
RxD0 pin
edge detection
INTP0
interval field
RxD0 pin
Timer40 channel 3 INTTM03
measurement pulse width
channel 3
sync field
RxD0 pin
Timer40 channel 3 INTTM03
measurement pulse width
accumulated 4 times

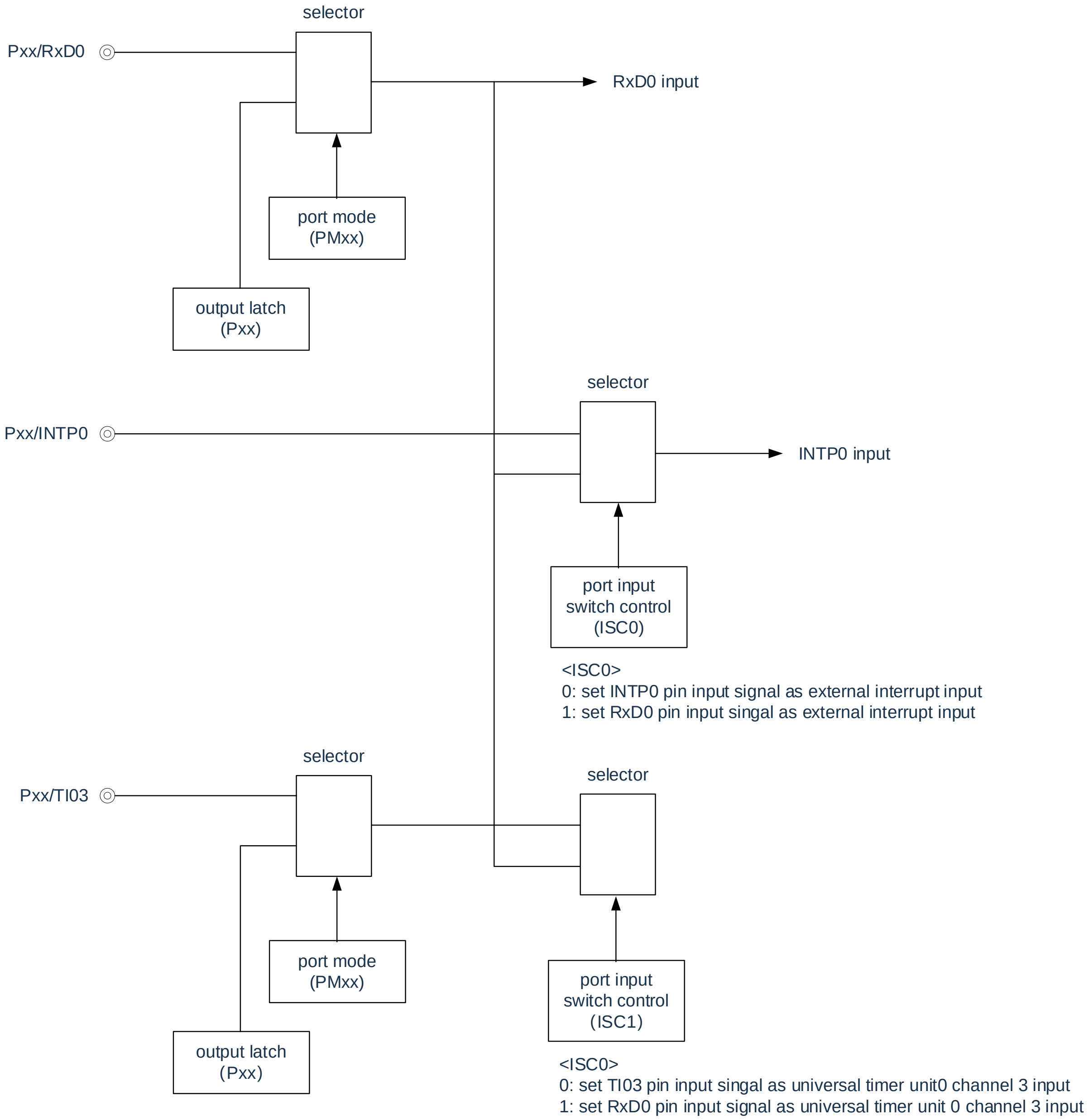
Note is only required during sleep.

The port structure diagram for LIN receive operation is shown in Figure 12-117.

The wake-up signal sent by the LIN master is received by the edge detection of the external interrupt (INTP0). It can be operated by external event capture of the universal timer unit, measuring the length of the synchronization segment sent by the LIN master and calculating the baud rate error.

With port input switching control (ISC0/ISC1), the input source for the received port input (RxD0) can be input to the external interrupt (INTP0) and timer array unit without external wiring.

Figure 12-117 Port block diagram for LIN receive operation


selector
Pxx/RxD0
RxD0 input
port mode
(PMxx)
output latch
(Pxx)
selector
Pxx/INTP0
INTP0 input
port input
switch control
(ISC0)
<ISC0>
0: set INTP0 pin input signal as external interrupt input
1: set RxD0 pin input singal as external interrupt input
selector
Pxx/TI03
selector
port mode
(PMxx)
port input
switch control
(ISC1)
output latch
(Pxx)
<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

Note ISC0, ISC1: Bit0 and bit1 of the input switching control register (ISC) (refer to Figure 14-19)

The peripheral functions used for LIN communication operation are summarized as follows:

< Peripheral Features Used >

- External Interrupt (INTP0): Detection of wake-up signals

Purpose: Detects the edge of the wake-up signal and the start of communication.

- Channel 3 of the universal timer unit: detection of baud rate error, detection of break field (BF).

Purpose: Detects the length of the sync field (SF) and detects baud rate error by dividing its length by the number of bits (the interval between RxD0 input edges is measured by snap mode). Measure the low-level width to determine if it is a break field (BF).

- Channel 0 and Channel 1 (UART0) of Universal Serial Communication Unit 0 (SCI0).

12.9 Simplified I^{2}C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication operation

This is a function that synchronizes clock communication with multiple devices through a total of 2 lines of serial clock (SCL) and serial data (SDA). Because this simplified I^{2}C is designed for single communication with EEPROM, flash memory, A/D converters, etc., it is only used as a master device.

For start and stop conditions, AC specifications must be adhered to and processed by software while operating the control registers.

[Transmit and receive data]

- Master sending, master receiving (limited to single master control functions).
- ACK output function Note, ACK detection function
- 8 bits of data length (when sending the address, specify the address with a high 7 bits, and use the lowest bit for R/W control).
- Generate start conditions and stop conditions through the software.

[Interrupt function]

- End of transfer interruption

[Error detection flag]

- ACK error

※[Features not supported by Simplified I^{2}C]

- Slave transmission, slave reception
- Multi-master function (arbitration failure detection function).
- Wait for detection function

Note When receiving the last data, if you write “0” to the SDOEmn bit (SDOEm register) to stop the output of the serial communication data, the ACK is not output. For details, please refer to “12.9.3(2) Processing Flow”.

Remark m: Unit number (m=0, 1) n: Channel number (n=0~3)mn=00~ 03, 10~11

Channels 0 to 3 of SCI0 and channels 0 to 1 of SCI1 support Simplified I^{2}C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) channels.

Simplified I^{2}C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21, IIC30, IIC31) have the following four types of communication operations:

- Address segment transmission (see 12.9.1).
- Datat transmission (see 12.9.2).
- Data reception (see 12.9.3).
- Generation of stop conditions (see12.9.4).

12.9.1 Address segment transmission

Address segment sending is the first transmission operation to specifically specify the transmitting object (slave device) that is the first to occur during I^2C communication. After generating the start condition, the address (7 bits) and the transmission direction (1 bit) are sent as 1 frame.

Simplified I^2C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21
Object channel	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1
The pin used	SCL00, SDA00Note1	SCL01, SDA01 Note 1	SCL10, SDA10 Note 1	SCL11, SDA11 Note 1	SCL20, SDA20 Note 1	SCL21, SDA21 Note 1
interrupt	INTIIC00	INTIIC01	INTIIC10	INTIIC11	INTIIC20	INTIIC21
	Limited to end-of-transmit interrupts (buffer null interrupts cannot be selected).					
Error detection flag	ACK Error Detection Flag (PEFmn).					
The length of the transmitted data	8 bits (send the highest 7 bits as the address and the lower 1 bit as R/W control).					
Transfer Rate Note 2	Max.f MCK/4[Hz] (SDRmn [15:9] ≥1) f_{MCK}: The operating clock frequency of the object channel, however, must meet the following conditions in each mode of I^2C: • Max.1MHz (Enhanced Fast Mode). • Max.400kHz (fast mode). • Max.100kHz (standard mode).					
Data level	Normal-phase output (default: high).					
Parity bits	No parity bits.					
Stop bit	Appeding 1 bit (for ACK reception).					
Data direction	MSB first					

Note 1. To communicate via Simplified I^2C, N-channel open-drain output mode (POMxx=1) must be set through the port output mode register (POMxx). For details, please refer to "Chapter 2 Pin Functions"

2. It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remarks m: Unit number (m=0, 1) n: Channel number (n=0~3)mn=00~ 03, 10~11.

(1) Register setting

Figure 12-118 Example of register setting contents when sending address segments of Simple I2C (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

Operation mode of channel n
0: Transmit completion 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	CKPmn 0	0	EOCmn 0	PTCmn1 0	PTCmn0 0	DIRmn 0	0	SLCmn1 0 Note2	SLCmn0 1	0	1	DLSmn1 1 Note3	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							

(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	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

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	SOEm3 0/1	SOEm2 0/1	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.

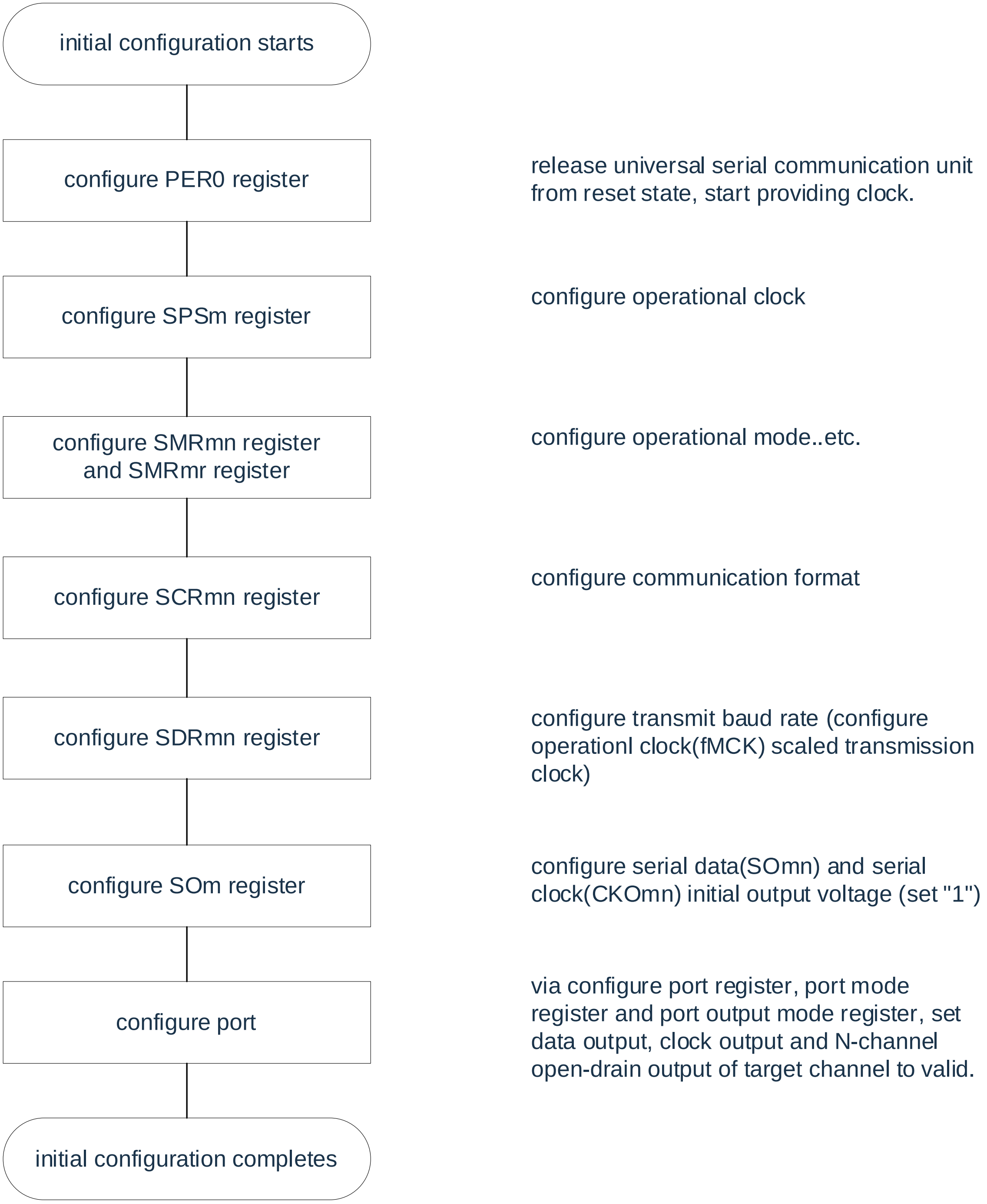
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	SSm3 0/1	SSm2 0/1	SSm1 0/1	SSm0 0/1

Note 1 Only for SMR00, SMR03, SMR11.
2. Limited to SCR00, SCR02, SCR10 only.
3. Limited to SCR00 register and SCR01 register, other fixed as “1”.

Note 1.m: Unit number (m=0, 1) n: channel number (n=0~3)r: IIC numbers (r=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11
2. ▣ : Fixed in IIC mode. ▨ : Cannot be set (initial value).
×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).
0/1: Set “0” or “1” according to the user's purpose.

(2) Operation Steps

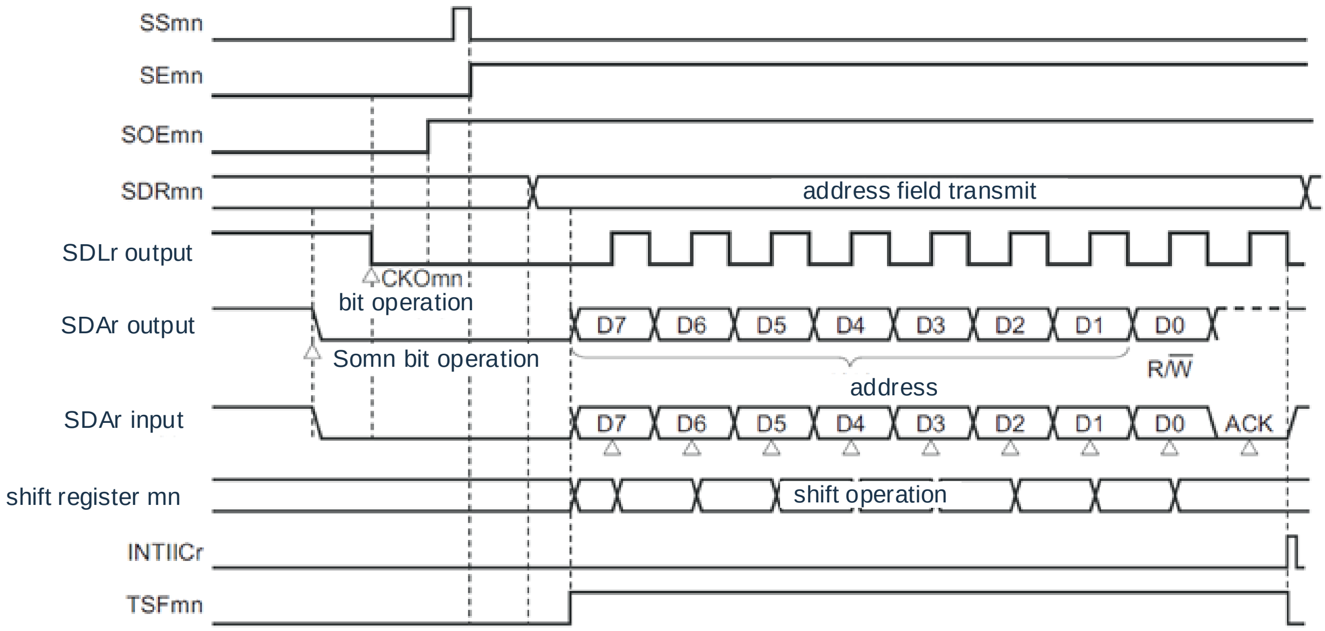
Figure 12-119 Initial setup step for the address segment transmission


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 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 SOm register
configure serial data(SOmn) and serial clock(CKOmn) initial output voltage (set "1")
configure port
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.
initial configuration completes

Note At the end of the initial setup, Simplified I2C (IIC00, IIC01, IIC10, IIC11, IIC20 IIC21) is output disabled and is in the operation stop state.

(3) Process flow

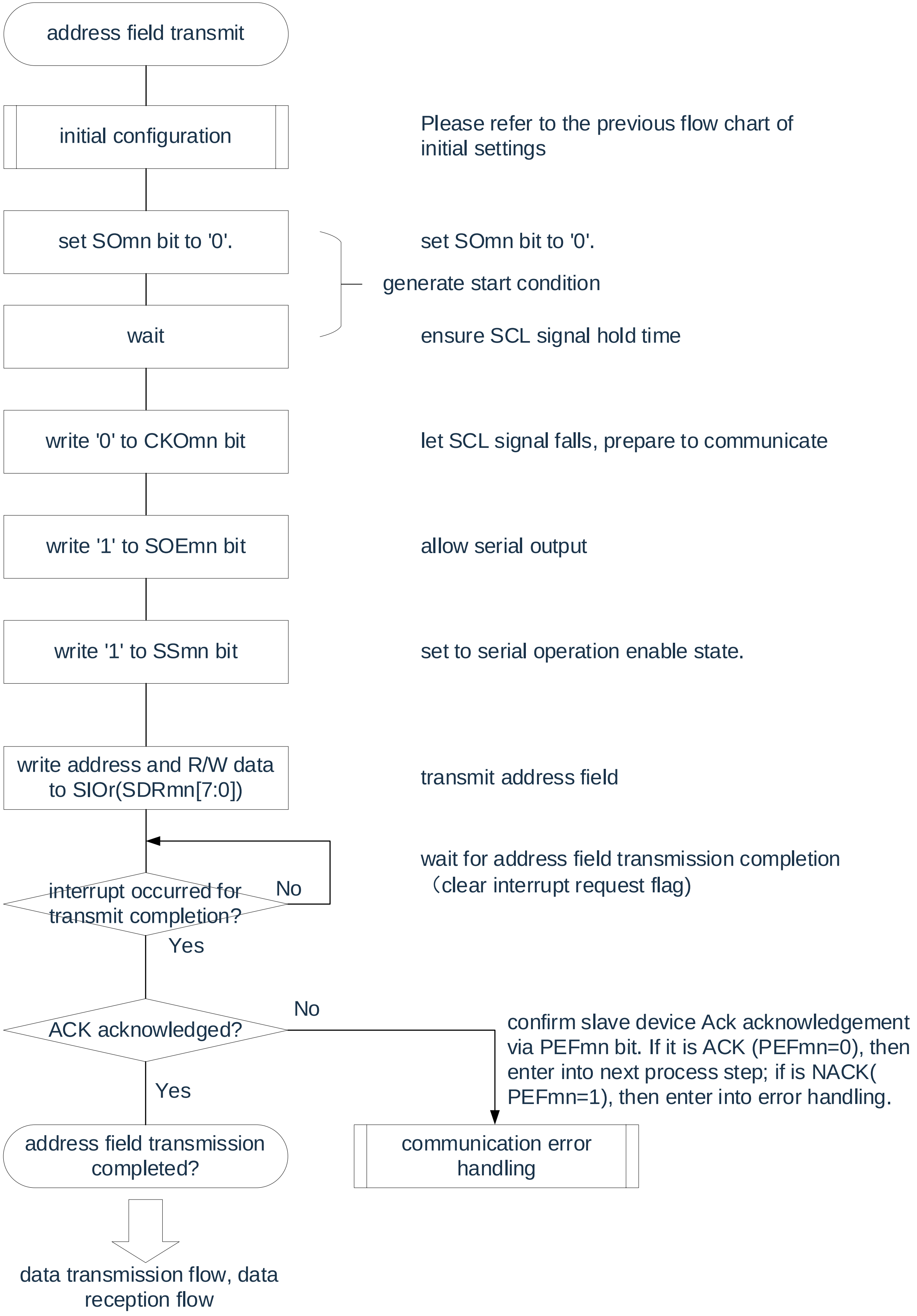
Figure 12-120 Timing diagram of the address segment transmission



Remarks m: Unit number (m=0, 1) n: Channel number (n=0~3) r: IIC numbers (r=00, 01, 10, 11, 20, 21)

mn=00~03, 10~11.

Figure 12-121 Flowchart of the address segment transmission


address field transmit
initial configuration
Please refer to the previous flow chart of initial settings
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

12.9.2 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 I2C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21
Object channel	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1
The pin used	SCL00, SDA00Note1	SCL01, SDA01 Note1	SCL10, SDA10 Note1	SCL11, SDA11 Note1	SCL20, SDA20 Note1	SCL21, SDA21 Note1
interrupt	INTIIC00	INTIIC01	INTIIC10	INTIIC11	INTIIC20	INTIIC21
	Limited to end-of-transmit interrupts (buffer null interrupts cannot be selected).					
Error detection flag	ACK error flag (PEFmn)					
The length of the transmitted data	8 bits					
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 be in I2 The following conditions are met in each mode of C: • Max.1MHz (Enhanced fast mode). • Max.400kHz (fast mode). • Max.100kHz (standard mode).					
Data level	Normal-phase output (default: high).					
Parity bits	No parity bits.					
Stop bit	Appending 1 bit (for ACK reception).					
Data direction	MSB first					

Note 1 To communicate via Simplified I2C, N-channel open-drain output mode (POMxx=1) must be set through the port output mode register (POMxx). For details, please refer to “2.3 Registers for Control Port Functions” and “2.5 Register Settings When Using the Multiplexing Function”.

2. It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark m: Unit number (m=0, 1) n: Channel number (n=0~3)mn=00~ 03, 10~11

(1) Register setting

Figure12-122 Example of register setting contents for simple 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 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

(b) serial communication operation configuration register mn (SCRmn)……do not operate bits other than TXEmn and RXEmn of this register wihle data is transmitting or receiving.

	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 [Note2]	SLCmn0 1	0	1	DLSmn1 1 [Note3]	DLSmn0 1

(c) serial data regsiter mn (SDRmn) (low 8 bit: SIOr) ……only lower 8 bits valid wihle data is transmitting or receiving.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDRmn	baud rate configuration [Note4]							0	configuration of transmit data							
									SIOr							

(d) serial output register m (Som) ……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
SOm	0	0	0	0	CKOm3 0/1 [Note5]	CKOm2 0/1 [Note5]	CKOm1 0/1 [Note5]	CKOm0 0/1 [Note5]	0	0	0	0	SOm3 0/1 [Note5]	SOm2 0/1 [Note5]	SOm1 0/1 [Note5]	SOm0 0/1 [Note5]

(e) serial otuput enable register m (SOEm) ……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
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	SOEm3 1	SOEm2 1	SOEm1 1	SOEm0 1

(f) serial channel start register m (SSm) ……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
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. Limited to SMR01, SMR03, SMR11 registers.
2. Limited to SCR00, SCR02, SCR10 registers.
3. Limited to SCR00 register and SCR01 register, other fixed as "1".
4. Because it is already set when sending the address segment, it does not need to be set.
5. During the operation of communication, the value changes due to the communication data.

Note 1.m: Unit number (m=0, 1) n: channel number (n=0~3)r: IIC numbers (r=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11
2. □ : Fixed in IIC mode. ▨ : Cannot be set (set initial value).
×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).
0/1: Set "0" or "1" according to the user's purpose.

(2) Process flow

Figure 12-123 Timing diagram of data transmission

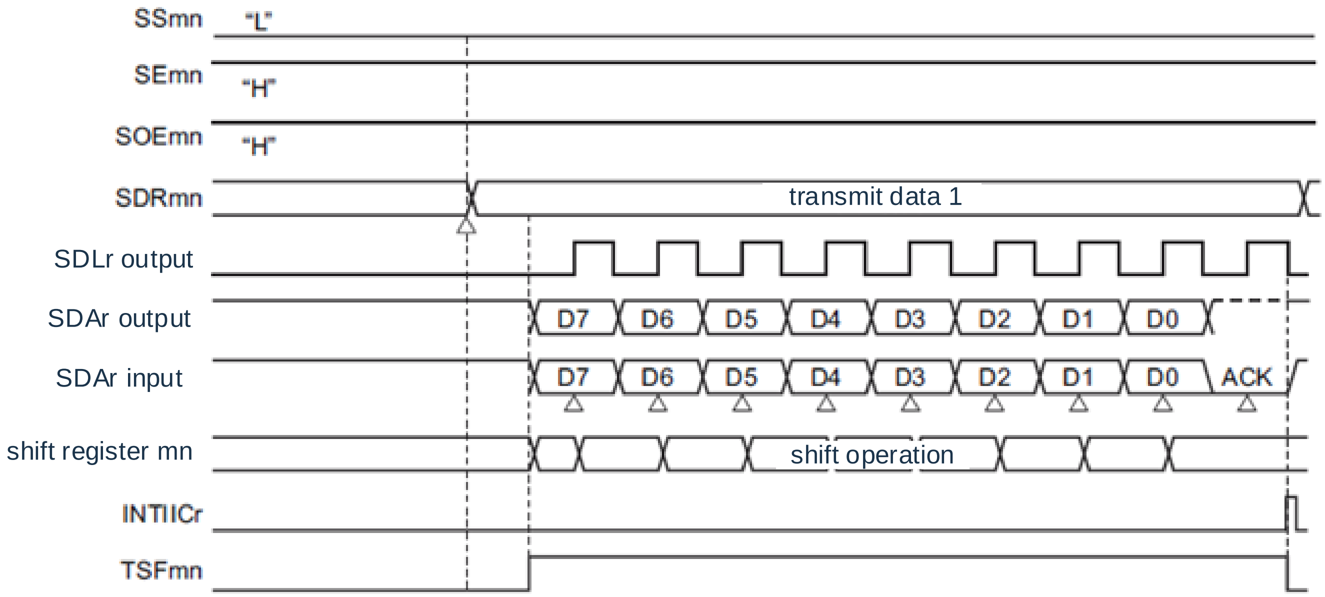
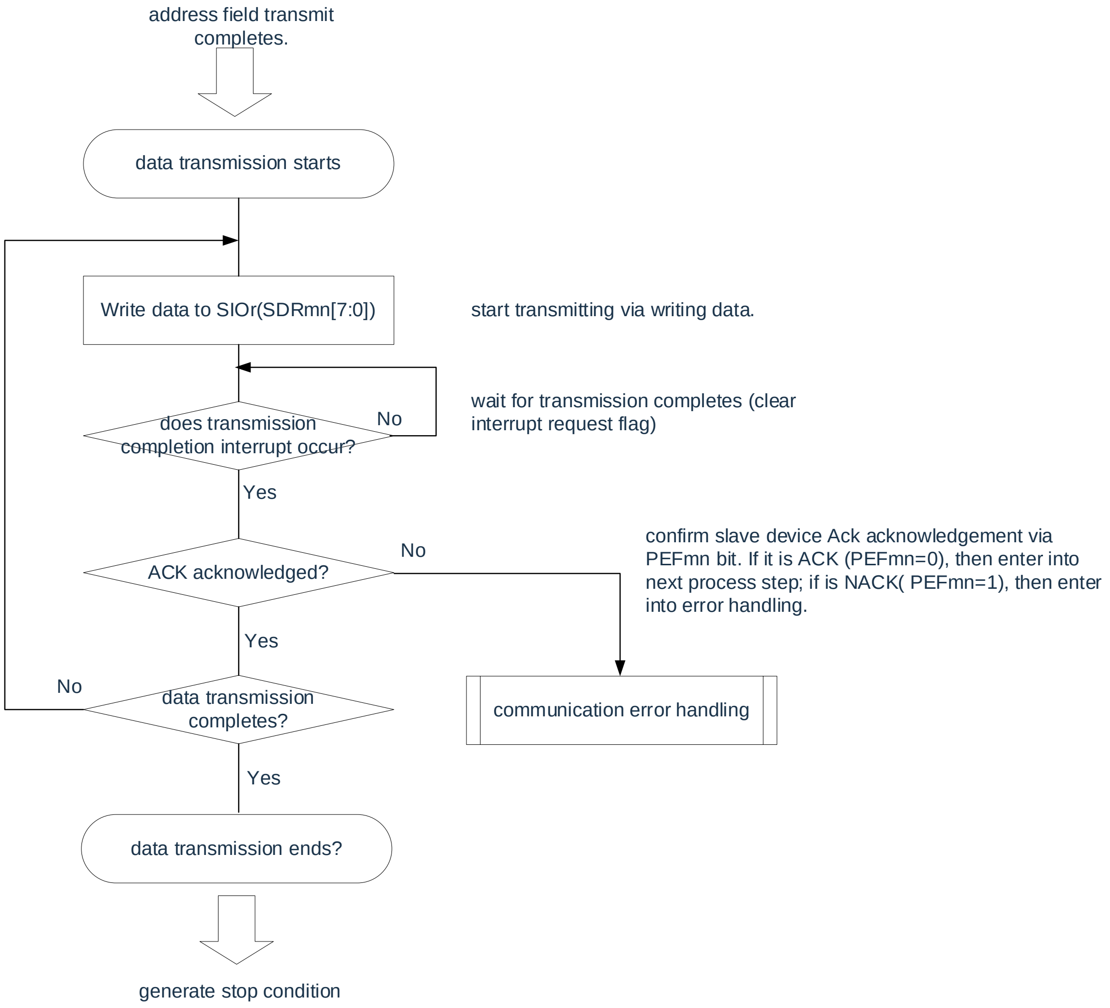


Figure 12-124 Flow chart of data transmission


address field transmit completes.
data transmission starts
Write data to SIOr(SDRmn[7:0])
does transmission completion interrupt occur?
No
Yes
ACK acknowledged?
No
Yes
data transmission completes?
No
Yes
communication error handling
data transmission ends?
generate stop condition

start transmitting via writing data.

wait for transmission completes (clear interrupt request flag)

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.

12.9.3 Data reception

Data reception is the operation of receiving data from a transmitting object (slave device) after sending an address segment. A stop condition is generated after receiving all data from an object slave and the bus is released.

Simplified I²C	IIC00	IIC01	IIC10	IIC11	IIC20	IIC21
Object channel	SCI0 Channel 0	SCI0 Channel 1	SCI0 Channel 2	SCI0 Channel 3	SCI1 Channel 0	SCI1 Channel 1
The pin used	SCL00, SDA00 Note1	SCL01, SDA01 Note1	SCL10, SDA10 Note1	SCL11, SDA11 Note1	SCL20, SDA20 Note1	SCL21, SDA21 Note1
interrupt	INTIIC00	INTIIC01	INTIIC10	INTIIC11	INTIIC20	INTIIC21
	Limited to end-of-transmit interrupts (buffer null interrupts cannot be selected).					
Error detection flag	Only the Overflow Error Detection Flag (OVFmn).					
The length of the transmitted data	8 bits					
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 be in I2 The following conditions are met in each mode of C: • Max.1MHz (Enhanced Fast Mode). • Max.400kHz (fast mode). • Max.100kHz (standard mode).					
Data level	Normal-phase output (default: high).					
Parity bits	No parity bits.					
Stop bit	Append 1 bit (ACK send).					
Data direction	MSB takes precedence					

Note 1 To communicate via Simplified I2C, N-channel open-drain output mode (POMxx=1) must be set through the port output mode register (POMxx). For details, please refer to “2.3 Registers for Control Port Functions” and “2.5 Register Settings When Using the Multiplexing Function”.

2. It must be used within the scope of the peripheral functional characteristics that meet this condition and meet the electrical characteristics (refer to the data sheet).

Remark m: Unit number (m=0, 1) n: Channel number (n=0~3) mn=00~ 03, 10~11.

(1) Register setting

Figure 12-125 Example of register setting contents for simple I2C data reception (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 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

(b) serial communication operation configuration register mn (SCRmn)……do not operate bits other than TXEmn and RXEmn of this register wihle data is transmitting or receiving.

	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 0	PTCmn1 0	PTCmn0 0	DIRmn 0	0	SLCmn1 0 Note2	SLCmn0 1	0	1	DLSmn1 1 Note3	DLSmn0 1

(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 Note4							0	virtual transmit data configuration (FFH)							
									SIOr							

(d) serial output register m (Som) ……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
SOm	0	0	0	0	CKOm3 0/1 Note5	CKOm2 0/1 Note5	CKOm1 0/1 Note5	CKOm0 0/1 Note5	0	0	0	0	SOm3 0/1 Note5	SOm2 0/1 Note5	SOm1 0/1 Note5	SOm0 0/1 Note5

(e) serial otuput enable register m (SOEm) ……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
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) ……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
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. SMR01, SMR03, SMR11 registers only.

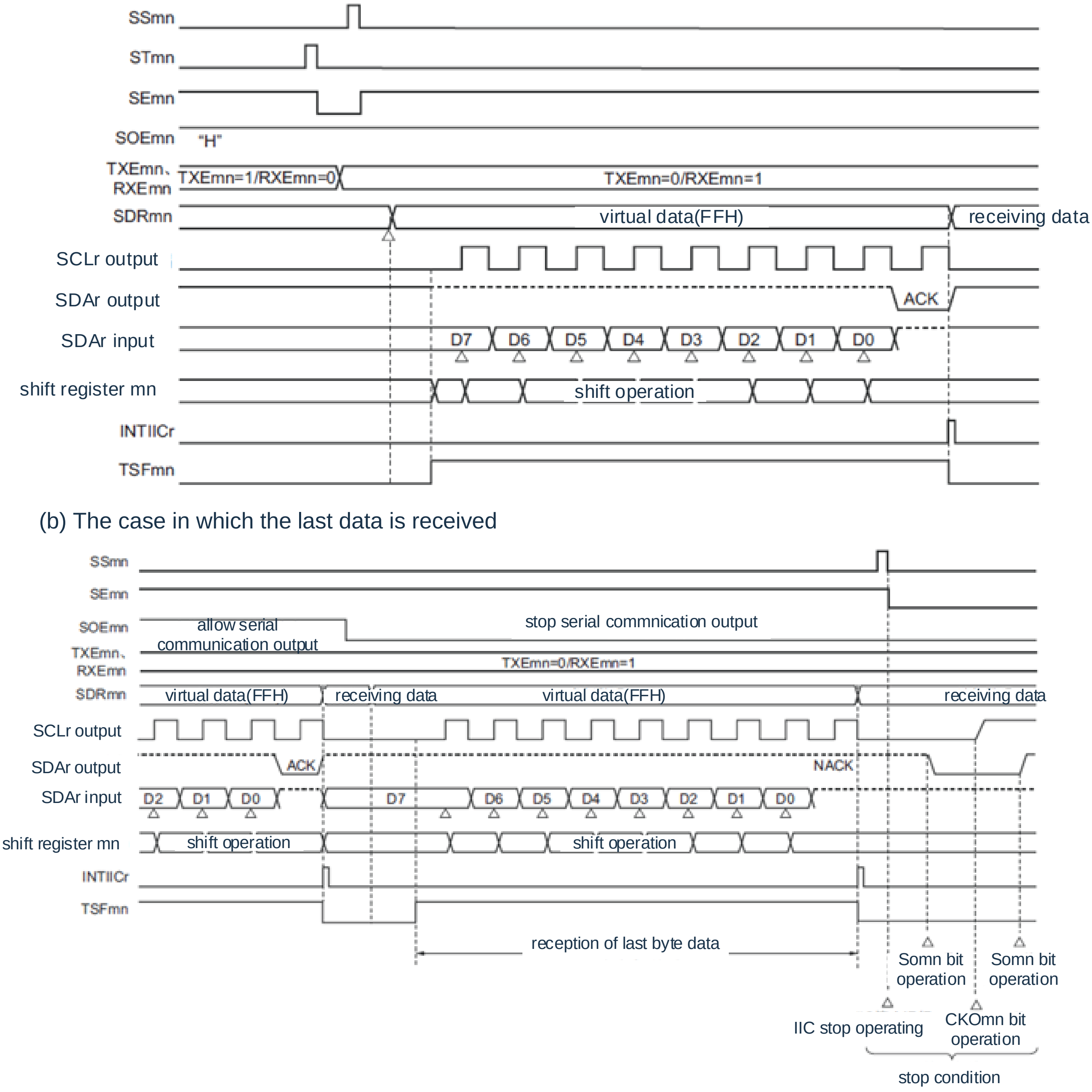
2. Limited to SCR00, SCR02, SCR10 registers only.
3. Limited to SCR00 register and SCR01 register, other fixed as “1”.
4. Because it is already set when sending the address segment, it does not need to be set.
5. During the operation of communication, the value changes due to the communication data.

Noticee 1.m: Unit number (m=0, 1) n: channel number (n=0~3)r: IIC numbers (r=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11.

2. □ : Fixed in IIC mode. ▨ : Cannot be set (set initial value).
×: This is the bit that cannot be used in this mode (set the initial value if it is not used in other modes either).
0/1: Set “0” or “1” according to the user's purpose.

(2) Process flow

Figure 12-126 Timing diagram of data reception

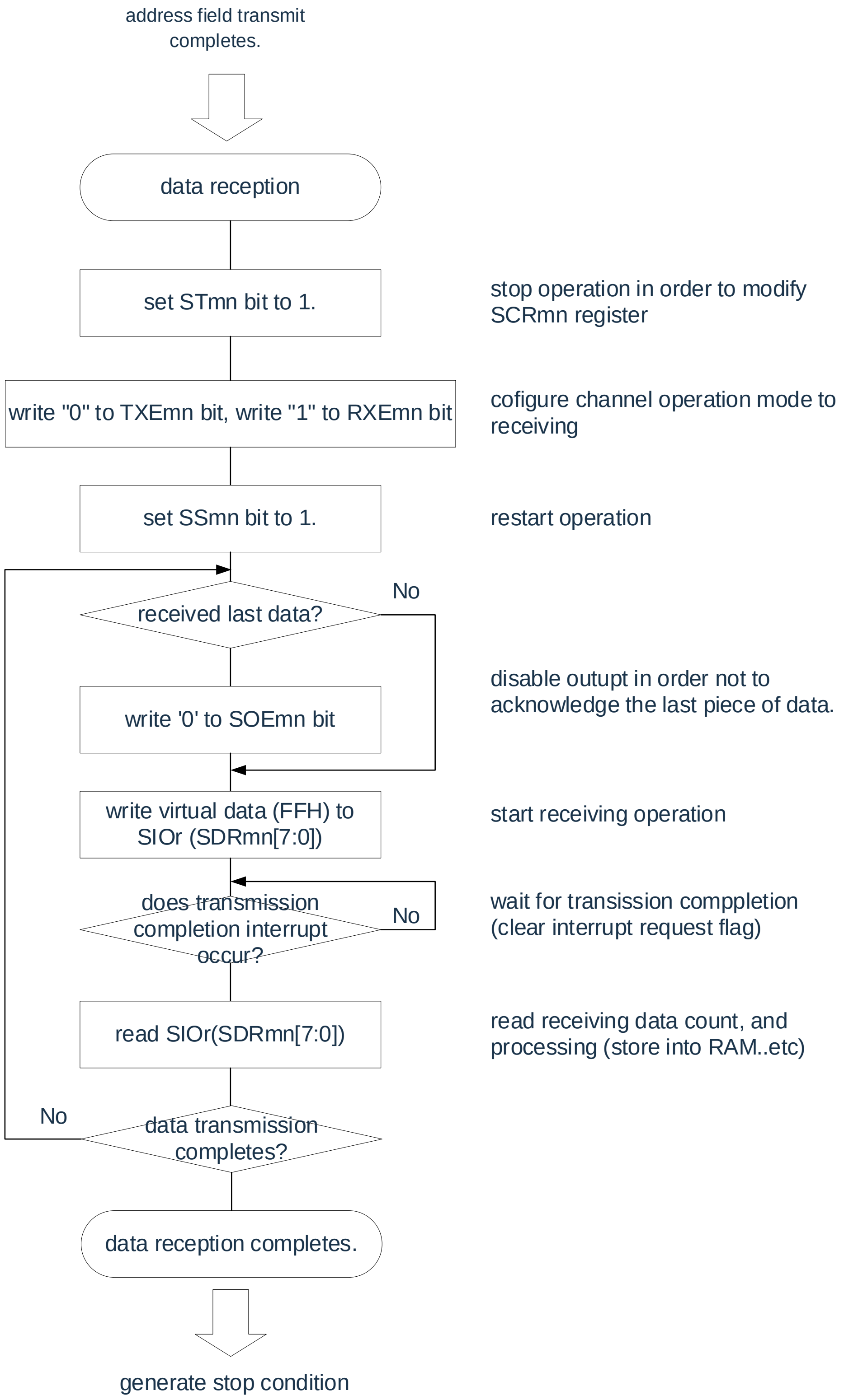


(a) Start of receiving data

(b) The case in which the last data is received

Remark m: Unit number (m=0, 1) n: Channel number (n=0~3) r: IIC numbers (r=00, 01, 10, 11, 20, 21)
mn=00~03, 10~11.

Figure 12-127 Flowchart 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
disable outupt in order not to acknowledge the last piece of data.
write '0' to SOEmn bit
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

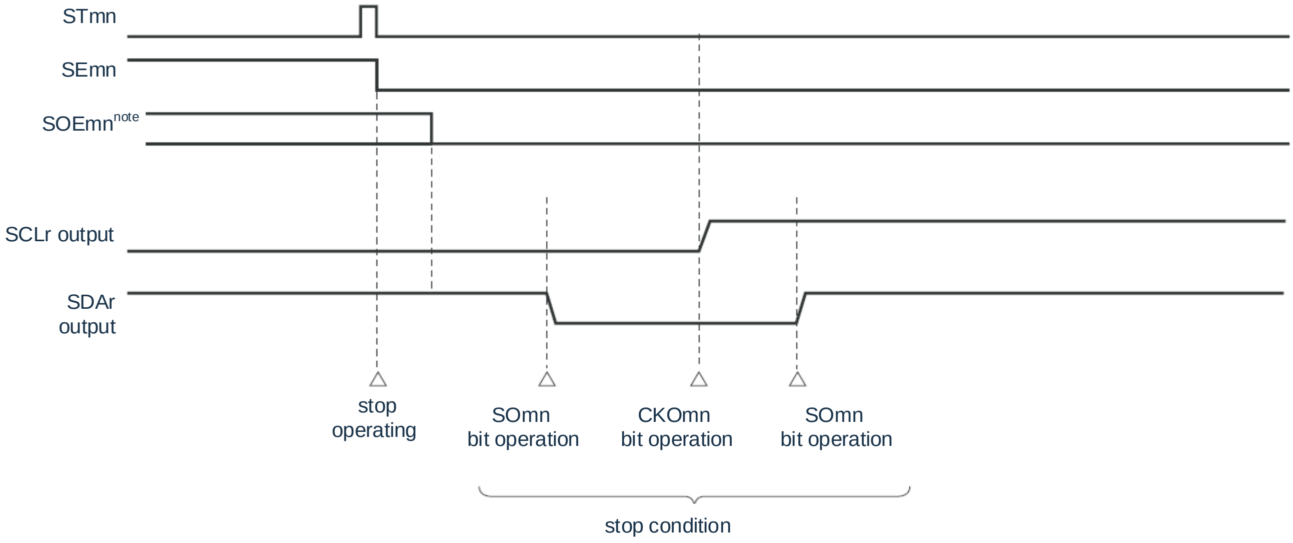
Note ACK is not output when receiving the last data (NACK). Thereafter, operation is stopped by setting the STmn bit of the serial channel stop register m (STm) to "1", and then a stop condition is generated to end communication.

12.9.4 Generation of stop condition

After all data is sent and received with the object slave, a stop condition is created and the bus is released.

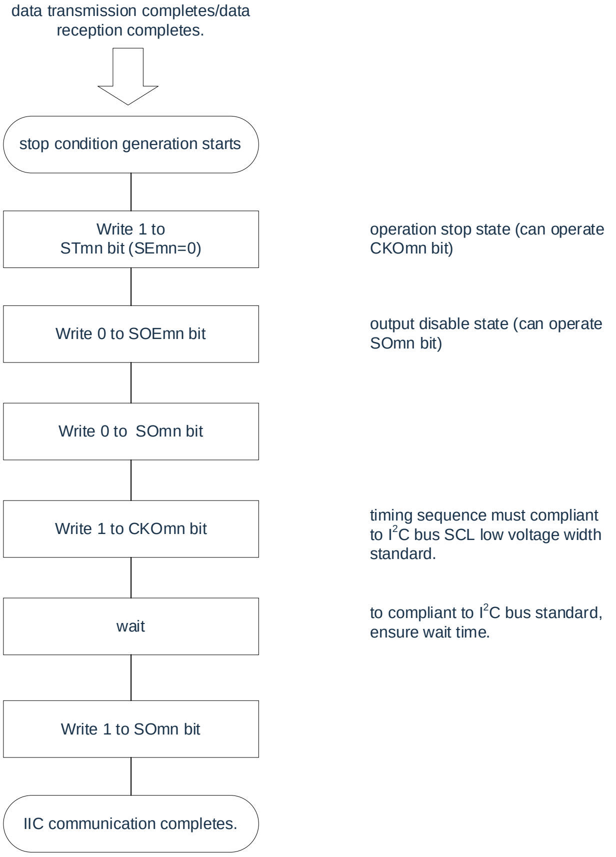
(1) Process flow

Figure 12-128 Timing diagram for generating stop condition



Note The SOEmn bit of the serial output allow register m (SOEm) is set to "0" before the last data is received.

Figure 12-129 Flow chart of generating a stop condition


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.

12.9.5 Calculation of the transfer rate

Simplified I^2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) The transmission rate of communication can be calculated using the following calculation formula.

(Transfer Rate) = {Running clock (f_{MCK}) frequency of the object channel}× (SDRmn[15:9]+ 1) ÷2

Notice Setting SDRmn[15:9] to “0000000B” is prohibited, and the SDRmn [15:9] must be greater than or equal to “0000001B”. The duty cycle of the SCL signal output by the simple I2C is 50%. In the I2 C-bus specification, the low width of the SCL signal is greater than the high level width. Therefore, if set to 400kbps in fast mode or 1Mbps in enhanced fast mode, the low level width of the SCL signal output is less than I2C The specification value of the bus. SDRmn[15:9] must be given a value that meets the I2 C-bus specification.

Note 1 Because the value of SDRmn[15:9] is the value of bit15~9 of the serial data register (SDRmn) (0000001B~1111111B).), so 1 to 127.

2. m: Unit number (m=0, 1) n: channel number (n=0~3) mn=00~03 , 10~11.

The operating clock (f_{MCK}) depends on the serial clock select register m (SPSm) and bit 15 (CKSmn bit) of the serial mode register mn (SMRmn).

Table 12-5 Simplified I²C Running Clock Selection

SMRmn register	SPSm register								Running clock (f_{MCK}) Note	
CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00		f_{CLK}=32MHz operation
0	X	X	X	X	0	0	0	0	f_{CLK}	32MHz
	X	X	X	X	0	0	0	1	$f_{CLK}/2$	16MHz
	X	X	X	X	0	0	1	0	$f_{CLK}/2^2$	8MHz
	X	X	X	X	0	0	1	1	$f_{CLK}/2^3$	4MHz
	X	X	X	X	0	1	0	0	$f_{CLK}/2^4$	2MHz
	X	X	X	X	0	1	0	1	$f_{CLK}/2^5$	1MHz
	X	X	X	X	0	1	1	0	$f_{CLK}/2^6$	500kHz
	X	X	X	X	0	1	1	1	$f_{CLK}/2^7$	250kHz
	X	X	X	X	1	0	0	0	$f_{CLK}/2^8$	125kHz
	X	X	X	X	1	0	0	1	$f_{CLK}/2^9$	62.5kHz
	X	X	X	X	1	0	1	0	$f_{CLK}/2^{10}$	31.25kHz
	X	X	X	X	1	0	1	1	$f_{CLK}/2^{11}$	15.63kHz
1	0	0	0	0	X	X	X	X	f_{CLK}	32MHz
	0	0	0	1	X	X	X	X	$f_{CLK}/2$	16MHz
	0	0	1	0	X	X	X	X	$f_{CLK}/2^2$	8MHz
	0	0	1	1	X	X	X	X	$f_{CLK}/2^3$	4MHz
	0	1	0	0	X	X	X	X	$f_{CLK}/2^4$	2MHz
	0	1	0	1	X	X	X	X	$f_{CLK}/2^5$	1MHz
	0	1	1	0	X	X	X	X	$f_{CLK}/2^6$	500kHz
	0	1	1	1	X	X	X	X	$f_{CLK}/2^7$	250kHz
	1	0	0	0	X	X	X	X	$f_{CLK}/2^8$	125kHz
	1	0	0	1	X	X	X	X	$f_{CLK}/2^9$	62.5kHz
	1	0	1	0	X	X	X	X	$f_{CLK}/2^{10}$	31.25kHz
	1	0	1	1	X	X	X	X	$f_{CLK}/2^{11}$	15.63kHz
Beyond the above									Disable settings.	

Notice To change the clock selected as f_{CLK} (change the value of the system clock control register (CKC)), you must stop the operation of the universal serial communication unit (SCI) (serial channel stop register m(STm)=000FH) after making the change.

Note 1.X: Ignore
2.m: Unit number (m=0, 1) n: channel number (n=0~3)mn=00~03 , 10~11.

An example of setting the I²C transfer rate at f_{MCK}=f_{CLK}=32MHz is shown below.

I²C transfer mode (Expected transfer rate)	f_{CLK}=32MHz			
	Running 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 Because the duty cycle of the SCL signal is 50%, the error cannot be set to about “0”%.

12.9.6 Processing steps when an error occurs in a simple I2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication process

The processing steps when an error occurs during a simple I2C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication are shown in Figure 12-130 and Figure 12-131.

Figure 12-130 Steps to handle when an overflow error occurs

Software operation	Hardware status	Remark
Read serial data register mn (SDRMN).	The BFF m n bit of the SSRm n register is "0" and channel n is receivale.	This is to prevent overflow errors from ending the next reception during mishandling.
Read serial status register mn(SSRmn).		The type of error is judged, and the reading value is used to clear the error flag.
Write "1" to the serial flag clear trigger register mn (SDIRmn).	Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors during read operations can only be cleared.

Figure 12-131 Processing steps when an ACK error occurs in a simplified I^{2}C mode

Software operation	Hardware status	remark
Read the serial status register mn(SSRmn).		Determine the error category, and read the value to remove the error marker.
Write the serial flag to clear the trigger register mn (SDIRmn)。	Clear the error flag.	By writing the read value of the SSRmn register directly to the SDIRmn register, errors during read operations can only be cleared.
Set the STmn bit of the serial channel stop register m (STm) to "1".	The SEmn bit of the Serial Channel Enable Status Register m (SEm) is "0" and channel n is running stop.	Because ACK is not returned, the slave device is not ready for receiving. Thus, a stop condition is generated and the bus is released, and communication is started again from the start bar, or a restart can also be generated and start again from the address to send.
Generate a stop condition.		
Generate a start conditions		
Set the serial channel start register m (SSm) to SSmn bit to "1".	The SEmn bit of the Serial Channel Enable Status Register m (SEm) is "1" and channel n is operational	

Remark m: Unit number (m=0, 1) n: Channel number (n=0~3) r: IIC numbers (r=00, 01, 10, 11, 20, 21)

mn=00~03, 10~1.

Chapter 13 Serial Interface SPI

13.1 Serial interface SPI function

The serial interface SPI has the following two modes.

(1) Operation Stop mode

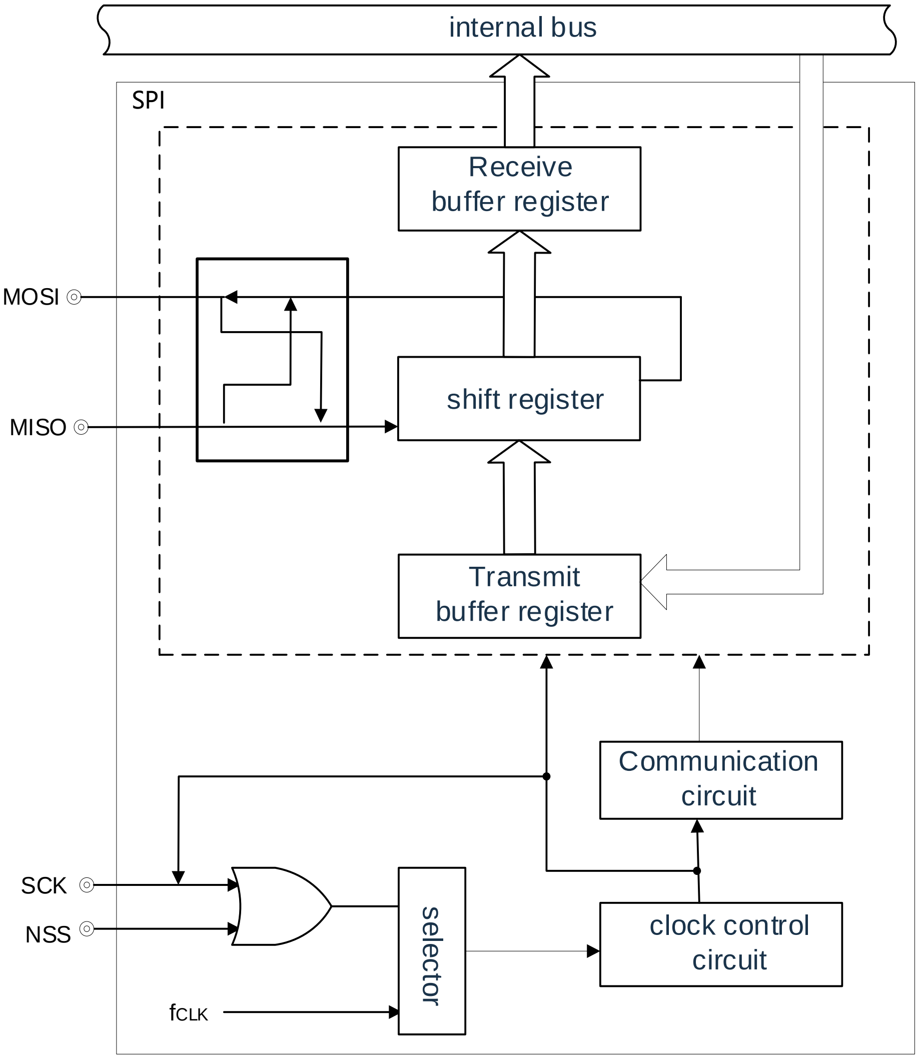
This is a mode used when no serial transfer is taking place, which reduces power consumption.

(2) 3-wire serial I/O mode

This mode transfers 8- or 16-bit data to multiple devices via 3 wires of the serial clock (SCK) and serial data bus (MISO and MOSI).

13.2 Structure of SPI

Figure13-1 Block diagram of serial interface SPI



13.3 Registers for controlling SPI

The serial interface SPI is controlled through the following registers.

- Peripheral enable register 0 (PER0).
- Serial operating mode register (SPIM).
- Serial clock selection register (SPIC).
- Transmit buffer register (SDRO).
- Receive buffer register (SDRI).
- Port mode register (PMxx).
- Port mode control register (PMCxx).
- Port register (Pxx).

13.3.1 Peripheral enable register 0 (PER0)

The PER0 register is a register that sets the clock to be allowed or disallowed to be supplied to each peripheral hardware.

Reduce power consumption and noise by stopping clocking unused hardware.

To use the SPI feature, SPIEN must be set to “1”.

See “4.3.6 Peripheral Enable Registers 0, 1 (PER0, PER1)” for details.

13.3.2 SPI operating mode register (SPIM)

SPIM is used to select the operating mode and control the allow or disallow of the operation.

SPIM can be set by 8-bit storage operation instructions.

A reset signal is generated to clear the register to 00H.

Figure 13-2 Format of mode control register (SPIM)

Address: 0x40042400 After reset: 00HR/W Note 1

symbol	7	6	5	4	3	2	1	0
SPIM	SPIES	TRMD	NSSE	You	INTMD	Dls	SDRIF	SPTF

SPIES	SPI operation enable
0	Stop running.
1	Allow to run.

TRMD[Note3]	Transmit/receive mode control
0	Receive mode
1	Transmit /receive mode

NSSE[Note4]	NSS pin use selection
0	The NSS pin is not used
1	Use the NSS pin

You	Data transfer order selection
0	Performs MSB-first input/output.
1	Perform LSB- first input/output.

INTMD	Interrupt source selection
0	The end of the transfer is interrupted
1	Null interrupt for sending buffers

Dls	The setting of the data length
0	8-bit data length
1	16-bit data length

SDRIF	Receive buffer non-null flag bits
0	There is no new received valid data in the receive cache
1	There is valid data received in the receive cache. When the register SDRI is read, the bit is cleared to 0

SPTF Note 2	Communication status flag bits
0	Communication stop
1	Communication is in progress

Note: 1. Bits 0 and 1 are read-only bits.
2. When SPTF=1 (during serial communication), rewriting TRMD, DIR, NSSE is prohibited.
3. The MO or SO output is fixed low when the TRMD is 0.
4. Fix the NSS pin input level to 0 or 1 before setting the bit to 1.

13.3.3 SPI clock selection register (SPIC)

This register specifies the timing of data sending/receiving and sets the serial clock.

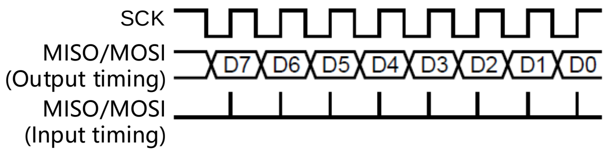
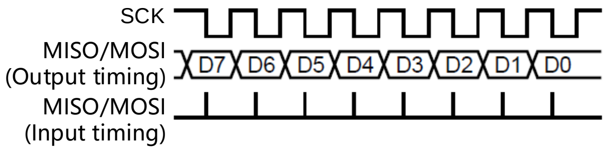
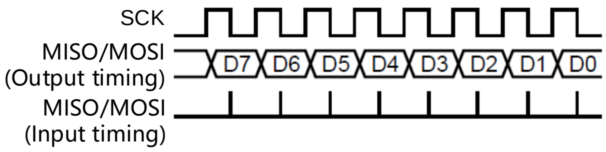
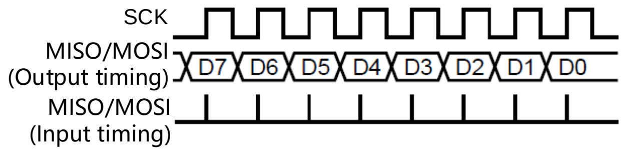
It can be set by 8-bit storage operation instructions.

A reset signal is generated to clear the register to 00H.

Figure13-3 Format of clock selection register (SPIC)

Address: 0x40042404 After reset: 00HR/W

symbol	7	6	5	4	3	2	1	0
SPIC	0	0	0	CKP	Dap	CKS2	CKS1	CKS0

CKP	DAP	Designation of data transmission/reception timing	Type
0	0	 SCK MISO/MOSI (Output timing) D7 D6 D5 D4 D3 D2 D1 D0 MISO/MOSI (Input timing)	1
0	1	 SCK MISO/MOSI (Output timing) D7 D6 D5 D4 D3 D2 D1 D0 MISO/MOSI (Input timing)	2
1	0	 SCK MISO/MOSI (Output timing) D7 D6 D5 D4 D3 D2 D1 D0 MISO/MOSI (Input timing)	3
1	1	 SCK MISO/MOSI (Output timing) D7 D6 D5 D4 D3 D2 D1 D0 MISO/MOSI (Input timing)	4

CKS2	CKS1	CKS0	SPI serial clock selection	mode
0	0	0	fCLK	Master mode
0	0	1	fCLK/2	
0	1	0	fCLK/2^2	
0	1	1	fCLK/2^3	
1	0	0	fCLK/2^4	
1	0	1	fCLK/2^5	
1	1	0	fCLK/2^6	
1	1	1	The external clock entered from the SCK	Slave mode

Note 1. Writing is disabled when SPPIE=1 (operation enable).

2. The phase type of the data clock after reset is type 1.

13.3.4 Transmit buffer registers (SDRO)

This register sets the data to be sent.

When bits 7 (SPIE) and 6 (TRMD) of the serial operating mode register (SPIM) are set to 1, data is written through SDRO starts sending/receiving.

Serial I/O shift registers convert data from the SDRO from parallel to serial data and output to the serial output pins.

SDRO can be written or read using 8-bit or 16-bit storage operation instructions.

A reset signal is generated to clear the register to 0000H.

Figure13-4 Format of transmit buffer register (SDRO)

Address: 0x40042408 After reset: 0000HR/W

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

13.3.5 Receive buffer register (SDRI)

This register stores the received data.

If bit 6 (TRMD) of the serial mode of operation register (SPIM) is set to 0, the reception begins by reading data from the SDRI.

During reception, data is read from the serial input pins into the SDRI.

The SDRI can be read using 8-bit or 16-bit storage operation instructions.

A reset signal is generated to clear the register to 0000H.

Figure13-5 Format of receive buffer register (SDRI)

Address: 0x4004240C After reset: 0000HR

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

13.3.6 SPI pin port function control register

When using SPI, the control registers (port mode registers (PMxx, PMCxx) of the port function that are multiplexed with the SPI input and output pins must be set. For details, please refer to “2.3.1 Port Mode Register (PMxx)”.

When using the SPI pin multiplexed port as an output of SCK/SO/MO, you must set bit "0" to the port mode register (PMxx, PMCxx) corresponding to each port. 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 must be set for each port. 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 Function".

13.4 Operation of SPI

In 3-wire serial I/O mode, data is sent or received in 8- or 16-bit units. The data bit is transmitted or received in synchronously with the serial clock.

After communication begins, bit 0 (SPTF) of SPIM is set to 1. When the communication of the data is complete, set the Communication Completion Interrupt Request Flag (SPIIF) and clear the SPTF to 0. Then enable the next communication.

Precautions

1. When SPTF=1 (during serial communication), access to control registers and data registers is prohibited.

2. It must be used within the scope of the SCLK Cycle Time (t_{KCY}) characteristics. Please refer to the data sheet for details.

13.4.1 Master tramission and reception

If bit 6 (TRMD) of the Serial Operating Mode Register (SPIM) is 1, data can be sent or received. When a value is written to the send buffer register (SDRO), the send/receive begins.

（1） Procedure

Figure 13-6 Initial setup steps of the master send/receive

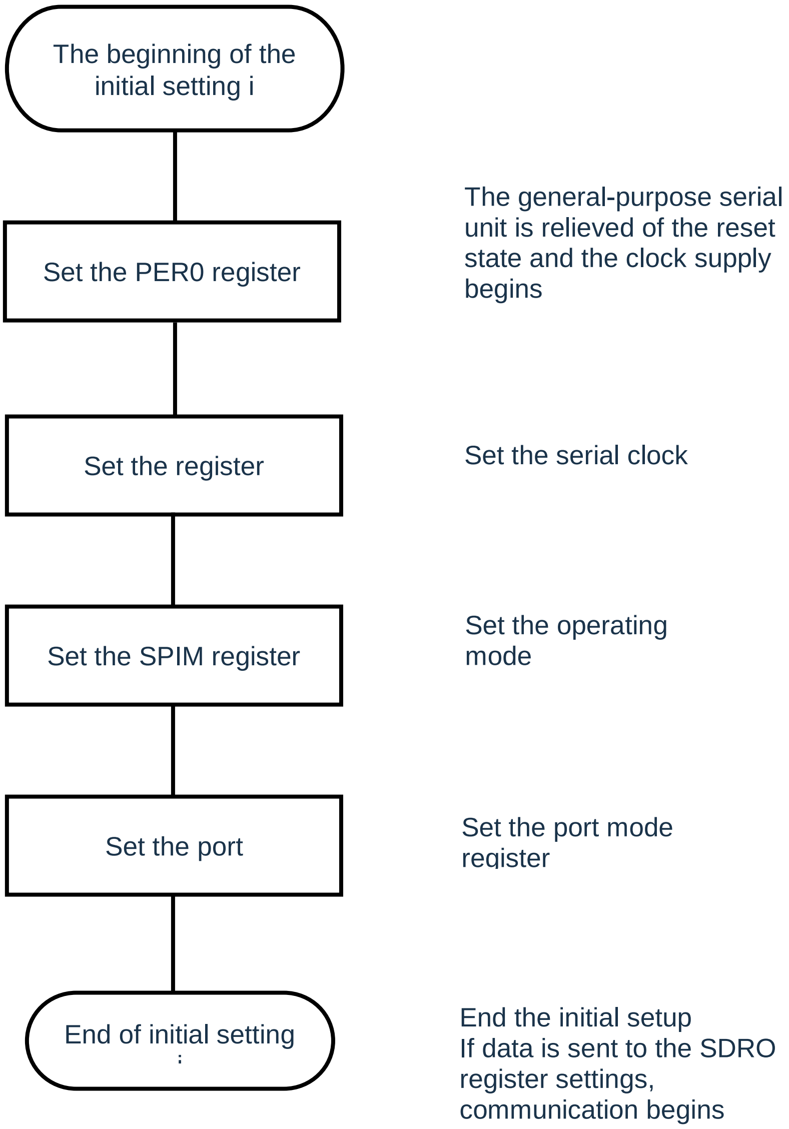
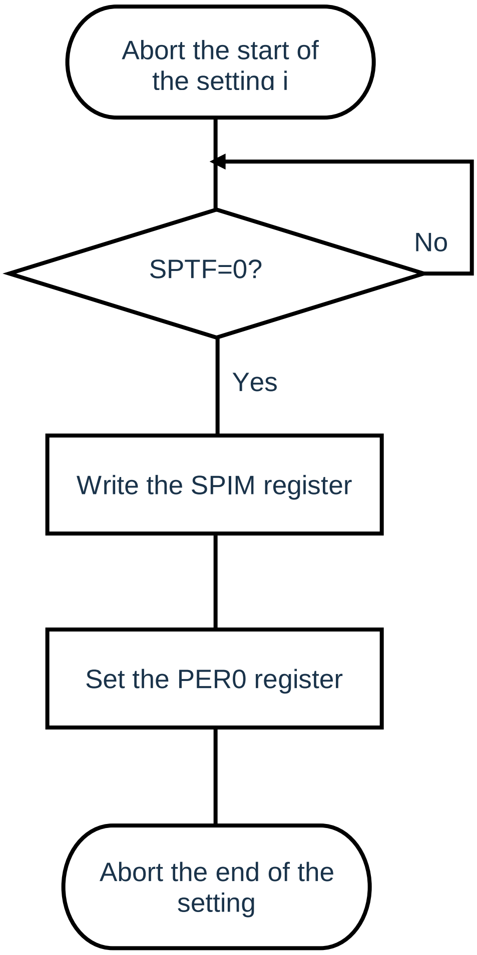

The beginning of the initial setting i
Set the PER0 register
Set the register
Set the SPIM register
Set the port
End of initial setting
:
The general-purpose serial unit is relieved of the reset state and the clock supply begins
Set the serial clock
Set the operating mode
Set the port mode register
End the initial setup
If data is sent to the SDRO register settings, communication begins

Figure13-7 Stop step of the master transmit/receive


Abort the start of the settina i
SPTF=0?
No
Yes
Write the SPIM register
Set the PER0 register
Abort the end of the setting

If there is data being transferred, wait for the transfer to end

Put the SPIE position "0" to stop the SPI from running

To use deep sleep mode, stop the clock of the SPI unit and set the reset state

(2) Processing process

Figure 13-8 Timing diagram of receive timing (single transmit mode) (INTMD=0, DAP=0, CKPmn=0)

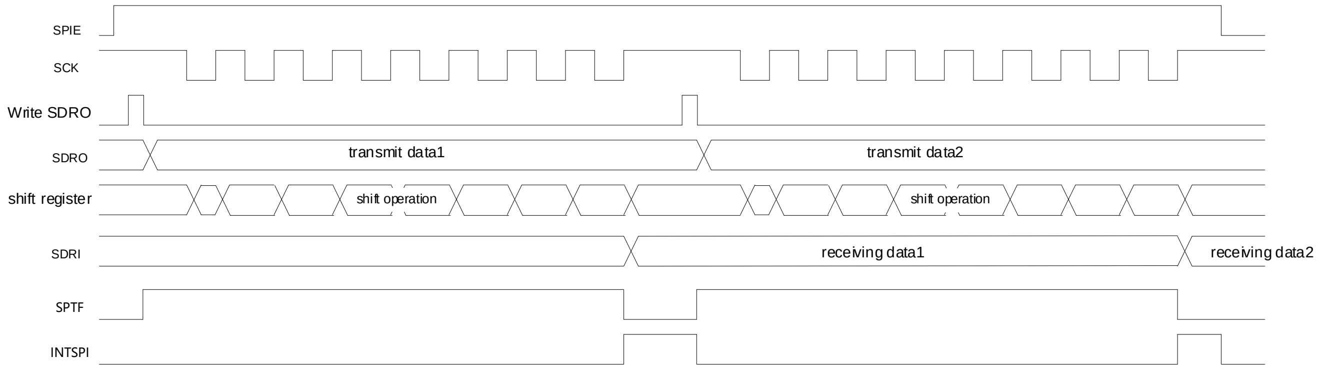
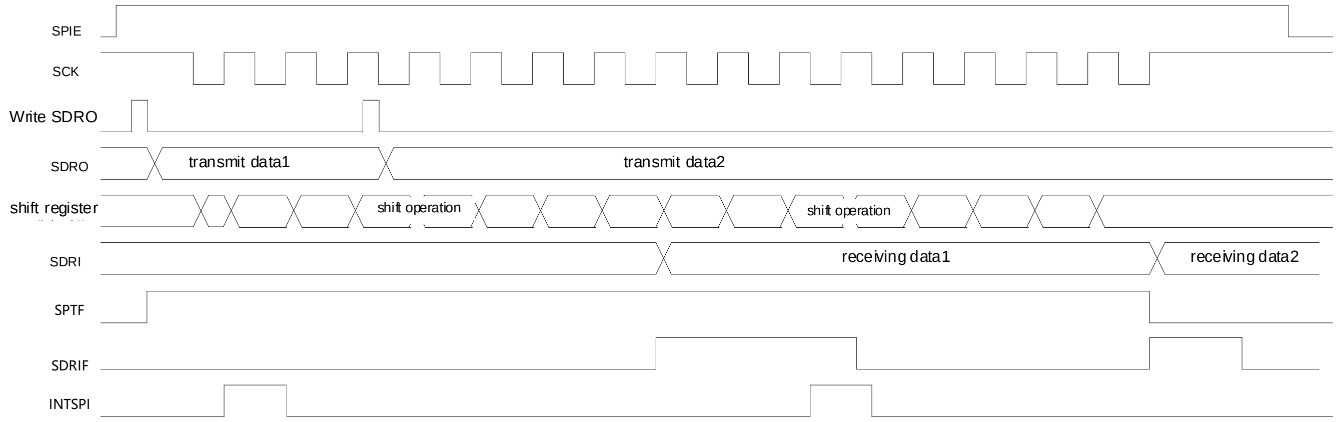


Fig13-9 Timing diagram of transmit/receive (continuous transmit mode) (INTMD=1, DAP=0, CKPmn=0)

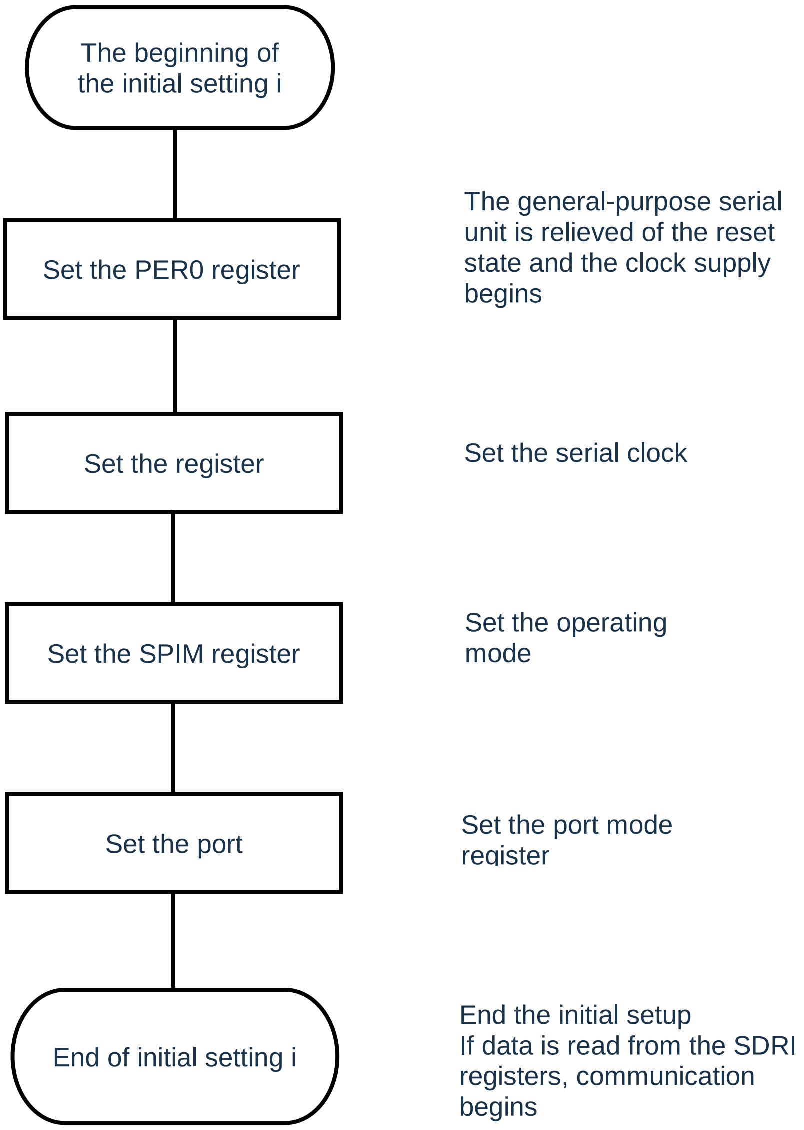


13.4.2 Master reception

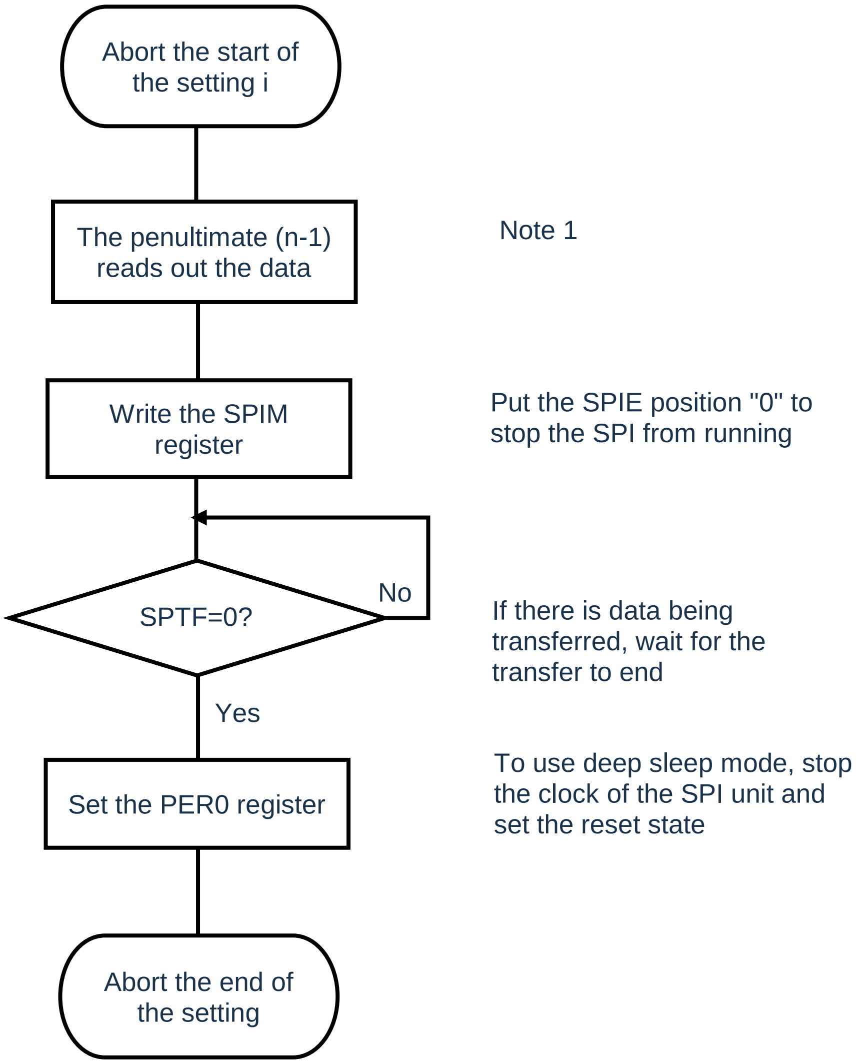
If Bit 6 (TRMD) of the Serial Operating Mode Register (SPIM) is 0, only data can be received. Receive begins when data is read from the receive buffer register (SDRI).

（1） Procedure

Figure 13-10 The initial setup step of the master reception


The beginning of the initial setting i
Set the PER0 register
The general-purpose serial unit is relieved of the reset state and the clock supply begins
Set the register
Set the serial clock
Set the SPIM register
Set the operating mode
Set the port
Set the port mode register
End of initial setting i
End the initial setup
If data is read from the SDRI registers, communication begins

Figures 13-11 Stop step of master receive



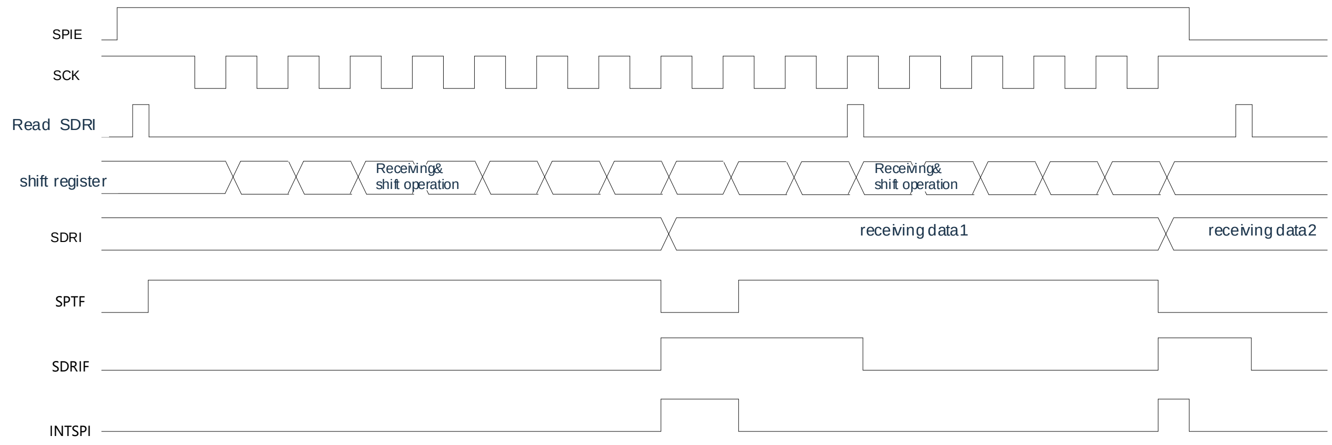
Abort the start of the setting i
The penultimate (n-1) reads out the data
Note 1
Write the SPIM register
Put the SPIE position "0" to stop the SPI from running
SPTF=0?
No
If there is data being transferred, wait for the transfer to end
Yes
Set the PER0 register
To use deep sleep mode, stop the clock of the SPI unit and set the reset state
Abort the end of the setting

Note 1: In receive-only mode, the SPI transmission is triggered by reading the value of the SDRI register. If SPI is not aborted in time, there may be a redundant transmission after the last reading of the SDRI.

If you want to avoid the last redundant transmission, you can turn off SPIES after reading out the data for the second time to the penultimate and wait for one SCK cycle. The transfer of SPI will be aborted after the last data transfer is completed.

(2) Processing process

Figure 13-12 Timing diagram of the receiving (DAP=0, CKPmn=0)



13.4.3 Slave send and receive

If bit CKS2-0 of the serial clock select register (SPIC) selects slave mode and bit 6 (TRMD) of the serial operation mode register (SPIM) is 1, the slave transmit/receive mode is entered. When a value is written to the transmit buffer register (SDRO), wait for the clock of the master device and start transmitting/receiving.

（1） Procedure

Figure 13-13 Initial setup steps of slave send/receive

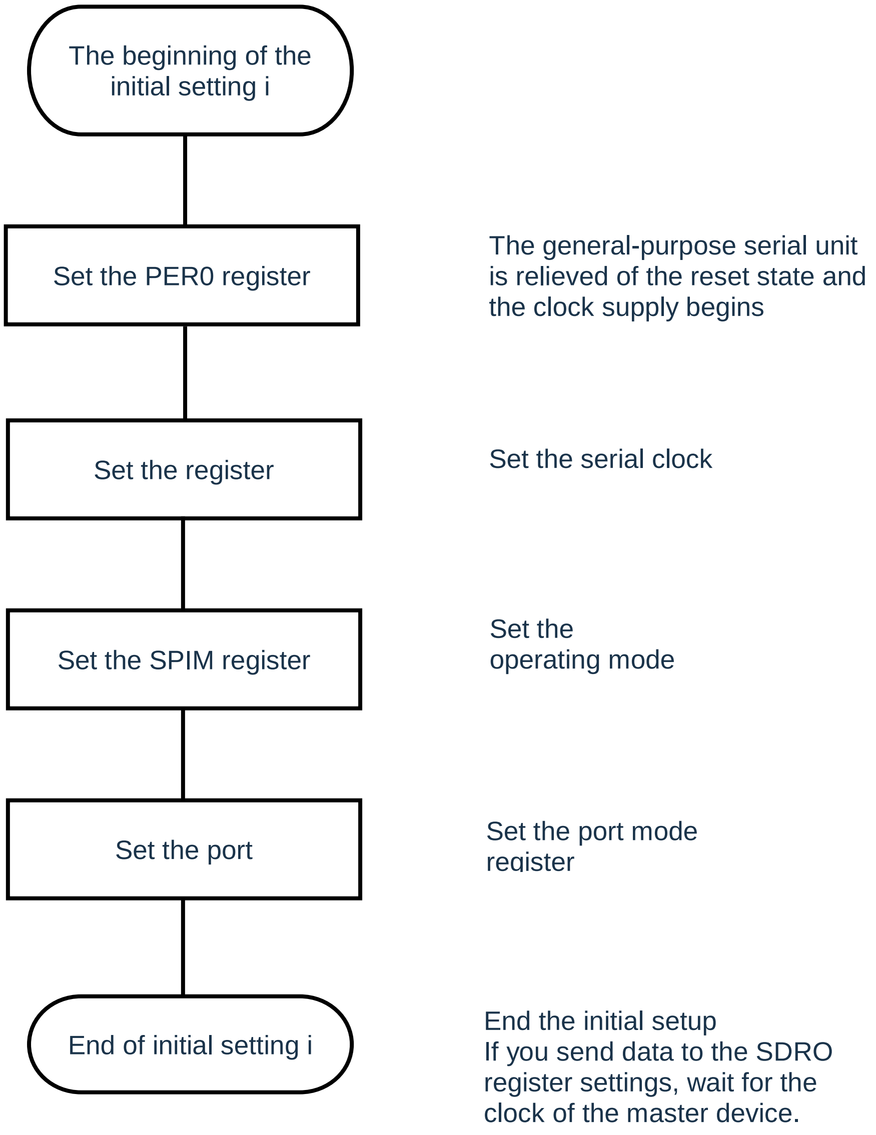
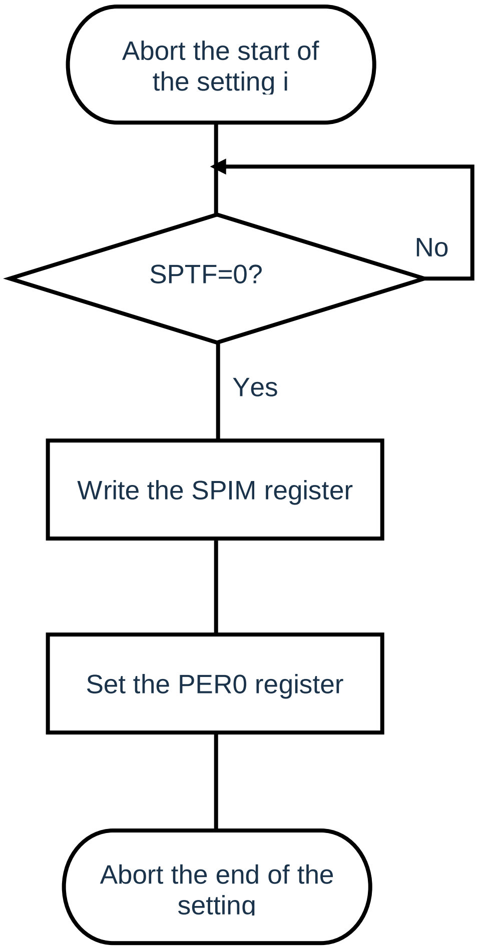

The beginning of the initial setting i
Set the PER0 register
The general-purpose serial unit is relieved of the reset state and the clock supply begins
Set the register
Set the serial clock
Set the SPIM register
Set the operating mode
Set the port
Set the port mode register
End of initial setting i
End the initial setup
If you send data to the SDRO register settings, wait for the clock of the master device.

Figure 13-14 Stop step of slave send/receive


Abort the start of the setting i
SPTF=0?
No
Yes
Write the SPIM register
Set the PER0 register
Abort the end of the setting

If there is data being transferred, wait for the transfer to end

Put the SPIE position "0" to stop the SPI from running

To use deep sleep mode, stop the clock of the SPI unit and set the reset state

(2) Processing

Figure 13-15 Transmit/receive timing diagram (single transmit mode) (INTMD=0, DAP=0, CKPmn=0).

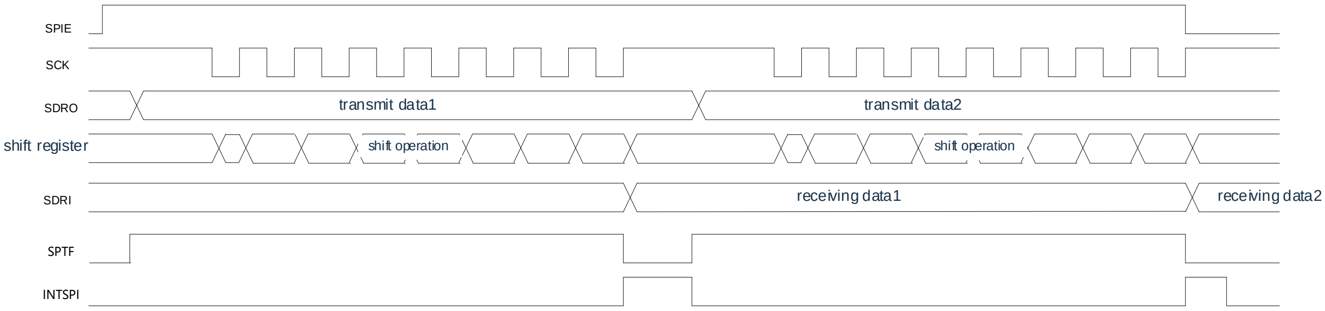
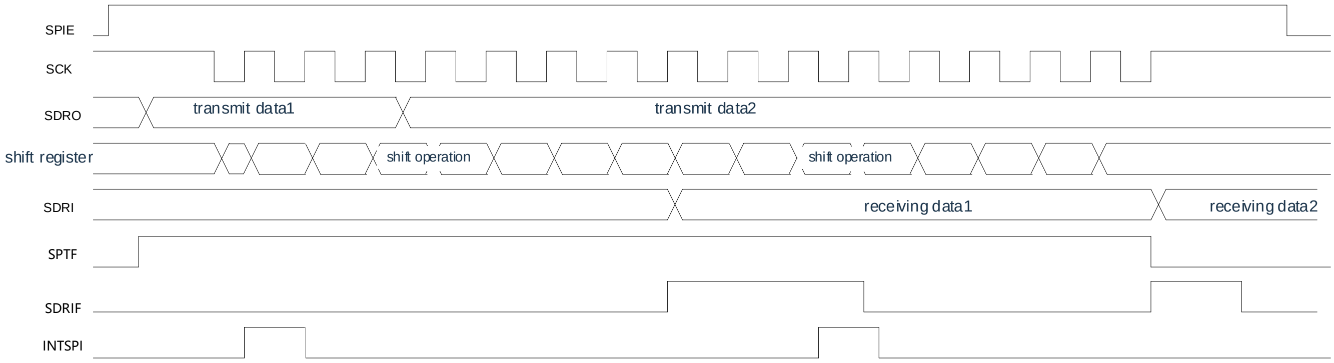


Figure 13-16 Timing diagram of transmit/receive timing (continuous transmit mode) (INTMD=1, DAP=0, CKPmn=0)



13.4.4 Slave reception

If bit CKS2-0 of the serial clock select register (SPIC) selects slave mode and bit 6 (TRMD) of the serial operation mode register (SPIM) is 0, the slave receive mode is entered. When data is read from the receive buffer register (SDRI), wait for the clock of the master device and start receiving.

（1） Procedure

Figure 13-17 Initial setup steps of slave reception

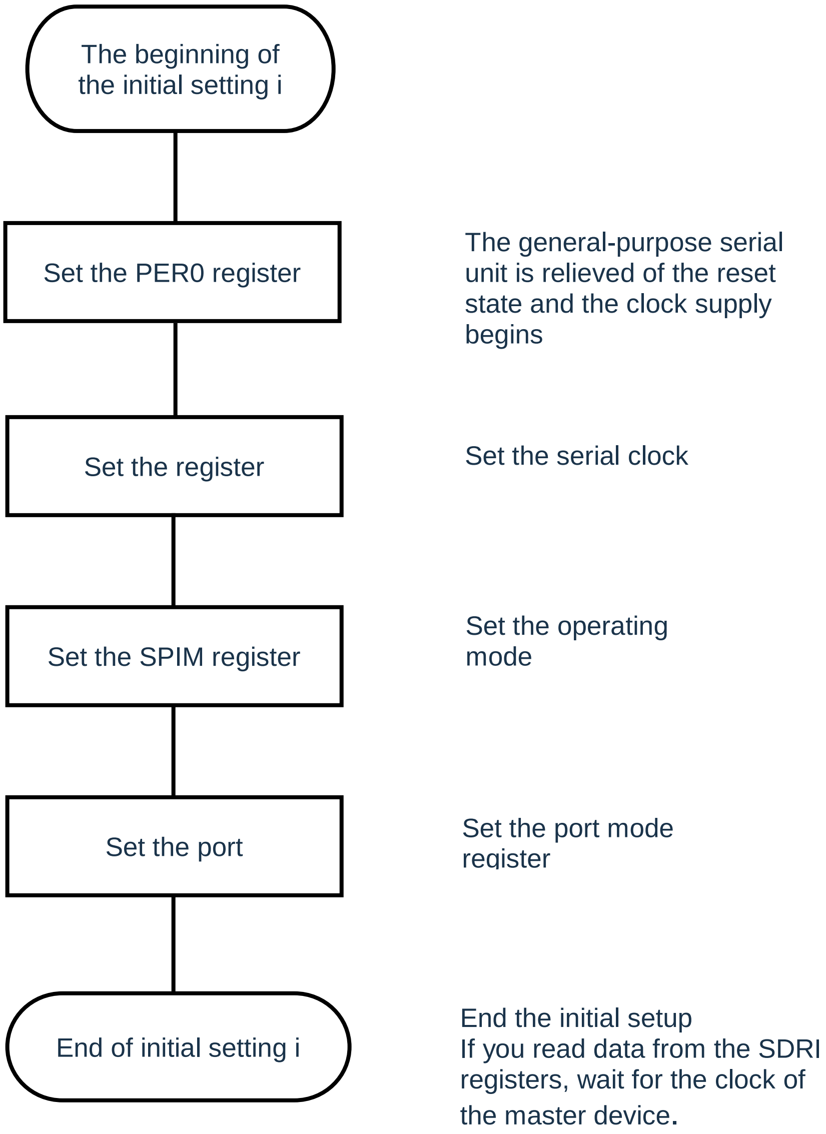
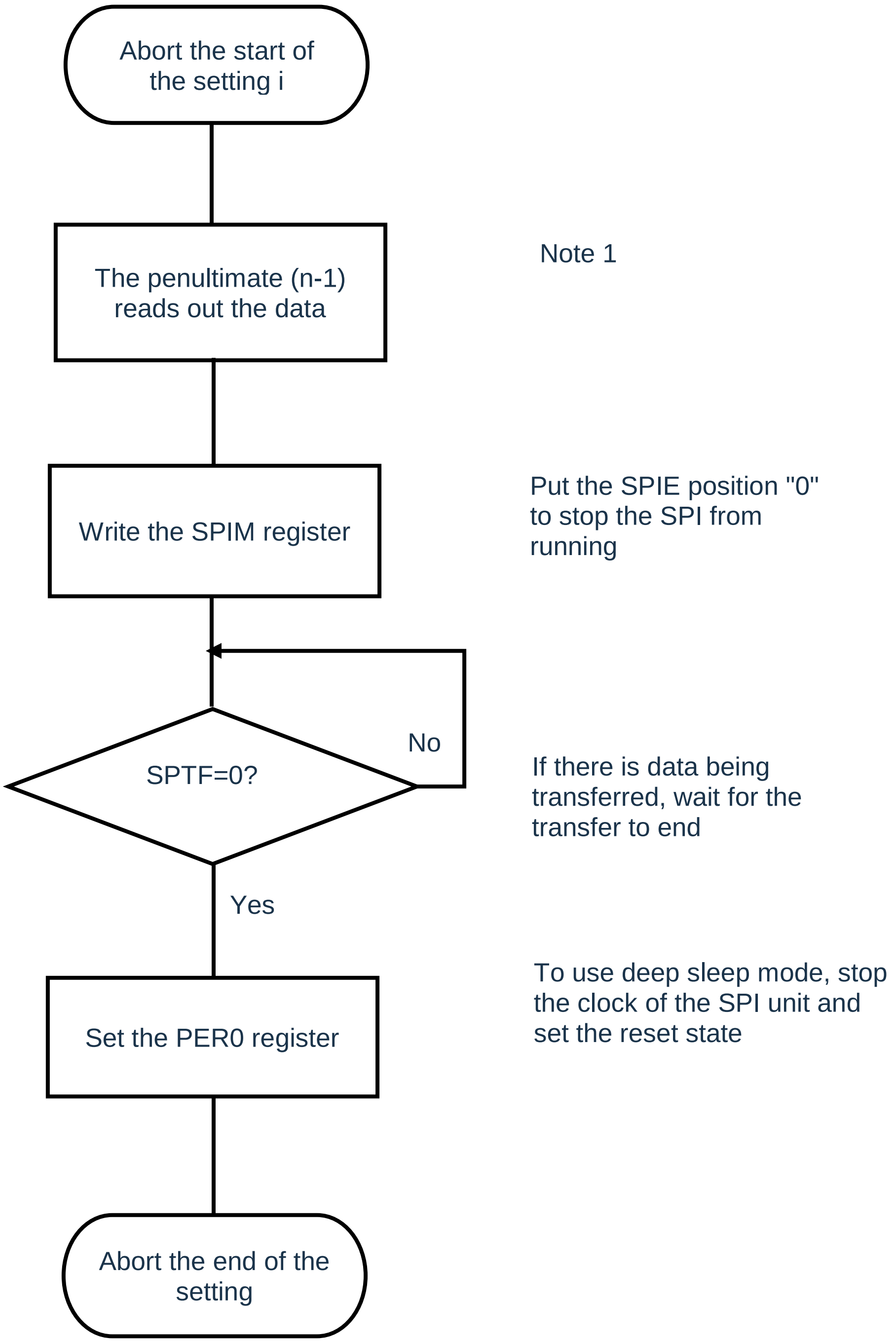

The beginning of the initial setting i
Set the PER0 register
The general-purpose serial unit is relieved of the reset state and the clock supply begins
Set the register
Set the serial clock
Set the SPIM register
Set the operating mode
Set the port
Set the port mode register
End of initial setting i
End the initial setup
If you read data from the SDRI registers, wait for the clock of the master device.

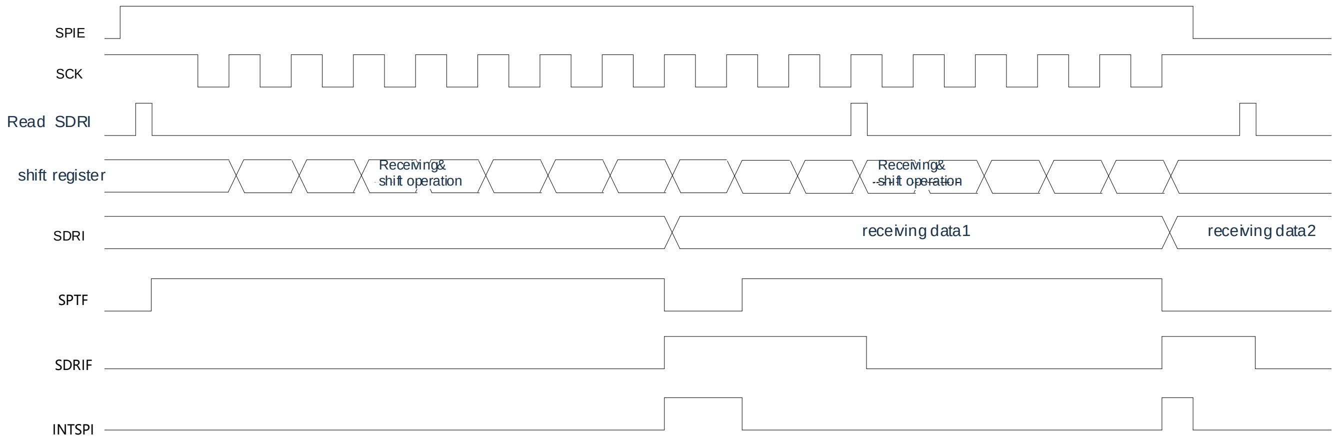
Figure 13-18 Stop step of slave reception


Abort the start of the setting i
The penultimate (n-1) reads out the data
Note 1
Write the SPIM register
Put the SPIE position "0" to stop the SPI from running
SPTF=0?
No
Yes
If there is data being transferred, wait for the transfer to end
Set the PER0 register
To use deep sleep mode, stop the clock of the SPI unit and set the reset state
Abort the end of the setting

Note 1: In receive-only mode, the SPI transmission is triggered by reading the value of the SDRI register. If SPI is not aborted in time, there may be a redundant transmission after the last reading of the SDRI.
If you want to avoid the last redundant transmission, you can turn off SPIE after reading out the data for the second time to the penultimate and wait for one SCK cycle. The transfer of SPI will be aborted after the last data transfer is completed.

(2) Processing

Figures 13-19 Timing diagram of the receiving (DAP=0, CKPmn=0).



Chapter 14 Serial interface IICA

14.1 Function of IICA

The serial interface IICA has the following three modes.

(1) Operation stop mode

This is a mode used when no serial transfer is taking place, which reduces power consumption.

(2) I^{2}C-bus mode (supports multi-master)

This mode transfers 8-bit data to multiple devices via 2 wires of the serial clock (SCLAn) and the serial data bus (SDAAn). In accordance with the I^{2}C-bus format, the master device can generate “start conditions” and “addresses” for the slave devices on the serial data bus, Directions of Transfer, Data, and Stop Conditions. The slave automatically detects the received status and data through the hardware. This feature simplifies the I2C-bus control portion of the application.

Because the SCLAn pins and SDAAn pins of the serial interface IICA are used as open-drain outputs, the serial clock line and serial data bus require pull-up resistors.

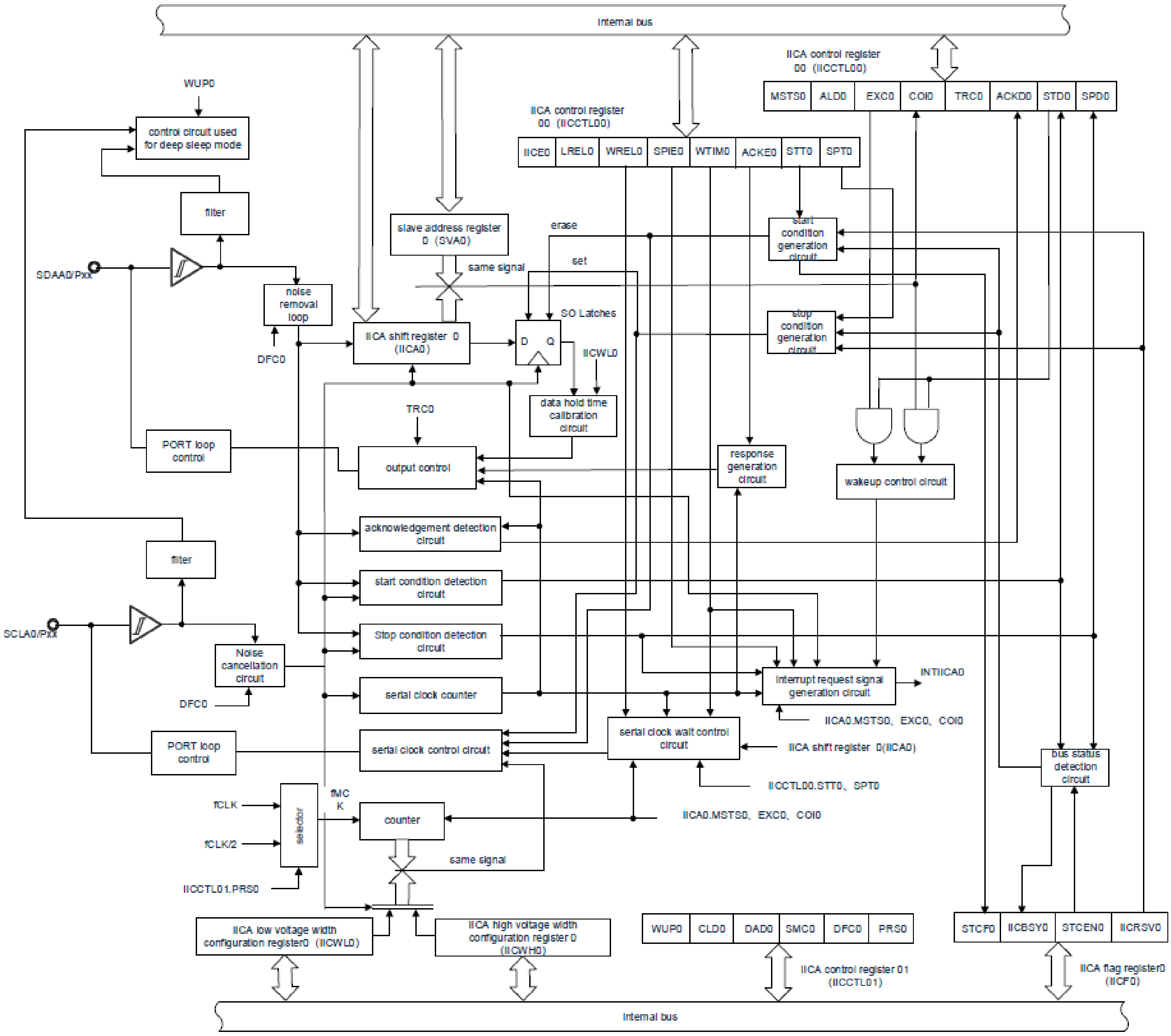
(3) Wake-up mode

In deep sleep mode, deep sleep mode can be released by generating an interrupt request signal (INTIICAn) when the extension code or local station address of the autonomous control device is received. This is set via the WUPn bit of IICA control register n1 (IICCTLn1).

A block diagram of the serial interface IICA is shown in Figure 14-1.

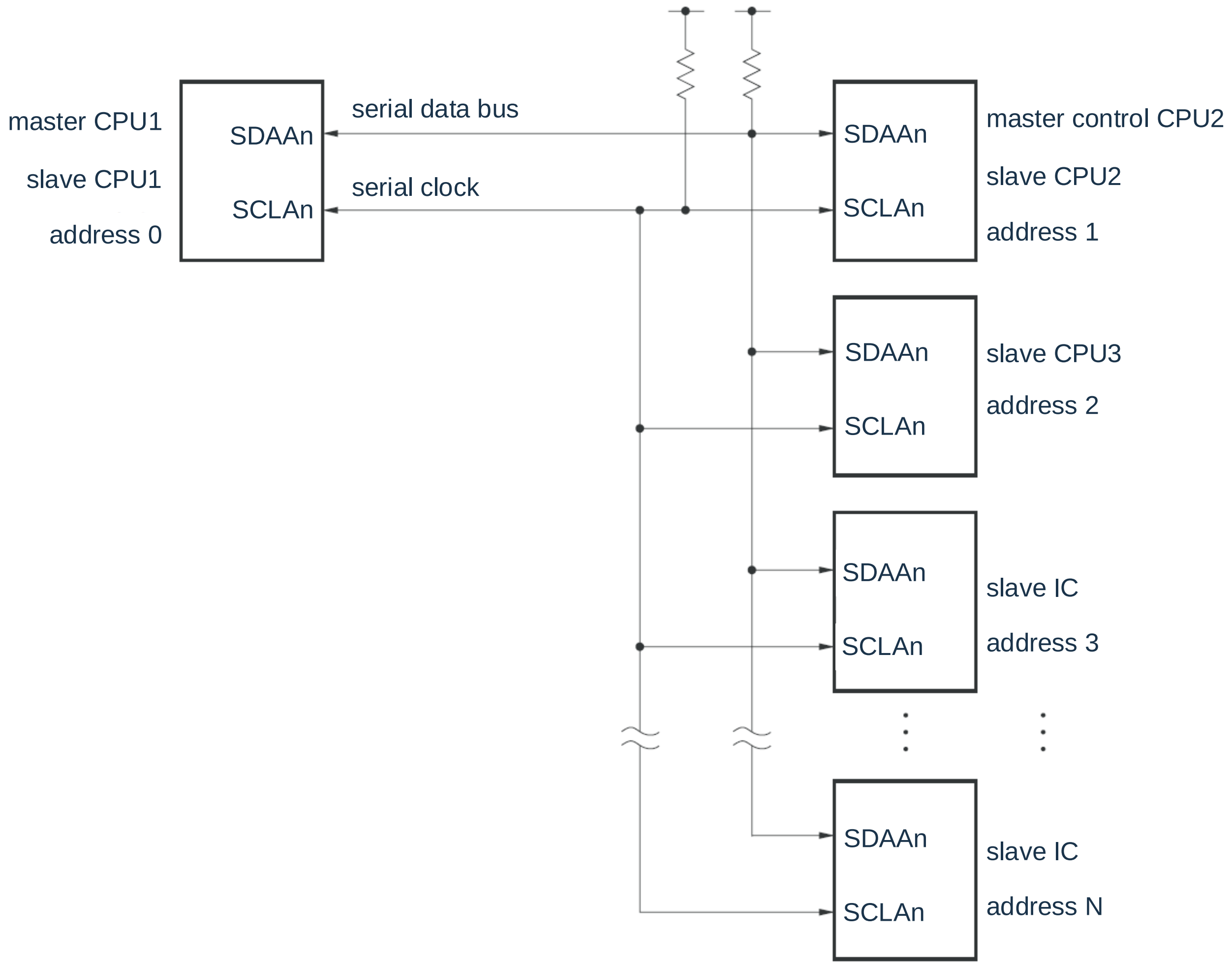
Note n=0

Figure 14-1 Block diagram of the serial interface IICA


Internal bus
IICA control register 00 (IICCTL00)
MSTS0
ALD0
EXC0
COI0
TRC0
ACKD0
STD0
SPD0
WUP0
control circuit used for deep sleep mode
IICA control register 00 (IICCTL00)
IICE0
LREL0
WREL0
SPIE0
WTIM0
ACKE0
STT0
SPT0
filter
slave address register 0 (SVA0)
erase
start condition generation circuit
SDAA0/Pxx
same signal
set
noise removal loop
IICA shift register 0 (IICA0)
D
Q
SO Latches
stop condition generation circuit
DFC0
IICWL0
TRC0
data hold time calibration circuit
PORT loop control
output control
response generation circuit
wakeup control circuit
acknowledgement detection circuit
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 wait control circuit
PORT loop control
serial clock control circuit
IICA shift register 0(IICA0)
bus status detection circuit
IICCTL00.STT0、SPT0
fCLK
fMCK
selector
counter
IICA0.MSTS0、EXC0、COI0
fCLK/2
same signal
IICCTL01.PRS0
IICA low voltage width configuration register0 (IICWL0)
IICA high voltage width configuration register 0 (IICWH0)
WUP0
CLD0
DAD0
SMC0
DFC0
PRS0
STCF0
IICBSY0
STCEN0
IICRSV0
IICA control register 01 (IICCTL01)
IICA flag register0 (IICF0)
Internal bus

An example of the structure of a serial bus is shown in Figure 14-2.

Figure 14-2 Example of a serial bus structure for a I2C bus



Note n=0

14.2 Structure of the serial interface IICA

The serial interface IICA consists of the following hardware.

Table 14-1 Structure of serial Interface IICA

Item	Structure
Register	IICA shift register n (IICAn) Slave address register n (SVAn).
Control registers	Peripheral enable register 0 (PER0). IICA control register n0 (IICCTLn0). IICA status register n (IICSn). IICA flag register n (IICFn). IICA control register n1 (IICCTLn1). IICA low width setting register n (IICWLn) IICA high width setting register n(IICWHn). Port mode register (PMxx). Port mode control register (PMCxx). Port multiplexing function configuration register (PxxCFG).

Note 1. n = 0.

2. This product can multiplex the IICA input/output pin function to any port. When a port is configured as a multiplexed function of the IICA pin, the N-channel open-drain output (V_{DD}/EV_{DD} withstand voltage) mode of the port is guaranteed to open automatically by design, i.e. the POMxx register does not require user settings.

(2) IICA shift register n (IICAn)

IICAn registers are registers that convert 8-bit serial data and 8-bit parallel data to and from the serial clock for transmission and receiving. The actual transmitting and receiving can be controlled by reading and writing IICAn registers.

During the wait, the wait is released by writing the IICAn register and the data transfer begins. The IICAn register is set via an 8-bit memory operation instruction. After the reset signal is generated, the value of this register becomes “00H”.

Figure 14-3 Format of IICAn shift register n (IICAn)

Address: 0x40041B50 After reset: 00HR/W

Symbol	7	6	5	4	3	2	1	0
IICAn								

Note 1 During data transfer, data cannot be written to the IICAn register.

2. IICAn registers can only be read and written while waiting. Access to IICAn registers in a communication state is prohibited except during the waiting period. However, in the case of a master device, the IICAn register can be written once after the communication trigger bit (STTn) is set to “1”.

3. When scheduling communication, data must be written to the IICAn register after detecting an interrupt caused by a stop condition.

Note n=0

(2) Slave address register n (SVAn)

This is the register that holds the 7-bit local station address {A6, A5, A4, A3, A2, A1, A0} when used as a slave.

The SVAn register is set via an 8-bit memory operation command. However, when the STDn bit is “1” (start condition detected), overriding this register is prohibited.

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

Figure 14-4 Format of dependent address register n (SVAn)

Address: 0x40041A34 After reset: 00HR/W

Symbol	7	6	5	4	3	2	1	0
SVAn	A6	A5	A4	A3	A2	A1	A0	0 Note

Note Bit0 is fixed as “0”.

(3) SO latch

The SO latch maintains the output level of the SDAAn pin.

(4) Wake up control circuitry

This 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 or when the extension code is received.

(5) Serial clock counter

During transmit or receive, this counter counts the serial clock of the output or input and checks whether 8-bit data has been transmitted and received.

(6) Interrupt request signal generation circuit

This circuit control generates an interrupt request signal (INTIICAn). An I^{2}C interrupt request is generated by the following two triggers.

- Drop of the 8th or 9th serial clock (set by WTIMn bit).
- Interrupt request due to detection of a stop condition (set via SPIEn bit).

Note WTIMn bit: Bit3 of IICA control register n0 (IICCTLn0).
SPIE nbit : Bit4 of IICA control register n0 (IICCTLn0).

(7) Serial clock control circuitry

In master mode, this circuit generates the output clock from the sample clock to the SCLAn pin.

(8) Serial clock wait control circuitry

This circuit controls the wait timing.

(9) Ack generation circuit, stop condition detection circuit, start condition detection circuit, Ack detection circuit

These circuits generate and detect various states.

(10) Data hold time correction circuit

This circuit generates a data hold time for the serial clock to fall.

(11) Start condition generation circuit

If the STTn bit is “1”, this circuit generates a start condition.

However, in a state where scheduled communication is disabled (IICRSVn bit=1) and the bus is not released (IICBSYnbit=1), the start condition request is ignored and the STCFn bit is “1” .

(12) Stop condition generation circuit

If the SPTn bit is “1”, this circuit generates a stop condition.

(13) Bus status detection circuitry

This circuit detects whether the bus is released by detecting the start and stop conditions. However, the bus state cannot be detected immediately during operation, so the initial state of the bus state detection circuit must be set via the STCENn bit.

Remark 1.STTn bit: bit1 of IICA control register n0 (IICCTLn0).
SPTn bit: bit0 of IICA control register n0 (IICCTLn0).
IICRSVn bit: bit0 of IICA flag register n (IICFn).
IICBSYn bit: Bit6 of IICA flag register n (IICFn).
STCFn bit: bit7 of IICA flag register n (IICFn).
STCENn bit: bit1 of IICA flag register n (IICFn).
2. n=0

14.3 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

14.3.1 Peripheral enable register 0 (PER0)

The PER0 register is a register that sets the clock to be allowed or disallowed to be supplied to each peripheral hardware. Reduce power consumption and noise by stopping clocking unused hardware.

To use the serial interface IICAn, bit4 (IICAEN) must be set to “1”.

The PER0 register is set via an 8-bit memory operation command.

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

Figure 14-5 the peripheral Enable register 0 (PER0)

Address: 40020 420H After reset: 00HR/W

symbol	7	6	5	4	3			0
PER0	RTCEN	IRDAANDN	ADCIN	IICAEN	SCI1IN	SCI0EN	TM41EN	TM40EN

IICAnEN	Provides control of the input clock of the serial interface IICA
0	Stop supplying the input clock. • Cannot write the serial interface IICA using SFR. • The serial interface IICA is in a reset state.
1	Allows the input clock to be provided. • SFR used by the serial interface IICA can be read and written.

Note 1 To set the serial interface IICA, the following registers must first be set in the state where the IICAEN bit is “1”. When the IICAEN bit is “0”, the value of the control register of the serial interface IICA is the initial value, ignoring the write operation (port multiplexing function configuration register (PxxCFG), port mode register (PM xx) and port mode control registers (PMCxx).

- IICA control register n0 (IICCTLn0).
- IICA flag register n (IICFn).
- IICA status register n (IICSn).
- IICA control register n1 (IICCTLn1).
- IICA low level width setting register n (IICWLn).
- IICA high level width setting register n (IICWHn).

Remark n=0

14.3.2 IICA control register n0 (IICCTLn0)

This is a register that allows or stops I^2C operation, sets the wait sequence, and sets other I^2C operations.

The IICCTLn0 register is set via an 8-bit memory operation command. However, the SPIEn, WTIMn, and ACKEn bits must be set when the IICEn bit is “0” or during the wait, and the IICEn must be set. Bits can be set simultaneously when they are set from “0” to “1”.

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

Remark n=0

Figure 14-6 Format of IICA control register n0 (IICCTLn0) (1/4)

Address: 0x40041A30 After reset: 00HR/W

symbol	7	6	5	4	3	2	1	0
IICCTLn0	IICEn	LRELn	WRELn	SPIEn	WTIMn	ACKEn	STTn	SPTn

IICEn	I^2C operation enable
0	Disable operation. Reset[Note 1] to IICA status register n (IICSn) and stop internal operation.
1	Enable operation.
This bit must be “1” in the state where the SCLAn line and the SDAAn line are high.	
Clear condition (IICEn=0).	Set condition (IICEn=1).
• Clear by command. • When reset	• Set by command.

LRELn Note 2,3	Exit of communication
0	Normal operation
1	Exit the current communication and enter standby. Automatically clear “0” after execution. It is used in cases where an extension code that is not related to the local station is received, etc. The SCLAn line and the SDAAn line become high-impedance. The following flags in IICA control register n0 (IICCTLn0) and IICA status register n (IICSn) are cleared “0”: •STTn•SPTn•MSTSn•EXCn•COIn•TRCn•ACKDn•STDn
Becomes standby to exit communication until the following communication participation conditions are met. •Starts as a master device after a stop condition is detected. •Address matching or receiving extension code after the start condition is detected.	
Clear condition (LRELn=0).	Set condition (LRELn=1).
• Automatically clear after execution. • When reset	• Set by command.

WRELn Note 2,3	Waiting for the release
0	Do not release the wait.
1	Release the wait. Automatically clears after the wait is released.
If the WRELn bit (unwait) is set during the 9th clock wait in the transmit state (TRCn=1), the SDAAn line becomes high impedance state (TRCn=0).	
Clear condition (WRELn=0).	Set condition (WRELn=1).
• Automatically clear after execution. • When reset	• Set by command.

Note 1. For IICA shift register n (IICAn), IICA flag register n (IICFn). STCFn bits and IICBSYn bits and CLDn of IICA control register n1 (IICCTLn1). The bits and DADn bits are reset.

2. In the state where the IICEn bit is “0”, the signal for this bit is invalid.

3. The read value of LRELn bits and WRELn bits is always “0”.

Notice If I^2C operation is allowed (IICEn=1) when the SCLAn line is high, the SDAAn line is low and the digital filter is ON (DFCn=1 of the IICCTLn1 register), the start condition is detected immediately. In this case, the LRELn bit must be set to "1" by the bit memory operation instruction continuously after I^2C operation is allowed (IICEn=1).

Remark n=0

Figure 14-6 Format of IICA control register n0 (IICCTLn0) (2/4)

SPIEn[Note1]	Enable or disable interrupt requests generated by stop condition detection
0	Disable
1	Enable
When the WUPn bit of IICA control register n1 (IICCTLn1) is “1”, even if the SPIEn is “1” It also does not produce a stop condition interrupt.	
Clear condition (SPIEn=0).	Set condition (SPIEn=1).
• Clear by command. • When reset	• Set by command.

WTIMn[Note1]	Wait for and interrupt the control of the request
0	An interrupt request signal is generated on the falling edge of the 8th clock. Master device: After 8 clocks are output, set the clock output to low and wait. Slave: After entering 8 clocks, set the clock low and wait for the master device.
1	An interrupt request signal is generated on the falling edge of the 9th clock. Main device: After 9 clocks are output, set the clock output to low and wait. Slave: After entering 9 clocks, set the clock low and wait for the master device.
During address transfer, regardless of the setting of this bit, an interrupt occurs on the falling edge of the 9th clock; After the end of the address transfer, this bit is set Effect. The master device enters the wait state on the 9th clock falling edge during address transmission. The slave device that receives the address of the local station is generating an answer The descending edge of the 9th clock (ACK) enters the waiting state, but the slave device that receives the expansion code enters the waiting state on the 8th clock falling edge.	
Clear condition (WTIMn=0).	Set condition (WTIMn=1).
• Clear by command. • When reset	• Set by command.

ACKEn Note 1,2	Answer control
0	Reply is prohibited.
1	Allow answers. Set the SDAAn line low during the 9th clock.
Clear condition (ACKEn=0).	Set condition (ACKEn=1).
• Clear by command. • When reset	• Set by command.

Note 1 In the state where the IICEn bit is “0”, the signal for this bit is invalid. This bit must be set during this time.

2. When the extension code is not in the address transmission process, the config valueis invalid. When a slave device and the address matches, an answer is generated regardless of the config value.

Remark n=0

Figure 14-6 Format of IICA control register n0 (IICCTLn0) (3/4)

<table>
<tr><td>STTn[Note1, 2]</td><td colspan="2">The trigger of the start condition</td></tr>
<tr><td>0</td><td colspan="2">Start conditions are not generated.</td></tr>
<tr><td>1</td><td colspan="2">When the bus is released (standby, IICBSYn bit is “0”): If this bit is “1”, a start condition is generated (as the start of the master device). While a third party is communicating:
• In cases where the communication reservation function is allowed (IICRSVn=0). Used as a starting condition reservation sign. If this bit is “1”, the start condition is automatically generated after the bus is released.
• In the case where the communication reservation function is prohibited (IICRSVn=1). Even if this bit “1” is removed, the STTn bit is cleared and the STTn clear flag (STCFn) is set to “1” without generating a start condition. Wait Status (Master Device):
A restart condition is generated after the wait is released.</td></tr>
<tr><td colspan="3">Considerations for set timing:
• Master Receive: Disables this bit to "1" during transmission. This bit "1" can only be placed during the waiting period when ACKEn is "0" and notifying the slave that receiving it has completed.
• Master Send: During the reply, the start condition may not be generated normally. This bit “1” must be placed during the waiting period after the 9th clock is output.
• It is forbidden to set “1” at the same time as the trigger of the stop condition (SPTn).
• After placing the STTn to “1”, it is forbidden to put this bit “1” again until the Clear condition is met.</td></tr>
<tr><td colspan="2">Clear condition (STTn=0).</td><td>Set condition (STTn=1).</td></tr>
<tr><td colspan="2">• Set STTn to “1” in a state where communication reservation is prohibited.
• In the event of a failed arbitration
• Master device generates start conditions.
• Clearance due to LRELn bit being “1” (exit communication).
• When the IICEn bit is “0” (stop running).
• When reset</td><td>• Set by command.</td></tr>
</table>

Note 1. In the state where the IICEn bit is “0”, the signal for this bit is invalid.

2. The read value of the STTn bit is always “0”.

Note 1 If bit1 (STTn) is read after setting the data, this bit becomes “0”.

2. IICRSVn: bit0 of the IICA flag register n (IICFn).

STCFn: Bit7 of the IICA flag register n (IICFn).

3.n=0

Figure 14-6 Format of IICA control register n0 (IICCTLn0) (4/4)

<table>
<tr><th>SPTn Note</th><th colspan="2">The trigger of the 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 (as the end of the transfer of the master device).</td></tr>
<tr><td colspan="3">Considerations for setting timing:
• Master Receive: Disables this bit setting to "1" during transmission. This bit can only be set to "1" during the waiting period when ACKEn is at "0" and notifying the slave that receiving it has completed.
• Master Send: During the Ack, the stop condition may not be generated properly. This bit must be set to "1" during the wait period after the 9th clock is output.
• It is forbidden to set “1” at the same time as the trigger of the start condition (STTn).
• SPTn can only be set to "1" in the case of the master device.
• When the WTIMn bit is "0", it must be noted that if the SPTn bit is set to "1" during the wait after 8 clocks of output, the stop condition is generated during the high level of the 9th clock after the release of the wait. The WTIMn bit must be set from "0" to "1" during the wait period after 8 clocks of output and the SPTn bit must be set to "1" during the wait period after the 9th clock of output.
• After setting the SPTn to "1", it is forbidden to set this bit "1" again until the clear condition is met.</td></tr>
<tr><td colspan="2">Clear condition (SPTn=0).</td><td>Set condition (SPTn=1).</td></tr>
<tr><td colspan="2">•When arbitration fails
•Automatically clear when a stop condition is detected.
• Clearance due to LRELn bit being “1” (exit communication).
• When the IICEn bit is “0” (stop running).
• When reset</td><td>• Set by command.</td></tr>
</table>

Note Read value of the SPTn bit is always “0”.

Notice When bit 3 (TRCn) of the IICA status register n (IICSn) is "1" (transmit status) and if bit 5 (WRELn) of the IICCTLn0 register is set to "1" at the 9th clock to release the wait, the SDAAn line is set to high impedance after clearing the TRCn bit (receive state). The wait must be released by writing "1" to the TRCn bit of the IICA shift register n (transmit status).

Remark n=0

14.3.3 IICA status register n (IICSn)

This is the register that represents the I²C state.

The 8-bit memory operation instruction can read the IICSn register only during the STTn bit being "1" and waiting. After the reset signal is generated, the value of this register becomes "00H".

Notice In deep sleep mode, the IICSn register is forbidden to read in the Allow Address Matching Wake-Up Function (WUPn=1) state. In the state where the WUPn bit is "1", it is not related to the INTIICAn interrupt request if the WUPn bit is changed from "1" to "0" (Stop Wake-On Operate), the change in state is not reflected until the next start condition or stop condition is detected. Therefore, when using the wake-up function, interrupts arising from the detection of a stop condition must be allowed (SPIEn=1) and the IICSn register must be read after the interrupt is detected.

Remark		
	STTn	: bit1 of IICA control register n0 (IICCTLn0).
	WUPn	: Bit7 of IICA control register n1 (IICCTLn1).

Figure 14-7 Format of IICA status register n (IICSn) (1/3)

Address: 0x40041B51 After reset: 00H R

symbol	7	6	5	4	3	2	1	0
IICSn	MSTSn	ALDn	EXCn	COIn	TRCn	ACKDn	STDn	SPDn

MSTSn	Confirmation flag for the master status	
0	Slave state or communication standby	
1	Master communication status	
Clear condition (MSTSn=0).		Set condition (MSTSn=1).
•When a stop condition is detected •When the ALDn bit is "1" (arbitration failed). • Clearance due to LRELn bit being "1" (exit communication). • When the IICEn bit changes from "1" to "0" (stop running). • When reset		•When generating a start condition

ALDn	Detection of arbitration failures	
0	Indicates that no arbitration occurred or that arbitration was won.	
1	Indicates that arbitration failed. Clear the MSTSn bit.	
Clear condition (ALDn=0).		Set condition (ALDn=1).
• Automatically clear the IICSn register after reading Note. • 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 performed on a bit other than the IICSn register. Therefore, when using the ALDn bit, the data of the ALDn bit must be read before reading the other bits.

Remark 1. LRELn: Bit6 of the IICA control register n0 (IICCTLn0).
IICEn: Bit7 of the IICA control register n0 (IICCTLn0).
2.n=0

Figure 14-7 Format of IICA status register n (IICSn) (2/3)

EXCn	Receive detection of expansion codes
0	The extension code was not received.
1	The extension code is received.
Clear condition (EXCn=0).	Set condition (EXCn=1).
•When a start condition is detected •When a stop condition is detected • Clearance due to LRELn bit being “1” (exit communication). • When the IICEn bit changes from “1” to “0” (stop running). • When reset	• When the high 4 bits of the received address data is “0000” or “1111” (Set on the rising edge of the 8th clock).

COIn	Detection of address matches
0	Different address.
1	Same address.
Clear condition (COIn=0).	Set condition (COIn=1).
•When a start condition is detected •When a stop condition is detected • Clearance due to LRELn bit being “1” (exit communication). • When the IICEn bit changes from “1” to “0” (stop running). • When reset	• When receiving address and local station address (slave address register n (SVAn)) is the same (Set on the rising edge of the 8th clock).

TRCn	Send/receive status detection
0	It is in the receive state (except for the send state). Set the SDAAn line to high impedance.
1	It is in the sending state. Set to output the value of the SOn latch to the SDAAn line (effective after the falling edge of the 9th clock of byte 1).
Clear condition (TRCn=0).	Set condition (TRCn=1).
< Master and slave devices> •When a stop condition is detected • Clearance due to LRELn bit being “1” (exit communication). • When the IICEn bit changes from “1” to “0” (stop running). •Clear due to WRELn bit being “1” (release wait). •When the ALDn bit changes from “0” to “1” (arbitration failed). • When reset • Cases of non-participation in communication (MSTSn, EXCn, COIn=0). < Master device> • When the LSB (Transfer Direction Indicator Bit) of the first byte outputs “1” < Slave device> •When a start condition is detected • When the LSB (transmission direction indication bit) of the 1st byte is entered as "0"	< Master device> • When generating a start condition • When the LSB (transmission direction indication bit) output of the first byte (address transmission) is “0” (master transmission). < Slave device> • When the LSB (transmission direction indication bit) of the first byte (address transmission) of the master device is "1" (slave transmission).

Note If bit 3 (TRCn) of IICA status register n (IICSn) is "1" (transmit status), if bit 5 (WRELn) of IICA control register n0 (IICCTLn0) is set to "1" at the 9th clock " to release the wait, the SDAAn line is set to high impedance after clearing the TRCn bit (receive state). The wait must be released by writing "1" to the TRCn bit of the IICA shift register n (transmit state).

Remark 1. LRELn: Bit6 of the IICA control register n0 (IICCTLn0).

IICEn: Bit7 of the IICA control register n0 (IICCTLn0).

2. n=0

Figure 14-7 Format of IICA status register n (IICSn) (3/3)

ACKDn	Detection of ACK	
0	No reply detected.	
1	An answer is detected.	
Clear condition (ACKDn=0).		Set condition (ACKDn=1).
• When a stop condition is detected • When the 1st clock of the next byte goes up • Clearance due to LRELn bit being “1” (exit communication). • When the IICEn bit changes from “1” to “0” (stop running). • When reset		• Set the SDAAn line low on the 9th clock rising edge of the SCLAn line

STDn	Start detection of conditions	
0	No start condition detected.	
1	A start condition was detected, indicating that it was during address transfer.	
Clear condition (STDn=0).		Set condition (STDn=1).
•When a stop condition is detected • When the 1st clock of the next byte after the address is transferred • Clearance due to LRELn bit being “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 conditions	
0	No stop condition detected.	
1	A stop condition is detected, the master device ends communication and the bus is released.	
Clear condition (SPDn=0).		Set condition (SPDn=1).
• After set this bit, when the 1st clock of the address transfer byte after detecting the start condition is detected • 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 of the IICA control register n0 (IICCTLn0).

IICEn: Bit7 of the IICA control register n0 (IICCTLn0).

2.n=0

14.3.4 IICA flag register n (IICFn)

This is the register that sets the I2C operating mode and represents the status of the I2C-bus.

The IICFn register is set via an 8-bit memory operation command. However, only the STTn clear flag (STCFn) and the I2C-bus status flag (IICBSYn) can be read.

The communication reservation function is allowed or disallowed by the IICRSVn bit setting, and the initial value of the IICBSYn bit is set by the STCENn bit. Only at bit7 (IICEn)=0) can only write IICRSVn bits and STCENn bits. Only IICFn registers can be read after they are allowed to operate. After the reset signal is generated, the value of this register becomes “00H”.

Figure 14-8 Format of IICA flag register n(IICFn)

Address: 0x40041B52 After reset: 00HR/W Note

symbol	7	6	5432				1	0
IICFn	STCFn	IICBSYn	0	0	0	0	STCENn	IICRSVn

STCFn	STTn clears the flag
0	Release start conditions.
1	The STTn flag could not be cleared while the start condition could be issued.
Clear condition (STCFn=0).	Set condition (STCFn=1).
• Clearance due to STTn bit being “1” • When the IICEn bit is “0” (stop running). • When reset	• When the STTn bit is cleared to “0” when the start condition cannot be issued when the communication reservation is disabled (IICRSVn=1) state cannot be issued

IICBSYn	I^2C-bus status flags
0	The bus release state (the initial state of communication at STCENn=1).
1	Bus communication state (initial state of communication at STCENn=0).
Clear condition (IICBSYn=0).	Set condition (IICBSYn=1).
•When a stop condition is detected • When the IICEn bit is “0” (stop running). • When reset	•When a start condition is detected • Set the IICEn bit when the STCENn bit is “0”

STCENn	Initial start allows triggering
0	After the allow run (IICEn=1), the start condition is allowed to be generated by detecting the stop condition.
1	After the allow run (IICEn=1), the start condition is allowed to be generated without detecting the stop condition.
Clear condition (STCENn=0).	Set condition (STCENn=1).
• Clear by command. •When a start condition is detected • When reset	• Set by command.

IICRSVn	Communication reservation function disable bit
0	Allow communication reservation.
1	Communication reservation is prohibited.
Clear condition (IICRSVn=0).	Set condition (IICRSVn=1).
• Clear by command. • When reset	• Set by command.

Note Bit6 and bit7 are read-only bits.

Notice 1 The STCENn bit can only be written when it is stopped (IICEn=0).

2. If the STCENn bit is “1”, the bus is considered to be a release state (IICBSYn=0) regardless of the actual bus state, so in order to avoid the first start condition (STTn=) in the release 1) When destroying other communications, it is necessary to confirm that there is no third party that is communicating.

3. IICRSVn can only be written when it is stopped (IICEn=0).

Remark 1. STTn: Bit1 of IICA control register n0 (IICCTLn0).

2. IICEn: bit7 of the IICA control register n0 (IICCTLn0).

14.3.5 IICA control register n1 (IICCTLn1)

This is a register used to set the I^{2}C operating mode and detect the status of the SCLAn pin and the SDAAn pin.

The IICCTLn1 register is set via an 8-bit memory operation command. However, only CLDn bits and DADn bits can be read.

In addition to the WUPn bit, bit7 must be disabled for I^{2}C to operate (IICA control register n0 (IICCTLn0).(IICEn)=0) when setting the IICCTLn1 register.

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

Figure 14-9 Format of IICA control register n1 (IICCTLn1) (1/2)

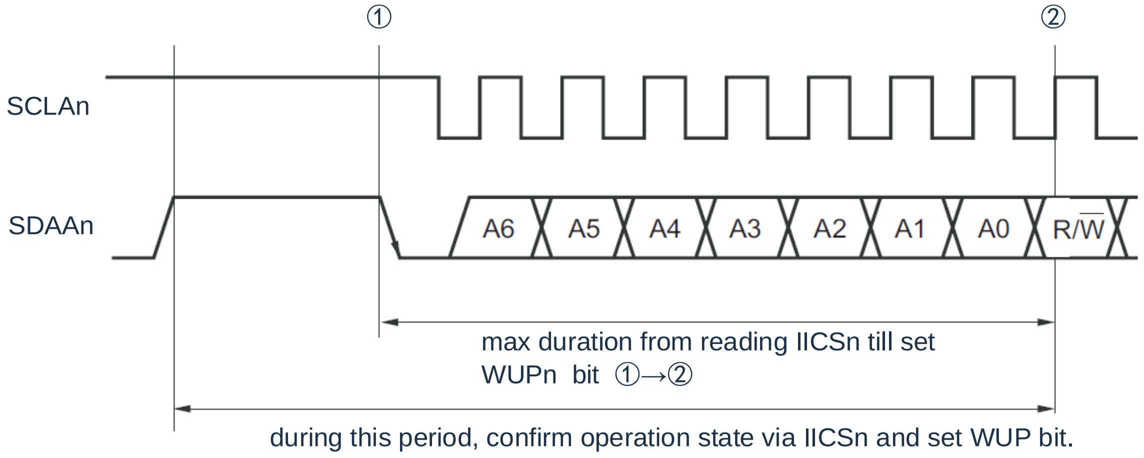
Address: 0x40041A31 After reset: 00HR/W Note 1

symbol	7	6	5	4	3	2	1	0
IICCTLn1	WUPn	0	CLDn	DADn	SMCn	DFCn	0	PRSn

<table>
<tr><td>WUPn</td><td colspan="2">Address matching control of wake-up</td></tr>
<tr><td>0</td><td colspan="2">In deep sleep mode, stop the operation of the address matching wake-up function.</td></tr>
<tr><td>1</td><td colspan="2">In deep sleep mode, address matching is allowed to operate for wake-up functions.</td></tr>
<tr><td colspan="3">To transfer to deep sleep mode by setting the WUPn bit to "1", at least three F_{MCK} clocks must pass after setting the WUPn bit to "1", and then the deep sleep instruction must be executed (refer to "Figure 14-28 Flow when setting the WUPn bit to "1"). The WUPn bit must be cleared to "0" after the address is matched or the extension code is received. It is possible to participate in subsequent communication by clearing the WUPn bit to "0" (it is necessary to release the wait and write send data after clearing the WUPn bit to "0").
In the state of WUPn bit "1", the interrupt timing when the address is matched or the extension code is received is the same as the interrupt timing when WUPn bit is "0".
(The delay difference of sampling error is generated according to the clock). In addition, when the WUPn bit is "1", even if the SPIEn bit is set to "1", no stop condition interrupt is generated.</td></tr>
<tr><td colspan="2">Clear condition (WUPn=0).</td><td>Set condition (WUPn=1).</td></tr>
<tr><td colspan="2">•Clear by command (after the address matches or the extension code is received).</td><td>• Set by instruction (MSTSn=0, EXCn=0, COIn=0, and STDn=0 (do not participate in communication)) Note 2.</td></tr>
</table>

Note 1. Bit4 and bit5 are read-only bits.

2. During the period shown below, it is necessary to confirm the status of the IICA status register n (IICSn) and set it up.


①
②
SCLAn
SDAAn
A6
A5
A4
A3
A2
A1
A0
$R/\overline{W}$
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

Figure 14-9 Format of IICA control register n1 (IICCTLn1) (2/2)

CLDn	Level detection of the SCLAn pin (valid only when the IICEn bit is “1”).
0	A low SCLAn pin is detected.
1	A high SCLAn pin is detected.

Clear condition (CLDn=0).	Set condition (CLDn=1).
• When the SCLAn pin is low • When the IICEn bit is “0” (stop running). • When reset	• When the SCLAn pin is high

DADn	Level detection of the SDAAn pin (valid only when the IICEn bit is “1”).
0	A low SDAAn pin is detected.
1	An SDAAn pin high is detected.

Clear condition (DADn=0).	Set condition (DADn=1).
• When the SDAAn pin is low • When the IICEn bit is “0” (stop running). • When reset	• When the SDAAn pin is high

SMCn	Switching of operating modes
0	Operates in standard mode (maximum transfer rate: 100kbps).
1	Operates in Fast Mode (Max Transfer Rate: 400kbps) or Enhanced Fast Mode (Max Transfer Rate: 1Mbps).

DFCn	Operation control of digital filters
0	Digital filter OFF
1	Digital filter ON
Digital filters must be used in fast mode or enhanced fast mode. Digital filters are used to remove noise. Whether the DFCn bit is “1” or clear “0”, the transmit clock remains unchanged.	

PRSn	Operate the clock (f_{MCK}) control
0	Select f_{CLK}(1MHz≤f_{CLK}≤20MHz).
1	Select f_{CLK}/2(20MHz<f_{CLK}).

Note 1.The maximum operating frequency of the IICA operating clock (f_{MCK}) is 20MHz (Max.). IICA control register n1 (IICCTLn1) must only be used when the f_{CLK} exceeds 20MHz bit0 (PRSn) is set to “1”.

2. In the case of setting the transmission clock, it is necessary to pay attention to the minimum operating frequency of f_{CLK}. The minimum operating frequency of the f_{CLK} for the serial interface IICA depends on the operating mode.

Fast mode: f_{CLK} = 3.5MHz (Min.)
Enhanced Fast Mode: f_{CLK} = 10MHz (Min.)
Standard mode: f_{CLK} = 1MHz (Min.)

Note 1. IICEn: IICA controls bit7 of register n0 (IICCTLn0).

2. n=0

14.3.6 IICA low level width setting register n (IICWLn)

This register controls the SCLAn pin signal low width (tLOW) and the SDAAn pin signal of the serial interface IICA output.

The IICWLn register is set via an 8-bit memory operation command.

It is necessary to run at bit7 (IICEn) at I^{2}C (IICCn) running (IICA control register n0 (IICCTLn0)) = 0) when the IICWLn register is set. After the reset signal is generated, the value of this register changes to “FFH”.

For information on how to set the IICWLn registers, refer to “14.4.2Setting the transmit clock via IICWLn register and IICWHn register”.

The data retention time is 1/4 of the time set by IICWLn.

Figure 14-10 Format of IICA low-level width setting register n (IICWLn)

Address: 0x40041A32 After reset: FFHR/W

Symbol	7	6	5	4	3	2	1	0
IICWLn								

14.3.7 IICA high level width setting register n (IICWHn)

This register controls the SCLAn pin signal high width and SDAAn pin signal of the serial interface IICA output. The IICWHn register is set via an 8-bit memory operation command.

The IICWHn register must be set when I2C operation is disabled (bit7(IICEn)=0 of IICA control register n0(IICCTLn0)). After a reset signal is generated, the value of this register becomes "FFH".

Figure 14-11 Format of high level width setting register n(IICWHn)

Address: 0x40041A33 After reset: FFHR/W

Symbol	7	6	5	4	3	2	1	0
IICWHn								

Remark 1. For the setting method of the master controller transmission clock, please refer to 14.4.2(1); For the method of setting the slave IICWLn register and the IICWHn register, refer to14.4.2).

2.n=0

14.3.8 Registers controlling the IICA pin port function

This product can multiplex IICA pin functions to any port except RESETB.

The SCALn pin and the SDAAn pin can be configured to both ports by setting the port multiplexing function configuration registers (SCLA0PCFG and SDAA0PCFG).

Place the port mode control register (PMCxx) bits corresponding to both ports and the port mode register (PMxx) at “0”.

When these two ports are configured for multiplexing of the IICA pins, the N-channel open-drain output (VDD/EVDD withstand voltage) mode of the port is guaranteed to open automatically by design, i.e. the POMxx register does not require user settings.

For more information on how to set up, see “Chapter 2 Pin Functions”

14.4 Function of I^{2}C-bus mode

14.4.1 Pin structure

The structure of the serial clock pin (SCLAn) and the serial data bus pin (SDAAn) is as follows.

(1) SCLAn...... input/output pin of the serial clock

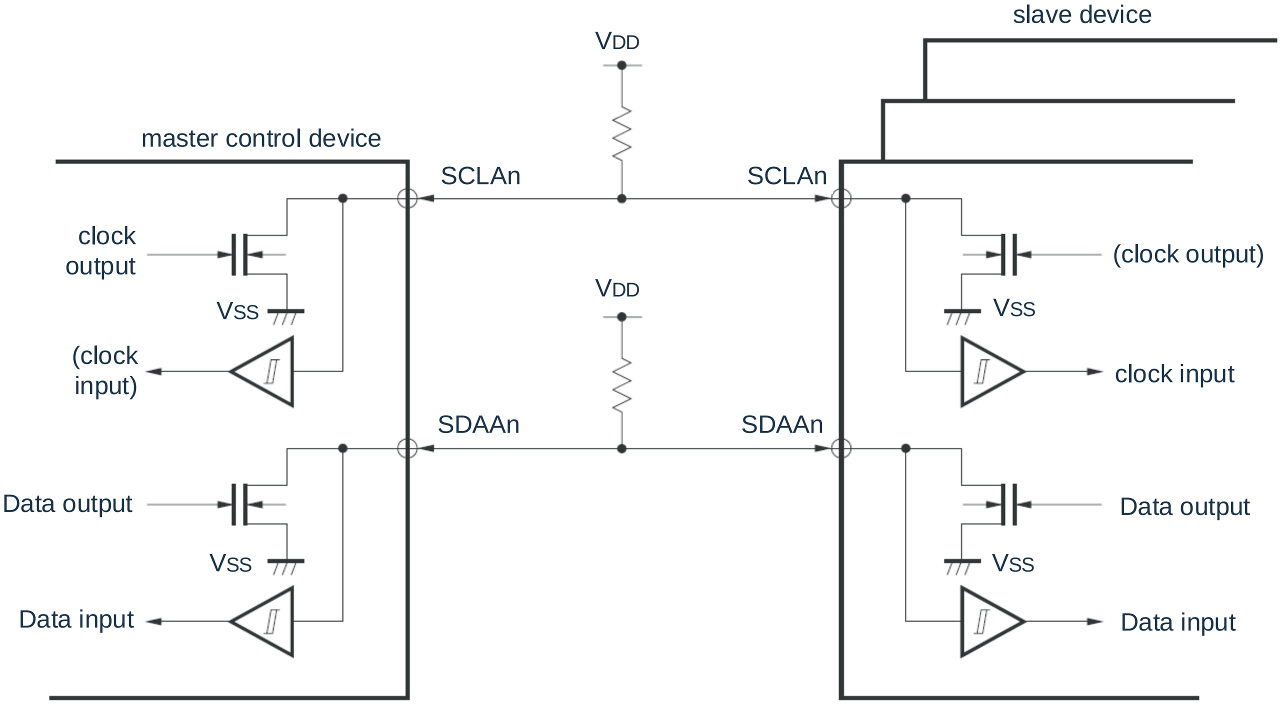
The outputs of the master and slave devices are N-channel open-drain outputs, and the inputs are Schmitt inputs.

(2) SDAAn...... input/output multiplexing pin for serial data

The outputs of the master and slave devices are N-channel open-drain outputs, and the inputs are Schmitt inputs.

Because the outputs of the serial clock line and serial data bus are N-channel open-drain outputs, an external pull-up resistor is required.

Figure 14-12 Pin structure diagram



Remark n=0

14.4.2 Setting the transmit clock via IICWLn register and IICWHn register

(1) The setting method of the master controller transmitting the clock

$$\text{Transmit clock} = \frac{f_{MCK}}{\text{IICWL} + \text{IICWH} + f_{MCK}(t_R + t_F)}$$

At this point, the best config values for the IICWLn register and the IICWHn register are as follows: (All fractional parts of the config valueare rounded)

- Quick mode

$$\text{IICWLn} = \frac{0.52}{\text{transmit clock}} \times f_{MCK}$$

$$\text{IICWHn} = (\frac{0.48}{\text{transmit clock}} - t_R - t_F) \times f_{MCK}$$

- Standard mode

$$\text{IICWLn} = \frac{0.47}{\text{transmit clock}} \times f_{MCK}$$

$$\text{IICWHn} = (\frac{0.53}{\text{transmit clock}} - t_R - t_F) \times f_{MCK}$$

- Enhanced quick mode

$$\text{IICWLn} = \frac{0.50}{\text{transmit clock}} \times f_{MCK}$$

$$\text{IICWHn} = (\frac{0.50}{\text{transmit clock}} - t_R - t_F) \times f_{MCK}$$

(2) Setting method for secondary IICWLn registers and IICWHn registers
(All fractional parts of the config valueare rounded)

- Quick mode

IICWLn=1.3us x f_{MCK}

IICWHn=(1.2us–t_R–t_F) x f_{MCK}

- Standard mode

IICWLn=4.7us x f_{MCK}

IICWHn=(5.3us–t_R–t_F) x f_{MCK}

- Enhanced quick mode

IICWLn=0.50us x f_{MCK}

IICWHn=(0.50us–t_R–t_F) x f_{MCK}

Note 1. The maximum operating frequency of the IICA operating clock (f_{MCK}) is 20MHz (Max.). IICA control register n1 (IICCTLn1) must only be used when the f_{CLK} exceeds 20MHz bit0 (PRSn) is set to “1”.

2. In the case of setting the transmission clock, it is necessary to pay attention to the minimum operating frequency of f_{CLK}. The minimum operating frequency of the f_{CLK} for the serial interface IICA depends on the operating mode.

Fast mode: f_{CLK} = 3.5MHz (Min.)
Enhanced Fast Mode: f_{CLK} = 10MHz (Min.)
Standard mode: f_{CLK} = 1MHz (Min.)

Remark 1. Because the rise time (t_R) and fall time (t_F) of the SDAAn signal and the SCLAn signal vary depending on the pull-up resistance and the wiring capacitance, they must be calculated separately.

2. IICWLn: IICA low level width n setting register
IICWHn: IICA high level width setting register n
t_F: Drop time for SDAAn signal and SCLAn signal
t_R: Rise time of SDAAn signal and SCLAn signal
f_{MCK}: IICA operation at the clock frequency

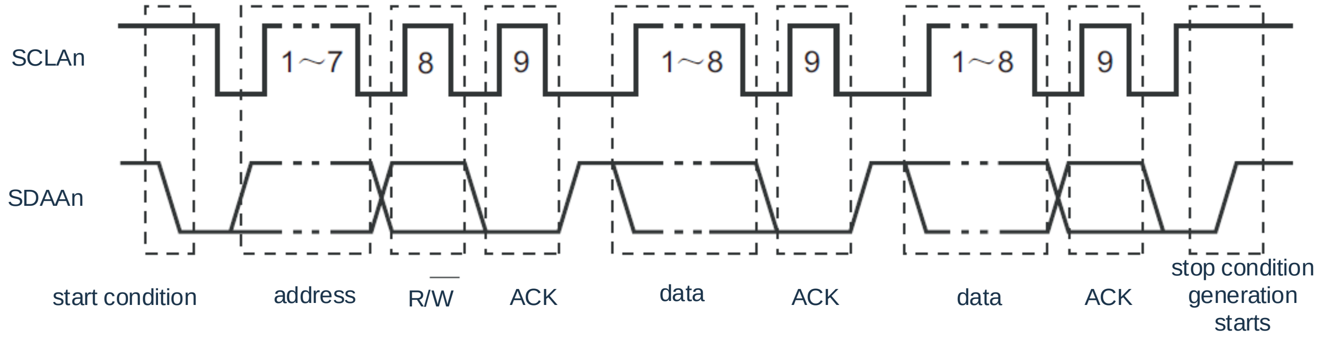
3. n=0

14.5 Definition and control method of I^{2}C-bus

The following describes the serial data communication format and signals used by the I^{2}C-bus.

Each transmission timing of "start condition", "address", "data" and "stop condition" generated on the serial data bus of the I^{2}C bus is shown in the figure below.

Figure 14-13 Serial data transfer timing for the I2C bus



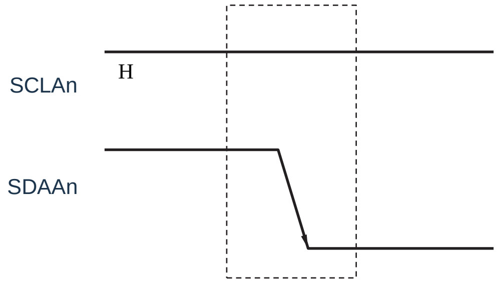
The master device generates start conditions, slave addresses, and stop conditions.

Both the master and slave devices generate a response (ACK) (in general, the receiver outputs 8 bits of data). The master device continuously outputs the serial clock (SCLAn). However, the slave can extend the low level of the SCLAn pin and plug in the wait.

14.5.1 Start conditions

When the SCLAn pin is high, a start condition is generated if the SDAAn pin changes from high to low. The start condition for the SCLAn pin and the SDAAn pin is the signal generated when the master device starts serial transmission to the slave. When used as a slave, the start bar is detected.

Figure 14-14 Start Conditions



If bit1 (STTn) of the IICA control register n0 (IICCTLn0) is set to "1" when a stop condition (SPDn: bit0=1 of the IICA status register n (IICSn)) is detected, the start condition is output. If the start condition is detected, bit1 (STDn) of the IICSn register is set to "1".

Remark n=0

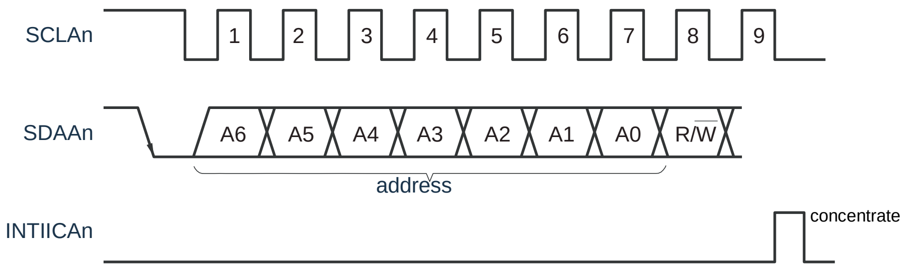
14.5.2 Address

The next 7 bits of data for the start condition are defined as addresses.

The address is the 7-bit data output by the master device in order to select a specific slave device from multiple slaves connected to the bus. Therefore, the slave devices on the bus need to set a completely different address.

The slave detects the start condition through the hardware and checks whether the 7-bit data is the same as the contents of the slave address register n (SVAn). At this point, if the 7-bit data and the SVAn register have the same value, the slave is selected to communicate with the master before it generates a start condition or a stop condition.

Figure 14-15 Address



Note If data other than the local station address or extension code is received at slave runtime, INTIICAn is not generated.

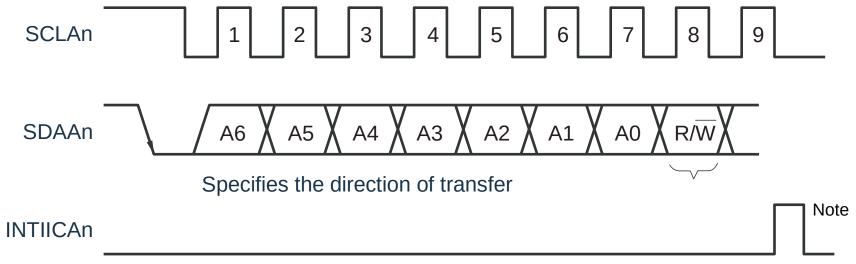
If the 8-bit data consisting of the slave address and the transfer direction described in "14.5.3 Designation of transmission direction" is written to the IICA shift register n (IICAn), the address is output. The received address is written to the IICAn register. The slave address is assigned a high 7 bits in the IICAn register.

14.5.3 Designation of transmission direction

The master device sends 1 bit of data specified in the direction of transmission after the 7-bit address.

When this transmission direction is specified as “0”, it means that the master device sends data to the slave device; When this transmission direction is specified as “1”, it means that the master device receives data from the slave device.

Figure 14-16 Direction of transmission



Note If data other than the local station address or extension code is received during the slave operation, no INTIICAn is generated.

Remark n=0

14.5.4 Acknowledge (ACK)

The serial data status of the sender and receiver can be confirmed by ACK. The receiver returns an answer each time it receives 8 bits of data.

Usually, the sender receives a reply after sending 8-bit data. When the receiver returns an answer, it considers that it has received normally and continues processing. bit2 (ACKDn) that can pass through IICA status register n (IICSn). Confirm the detection of the response. When the master device receives the last data in the received state, a stop condition is generated without returning a reply. When the slave device does not return an answer after receiving data, the master device outputs a stop condition or a restart condition and aborts the transmission. The reason for not returning an answer is as follows:

(1) There is no normal reception.

(2) The receipt of the last data has ended.

(3) There is no receiver specified at the address.

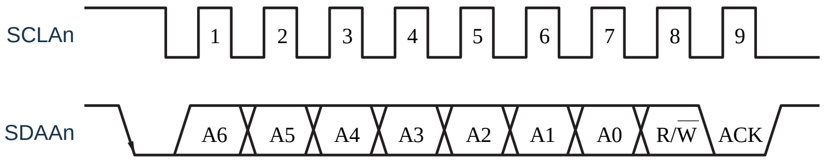
The receiver sets the SDAAn line low on the 9th clock to generate a response (normal reception).

By setting the bit2 (ACKEn) of the IICA control register n0 (IICCTLn0) to "1", it becomes a state that automatically generates a response. Sets bit3 (TRCn) of the IICSn register by the 8th bit of data that follows from the 7-bit address information. In the case of receiving (TRCn=0), it is usually necessary to set the ACKEn bit to "1".

When the data cannot be received during the slave receive operation (TRCn=0) or the next data is not required, the ACKEn bit must be cleared to “0” to inform the master controller that the data cannot be received.

When the next data is not required during the master receive operation (TRCn=0), in order not to generate a reply, the ACKEn bit must be cleared to “0” to notify the end of the slave sender's data (stop sending).

Figure 14-17 ACK



When the address of the local station is received, it has nothing to do with the value of the ACKEn bit, and a reply is automatically generated; When an address from a non-local station is received, no answer (NACK) is generated.

When the extension code is received, an answer is generated by setting the ACKEn bit to “1” in advance. The method of generating an answer when receiving data varies depending on the setting of the waiting time series, as shown below.

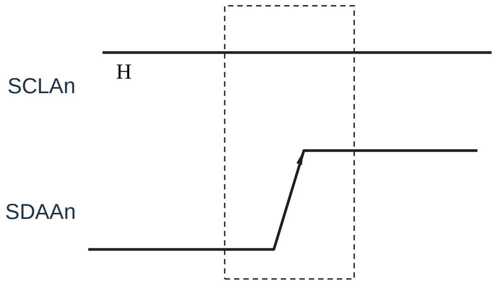
- When 8 clocks of wait are selected (bit3 (WTIMn) = 0 in the IICCTLn0 register): an answer is generated synchronously with the 8th clock falling edge of the SCLAn pin by setting the ACKEn bit to "1" before releasing the wait.
- When selecting 9 clocks of waiting (bit3 (WTIMn) = 1 for the IICCTLn0 register): by placing beforehand ACKEn position “1”, generates an answer.

Remark n=0

14.5.5 Stop Conditions

When the SCLAn pin is high, a stop condition is generated if the SDAAn pin changes from low to high. The stop condition is the signal generated when the master device ends the serial transmission to the slave device. When used as a slave, a stop condition can be detected.

Figure 14-18 Stop conditions



If bit0 (SPTn) of IICA control register n0 (IICCTLn0) is set to “1”, a stop condition is generated. If a stop condition is detected, set bit0 (SPDn) of IICA status register n (IICSn) to “1”, and INTIICAn is generated when bit4 (SPIEn) of the IICCTLn0 register is “1”.

Note n=0

14.5.6 Wait

Wait to notify the other party that the master or slave device is preparing for the send/receive of data (waiting status).

By setting the SCLAn pin low, the other party is notified that it is waiting. If both the master and slave wait states are released, the next transfer can begin.

Figure 14-19 Wait (1/2)

(1) The case where the master device waits for 9 clocks and the slave device waits for 8 clocks (Master device: send, slave device: receive, ACKEn=1).

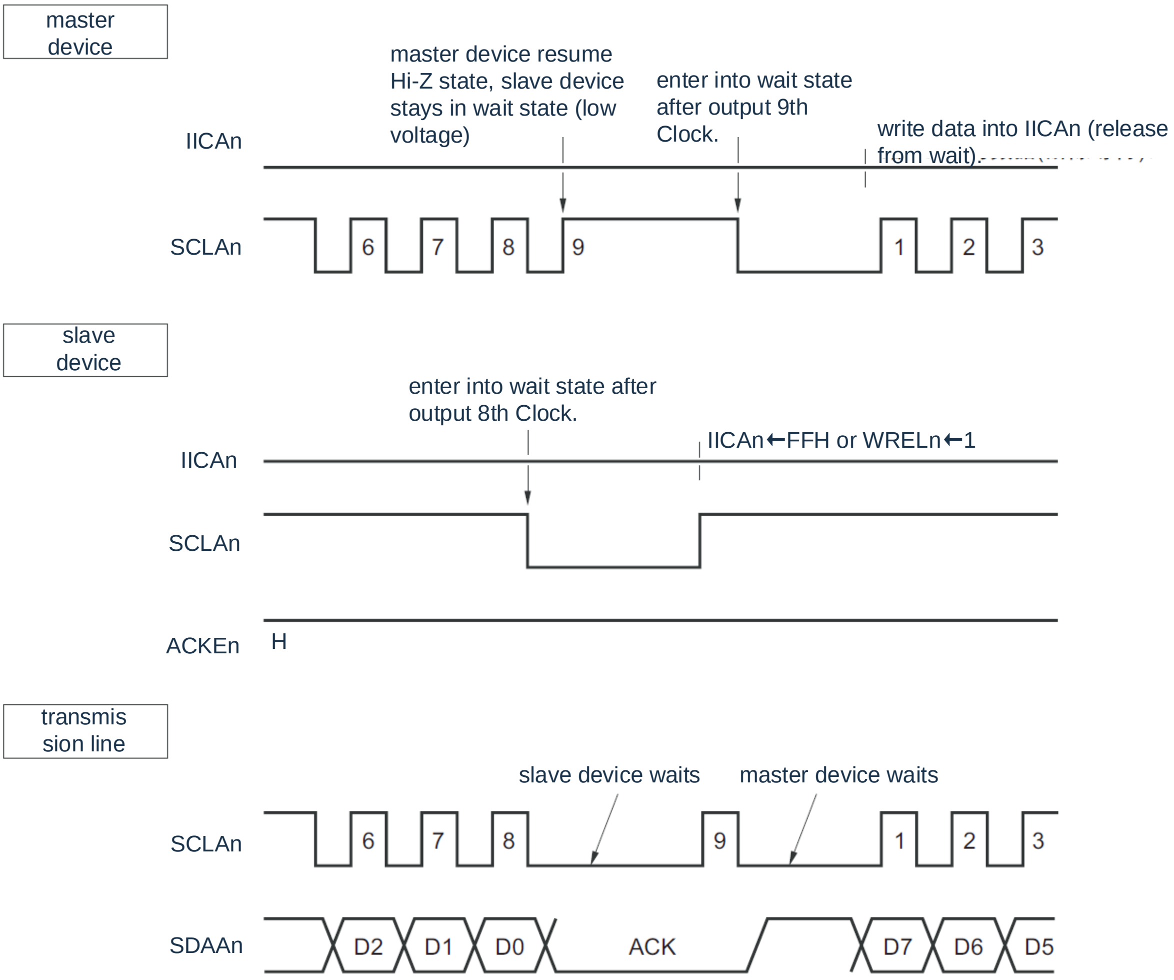
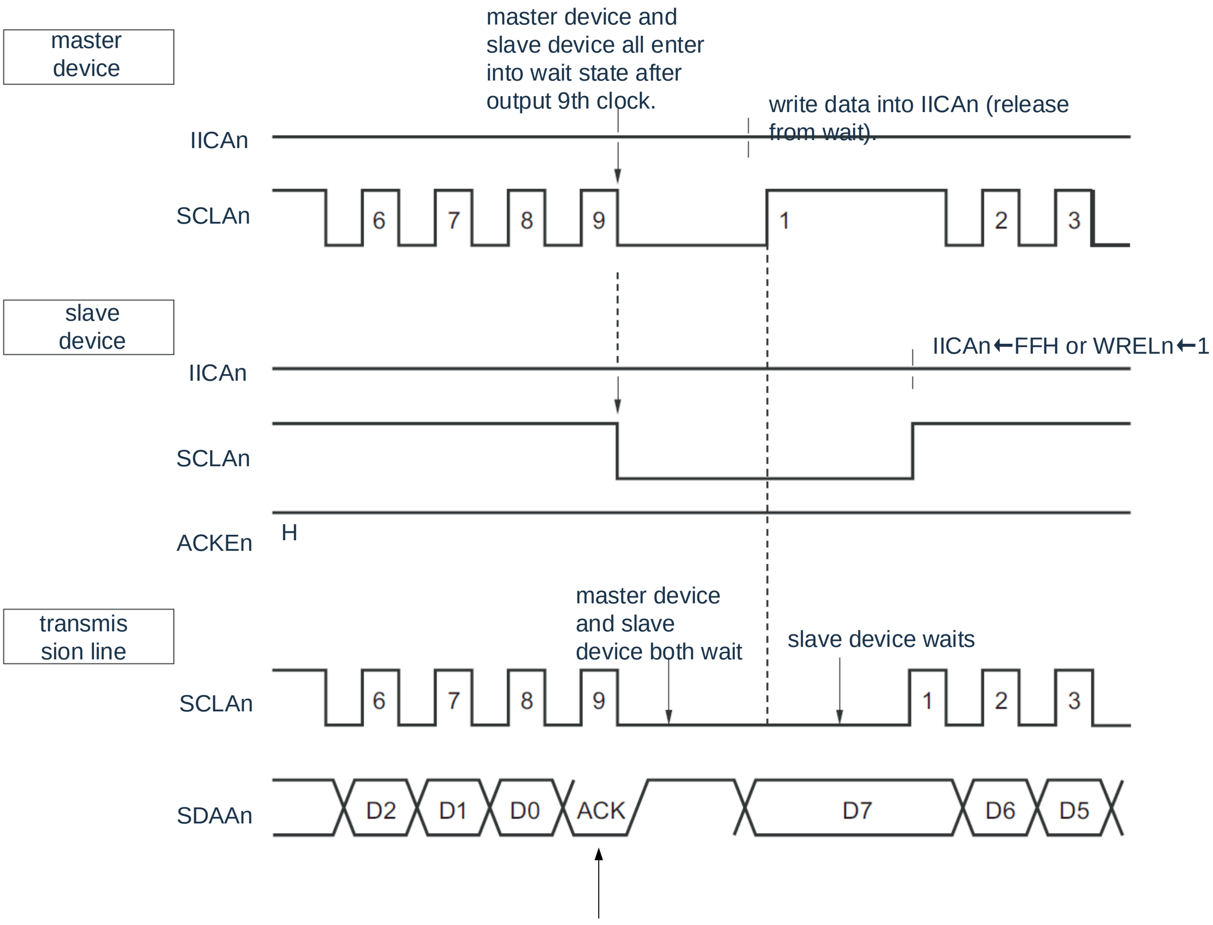

master device
master device resume Hi-Z state, slave device stays in wait state (low voltage)
enter into wait state after output 9th Clock.
write data into IICAn (release from wait).
IICAn
SCLAn
6
7
8
9
1
2
3
slave device
enter into wait state after output 8th Clock.
IICAn←FFH or WRELn←1
IICAn
SCLAn
ACKEn
H
transmis sion line
slave device waits
master device waits
SCLAn
6
7
8
9
1
2
3
SDAAn
D2
D1
D0
ACK
D7
D6
D5

Figure 14-19 Wait (2/2)

(2) The case where both the master and slave devices are waiting for 9 clocks (Master device: send, slave device: receive, ACKEn=1).


master device
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 device
IICAn←FFH or WRELn←1
IICAn
SCLAn
ACKEn
H
transmis sion 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.

Note ACKEn: bit2 of IICA control register n0 (IICCTLn0).
WRELn: Bit5 of IICA control register n0 (IICCTLn0).

The wait state is automatically generated by setting bit3 (WTIMn) of IICA control register n0 (IICCTLn0). Typically, on the receiver, if bit5 (WRELn) of the IICCTLn0 register is “1” or if the IICA shift register is given n (IICAn) writes “FFH” and releases the wait; On the sender, if data is written to the IICAn register, the wait is released. The master device can also unwait in the following ways:

- Set bit1 (STTn) of the IICCTLn0 register to “1”.
- Set bit0 (SPTn) of the IICCTLn0 register to “1”.

Note n=0

14.5.7 Release method of wait

In general, I2C can release the wait with the following processing.

- Write data to IICA shift register n (IICAn).
- Set bit5 (WRELn) of IICA control register n0 (IICCTLn0) (wait release).
- Set bit1 (STTn) of the IICCTLn0 register (generate start condition) Note.
- Set bit0 (SPTn) of the IICCTLn0 register (generate stop condition) Note.

Note: Only for the master control device.

If these wait releases handling is performed, I2C releases the wait and resumes communication. To send data (including addresses) after release wait, data must be written to the IICAn register.

To receive data after released wait or to end sending data, bit5 (WRELn) of the IICCTLn0 register must be set to “1”. To generate a restart condition after release wait, bit1 (STTn) of the IICCTLn0 register must be set to “1”. To generate a stop condition after release wait, bit0 (SPTn) of the IICCTLn0 register must be set to “1”. Release processing can only be performed once for one wait.

For example, if data is written to the IICAn register after release wait by removing the wait at the WRELn is “1”, the timing of the change in the SDAAn line may conflict with the write timing of the IICAn register, resulting in the wrong value being output to SDAAn line. In addition to these processing, in the case of a neutral stop of communication, if the IICEn bit is cleared to “0”, the communication is stopped, so the wait can be releaseed. In the case where the I2C-bus state is deadlocked due to noise, if bit6 (LRELn) of the IICCTLn0 register is set “1”, the communication is exited, so the wait can be released.

Note If the pending dismissal process is performed when the

WUPn bit is “1”, the wait is not released.

Remark n=0

14.5.8 Interrupt request (INTIICAn) generation timing and wait control

Control register n0 (IICCTLn0) by setting bit3 (WTIMn) by setting IICA in Table 14-2 The timing shown generates INTIICAn and performs wait control.

Table 14-2 Timing and wait control of INTIICAn

WTIMn	Slave operation			Master operation		
	address	Data reception	Data sending	address	Data reception	Data sending
0	9Note1, 2	8Note2	8Note2	9	8	8
1	9Note1, 2	9Note2	9Note2	9	9	9

Note 1. The slave generates INTIICAn on the falling edge of the 9th clock only if the received address and the set address of the slave address register n (SVAn) are the same signal and enter a waiting state.

At this point, regardless of the setting of bit2 (ACKEn) of the IICCTLn0 register, a response is generated. The slave device that receives the extension code generates INTIICAn on the falling edge of the 8th clock. If the address is different after the restart, INTIICAn is generated on the falling edge of the 9th clock, but does not enter the waiting state.

2. If the contents of the received address and the dependent address register n (SVAn) are different and the extension code is not received, INTIICAn is not generated and does not enter the wait state.

The numbers in the Remarks table represent the number of clocks for the serial clock. Both interrupt request and wait control are synchronized to the falling edge of the serial clock.

(1) Transmitting and receiving addresses

- Slave operation: Independent of the WTIMn bit, the timing of interruptions and waits is determined according to the conditions in Notes 1 and 2 above.
- Master Operation: Independent of the WTIMn bit, the timing of interrupts and waits is generated on the falling edge of the 9th clock.

(2) Data reception

- Master operation/Slave operation: Determines the timing of interrupts and waits by WTIMn bit.

(3) Data transmission

- Master operation/Slave operation: Determines the timing of interrupts and waits by WTIMn bit.

Note n=0

(4) Release method of the wait

There are 4 ways to undo the waiting:

- Write data to IICA shift register n (IICAn).
- Set bit5 (WRELn) of IICA control register n0 (IICCTLn0) (unwait).
- Set bit1 (STTn) of the IICCTLn0 register (generate start condition) Note.
- Set bit0 (SPTn) of the IICCTLn0 register (generate stop condition) Note.

Note: Only for the master control device.

When selecting a wait for 8 clocks (WTIMn=0), you need to decide whether to generate a reply before release wait.

(5) Detection of stop conditions

If a stop condition is detected, INTIICAn is generated (only in the case of SPIEn=1).

14.5.9 Detection method for address matching

In I^2C-bus mode, the master device can select a specific slave by sending a slave address. Address matching can be automatically detected by hardware. An INTIICAn interrupt request is generated when the slave address sent by the master device and the set address of the slave address register n (SVAn) are the same or only the extension code is received.

14.5.10 Detection of errors

In I^2C-bus mode, because the state of the serial data bus (SDAAn) during the transmit is fetched to the IICA shift register n (IICAn) of the transmitting device, Therefore, I can detect sending errors by comparing the IICA data before and after the start of the transmission. In this case, if the two data are different, it is judged that a sending error has occurred.

Remark n=0

14.5.11 Extension code

(1) When the high 4 bits of the receive address are “0000” or “1111”, as the extension code is received, the extension code receive flag (EXCn) is set to “1”, and in the 8th The falling edge of the clocks generates an interrupt request (INTIICAn).

Does not affect the local station address saved in the slave address register n (SVAn).

(2) When the setting value of the SVAn register is “11110xx0”, if “11110xx0” is sent from the master device through a 10-bit address, the following assertions occur. However, an interrupt request (INTIICAn) is generated on the falling edge of the 8th clock.

- High 4 bits data are the same: EXCn=1
- 7 bits of data are the same: COIn=1

Note EXCn: bit5 of the IICA status register n (IICSn).
COIn: bit4 of IICA status register n (IICSn).

(3) The processing after the interrupt request occurs differs depending on the subsequent data of the extension code, which is processed by software. If the extension code is received while the slave is running, it is participating in the communication even if the address is different. For example, if you do not want to operate as a slave after receiving an extension code, you must set bit 6 (LRELn) of IICA control register n0 (IICCTLn0) to "1" to enter the standby state for the next communication.

Table 14-3 Bit definitions of major extension codes

Slave address	R/W bit	Illustrate
0000000	0	Full call address
11110xx	0	Specification of a 10-bit slave address (when the address is authenticated).
11110xx	1	The designation of a 10-bit slave address (when a read command is issued after the address is identical).

Note 1. For extensions other than those listed above, please refer to the I^{2}C-bus datasheet issued by NXP Corporation.
2.n=0

14.5.12 Arbitration

When multiple master devices generate start conditions at the same time (in the case of STTn is “1” before the STDn bit becomes “1”), the communication of the master device is carried out while adjusting the clock until the data is different. This operation is called arbitration.

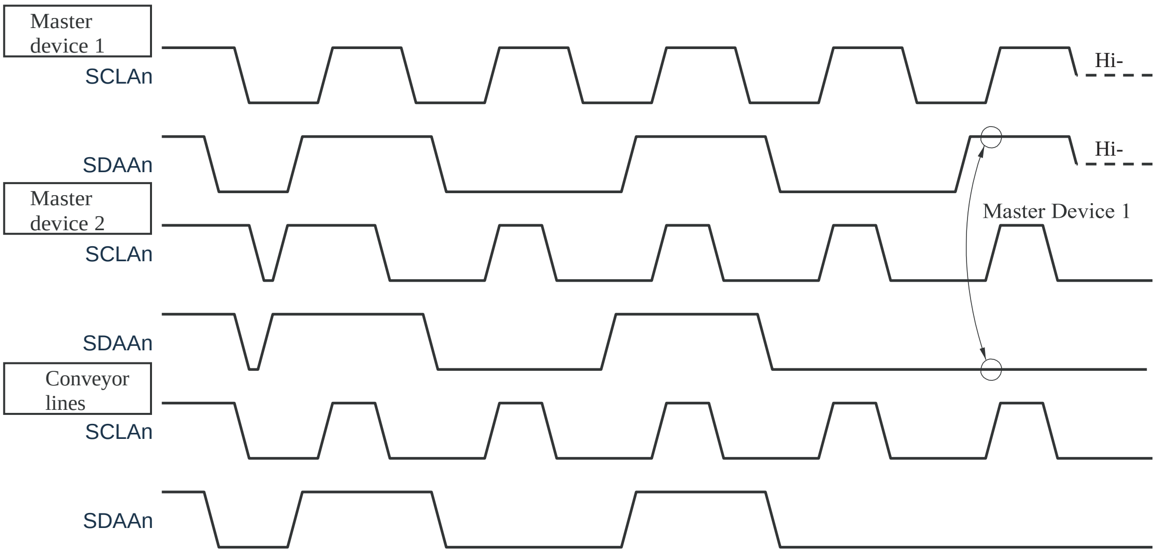
In case of arbitration failure, the master device with arbitration failure sets the arbitration failure flag (ALDn) in the IICA status register n (IICSn) to "1" and sets both the SCLAn and SDAAn lines to the high impedance state to release the bus.

In the event of the next interrupt request (e.g. a stop condition detected on the 8th or 9th clock), the software is passed through the ALDn bit as “1” to detect the failure of arbitration.

For the generation timing of interrupt requests, refer to “16.5.8 Generation Timing and Wait Control for Interrupt Requests (INTIICAn)”.

Note STDn: bit1 of the IICA status register n (IICSn).
STTn: Bit1 of IICA control register n0 (IICCTLn0).

Figure 14-20 Example of arbitration timing



Note n=0

Table 14-4 Status of the arbitration and the timing of the interrupt request

Status at the time arbitration occurred	Timing of the interrupt request
The address is sent during the sending process	On the falling edge of the 8th or 9th clock after byte transfer[Note 1]
Read and write information after sending the address	
During the sending of extension codes	
Read and write information after sending the extension code	
During data transmission	
During the delivery of the reply after the data is sent	
A restart condition was detected during data transfer.	
A stop condition was detected during data transfer.	When generating the stop condition (SPIEn=1)[Note 2].
You want to generate a restart condition, but the data is low.	On the falling edge of the 8th or 9th clock after byte transfer[Note 1]
You want to generate a restart condition, but a stop condition is detected.	When generating the stop condition (SPIEn=1)[Note 2].
You want to generate a stop condition, but the data is low.	On the falling edge of the 8th or 9th clock after byte transfer[Note 1]
You want to generate a restart condition, but SCLAn is low.	

Note 1. When WTIMn bit (bit 3 of IICA control register n0 (IICCTLn0)) is "1", an interrupt request is generated on the falling edge of the 9th clock; when WTIMn bit is "0" and the slave address of the extension code is received, an interrupt request is generated on the falling edge of the 8th clock.

2. When arbitration is possible, the SPIEn bit must be set to "1" when the master is running.

Note 1. SPIEn: Bit4 of the IICA control register n0 (IICCTLn0).

2. n=0

14.5.13 Wake-up function

This is a subordinate function of I2C, which is the function of generating an interrupt request signal (INTIICAn) when the local station address and extension code are received. The processing efficiency is improved by not generating unwanted INTIICAn signals at different addresses. If a start condition is detected, wake-up standby is entered. Because the master device (in the case where a start condition has been generated) may also become a slave due to a failed arbitration, it enters wake-up standby at the same time as sending the address.

To use the wake-up function in deep sleep mode, the WUPn bit must be set to "1". The address can be received independent of the running clock. Even in this case, the interrupt request signal (INTIICAn) is generated when the local station address and extension code are received. After generating this interrupt, the WUPn bit is cleared to "0" by the instruction and returns to normal operation.

The flow when the WUPn bit is set to "1" is shown in Figure Figure 14-21, and the flow when the WUPn bit is set to "0" by address matching is shown in Figure 14-22.

Figure 14-21 Flow when the WUPn bit is set to "1"

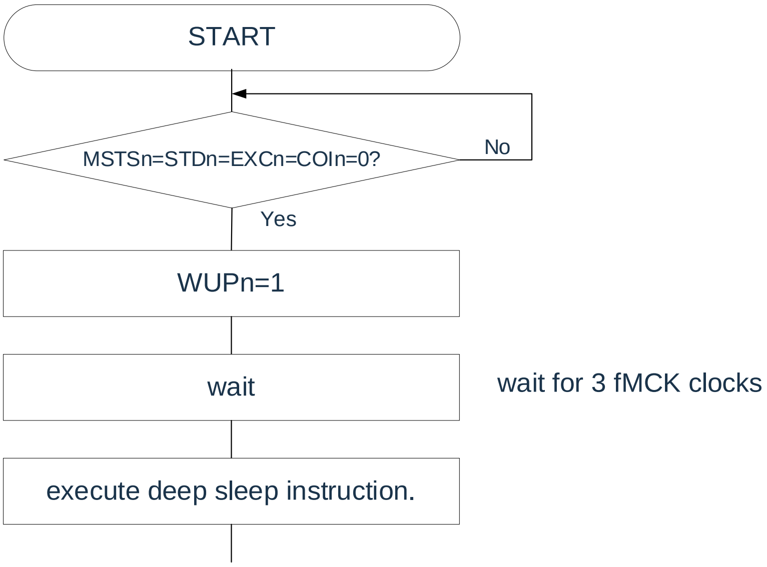
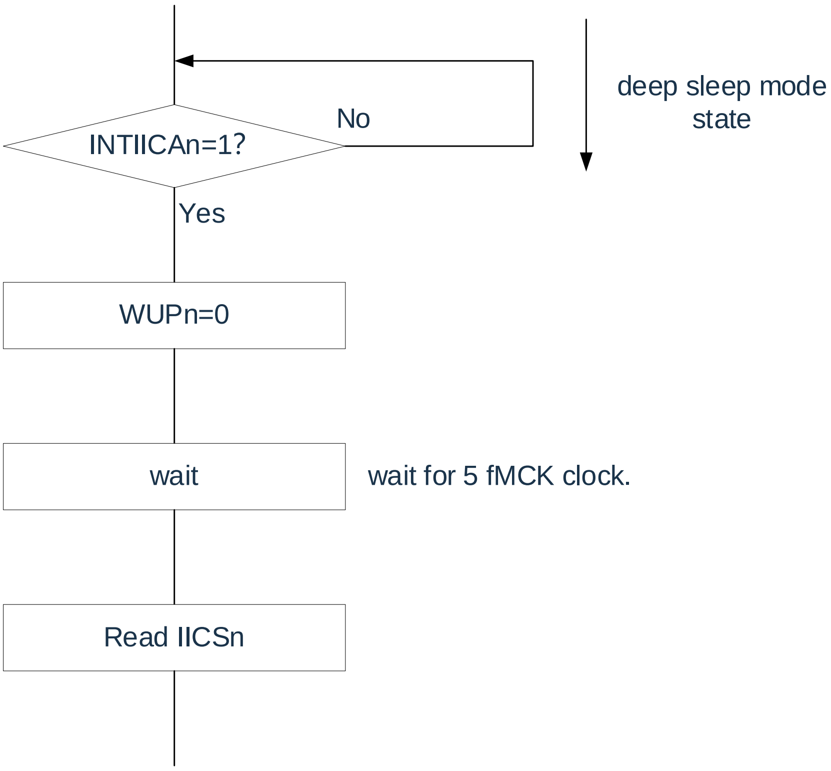


Figure 14-22 Flow when the WUPn bit is set to "0" by address matching (including receiving extension codes)



after confirming serial interface IICA operation status, process accordingly.

In addition to interrupt requests (INTIICAn) generated by the serial interface IICA, deep sleep mode must be released through the following procedure.

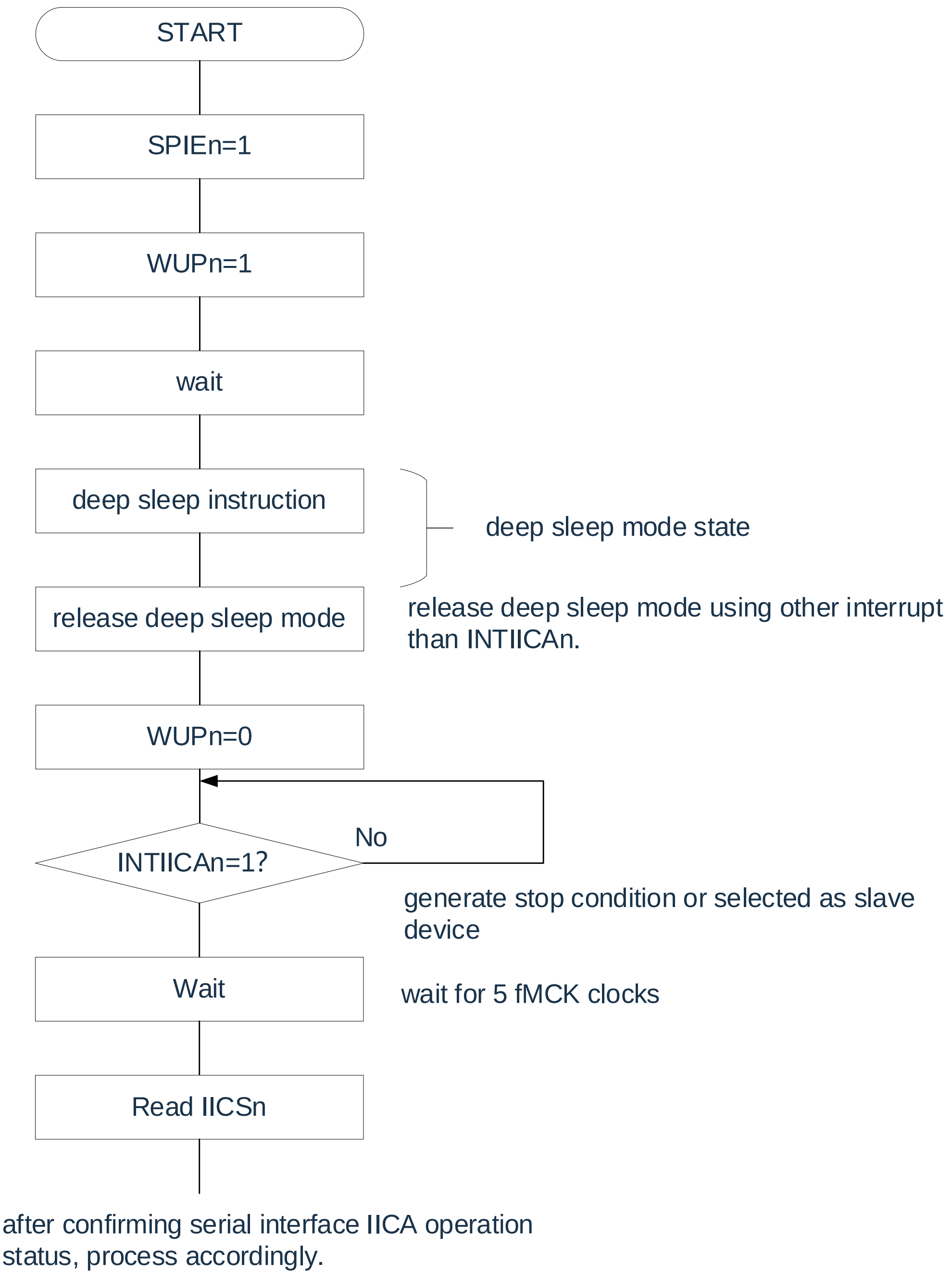
- The next time IIC communication is in operation with the master control device: Figure 14-23
- The next time the IIC communication is in the case of the slave running for the device:

The case returned by INTIICAn interrupt: Same as Figure 14-22.

Cases returned by interrupt other than INTIICAn interrupt: The WUPn bit must be kept at "1" to continue operation until the INTIICAn interrupt is generated.

Remark n=0

Figure 14-23 Operation as a master device after being released from deep sleep mode 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

14.5.14 Communicate with reservation

(1) Cases where the communication reservation function is allowed (bit0(IICRSVn) = 0 for IICA flag register n (IICFn))

To perform the next master communication without joining the bus, you can send a start condition when the bus is released by making a communication appointment. At this time, do not join the bus includes the following two states:

- When the arbitration result is neither a master nor a slave
- Does not run as a slave device after receiving the extension code (does not return a reply and sets bit6 (LRELn) of the IICA control register n0 (IICCTLn0). “1”, the bus is released after exiting communication).

If bit1 (STTn) of the IICCTLn0 register is set to “1” without joining the bus, a start condition is automatically generated after the bus is released (stop condition detected) and enters a waiting state.

Put bit4 (SPIEn) of the IICCTLn0 register to “1” after the release of the bus (stop condition detected) is detected by the resulting interrupt request signal (INTIICAn), if given The IICA shift register n (IICAn) writes the address and automatically begins communication as the master device. The data written to the IICAn register is invalid until a stop condition is detected.

When stTn is set to "1", it is decided whether to run as a start condition or as a communication appointment depending on the bus state.

- When the bus is in a released state......... Generate start conditions
- When the bus is not in the released state (standby)... Communicate with an appointment

After setting the STTn bit to "1" and the wait time has elapsed, the MSTSn bit (bit 7 of the IICA status register n (IICSn)) is used to confirm whether or not to operate as a communication reservation.

The wait time calculated by the following calculation must be ensured by software:

Wait time from placing STTn position "1" until the MSTSn flag is confirmed:

$(\text{IICWLn} + \text{IICWHn} + 4) / f_{MCK} + t_F \times 2$

Note 1. IICWLn: IICA low width setting register n

IICWHn : IICA high level width setting register n

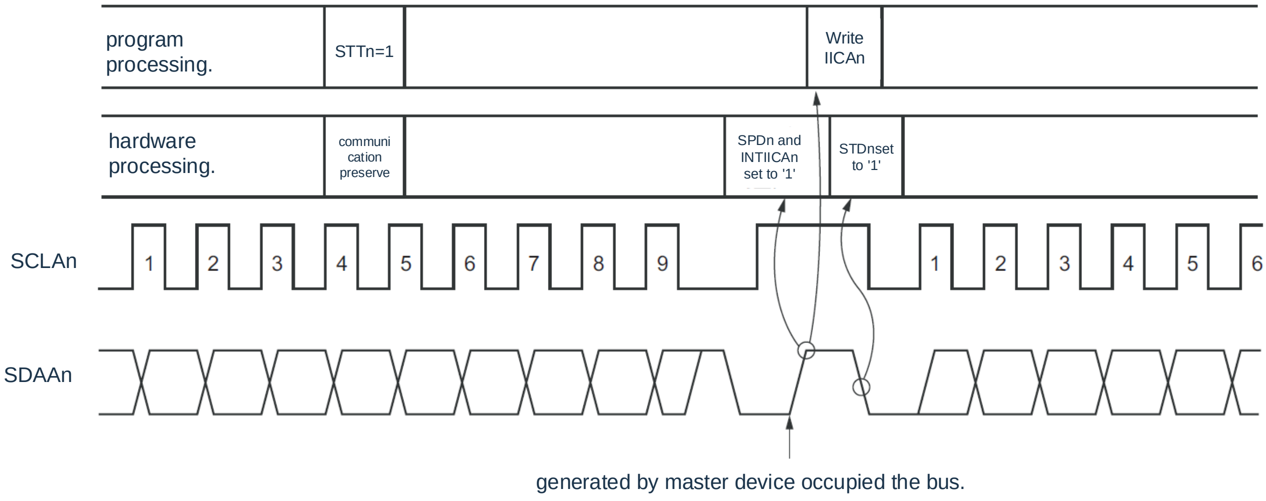
t_F: Falling time for SDAAn signal and SCLAn signal

f_{MCK}: IICA operates at the clock frequency

2. n=0

The timing of the communication reservation is shown in the following figure.

Figure 14-24 Timing of communication reservation



Note IICAn: IICA shift register n

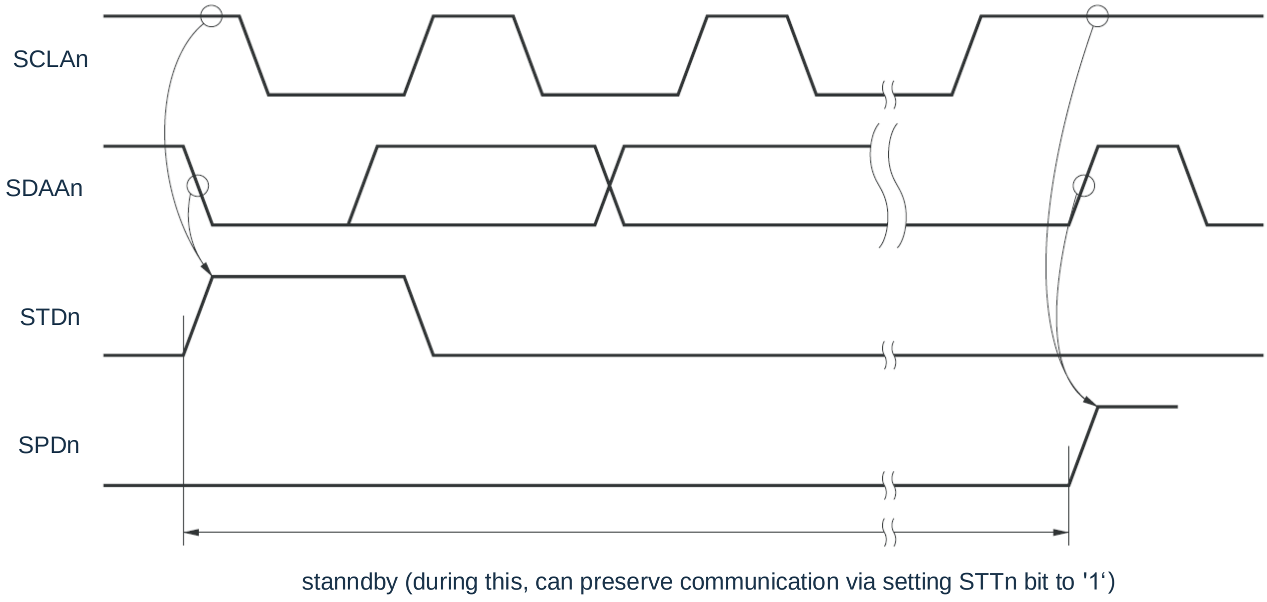
STTn: bit1 of IICA control register n0 (IICCTLn0).

STDn: bit1 of IICA status register n (IICSn).

SPDn: bit0 of IICA status register n (IICSn).

The communication reservation is accepted by the timing sequence shown in Figure 14-25. After bit 1 (STDn) of the IICA status register n (IICSn) becomes "1" and before the stop condition is detected, set bit 1 (STTn) of the IICA control register n0 (IICCTLn0) to "1 " for communication reservation.

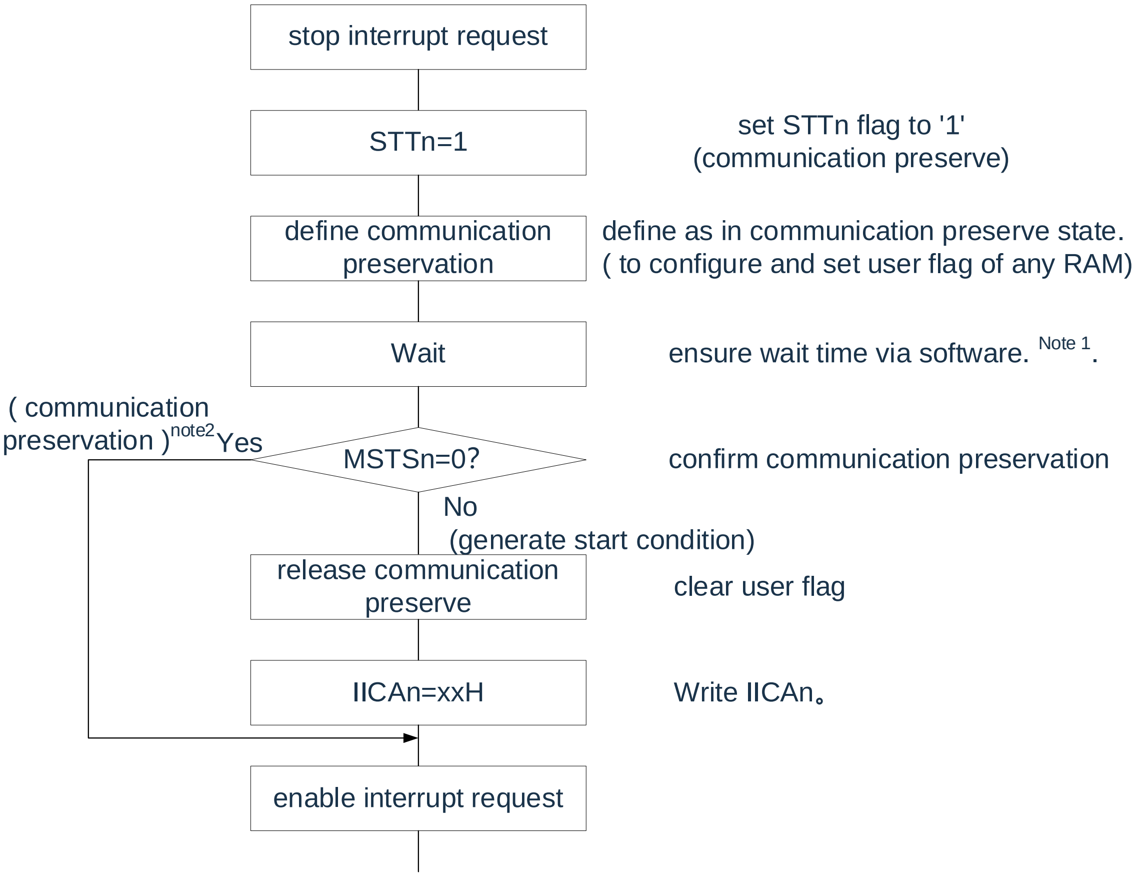
Figure 14-25 Receiving timing of communication reservation


SCLAn
SDAAn
STDn
SPDn
standby (during this, can preserve communication via setting STTn bit to '1')

The steps to communicate a reservation is shown in Figure 14-26.

Remark n=0

Figure 14-26 Steps to communicate an appointment


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 times are as follows: (config value for IICWLn + config valuefor IICWHn +4)/f_{MCK}+t_F×2

2. Write IICA shift register n (IICAn) by stopping conditional interrupt requests while the communication reservation is running.

Note 1. STTn	: Bit1 of IICA control register n0 (IICCTLn0).
MSTSn	: bit7 of IICA status register n (IICSn).
IICAn	: IICA shift register n
IICWLn	: IICA low level width setting register n
IICWHn	: IICA high level width setting register n
t_F	: Drop time for SDAAn signal and SCLAn signal
f_{MCK}	: IICA operates at the clock frequency

2.n=0

(2) When the communication reservation function is disabled (bit0 (IICRSVn) of IICA flag register n (IICFn) = 1)

During bus communication, if bit1(STTn) of IICA control register n0 (IICCTLn0) is set to “1” in the state of not participating in this communication, this request is rejected without generating a start condition. At this time, do not join the bus includes the following two states:

- When the arbitration result is neither a master nor a slave
- Does not run as a slave device after receiving the extension code (bit6 (LRELn) of the IICCTLn0 register is set to “1” without returning a reply, and the bus is released after exiting communication).

The STCFn (bit 7 of the IICFn register) can be used to confirm whether a start condition has been generated or a request has been rejected. Since it takes 5 f_{MCK} clocks from the time the STTn bit is "1" to the time the STCFn bit is set to "1", this time must be ensured by software.

Remark n=0

14.5.15 Other cautions

(1) When the STCENn bit is “0”

After just allowing I^2C to run (IICEn=1), it is considered a communication state (IICBSYn=1) independent of the actual bus state. To communicate with the master without detecting a stop condition, it is necessary to create a stop condition and communicate with the master after the bus is released. For multi-master, master communication cannot occur in the state where the bus is not released (no stop condition detected). Stop conditions are generated in the following order:

(1) Set IICA control register n1 (IICCTLn1).
(2) Set bit7 (IICEn) of IICA control register n0 (IICCTLn0) to “1”.
(3) Set bit0 (SPTn) of the IICCTLn0 register to “1”.

(2) When the STCENn bit is “1”

After I^2C is just allowed to run (IICEn=1), it is considered a release state (IICBSYn=0) regardless of the actual bus state. Therefore, when the first starting condition (STTn=1) is generated, it is necessary to confirm that the bus has been released in order not to disrupt other communications.

(3) When I^2C communication is being made with other devices

When the SDAAn pin is low and the SCLAn pin is high, I^2C macros are considered SDAAn citations if I^2C is allowed to run and participate in communication in the middle the foot changes from high to low (start condition detected). If the value on the bus is recognized as an extension code at this point, a reply is returned that interferes with I^2C communication with other devices. To avoid this, I^2C must be started in the following order:

(1) Clear the bit4 (SPIEn) of the IICCTLn0 register to "0" to disable the generation of an interrupt request signal (INTIICAn) when a stop condition is detected.
(2) Set bit7 (IICEn) of the IICCTLn0 register to “1” to allow I2C to run.
(3) Wait for the start condition to be detected.
(4) Before returning the answer (within 4 to 72 F_{MCK} clocks after setting the IICEn bit to "1"), set bit 6 (LRELn) of the IICCTLn0 register to "1" to force the detection to be disabled.

(4) After setting the STTn bit and SPTn bit (bit1 and bit0 of the IICCTLn0 register), the reset before clearing "0" is prohibited.

(5) If a communication appointment is made, the SPIEn bit (bit 4 of the IICCTLn0 register) must be set to "1" to generate an interrupt request when a stop condition is detected. After the interrupt request is generated, the communication data is written to the IICA shift register n (IICAn) to start the transmission. If no interrupt occurs when the stop condition is detected, the communication stops in the wait state because no interrupt request is generated at the start of communication. However, when the MSTSn bit (bit 7 of the IICA status register n(IICSn)) is detected by software, it is not necessary to set the SPIEn bit to "1".

Remark n=0

14.5.16 Communication operation

The following 3 running steps are represented here by a flowchart.

(1) Master operation of a single-master system

The flowchart used as a master device in a single master system is shown below.

This process is broadly divided into “initialization” and “communication processing”. Performs the Initial Setup section at startup and, if communication with the slave is required, the Communication Handling section after the preparation required for communication.

.

(2) Master operation of multi-master systems

In a multi-master system of the I^{2}C-bus, it is not possible to determine whether the stage bus participating in the communication is in the release state or in the use state according to the specifications of the I^{2}C-bus alone. Here, if the data and clock are high within a certain period of time (1 frame), the bus participates in the communication as a release state. This process is broadly divided into Initialization, Communication Waiting, and Communication Processing. The processing that was designated as a slave due to arbitration failure is omitted here, and only indicates the processing used as the master device. Join the bus after performing the “Initial Setup” section at startup, and then wait for the communication request of the master device or the designation of the slave device through “Communication Wait”. The actual communication is carried out in the “Communication Processing” part, which supports arbitration with other master devices in addition to data transmission and reception with the slave device.

(3) Slave operation

An example of a slave device used as an I^{2}C-bus is shown below.

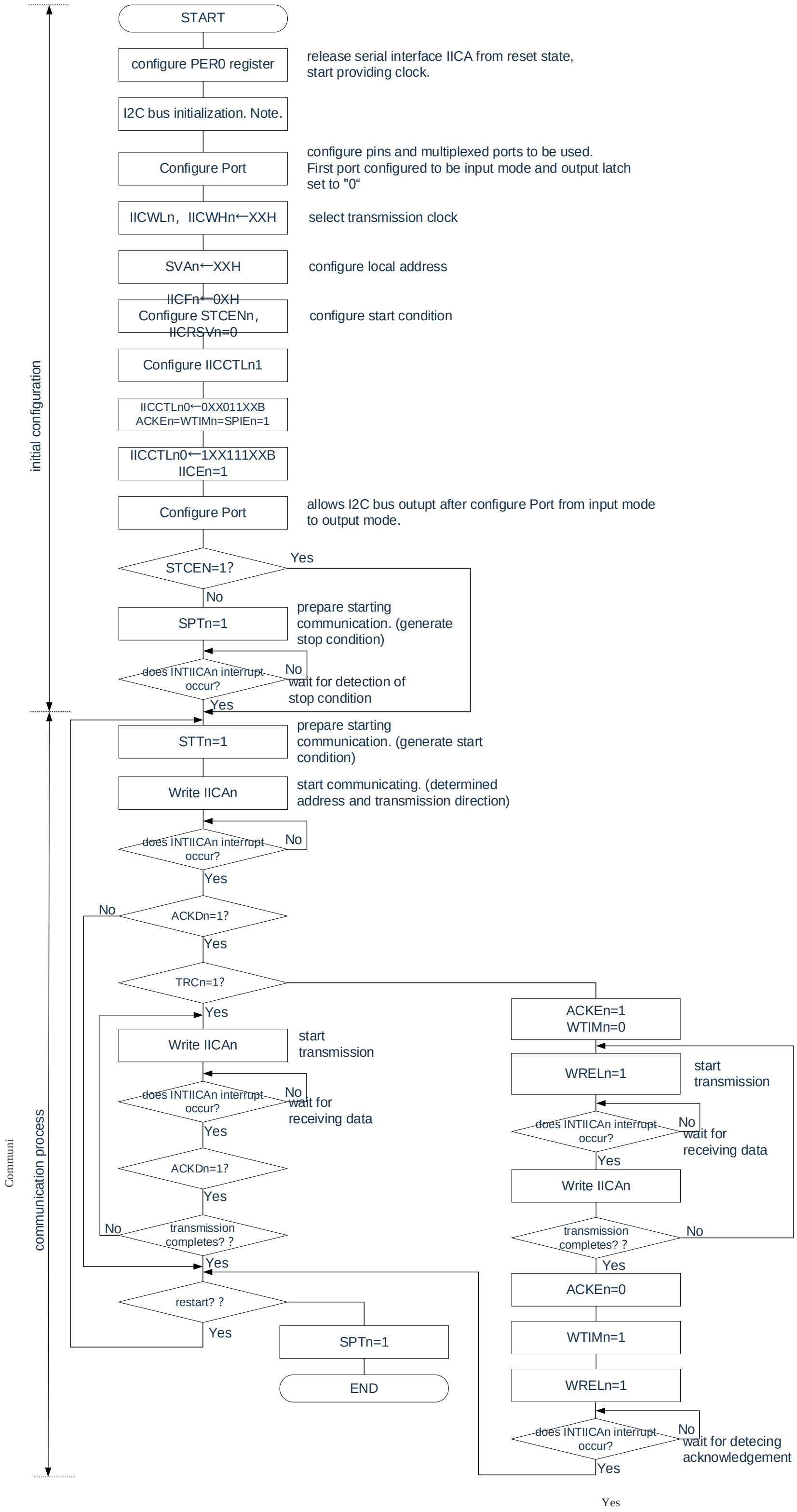
When used as a slave, it starts operating by interrupt. Perform the “Initialization” section at startup and then wait for the INTIICAn interrupt to occur via “Communication Wait”. If an INTIICAn interruption occurs, the communication status is determined and the flag is passed to the main processing department.

By checking each flag, the required “Communication processing” is performed.

Remark n=0

(1) Master operation of a single-master system

Figure 14-27 Master operation of a single-master system


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.
initial configuration
STCEN=1?
Yes
No
SPTn=1
prepare starting communication. (generate stop condition)
does INTIICAn interrupt occur?
No
wait for detection of stop condition
Yes
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
Communi
communication process
Yes

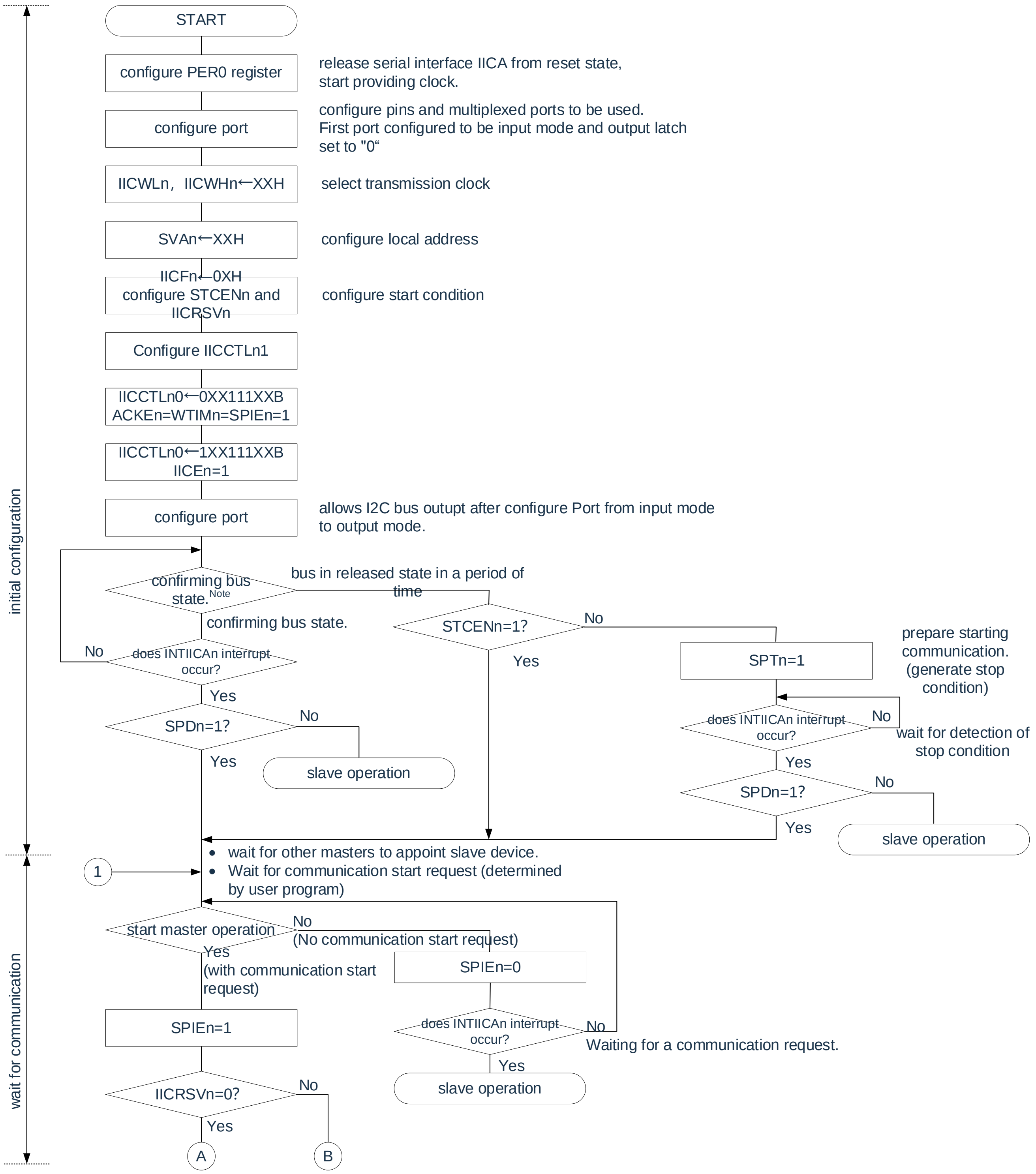
Note I²C-bus (SCLAn pins and SDAAn pins are high) must be released according to the specifications of the product in the communication. For example, if the EEPROM is in a state of output low to the SDAAn pin, the SCLAn pin must be set to the output port and clock pulses from the output port before the SDAAn pin is fixed high.

Remark 1. The format sent and received must conform to the specifications of the product in the communication.

2.n=0

(2) Master operation of multi-master systems

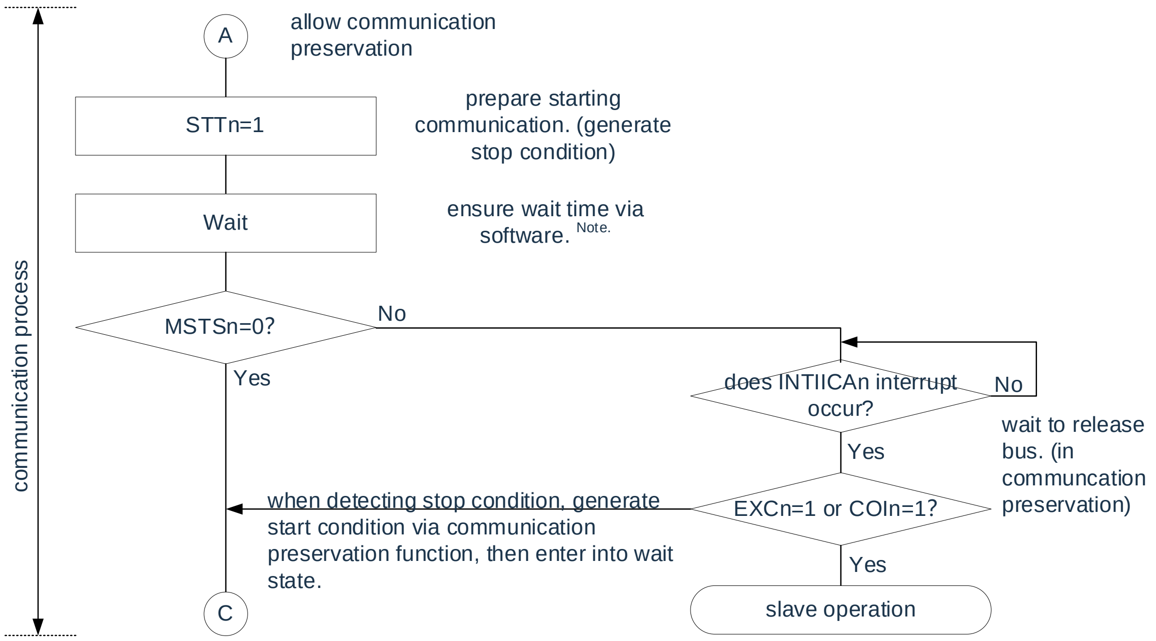
Figure 14-28 Master operation of multi-master systems (1/3)



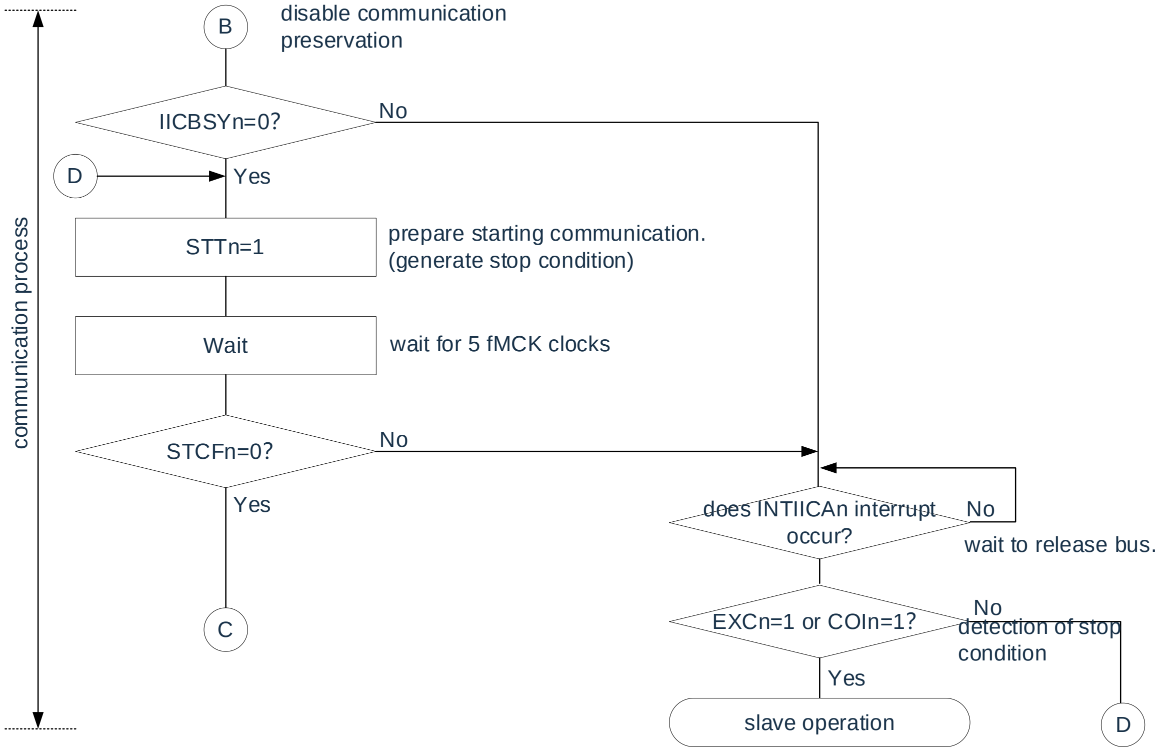
Note You must confirm that the bus is in the freed state for a certain period of time (e.g., 1 frame) (CLDn bit = 1, DADn bit = 1). When the SDAAn pin is fixed to a low level, it must be determined whether to release the I2C-bus (SCLAn pin and SDAAn) according to the specifications of the product in the communication pin is high).

Note n=0

Figure 14-28 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
Yes
wait to release bus. (in communcation preservation)
EXCn=1 or COIn=1?
Yes
when detecting stop condition, generate start condition via communication preservation function, then enter into wait state.
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. IICWL: IICA low width setting register n

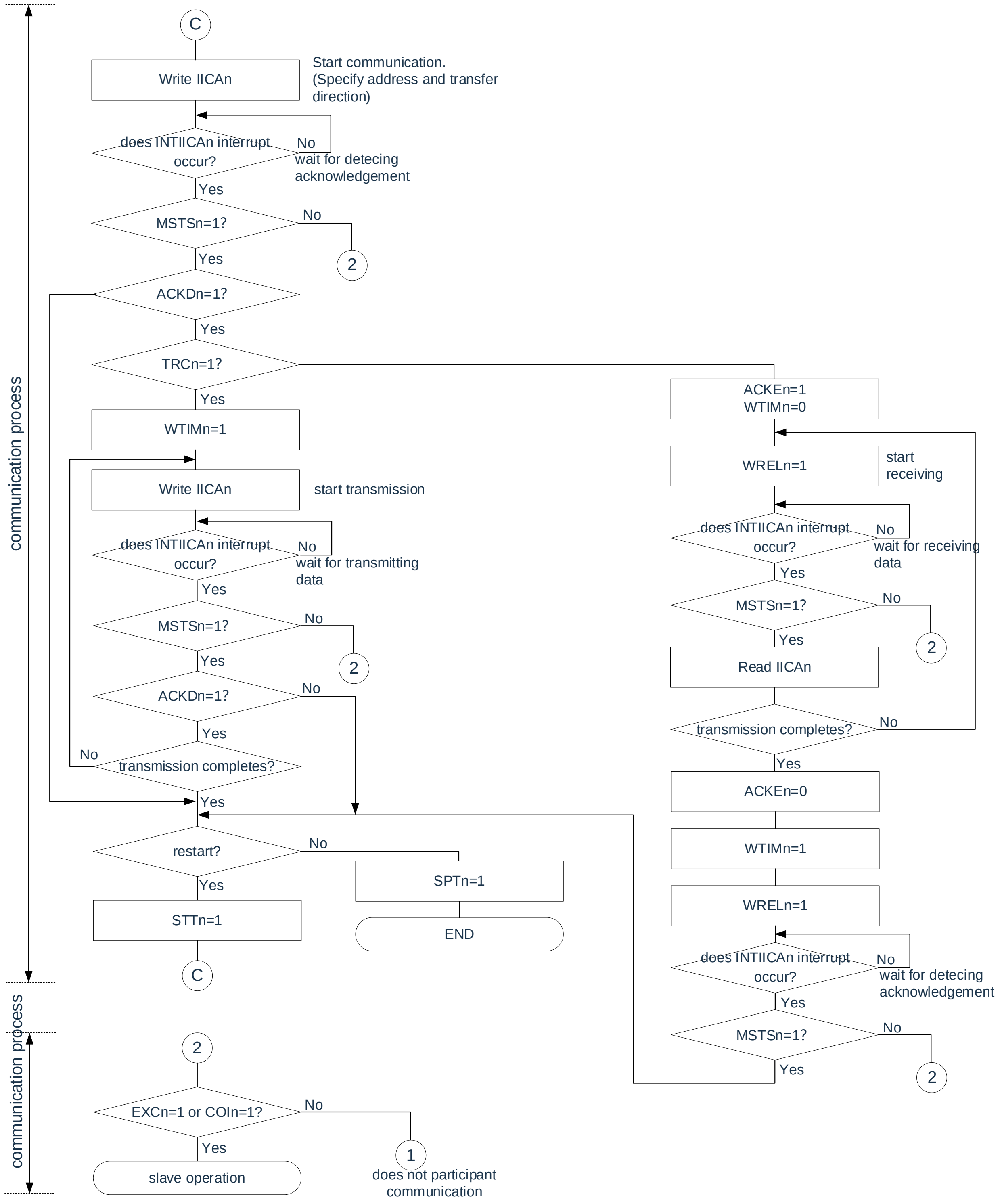
IICWHn: IICA high level width setting register n

t_F: Drop time for SDAAn signal and SCLAn signal

f_{MCK}: IICA operates at the clock frequency

2. n=0

Figure 14-28 Master operation of multi-master systems (3/3)



Note 1. The format of transmission and reception must conform to the specifications of the product in the communication.

2. In the case of being used as a master device in a multi-master system, the MSTSn bit must be read every time an INTIICAn interruption occurs to confirm the arbitration result.

3. In the case of use as a slave device in a multi-master system, the IICA status register n (IICSn) and IICA flag registers must be passed every time an INTIICAn interrupt occurs n (IICFn) confirms the status and decides on future processing.

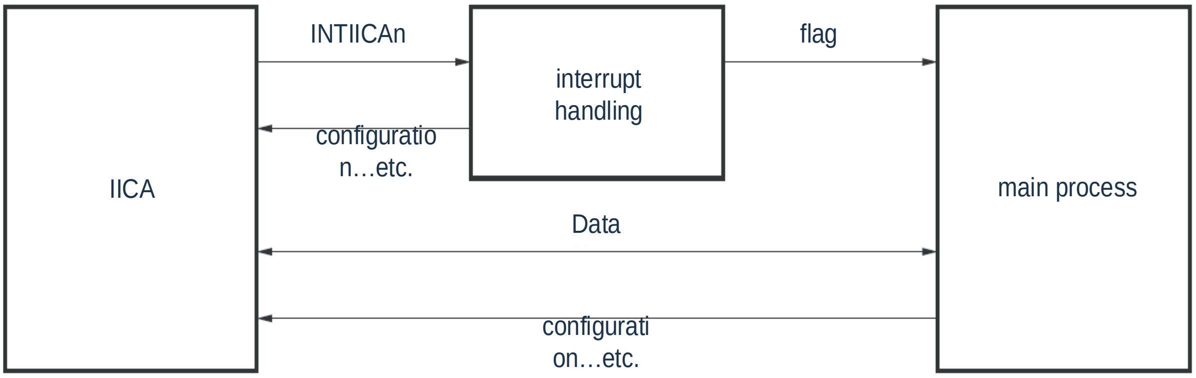
4. n=0

(3) Slave operation

The processing steps for the slave operation are as follows.

Dependent operations are basically event-driven, so they need to be handled by INTIICAn interrupts (which requires a lot of change processing of the operational state such as stop condition detection in communications).

In this note, it is assumed that the data communication does not support the extension code, the INTIICAn interrupt processing only performs the state transfer processing and the actual data communication is carried out by the main processing department.



Therefore, the following three flags are prepared and instead of INTIICAn the flag is passed to the main processing department for data communication processing.

① Communication mode flag

This flag indicates the following two communication states:

- Clear Mode: Not in the presence of data communication
- Communication mode: the status of ongoing data communication (detection of valid address ~ detection of stop condition, no response of the master control device is detected, different addresses).

(2) Ready sign

This flag indicates that data communication can occur. In normal data communication, the same as the INTIICAn interrupt, which is asserted by the interrupt handling part and cleared by the main processing department. When communication begins, the flag is cleared by the interrupt handling department. However, when sending the first data, the interrupt processing department does not assert the readiness flag, so the first data is sent without clearing the flag (address matching is interpreted as the next data request).

(3) Communication direction signs

This flag indicates the direction of communication, and the value of the TRCn bit is the same.

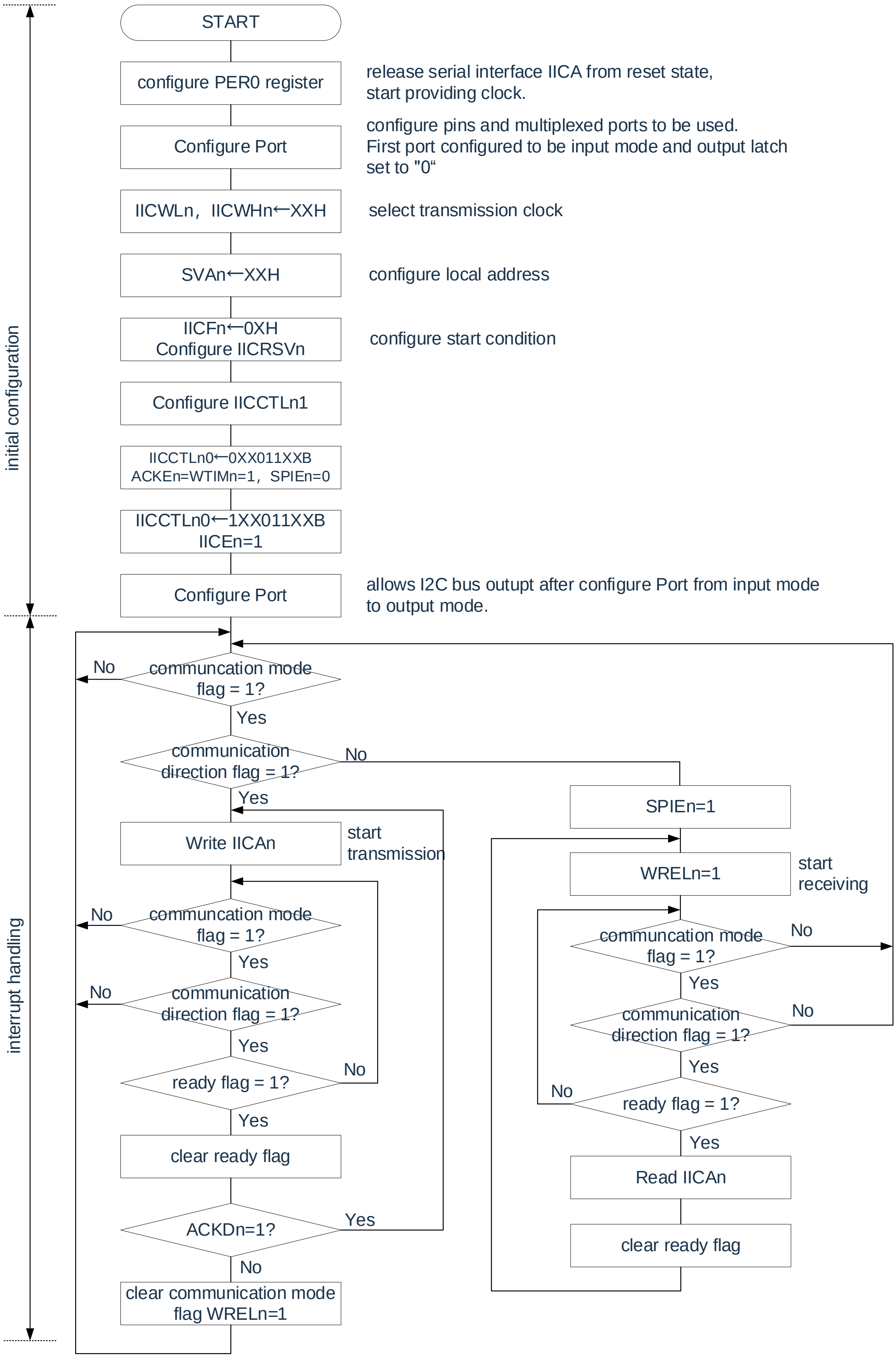
Remark n=0

The operation of the master processing part of the slave operation is shown below.

Start the serial interface IICA and wait for it to become communicable. If it becomes communicable, the communication mode flag and the readiness flag are used to communicate (because the stop condition and start condition are processed by interrupt, the status is confirmed by the flag here).

At the time of sending, the send is repeated until the master device does not return a reply. If the master does not return a reply, communication ends. At the time of receiving, the required amount of data is received. If the communication ends, no reply is returned at the next data. Thereafter, the master device generates a stop condition or a restart condition to exit the communication state.

Figure 14-29 Slave operation step (1)



Note 1 The format of transmission and reception must conform to the specifications of the product in the communication.

2.n=0

An example of a step for a slave to process via an INTIICAn interrupt is shown below (in this case, it is assumed that no extension code is used). The status is confirmed by INTIICAn interrupt and the following processing is performed.

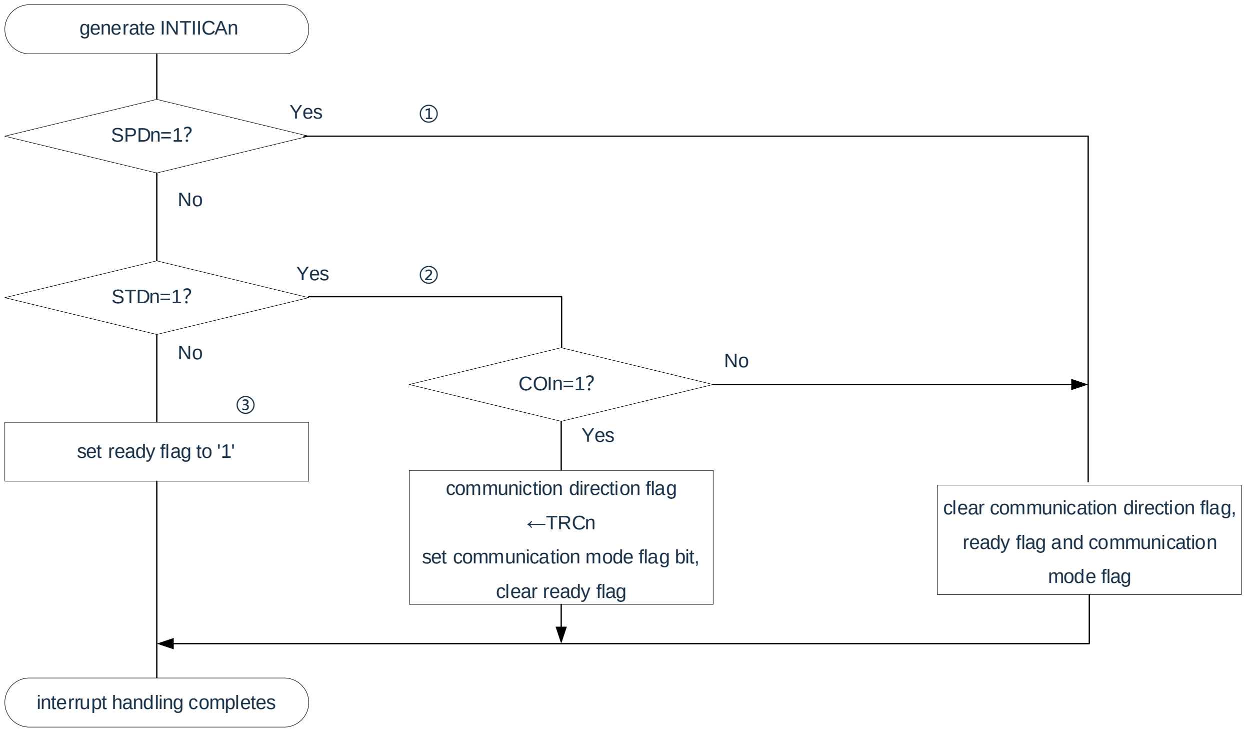
(1) If a stop condition is generated, the communication ends.

(2) If a start condition is generated, confirm the address. If the addresses are different, the communication ends. If the addresses are the same, set to communication mode and unwait, and then return from the interrupt (clear the readiness flag).

(3) When sending and receiving data, as long as the ready flag is asserted, the I2Cbus remains in a waiting state and returns from the interrupt.

Remark (1) to (3) above corresponds to (1) ~ (3) of “Figure 14-30 Slave operation step (2)

Figure 14-30 Slave operation step (2)


generate INTIICAn
SPDn=1?
Yes
①
No
STDn=1?
Yes
②
No
COIn=1?
No
③
set ready flag to '1'
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

14.5.17 Generation timing of I^{2}C interrupt request (INTIICAn)

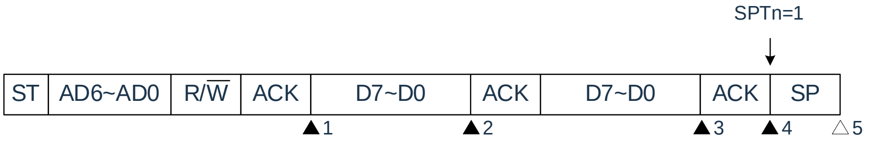
The values of the transmit and receive timing of the data, the generation timing of the INTIICAn interrupt request signal, and the IICA status register n (IICSn) when the INTIICAn signal is generated are shown below.

Remarks 1. ST : Start conditions
AD6 to AD0 : Address
R/W : The designation of the transmission direction
ACK : Acknowledge
D7~D0 : Data
SP : Stop condition
2. n=0

(1) Master operation

(a) Start~Address~Data~Data~Stop

(i) When WTIMn = 0



▲1:IICSn=1000X110B

▲2:IICSn=1000X000B

▲3:IICSn=1000X000B(set WTIMn bit to“1”) Note

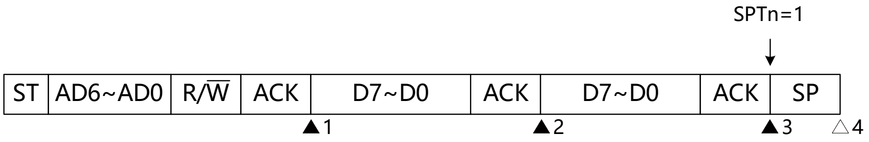
▲4:IICSn=1000XX00B(set SPTn bit to “1”)

△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”)

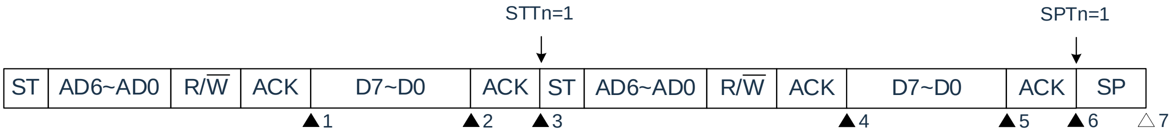
△4:IICSn=00000001B

Remark	▲	must generate
	△	only generate while SPIEn bit is '1'
	X	any

Note n=0

(b) Start~Address~Data~Start~Address~Data~Stop (Restart)

(i) When WTIMn = 0



▲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'.

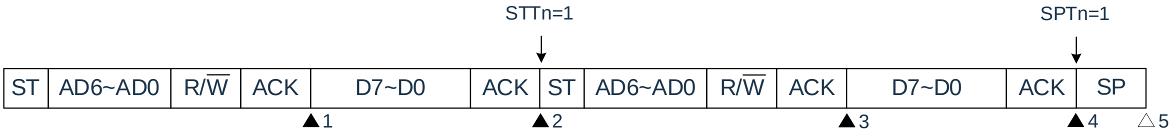
3. to generate stop condition, must set WITIMn bit to '1' and modify INTIICAn interrupt request signal generation timing sequenœ.

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

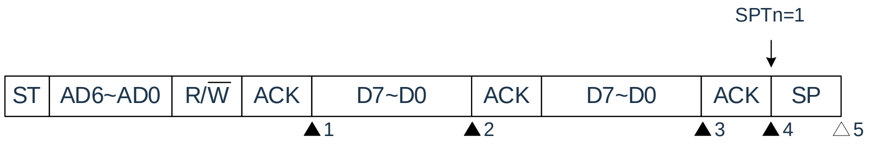
△ only generate while SPIEn bit is '1'

X any

Note n=0

(c) Start~Code~Data~Data~Stop (Send extension code)

(i) When WTIMn = 0



▲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

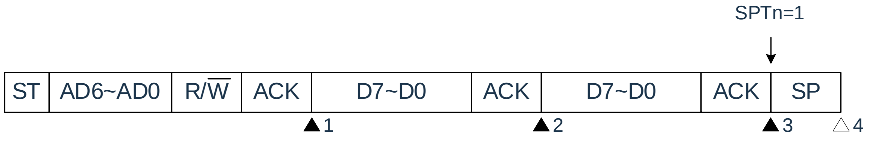
Note: to generate stop condition, must set WITIMn bit to '1' and modify INTIICAn interrupt request signal generation timing sequence.

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'

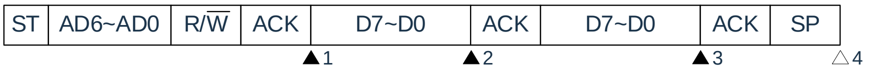
X any

Note n=0

(2) Slave operation (When receiving a slave address)

(a) Start~Address~Data~Data~Stop

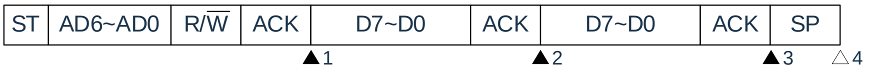
(i) When WTIMn = 0



▲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



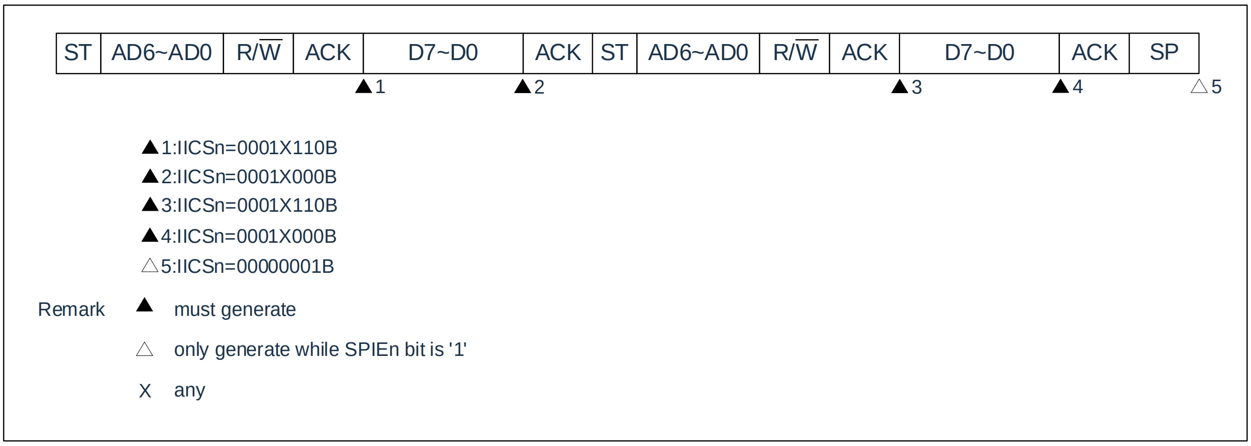
▲1:IICSn=0001X110B
▲2:IICSn=0001X100B
▲3:IICSn=0001XX00B
△4:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

Note n=0

(b) Start~Address~Data~Start~Address~Data~Stop

(i) When WTIMn = 0 (SVAn is the same after restart).



ST	AD6~AD0	R/$\overline{W}$	ACK	D7~D0	ACK	ST	AD6~AD0	R/$\overline{W}$	ACK	D7~D0	ACK	SP

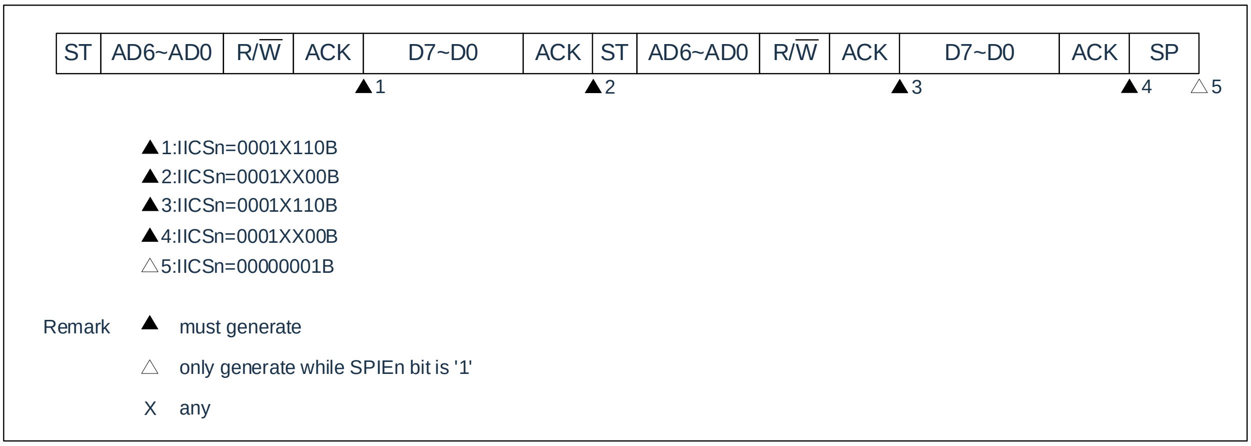
▲1:IICSn=0001X110B
▲2:IICSn=0001X000B
▲3:IICSn=0001X110B
▲4:IICSn=0001X000B
△5:IICSn=00000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

(ii) When WTIMn = 1 (SVAn is the same after restart).



ST	AD6~AD0	R/$\overline{W}$	ACK	D7~D0	ACK	ST	AD6~AD0	R/$\overline{W}$	ACK	D7~D0	ACK	SP

▲1:IICSn=0001X110B
▲2:IICSn=0001XX00B
▲3:IICSn=0001X110B
▲4:IICSn=0001XX00B
△5:IICSn=00000001B

Remark ▲ must generate

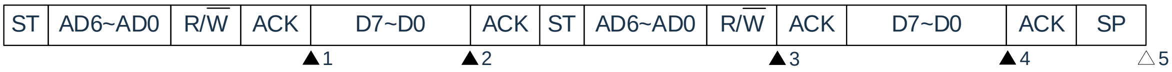
△ only generate while SPIEn bit is '1'

X any

Note n=0

(c) Start~Address~Data~Start~Code~Data~Stop

(i) I When WTIMn=0 (different addresses after restarting (extension code)).



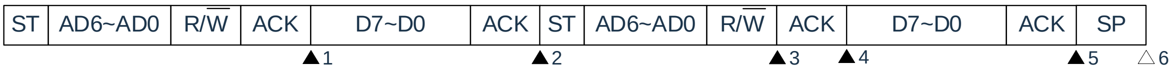
▲1:IICSn=0001X110B
▲2:IICSn=0001X000B
▲3:IICSn=0010X010B
▲4:IICSn=0010X000B
△5:IICSn=00000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

(ii) When WTIMn=1 (different addresses 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

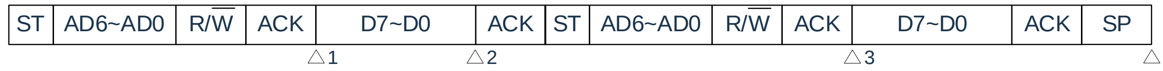
△ only generate while SPIEn bit is '1'

X any

Note n=0

(d) Start~Address~Data~Start~Address~Data~Stop

(i) When WTIMn=0 (different addresses after restart (non-extension)).



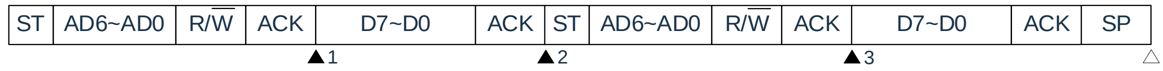
△1:IICSn=0001X110B
△2:IICSn=0001X000B
△3:IICSn=00000110B
△4:IICSn=00000001B

备注 △ must generate

△ only generate while SPIEn bit is '1'

X any

(ii) When WTIMn=1 (different addresses after restarting (non-extension)).



▲1:IICSn=0001X110B
▲2:IICSn=0001XX00B
▲3:IICSn=00000110B
△4:IICSn=00000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

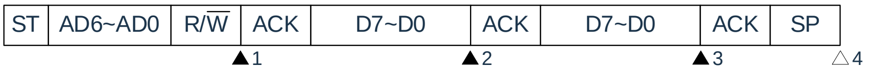
Note n=0

(3) Slave operation (case of receiving extension code).

Always participate in communication when receiving extension codes.

(a) Start~Code~Data~Data~Stop

(i) When WTIMn = 0



▲1:IICSn=0010X010B

▲2:IICSn=0010X000B

▲3:IICSn=0010X000B

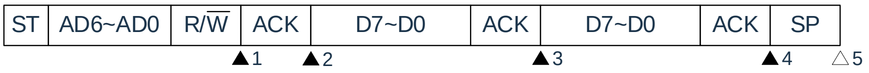
△4:IICSn=00000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

(ii) In the case of 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

Note n=0

(b) Start~Code~Data~Start~Address~Data~Stop

(i) When WTIMn = 0 (SVAn is the same after restart).

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		△5

▲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 (SVAn is the same after restart).

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		▲5	△6

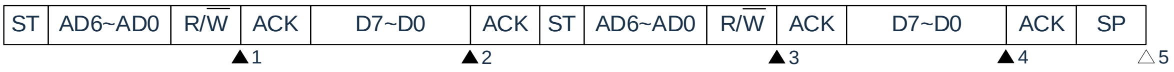
▲1:IICSn=0010X010B
▲2:IICSn=0010X110B
▲3:IICSn=0010XX00B
▲4:IICSn=0001X110B
▲5:IICSn=0001XX00B
△6:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

Note n=0

(c) Start~Code~Data~Start~Code~Data~Stop

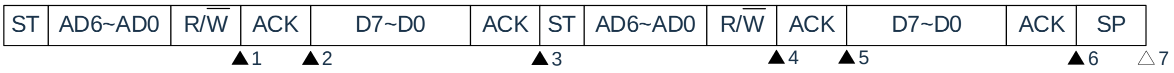
(i) When WTIMn=0 (extension code received after restarting).



▲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 (extension code received after restarting).



▲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

Note n=0

(d) Start~Code~Data~Start~Address~Data~Stop

(i) When WTIMn=0 (different addresses after restart (non-extension)).

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=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 addresses after restarting (non-extension)).

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			△5

▲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

Note n=0

(4) Do not participate in the running of the communication

(a) Start~Code~Data~Data~Stop

ST	AD6~AD0	R/$\overline{W}$	ACK	D7~D0	ACK	D7~D0	ACK	SP
								△1

△1:IICSn=00000001B

Remark △ only generate while SPIEn bit is '1'

(5) The failed operation of the arbitration (running as a slave after the arbitration failed).

When used as a master device in a multi-master system, the MSTSn bit must be read each time the INTIICAn interrupt request signal is generated to confirm the arbitration result.

(a) A condition in which arbitration fails during the sending of slave address data

(i) When WTIMn = 0

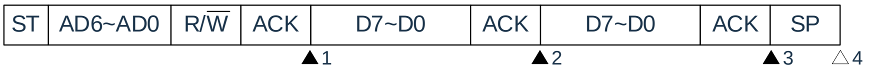
ST	AD6~AD0	R/$\overline{W}$	ACK	D7~D0	ACK	D7~D0	ACK	SP
			▲1	▲2		▲3		△4

▲1:IICSn=0101X110B
▲2:IICSn=0001X000B
▲3:IICSn=0001X000B
△4:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

Note n=0

(ii) When WTIMn=1



▲1:IICSn=0101X110B
▲2:IICSn=0001X100B
▲3:IICSn=0001XX00B
△4:IICSn=00000001B

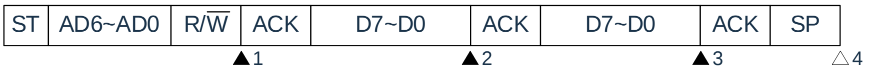
Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

(b) A condition in which arbitration fails during the sending of the extension code

(i) When WTIMn = 0



▲1:IICSn=0110X010B
▲2:IICSn=0010X000B
▲3:IICSn=0010X000B
△4:IICSn=00000001B

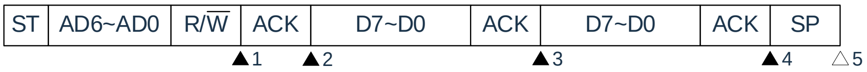
Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

Note n=0

(ii) When WTIMn=1



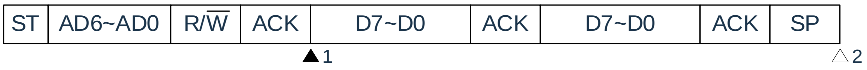
▲1:IICSn=0110X010B
▲2:IICSn=0010X110B
▲3:IICSn=0010X100B
▲4:IICSn=0010XX00B
△5:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

(6) The failed operation of the arbitration (do not participate in the communication after the arbitration fails).

When used as a master device in a multi-master system, the MSTSn bit must be read each time the INTIICAn interrupt request signal is generated to confirm the arbitration result.

(a) Arbitration failure condition (WTIMn=1) during the sending of slave address data

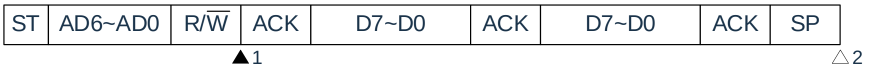


▲1:IICSn=01000110B
△2:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'

Note n=0

(b) A condition in which arbitration fails during the sending of the extension code



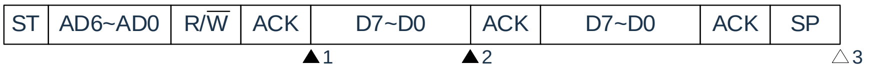
▲1:IICSn=01000110B
set LRELn bit to '1' via software
△2:IICSn=00000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

(c) A condition in which arbitration fails while transferring data

(i) When WTIMn = 0



▲1:IICSn=10001110B
▲2:IICSn=01000000B
△3:IICSn=00000001B

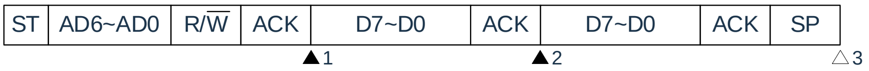
Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

Note n=0

(ii) When WTIMn=1



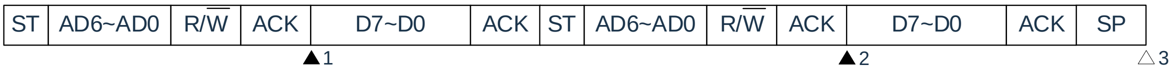
▲1:IICSn=10001110B
▲2:IICSn=01000100B
△3:IICSn=00000001B

Remark ▲ must generate

△ only generate while SPIEn bit is '1'

(d) A situation in which arbitration fails due to restart conditions when transferring data

(i) Non-extension code (for example, SVAn is different).



▲1:IICSn=1000X110B
▲2:IICSn=01000110B
△3:IICSn=00000001B

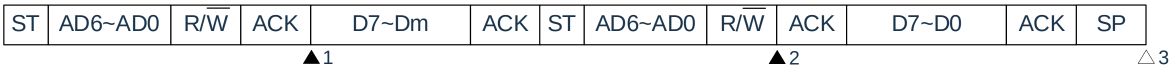
Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

Note n=0

(ii) Extension code



▲1:IICSn=1000X110B
▲2:IICSn=01000010B
set LRELn bit to '1' via software
△3:IICSn=00000001B

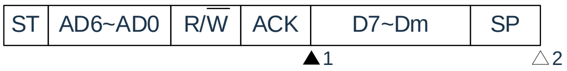
Remark ▲ must generate

△ only generate while SPIEn bit is '1'

X any

m=0~6

(e) A situation in which arbitration fails at the time of transmission due to a stop condition



▲1:IICSn=10000110B
△2:IICSn=01000001B

Remark ▲ must generate

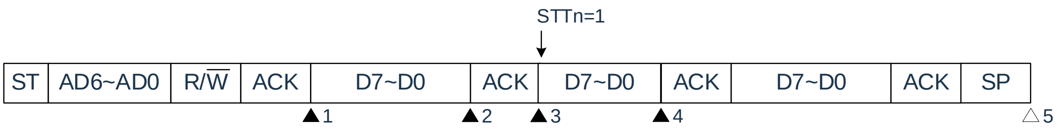
△ only generate while SPIEn bit is '1'

m=0~6

Note n=0

(f) A condition in which arbitration fails because the data is low when you want to generate a restart condition

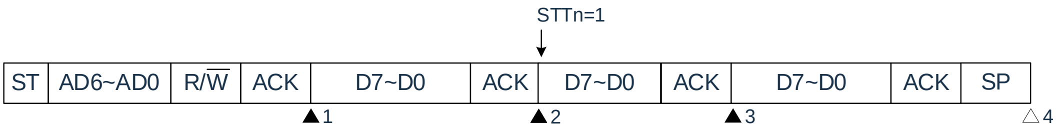
(i) When WTIMn = 0



▲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



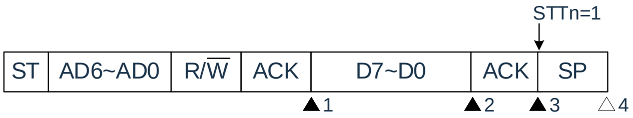
▲1:IICSn=1000X110B
▲2:IICSn=1000X100B(set STTn bit to “1”)
▲3:IICSn=01000100B
△4:IICSn=00000001B

Remark ▲ must generate
△ only generate while SPIEn bit is '1'
X any

Note n=0

(g) A situation in which arbitration fails because of a stop condition when you want to generate a restart condition

(i) When WTIMn = 0



▲1:IICSn=1000X110B

▲2:IICSn=1000X000B(set WTIMn bit to"1")

▲3:IICSn=1000XX00B(set STTn bit to "1")

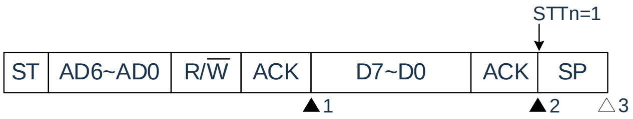
△4:IICSn=01000001B

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

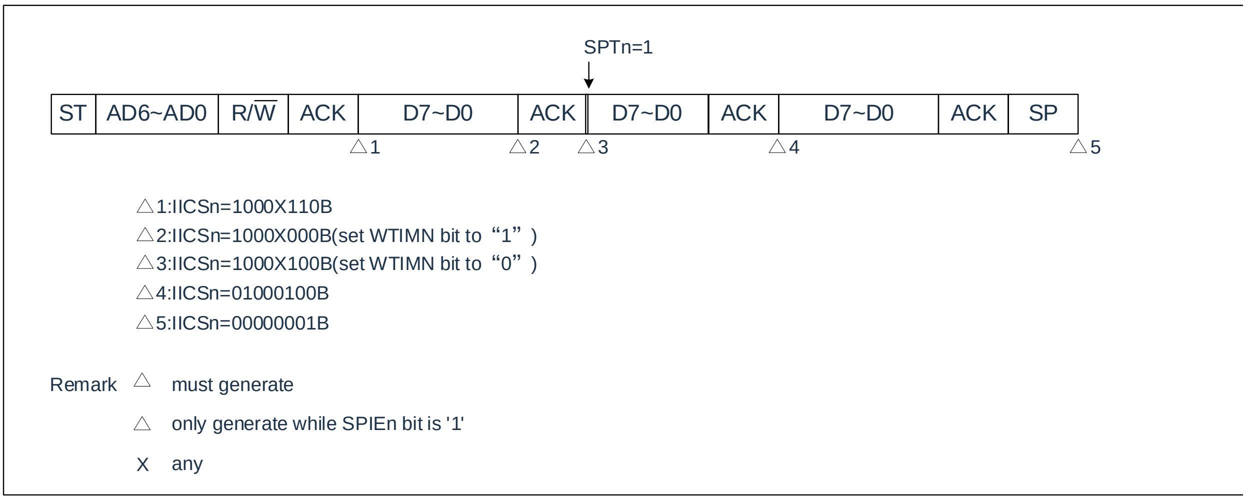
△ only generate while SPIEn bit is '1'

X any

Note n=0

(h) A condition in which arbitration fails because the data is low when you want to generate a stop condition

(i) When WTIMn = 0



△1:IICSn=1000X110B

△2:IICSn=1000X000B(set WTIMN bit to “1”)

△3:IICSn=1000X100B(set WTIMN bit to “0”)

△4:IICSn=01000100B

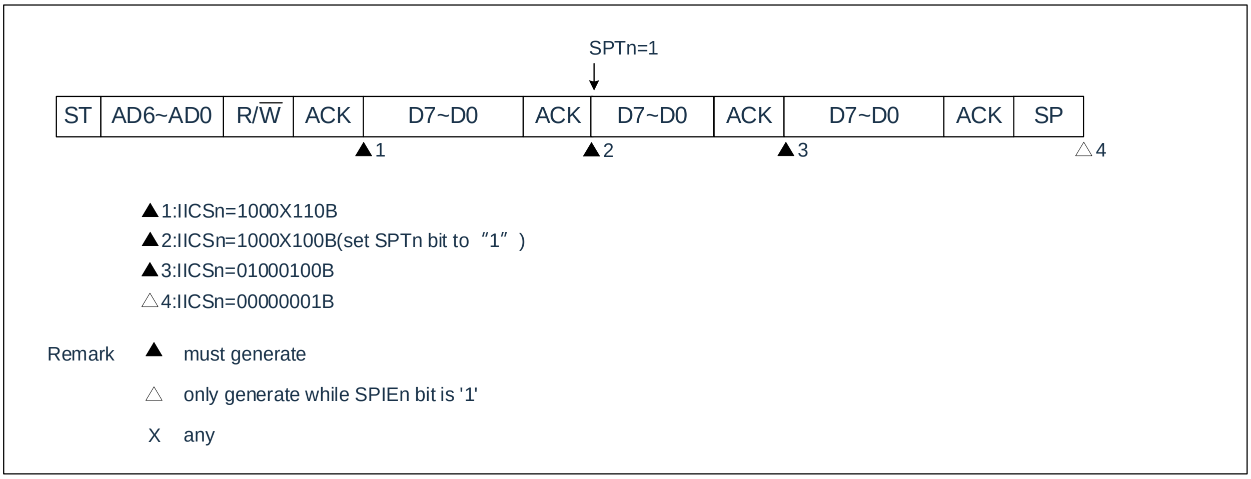
△5:IICSn=00000001B

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

Note n=0

14.6 Timing diagram

In I²C-bus mode, the master device selects a slave device for a communication object from multiple slave devices by giving the serial bus output address. The master device sends the TRCn bit (bit3 of the IICA status register n (IICSn)) indicating the direction of data transmission after the slave device address serial communication with the slave begins. The timing diagram of the data communication is shown in Figure 14-31.

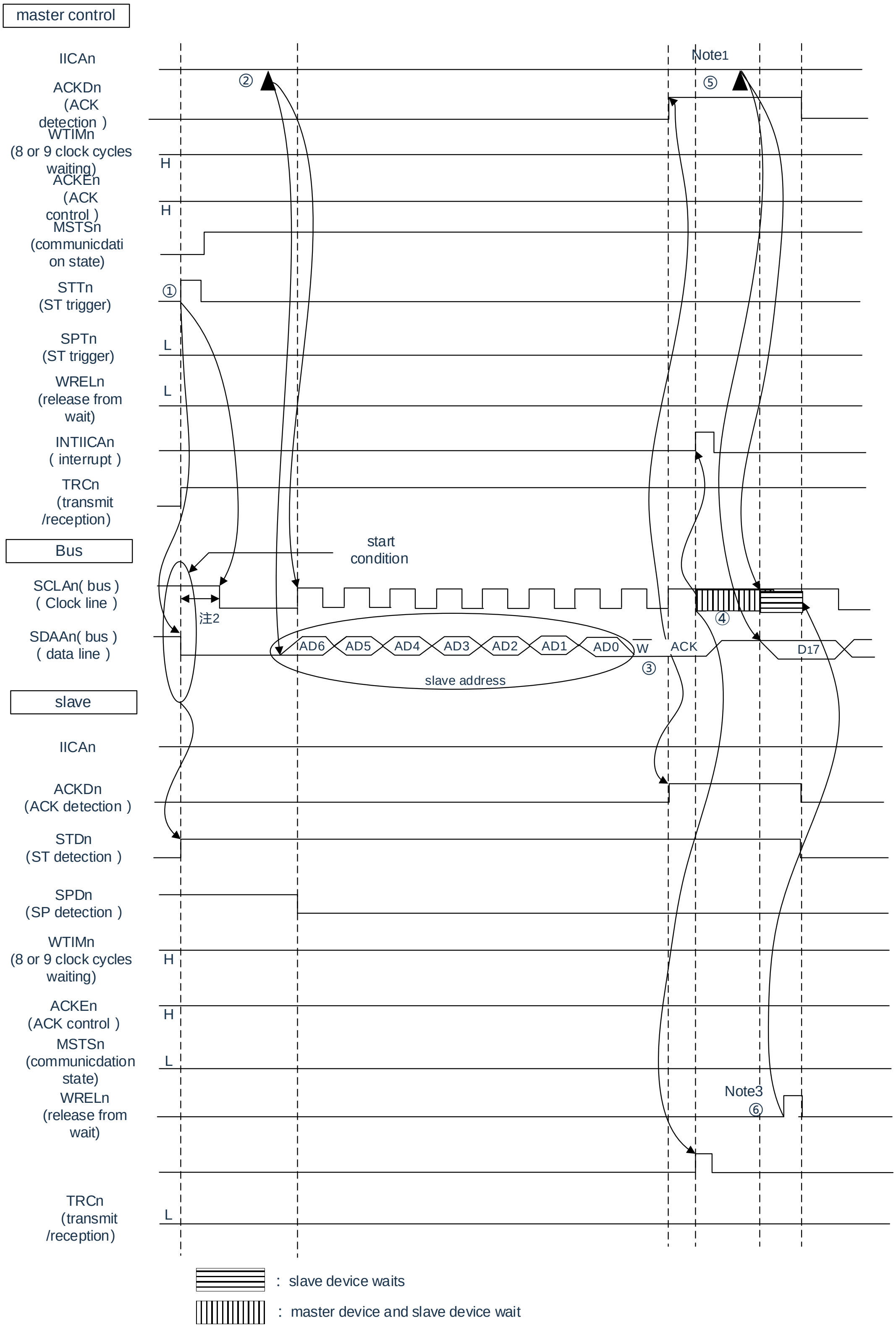
The IICA shift register n (IICAn) is synchronized with the falling edge of the serial clock (SCLAn) and the transmitted data is transferred to the SO latch to the MSB Data is output from the SDAAn pin first.

The data input to the SDAAn pin is fetched to IICAn on the rising edge of SCLAn.

Note n=0

Figure 14-31 Example of a slave device→a master device
(Master device: select 9 clocks of waiting, slave device: select 9 clocks of waiting) (1/4)

(1) Start condition ~ address ~ data


master control
IICAn
ACKDn (ACK detection)
WTIMn (8 or 9 clock cycles waiting)
ACKEn (ACK control)
MSTSn (communicdation state)
STTn (ST trigger)
SPTn (ST trigger)
WRELn (release from wait)
INTIICAn (interrupt)
TRCn (transmit /reception)
Bus
SCLAn(bus) (Clock line)
SDAAn(bus) (data line)
slave
IICAn
ACKDn (ACK detection)
STDn (ST detection)
SPDn (SP detection)
WTIMn (8 or 9 clock cycles waiting)
ACKEn (ACK control)
MSTSn (communicdation state)
WRELn (release from wait)
TRCn (transmit /reception)
H
L
①
②
③
④
⑤
⑥
Note1
Note3
注2
start condition
AD6
AD5
AD4
AD3
AD2
AD1
AD0
W
ACK
D17
slave address
▤ : slave device waits
▥ : master device and slave device wait

Note 1. To remove the wait during the master send, the IICAn must be written to the data instead of the set the WRELn bit.

2. The time from the SDAAn pin signal drop to the SCLAn pin signal drop is at least 4.0µs when set to standard mode and at least 0.6µs when set to fast mode.

3. To release the wait during the slave receive, the IICAn must be set to "FFH" or set the WRELn bit.

Figure 14-31 “(1) start condition ~ address ~ data” (1) ~ (6) descriptions are as follows:

(1) If the start condition is triggered by the master (STTn=1), the bus data line (SDAAn) drops, generating a start condition (changing SDAAn from “1” to “0” by SCLAn=1 “). Thereafter, if a start condition is detected, the master enters the master communication state (MSTSn=1), and the bus clock line drops (SCLAn=0) after the hold time elapses, ending the communication readiness.

(2) If the master parity writes address +W (send) to the IICA shift register n (IICAn), the slave address is sent.

(3) On the slave side, if the receiving address and the local station address (the value of SVAn) are the same, the ACK is sent to the master controller through the hardware. The master detected ACK (ACKDn=1) on the rising edge of the 9th clock.

(4) The master generates an interrupt on the falling edge of the 9th clock (INTIICAn: address send end interrupt). A slave of the same address enters a waiting state (SCLAn=0) and generates an interrupt (INTIICAn: Address Matching Interrupt) note.

(5) The master writes and sends data to the IICAn register, relieving the master of waiting.

(6) If the slave unwaits (WRELn=1), the master begins to transmit data to the slave.

Note If the sending address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master and does not generate an INTIICAn interrupt (address matching interrupt), nor does it enter a waiting state.

However, the main controller generates an INTIICAn interrupt (address send end interrupt) for both ACK and NACK.

Remark 1. Figure 14-31 ① to ⑮ shows a series of operational steps for data communication via the I^{2}C bus.

Figure 14-31 “(1) start condition ~ address ~ data” illustrates steps ①~⑥.

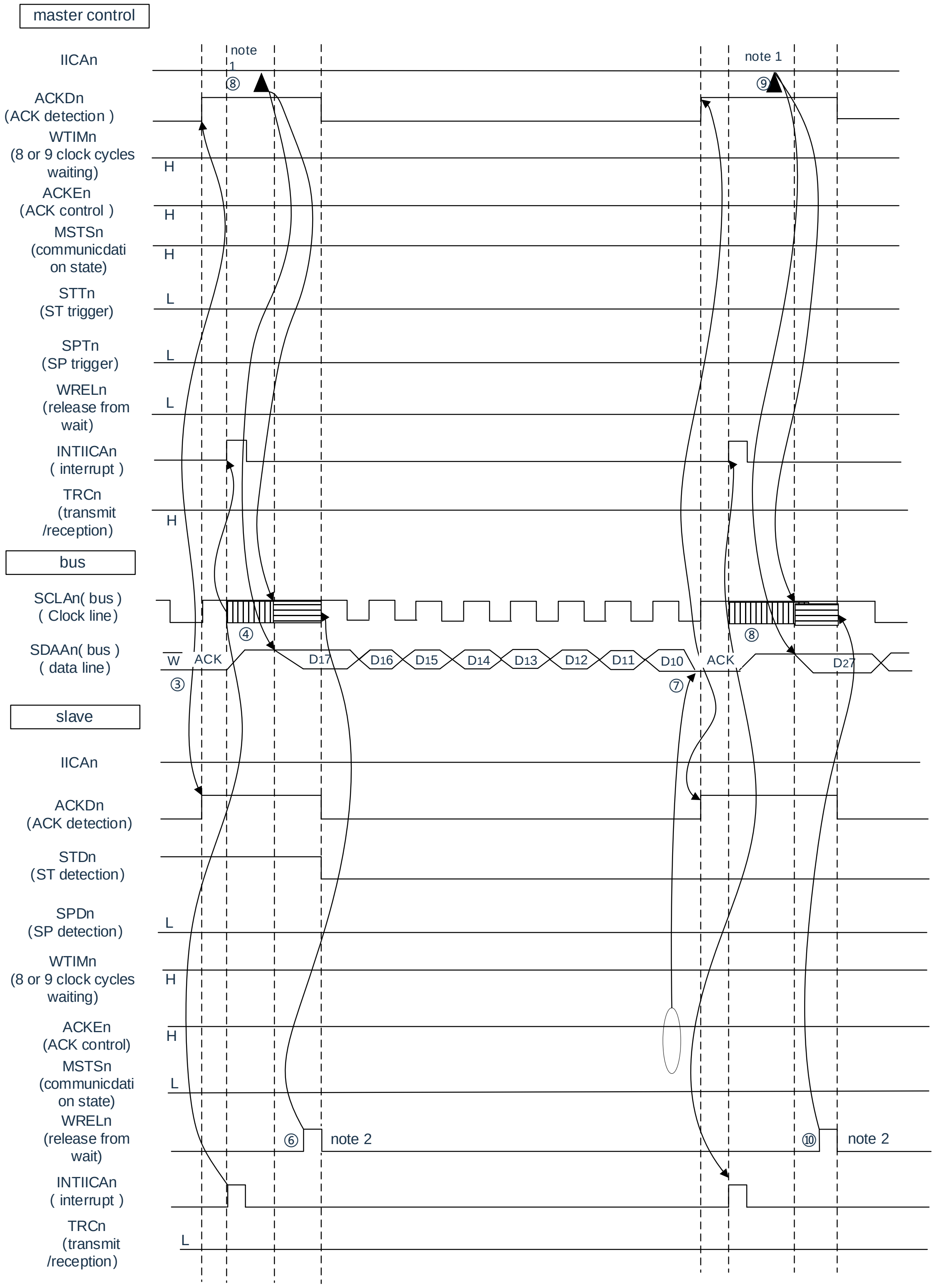
Figure 14-31 “(2) address ~ data ~ data” illustrates steps ③~⑩.

Figure 14-31 “(3) data ~ data ~ stop conditions” illustrates steps ⑦ ~ ⑮.

2. n=0

Figure 14-31 Communication example of a master device → slave device
(Master device: select 9 clocks of waiting, slave device: select 9 clocks of waiting) (2/4)

(2) Address ~ Data ~ Data



▤ : slave device waits

▥ : master device and slave device wait

Note 1. To release the master from waiting during transmit, 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 14-31 “(2) address ~ data ~ data” (3) ~ (10) of the descriptions are as follows:

(3) On the slave, if the receiving address and the local station address (the value of the SVAn) are the same note, the ACK is sent to the master through the hardware. The master detects ACK on the rising edge of the 9th clock (ACKDn=1).

(4) The master generates an interrupt on the falling edge of the 9th clock (INTIICAn: address send end interrupt). Slaves with the same address enter a waiting state (SCLAn=0) and an interrupt (INTIICAn: address matching interrupt) Note.

(5) The master writes the transmit data to the IICA shift register n (IICAn) to relieve the master 's wait.

(6) If the slave unwaits (WRELn=1), the master controller begins to transmit data to the slave.

(7) After the data transmission is completed, because the ACKEn bit of the slave party is “1”, the ACK is sent to the master controller through the hardware. The master detected ACK (ACKDn=1) on the rising edge of the 9th clock.

(8) Both the master and slave enter a waiting state (SCLAn=0) on the falling edge of the 9th clock, and both produce interrupts (INTIICAn: End of Transmission Interrupt).

(9) The main controller writes the transmitted data to the IICAn register and releases the main controller's wait.

(10) If the slave reads the received data and unwaits (WRELn=1), the master controller begins to transmit data to the slave.

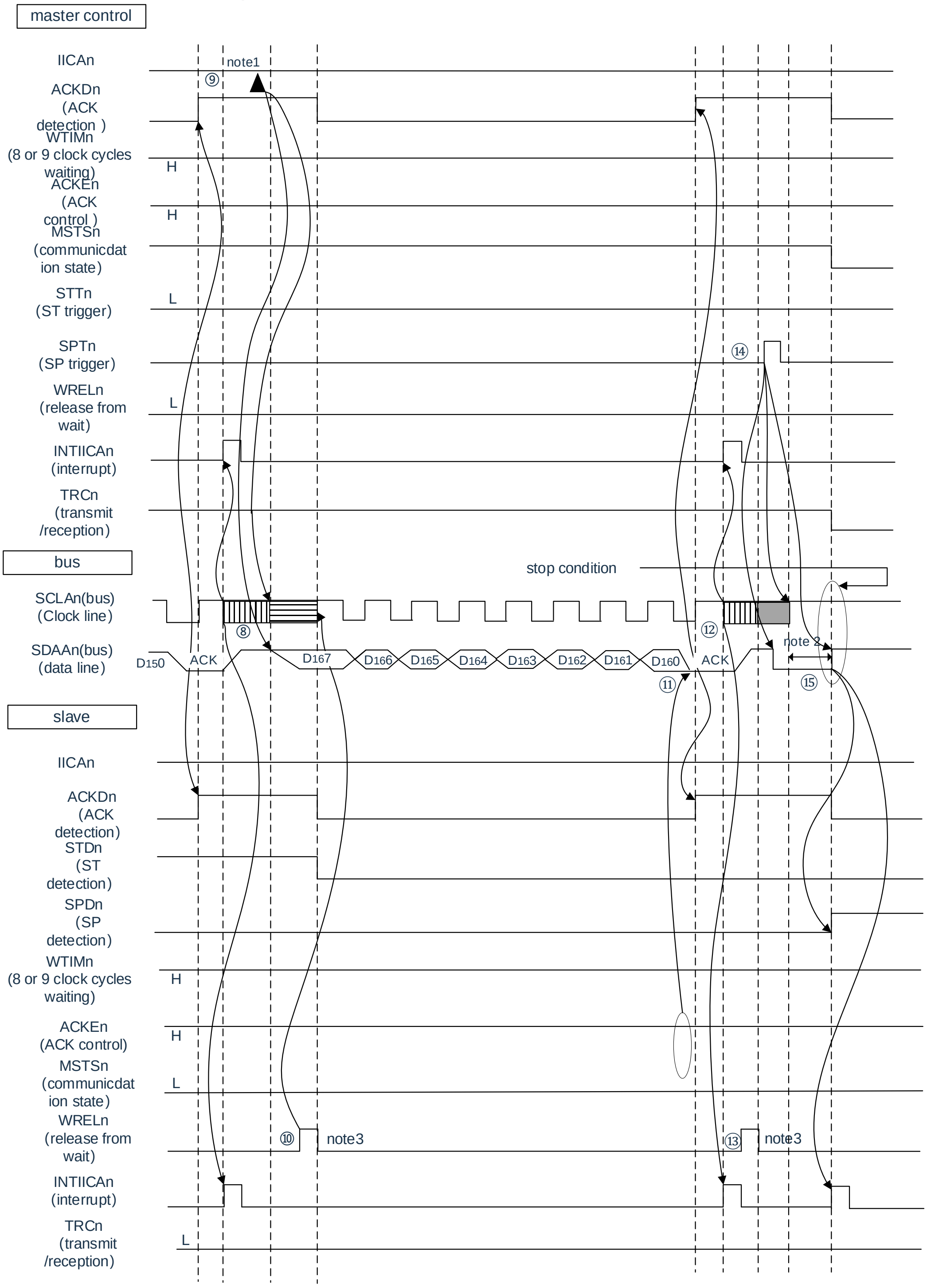
Note If the sending address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master and does not generate an INTIICAn interrupt (address matching interrupt), nor does it enter a waiting state.
However, the main controller generates an INTIICAn interrupt (address send end interrupt) for both ACK and NACK.

Note 1. Figure 14-31 ① to ⑮ shows a series of operational steps for data communication via the I2C bus.
Figure 14-31 “(1) start condition ~ address ~ data” illustrates steps (1) ~ (6).
Figure 14-31 “(2) address ~ data ~ data” illustrates steps (3) ~ (10).
Figure 14-31 “(3) data ~ data ~ stop conditions” illustrates steps (7) ~ (15)

2. n=0

Figure 14-31 Communication example of a master device → slave device
(Master device: select 9 clocks of waiting, slave device: select 9 clocks of waiting) (3/4)

(3) 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 transmit, you must write data to the IICAn instead of setting the WRELn bit.

2. The time from the SCLAn pin signal to the generation of the stop condition after the stop condition is issued is at least 4.0 u s when set to standard mode and at least 0.6us when set to fast mode.

3. To release the wait during the slave receive, the IICAn must be set to "FFH" or set the WRELn bit.

Figure 14-31The descriptions of (7) to (15) of "(3) Data - Data - Stop condition" in Figure 14-31are as follows:

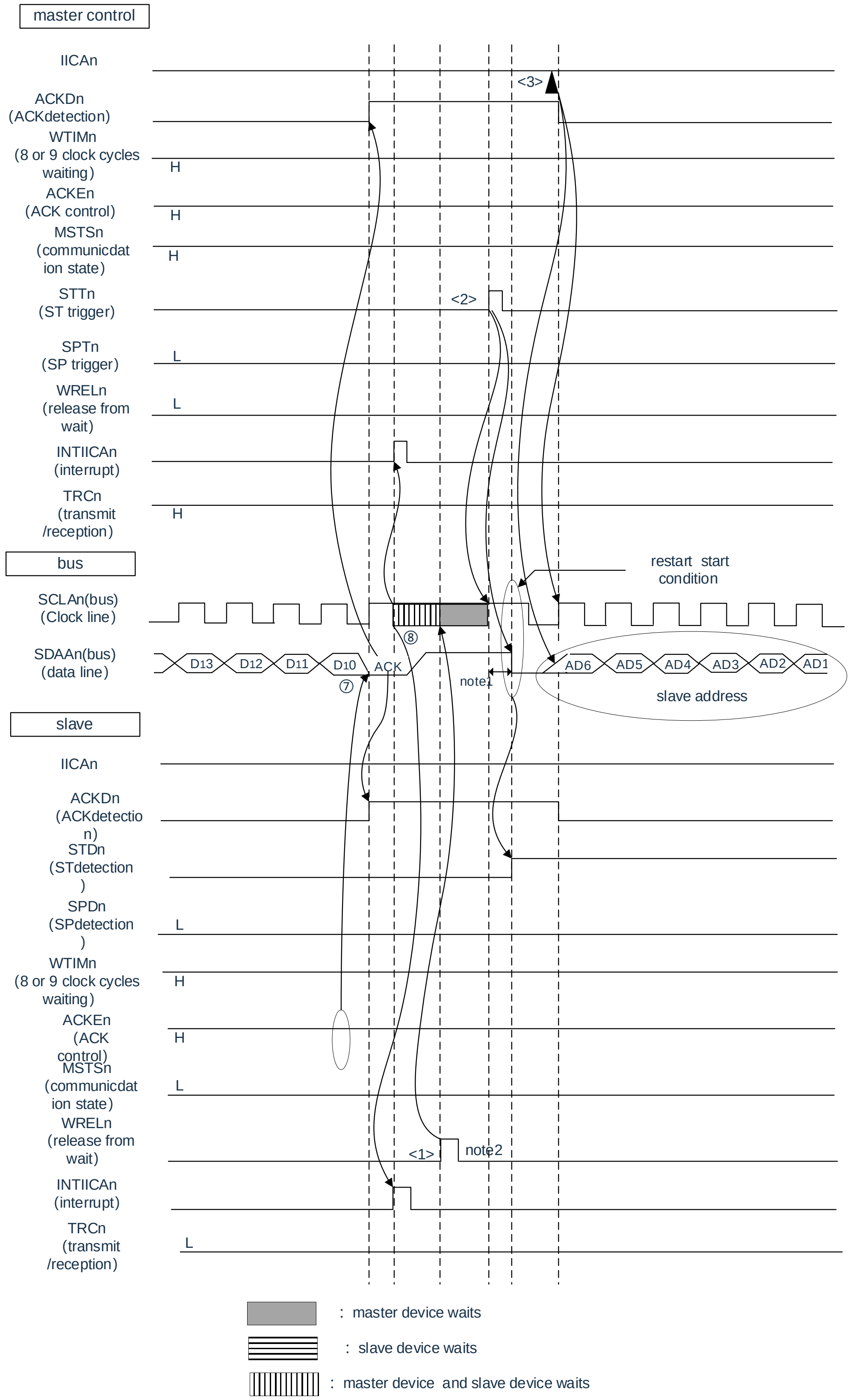
⑦ At the end of the data transfer, the ACK is sent to the master controller through the hardware because the ACKEn bit of the slave is “1”. The master detected ACK (ACKDn=1) on the rising edge of the 9th clock.
⑧ Both the master and slave enter a waiting state (SCLAn=0) on the falling edge of the 9th clock and both generate interrupts (INTIICAn: End of Transmission Interrupt).
⑨ The master transmits data to the IICA shift register n (IICAn) to relieve the master from waiting.
⑩ If the slave reads the received data and releases the wait (WRELn=1), the master controller begins to transmit the data to the slave.
⑪ At the end of the data transfer, the slave party (ACKEn=1) sends the ACK to the master controller through the hardware. The master detected ACK (ACKDn=1) on the rising edge of the 9th clock.
⑫ Both the master and slave enter a waiting state (SCLAn=0) on the falling edge of the 9th clock and both generate interrupts (INTIICAn: End of Transmission Interrupt).
⑬ The slave reads the received data and releases the wait (WRELn=1).
⑭ If the stop condition is triggered to assert (SPTn=1) in the master controller, the bus data line (SDAAn=0) is cleared and the bus clock line is asserted (SCLAn=1), and the bus data line is asserted (SDAAn=1) after the preparation time of the stop condition has passed. Generate a stop condition (change SDAAn from “0” to “1” by SCLAn=1).
⑮ If a stop condition is generated, the slave detects the stop condition and generates an interrupt (INTIICAn: Stop condition interrupt).

Note 1. Figure 14-31 ① to ⑮ shows a series of operational steps for data communication via the I^{2}C bus.
Figure 14-31 “(1) start condition ~ address ~ data” describes steps (1) ~ (6).
Figure 14-31 “(2) address ~ data ~ data” illustrates steps (3) ~ (10).
Figure 14-31 “(3) data ~ data ~ stop conditions” illustrates steps (7) ~ (15)

2.n=0

Figure 14-31 Communication example of a master device → slave device
(Master device: select 9 clocks of waiting, slave device: select 9 clocks of waiting) (4/4)

(4) Data~Restart condition~Address



Note 1 The time from the SCLAn pin signal to the generation of the start condition after the restart condition is released is at least 4.7 u s when set to standard mode and at least 0.6us when set to 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 14-31The operation of "(4) Data ~ Restart condition ~ Address" in Figure 14-31 is explained as follows. After executing steps ⑦ and ⑧, execute <1> to <3>, and return to the data sending step in step (3).

(7) At the end of the data transfer, an ACK is sent to the master via hardware because the slave's ACKEn bit is "1." the master detects the ACK on the rising edge of the 9th clock (ACKDn=1).

(8) Both the master and slave enter a waiting state (SCLAn=0) on the falling edge of the 9th clock, and both produce interrupts (INTIICAn: End of Transmission Interrupt).

<1> The slave reads the received data and releases the wait (WRELn=1).

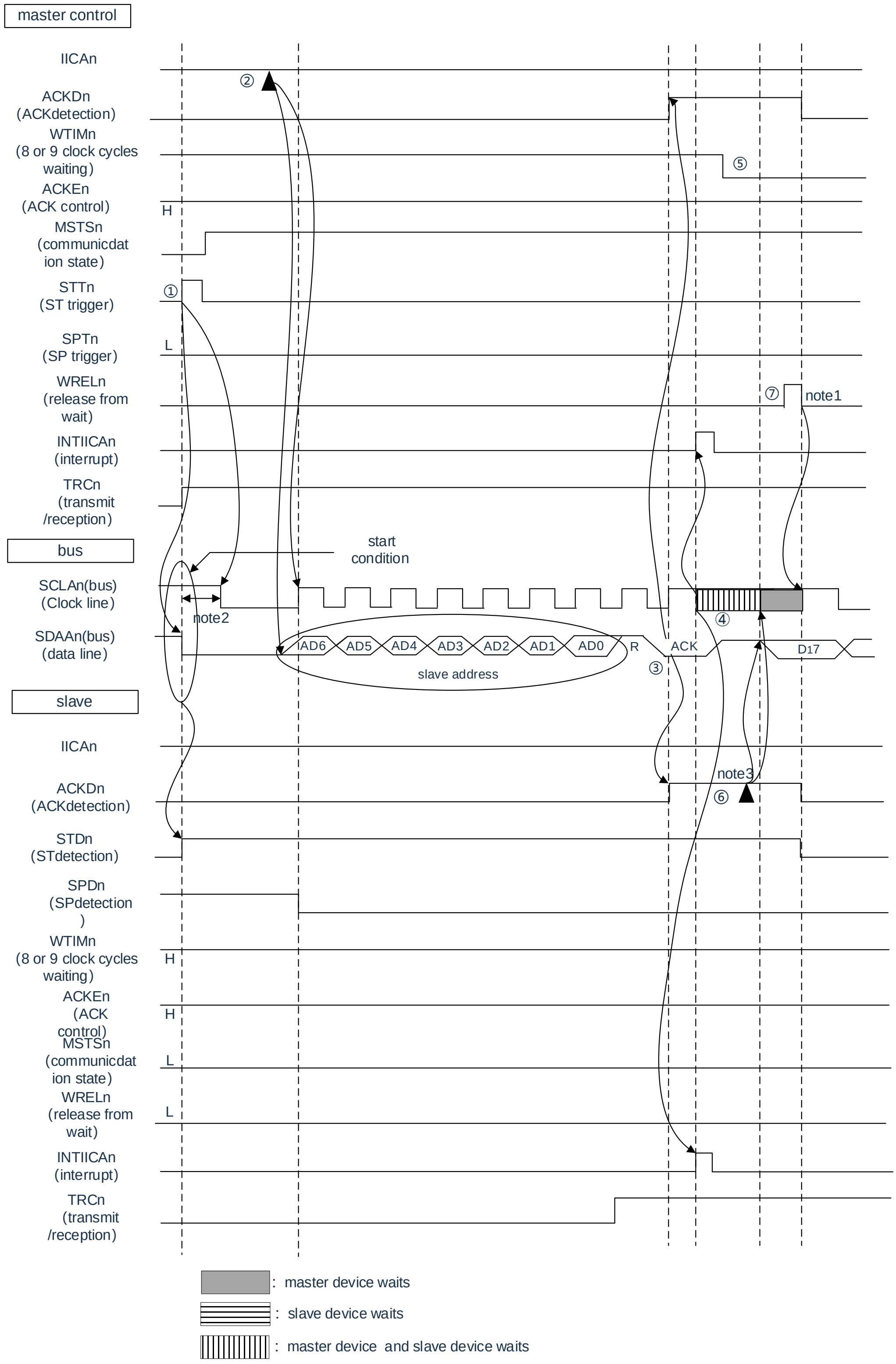
<2> If the start condition is triggered again (STTn=1) on the master controller, the bus clock line rises (SCLAn=1) and the bus data line drops (SDAAn=0) after the renewal start condition preparation time), generate a start condition (change SDAAn from “1” to “0” by SCLAn=1). Then, if a start condition is detected, the bus clock line drops (SCLAn=0) after the hold time elapses, and the communication is ready to be concluded.

<3> If the master writes the address +R/W (send) to the IICA shift register n (IICAn), the slave address is sent.

Note n=0

Figure 14-32 Communication example of a slave device→master device
(Master device: select 8 clocks of waiting, slave device: select 9 clocks of 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 from the SDAAn pin signal drop to the SCLAn pin signal drop is at least 4.0us when set to standard mode and at least 0.6us when set to fast mode.

3. To release the slave from waiting during transmission, you must write data to IICAn instead of setting the WRELn bit.

The descriptions of ① to ⑦ of "(1) Start condition ~ Address ~ Data" in Figure 14-32 are as follows:

(1) If the start condition is triggered by the master (STTn=1), the bus data line (SDAAn) drops, generating a start condition (changing SDAAn from “1” to “0” by SCLAn=1 “). Thereafter, if a start condition is detected, the master enters the master communication state (MSTSn=1), and the bus clock line drops (SCLAn=0) after the hold time elapses, ending the communication readiness.

(2) If the master party writes the address +R (receive) to the IICA shift register n (IICAn), the slave address is sent.

(3) On the slave, if the receiving address and the local station address (the value of SVAn) are the same, the ACK is sent to the master controller through the hardware. The master detected ACK (ACKDn=1) on the rising edge of the 9th clock.

(4) The master generates an interrupt on the falling edge of the 9th clock (INTIICAn: address send end interrupt). A slave of the same address enters a waiting state (SCLAn=0) and generates an interrupt (INTIICAn: Address Matching Interrupt) Note.

(5) The master changes the waiting timing to the 8th clock (WTIMn=0).

(6) The slave party writes the transmit data to the IICAn register to relieve the slave party of waiting.

(7) The master and controller release the wait (WRELn=1) and start the data transfer from the slave device.

Note If the sending address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master and does not generate an INTIICAn interrupt (address matching interrupt), nor does it enter a waiting state.

However, the main controller generates an INTIICAn interrupt (address send end interrupt) for both ACK and NACK.

Note 1. Figure 14-32 ①~⑲ shows a series of operation steps for data communication via the I2C bus.

Figure 14-32 “(1) start condition ~ address ~ data” describes steps (1) ~ (7).

Figure 14-32 “(2) address ~ data ~ data” illustrates steps (3) ~ (12).

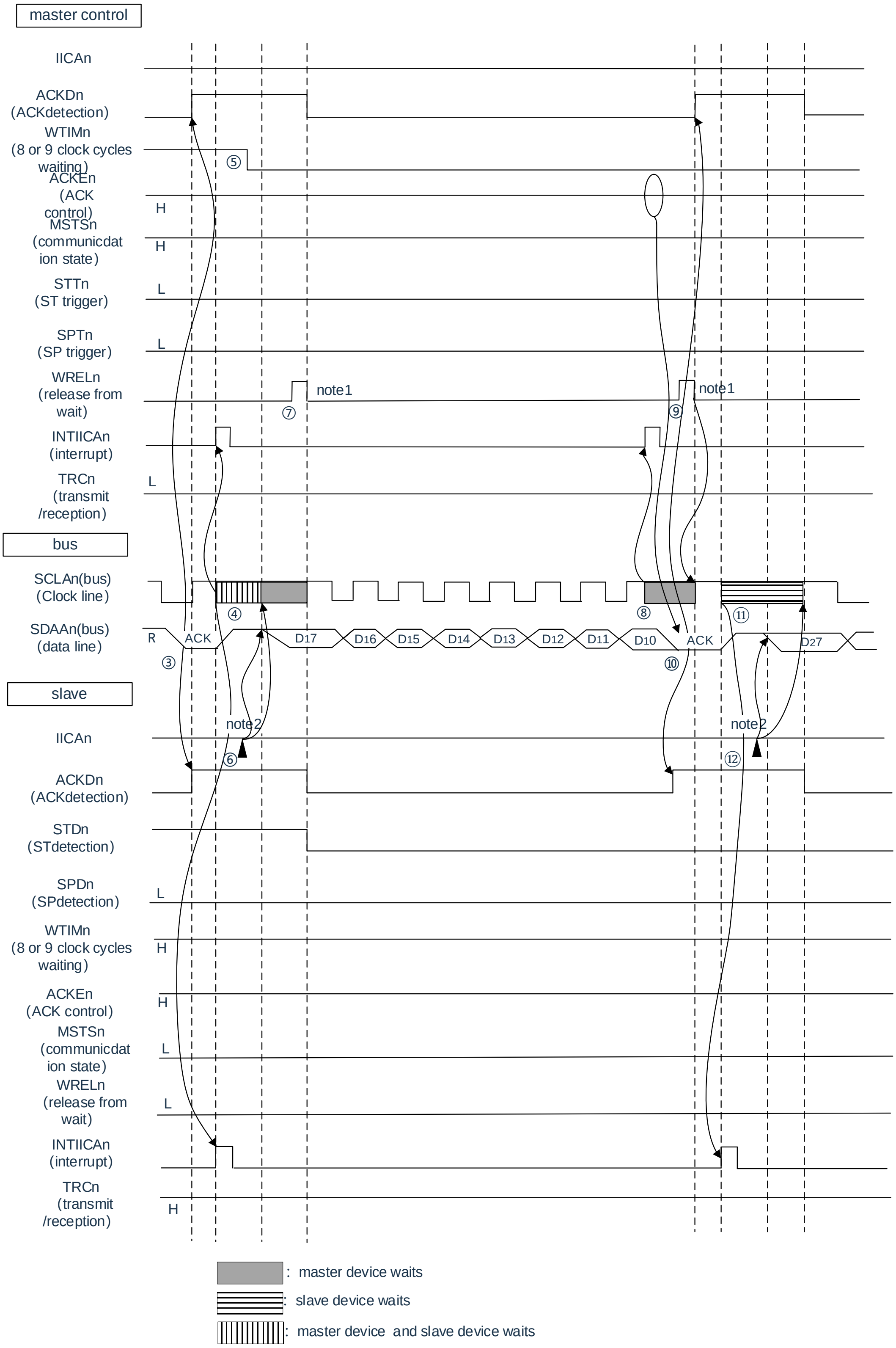
Figure 14-32 “(3) data ~ data ~ stop conditions” shows steps (8) ~ (19).

2. n=0

Figure 14-32 Communication example of a slave device→master device

(Master device: select 8 clocks of waiting, slave device: select 9 clocks of 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.

The description of ③ to ⑫ of "(2) Address - Data - Data" in Figure is as follows:

(3) On the slave side, if the receiving address and the local station address (the value of SVAn) are the same, the ACK is sent to the master controller through the hardware. The master detected ACK (ACKDn=1) on the rising edge of the 9th clock.

(4) The master generates an interrupt on the falling edge of the 9th clock (INTIICAn: address send end interrupt). A slave of the same address enters a waiting state (SCLAn=0) and generates an interrupt (INTIICAn: Address Matching Interrupt) note.

(5) The master changes the waiting timing to the 8th clock (WTIMn=0).

(6) The slave party writes the send data to the IICA shift register n (IICAn) to relieve the slave party of waiting.

(7) The master releases the wait (WRELn=1) and start the data transfer from the slave device.

(8) The master enters a waiting state (SCLAn=0) on the falling edge of the 8th clock and generates an interrupt (INTIICAn: Transmission end intermediate interruption). Because the ACKEn bit of the master controller is “1”, the ACK is sent to the slave through the hardware.

(9) The master reads the received data and releases the wait (WRELn=1).

(10) The slave detected an ACK (ACKDn=1) on the rising edge of the 9th clock.

(11) The slave enters the waiting state (SCLAn=0) on the falling edge of the 9th clock and generates an interrupt (INTIICAn: transmit end interrupt).

(12) If the slave writes and sends data to the IICAn register, the slave is relieved of the wait and the data transfer from the slave party to the main controller begins.

Note If the sending address and the slave address are different, the slave does not return an ACK (NACK: SDAAn=1) to the master and does not generate an INTIICAn interrupt (address matching interrupt), nor does it enter a waiting state.

However, the main controller generates an INTIICAn interrupt (address send end interrupt) for both ACK and NACK.

Note 1. Figure 14-32 ①~⑲ shows a series of operation steps for data communication via the I2C bus.

Figure 16-32 “(1) start condition ~ address ~ data” describes steps (1) ~ (7).

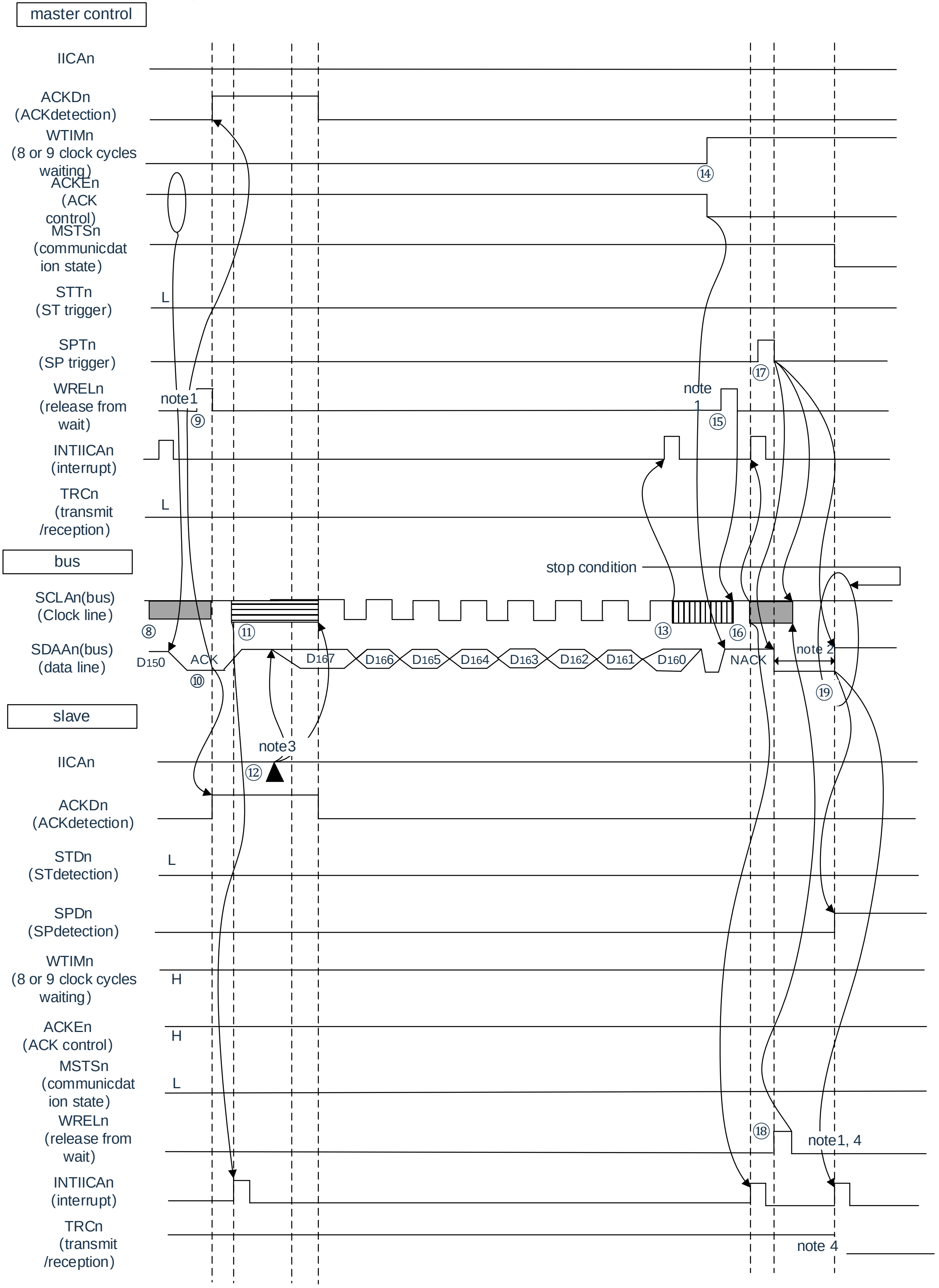
Figure 16-32 “(2) address ~ data ~ data” describes steps (3) ~ (12).

Figure 16-32 “(3) data ~ data ~ stop conditions” shows steps (8) ~ (19).

2. n=0

Figure 14-32 Communication example of a slave device→master device
(Master device: select 8 → 9 clocks waiting, slave device: select 9 clocks waiting) (3/3)

(3) Data ~ Data ~ Stop Condition



▨ : master device waits

☰ : slave device waits

▥ : master device and slave device waits

Note 1. To release the wait, the IICAn must be set to "FFH" or the WRELn bit must be set.

2. After the release of the stop condition, the time from the SCLAn pin signal to generate the stop condition is at least 4.0us when set to standard mode and at least 0.6us when set to fast mode.
3. To release the slave from waiting during transmit, the data must be written to the IICAn instead of the WRELn bit.
4. If the wait is released by setting the WRELn bit during the slave's transmit, the TRCn bit is cleared.

The description of ⑧~⑲ of "(3) Data - Data - Stop condition" in Fig. 16-32 is as follows:

⑧. The master enters a waiting state (SCLAn=0) on the falling edge of the 8th clock and generates an interrupt (INTIICAn: End of Transmission Neutral). Since the ACKEn bit of the master is “0”, the ACK is sent to the slave through the hardware.
⑨. The master receiver reads the received data and unwaits (WRELn=1).
⑩. The slave detected an ACK (ACKDn=1) on the rising edge of the 9th clock.
⑪. The slave enters a waiting state (SCLAn=0) on the falling edge of the 9th clock and generates an interrupt (INTIICAn: transmit end interrupt).
⑫. If the slave writes the send data to the IICA shift register n (IICAn), the slave is relieved of the wait and the data transfer from the slave to the master controller begins.
⑬. The master generates an interrupt on the falling edge of the 8th clock (INTIICAn: transmit end interrupt) and enters a waiting state (SCLAn=0). Because of the ACK control (ACKEn=1), the bus data line at this stage becomes low (SDAAn=0).
⑭. The master sets the NACK Acknolwdge (ACKEn=0) and changes the wait sequence to the 9th clock (WTIMn=1). If the master relieshes the wait (WRELn=1), the slave detects THEACK (ACKDn=0) on the rising edge of the 9th clock.
⑮. Both the master and slave enter a waiting state (SCLAn=0) on the falling edge of the 9th clock and both generate interrupts (INTIICAn: End of Transmission Interrupt).
⑯. If the master issues a stop condition (SPTn=1), the bus data cable (SDAAn=0) is cleared and the master's wait is released. Thereafter, the master is in standby until the bus clock line is asserted (SCLAn=1).
⑰. The slave stops sending after acknowledging the NACK, and in order to end the communication, the wait is released (WRELn=1). If the slave is relieved of waiting, the bus clock line is set (SCLAn=1).
⑱. If the master confirms that the bus clock line is being set (SCLAn=1), the bus data line is set after the stop condition preparation time has elapsed
⑲. (SDAAn=1), and then issue a stop condition (change SDAAn from “0” to “1” by SCLAn=1). If a stop condition is generated, the slave detects the stop condition and generates an interrupt (INTIICAn: Stop condition interrupt).

Chapter 15 IrDA

IrDA enables the transmission and reception of IrDA communication waveforms in accordance with the IrDA (InfraredDataAssociation) 1.0 protocol in cooperation with the Universal Serial Communication Unit (SCI).

15.1 Function of IrDA

If the IrDA function is set to active through the IRE bit of the IRCR register, SCI's TxD2 signal and RxD2 signal can encode or decode the waveform that conforms to the IrDA1.0 protocol (IrTxD/IrRxD pins), and then implement infrared transmission and reception that supports the IrDA1.0 protocol by connecting the transmitter or receiver that transmits/receives infrared rays.

In systems that support the IrDA1.0 protocol, after communication begins at a transfer rate of 9600bps, the transfer rate can be changed as needed. IrDA does not have a built-in function to automatically change the transfer rate, so the settings must be changed by software to change the transfer rate.

When selecting a high-speed internal oscillator (f_{IH}=24, 12, 6, 3MHz), the following baud rates can be set.

- 115.2kbps, 57.6kbps, 38.4kbps, 19.2kbps, 9600bps, 2400bps

A schematic block diagram of the collaboration between IrDA and SCI is shown in Figure 15-1.

Figure 15-1 Block diagram of the cooperation between IrDA and SCI

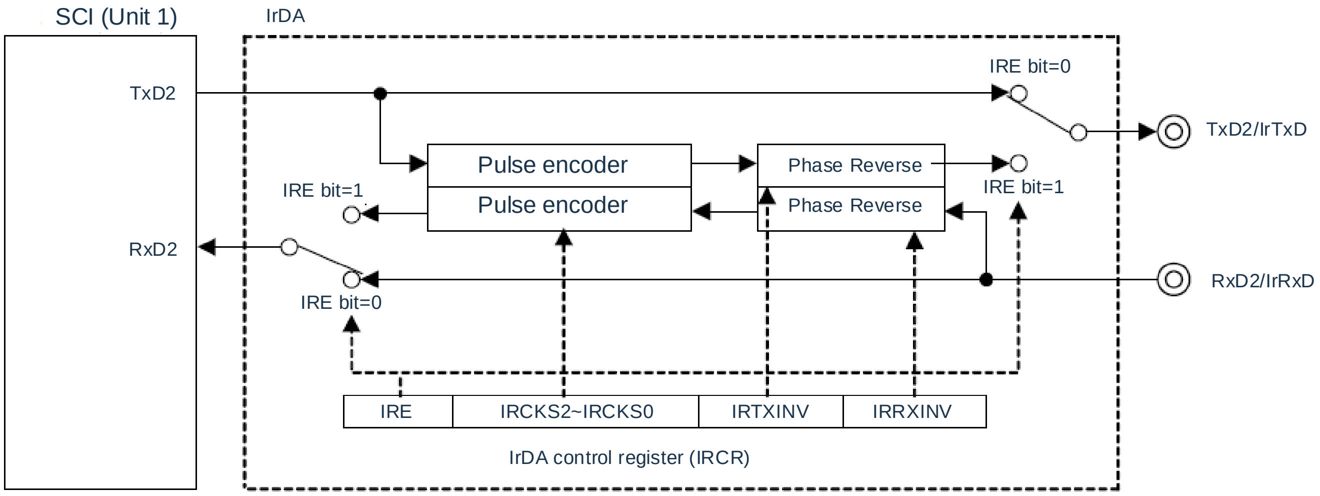


Table 15-1 Pin structure of the IrDA

Pin name	Input/output	Function
IrTxD	output	The output pin that sends the data
IrRxD	input	The input pin that receives the data

15.2 Registers for controlling the IrDA

Control the IrDA function through the following registers.

- Peripheral enable register 0 (PER0).
- IrDA control register (IRCR).

15.2.1 Peripheral enable register 0 (PER0)

The PER0 register is a register that sets the clock to be enable or disable to be supplied to each peripheral hardware. Reduce power consumption and noise by stopping clocking unused hardware.

To use IrDA, you must set bit6 (IRDAEN) to “1”.

The PER0 register is set via an 8-bit memory operation command.

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

Figure 15-2 Peripheral enable register 0 (PER0)

Address: 40020 420H After reset: 00HR/W

symbol	7	6	5	4	3			0
PER0	RTCEN	IRDAANDN	ADCIN	IICAEN	SCI1IN	SCI0EN	TM41EN	TM40EN

IRDAEN	Control of the input clock of the IrDA
0	Stop supplying the input clock. • You cannot write SFR used by IrDA. • IrDA is in a reset state.
1	An input clock is provided. • SFR used by IrDA can be read and written.

Note 1 When setting the IrDA, the IRDAEN bit must be set to "1" first. When the IRDAEN bit is "0", the write operation of the IrDA control register is ignored, and the read value is all initial.

15.2.2 IrDA control register (IRCR)

This is the register that controls the IrDA function. Selects for polarity switching of received and transmitted data, clock selection for IrDA, and switching of serial input/output pin functions (typically serial and IrDA functions). The IRCR register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register becomes “00H”.

Figure 15-3 Format of IrDA Control Register (IRCR)

Address: 40044000H After reset: 00HR/W

symbol	7	6	5	4	3	2	1	0
IRCR	IRE	IRCKS2	IRCKS1	IRCKS0	IRTXINV	IRRXINV	0	0

IRE	IrDA enable
0	Serial input/output pins are used as the usual serial function
1	The serial input/output pins are used as IrDA functions

IRCKS2	IRCKS1	IRCKS0	Clock selection for IrDA
0	0	0	B3/16 (B = bit rate).
0	0	1	$f_{CLK}/2$
0	1	0	$f_{CLK}/4$
0	1	1	$f_{CLK}/8$
1	0	0	$f_{CLK}/16$
1	0	1	$f_{CLK}/32$
1	1	0	$f_{CLK}/64$
1	1	1	Disable settings

IRTXINV	Polarity switching of IrTxD data
0	IrTxD output of the transmitted data
1	Reverse the data sent for IrTxD output

IRRXINV	Polarity switching of IrRxD data
0	Use the input data from the IrRxD pin as the receive data
1	The data after inverting the input data of the IrRxD pin is used as the received data

Note 1 You must set bit1 and bit0 to “0”.

2. IRCKS [2:0] bits, IRTXINV bits, and IRRXINV bits can be set only when the IRE bit is “0”.

15.3 Operation of IrDA

15.3.1 Operating steps for IrDA communication

(1) Initial setup process for IrDA communication

Follow the steps below to initialize IrDA.

1. Set the IRDAEN bit of the PER0 register to "1".
2. Set the IRCR register.
3. Set the relevant registers for SCI (refer to the setting steps of the UART mode).

(2) Stop process for IrDA communication

1. Set the IrTxD pin state after IrDA communication stops by setting the port register and the port mode register.

Note: When performing an IrDA reset via step 3, the IrTxD pin may change the output state by switching to the data output of the usual serial interface UART.

- Output low level from the IrTxD pin

Set the port register to “0”. Immediately after this setting, the IrTxD pin is fixed low.

- Output high level from the IrTxD pin

Set the port register to “1”. With this setting, the IrTxD pin is fixed to high immediately after the IrDA reset in step 3.

- Set the IrTxD pin to the Hi-Z state to set the port mode register to “1”. Immediately after this setting, the IrTxD pin changes to the Hi-Z state.

2. Set the STm0 and STm1 bits of the STm register (SCI's associated register) to "1" (to stop the operation of SCI's channel 0 and channel 1).
3. Reset IrDA by setting IRDAEN to "0" in PER0 register.

You cannot set the STm0 and STm1 bits of the STm register to "1" or the IRE bit of the IrDA to "0" in cases other than the above steps.

(3) To send an IrDA frame error

When a frame error occurs during IrDA communication, the following settings must be made in order to set the state in which subsequent data can be received.

1. Set the STm1 bit of SCI's STm register to "1" (to stop the operation of SCI's channel 1).
2. Set the SSm1 bit of the SSm register of SCI to "1" (start the operation of channel 1 of SCI).

Remark m: Unit number (m=0).

For frame error handling for SCI, refer to Chapter 14, Universal Serial Communication Unit.

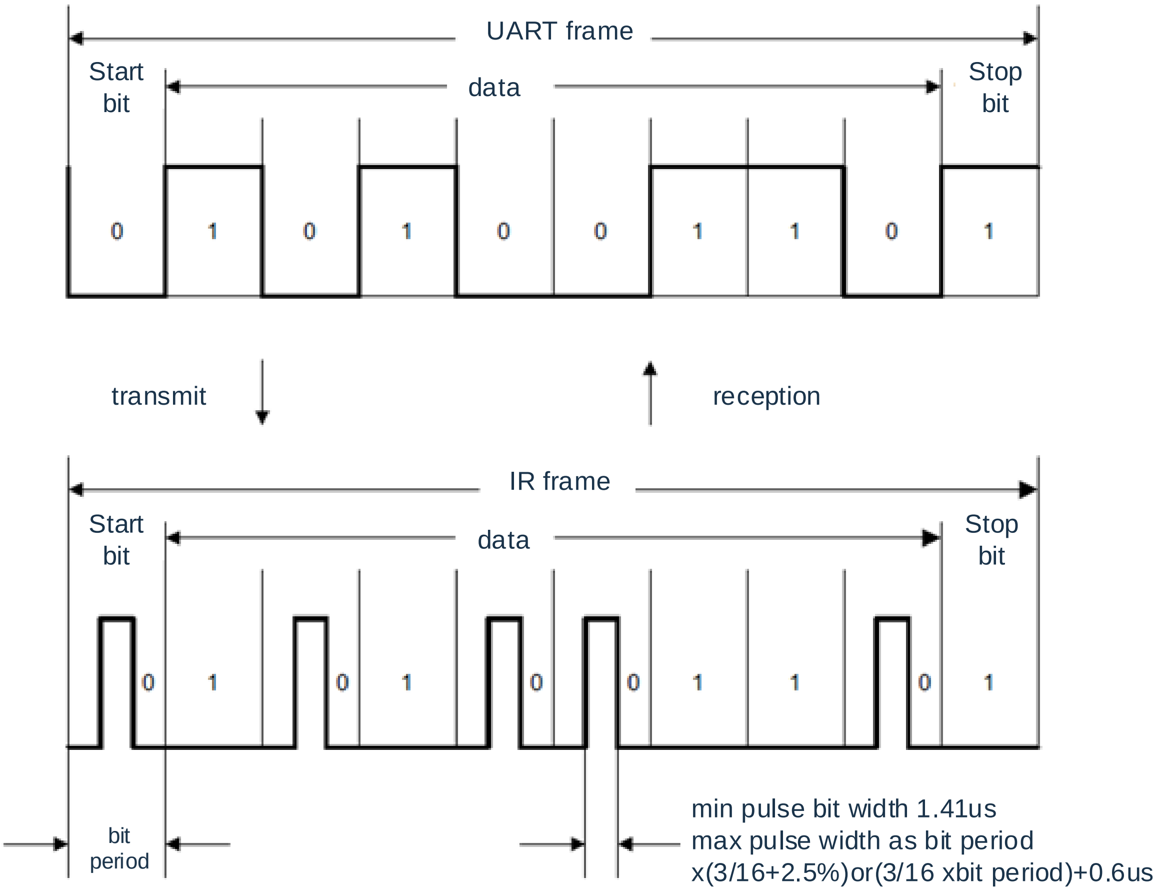
15.3.2 Transmission

At the time of transmission, the output signal (UART frame) from the SCI is converted to an IR frame via IrDA (see Figure 15-4).

At IRTXINV bit “0” and serial data is “0”, the output bit period (1-bit width period) x 3/16 high level pulse (initial value). In addition, the high pulse width can be changed according to the setting value of IRCKS2 to IRCKS0 bits. As standard, a minimum pulse width of 1.41 u s for high levels is specified for a maximum of (3/16+2.5%) x bit period, or (3/16 x bit period) of +0.6 us.

When the CPU or peripheral hardware clock (fCLK) is 24MHz, the minimum high pulse width that can be set is 1.5 u s (the condition that the high-level pulse width specified above is not less than 1.41us is satisfied). In addition, when the serial data is “1”, no pulse is output.

Figure 15-4 Transmit/receive operation diagram of IrDA



15.3.3 Reception

When received, the data of the IR frame is converted to a UART frame via IrDA and then entered into the SCI. When the IRRXINV bit is “0” and a high pulse is detected, low data is output. If there is no pulse within the 1-bit period, high level data is output. Care must be taken that pulses smaller than the minimum pulse width of 1.41us cannot be recognized.

15.3.4 High level pulse width selection

If the pulse width at the time of transmission is less than the bit rate x 3/16, the applicable IRCKS2 ~ IRCKS0 bit setting (minimum pulse width) and the high-level pulse width at the time of setting are shown in Table 15-2.

Table 15-2 IRCKS2 ~ IRCKS0 bits

fCLK [MHz]	project	<Upper> Bit Rate [kbps] <Lower >Bit Rate x 3/16[us]					
		2.4	9.6	19.2	38.4	57.6	115.2
		78.13	19.53	9.77	4.87	3.26	1.63
1	IRCKS2~IRCKS0	001	001	001	- Note 1	- Note 1	- Note 1
	High level pulse width [μs].	2.00	2.00	2.00	- Note 1	- Note 1	- Note 1
2	IRCKS2~IRCKS0	010	010	010	010	010	- Note 1
	High level pulse width [μs].	2.00	2.00	2.00	2.00	2.00	- Note 1
3	IRCKS2~IRCKS0	011	011	011	011	011	- Note 1
	High level pulse width [μs].	2.67	2.67	2.67	2.67	2.67	- Note 1
4	IRCKS2~IRCKS0	011	011	011	011	011	000 Note 2
	High level pulse width [μs].	2.00	2.00	2.00	2.00	2.00	1.50
6	IRCKS2~IRCKS0	100	100	100	100	100	000 Note 2
	High level pulse width [μs].	2.67	2.67	2.67	2.67	2.67	1.50
8	IRCKS2~IRCKS0	100	100	100	100	100	000 Note 2
	High level pulse width [μs].	2.00	2.00	2.00	2.00	2.00	1.50
12	IRCKS2~IRCKS0	101	101	101	101	101	000 Note 2
	High level pulse width [μs].	2.67	2.67	2.67	2.67	2.67	1.50
16	IRCKS2~IRCKS0	101	101	101	101	101	000 Note 2
	High level pulse width [μs].	2.00	2.00	2.00	2.00	2.00	1.50
24	IRCKS2~IRCKS0	110	110	110	110	110	000 Note 2
	High level pulse width [μs].	2.67	2.67	2.67	2.67	2.67	1.50

Note 1.” -” indicates that the communication standard is not met.

2. The pulse width cannot be less than the bit rate 3/16.

15.4 Considerations when using IrDA

1. The operating clock of the IrDA can enable or disable by a peripheral enable register setting. The initial state is to disable the supply of clocks, so the registers cannot be accessed. Before the registers can be set, the peripheral allow registers must be set to allow the state of the IrDA operating clock to be provided.
2. In sleep mode, the IrDA function is continuously operated.
3. During IrDA communication, the initialization function of SCI (SS bit = 1) is prohibited.
4. The IRRXINV bit, IRTXINV bit, and IRCKS [2:0] bit of the IRCR register can be set only when the IRE bit is “0”.

Chapter 16 Enhanced DMA

16.1 The function of DMA

DMA is a function that does not use a CPU and transfers data between memories. Initiate DMA for data transfer via peripheral function interrupts. When DMA and CPU access the same unit in FLASH, SRAM0, SRAM1, or peripheral modules at the same time, their bus usage rights are higher than those of the CPU. When DMA and CPU access different units in FLASH, SRAM0, SRAM1, or peripheral modules, respectively, the two do not interfere with each other and can be executed in parallel.

The specifications of DMA are shown in Table 16-1.

Table 16-1 DMA specification (1/2)

Item		Specification
Start the source		Up to 24 boot sources
Distributable control data		24 groups
The address space that can be transferred	Address space	Full address range space
	source	Full address range space is optional
	target	Full address range space is optional
The maximum number of transfers	Normal mode	65535 times
	Repeating pattern	65535 times
The maximum transfer block size	Normal mode (8-bit transfer).	65535 bytes
	Normal mode (16-bit transmission).	131070 bytes
	Normal mode (32-bit transfer).	262140 bytes
	Repeating pattern	65535 bytes
Transmission units		8-bit/16-bit/32-bit
Transfer mode	Normal mode	Ends after transferring the DMACTj register from “1” to “0”.
	Repeating pattern	At the end of the transfer of the DMACTj register from “1” to “0”, the address of the duplicate area is initialized before the DMRLDj is placed The value of the register is reloaded into the DMACTj register and then transferred.
Address control	Normal mode	Fixed or incremental
	Repeating pattern	Fixed or incremented distinct addresses.
The priority of the startup source		Refer to Table 16-5 DMA startup source and vector address

Table 16-1 DMA specification (2/2)

item		Specification
Interrupt the request	Normal mode	When transferring the DMACTj register from “1” to “0”, an interrupt from the startup source is requested to the CPU and interrupt handling is performed.
	Repeating pattern	The RPTINT bit of the DMACRj register is “1” to allow interrupts to be generated) and the DMACTj register is transferred from “1” to “0” when the data transfer is made The CPU requests an interrupt from the start source and performs interrupt handling.
The transfer starts		If the DMAENi0~DMAENi7 bits of the DMAENi register is "1" (boot allowed), the transmission of data begins each time the DMA boot source occurs.
Delivery stopped	Normal mode	• Set DMAENi0~DMAENi7 bits to "0" (boot is prohibited). • When the DMACTj register changes from “1” to “0” at the end of the data transfer
	Repeating pattern	• Set DMAENi0~DMAENi7 bits to "0" (boot is prohibited). • When the RPTINT bit is “1” (allows interrupts to occur) and the DMACTj register changes from “1” to “0” at the end of the data transfer

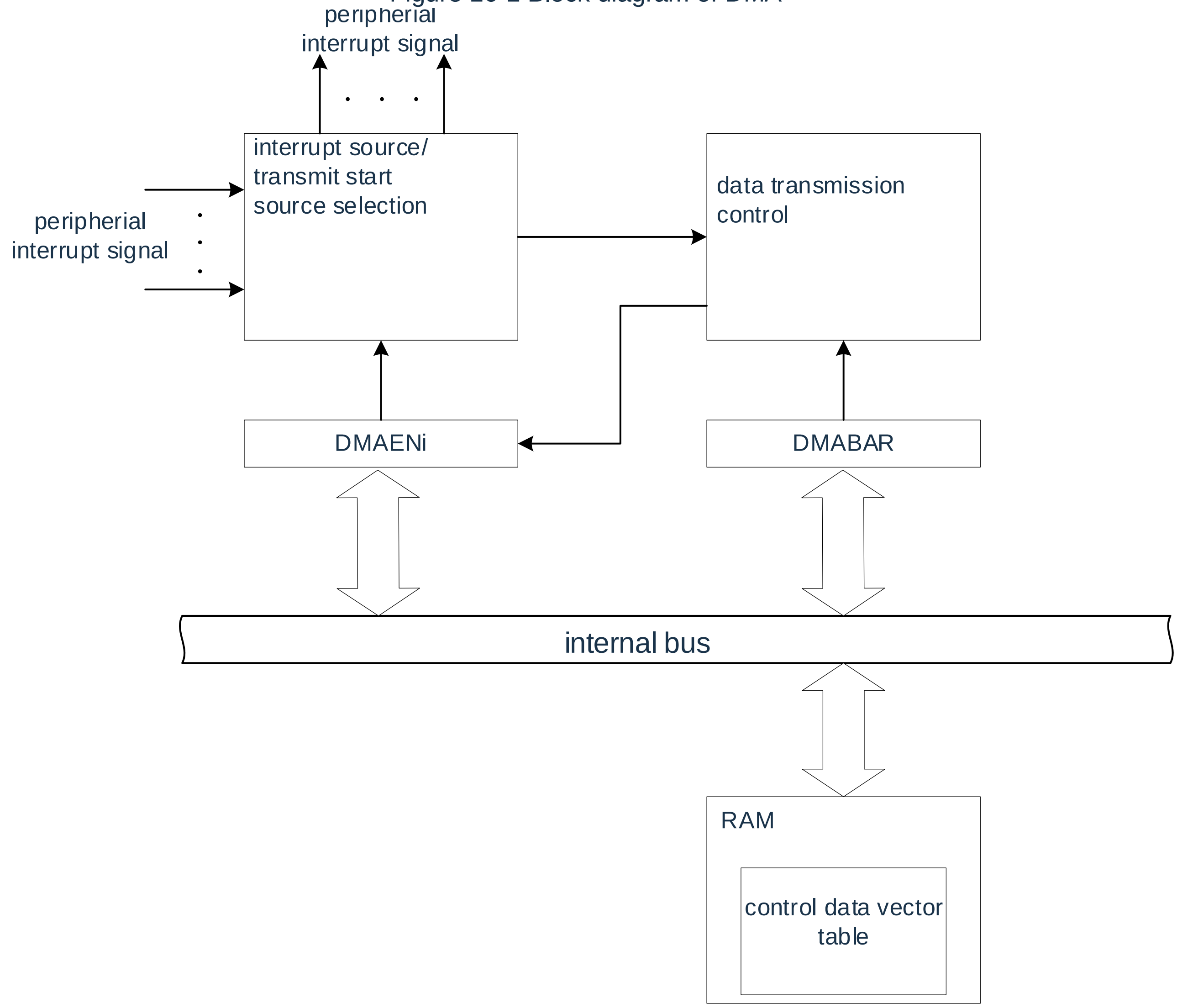
Note In deep sleep mode because the flash memory stops functioning and therefore cannot be used as a DMA transfer source.

Note i=0~2, j=0~23

16.2 Structure of DMA

The block diagram of DMA is Figure 16-1

Figure 16-1 Block diagram of DMA



16.3 Registers for controlling DMA

The registers that control the DMA are shown in Table 16-2.

Table 16-2 Registers for controlling DMA

Register Name	Symbol
Peripheral enable register 1	PER1
DMA boot enable register 0	DMAEN0
DMA boot enable register 1	DMAEN1
DMA boot enable register 2	DMAEN2
DMA base address register	DMABAR

The control data of the DMA is shown in Table 16-3.

The DMA control data is distributed in the DMA control data area of the RAM. The DMA control data area and the 416-byte region containing the DMA vector table area (the starting address where the control data is saved) are set via the DMABAR register.

Table 16-3 Control data for DMA

Register name	Symbol
DMA control register j	DMACRj
DMA block size register j	DMBLSj
DMA transfer count register j	DMACTj
DMA transfer number of times to reload register j	DMRLDj
DMA source address register j	DMSARj
DMA destination address register j	DMDARj

Note j=0~23

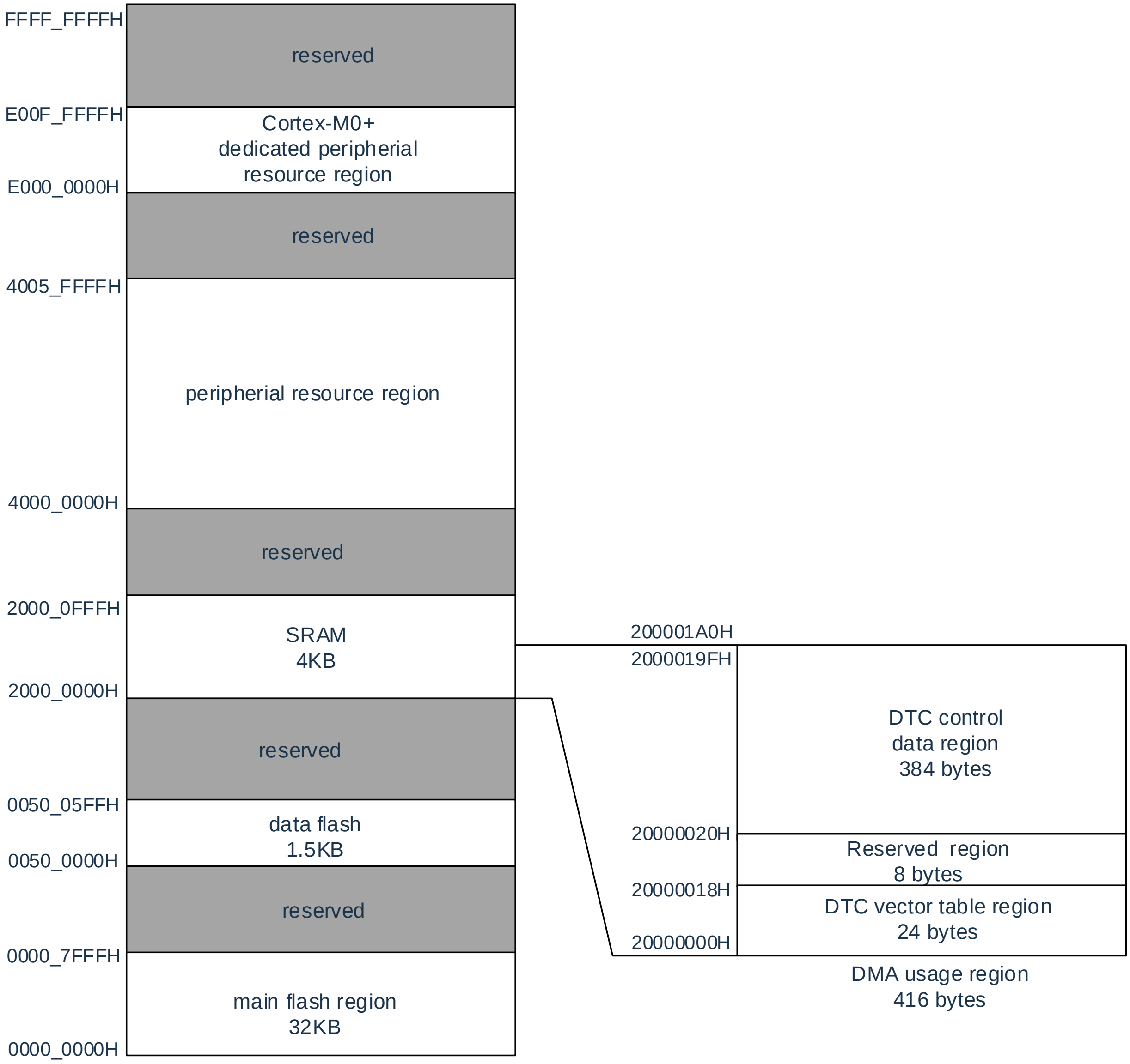
16.3.1 DMA control data areas and DMA vector table areas allocation

The control data allocated to the DMA and the 416-byte region of the vector table are set to the RAM area via the DMABAR register.

An example of a memory image with a DMABAR register set to “20000000H” is shown in Figure 16-2.

The 384 bytes of DMA control data area in the DMA unused space can be used as RAM.

Figure 16-2 Example of memory image when the DMABAR register is set to “20000000H”



16.3.2 Control data allocation

Starting from the start address, follow DMACRj, DMBLSj, DMACTj, DMRLDj, DMSARj, DMDARj (j=0~23) registers are assigned control data sequentially.

The start address is set by the DMABAR register, and the lower 10 bits are set separately by the vector table assigned by each startup source.

The distribution of control data is shown in Figure 16-3.

Note 1 The DMAENi0~DMAENi7 bits must be "0" in the corresponding DMAENi (i=0~2). (Disable Startup) when changing DMACRj, DMBLSj, DMACTj, DMRLDj, DMSARj, DMSARj, Data for the DMDARj register.

2. DMACRj, DMBLSj, DMACTj, DMRLDj, DMSARj and DMA cannot be transmitted via DMA Access to DMDARj.

Figure 16-3 Control data allocation (DMABAR set to 2000000H)

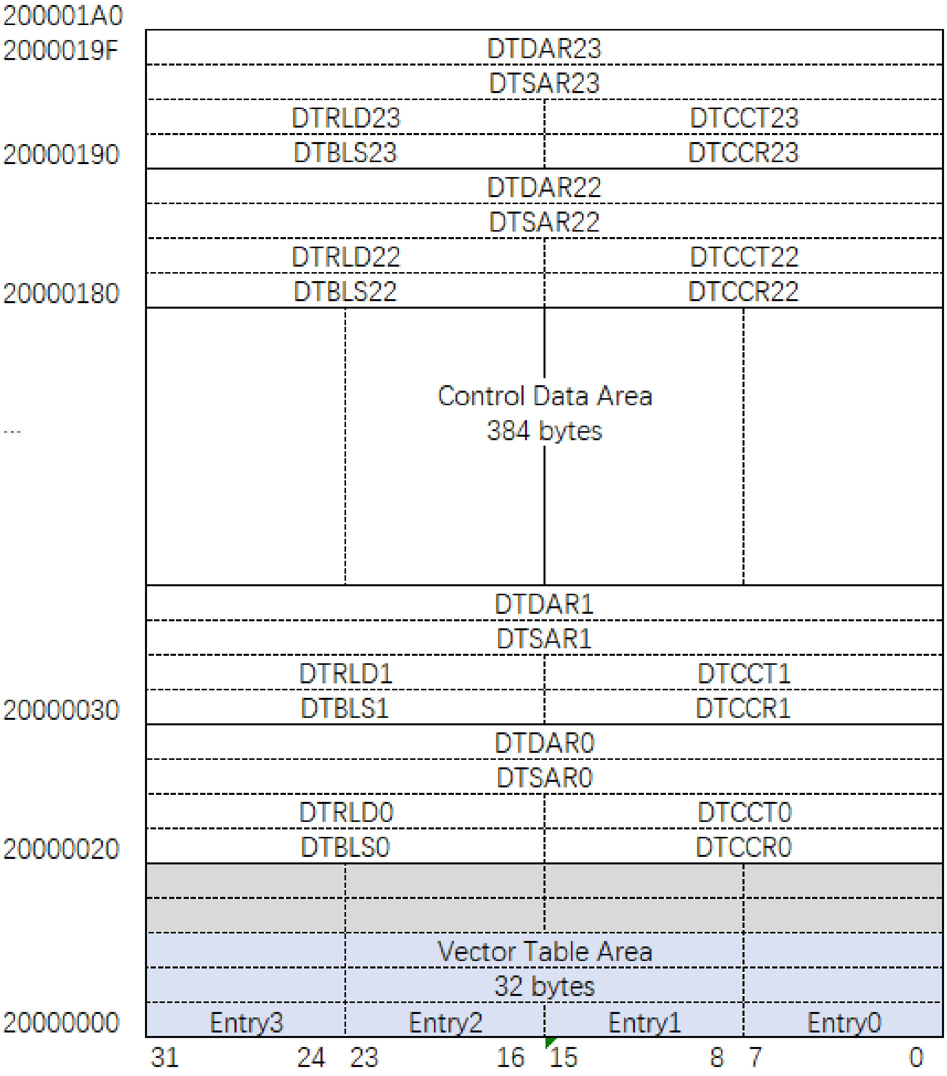


Table 16-4 Starting address of control data

j	address		j	address
11	baseaddr+D0H		23	baseaddr+190H
10	baseaddr+C0H		22	baseaddr+180H
9	baseaddr+B0H		21	baseaddr+170H
8	baseaddr+A0H		20	baseaddr+160H
7	baseaddr+90H		19	baseaddr+150H
6	baseaddr+80H		18	baseaddr+140H
5	baseaddr+70H		17	baseaddr+130H
4	baseaddr+60H		16	baseaddr+120H
3	baseaddr+50H		15	baseaddr+110H
2	baseaddr+40H		14	baseaddr+100H
1	baseaddr+30H		13	baseaddr+F0H
0	baseaddr+20H		12	baseaddr+E0H

Note baseaddr: The setting value of the DMABAR register

16.3.3 Vector table

Once the DMA is started, the control data is determined by reading the data from the vector table allocated by each startup source, and the control data assigned to the DMA control data area is read.

The DMA boot source and vector addresses are shown in Table 16-5. Each startup source vector table has 1 byte, holds the data from “00H” to “17H”, and selects 1 from 24 groups of control data Group data. The upper 22 bits of the vector address are set by the DMABAR register, and the lower 10 bits are assigned “00H” to “17H” for the corresponding startup source.

Note The DMAENi0~DMAENi7 bits must be “0” in the corresponding DMAENi (i=0~2) registers (Disable Startup) when setting the starting address of the DMA control data area in the vector table.

Figure 16-4 Starting address and vector table of control data
when DMABAR register is“2000000H” (example)

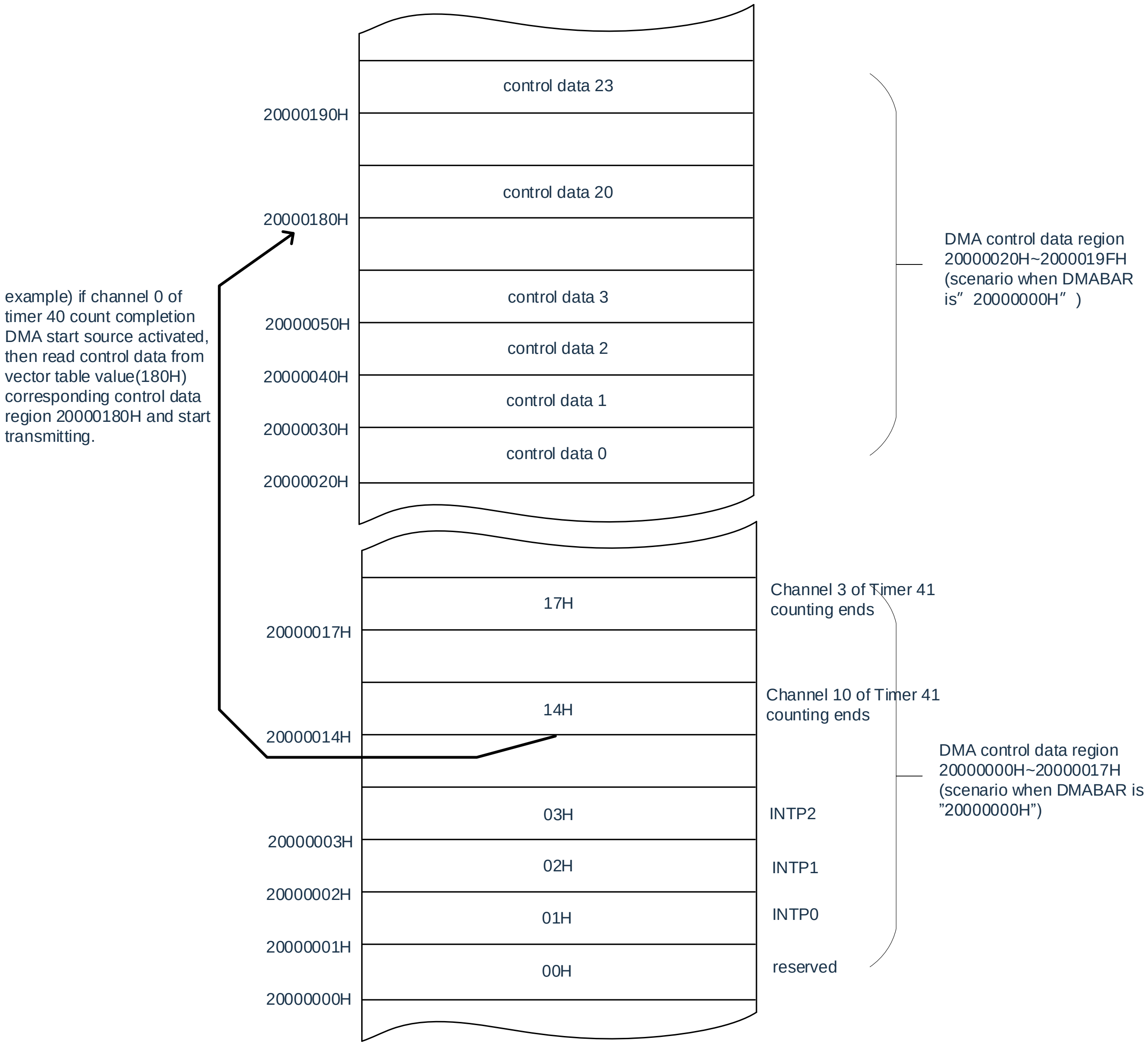


Table 16-5 DMA startup source and vector address

DMA start source (the source where the interrupt request occurred).	The source number	The address of the vector	Priority
Flash read-write erase ends	0	The setting address of the DMABAR register is +00H	high
INTP0	1	The setting address of the DMABAR register is +01H	
INTP1	2	The setting address of the DMABAR register is +02H	
INTP2	3	The setting address of the DMABAR register is +03H	
INTP3	4	The setting address of the DMABAR register is +04H	
The A/D conversion ends	5	The setting address of the DMABAR register is +05H	
retain	6	The setting address of the DMABAR register is +06H	
retain	7	The setting address of the DMABAR register is +07H	
The end of transmission received by UART0 / the end of transmission of SSPI01 or the end of transmission of buffer NULL/IIC01	8	The setting address of the DMABAR register is +08H	
The end of the UART0 transmission / the end of the SSPI00 transmission or the end of the buffer NULL/IIC00 transmission	9	The setting address of the DMABAR register is +09H	
The end of transmission received by UART1 / the end of transmission of SSPI11 or the end of transmission of buffer NULL/IIC11	10	The setting address of the DMABAR register is +0AH	
End of transmission for UART1 transmission/end of transmission for SSPI10 or end of transmission for buffer NULL/IIC10/end of transmission for SPI	11	The setting address of the DMABAR register is +0BL	
The end of transmission received by UART2 / the end of transmission of SSPI21 or the end of transmission of buffer NULL/IIC21	12	The setting address of the DMABAR register is +0CH	
The end of the UART2 transmission / the end of the SSPI20 transmission or the end of the buffer null/IIC20 transmission	13	The DMABAR register is set to address +0DH	
IICA0 communication ends.	14	The setting address of the DMABAR register is +0EI	
A 15-bit interval timer generates a count interrupt	15	The setting address of the DMABAR register is +0FF	
Timer40 ends with the count of channel 0 or capture	16	The setting address of the DMABAR register is +10H	
Timer40 for channel 1 counts or captures end	17	The setting address of the DMABAR register is +11H	
Timer40 for channel 2 counts or snaps ends	18	The setting address of the DMABAR register is +12H	
Timer40 ends with the count or snap of channel 3	19	The setting address of the DMABAR register is +13H	
Timer41 ends counting or snapping of channel 0	20	The setting address of the DMABAR register is +14H	
Timer41 ends with the counting or snapping of channel 1	21	The setting address of the DMABAR register is +15H	
Timer41 ends with the count or snap of channel 2	22	The setting address of the DMABAR register is +16H	
Timer41 ends with the counting or snapping of channel 3	23	The setting address of the DMABAR register is +17H	lo low

16.3.4 Peripheral Enable Register 1 (PER1)

The PER1 register is a register that sets the clock that enable or disables clocking each peripheral hardware. Reduce power consumption and noise by stopping clocking unused hardware.

To use DMA, bit3 (DMAEN) must be set to “1”.

The PER1 register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register becomes “00H”.

Figure 16-5 Format of the peripheral enable register 1 (PER1).

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

symbol	7	6	5	4	3			0
PER1	0	0	–	0	DButIn	0	0	0

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

16.3.5 DMA control register j(DMACRj) (j=0~23).

The DMACRj register controls the operating mode of the DMA.

Figure 16-6　　Format of DMA control register j (DMACRj)

Address: Refer to “16.3.2 Control data allocation “. After reset: Indefinite value　　R/W

Symbol:	15	14	13	12	11	10	9	8
DMACRj	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	S		RPTINT	CHNE	DAMOD	SAMOD	RPTSEL	MODE

S	Selection of transmitted data length
00	8 bits
01	16 bits
10	32-bit
11	Disable the setting

RPTINT	Repeating pattern interrupts allow/disable
0	Interrupts are prohibited.
1	Interrupts are allowed.
When the MODE bit is “0” (normal mode), the RPTINT bit is not set.	

CHNE	Allow/disallow for chain transfers
0	Chain transmission is prohibited.
1	Allow chain transfer.
The CHNE bit of the DMACR23 register must be “0” (chain transfer is prohibited).	

DAMOD	Control of the transmitting destination address
0	fixed
1	Increasing
When the MODE bit is “1” (repeat pattern) and the RPTSEL bit is “0” (the transfer target is the repeat area), the DAMOD bit is not set.	

SAMOD	Control of the transmitting source address
0	fixed
1	Increasing
When the MODE bit is “1” (repeat pattern) and the RPTSEL bit is “1” (the delivery source is the repeat region), the SAMOD bit is not set.	

RPTSEL	Selection of repeating areas
0	The delivery target is a repeating area.
1	The delivery source is a repeat.
When the MODE bit is “0” (normal mode), the setting of the RPTSEL bit is invalid.	

MODE	Selection of transfer mode
0	Normal mode
1	Repeating pattern

Note DMACRj register cannot be accessed via DMA transfer.

16.3.6 DMA block size register j (DMBLSj) (j=0~23)

This register sets the block size of the 1 initiation transfer of data.

Figure 16-7of DMA block size register j (DMBLSj).

Address: Refer to 16.3.2 Control data allocation“. After reset: Indefinite value R/W

Symbol:	15	14	13	12	11	10	9	8
DMBLSj	DMBLSj15	DMBLSj14	DMBLSj13	DMBLSj12	DMBLSj11	DMBLSj10	DMBLSj9	DMBLSj8
	7	6	5	4	3	2	1	0
	DMBLSj7	DMBLSj6	DMBLSj5	DMBLSj4	DMBLSj3	DMBLSj2	DMBLSj1	DMBLSj0

DMBLSj	The transfer block size		
	8-bit transfer	16-bit transfer	32-bit transfer
00H	Disable the setting	Disable the setting	Disable the setting
01H	1 byte	2 bytes	4 bytes
02H	2 bytes	4 bytes	8 bytes
03H	3 bytes	6 bytes	12 bytes
• • •	• • •	• • •	• • •
FDH	253 bytes	506 bytes	1012 bytes
FEH	254 bytes	508 bytes	1016 bytes
FFH	255 bytes	510 bytes	1020 bytes
• • •	• • •	• • •	• • •
FFFFH	65535 bytes	131070 bytes	262140 bytes

Note 1. DMBLSj register cannot be accessed via DMA transfer.

16.3.7 DMA transmit count register j(DMACTj) (j=0~23)

This register sets the number of data transfers to the DMA. Decrements 1 for every DMA transfer started.

Figure 16-8 Format of DMA transmit count register J (DMACTj).

Symbol:	15	14	13	12	11	10	9	8
DMACTj	DMACTj15	DMACTj14	DMACTj13	DMACTj12	DMACTj11	DMACTj10	DMACTj9	DMACTj8
	7	6	5	4	3	2	1	0
	DMACTj7	DMACTj6	DMACTj5	DMACTj4	DMACTj3	DMACTj2	DMACTj1	DMACTj0

Address: Refer to 16.3.2 Control data allocation”.   After reset: Indefinite value   R/W

DMACTj	Number of transfers
00H	Disable the setting
01H	1 time
02H	2 times
03H	3 times
• • •	• • •
FDH	253 times
FEH	254 times
FFH	255 times
• • •	• • •
FFFFH	65535 times

Note 1 DMACTj registers cannot be accessed via DMA transfer.

16.3.8 DMA transmit count reload register j(DMRLDj) (j=0~23).

This register sets the initial value of the number of transfers register in repeat mode. In repeat mode, because the value of this register is reloaded into the DMACT register, the set value must be the same as the initial value of the DMACT register.

Figure 16-9 Format of DMA transmit count reload register j (DMRLDj)

Address: Refer to 16.3.2 Control data allocation”. After reset: Indefinite value R/W

Symbol:	15	14	13	12	11	10	9	8
DMRLDj	DMRLDj15	DMRLDj14	DMRLDj13	DMRLDj12	DMRLDj11	DMRLDj10	DMRLDj9	DMRLDj8
	7	6	5	4	3	2	1	0
	DMRLDj7	DMRLDj6	DMRLDj5	DMRLDj4	DMRLDj3	DMRLDj2	DMRLDj1	DMRLDj0

Note 1. DMRLDj register access is not possible via DMA transfer.

16.3.9 DMA source address register j(DMSARj) (j=0~23).

This register specifies the source address at which data is transferred.

When the SZ bit of the DMACRj register is “01” (16 bits transferred), the lowest bit is ignored and processed as an even address.

When the SZ bit of the DMACRj register is “10” (32-bit transfer), the lower 2 bits are ignored and processed as a word address.

Figure 16-10 Format of DMA source address register j (DMSARj)

Address: Refer to “16.3.2 Control data allocation”. After reset: Indefinite value R/W

symbol	31	30	29	28	27	26	25	24
DMSARj	DMSARj31	DMSARj30	DMSARj29	DMSARj28	DMSARj27	DMSARj26	DMSARj25	DMSARj24
	23	22	21	20	19	18	17	16
	DMSARj23	DMSARj22	DMSARj21	DMSARj20	DMSARj19	DMSARj18	DMSARj17	DMSARj16
	15	14	13	12	11	10	9	8
	DMSARj15	DMSARj14	DMSARj13	DMSARj12	DMSARj11	DMSARj10	DMSARj9	DMSARj8
	7	6	5	4	3	2	1	0
	DMSARj7	DMSARj6	DMSARj5	DMSARj4	DMSARj3	DMSARj2	DMSARj1	DMSARj0

Note 1 The DMSARj register cannot be accessed via DMA transfer.

16.3.10 DMA destination address register j(DMDARj) (j=0~23).

This register specifies the destination address at which data is transferred.

When the SZ bit of the DMACRj register is “01” (16 bits transferred), the lowest bit is ignored and processed as an even address.

When the SZ bit of the DMACRj register is “10” (32-bit transfer), the lower 2 bits are ignored and processed as a word address.

Figure 16-11 format of the DMA destination address register j (DMDARj).

Address: Refer to “16.3.2 Control data allocation “. After reset: Indefinite value R/W

symbol	31	30	29	28	27	26	25	24
DMDARj	DMDARj31	DMDARj30	DMDARj29	DMDARj28	DMDARj27	DMDARj26	DMDARj25	DMDARj24
	23	22	21	20	19	18	17	16
	DMDARj23	DMDARj22	DMDARj21	DMDARj20	DMDARj19	DMDARj18	DMDARj17	DMDARj16
	15	14	13	12	11	10	9	8
	DMDARj15	DMDARj14	DMDARj13	DMDARj12	DMDARj11	DMDARj10	DMDARj9	DMDARj8
	7	6	5	4	3	2	1	0
	DMDARj7	DMDARj6	DMDARj5	DMDARj4	DMDARj3	DMDARj2	DMDARj1	DMDARj0

Note: DMDARj registers cannot be accessed via DMA transfer.

16.3.11 DMA boot enable register i (DMAENi) (i=0~2).

This is the 8-bit register that controls the boot of the DMA through each interrupt source. The corresponding connection between the interrupt source and the DMAENi0~DMAENi7 bits is shown in Table 16-6. DMAENi registers can be set via 8-bit memory operation instructions.

Note 1. The DMAENi0~DMAENi7 bits must be changed at the boot source that does not produce the corresponding bits.
2. DMAENi registers cannot be accessed via DMA transfer.
3. The assigned function varies from product to product, and the bits without the assigned function must be set to "0".

Figure 16-12 Format of the enable register i (DMAENi) (i=0~2)

Address:40005000H(DMAEN0), 40005001H(DMAEN1),
40005002H(DMAEN2) After reset:00H R/W

symbol	7	6	5	4	3	2	1	0
DMAENi	DMAENj7	DMAENj6	DMAENj5	DMAENj4	DMAENj3	DMAENj2	DMAENj1	DMAENj0

DMAENi7	DMA boot enable i7
0	Disable startup.
1	Enable startup.
Depending on the condition under which the end-of-transmission interrupt occurs, the DMAENi7 bit becomes “0” (disable start).	

DMAENi6	DMA boot enable i6
0	Disable startup.
1	Enable startup.
Depending on the conditions under which the end-of-transmission interrupt occurs, the DMAENi6 bit becomes “0” (disable start-up).	

DMAENi5	DMA boot enable i5
0	Disable startup.
1	Enable startup.
Depending on the condition under which the end-of-transmission interrupt occurs, the DMAENi5 bit becomes “0” (disable start).	

DMAENi4	DMA boot enable i4
0	Disable startup.
1	Enable startup.
Depending on the conditions under which the end-of-transmission interrupt occurs, the DMAENi4 bit becomes “0” (disable start-up).	

DMAENi3	DMA boot enable i3
0	Disable startup.
1	Enable startup.
Depending on the condition under which the end-of-transmission interrupt occurs, the DMAENi3 bit becomes “0” (disable start-up).	

DMAENi2	DMA boot enable i2
0	Disable startup.
1	Enable startup.
Depending on the condition under which the end-of-transmission interrupt occurs, the DMAENi2 bit becomes “0” (disable start).	

DMAENi1	DMA boot enable i1
0	Disable startup.
1	Enable startup.
Depending on the conditions under which the end-of-transmission interrupt occurs, the DMAENi1 bit becomes “0” (disable start).	

DMAENi0	DMA boot enable i0
0	Disable startup.
1	Enable startup.
Depending on the condition under which the end-of-transmission interrupt occurs, the DMAENi0 bit becomes “0” (disable start-up).	

Table 16-6 Interrupt source corresponds to DMAENi0~DMAENi7 bits

register	DMAENi 7 bits	DMAENi 6 bits	DMAENi 5 bits	DMAENi 4 bit	DMAENi 3 bits	DMAENi 2 bits	DMAENi 1 bit	DMAENi0 bit
DMAEN0	retain	retain	The A/D conversion ends	INTP3	INTP2	INTP1	INTP0	Flash erase/write ends
DMAEN1	15-bit interval timer interrupt	IICA0 communication ends	The end of the UART2 transmission / the end of the SSPI20 transmission or the end of the buffer null/IIC20 transmission	The end of transmission received by UART2 / the end of transmission of SSPI21 or the end of transmission of buffer NULL/IIC21	End of transmission for UART1 transmission/end of transmission for SSPI10 or end of transmission for buffer NULL/IIC10/end of transmission for SPI	The end of transmission received by UART1 / the end of transmission of SSPI11 or the end of transmission of buffer NULL/IIC11	The end of the UART0 transmission / the end of the SSPI00 transmission or the end of the buffer NULL/IIC00 transmission	The end of transmission received by UART0 / the end of transmission of SSPI01 or the end of transmission of buffer NULL/IIC01
DMAEN2	The counting end of channel 3 of the timer array unit 1 ends or the capture ends	The counting end of channel 2 of the timer array unit 1 ends or the capture ends	The counting end of channel 1 of timer array unit 1 or the end of the snap	The counting end of channel 0 of timer array unit 1 ends or the snap ends	The counting end of channel 3 of the timer array unit 0 or the end of the snap	The counting end of channel 2 of timer array unit 0 ends or the snap ends	The counting end of channel 1 of the timer array unit 0 ends or the capture ends	The counting end of channel 0 of the timer array unit 0 ends or the snap ends

Note Bits that are not assigned a function must be set to "0".

Note i=0~2

16.3.12 DMA base address register (DMABAR).

This is a 32-bit register that sets the vector address that holds the start address of the DMA control data area and the address of the DMA control data area.

Note 1. The DMABAR register must be changed with all DMA boot sources set to a state that disables startup.

2. DMABAR registers can only be rewritten once.

3. DMABAR register access is not possible via DMA transfer.

4. For the allocation of DMA control data area and DMA vector table area, please refer to the note " 16.3.1DMA control data areas and DMA vector table areas allocation".

5. Set the register to keep 512-byte aligned, i.e. the low 8 bits set to zero. DMA hardware ignores the low 8 bits.

6. This register can only be accessed by WORD, ignored by BYTE and HALFWORD access.

Figure 16-13 Format of DMA base address register (DMABAR)

Address: 40005008H After reset: 00000000H R/W

symbol	31	30	29	28	27	26	25	24
DMABARj	DMABARj31	DMABARj30	DMABARj29	DMABARj28	DMABARj27	DMABARj26	DMABARj25	DMABARj24
	23	22	21	20	19	18	17	16
	DMABARj23	DMABARj22	DMABARj21	DMABARj20	DMABARj19	DMABARj18	DMABARj17	DMABARj16
	15	14	13	12	11	10	9	8
	DMABARj15	DMABARj14	DMABARj13	DMABARj12	DMABARj11	DMABARj10	DMABARd9	DMABARd8
	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	0

16.4 DMA operation

Once the DMA is started, the control data is read from the DMA control data area, the data is transmitted according to this control data, and the control data after the data transmission is written back to the DMA control data area. It can save 24 groups of control data to the DMA control data area and transfer 24 groups of data. There are normal and repeat modes in the transfer mode, and the transfer sizes are 8-bit transfer, 16-bit transfer, and 32-bit transfer. When the CHNE bit of the DMACRj(j=0~23) register is “1” (chain transfer enable), 1 passes The initiation source reads multiple control data for continuous data transfer (chain transfer).

The transmit source address and the transmit destination address are specified through the 32-bit DMSARj register and the 32-bit DMDARj register, respectively. After data transfer, the values of the DMSARj register and the DMDARj register are incremented or fixed according to the control data.

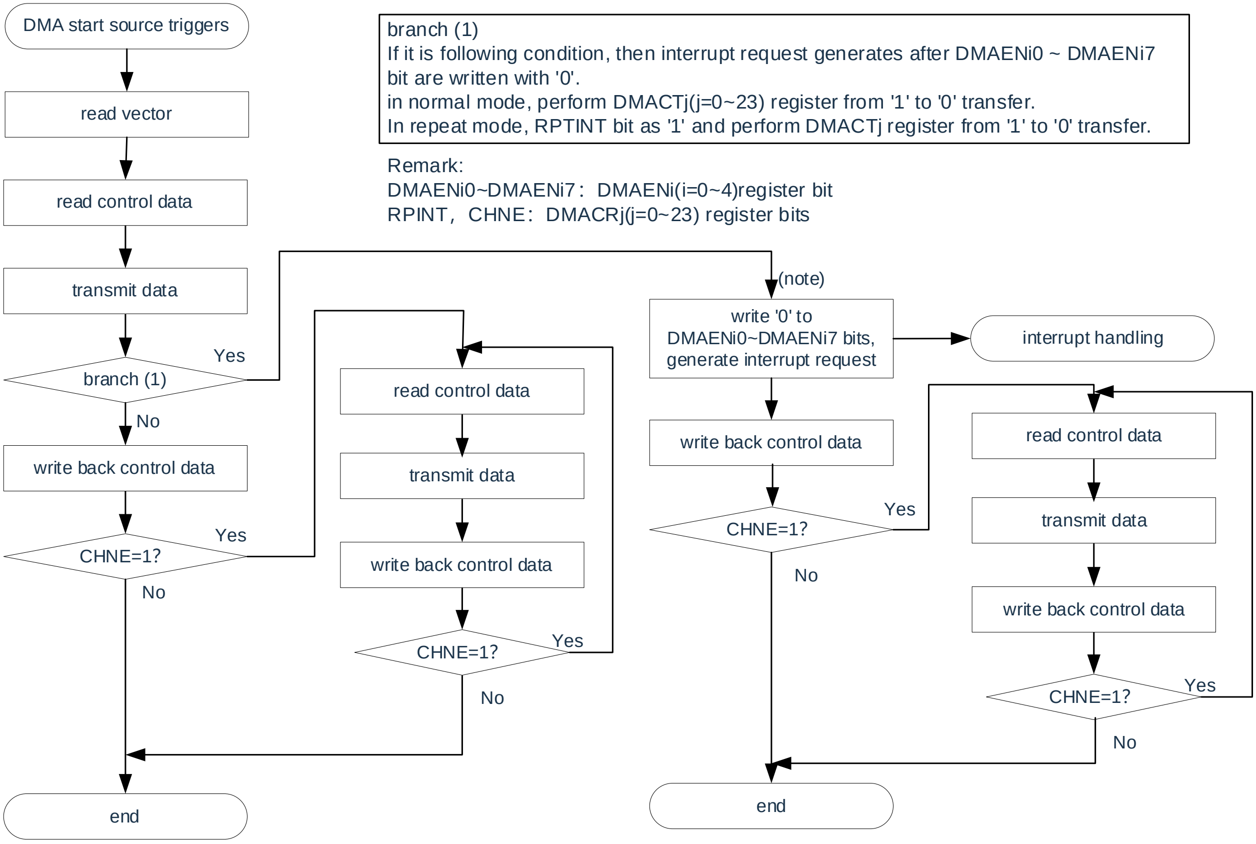
16.4.1 Start the source

The DMA is initiated by the interrupt signal of the peripheral function, and the interrupt signal to start the DMA is selected through the DMAENi (i=0~2) register. When the data transmission (in the case of chain transmission, continuous initial transmission) is set to the DMAENi0~DMAENi7 bits of the corresponding DMAENi register in the DMA operation "0" (disables startup).

- In normal mode, a DMACTj (j=0~23) register is transferred to "0".
- In repeat mode, the RPTINT bit of the DMACRj register is “1” (interrupts are enable) and the DMACTj register is transferred to “0”.

The internal operation flowchart of DMA is shown in Figure 16-14.

Figure 16-14 Flowchart of DMA internal operation



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=0~23) 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=0~23) register bits

Note: In data transfers initiated through the setting of Enable Chain Transfer (CHNE=1), DMAENi0~DMAENi7 bits are not written “0” and no interrupt requests are generated.

16.4.2 Normal mode

In the case of 8-bit transmission, the transmission data for 1 start is 1 to 65535 bytes; in the case of 16-bit transmission, the transmission data for 1 start is 2 to 131070 bytes; in the case of 32-bit transmission, the transmission data for 1 start is 4 to 262140 bytes. The number of transmissions is 1 to 65535 times. If the DMACTj (j=0 to 23) register becomes "0", the interrupt request corresponding to the start-up source is generated to the interrupt controller during DMA operation, and the DMAENi0 to DMAENi7 bits of the corresponding DMAENi (i=0 to 2) register are set to "0" (disable start-up).

The register function and data transfer in normal mode are shown in Table 16-7 and Figure 16-15.

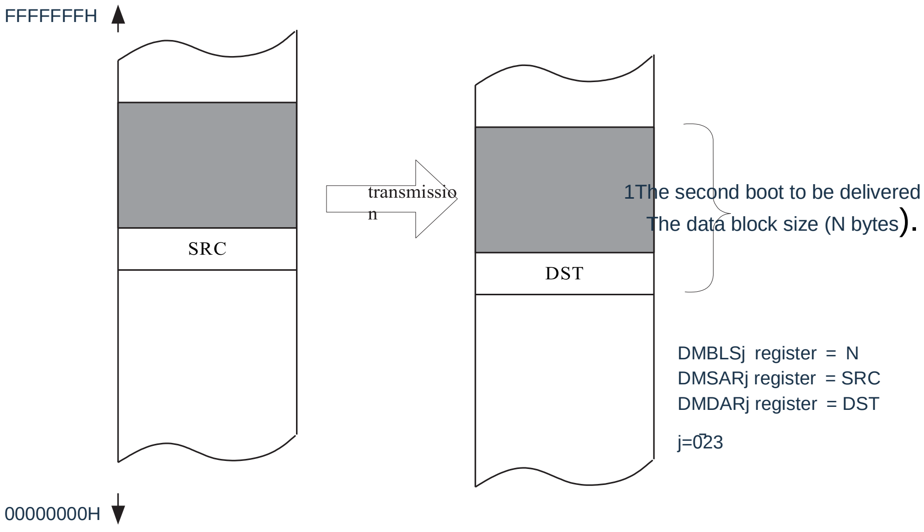
Table 16-7 Register function in normal mde

Register name	Symbol	Function
DMA block size register j	DMBLSj	The size of the data block to be transferred by 1 start
DMA transfer count register j	DMACTj	The number of times the data was transmitted
DMA transfer number of times to reload register j	DMRLDj	Not used [Note].
DMA source address register j	DMSARj	The address of the source of the data
DMA destination address register j	DMDARj	The destination address of the data

Note When parity error reset (RPERDIS=0) is allowed by RAM parity error detection function, initialization (00H) must be performed.

Note j=0~23

Figure 16-15 Data transfer in normal mode



Setting of the DMACR register				Control of the source address	Control of the destination address	The source address after transfer	Destination address after transfer
DAMOD	SAMOD	RPTSEL	MODE				
0	0	X	0	fixed	fixed	SRC	Dst
0	1	X	0	Increasing	fixed	SRC+N	Dst
1	0	X	0	fixed	Increasing	SRC	DST+N
1	1	X	0	Increasing	Increasing	SRC+N	DST+N

X: “0” or “1”

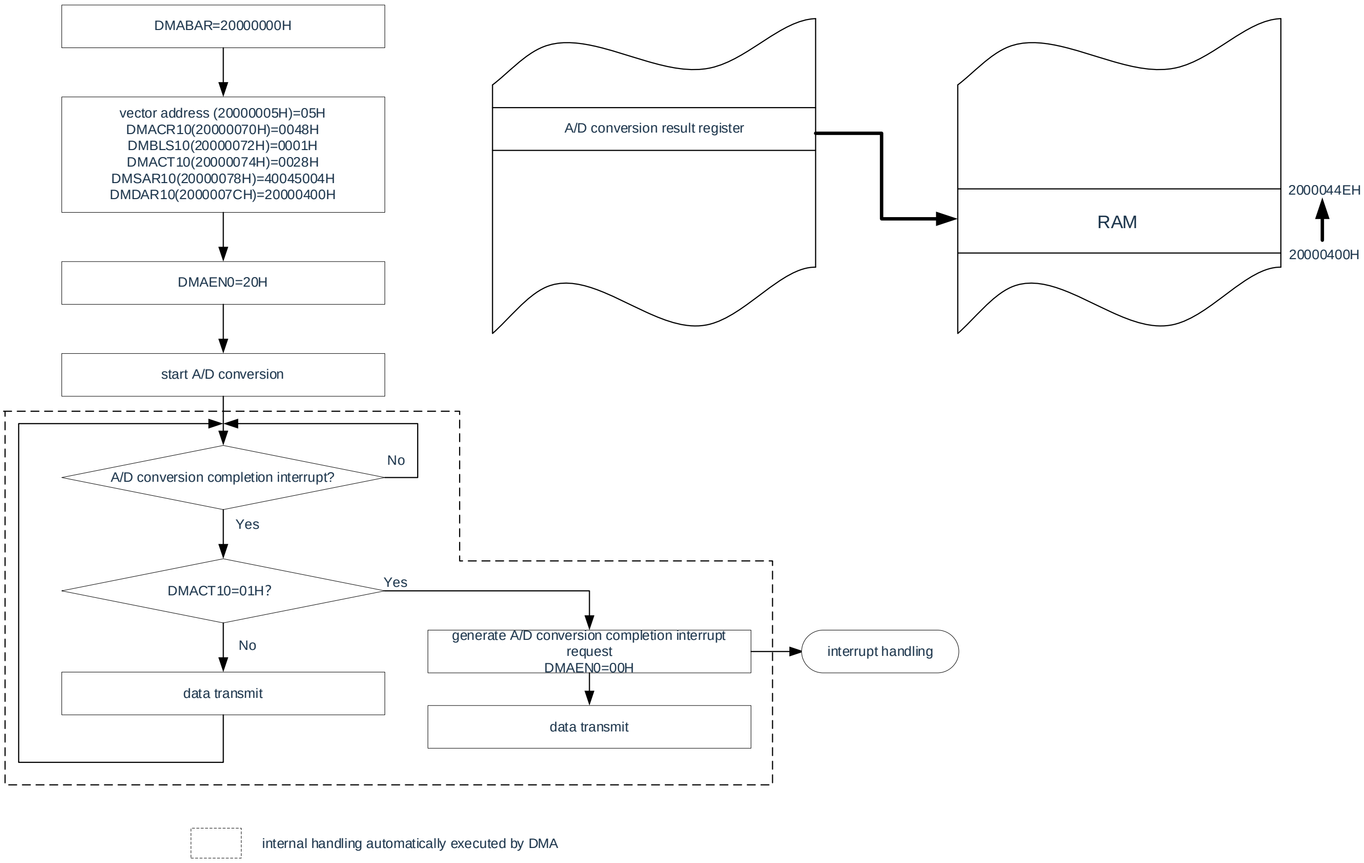
(1) Example 1 of the use of normal mode: Continuous A/D conversion results

DMA is started by an A/D conversion end interrupt, and the value of the A/D conversion result register is transferred to RAM.

• The vector address is allocated at 200,00005H, and the control data is distributed at 20000070 H~2000007FH.

• Transfer 2 bytes of data from the A/D conversion result registers (40045004H, 40045005H) 40 times to 20000400H~of RAM 2000044FH 80 bytes.

Figure 16-16 Normal mode usage example 1: Continuously take the A/D conversion result


DMABAR=20000000H
vector address (20000005H)=05H
DMACR10(20000070H)=0048H
DMBLS10(20000072H)=0001H
DMACT10(20000074H)=0028H
DMSAR10(20000078H)=40045004H
DMDAR10(2000007CH)=20000400H
DMAEN0=20H
start A/D conversion
A/D conversion completion interrupt?
No
Yes
DMACT10=01H?
Yes
No
data transmit
generate A/D conversion completion interrupt request
DMAEN0=00H
data transmit
interrupt handling
A/D conversion result register
RAM
2000044EH
20000400H
internal handling automatically executed by DMA

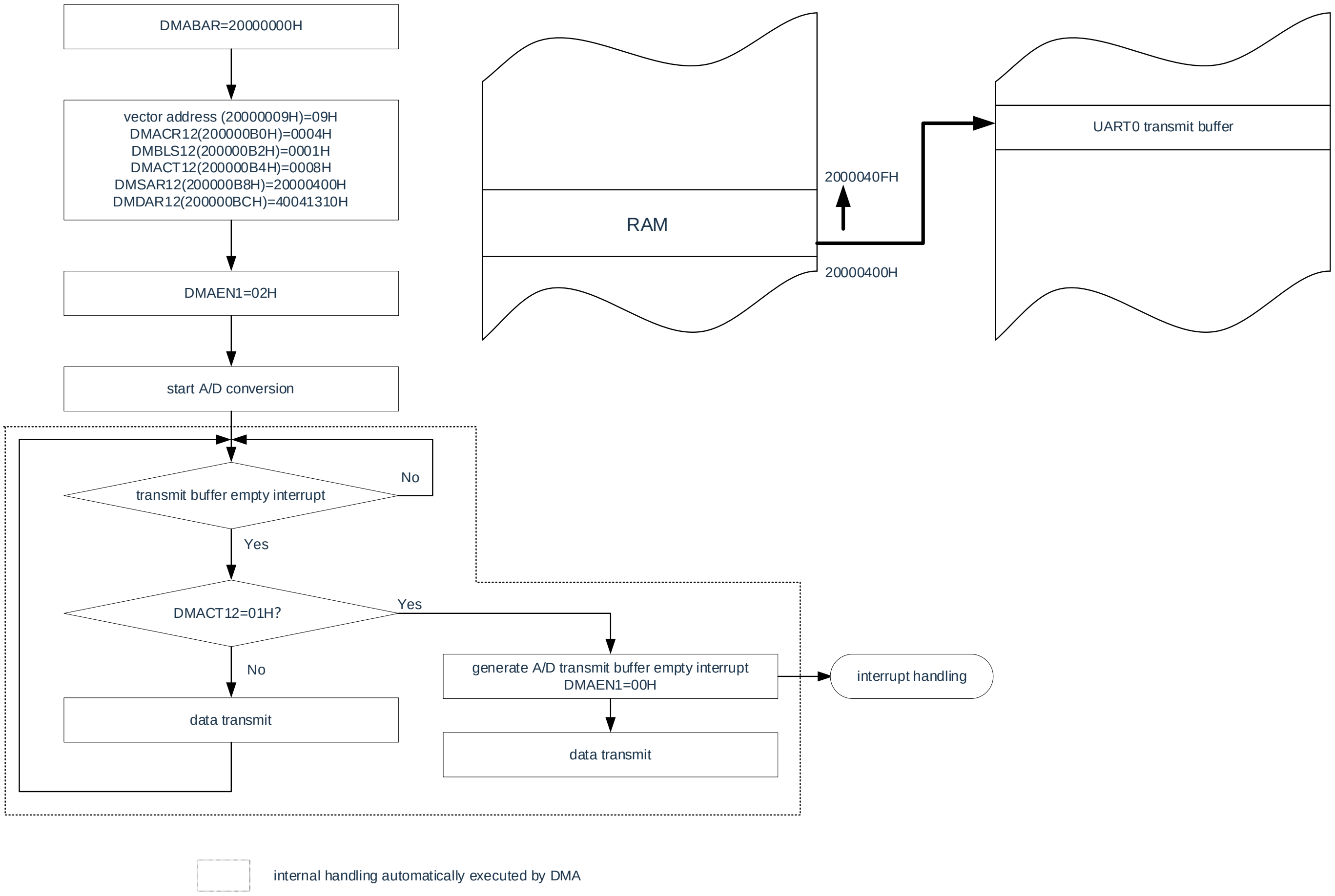
Because it is in normal mode, the value of the DMRLD10 register is not used. However, when parity error reset (RPERDIS=0) is allowed to occur via the RAM parity error detection function, the DMRLD10 register must be initialized (0000H).

(2) Example 2 of the use of normal mode: UART0 transmits continuously

DMA is started through a blank interrupt from UART0's send buffer, and the value of RAM is transferred to UART0's send buffer.

- The vector address is allocated at 20000009H, and the control data is allocated at 200000B0H~200000BFH.
- Transfer 8 bytes of RAM 20000400H~20000407H to UART0's send buffer (40041310H).

Figure 16-17 Example 2 of the use of normal mode: UART0 transmits continuously



Because it is in normal mode, the value of the DMRLD12 register is not used. However, when parity error reset (RPERDIS=0) is allowed through the RAM parity error detection function, the DMRLD12 register must be initialized (0000H).

The first UART0 send must be started through the software. Start DMA with an empty interrupt from the send buffer and then automatically send after the second time.

16.4.3 Repeat pattern

The transfer data for one initiation is 1 to 65535 bytes. The source or destination is designated as a repeat area, and the number of transfers is 1 to 65535 times. Once the specified number of transfers is complete, initialize the DMACTj(j=0~23) register and the address specified as a repeat, and then repeat the transfer. This is when the RPTINT bit of the DMACRj register is “1” (interrupts are allowed) and a data transfer is made where the DMACTj register becomes “0” DMA generates an interrupt request for the corresponding start source to the interrupt controller during operation, and the DMAENi0~DMAENi7 of the corresponding DMAENi (i=0~2) registers Position “0” (disable startup). When the RPTINT bit of the DMACRj register is “0” (interrupt is prohibited), even if the DMACTj register becomes “0” data transfer, no interrupt requests are generated, and the DMAENi0~DMAENi7 bits are unchanged from “0”.

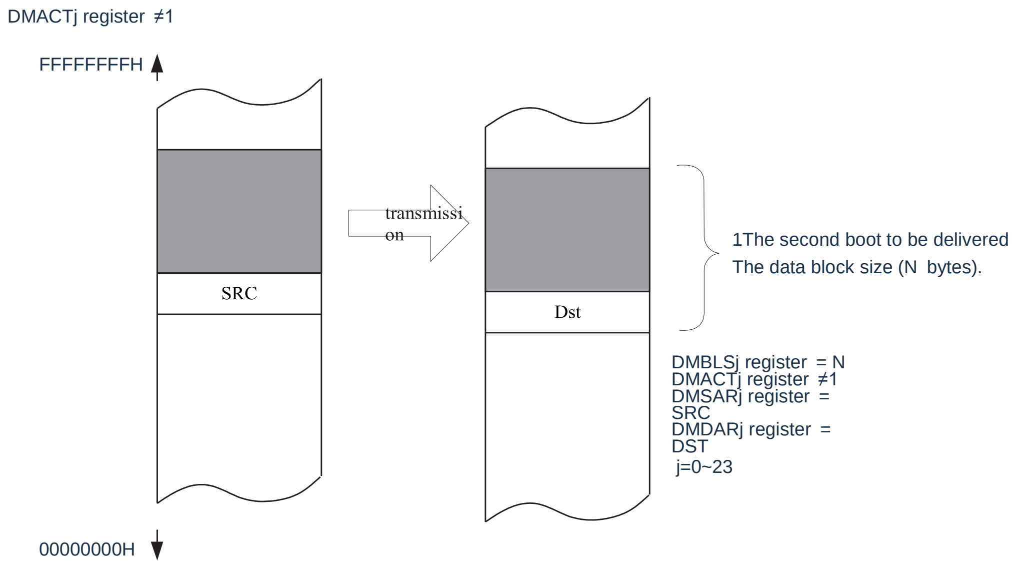
The register function and data transfer of the repeating pattern are shown in Table 16-8 Figure 16-18, respectively.

Table 16-8 Register Functions for Repeat Pattern

The register name	symbol	function
DMA block size register j	DMBLSj	The size of the data block to be transferred by 1 start
DMA transfer count register j	DMACTj	The number of times the data was transmitted
DMA transfer number of times to reload register j	DMRLDj	Reload the value of this register into the DMACT register. (Initialize the number of data transfers)
DMA source address register j	DMSARj	The address of the source of the data
DMA destination address register j	DMDARj	The destination address of the data

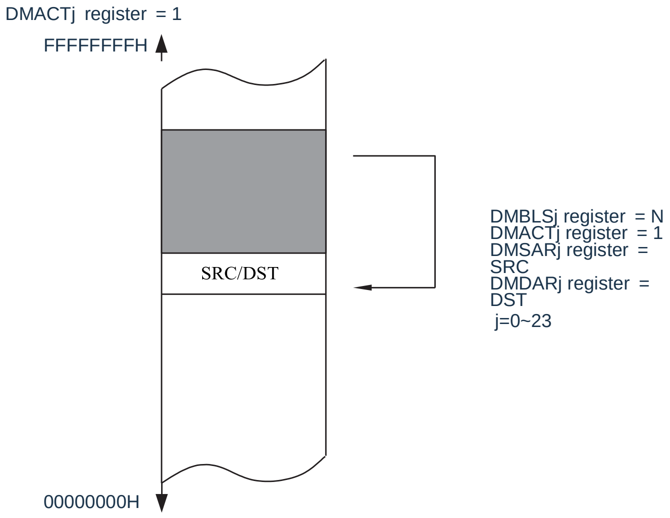
Note j=0~23

Figure 16-18 Data transfer in repeat mode



The setting of the DMACR register				Control of the source address	Control of the destination address	The source address after transmission	The destination address after transmission
DAMOD	SAMOD	RPTSEL	MODE				
0	X	1	1	Repeating area	fixed	SRC+N	Dst
1	X	1	1	Repeating area	Increasing	SRC+N	DST+N
X	0	0	1	fixed	Repeating area	SRC	DST+N
X	1	0	1	Increasing	Repeating area	SRC+N	DST+N

X: “0” or “1”



The setting of the DMACR register				Control of the source address	Control of the destination address	The source address after transmission	The destination address after transmission
DAMOD	SAMOD	RPTSEL	MODE				
0	X	1	1	Repeating area	fixed	SRC	Dst
1	X	1	1	Repeating area	Increasing	SRC	DST+N
X	0	0	1	fixed	Repeating area	SRC	Dst
X	1	0	1	Increasing	Repeating area	SRC+N	Dst

X: “0” or “1”

Note 1 When using repeat patterns, the data length of the repeat must be set to less than 65535 bytes.

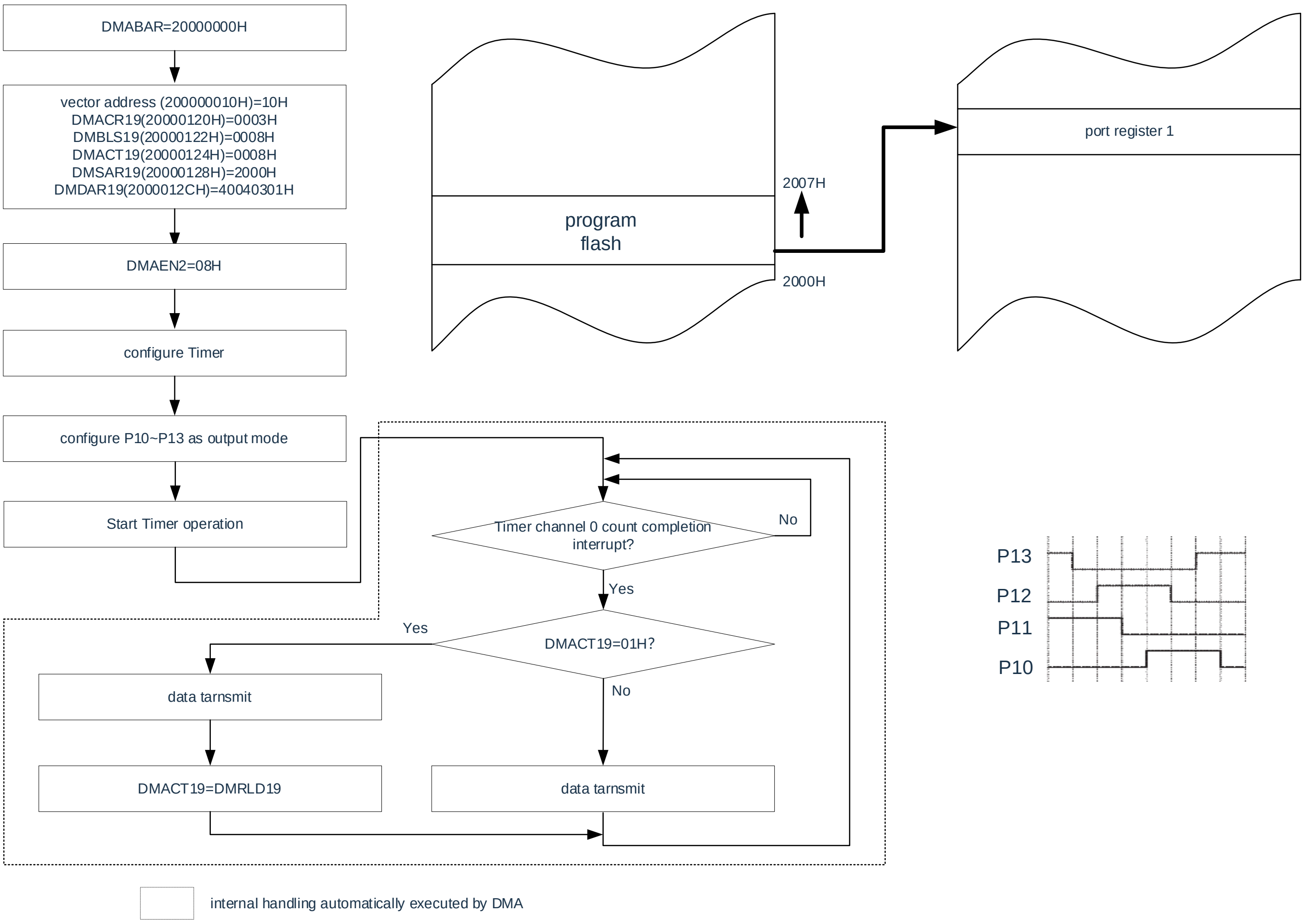
(1) Example of the use of repeat mode: Use the stepper motor of the port to control the pulse output

The DMA is started using the Channel 0 interval timer function of the Timer40, and the mode of the motor control pulse saved in the code flash memory is transferred to the universal port.

•The vector address is allocated at 20000010H, and the control data is allocated at 20000120H~2000012PH.

- Transfer 8 bytes of code flash 02000H~02007H to port register 1 (40040301H).
- Disable repeat mode interruption.

Figure 16-19 Example use of repeat mode: A stepper motor using a port is used to control the pulse output



To stop the output, bit0 of DMAEN2 must be cleared after stopping the operation of the timer.

16.4.4 Chain transfer

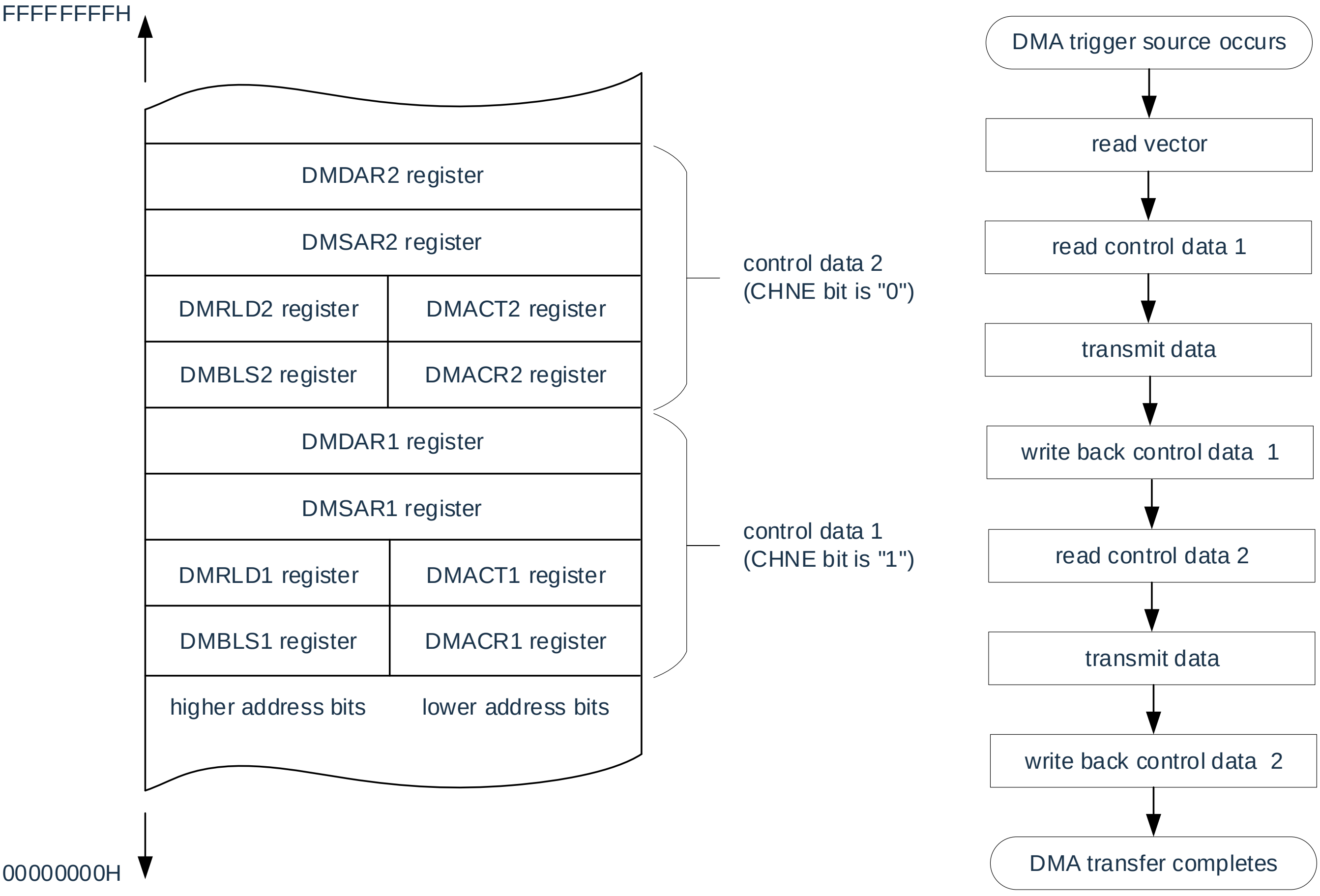
When the CHNE bit of the DMACRj(j=0~23) register is "1" (allow chain transfer), multiple data can be transferred continuously through one startup source.

Once the DMA is started, the control data is selected by reading the data from the corresponding vector address of the startup source, and the control data assigned to the DMA control data area is read. If the CHNE bit of the read control data is "1" (allowing chain transfer), the transfer continues after the transfer is completed by reading the next assigned control data. Repeat this operation until the control data transfer with the CHNE bit "0" (disable chain transmission) ends.

When multiple control data are used for chain transfer, the number of transmissions set by the first control data is valid, while the number of transmissions of the control data processed after the second is invalid.

The flowchart of chain transfer is shown in Figure 16-20.

Figure 16-20 Flow chart of chain transfers



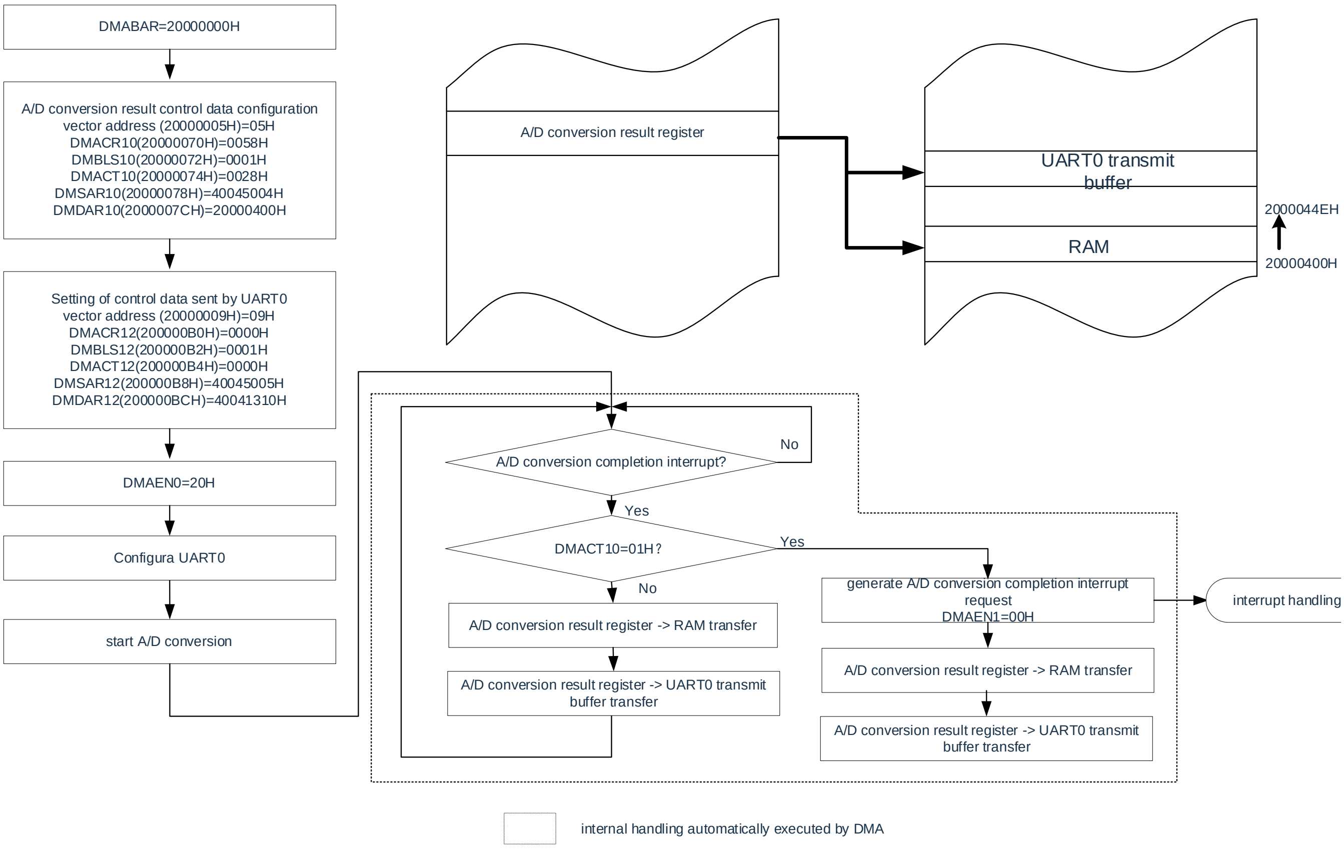
Note 1. The CHNE bit of the DMACR23 register must be "0" (chain transfer is prohibited).
2. In the data transfer after the second time of the chain transfer, the bits DMAENi0~DMAENi7 of DMAENi (i=0~2) register does not change to "0" (DMA is prohibited from starting) and no interrupt requests are generated.

(1) Example of using chain transfer: Continuous A/D conversion result for UART0 transmission

DMA is started by interrupting the end of the A/D conversion, and the A/D conversion result is transferred to RAM for UART0 transmission.

- The vector addresses are 20000005 H and 20000009H, respectively.
- Control data distribution of A/D conversion results is 20000070H~2000007FH.
- The control data sent by UART0 is distributed between 200000B0 H and 200000BFH.
- Transfer 2 bytes of data from the A/D conversion result registers (40045004H, 40045005H) to 20000400H~2000044FH of RAM , and the high bit 1 byte (40045005H) of the A/D conversion result register is transferred to the send buffer (40041310 of UART 0 H).

Figure 16-21 Example of chain transfer: Continuous A/D conversion results are used for UART0 transmission


DMABAR=20000000H
A/D conversion result control data configuration
vector address (20000005H)=05H
DMACR10(20000070H)=0058H
DMBLS10(20000072H)=0001H
DMACT10(20000074H)=0028H
DMSAR10(20000078H)=40045004H
DMDAR10(2000007CH)=20000400H
Setting of control data sent by UART0
vector address (20000009H)=09H
DMACR12(200000B0H)=0000H
DMBLS12(200000B2H)=0001H
DMACT12(200000B4H)=0000H
DMSAR12(200000B8H)=40045005H
DMDAR12(200000BCH)=40041310H
DMAEN0=20H
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
interrupt handling
A/D conversion result register -> RAM transfer
A/D conversion result register -> UART0 transmit buffer transfer
internal handling automatically executed by DMA

16.5 Precautions when using DMA

16.5.1 DMA controls the settings of data and vector tables

- The DMA Base Address Register (DMABAR) must be changed with all DMA boot sources set to a state that disables startup.
- DMA Base Address Register (DMABAR) can only be overridden once.
- The DMAENi0~DMAENi 7 bits must be "0" in the corresponding DMAENi (i=0~2) registers (DMA is prohibited Startup) when changing DMACRj, DMBLSj, DMACTj, DMRLDj, DMSARj, Data for the DMDARj register.
- The DMAENi0~DMAENi 7 bits must be "0" in the corresponding DMAENi (i=0~2) registers (DMA is prohibited Start) when setting the starting address of the DMA control data area in the vector table.

16.5.2 DMA controls the allocation of data areas and DMA vector table areas

The areas in which DMA control data and vector tables can be assigned vary depending on the product and conditions of use.

- The stack area, DMA control data area, and DMA vector table area cannot overlap.
- When parity error reset (RPERDIS=0) is allowed to occur via RAM parity error detection function, the DMRLD register must be initialized even when using normal mode (0 000H).

16.5.3 Number of execution clocks for DMA

The execution and number of clocks required at DMA startup are shown in Table 16-9.

Table 16-9 Execution and number of clocks required when DMA is started

Read vector	Control data		Read the data	Write data
	read	Write back		
1	4	Note 1	Note 2	Note 2

Note 1. For the number of clocks required to write back control data, refer to Table 16-10 Number of clocks required to write back control data

2. For the number of clocks required to read and write data, please refer to "Table 16-11 Number of clocks required to read and write data

Table 16-10 Number of clocks required to write back control data

Setting of the DMACR register				Address settings		Controls the write-back of registers				The number of clocks
DAMOD	SAMOD	RPTSEL	MODE	source	target	DMACTj register	DMRLDj register	DMSARj register	DMDARj register	
0	0	X	0	fixed	fixed	Write back	Write back	Do not write back	Do not write back	1
0	1	X	0	Increasing	fixed	Write back	Write back	Write back	Do not write back	2
1	0	X	0	fixed	Increasing	Write back	Write back	Do not write back	Write back	2
1	1	X	0	Increasing	Increasing	Write back	Write back	Write back	Write back	3
0	X	1	1	Repeating area	fixed	Write back	Write back	Write back	Do not write back	2
1	X	1	1		Increasing	Write back	Write back	Write back	Write back	3
X	0	0	1	fixed	Repeating area	Write back	Write back	Do not write back	Write back	2
X	1	0	1	Increasing		Write back	Write back	Write back	Write back	3

Note j=0~23, X: "0" or "1"

Table 16-11 Number of clocks required to read and write data

Execution status	RAM	Code flash	Data flash	Special function registers (SFR)	Extended Special Function Register (2ndSFR)	
					No waiting	wait
Read the data	1	2	4	1	1	1+ wait number$^{\text{Note}}$
Write data	1	—	—	1	1	1+ wait number$^{\text{Note}}$

16.5.4 Response time of DMA

The DMA response time is shown in Table 16-12. DMA response time refers to the time from the time the DMA boot source is detected to the start of the DMA transfer, excluding the number of execution clocks for the DMA.

Table 16-12 Response time for DMA

	Minimum time	Maximum time
Response time	3 clocks	23 clocks

However, the response of the DMA may also be delayed in the following cases. The number of clocks delayed varies depending on the condition.

- Maximum response time for the execution of instructions from internal RAM: 20 clocks

Note 1 clock: 1/f CLK (fCLK: CPU/peripheral hardware clock).

16.5.5 Startup source for DMA

• You cannot enter the same startup source between entering the DMA startup source and ending the DMA transfer.

• At the location where the DMA boot source is generated, the DMA boot allow bit corresponding to that boot source cannot be manipulated.

• If the DMA boot source sends a race, the priority is determined when the CPU accepts the DMA transmission and determines the boot source. For priority of startup sources, refer to the 16.3.3 Vector table.

16.5.6 Operation in standby mode

state	DMA operation
Sleep mode	Can be operated (disable operation in low-power RTC mode).
Deep sleep mode	Can accept the DMA start source and make DMA transfer Note 1

Note 1 In deep sleep mode, DMA transmission can be performed after the DMA startup source is detected, and the deep sleep mode can be returned after the transfer is completed. However, because the code flash and data flash stop running in deep sleep mode, you cannot set flash as the transfer source.

Chapter 17 Linkage Controller (EVENTC)

17.1 Feature of EVENTC

EVENTC links the events output by each peripheral function to each other between the peripheral functions. It can be operated directly through the event chain without going through the CPU, and can be operated directly between peripheral functions.

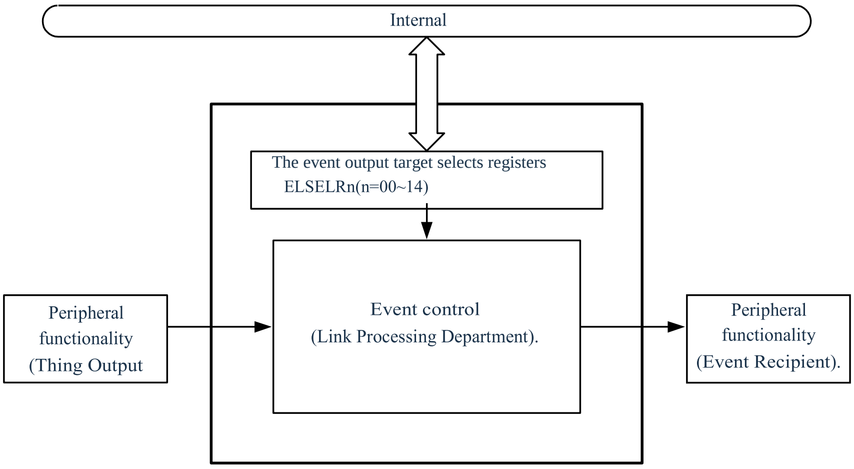
EVENTC has the following features:

- Depending on the product, the event signals of 15 peripheral functions can be directly linked to the specified peripheral functions.
- Depending on the product, the event signal can be used as the startup source for the operation of one of the four peripheral functions.

17.2 Structure of EVENTC

The block diagram of EVENTC is shown in Figure 17-1.

Figure 17-1 Diagram of EVENTC



17.3 Control registers

The controller registers are shown in Table 17-1.

Table 17-1 Control registers of EVENTC

Register name	Symbol
Event output target selects register 00	ELSELR00
Event output target selects register 01	ELSELR01
Event output target selects register 02	ELSELR02
Event output target selects register 03	ELSELR03
Event output target selects register 04	ELSELR04
Event output target selects register 05	ELSELR05
Event output target selects register 06	ELSELR06
Event output target selects register 07	ELSELR07
Event output target selects register 08	ELSELR08
Event output target selects register 09	ELSELR09
Event output target selects register 10	ELSELR10
Event output target selects register 11	ELSELR11
Event output target selects register 12	ELSELR12
Event output target selects register 13	ELSELR13
Event output target selects register 14	ELSELR14

17.3.1 Output target selection register n (ELSELRn) (n=00~14)

The ELSELRn register links each event signal to the event receiver peripheral function (link target peripheral function) to run when the event accepts the event. You cannot link multiple event inputs to the same event output destination (event acceptor). Otherwise, the event receiver's peripheral functionality may operate uncertainly and the event signal may not be accepted properly. Also, you cannot set the event link occurrence source and event output destination to the same function.

The ELSELRn register must be set during the period when the peripheral functions of all event outputs do not generate an event signal.

The correspondence between the ELSELRn register (n=00~14) and the peripheral functions is shown in Table 17-2. The corresponding operation between the config value of the ELSELRn register (n=00 ~14) and the link target peripheral function when receiving the event is shown Table 17-3.

Figure 17-2 Format of event output target selection register n (ELSELRn)

Address: 40043400H(ELSELR00) ~ 4004340EH (ELSELR14) After reset: 00HR/W

symbol	7	6	5	4	3	2	1	0
ELSELRn	0	0	0	0	0	ELSELn2	ELSELn1	ELSELn0

ELSELn2	ELSELn1	ELSELn0	Selection of event links
0	0	0	Disable event linking.
0	0	1	Selection of the linked peripheral function 1 operation[Note 1].
0	1	0	Selection of the linked peripheral function 2 operation[Note 1].
0	1	1	Selection of the linked peripheral function 3 operation[Note 1].
1	0	0	Selection of the linked peripheral function 4 operation[Note 1].
other			Set Prohibited

Note 1 Please refer to Table 17-3 Correspondence between the setting value of ELSELRn register (n=00~14) and the operation when the link target peripheral function accepts the event

Table 17-2 ELSELRn registers (n=00~14) and peripheral functions

Register	Event occurrence source (output source for event input n)	Content
ELSELR00	External interrupt edge detection 0	INTP0
ELSELR01	External interrupt edge detection1	INTP1
ELSELR02	External interrupt edge detection2	INTP2
ELSELR03	External interrupt edge detection3	INTP3
ELSELR04	RTC fixed period/alarm clock consistent detection	INTRTC
ELSELR05	Timer40 channel 00 count end/capture end	INTTM00
ELSELR06	Timer40 channel 01 count end/capture end	INTTM01
ELSELR07	Timer40 channel 02 count end/capture end	INTTM02
ELSELR08	Timer40 channel 03 count end/capture end	INTTM03
ELSELR09	Timer41 channel 00 count end/capture end	INTTM10
ELSELR10	Timer41 channel 01 count end/capture end	INTTM11
ELSELR11	Timer41 channel 02 counts end/capture end	INTTM12
ELSELR12	Timer41 channel 03 count end/capture end	INTTM13
ELSELR13	reserve	INTCMP0
ELSELR14	reserve	INTCMP1

Table 17-3 Correspondence between the setting value of ELSELRn register (n=00~14) and the operation when the link target peripheral function accepts the event

ELSELRn register ELSELn2~ELSELn0 bits	Link the target No.	Link target perimeter functionality	Operation when the event is accepted
001B	1	A/D converter	Start the A/D conversion.
010B	2	Timer40 channel 0 Timer input Note 1	Delay counter, measurement of input pulse interval, external event counter
011B	3	Timer40 channel 1 Timer input Note 2	Delay counter, measurement of input pulse interval, external event counter
100B	4	The EPWM output controls the truncation source	Forced cut-off of the pulse output

Note 1 To select the timer input of Timer40 channel 0 as the link target peripheral function, you must first set the operating clock of channel 0 to f_{CLK} via Timer Clock Select Register 0 (TPS0), set the noise filter of TI00 pin to OFF via Noise Filter Enable Register 1 (NFEN1) (TNFEN00=0), and set the timer input used by channel 0 to the event input signal of the link controller via Timer Input Selection Register 0 (TIS0).

2. To select the timer input of Timer40 channel 1 as the link target peripheral function, you must first set the operating clock of channel 1 to f_{CLK} via Timer Clock Selection Register 0 (TPS0), set the noise filter of TI01 pin to OFF via Noise Filter Enable Register 1 (NFEN1) (TNFEN01=0), and set the timer input used by channel 1 to the event input signal of EVENTC via Timer Input Select Register 0 (TIS0).

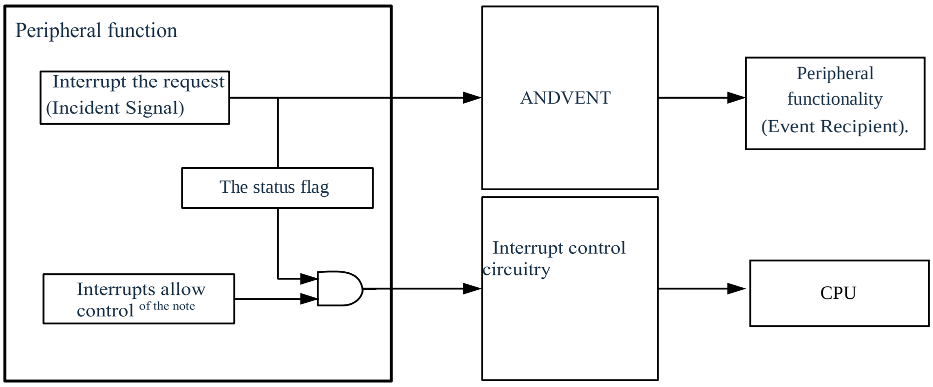
17.4 Operation of EVENTC

The path used by the event signal generated by each peripheral function as an interrupt request for the interrupt control circuit and the path used as an eventc event are independent of each other. Therefore, each event signal is independent of interrupt control and can be used as an event signal for the operation of peripheral functions of the event receiver.

The relationship between interrupt handling and EVENTC is shown in Figure 17-3. This figure takes the relationship between peripheral functions with interrupt request status flags and interrupt allow bits (which control whether to allow or disable) as an example.

The peripheral function that accepts an event through EVENTC operates according to the operation of the receiver peripheral function after receiving the event (refer to “Table 17-3 Correspondence between the setting value of ELSELRn register (n=00~14) and the operation when the link target peripheral function accepts the event

Figure 17-3 The relationship between interrupt processing and EVENTC


Peripheral function
Interrupt the request (Incident Signal)
The status flag
Interrupts allow control of the note
ANDVENT
Peripheral functionality (Event Recipient).
Interrupt control circuitry
CPU

Note Some peripheral features do not have this feature.

The response to the perimeter function that accepts the event is shown in Table 17-4.

Table 17-4 Response of the peripheral function of the received event

Event Acceptance Target No.	Function of event link target	Operation after event acceptance	Response
1	A/D converter	A/D conversion	The EVENTC event becomes a hardware trigger for A/D conversion directly.
2	Timer input for Timer40 channel 0	The delay counter enters the measurement external event counter for pulse width	Edge detection is performed after 3 or 4 f_{CLK} cycles from the occurrence of the EVENTC event.
3	Timer4 Timer input for 0 channel 1	The delay counter enters the measurement external event counter for pulse width	Edge detection is performed after 3 or 4 f_{CLK} cycles from the occurrence of the EVENTC event.
4	The EPWM output controls the truncation source	Forced cut-off of the pulse output	Becomes a forced cutoff state after 2 or 3 EPWM operating clock cycles from the occurrence of an EVENTC event.

Chapter 18 Interrupt Function

The Cortex-M0+ processor has a built-in Nested Vector Interrupt Controller (NVIC) that supports up to 32 interrupt request (IRQ) inputs, as well as one non-maskable interrupt (NMI) input, and multiple internal exceptions.

The interrupt source for 32 interrupt request (IRQ) inputs and 1 non-maskable interrupt (NMI) input is processed in this system. This user manual only describes the handling in this system, Cortex-M0+ processors built-in NVIC functions, please refer to the Cortex-M0+ processor user manual.

18.1 Types of interrupt function

There are two types of interrupt functions.

(1) Interrupts can be masked

This is a shielded controlled interrupt. If the interrupt mask flag register is not open, the interrupt request will not be responded to even if it is generated.

It can generate a standby release signal to cancel the deep sleep mode and sleep mode.

Masked interrupts are divided into external interrupt requests and internal interrupt requests.

(2) Interrupts cannot be masked

This is an unmasked interrupt that the CPU must respond to once the interrupt request is made.

18.2 Interrupt source and structure

Suspend source columns table reference Table 18-1.

Table 18-1 List of interrupt sources (1/3)

Interrupt handling	The source of the interrupt numbering	The source of the interrupt		Internal/External	Basic structure Type Note 1
		name	trigger		
Maskable	0	INTLVI	Voltage detection Note 2	interior	(A)
	1	INTP0	Detection of pin input edges	exterior	(B)
	2	INTP1	Detection of pin input edges		
	3	INTP2	Detection of pin input edges		
	4	INTP3	Detection of pin input edges		
	5	INTTM01H	The counting end of timer channel 01 or the end of the capture (when the high 8-bit timer is operating).	interior	(A)
	6	INTKR	Key interrupt		
	7	INTST2/ INTSSPI20/ INTIIC20	The end of the UART2 transmission or the end of the transmission of the buffer null interrupt/SSPI20 or the end of the transmission of the buffer null interrupt/IIC20		
	8	INTSR2/ INTSSPI21/ INTIIC21	The end of transmission received by UART2 / the end of transmission of SSPI21 or the end of transmission of buffer null interrupt/IIC21		
	9	INTSRE2	A communication error received by UART2 occurred		
	10	INTST0/ INTSSPI00/ INTIIC00	The end of the UART0 transmission or the end of the buffer null interrupt/SSPI00 or the end of the buffer null interrupt/IIC00 transfer		
	11	INTSR0/ INTSSPI01/ INTIIC01	The end of transmission received by UART0 / the end of transmission of SSPI01 or the end of transmission of buffer null interrupt/IIC01		
	12	INTSRE0	A communication error received by UART0 occurred		

Note 1. The basic composition types (A) to (C) correspond to Figure 18-1 (A)~(C).

2. This is when bit7 (LVIMD) of the voltage sense level register (LVIS) is set to “0”.

Table 18-1 List of interrupt sources (2/3)

Interrupt handling	Source of the interrupt numbering	Source of the interrupt		Internal/External	Basic structure Type Note 1
		Name	Trigger		
Maskable	13	INTST1/ INTSSPI10/ INTIIC10/ INTSPI	The end of transmission sent by UART1 or the end of transmission of buffer null interrupt/SSPI10 or the end of transmission of buffer null interrupt/IIC10/end of transmission of serial interface SPI interrupt	interior	(A)
	14	INTSR1/ INTSSPI11/ INTIIC11	The end of transmission received by UART1 / the end of transmission of SSPI11 or the end of transmission of buffer null interrupt/IIC11		
	15	INTSRE1	A communication error received by UART1 occurred		
	16	INTIICA0	IICA0 communication ends		
	17	INTTM00	The count end of timer channel 00 or the end of the snap		
	18	INTTM01	The counting end of timer channel 01 or the end of the snap		
	19	INTTM02	The count end of timer channel 02 or the end of the snap		
	20	INTTM03	The counting end of timer channel 03 or the end of the snap		
	21	INTAD	The A/D conversion ends		
	22	INTRTC	Fixed period/of the real-time clock Alarm clock consistent detection		
	23	INTIT	Detection of interval signals		
	24	INTOCRV	Internal high-speed oscillator correction function		
	25	retain			
	26	retain			
	27	INTTM10	Timer channel 1 0 ends in count or snap ends		
	28	INTTM11	Timer channel 1 1 ends the count or snaps ends		
	29	INTTM12	Timer channel 12 counts end or snap ends		
	30	INTTM13	Timer channel 13 counts end or snap ends		
	31	INTFL	Flash programming is over		

Note 1 The basic composition types (A) to (C) correspond to Figure 18-1 (A)~(C).

Table 18-1 List of interrupt sources (3/3)

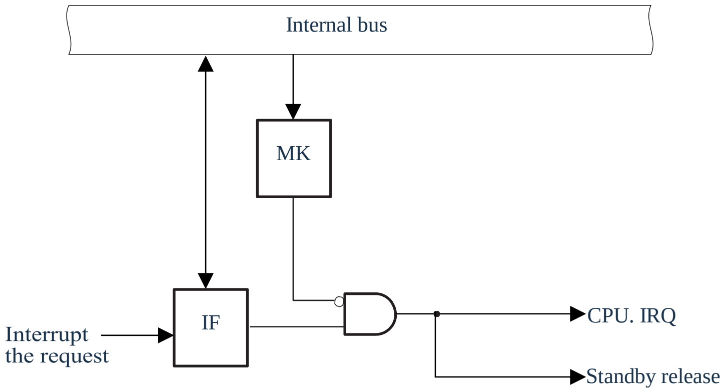
Interrupt handling	Interrupt source number	Source of the interrupt		Internal/External	Note 1 to the basic
		Name	Trigger		
Not blockable		INTWDT	Watchdog timer interval interrupt Note 2	interior	(C)

Note 1 The basic composition types (A) to (C) correspond to Figure 18-1 (A)~(C).

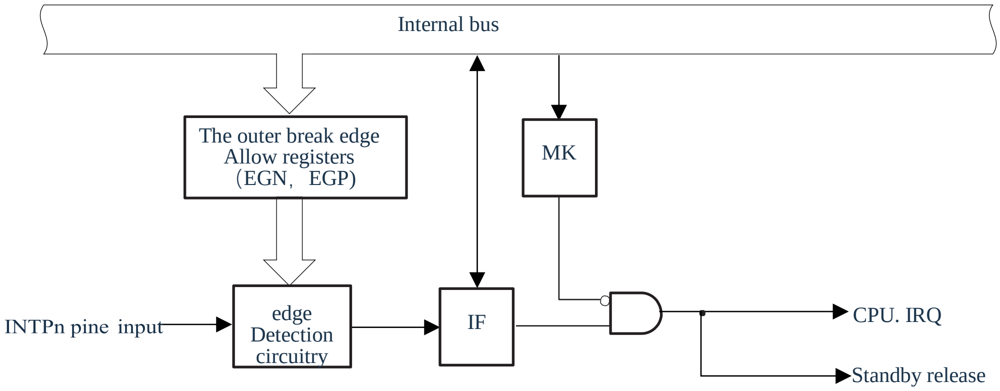
2. This is the case where bit7 (WDTINT) of option byte (000C0H) is set to “1”.

Figure 18-1　　Basic structure of the interrupt function

(A) Internally maskable interrupts

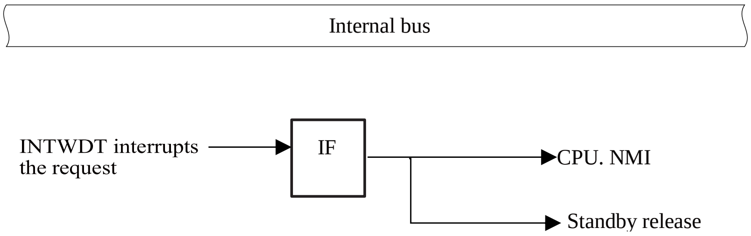


(B)Externally maskable interrupt (INTPn)



Note　n=0~3

(C) Non-maskable interrupts



Note: The interrupt request flag for non-maskable interrupts has no entity registers and cannot generate interrupt requests through bus read and write registers.

18.3 Registers controlling interrupt function

The interrupt function is controlled by the following four registers.

- Interrupt request flag register (IF00~IF31).
- Interrupt mask flag register (MK00~MK31).
- External interrupt rising edge enable register (EGP0).
- External interrupt falling edge enable register (EGN0).

18.3.1 Interrupt request flag registers (IF00 to IF31)

By incurring a corresponding interrupt request or executing instructions, the interrupt request flag is set to “1”.

By generating a reset signal or executing an instruction, the interrupt request flag is clear to “0”.
The IF00L to IF31L registers are set via 8-bit memory operation instructions
Or set the IF00~IF31 registers via 32-bit memory operation instructions.
After the reset signal is generated, the values of these registers become “00 00_0000H”.

Figure 18-2 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_0000HR/W

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

IFmL	The interrupt request flag for the interrupt source numbered 0 to 31
0	No interrupt request signal is generated.
1	An interrupt request is generated and is in the interrupt request state.

Note: 1. The correspondence between the interrupt source and the interrupt request flag register is shown in Table 18-2

2. The correspondence between the interrupt request flag register and CPU.IRQ is shown in Figure 18-4

18.3.2 Interrupt mask flag register (MK00~MK31)

The interrupt masking flag setting allows or disables the corresponding maskable interrupt processing.

Set the MK00L~MK31L registers via 8-bit memory operation instructions or set MK00~MK31 registers via 32-bit memory operation instructions.

After the reset signal is generated, the values of these registers become “FFFF_FFFF”.

Figure 18-3 Format of interrupt request masking register (MKm) (m=0~31)

address:MK00:40006100H,MK01:40006104H,MK02:40006108H,MK03:4000610CH
MK04:40006110H,MK05:40006114H,MK06:40006118H,MK07:4000611CH
MK08:40006120H,MK09:40006124H,MK10:40006128H,MK11:4000612CH
MK12:40006130H,MK13:40006134H,MK14:40006138H,MK15:4000613CH
MK16:40006140H,MK17:40006144H,MK18:40006148H,MK19:4000614CH
MK20:40006150H,MK21:40006154H,MK22:40006158H,MK23:4000615CH
MK24:40006160H,MK25:40006164H,MK26:40006168H,MK27:4000616CH
MK28:40006170H,MK29:40006174H,MK30:40006178H,MK31:4000617CH
Reset value: FFFF_FFFFHR/W

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

MKmL	Interrupt handling control for interrupt sources numbered 0 to 31Note 1
0	Interrupt handling is allowed.
1	Interrupt processing is prohibited.

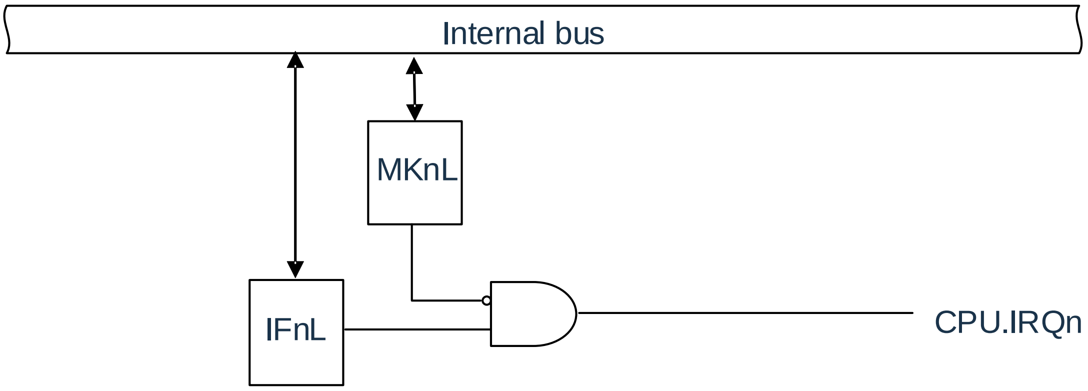
Note: 1. The correspondence between the interrupt source and the interrupt request masking register is shown in Table 18-2

2. The correspondence between the interrupt request flag register and CPU.IRQ is shown in Figure 18-4

Table 18-2 Relationship between interrupt sources and flag registers

Number	Source of the interrupt	Interrupt request flag register	Interrupt mask flag register
0	INTLVI	IF00. IFL	MK00. MKL
1	INTP0	IF01.IFL	MK01.MKL
2	INTP1	IF02.IFL	MK02.MKL
3	INTP2	IF03.IFL	MK03.MKL
4	INTP3	IF04.IFL	MK04.MKL
5	INTTM01H	IF05.IFL	MK05.MKL
6	INTKR	IF06.IFL	MK06.MKL
7	INTST2/INTSSPI20/INTIIC20	IF07.IFL	MK07.MKL
8	INTSR2/INTSSPI21/INTIIC21	IF08.IFL	MK08.MKL
9	INTSRE2	IF09.IFL	MK09.MKL
10	INTST0/INTSSPI00/INTIIC00	IF10.IFL	MK10.MKL
11	INTSR0/INTSSPI01/INTIIC01	IF11.IFL	MK11.MKL
12	INTSRE0	IF12.IFL	MK12.MKL
13	INTST1/INTSSPI10/INTIIC10	IF13.IFL	MK13.MKL
14	INTSR1/INTSSPI11/INTIIC11	IF14.IFL	MK14.MKL
15	INTSRE1	IF15.IFL	MK15.MKL
16	INTIICA0	IF16.IFL	MK16.MKL
17	INTTM00	IF17.IFL	MK17.MKL
18	INTTM01	IF18.IFL	MK18.MKL
19	INTTM02	IF19.IFL	MK19.MKL
20	INTTM03	IF20.IFL	MK20.MKL
21	INTAD	IF21.IFL	MK21.MKL
22	INTRTC	IF22.IFL	MK22.MKL
23	INTKR	IF23.IFL	MK23.MKL
24	INTCMP0	IF24.IFL	MK24.MKL
25	INTCMP1	IF25.IFL	MK25.MKL
26	INTRAMPRTERR	IF26.IFL	MK26.MKL
27	INTTM10	IF27.IFL	MK27.MKL
28	INTTM11	IF28.IFL	MK28.MKL
29	INTTM12	IF29.IFL	MK29.MKL
30	INTTM13	IF30.IFL	MK30.MKL
31	INTFL	IF31.IFL	MK31.MKL

Figure 18-4 Relationship between each flag register and CPU.IRQ



18.3.3 External interrupt rising edge enable register (EGP0), External interrupt falling edge enable register (EGN0)

These registers set the effective edges of INTP0 to INTP3.
Set the EGP0 and EGN0 registers via 8-bit memory operation instructions.
After the reset signal is generated, the values of these registers become “00H”.

Figure 18-5 Format of external interrupt rising edge enable register (EGP0), external interrupt falling edge enable register (EGN0)

Address: 40045B38H After reset: 00HR/W

symbol	7	6	5	4	3	2	1	0
EGP0	0	0	0	0	EGP3	EGP2	EGP1	EGP0

Address: 40045B39H after reset: 00HR/W

symbol	7	6	5	4	3	2	1	0
EGN0	0	0	0	0	EGN3	EGN2	EGN1	EGN0

EGPn	EGNn	Valid edge selection for the INTPn pin (n=0~11)
0	0	Disable detection of edges.
0	1	Falling edge
1	0	Rising edge
1	1	Rising and falling edges

The ports corresponding to the EGPn bit and the EGNn bit are shown in Table 18-3.

Table 18-3 Interrupt request signals corresponding to the EGPn and EGNn bits

Detect enable bits		Interrupt request signal
EGP0	EGN0	INTP0
EGP1	EGN1	INTP1
EGP2	EGN2	INTP2
EGP3	EGN3	INTP3

Note If you switch the input port used by the external interrupt function to output mode, a valid edge may be detected and an INTPn interrupt may be generated. When switching to output mode, the port mode register (PMxx) must be set to “0” after the disable detection edge (EGPn, EGNn=0, 0).

Note 1. For ports detected by edge, refer to “2.1 Port Functions”.

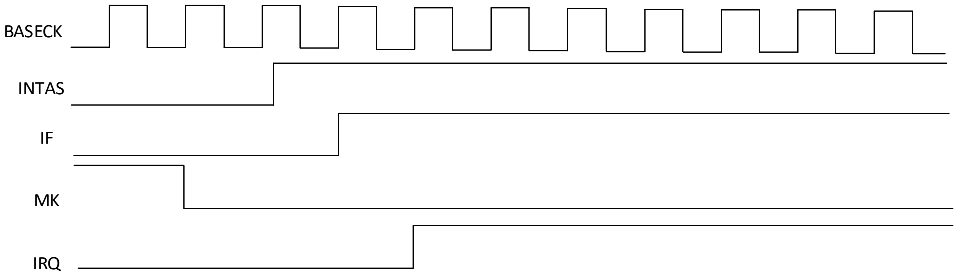
2.n=0~3

18.4 Operation of interrupt handling

18.4.1 Acceptance of maskable interrupt requests

If the interrupt request flag is set to “1” and the masked (MK) flag for the interrupt request is cleared “0”, it enters a state that accepts maskable interrupt requests and can pass the interrupt request to NVIC.

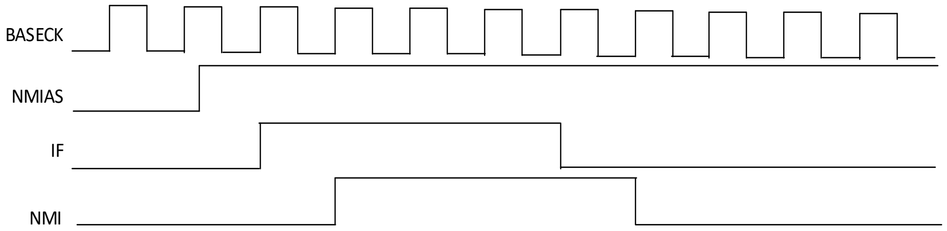
From setting the interrupt request flag to 1 to setting the IRQ of the CPU to 1, only 1 clock is required.



18.4.2 Acceptance of non-maskable interrupt requests

If a non-maskable interrupt request is generated, the interrupt request flag is set to “1” and passed directly to NVIC.

From the interrupt request flag being set to 1 to the CPU's NMI being set to 1, only 1 clock is required.



Chapter 19 Key Interrupt Function

The number of channels entered by key interrupt varies by product.

19.1 Function of key interrupt

A key interrupt (INTKR) can be generated by giving the key interrupt input pin (KR0 to KR5) on the falling edge of the input.

Table 19-1 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

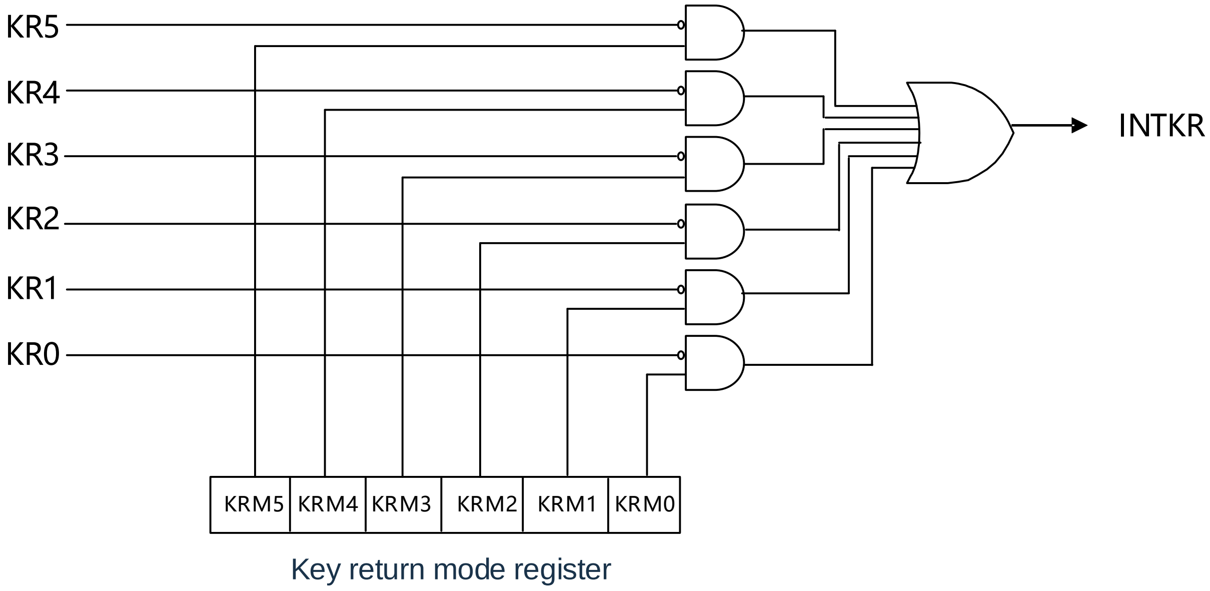
19.2 Structure of key interrupt

The key interrupt consists of the following hardware.

Table 19-2 Structure of key interrupts

Item	Control registers
Control registers	Key return mode register (KRM) Port mode register (PMx). Port mode control register (PMCx).

Figure 19-1 Diagram of the key interrupt



19.3 Registers for controlling key interrupt

Interrupt function via the following register control keys.

- Key return mode register (KRM).
- Port mode register (PMx).

19.3.1 Key return mode register (KRM)

KRM0~KRM5-bit control KR0~KR5 signal.

The KRM registers are set via 8-bit memory operation instructions.

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

Figure 19-2 Format of mode register (KRM)

Address: 40044B37H after reset: 00HR/W

Symbol	7	6	5	4	3	2	1	0
KRM	0	0	KRM5	KRM4	KRM3	KRM2	K RM1	KRM0

KRMn	Control of key interrupt mode
0	The key interrupt signal is not detected.
1	Detects key interrupt signals.

Note 1. The internal pull-up resistor can be used by setting the object bit of the pull-up resistor register (PUx) of the key interrupt input pin to "1".

2. An interrupt occurs if the object position bit of the KRM register is entered low at the input pin of the key interrupt. To ignore this interrupt, the KRM register must be set after the interrupt handling is disabled by the interrupt masking flag. The interrupt request flag must then be cleared after waiting for the key interrupt input to be low level width (tKR) (see data sheet) to allow interrupt handling.

3. Pins that are not used in key interrupt mode can be used as usual ports.

Remark 1. n=0~5

19.3.2 Port mode register (PMx)

When used as a key interrupt input pin (KR0~KR5), the PMCxn bit must be set to “0” and the PMxn bit must be set to “1” respectively. In this case, the output latch of Pxn can be “0” or “1”.

The PM x register is set via an 8-bit memory operation command.

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

An internal pull-up resistor can be used in bits using a pull-up resistor select register (PU x).

For the format of the port mode registers, refer to “2.3.1 Port Mode Registers (PMxx)”.

Chapter 20 Standby Function

20.1 Standby function

The standby function is a function that further reduces the operating current of the system, and there are two modes.

(1) Sleep mode

Sleep mode is the mode that stops the CPU from running the clock. Each clock continues to oscillate before setting the sleep mode, such as if the high-speed system clock oscillation circuit, the high-speed internal oscillator, or the subsystem clock oscillation circuit is oscillating. Although this mode does not allow the operating current to drop to the level of deep sleep mode, it is an effective mode when you want to restart processing immediately with an interrupt request or if you want to do intermittent operation frequently.

(2) Deep sleep mode

Deep sleep mode is a mode that stops the oscillation of the high-speed system clock oscillation circuit and the high-speed internal oscillator and stops the entire system. It can greatly reduce the operating current of the CPU.

Because deep sleep mode can be released by interrupt requests, it can also be run intermittently. However, in the case of the X1 clock, because the wait time to ensure oscillation stability is required when the deep sleep mode is released, if you need to start processing immediately with an interrupt request, you must choose the sleep mode.

In deep sleep mode, except for partial power loss, registers, flags, and data memory all remain what was before they were set to standby mode, and the output latches and output buffers of the input/output ports are also maintained.

Note 1 Deep sleep mode can only be used when the CPU is running at the main system clock. When the CPU is running on the secondary system clock, it cannot be set to deep sleep mode. Sleep mode can be used regardless of whether the CPU is running on the primary system clock or the secondary system clock.

2. When transferring to deep sleep mode, WFI instructions must be executed after stopping peripheral hardware running at the main system clock.

3. To reduce the operating current of the A/D converter, the A/D converter mode register 0 (ADM0) must be placed at bit7 (ADCS) and bit0 (ADCE) clear “0” and execute the WFI instruction after stopping the A/D conversion operation.

4. The option byte allows you to choose whether to continue or stop oscillating the low-speed internal oscillator in sleep mode or deep sleep mode. For details, please refer to Chapter 26 Option Bytes.

20.2 Sleep mode

20.2.1 Setting of the sleep mode

When the SLEEPDEEP bit of the SCR register is 0, the WFI instruction is executed and sleep mode is entered. In sleep mode, the CPU stops operating, but the values of the internal registers are still maintained and peripheral modules remain in the state they were in before they entered sleep mode. The status of peripheral modules, oscillators, etc. in sleep mode is shown in Table 20-1

Sleep mode can be set regardless of whether the CPU clock before setting is a high-speed system clock, a high-speed internal oscillator clock, or a subsystem clock.

Note When the interrupt mask flag is “0” (allow interrupt processing) and the interrupt request flag is “1” (generating an interrupt request signal), the interrupt request signal is used to release sleep mode. Therefore, even when WFI instructions are performed in this case, it is not transferred to sleep mode.

Table 20-1 Operation status in sleep mode (1/2)

Setting of the sleep mode Item			Execution of WFI instructions while the CPU is running at the main system clock		
			CPU with high speed internal oscillator clock (F_{IH}) run	CPU runs on X1 clock (f_X).	CPU with external master system clock (F_{EX}) run
System clock			Stop supplying clocks to the CPU.		
	Main system clock	f_{IH}	Continue to run (cannot be stopped).	Disable operation.	
		f_X	Disable operation.	Continue to run (cannot be stopped).	Cannot run.
		f_{EX}		Cannot run.	Continue to run (cannot be stopped).
	Subsystem Clock	f_{XT}	Remains in the state it was in before it was set to sleep mode.		
		f_{EXS}			
	Low-speed internal oscillation Clock of the device	f_{II}	Bit0 (WDSTBYON) and bit4 (WDTON) and the secondary system clock via option bytes (000C0H) are available Allows the WUTMMCK0 bit of the mode control register (OSMC) to be set. WUTMMCK0=1: Oscillation WUTMMCK0=0 and WDTON=0: Stop WUTMMCK0=0, WDTON=1 and WDSTBYON=1: Oscillation WUTMMCK0=0, WDTON=1 and WDSTBYON=0: Stop		
CPU			Stop running.		
Code flash					
RAM			Stop running (can run when DMA is executed).		
Port (latch)			Remains in the state it was in before it was set to sleep mode.		
Universal timer unit			Can run.		
Real-time clock (RTC).					
1 5-bit interval timer					
Watchdog timer			See “Chapter 10: Watchdog Timer“.		
Clock output/buzzer output			Can run.		
A/D converter					
Universal Serial Communication Unit (SCI).					
Serial Interface (IICA).					
Data Transfer Controller (DMA).					
Linkage controller			Links can be made between runnable function blocks.		
Power-on reset function			Can run.		
Voltage detection function					
External interrupts					
CRC operation	High-speed CRC				

function	General CRC	When DMA is executed in the operation of the RAM area, it can be run.
RAM parity function		It can run when performing DMA.
SFR protection function		

Note Stop running: Automatically stops running when transferred to sleep mode.

Disable Running: Stop running before moving to sleep mode.

f_{IH}: High Speed Internal Oscillator Clock fIL: Low Speed Internal Oscillator Clock

fX: X1 clock fEX: External master system clock

fXT: XT1 clock fEXS: External subsystem clock

Table 20-1 Operation status in sleep mode (2/2)

Setting of the sleep mode 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 supplying clocks to the CPU.	
	Main system clock	f_{IH}	Disable operation.	
		f_X		
		f_{EX}		
	Subsystem clock	f_{XT}	Continue to run (cannot be stopped).	Cannot run.
		f_{EXS}	Cannot run.	Continue to run (cannot be stopped).
	Low-speed internal oscillation Clock of the device	f_{Il}	Mode control registers (OSMC) are provided via bit0 (WDSTBYON) and bit4 (WDTON) of option bytes (000C0H) and the subsystem clock WUTMMCK0 bit is set. • WUTMMCK0=1: Oscillation • WUTMMCK0=0 and WDTON=0: Stop • WUTMMCK0=0, WDTON=1 and WDSTBYON=1: Oscillation • WUTMMCK0=0, WDTON=1 and WDSTBYON=0: Stop	
CPU			Stop running.	
Code flash				
RAM			Stop running (can run when DMA is executed).	
Port (latch)			Remains in the state it was in before it was set to sleep mode.	
Universal timer unit			When RTCLPC=0, it can run (otherwise it is disabled).	
Real-time clock (RTC).			Can run.	
15-bit interval timer				
Watchdog timer			See “Chapter 10: The Watchdog Timer“.	
Clock output/buzzer output			When RTCLPC=0, it can run (otherwise it is disabled).	
A/D converter			Disable operation.	
Universal Serial Communication Unit (SCI)			When RTCLPC=0, it can run (otherwise it is disabled).	
Serial Interface (IICA).			Disable operation.	
Data Transfer Controller (DMA).			When RTCLPC=0, it can run (otherwise it is disabled).	
Linkage controller			Links can be made between runnable function blocks.	
Power-on reset function			Can run.	
Voltage detection function				
External interrupts				
CRC operations function	High-speed CRC		Disable operation.	
	Generic CRC		When DMA is executed in the operation of the RAM area, it can be run.	
RAM parity error detection function			It can run when performing DMA.	
SFR protection function				

Note Stop running: Automatically stops running when transferred to sleep mode.

Disable Run: Stop running before moving to sleep mode.

f_{IH}: High Speed Internal Oscillator Clock fIL: Low Speed Internal Oscillator Clock

fX: X1 clock fEX: External master system clock

fXT: XT1 clock fEXS: External subsystem clock

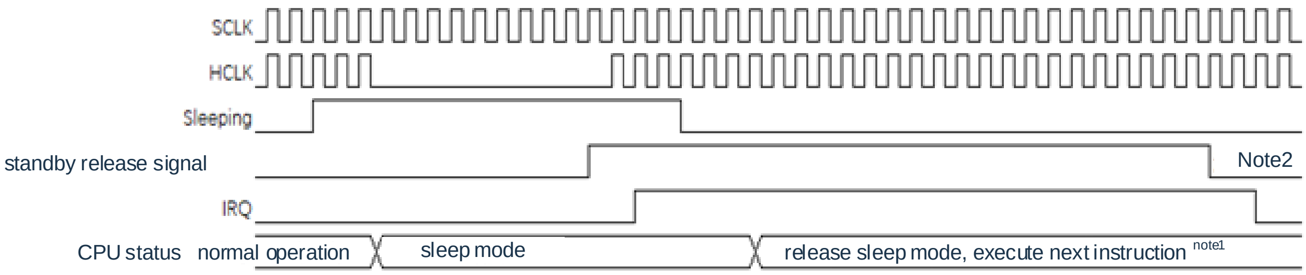
20.2.2 Release of sleep mode

Sleep mode can be interrupted with any interrupt as well as an external reset terminal, POR reset, low voltage sense reset, RAM parity error reset, WDT reset, and software reset to be released.

(1) Release by interrupting

When an unmasked interrupt is generated and it is in a state that allows interrupts to be accepted, sleep mode is released and the CPU begins processing the interrupt service program.

Figure 20-1 Release sleep mode by interrupting requests



Note 1. From the standby dismissal signal generation to the sleep mode dismissal, it takes 16 clocks to start the interrupt service program.

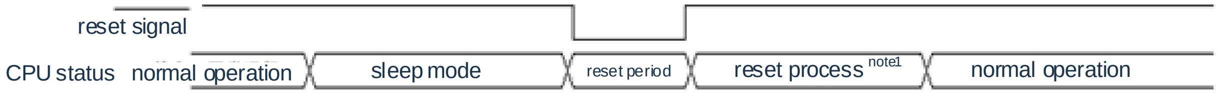
2. The standby release signal cannot be cleared by itself, and the register must be cleared. Register clearing is usually written in an interrupt service program.

Note: Before entering sleep mode, only the mask bit that is expected to be used to release the interrupt in sleep mode should be cleared.

(2) Release by reset

When a reset signal is generated, the CPU is reset and sleep mode is released. As with the usual reset, the program is executed after transfer to the reset vector address.

Figure 20-2 Release sleep mode by reset



Note 1: For reset processing, please refer to “Chapter 21 Reset Function”. For the reset of power-on reset (POR) circuits and voltage detection (LVD) circuits, refer to Chapter 2, Power-On Reset Circuits.

20.3 Deep sleep mode

20.3.1 The setting for deep sleep mode

When the SLEEPDEEP bit of the SCR register is 1, the WFI instruction is executed and deep sleep mode is entered. In this mode, the CPU, most peripheral modules, and the oscillator stop functioning. However, the values of the CPU's internal registers, RAM data, peripheral modules, and I/O status are maintained. The operating status of the peripheral module and the oscillator in deep sleep mode is shown in Table 20-2.

Deep sleep mode can only be set if the CPU clock before setting is the primary system clock.

Note When the interrupt mask flag is “0” (allow interrupt processing) and the interrupt request flag is “1” (generates 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 dismissed as soon as it enters deep sleep mode. Returns to run mode after executing the WFI instruction and after the deep sleep mode is released.

Table 20-2 Operating status in deep sleep mode

Setting of the deep sleep mode Item			Execution of WFI instructions while the CPU is running at the main system clock		
			CPU runs on a high-speed internal oscillator clock (F_{IH})	CPU runs on X1 clock (f_X).	CPU runs on an external main system clock (F_{EX})
System clock			Stop supplying clocks to the CPU.		
	Main system clock	f_{IH}	Stop it		
		f_X			
		f_{EX}			
	Subsystem clock	f_{XT}	Remains in the state it was in before it was set to deep sleep mode.		
		f_{EXS}			
	f_{IL}		Bit0 (WDSTBYON) and bit4 (WDTON) and the secondary system clock via option bytes (000C0H) are available Allows the WUTMMCK0 bit of the mode control register (OSMC) to be set. WUTMMCK0=1: Oscillation WUTMMCK0=0 and WDTON=0: Stop WUTMMCK0=0, WDTON=1 and WDSTBYON=1: Oscillate WUTMMCK0=0, WDTON=1 and WDSTBYON=0: Stop		
CPU			Stop running.		
Code flash					
RAM					
Port (latch)			Remains in the state it was in before it was set to deep sleep mode.		
Universal timer unit			Disable operation.		
Real-time clock (RTC).			Can run.		
1 5-bit interval timer					
Watchdog timer			See “Chapter 10: The Watchdog Timer“.		
Clock output/buzzer output			Capable of running when the secondary system clock is selected as the count clock and the RTCLPC bit is “0”, otherwise it is disabled.		
A/D converter			Can wake up.		
Universal Serial Communication Unit (SCI)			Only SSPIp and UARTq can wake up. Except for SSPIp and UARTq, it is forbidden to run.		
SPI			Disable operation.		
Serial Interface (IICA).			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.		
Power-on reset function			Can run.		
Voltage detection function					
External interrupts					
CRC operations function	High-speed CRC		Stop running.		
	Generic CRC				

RAM parity function	
SFR protection function	

Note Stop Running: Automatically stops running when transferred to deep sleep mode.

Disable Run: Stop running before moving to deep sleep mode.

f_{IH}	: High-speed internal oscillator clock	f_{Il}	: Low-speed internal oscillator clock
f_{X}	: X1 clock	f_{EX}	: External master system clock
f_{XT}	: XT1 clock	f_{EXS}	: External subsystem clock

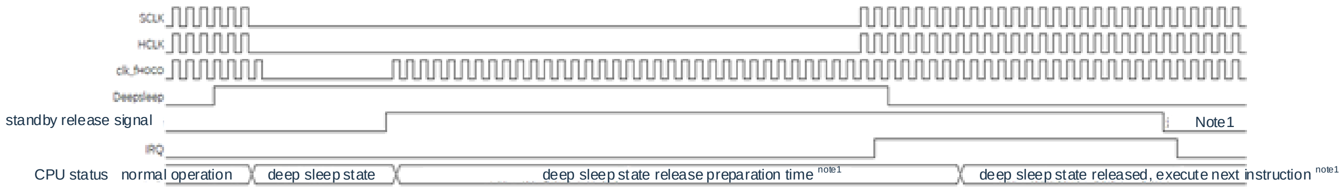
20.3.2 Release of deep sleep mode

Deep sleep mode can be released in the following two ways.

(1) Release through an unmasked interrupt request

If an unmasked interrupt request occurs, deep sleep mode is released. After the oscillation settling time, the deep sleep mode is released and the CPU begins to process the interrupt service program.

Figure 20-3 Release deep sleep mode by interrupting requests


SCLK
HCLK
clk_fHOCO
Deepsleep
standby release signal
IRQ
Note1
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 more information on standby release signals, refer to “Figure 20-1 Basic Structure of the Interrupt Function“.

2. Deep sleep state release preparation time:

When the CPU clock is a high-speed internal oscillating clock or an external clock input before entering deep sleep mode: at least 20us

When the CPU clock is a high-speed system clock (X1 oscillation) before entering deep sleep mode: at least 20us with a longer oscillation settling time (set via OSTS).

3. Wait: From CPU The IRQ is valid until the interrupt service program is started, which takes 14 clocks.

Note: 1. Before entering sleep mode, only the mask bit corresponding to the interrupt that is expected to be used to release sleep mode should be cleared.

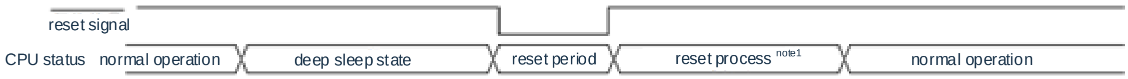
2. When the CPU is running at a high-speed system clock (X1 oscillation) and to reduce the oscillation settling time after the deep sleep mode is released, the CPU clock must be temporarily switched to a high-speed internal oscillator clock before executing WFI instructions.

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

(2) Release by generating a reset signal

Deep sleep mode is released by generating a reset signal. Then, as with the usual reset, execute the program after transferring to the reset vector address.

Figure 20-4 Release the deep sleep mode by resetting



Note For reset processing, please refer to “Chapter 21 Reset Function“. For the reset of power-on reset (POR) circuits and voltage detection (LVD) circuits, refer to Chapter 2 Power-On Reset Circuits.

Chapter 21 Reset Function

The following 7 methods generate a reset signal.

(1) An external reset is entered via the RESETB pin.
(2) An internal reset is generated by a program runaway detection of the watchdog timer.
(3) An internal reset is generated by comparing the supply voltage to the sense voltage of the power-on reset (POR) circuit.
(4) An internal reset is generated by comparing the supply voltage of the voltage detection circuit (LVD) with the sense voltage.
(5) Request register bit due to system reset (AIRCR. SYSRESETREQ) is set to 1 to produce an internal reset.
(6) Internal reset due to RAM parity error.
(7) Internal reset due to access to illegal memory.

Internal reset is the same as external reset, and after a reset signal is generated, the program is executed from the user-defined program start address.

When a low level is entered into the RESET B pin, or the watchdog timer detects a program runaway, or detects the voltage of the POR circuit and the LVD circuit, or the system reset request bit is assessed, or a RAM parity error occurs, or the illegal memory is accessed, A reset is generated and each hardware becomes a state as shown in Table 21-1.

Note 1 During an external reset, a low level of at least 10us must be entered into the RESETB pin. If an external reset is performed when the supply voltage rises, the supply must be turned on after the RESETB pin is low and maintained at least 10u over the operating voltage range shown in the AC characteristics of the user manual s low level, then enter high.

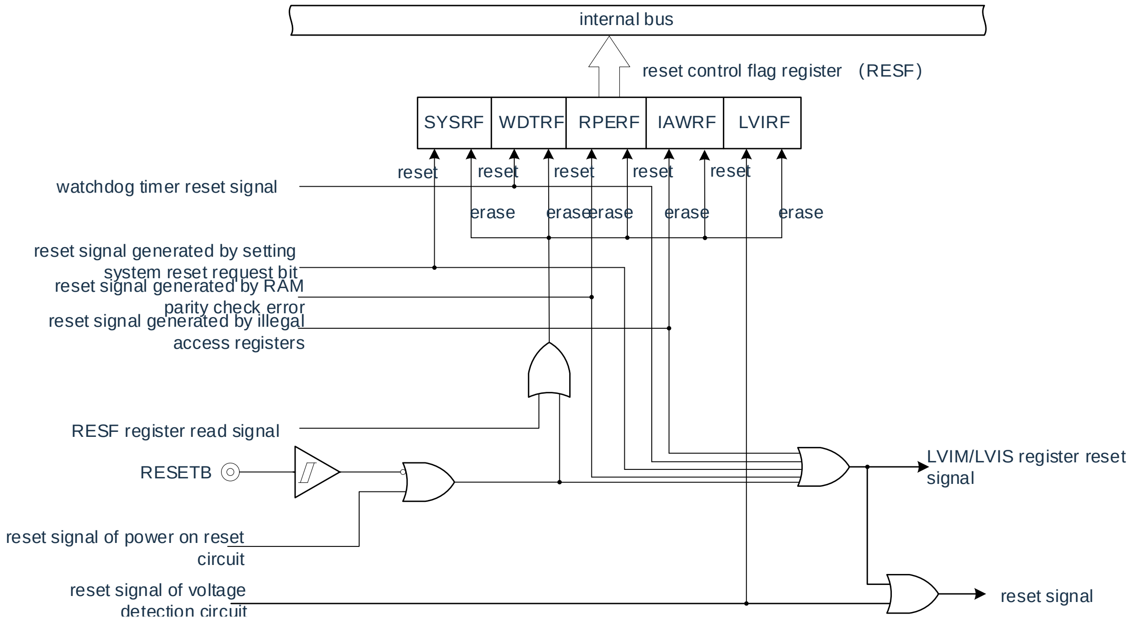
2. Stop oscillating the X1 clock, XT1 clock, high-speed internal oscillator clock, and low-speed internal oscillator clock during the reset signal. The inputs to the external master system clock and external subsystem clock are invalid.

3. If a reset occurs, each SFR is initialized so that the port pins become the following state:

• P10, P26, P 40, P137: High impedance during external reset or POR reset. High during other resets and after receiving the reset (internal pull-up resistors are connected).

• Ports other than P10, P26, P 40, P137: High impedance during and after receiving a reset.

Figure 21-1 Block diagram of 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

Note that the internal reset of the LVD circuit does not reset the LVD circuit.

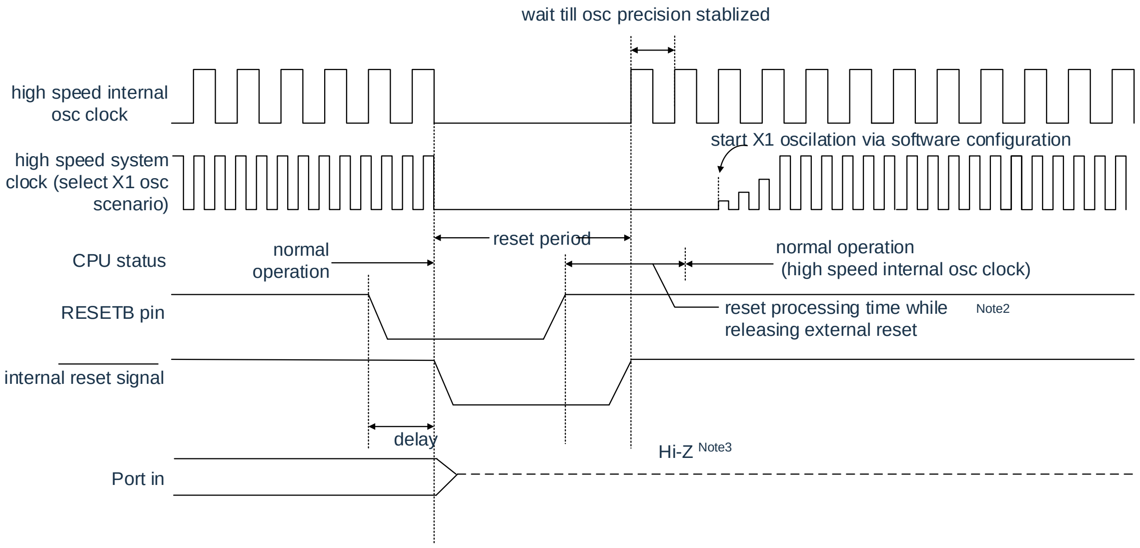
Note 1. LVIM: Voltage detection register

2. LVIS: Voltage detection level register

Reset timing

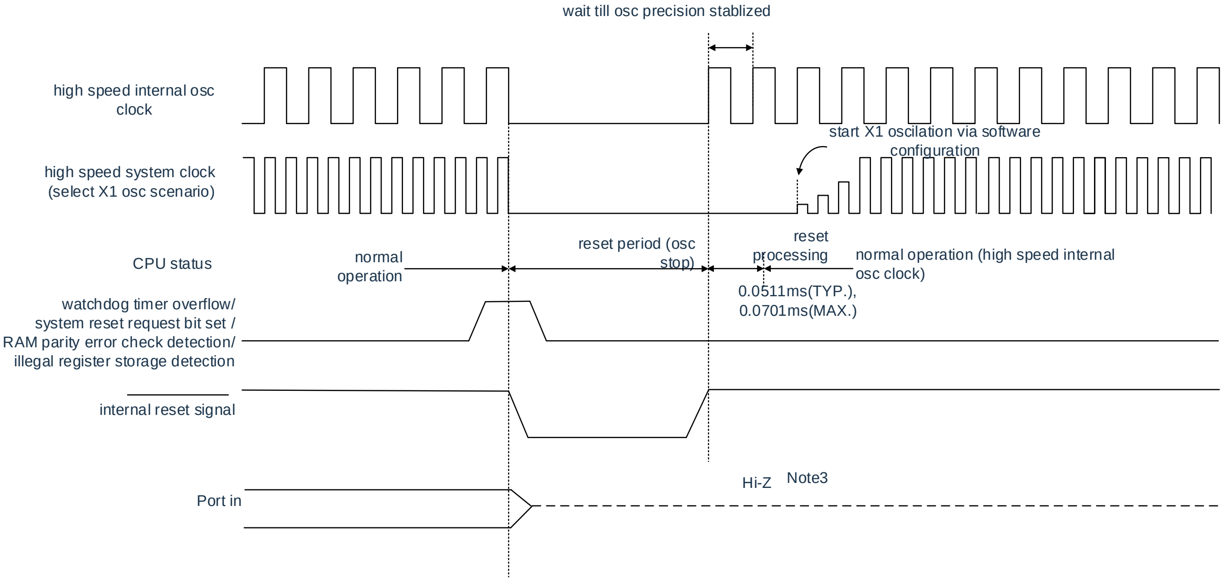
When low is input to the RESETB pin, a reset is generated. Then, if RESETB is quoted high, the reset state is released, and execution begins with a high-speed internal oscillator clock after the reset process is complete.

Figure 21-2 Timing of the 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 Note3
Port in

For resets caused by watchdog timer overflow, system reset request bit assertion, RAM parity error detection, or detection of illegal memory access, the reset state is automatically released and execution begins with a high-speed internal oscillator clock after the reset process is completed.

Figure 21-3 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)
0.0511ms(TYP.), 0.0701ms(MAX.)
watchdog timer overflow/ system reset request bit set / RAM parity error check detection/ illegal register storage detection
internal reset signal
Hi-Z Note3
Port in

Note 1 Port pins P10, P26, P40, P137 become the following states:

- High impedance during external reset or POR reset.
- High during other resets and after receiving the reset (internal pull-up resistor is connected).

Note that the watchdog timer is no exception, resetting when an internal reset occurs.

For resets generated by voltage sensing of POR circuits and LVD circuits, if the VDD≥ V POR or VDD≥ is satisfied after the reset VLVD is released from the reset state and execution begins with a high-speed internal oscillator clock after the reset process. For details, please refer to “Chapter 24 Power-on Reset Circuit” and “Chapter 23 Voltage Detection Circuit”.

Note V_{POR}: The POR supply voltage rises to detect the voltage
V_{LVD}: LVD sense voltage

Table 21-1 Operational status during reset

Item			During reset
System clock			Stop supplying clocks to the CPU.
	The master system clock	f_{IH}	Stop running.
		f_X	Stops operation (pins X1 and X2 are in input port mode).
		f_{EX}	The clock input is invalid (the pin is in input port mode).
	Auxiliary system clock	f_{XT}	Can run.
		f_{EXS}	The clock input is invalid (the pin is in input port mode).
	f_{II}		Stop running.
CPU			
Code flash			Stop running.
RAM			Stop running.
Port (latch)			High Impedance Note 1
Universal timer unit			Stop running.
Real-time clock (RTC).			
1 5-bit interval timer			
Watchdog timer			
Clock output/buzzer output			
A/D converter			
Universal Serial Communication Unit (SCI)			
Serial Interface (IICA).			
Data Transfer Controller (DMA).			
Power-on reset function			It can perform detection runs.
Voltage detection function			It can be operated when the LVD is reset. In other resets, stop running.
External interrupts			Stop running.
Key interrupt function			
CRC operation function	High-speed CRC		
	Generic CRC		
RAM parity function			
SFR protection function			

Note 1 Port pins P 10, P26, P 40, P137 become the following states:
High impedance during an external reset or POR reset. High during other reset periods (internal pull-up resistor is connected).

Remark				
Remark	f_{IH}	: High-speed internal oscillator clock	f_X	: X1 oscillating clock
	f_{EX}	: External master system clock	f_{XT}	: XT1 oscillating clock
	f_{EXS}	: External subsystem clock	f_{II}	: Low-speed internal oscillator clock

21.1 Register for confirming the reset source

21.1.1 Reset control flag register (RESF)

The CMS32L051 microcontroller has multiple internal reset sources. The Reset Control Flag Register (RESF) holds the reset source where the reset request occurred. RESF registers can be read via 8-bit memory operation instructions.

SYSRF, WDTRF, RPERF, are cleared by the input of RESETB, the reset of the power-on reset (POR) circuit, and the reading of the RESF register IAWRF, LVIRF logo. To determine the reset source, the value of the RESF register must be saved to any RAM and then judged by its RAM value.

Figure 21-4 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
FRSR	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 requests were made or the RESF registers were cleared.
1	An internal reset request is generated.

WDTRF	The watchdog timer (WDT) generates an internal reset request
0	No internal reset requests were made or the RESF registers were cleared.
1	An internal reset request is generated.

RPERF	An internal reset request is generated by a RAM parity error
0	No internal reset requests were made or the RESF registers were cleared.
1	An internal reset request is generated.

IAWRF	Internal reset requests generated by access illegal memory
0	No internal reset requests were made or the RESF registers were cleared.
1	An internal reset request is generated.

LVIRF	An internal reset request generated by a voltage sense circuit (LVD)
0	No internal reset requests were made or the RESF registers were cleared.
1	An internal reset request is generated.

Note 1 Varies depending on the reset source. Please refer to Table 21-2.

Note In the case of allowing RAM parity error reset (RPERDIS=0), the “RAM area” must be initialized when accessing data. When executing instructions from the RAM area, the area of “used RAM area + 10 bytes” must be initialized. By generating a reset, it enters a state that allows RAM parity error reset (RPERDIS=0). For more information, see “26.3.3 RAM Parity Error Detection Function”.

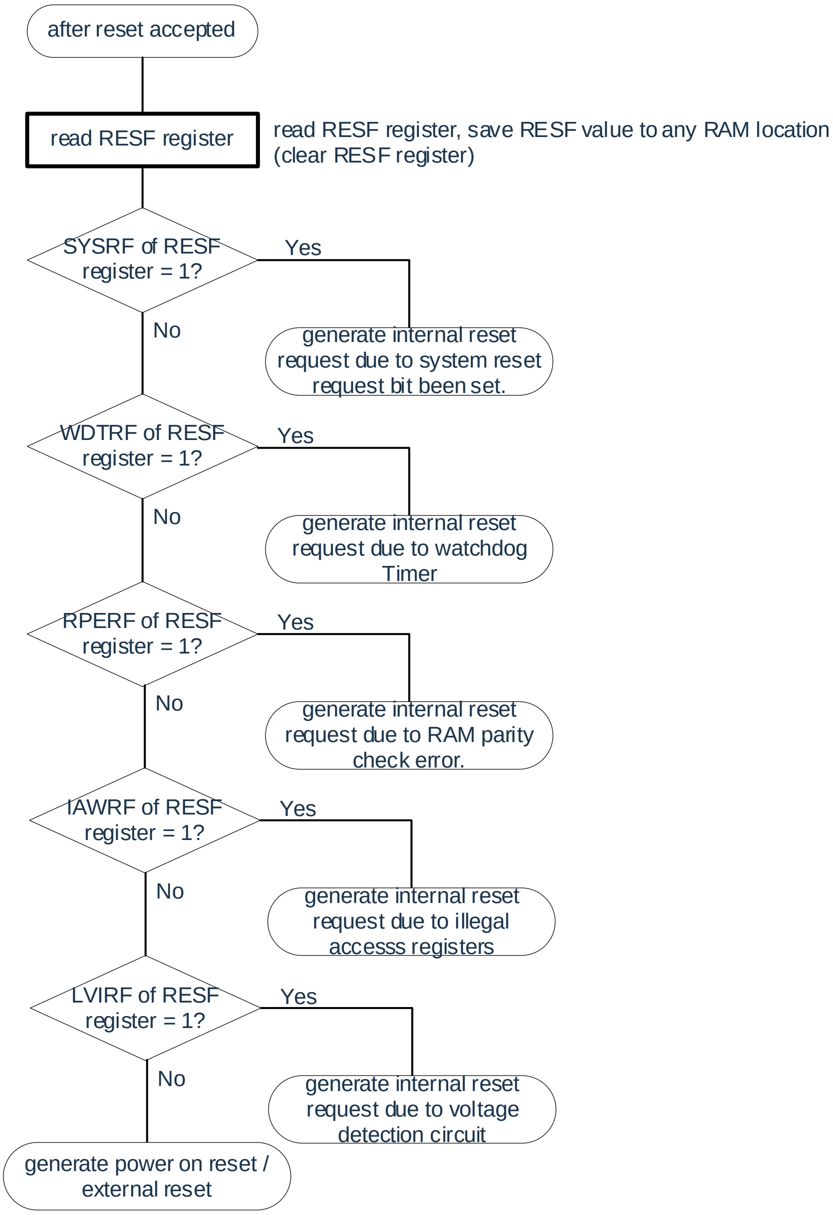
The status of the RESF registers at the time of the reset request is shown in Table 21-2.

Table 21-2 RESF register status when a reset request occurs

Reset the source sign	RESETB input	The reset generated by the POL	The reset caused by the system resetting the requested position bit	The reset generated by WDT	Reset resulting from RAM parity errors	Access the reset generated by the illegal memory	The reset generated by LVD
SYSRF	Clear “0”	Clear “0”	Set “1”	keep	keep	keep	keep
WDTRF			keep	Set “1”			
RPERF				keep	Set “1”		
IAWRF					keep	Set “1”	
LVIRF						keep	Set “1”

The confirmation step of the reset source is shown in Figure 21-5.

Figure 21-5　Confirmation steps for resetting the source


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 process is an example of a confirmation step.

Chapter 22 Power-On Reset Circuit

22.1 Function of power-on reset circuit

The power-on reset circuit (POL) 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_{POL}), 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.

Note that when the power-on reset circuit generates an internal reset signal, clear the reset control flag register (RESF) to “00H”.

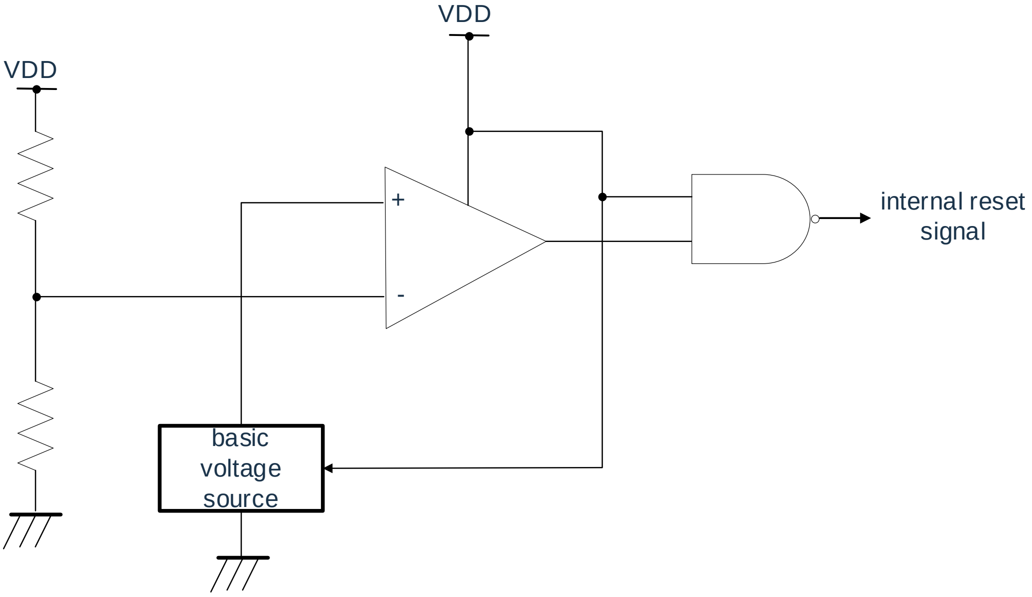
Note 1 The CMS32L051 contains several hardware that generates an internal reset signal. Flags used to indicate the reset source are assigned when an internal reset signal is generated by the access of a watchdog timer (WDT), voltage detection (LVD) circuit, system reset request position bit, RAM parity error, or illegal memory RESF registers; When an internal reset signal is generated by WDT, LVD, assertion of the system reset request bit, RAM parity error, or access to illegal memory, the RESF register is not cleared to “00H” Set the flag to “1”. For more information on RESF registers, refer to Chapter 21 Reset Functions.

2.V_{POR}: POR supply voltage rise detection voltage
V_{PDR}: POR supply voltage drop detection voltage
For details, please refer to the POR circuit characteristics in the data sheet.

22.2 Structure of power-on reset circuit

A block diagram of the power-on reset circuit is shown in Figure 22-1.

Figure 22-1 Block diagram of power-on reset circuit

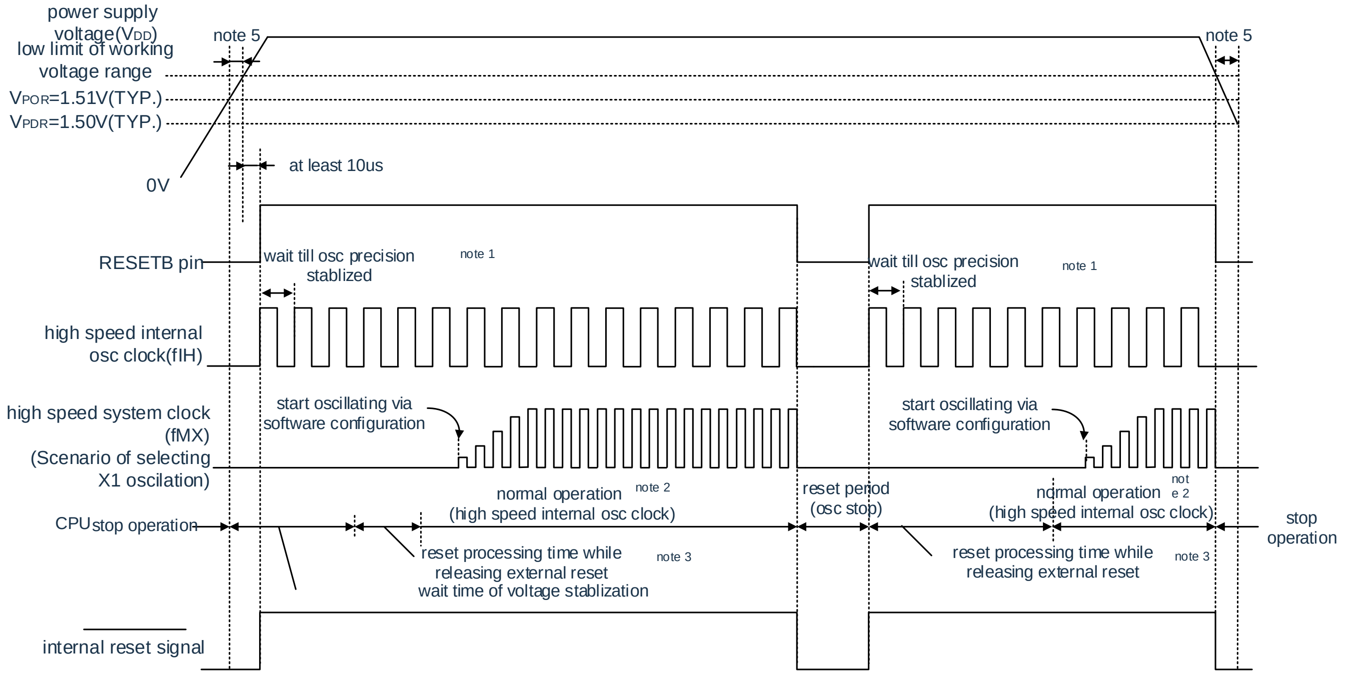


22.3 Operation of power-on reset circuit

The timing of the generation of the internal reset signals of the power-on reset circuit and the voltage detection circuit is shown below.

Figure 22-2 Timing of internal reset signal generation for power-on reset circuit and voltage detection circuit (1/3)

(1) A case of 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
CPU stop operation
normal operation (high speed internal osc clock)
note 2
reset period (osc stop)
normal operation (high speed internal osc clock)
note 2
stop operation
reset processing time while releasing external reset wait time of voltage stablization
note 3
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.

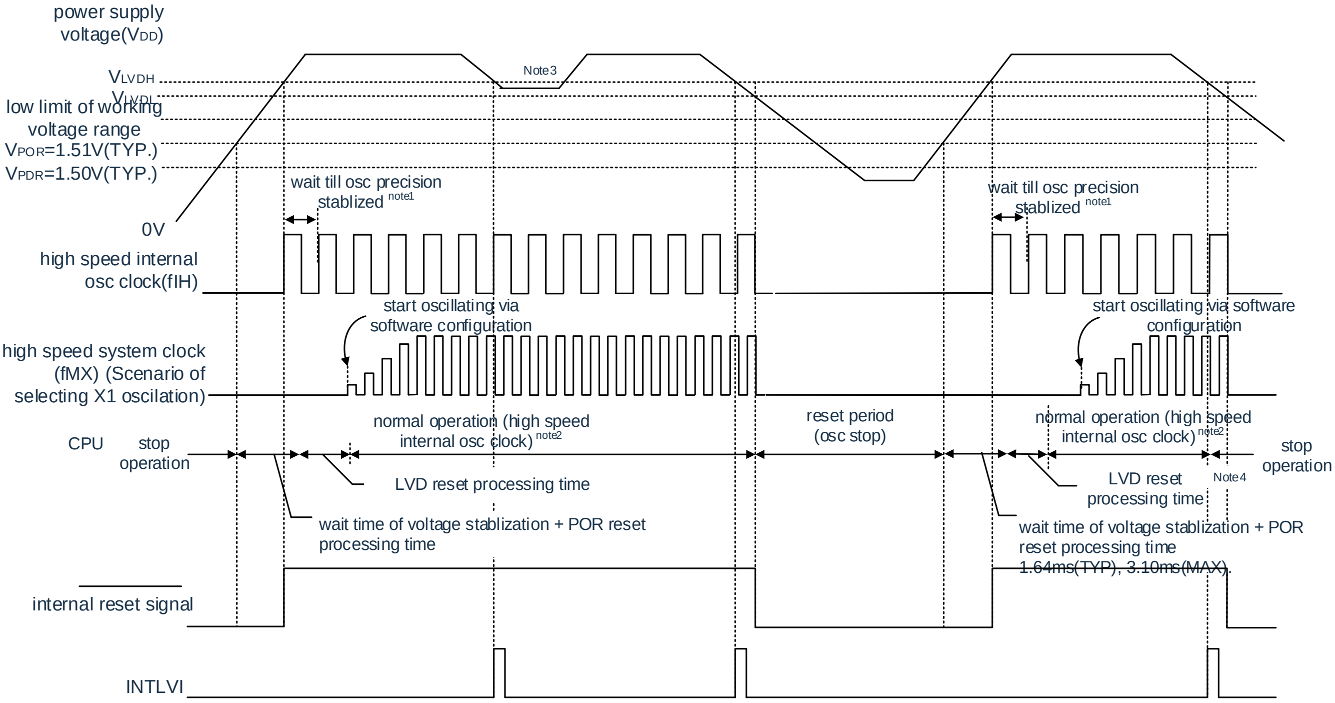
Note V_{POR}: The POR supply voltage rises to detect the voltage

V_{PDR}: The POR supply voltage drops the detection voltage

Notice When the LVD is OFF, an external reset of the RESET B pin must be used. For details, please refer to “Chapter 23 Voltage Detection Circuits“.

Figure 22-2 Timing of internal reset signal generation for power-on reset circuit and voltage detection circuit (2/3)

(2) LVD is in interrupt & reset mode (option bytes 000C1H LVIMDS1, LVIMDS0=1, 0).


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
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.

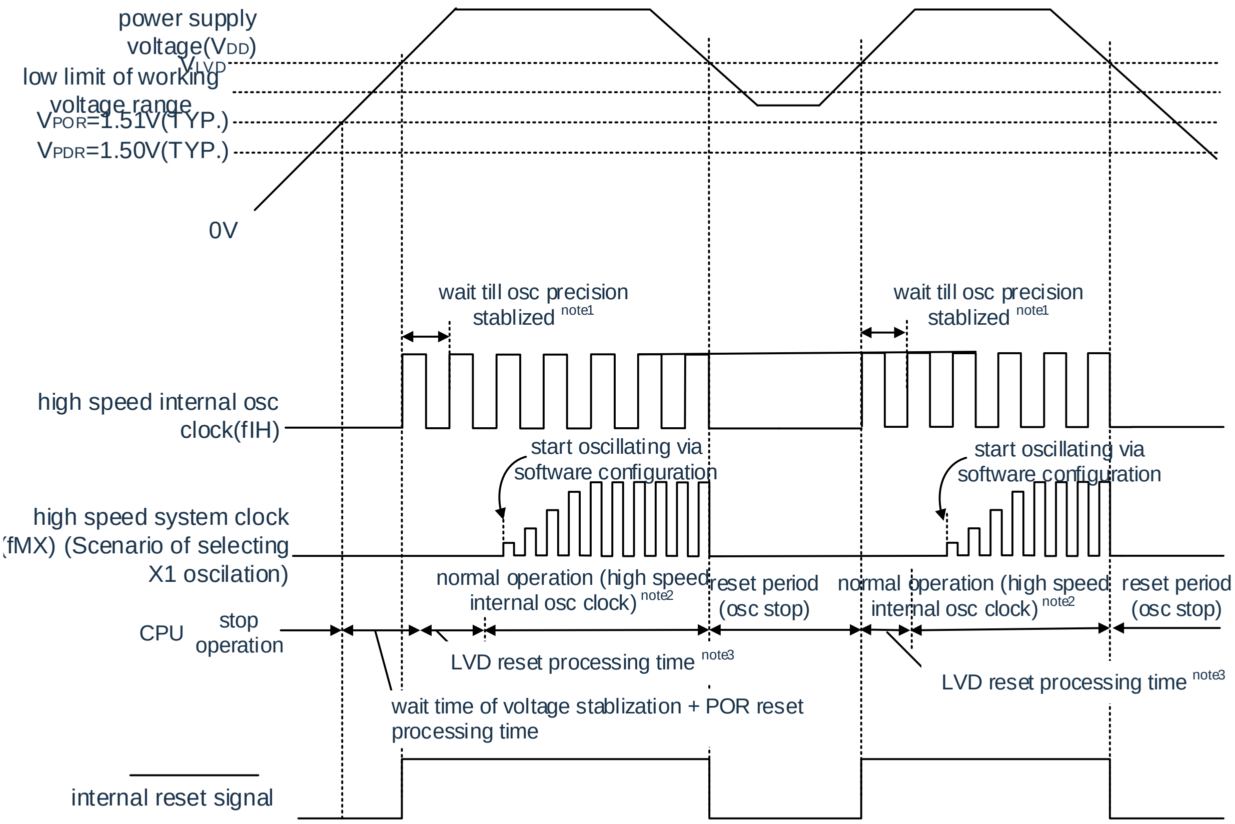
3. After generating the interrupt request signal (INTLVI), the LVILV bit and the LVIMD bit of the voltage detection level register (LVIS) are automatically set to "1". Therefore, it must be considered that the supply voltage may return to the high voltage detection voltage (VLVDH) or higher without falling below the low voltage detection voltage (VLVDL), and after generating INTLVI, follow the steps in “Figure 23-8 Setting procedure for confirmation/reset of operating voltage”and “Figure 23-9: 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 (VLVDH) until the start of normal operation ".

Note		
	V_{LVDH}, V_{LVDL}:	LVD sense voltage
	V_{POR}	: POR supply voltage rise detection voltage
	V_{PDR}	: POR supply voltage drop detection voltage

Figure 22-2 Timing of internal reset signal generation for power-on reset circuit and voltage detection circuit (3/3)

(3) LVD reset mode case (option byte 000C1H LVIMDS1, LVIMDS0=1, 1).


power supply voltage(VDD)
VLVD
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 oscilation)
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 note3
wait time of voltage stablization + POR reset processing time
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 VPOR (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”.

Note 1 V_{LVDH}, V_{LVDL}: LVD sense voltage

V_{POR} : The POR supply rises the sense voltage

V_{PDR} : The POR supply drops the sense voltage

2. When the LVD interrupt mode is selected (option bytes 000C1H LVIMD1, LVIMD0=0, 1), the time from power-on to start the usual operation is the same as the time in Note 3 of Figure “LVD Reset Mode”.

Chapter 23 Voltage Detection Circuit

23.1 Function of voltage detection circuit

The voltage detection circuit sets the operating mode and sense voltage (V_{LVDH}, V_{LVDL}, V_{LVD}) via option bytes (000C1H). Voltage Detection (LVD) circuitry has the following functions.

•Compare the supply voltage (V_{DD}) with the sense voltage (V_{LVDH}, V_{LVDL}, V_{LVD}). to generate an internal reset or internal interrupt signal.

•The sense voltage of the supply voltage (V_{LVDH}, V_{LVDL}) can select 12 detection levels via option bytes (refer to “Chapter 26“ option bytes“).

• Also runs in deep sleep mode

•When the supply voltage rises, the reset state must be maintained by voltage detection circuit or external reset before the supply voltage reaches the operating voltage range shown in the AC characteristics of the data sheet; When the supply voltage drops, it must be reset through deep sleep mode transfer, voltage detection circuitry, or external reset before the supply voltage drops below the operating voltage range. The operating voltage range depends on the setting of the user option bytes (000C2H/010C2H).

(a) Interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0 for option bytes)

Select 2 sense voltages (V_{LVDH}, V_{LVDL}) and high voltage sense levels (V_{LVDH}) via option byte 000C1H) is used to release reset or interrupt, and low voltage sense level (V_{LVDL}) is used to generate a reset.

(b) Reset mode (LVIMDS1, LVIMDS0=1, 1 for option bytes).

Use option byte 000C1H to select 1 sense voltage (V_{LVD}) to generate or dereset.

(c) Interrupt mode (LVIMDS1, LVIMDS0=0, 1 for option bytes).

Use option byte 000C1H to select one sense voltage (V_{LVD}) to generate an interrupt or to release reset. In each mode, the following interrupt signals and internal reset signals are generated.

Interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0)	Reset mode (LVIMDS1, LVIMDS0=1, 1)	Interrupt mode (LVIMDS1, LVIMDS0=0, 1)
When the operating voltage drops, when detectedVDD＜ VLVDH, an interrupt request signal is generated; When detected When VDD＜VLVDL, an internal reset is generated; When V DD≥VLVDH is detected, the internal reset is released.	When V DD≥VLVD is detected, the internal reset is released; When detectedVDD＜VLVD, an internal reset is generated.	After a reset occurs, the internal reset state of the LVD is continued Continued until V DD≥V LVD. When detected When V DD≥V LVD, the internal reset of the LVD is released. After the internal reset of the LVD is released, if detected VDD＜VLVDorVDD≥VLVD when, just Generates an interrupt request signal (INTLVI).

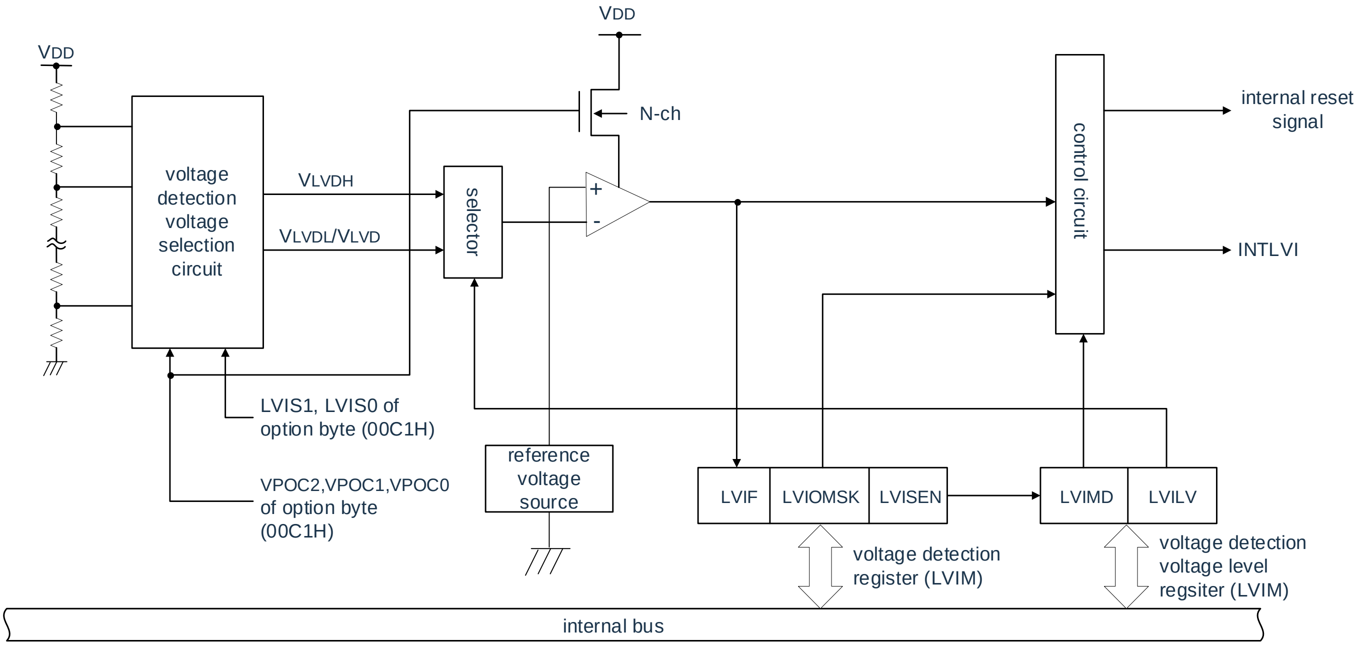
When the voltage detection circuit is running, it is possible to confirm whether the supply voltage is greater than or equal to the sense voltage or less than the sense voltage by reading the voltage detection flag (LVIF: bit0 of the voltage detection register (LVIM)).

If a reset occurs, bit0 (LVIRF) of the reset control flag register (RESF) is set to “1”. For more information on RESF registers, refer to Chapter 21 Reset Functions.

23.2 Structure of voltage detection circuit

A block diagram of the voltage detection circuit is shown in Figure 23-1.

Figure 23-1 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 (LVIM)
internal bus

23.3 Registers for controlling voltage detection circuit

The voltage detection circuit is controlled by the following registers.

- Voltage Sense Register (LVIM).
- Voltage Sense Level Register (LVIS).

23.3.1 Voltage sense register (LVIM).

This register setting enables or disables overriding the voltage sense register (LVIS) and confirms the shielding status of the LVD output. The LVIM registers are set via 8-bit memory operation instructions.

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

Figure 23-2 Format of voltage sense register (LVIM)

Address: 40020441H After reset: 00HNote 1 R/W Note 2

symbol	7	6	5	4	3	2	1	0
LVIM	LVISEN Note3	0	0	0	0	0	LWeOTBSP	LVIF

LVISEN Note 3	Enable/disable override setting for voltage sense level registers (LVIS).
0	It is forbidden to overwrite the LVIS register (LVIOMSK=0 (LVD output mask is invalid)).
1	Rewriting of LVIS registers (LVIOMSK=1 (LVD output masking active)) is Enableed.

LVIOMSK	Mask status flag of the LVD output
0	The LVD output mask is invalid.
1	LVD output shielding valid Note 4.

LVIF	Voltage detection flag
0	The supply voltage ($V_{DD)}$ ≥ the sense voltage (V_{LVD}) or LVD is OFF.
1	Supply voltage (V_{DD}< sense voltage (V_{LVD})

Note 1. The reset value varies depending on the reset source.

When the LVD is reset, the value of the LVIM register is not reset and the original value is maintained; On other resets, clear LVISEN to “0”.

2. Bit0 and bit1 are read-only bits.

3. it can only be set when interrupt & reset mode is selected (the LVIMDS1 bit and LVIMDS0 bits 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 (the LVIMDS1 bit and LVIMDS0 bits of the option bytes are “1” and “0”, respectively). The LVIOMSK bit automatically changes to “1” during the following periods, masking the reset or interrupt caused by LVD.

- LVISEN=1 period
- Wait time from the time the LVD interrupt occurs until the LVD sense voltage stabilizes
- Wait time from changing the value of the LVILV bit until the LVD sense voltage stabilizes

23.3.2 Voltage sense level register (LVIS)

This is the register that sets the voltage sense level.

The LVIS registers are set via 8-bit memory operation instructions. After the reset signal is generated, the value of this register becomes “00H/01H/81H” Note 1.

Figure 23-3 Format of voltage sense level register (LVIS)

Address: 40020442H After reset: 00H/01H/81HNote 1 R/W

symbol	7	6	5	4	3	2	1	0
LVIS	LWeMD Note2	0	0	0	0	0	0	LVILV Note2

LVIMD Note 2	Operating mode of voltage detection
0	Interrupt mode
1	Reset mode

LVILV Note 2	LVD detection level
0	High voltage sense level (V_{LVDH}).
1	Low voltage detection level (V_{LVDL} or V_{LVD}).

Note 1. The reset value varies depending on the reset source and option byte settings. When an LVD reset occurs, this register is not cleared to “00H”.

In the event of a reset other than LVD, the value of this register is as follows:

- • Option bytes LVIMDS1, LVIMDS0=1, 0:00H
- • Option bytes LVIMDS1, LVIMDS0=1, 1 when: 81H
- • Option bytes LVIMDS1, LVIMDS0=0, 1:01H

2. You can write “0” only when interrupt & reset mode is selected (the LVIMDS1 bit and LVIMDS0 bits of the option byte are “1” and “0”, respectively) . Cannot be set in other cases. In interrupt & reset mode, value replacement is performed automatically by generating a reset or interrupt.

Note 1. To override the LVIS registers, the steps of Figure 23-7 and Figure 23-8 must be followed.

2. Select the operating mode of LVD and the detection voltage of each mode (VLVDH, VLVDL, V) by option byte 000C1H LVD). The format of the user option bytes (000C1H/010C1H) is shown in Table 23-1. For more information about option bytes, refer to Chapter 26, Option Bytes.

Table 23-1 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 the option byte						
IN$_{LVDH}$		IN$_{LVDL}$	VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rise	decline	decline						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		
—			Values other than those above are prohibited.						

- LVD setting (reset mode)

Detection voltage		Setting value of the option byte						
V$_{LVD}$		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rise	decline						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		
—		Values other than those above are prohibited.						

Note 1 The detection voltage is TYP Value. For details, please refer to the LVD circuit characteristics in the data sheet.

Table 23-1 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 the option byte						
V_{LVD}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rise	decline						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		
—		Values other than those above are prohibited.						

- LVD is OFF (Use RESETB External reset of the pin)

Detection voltage		The setting value of the option byte						
V_{LVD}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rise	decline						LVIMDS1	LVIMDS0
—	—	1	×	×	×	×	×	1
—		Values other than those above are prohibited.						

Note 1 You must write "1" to bit4.

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 working voltage range shown in the AC characteristics of the data sheet; When the supply voltage drops, it must be reset through deep sleep mode transfer, voltage detection circuitry, or external reset before the supply voltage drops below the operating voltage range.
The operating voltage range depends on the setting of the user option byte (000C2H).

Note 1 ×: Ignore

2. The detection voltage is TYP Value. For details, please refer to the LVD circuit characteristics in the data sheet.

23.4 Operation of voltage detection circuit

23.4.1 Settings when used in reset mode

The operating mode (reset mode (LVIMDS1, LVIMDS0=1, 1)) and the sense voltage (V) are set by option byte 000C1H LVD). If reset mode is set, operation begins in the following initial state.

• Set bit7 (LVISEN) of the voltage sense register (LVIM) to “0” (disable overriding of the voltage sense register (LVIS)).

• Set the initial value of the voltage sense level register (LVIS) to “81H”. bit7 (LVIMD) is “1” (reset mode). bit0 (LVILV) is “1” (voltage detection level: VLVD).

● Operation of LVD reset mode

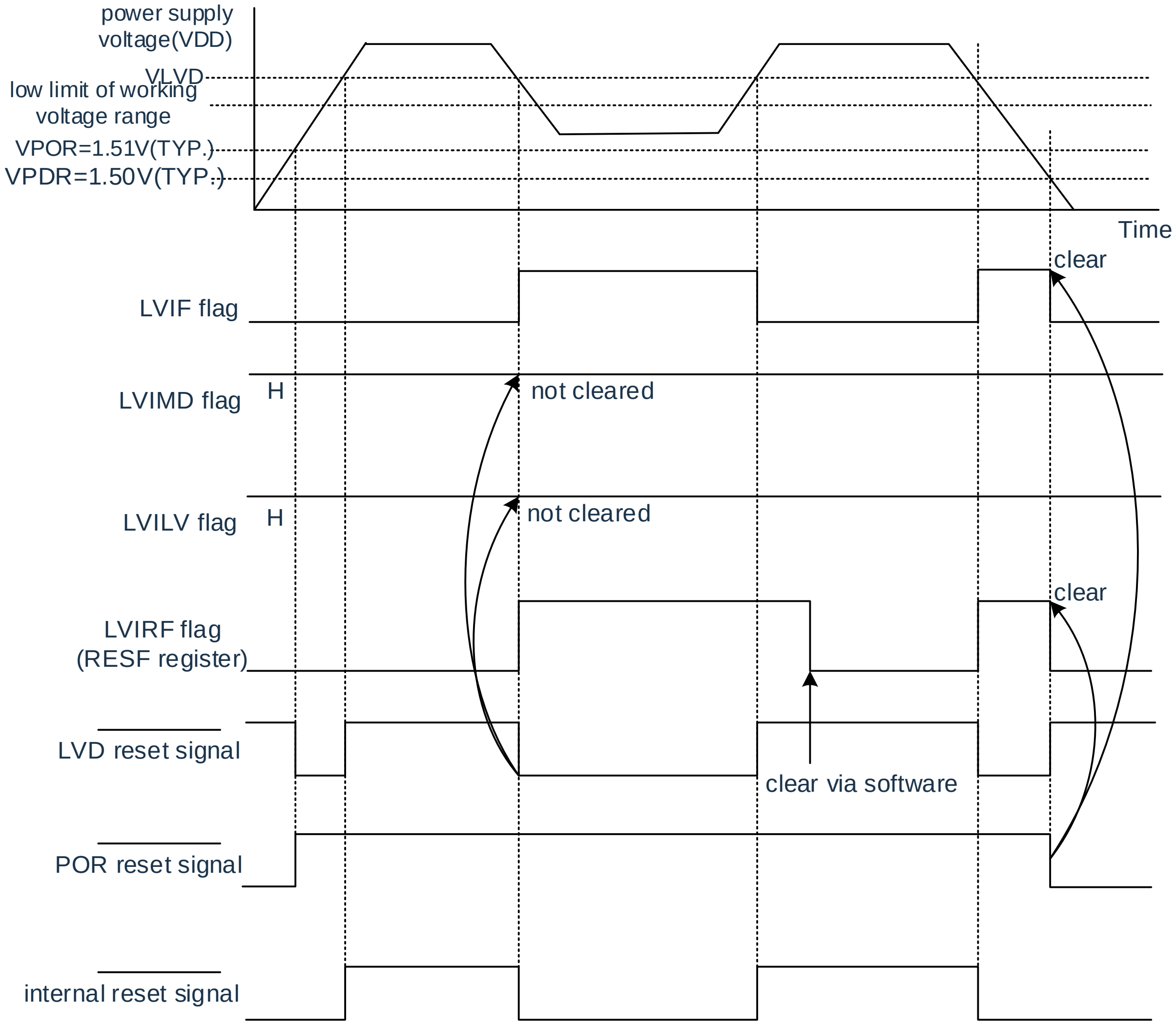
When the power is turned on, the reset mode (LVIMDS1, LVIMDS0=1, 1 for option bytes) exceeds the voltage detection level (V_{LVD}) at the supply voltage (V_{DD})) before maintaining the internal reset state of the LVD. If the supply voltage (V_{DD}) exceeds the voltage sense level (VLVD), the internal reset is released.

When the operating voltage drops, an internal reset of the LVD is generated if the supply voltage (V_{DD}) falls below the voltage sense level (V_{LVD}).

The timing of the generation of the internal reset signal for the LVD reset mode is shown in

Figure 23-4.

Figure 23-4 Generation timing of the internal reset signal (LVIMDS1, LVIMDS0=1, 1 for option bytes)



Note V_{POR}: The POR supply voltage rise detection voltage

V_{PDR}: The POR supply voltage drop detection voltage

23.4.2 Settings when used in interrupt mode

The operating mode (interrupt mode (LVIMDS1, LVIMDS0=0, 1)) and the sense voltage (V) are set by option byte 000C1H LVD). If you set the interrupt mode, it starts operating in the following initial state.

• Set bit7 (LVISEN) of the voltage sense register (LVIM) to "0" (disable overriding of the voltage sense register (LVIS)).

•Set the initial value of the voltage sense level register (LVIS) to "01H". Bit7 (LVIMD) is "0" (interrupt mode). bit0 (LVILV) is "1" (voltage detection level: V_{LVD}).

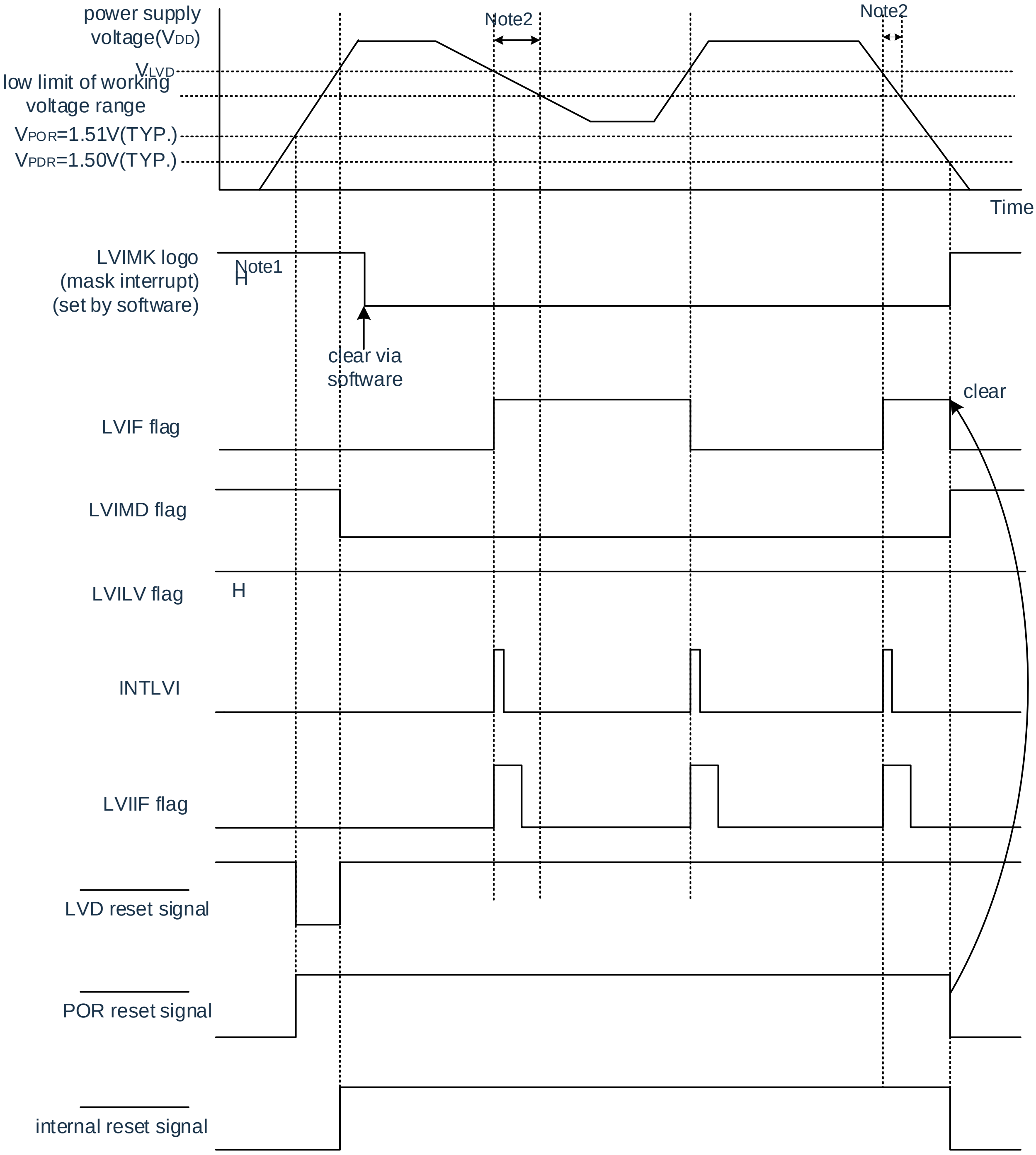
● LVD interrupt mode operation

After the reset is generated, the interrupt mode (LVIMDS1 for option bytes, LVIMDS0=0, 1) exceeds the voltage detection level (V_{LVD}) at the supply voltage (VDD)) before maintaining the internal reset state of the LVD. If the supply voltage (V_{DD}) exceeds the voltage sense level (V_{LVD}), the internal reset of the LVD is released.

After the internal reset of the LVD is released, if the supply voltage (V_{DD}) exceeds the voltage detection level (V_{LVD}), an interrupt request signal for the LVD is generated (INTLVI). When the operating voltage drops, it must be reset by transferring in deep sleep mode or externally resetting before the operating voltage drops below the operating voltage range shown in the AC characteristics of the data sheet. During restart operation, it is important to confirm that the supply voltage has returned to the operating range.

The timing of the generation of interrupt request signals for LVD interrupt mode is shown in Figure 23-5.

Figure 23-5 Generation timing of interrupt signals (option bytes LVIMDS1, LVIMDS0=0, 1)



Note 1 After the reset signal is generated, the LVIMK flag changes to "1".

2. When the operating voltage drops, it must be set to the reset state by transferring or externally resetting the operating voltage below the operating voltage range shown in the AC characteristics of the data sheet. During restart operation, it must be confirmed that the supply voltage returns to the operating voltage range.

Note V_{POR}: The POR supply voltage rise detection voltage

V_{PDR}: The POR supply voltage drop detection voltage

23.4.3 Settings for interrupt & reset mode

The operating mode (interrupt & reset mode (LVIMDS1, LVIMDS0=1, 0)) and the sense voltage (V_{LVDH}, VLVDL) are set by option byte 000C1H).

If the interrupt & reset mode is set, it starts operating in the following initialization state.

• Set bit7 (LVISEN) of the voltage sense register (LVIM) to “0” (disable overriding of the voltage sense register (LVIS)).

•Set the initial value of the voltage sense level register (LVIS) to “00H”. bit7 (LVIMD) is “0” (interrupt mode).

Bit0 (LVILV) is “0” (high voltage sense level: V_{LVDH}).

● Operation of LVD interrupt & reset mode

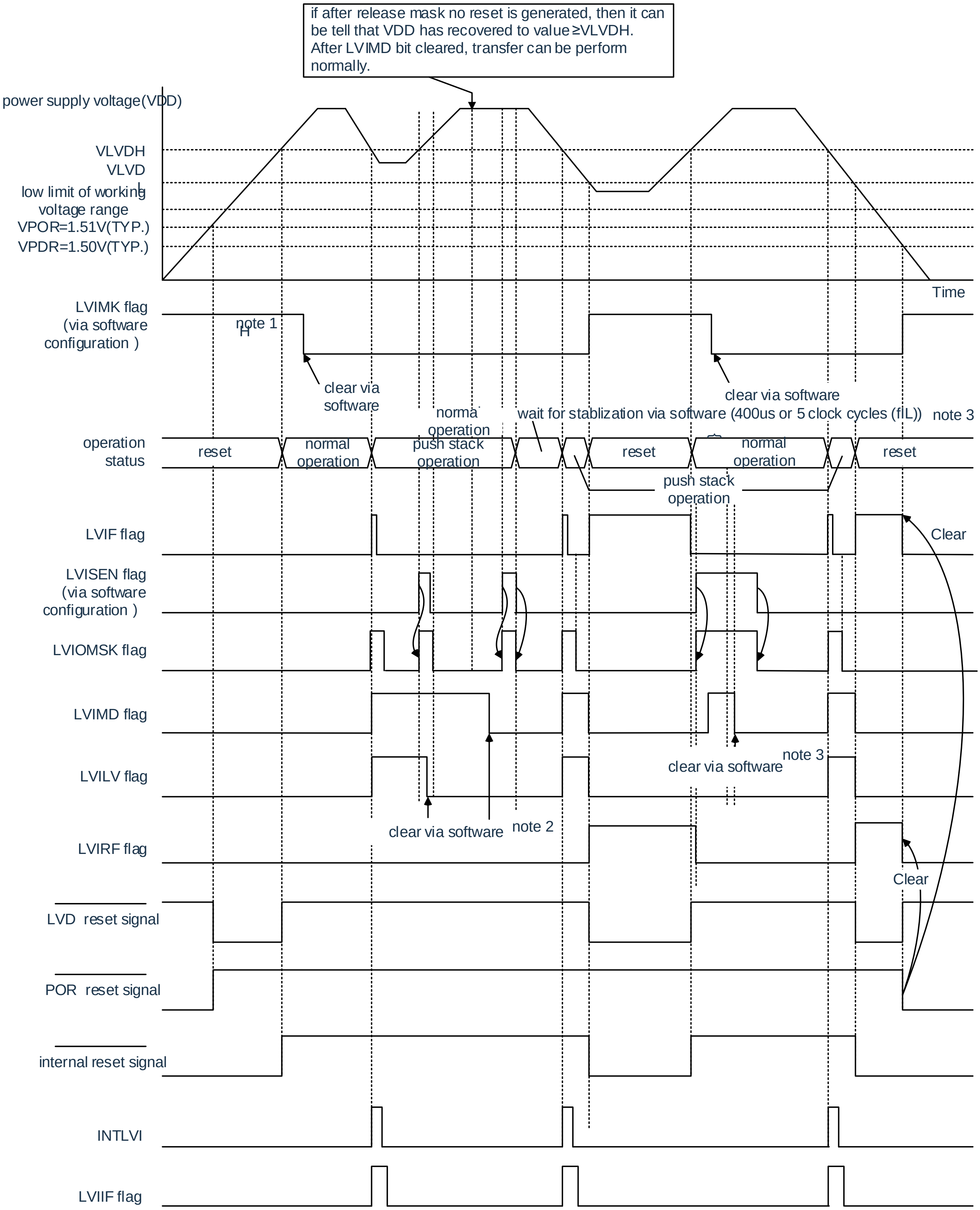
When power is turned on, the interrupt & reset mode (LVIMDS1, LVIMDS0 = 1, 0 of the option byte) maintains the internal reset state of LVD until the power supply voltage (V_{DD}) exceeds the high voltage detect level (V_{LVDH}). If the supply voltage (V_{DD}) exceeds the high voltage detection level (V_{LVDH}), the internal reset is released.

When the operating voltage drops, if the supply voltage (V_{DD}) is below the high voltage detection level (V_{LVDH}), an interrupt request signal (INTLVI) is generated for the LVD and any stack processing can be performed. After that, if the supply voltage (V_{DD}) is lower than the low voltage detection level (V_{LVDL}), an internal reset of LVD is generated. However, after INTLVI occurs, 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 without falling below the low voltage detection voltage (V_{LVDL}).

When using the LVD interrupt & reset mode, you must follow the steps in the flowchart shown in "Figure 23-7 Setup steps for confirmation/reset of the operating voltage " and "Figure 23-8 Initial Setting Procedure for Interrupt & Reset Mode".

The internal reset signal and the generation timing of the interrupt signal in the LVD interrupt & reset mode are Figure 23-6.

Figure 23-6 Generation timing of reset & interrupt signal (LVIMDS1 for option bytes, LVIMDS0=1, 0) (1/2)



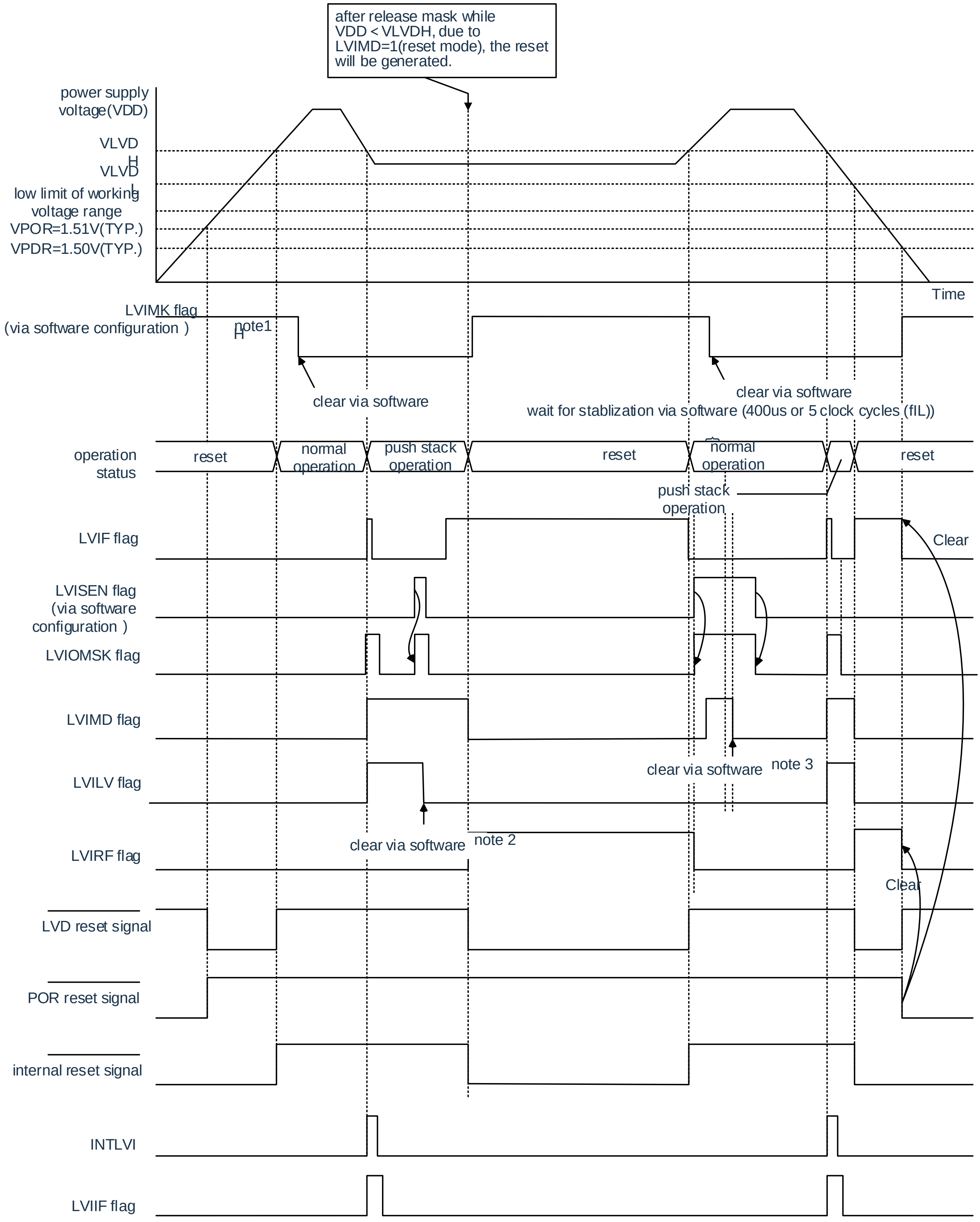
Note 1. After the reset signal is generated, the LVIMK flag changes to “1”.

2. When using the interrupt & reset mode, it must be set after the interrupt occurs in accordance with the “Confirmation Figure 23-7 Setup steps for confirmation/reset of the operating voltage“.

3. When using interrupt & reset mode, it must be set in accordance with the Initial Setup Step of “Figure 23-8 Initial setup steps for interrupt & reset mode” is released.

Note VPOR: POR supply voltage rise detection voltage
VPDR: POR supply voltage drop detection voltage

Figure 23-6 Generation timing of interrupt & reset signal (LVIMDS1 for option bytes, LVIMDS0=1, 0) (2/2)



Note 1. After the reset signal is generated, the LVIMK flag changes to “1”.

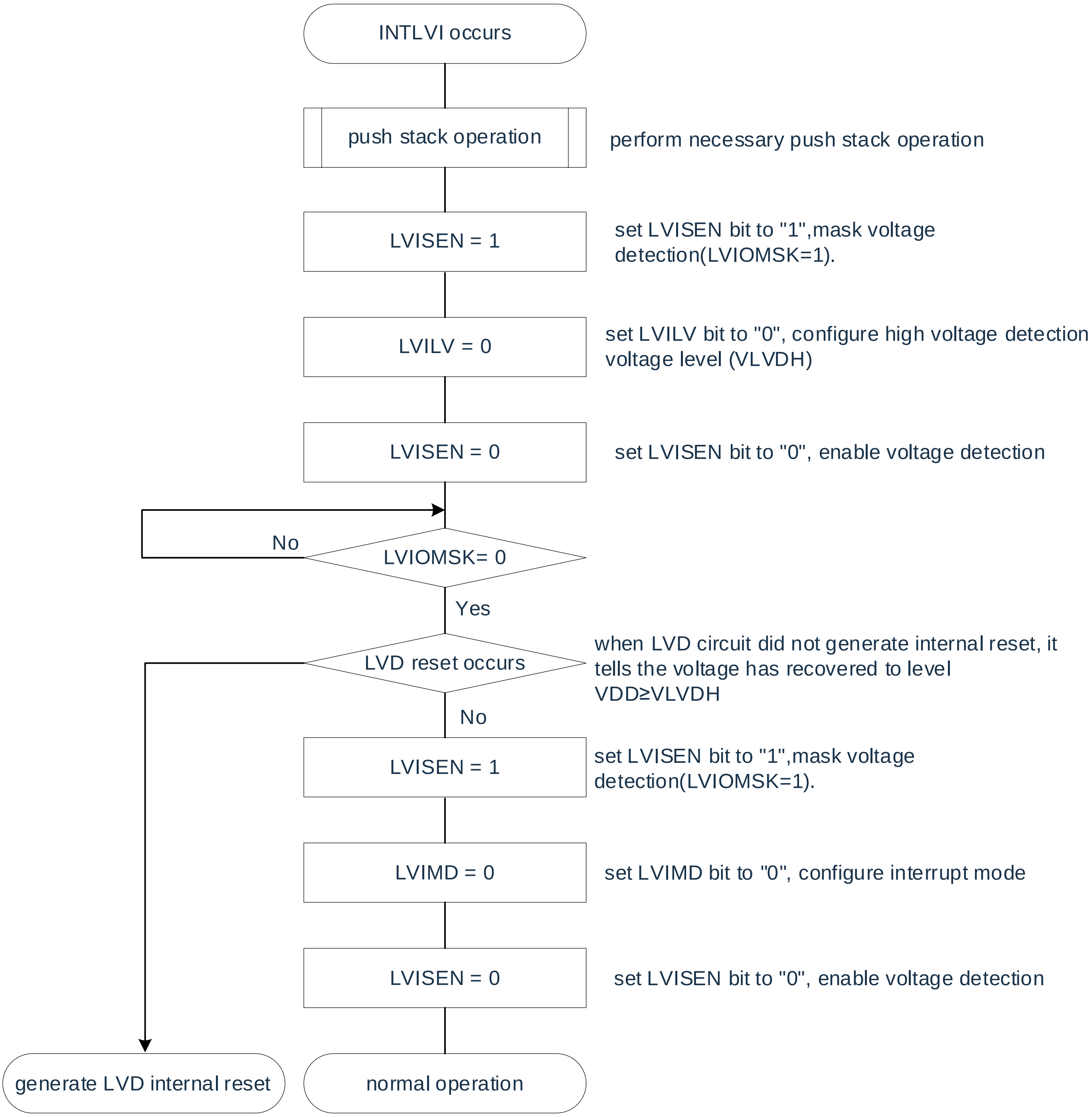
2. When using the interrupt & reset mode, it must be set after the interrupt occurs in accordance with the “Confirmation Figure 23-7 Setup steps for confirmation/reset of the operating voltage“.

3. When using interrupt & reset mode, it must be set in accordance with the “Figure 23-8 Initial setup steps for interrupt & reset mode” is released.

Note V_{POR}: POR supply voltage rise detection voltage

V_{PDR}: POR supply voltage drop detection voltage

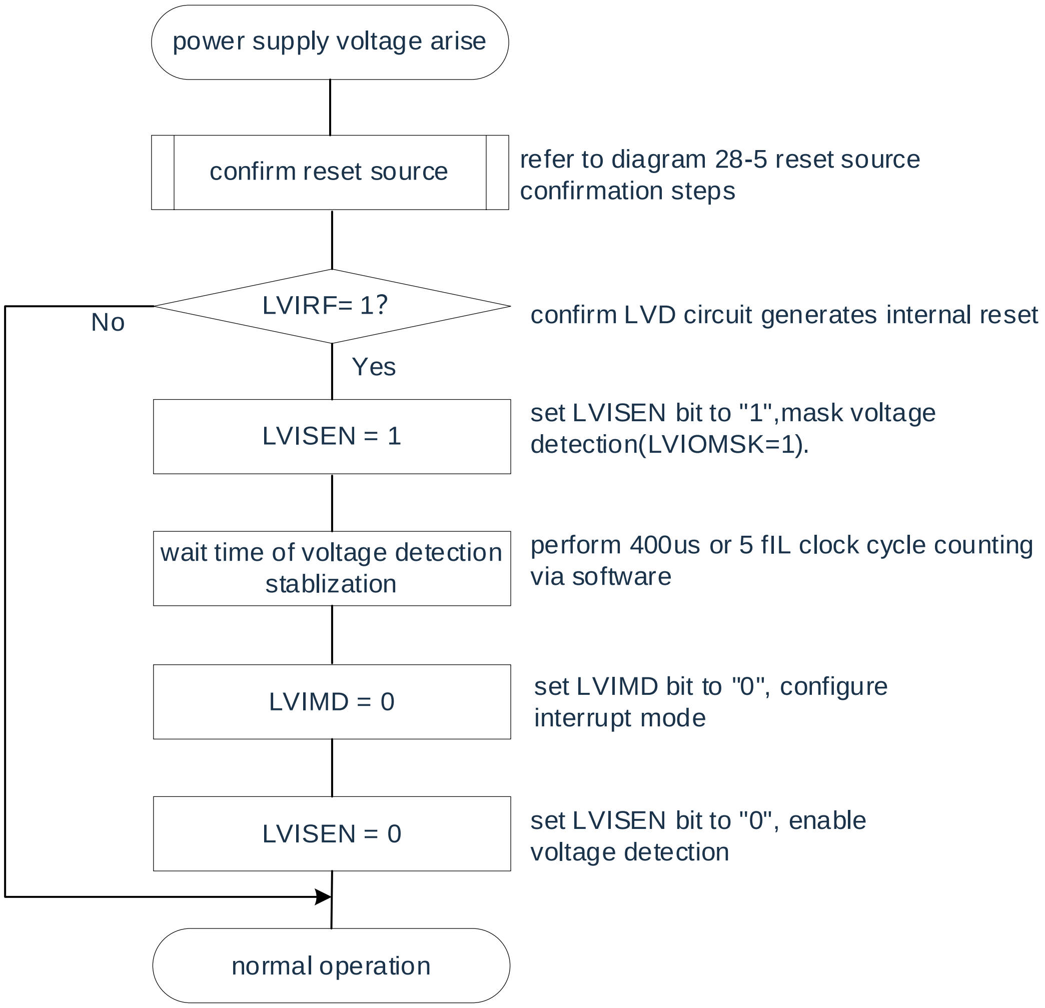
Figure 23-7 Setup steps for confirmation/reset of the operating voltage


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), it is required after the LVD reset (LVIRF=1) is released. Voltage detection settling wait time of 400us or 5 f-IL clocks. The LVIMD bit clear “0” must be initialized after waiting for the voltage detection to stabilize. During the counting of the voltage detection stabilization wait time and when overwriting the LVIMD bit, the LVISEN position “1” must be used to mask the reset or interrupt generation caused by the LVD.

The initial setup steps for the interrupt & reset mode are shown in Figure 23-8.

Figure 23-8 Initial setup steps 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

Note $f_{IL:}$ Low speed internal oscillator clock frequency

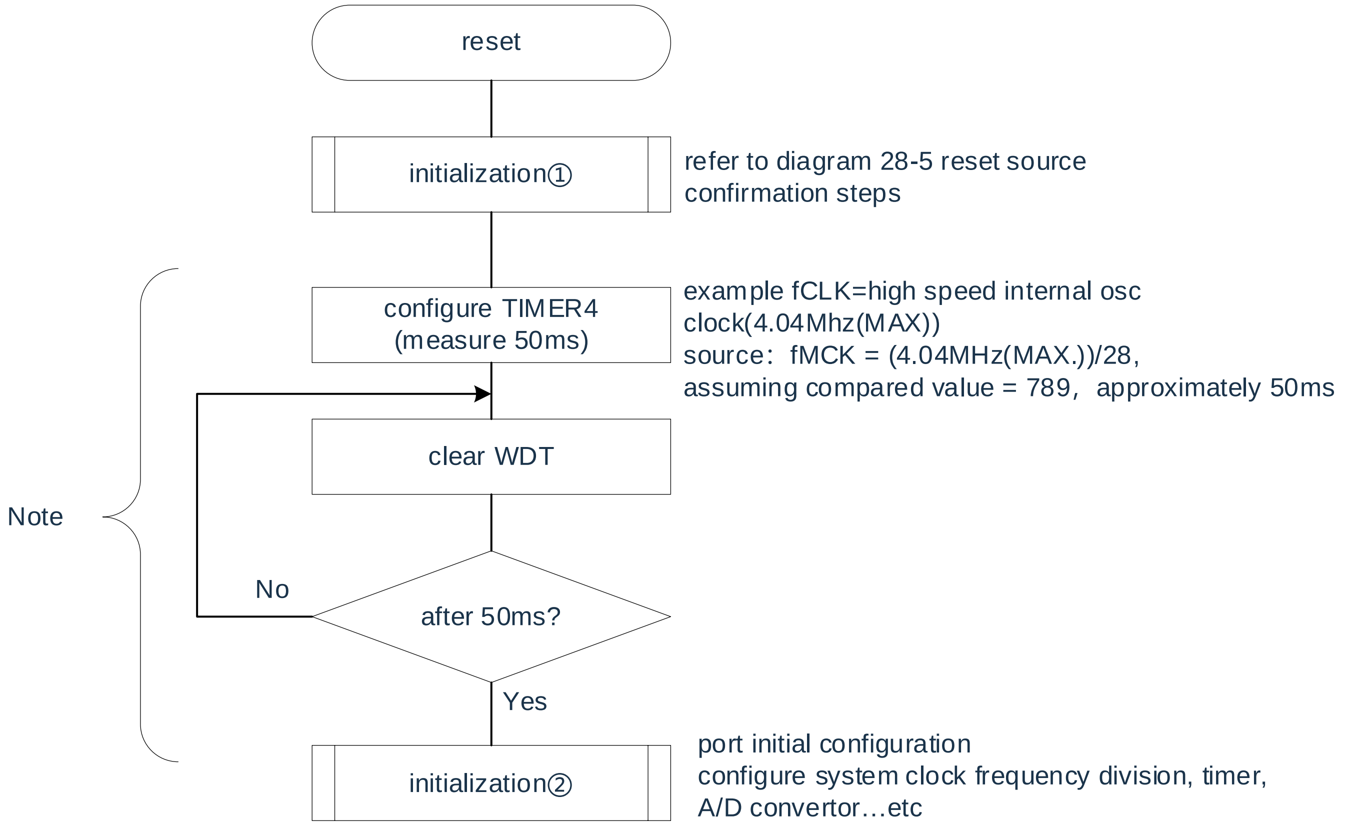
23.5 Considerations for voltage detection circuits

(1) Regarding voltage fluctuations when the power is turned on

For systems where the supply voltage (V_{DD}) fluctuates for a certain amount of time near the LVD sense voltage, it is possible to repeatedly enter the reset state and the reset release state. The following processing can be used to set the time of release reset to the start of the microcontroller operation arbitrarily.

< processing > after the reset is released, the initial setting of the port, etc. must be made by using the software counter of the timer and waiting for different supply voltage fluctuation times for each system.

Figure 23-9 Example of software processing when the supply voltage fluctuation near the LVD detection voltage does not exceed 50ms


reset
initialization①
refer to diagram 28-5 reset source confirmation steps
configure TIMER4 (measure 50ms)
example fCLK=high speed internal osc clock(4.04Mhz(MAX))
source: fMCK = (4.04MHz(MAX.))/28,
assuming compared value = 789, approximately 50ms
clear WDT
Note
No
after 50ms?
Yes
initialization②
port initial configuration
configure system clock frequency division, timer, A/D convertor…etc

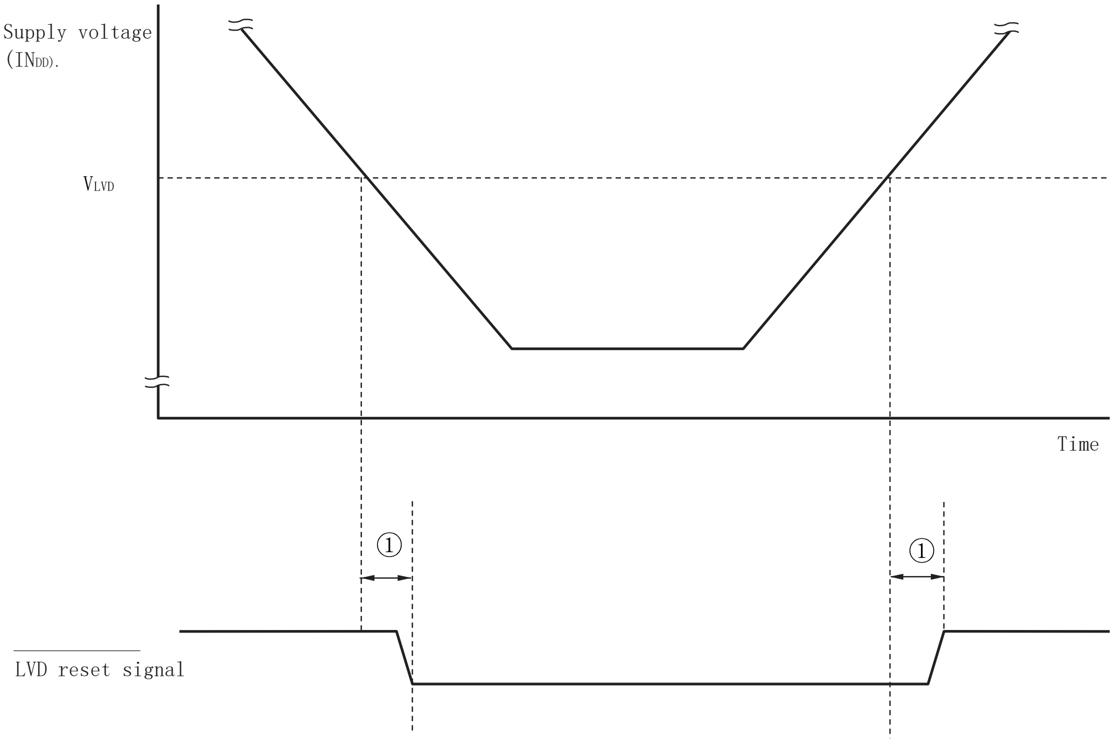
Note If a reset occurs again during this period, it is not transferred to initialization processing (2).

(2) The delay from the generation of the LVD reset source to the generation or release of the LVD reset

From meeting the supply voltage (V_{DD}) < LVDDetection voltage (V_{LVD}) to generate LVDA delay occurs until the reset. Again, from LVD Detection voltage (V_{LVD}≤ supply voltage (V_{DD}) to Dismiss LVDA delay can also occur until the reset (reference

Figure 23-10).

Figure 23-10 Delay from generation of LVD reset source to generation or release of LVD reset



① : Detection delay (300us (MAX.)).

(3) For situations where LVD is plugged in when LVD is set to OFF

When setting the LVD to OFF, an external reset of the RESET B pin must be used.

During an external reset, a low level of at least 10us must be entered into the RESETB pin. If an external reset is performed when the supply voltage rises, the supply must be turned on after entering low on the RESETB pin and at least 10us low over the operating voltage range shown in the AC characteristics of the data sheet, and then enter high.

(4) The operating voltage drops when LVD is set to OFF and set to LVD interrupt mode

With LVD set to OFF and set to LVD interrupt mode, if the operating voltage drops, it must be reset by transfer from deep sleep mode or external reset before the operating voltage drops below the operating voltage range shown in the AC characteristics of the data sheet. During restart operation, it must be confirmed that the supply voltage returns to the operating voltage range.

Chapter 24 Security Features

24.1 Overview

In response to IEC60730 and EC61508 safety standards, the CMS32L051 has the following built-in safety features.

The purpose of this safety function is to safely stop working when a fault is detected through self-diagnosis of the microcontroller.

(1) Flash CRC computing function (high-speed CRC, general-purpose CRC).

CRC operation detects data errors in flash memory. The following two CRCs can be used according to different uses and conditions of use.

- “High Speed CRC”... In the initializer, it stops the CPU and checks the entire code flash area at high speed.
- “Generic CRC”.... In CPU operation, it is not limited to the code flash memory area but can be used for multi-purpose inspection.

(2) RAM parity error detection function

When reading RAM data, parity errors are detected.

(3) SFR protection function

Prevent overwriting SFR due to CPU runaway.

(4) Frequency detection function

Self-test CPU/peripheral hardware clock frequency can be performed using a general-purpose timer unit.

(5) A/D test function

It can perform A/D converter self-test by A/D conversion of positive (+) reference voltage, negative (-) reference voltage, analog input channel (ANI), temperature sensor output and internal reference voltage output of the A/D converter.

(6) Digital output signal level detection function for input/output ports

When the input/output port is in output mode, the output level of the pin can be read.

24.2 Registers used by security functions

Each function of the safety function uses the following registers.

Register name	Functions of the security function
• Flash CRC Control Register (CRC0CTL). • Flash CRC Result Register (PGCRCL).	Flash CRC operation function (High Speed CRC).
• CRC Input Register (CRCIN). • CRC Data Register (CRCD).	CRC operation function (General CRC).
• RAM Parity Error Control Register (RPECTL).	RAM parity error detection function
• Special SFR Protection Control Register (SFRGD).	SFR protection function
• Timer input selects register 0 (TIS0).	Frequency detection function
• A/D Test Registers (ADTES).	A/D test function
• Port Mode Select Register (PMS).	Digital output signal level detection function at the input/output pin

The contents of each register are described in “24.3 Operation of security functions.

24.3 Operation of security functions

24.3.1 Flash CRC operation function (high-speed CRC)

The IEC60730 standard requires the confirmation of data in flash memory and recommends CRC as a means of confirmation. This high-speed CRC inspects the entire code flash memory area during the initialization (initialization) program.

The high-speed CRC stops the operation of the CPU and reads 32-bit data from the flash memory through 1 clock for operation. Therefore, it is characterized by a shorter time to complete the check (e.g., 64KB of flash: 512us@32MHz).

CRC generates a polynomial corresponding to CRC-16-CCITT's “X^{16} +X^{12} +X^{5} +1”.

Operate on the MSB of bit31→bit0 first.

Note Because the generic CRC is LSB first, the results are different.

Flash CRC control register (CRC0CTL)

This is a register that sets the operating control and operation range of a high-speed CRC operator. The CRC0CTL register is set via an 8-bit memory operation command. After the reset signal is generated, the value of this register becomes “00H”.

Figure 24-1 Format of flash CRC control register (CRC0CTL)

Address: 40021810H After reset: 00HR/W

Symbol	7	6	5	4	3	2	1	0
CRC0CTL	CRC0EN	CRCCHK60	0	0	0	FEA2	FEA1	FEA0

CRC0EN	Operation control of high-speed CRC operators
0	Stop running.
1	Start the operation by executing the WFE instruction.

CRCCHK60	FEA2	FEA1	FEA0	Calculation range of high-speed CRC
0	0	0	0	00000H~1FFBH(8K-4byte)
0	0	0	1	00000H ~ 3FFBH(16K-4byte)
0	0	1	0	00000H ~ 5FFBH(24K-4byte)
0	0	1	1	00000H ~ 7FFBH(32K-4byte)
0	1	0	0	00000H ~ 9FFBH(40K-4byte)
0	1	0	1	00000H ~ BFFBH(48K-4byte)
0	1	1	0	00000H ~ DFFBH(56K-4byte)
0	1	1	1	00000H ~ FFFBH(64K-4byte)
1	0	0	0	00000H ~ EFFFBH(60K-4byte)

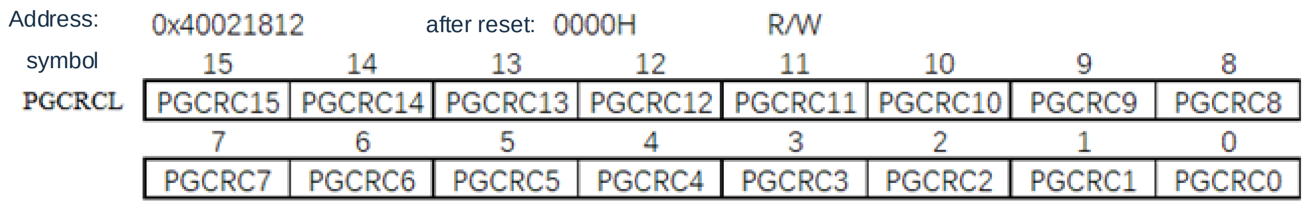
Note: bit3~6 must be set to 0.

Remark The expected value of the CRC operation used for comparison must be stored in the last 4 bytes of flash memory in advance, so the operation range is subtracted from the range of 4 bytes.

24.3.1.1 Flash CRC result register (PGCRCL).

This is the register that holds the results of high-speed CRC operations.
The PGCRCL register is set via a 16-bit memory operation command.
After the reset signal is generated, the value of this register changes to “0000H”.

Figure 24-2 Format of flash CRC result register (PGCRCL)



Address: 0x40021812 after reset: 0000H R/W

symbol	15	14	13	12	11	10	9	8
PGCRCL	PGCRC15	PGCRC14	PGCRC13	PGCRC12	PGCRC11	PGCRC10	PGCRC9	PGCRC8
	7	6	5	4	3	2	1	0
	PGCRC7	PGCRC6	PGCRC5	PGCRC4	PGCRC3	PGCRC2	PGCRC1	PGCRC0

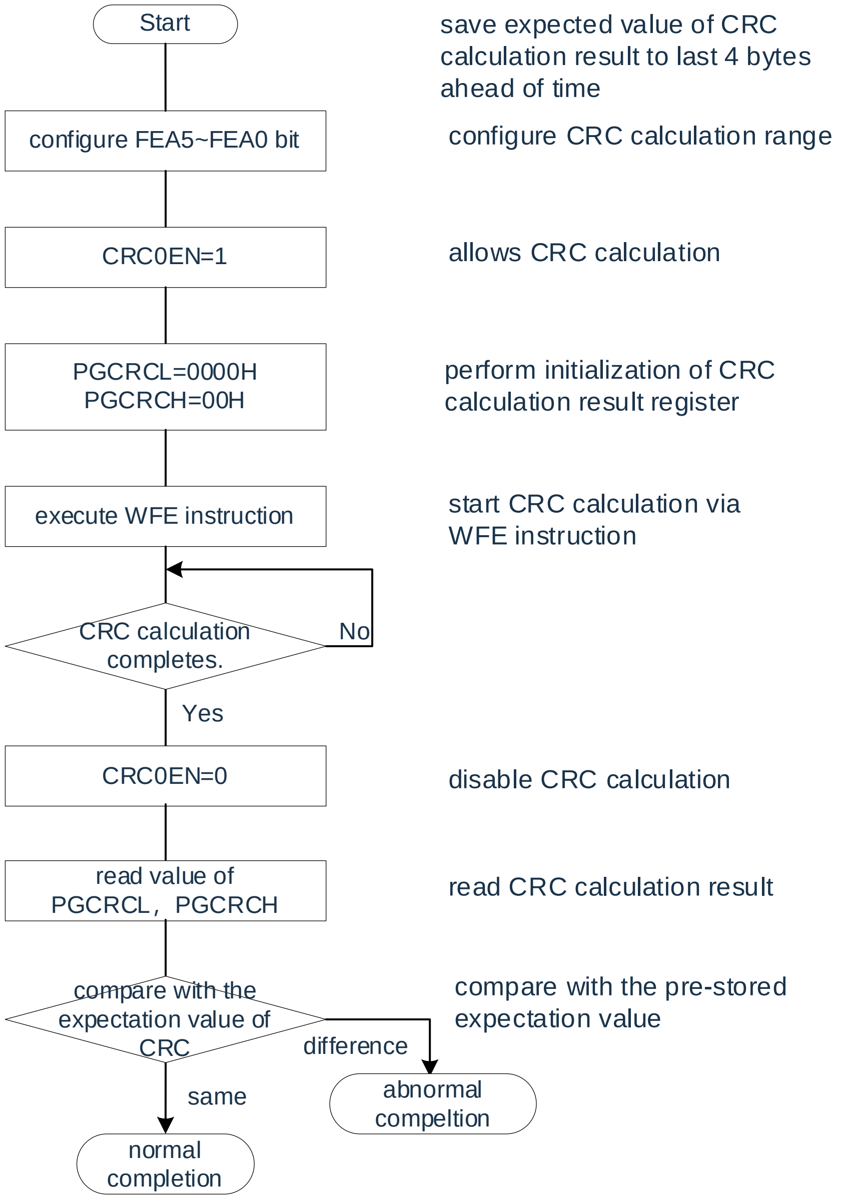
PGCRC15~0	The result of a high-speed CRC
0000H~FFFFH	Save the results of high-speed CRC.

Note that the PGCRCL register can only be written if the CRC0EN (bit7 of the CRC0CTL register) bit is “1”.

A flowchart of the flash CRC operation function (high-speed CRC) is shown in Figure 24-3.

< Operation Flow >

Figure 24-3 Flow chart of flash CRC operation function (high-speed CRC)


Start
configure FEA5~FEA0 bit
CRC0EN=1
PGCRCL=0000H
PGCRCH=00H
execute WFE instruction
CRC calculation completes.
No
Yes
CRC0EN=0
read value of
PGCRCL， PGCRCH
compare with the expectation value of CRC
difference
abnormal compeltion
same
normal completion
save expected value of CRC calculation result to last 4 bytes ahead of time
configure CRC calculation range
allows CRC calculation
perform initialization of CRC calculation result register
start CRC calculation via WFE instruction
disable CRC calculation
read CRC calculation result
compare with the pre-stored expectation value

Note 1. Only the code flash memory is an object for CRC operations.

2. The expected value of the CRC operation must be saved in the area behind the operation range in the code flash.

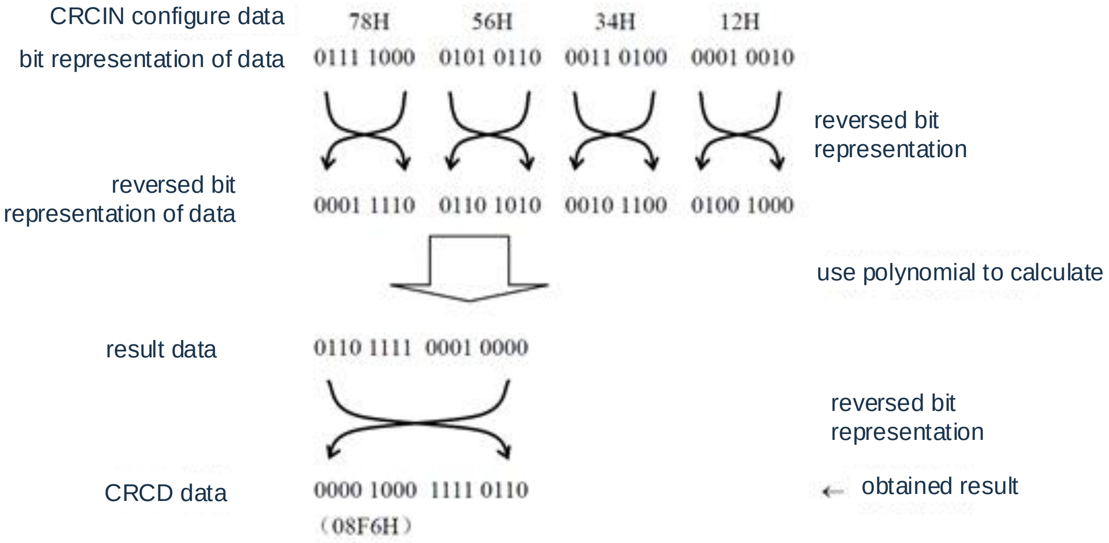
24.3.2 CRC operation function (general CRC)

In order to ensure safety during operation, the IEC61508 standard requires that the data need to be confirmed even in CPU operation.

This generic CRC can be used as a peripheral function for CRC operations in CPU operation. Generic CRC is not limited to code flash areas but can be used for multi-purpose inspections. The data to be confirmed is specified through the software (user program). The CRC operation function in sleep mode can only be used during DMA transmission.

CRC arithmetic functions can be used in either the main system clock operation mode or the secondary system clock operation mode.

CRC generates polynomials using CRC-16-CCITT's “$X^{16}+X^{12}+X^{5}+1$”. Because the communication is considered to be carried out in LSB first, the calculation is performed after the bit order of the input data is reversed. For example, in the case of sending data “12345678H” from the LSB, follow the requirements of “78H”, “56H”, “34H”, “34H” The “12H” sequence writes the value to the CRCIN register, and the value of “08F6H” is obtained from the CRCD register. This is the result of CRC operations for the following bit order after inverting the bit order of the data “12345678H”.



Note During the execution of the program, because the modulator rewrites the set line of the software breakpoint as a breakpoint instruction, if you set the software breakpoint in the object area of the CRC operation, the CRC operation result is different.

24.3.2.1 CRC input registers (CRCINs)

This is the 8-bit register that sets the CRC calculation data for the general-purpose CRC. The range that can be set is “00H~FFH”.

The CRCIN registers are set via 8-bit memory operation instructions. After the reset signal is generated, the value of this register becomes “00H”.

Figure 24-4 Format of CRC input register (CRCINs)

Address: 400433ACH After reset: 00HR/W

Symbol	7	6	5	4	3	2	1	0
CRCIN								

Bit7~0	Function
00H~FFH	Data input

24.3.2.2 CRC data register (CRCD)

This is the register that holds the results of a general-purpose CRC operation. The range that can be set is “0000H~FFFFH”.

After writing the CRCIN register, a CPU/peripheral hardware clock (f_{C}LK) is passed to save the CRC operation results to the CRCD Register. The CRCD registers are set via 16-bit memory operation instructions. After the reset signal is generated, the value of this register changes to “0000H”.

Figure 24-5 Format of CRC data register (CRCD)

Address: 400432FAH after reset: 0000HR/W

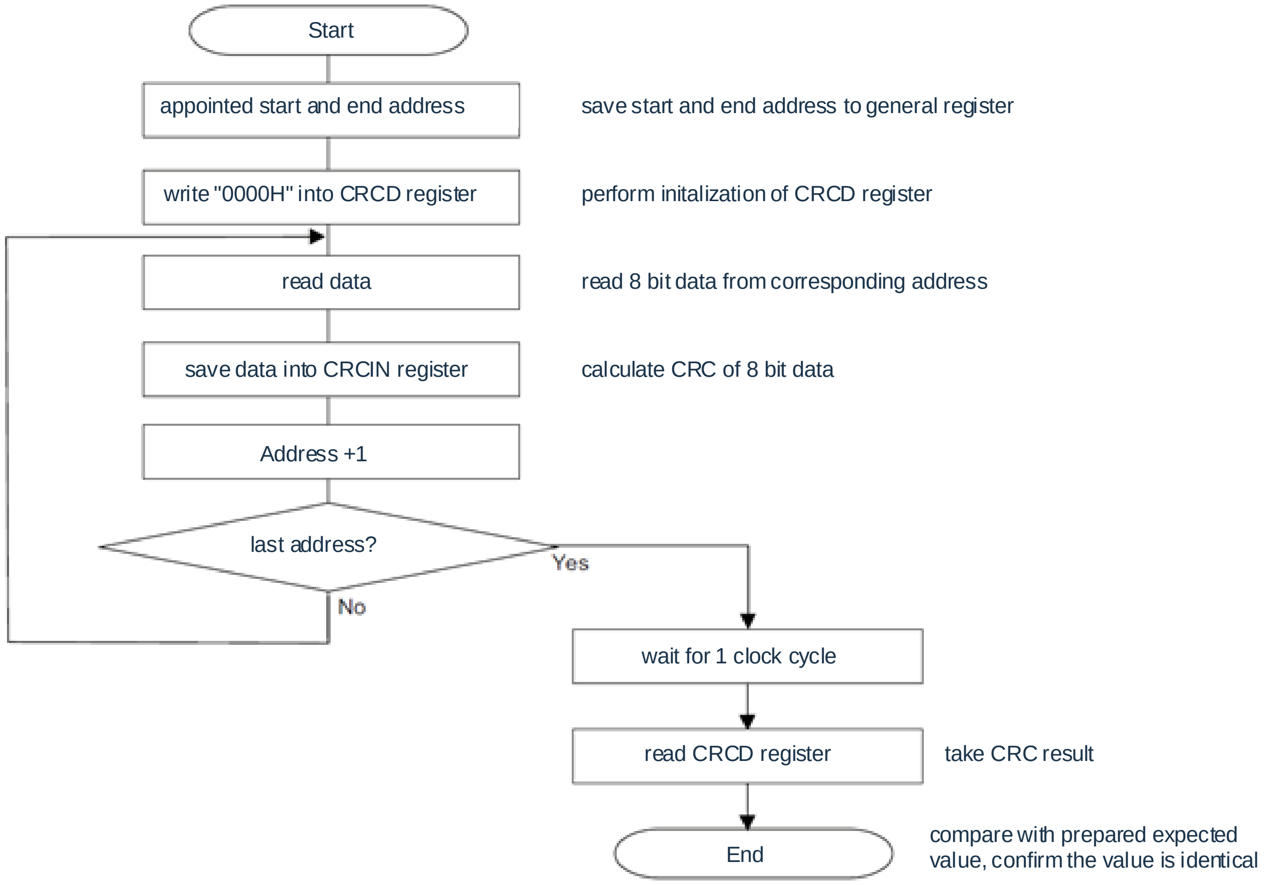
Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCD																

Note 1. To read the write value of a CRCD register, the CRCD register must be read before the CRCIN register is written.

2. If the write operation of the CRCD register competes with the saving of the operation result, the write operation is ignored.

< Operation process >

Figure 24-6 Flowchart of CRC operation function (general 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

24.3.3 RAM parity error detection function

The IEC60730 standard requires confirmation of RAM data. Therefore, the CMS32L051's RAM appends 1-bit parity bit every 8 bits. The RAM parity error detection function appends parity bits when writing data, checks parity bits when reading data, and can produce a reset when parity errors occur.

24.3.3.1 RAM parity error control register (RPECTL)

This register controls the false positive bit of parity and the reset due to parity errors.

The RPECTL registers are set via 8-bit memory operation instructions.

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

Figure 24-7 Format of RAM parity error control register (RPECTL)

Address: 40020425H After reset: 00HR/W

symbol	7	6	5	4	3	2	1	0
RPECTL	RPERDIS	0	0	0	0	0	0	RPEF

RPERDIS	Parity false reset of the mask flag
0	Allows parity error reset.
1	Parity error reset is prohibited.

RPEF	Parity error status flag
0	No parity errors occurred.
1	A parity error occurred.

Note Parity bits are appended when writing data and checked when reading data.

Therefore, to allow RAM parity error reset (RPERDIS=0), the “RAM area used” must be paired when accessing the data and before reading the data Initialize.

Because it is running on a pipeline, the CPU performs a pre-read, and a RAM parity error may occur due to the uninitialized RAM area before reading the RAM area used. Therefore, to allow the generation of RAM parity error reset (RPERDIS=0), the instructions must be executed from the RAM area to the “RAM area used.” +10 bytes” area is initialized.

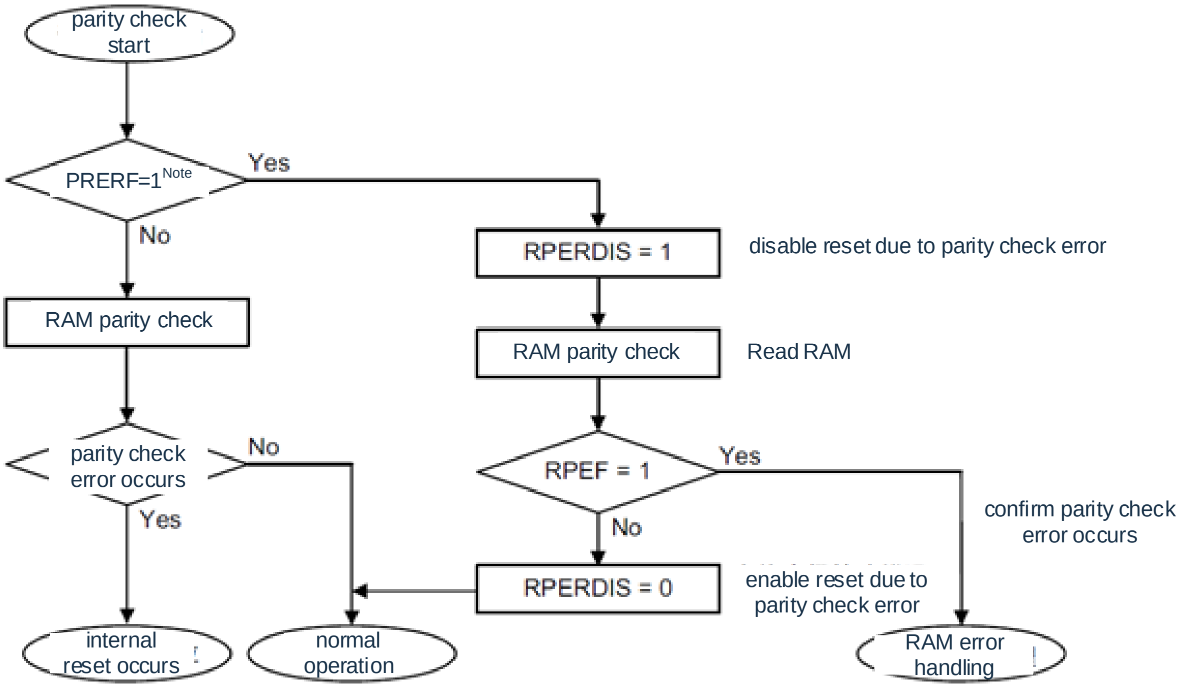
Note 1 The initial state is to allow for parity error reset (RPERDIS=0).

2. Even if the parity error reset is disabled (RPERDIS=1), the RPEF flag is set to “1” in the event of a parity error. If the RPEF bit is set to allow parity error reset (RPERDIS=0) in the state of “1”, the RPERDIS is cleared “0” A parity error is generated when the reset occurs.

3. Set the RPEF flag of the RPECTL register to “1” due to RAM parity error, and RPEF is reset by writing “0” or all the reset sources Flag clear “0”. When the RPEF flag is “1”, the RPEF flag remains in the state of “1” even if the RAM without parity errors is read.

4. The range of RAM parity detection does not include general-purpose registers.

Figure 24-8 Flow of RAM parity check



Note For confirmation of the internal reset of RAM parity errors, see Chapter 21 Reset Function.

24.3.4 SFR protection function

In order to ensure safety during operation, the IEC61508 standard requires that even if the CPU is out of control, important SFRs need to be protected from overriding the SFR protection function for the protection of port functions, interrupt functions, clock control functions, voltage detection circuitry and RAM The parity error detection function controls the data of the register.

If the protection function is set to SFR, the write operation of the protected SFR is invalid, but it can be read normally.

24.3.4.1 SFR protection control register (SFRGD)

This register controls whether the SFR protection function is effective.
The SFR protection function uses the G-COMP BIT, GPORT BIT, GINT BIT, and GCSC bits.
The SFRGD register is set via an 8-bit memory operation command.
After the reset signal is generated, the value of this register becomes “00H”.

Figure 24-9 Format of SFR protection control register (SFRGD)

Address: 40040478H After reset: 00HR/W

Symbol	7	6	5	4	3	2	1	0
SFRGD	0	0	0	0	0	GAfterRT	GINT	GCSC

GPORT	Protection of control registers for port functions
0	Invalid. Control registers that can read and write port functions.
1	Valid. The port function of the control register is invalid and can be read. [SFR] PMxx protected, PUxx, PDxx, POMxx, PMCxx, PxxCFG, PIORx[Note]

GINT	Interrupt function register protection
0	Void. Control registers that can read and write interrupts.
1	Effective. The write operation of the control register of the interrupt function is invalid and can be read. [SFR protected] IFxx , MKxx, PRxx, EGPx, EGNx

GCSC	Protection of control registers for clock control functions, voltage detection circuits, and RAM parity error detection functions
0	Void. Control registers that can read and write clock control functions, voltage detection circuitry, and RAM parity error detection functions.
1	Effective. The write operation of the control register of the clock control function, voltage detection circuit, and RAM parity error detection function is invalid and can be read. [Protected SFR]CMC, CSC, OSTS, CKC, PERx, OSMC, LVIM, LVIS, RPECTL

Note Pxx (port registers) are not protected.

24.3.5 Frequency detection function

The IEC60730 standard requires confirmation of whether the oscillation frequency is normal.

The frequency detection function uses the clock frequency (f_{CLK}) of the CPU/peripheral hardware and can determine whether the ratio relationship between the two clocks is correct by measuring the Timer40 channel 1 input pulse.

However, if 1 clock or 2 clocks stop oscillating, the ratio of 2 clocks cannot be determined.

< clock to compare >

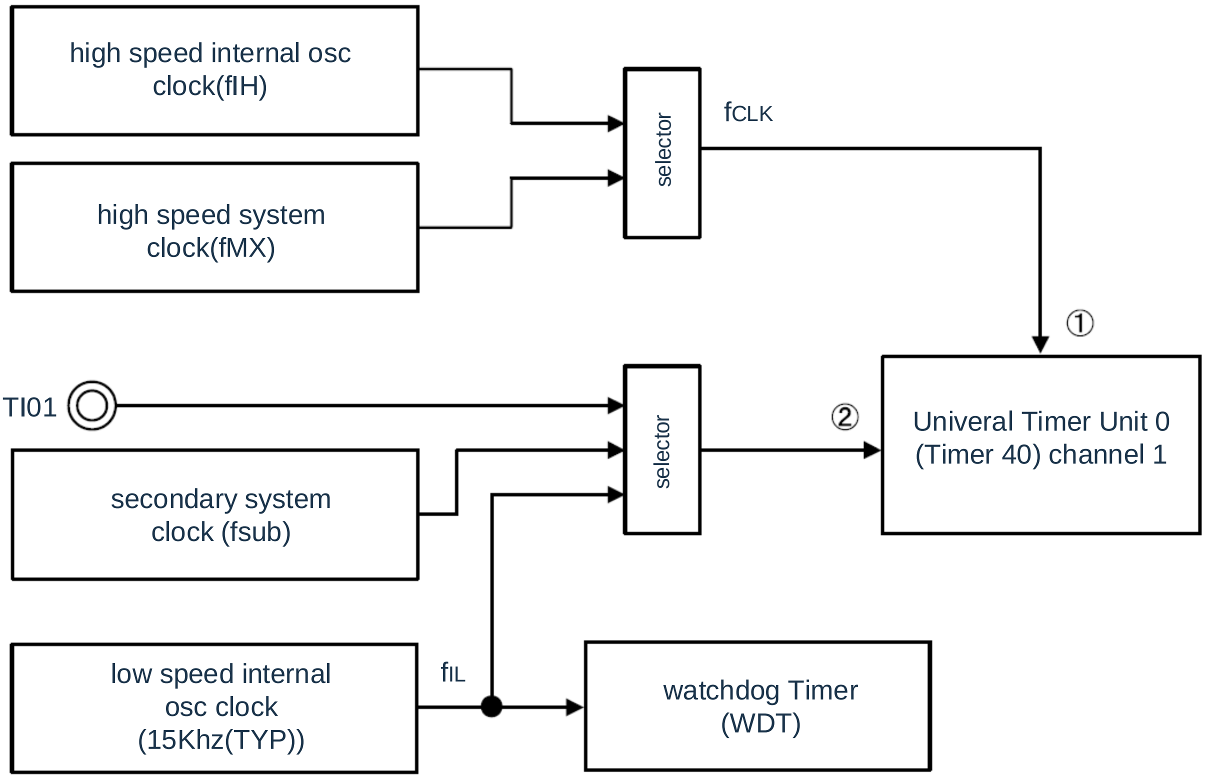
(1) Clock frequency of CPU/peripheral hardware (f_{CLK}):

- High speed internal oscillator clock (f_{IH}).
- High Speed System Clock (f_{MX}).

(2) Timer40 channel 1 input:

- Timer input for channel 1 (TI01).
- Low-speed internal oscillator clock (f_{IL}: 15kHz (TYP.))
- Sub-System Clock (f_{SUB}) Note

Figure 24-10 Structure of frequency detection function


high speed internal osc clock(fIH)
high speed system clock(fMX)
selector
fCLK
①
TI01
secondary system clock (fsub)
selector
②
Univeral Timer Unit 0 (Timer 40) channel 1
low speed internal osc clock (15Khz(TYP))
fIL
watchdog Timer (WDT)

When the measurement result of the input pulse interval is an outlier, it can be judged as “clock frequency anomaly”. For the measurement method of the input pulse interval, refer to “5.8.4”.

Note Only products with a built-in sub-system clock can be selected.

24.3.5.1 Timer input selection register 0 (TIS0)

Refer to Section 5.3.8 for a description of the registers.

24.3.6 A/D test function

The IEC60730 standard requires testing of A/D converters. This A/D test function is performed with a positive (+) reference voltage for the A/D converter, a negative (–) reference, an analog input channel (ANI), an output voltage for a temperature sensor, and an internal reference A/D conversion to confirm that the A/D converter is operating properly.

The analog multiplexer can be verified by following these steps:

① The ANIx pin is selected as the A/D conversion object through the ADTES register (ADTES2, ADTES1, ADTES0=0, 0, 0).

② A/D conversion of the ANIx pin (conversion result 1-1).

③ The negative (–) reference voltage of the A/D converter is selected as the A/D conversion object through the ADTES register (ADTES2, ADTES1, ADTES0=0, 0, 1).

④ A/D conversion of the negative (–) reference voltage of the A/D converter (results 2-1).

⑤ The ANIx pin is selected as the A/D conversion object through the ADTES register (ADTES2, ADTES1, ADTES0=0, 0, 0).

⑥ A/D conversion of the ANIx pin (conversion results 1-2).

⑦ The positive (+) reference voltage of the A/D converter is selected from the ADTES register as the A/D conversion object (ADTES2, ADTES1, ADTES0=1, 0, 1).

⑧ A/D conversion of the positive (+) reference voltage of the A/D converter (conversion results 2-2).

⑨ The ANIx pin is selected as the A/D conversion object through the ADTES register (ADTES2, ADTES1, ADTES0=0, 0 , 0).

⑩ A/D conversion of the ANIx pin (conversion result 1-3).

⑪ Verify that the Conversion Results 1-1, Conversion Results 1-2, and Conversion Results 1-3 are the same.

⑫ Confirm that the A/D conversion result of “Conversion Result 2-1” is all “0” and the A/D of “Conversion Result 2-2” The result of the conversion is all “1”. With these steps, you can select an analog multiplexer and verify that the wiring is not broken.

Note 1 During the conversion of (1) to (10), if the analog input voltage is variable, other methods must be used to confirm the analog multiplexer.

2. The conversion result contains errors, so the error must be properly considered when comparing the conversion results.

24.3.6.1 A/D test registers (ADTES).

This register selects the A/D converter's positive (+) reference, negative (–) reference, analog input channel (ANIxx), temperature sensor output voltage, and internal reference voltage (1.45V). as an A/D conversion object.

When used as an A/D test function, the following settings are made:

- When measuring the zero scale, select the negative (–) reference voltage as the A/D conversion object.
- When measuring the full scale, select the positive (+) reference voltage as the A/D conversion object.

Refer to 11.2 10.

24.3.6.2 The analog input channel specifies registers (ADS).

This register specifies the input channel for the analog voltage of the A/D conversion.

To measure an ANIxx, temperature sensor output, or internal reference voltage (1.45V) through the A/D test function, the A/D test register (ADTES) must be set to “00H”.

Refer to 11.2.7 for a description of the registers.

24.3.7 Digital output signal level detection function for input/output pin

The IEC60730 standard requires confirmation of proper I/O functionality.

The digital output signal level detection function of the input/output pin reads the digital output level of the pin when the pin is in output mode.

24.3.7.1 Port mode selection register (PMS)

This register selects whether to read the value of the port's output latch or the output level of the read pin when the pin is in output mode (the PMmn bit of the port mode register (PMm) is “0”).

The PMS registers are set via 8-bit memory operation instructions.

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

Figure 24-11 Format of port mode selection register (PMS)

Address: 4004087BH after reset: 00HR/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	Read the digital output level of the pin.

Note 1 For pins that use the pulse output forced cutoff function of timer M to make the pin into a high impedance state, if the digital output level of the pin is read, the read value is “0”.

Notes m=0~7,12~14
n=0~7

24.3.8 Product unique identification register

A product's unique identification is ideal for:

- Used as a serial number (e.g. USB character serial number or other terminal application).
- Used as a password, this unique identifier is used in conjunction with a software encryption and decryption algorithm when writing flash memory to improve the security of the code in flash memory.
- Used to activate the bootstrap process with a safety mechanism

The reference number provided by the 128-bit unique product identifier is unique to any BAT32 microcontroller and is unique in any case. Under no circumstances can the user modify this identity.

Base address: 0x0050_0894

Address offset: 0x00

Read-only, whose values are written at the factory

U_ID[31:0]

Address offset: 0x04

Read-only, whose values are written at the factory

U_ID[63:32]

Address offset: 0x08

Read-only, whose values are written at the factory

U_ID[95:64]

Address offset: 0x0C

Read-only, whose values are written at the factory

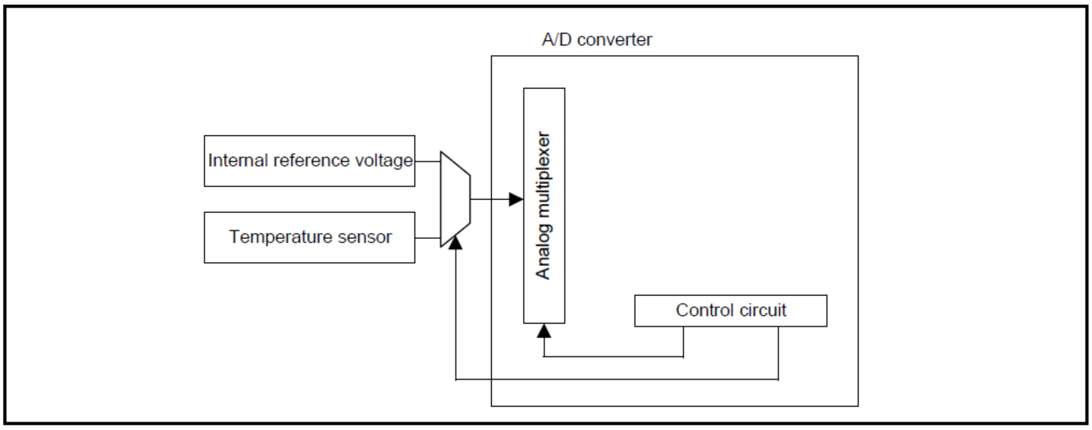
U_ID[127:96]

Chapter 25 Temperature Sensor

25.1 Function of temperature sensor

The on-chip temperature sensor measures and monitors the core temperature of the product, thus ensuring the reliable operation of the product. The voltage output by the temperature sensor is proportional to the core temperature, and there is a linear relationship between voltage and temperature. Its output voltage is supplied to the ADC for conversion. FigureFigure25-1shows a block diagram of a temperature sensor.

Figure25-1 Block diagram of temperature sensor



25.2 Register for temperature sensor

25.2.1 Temperature sensor calibration data register TSN25

Address: 0x0050_066C

symbol	15	0	After reset	R/W
TSN25	TSN25[11:0]		-	R

A read-only register that records calibration data for the temperature sensor1 is automatically loaded when power is turned on or reset starts, and each chip has its own calibration data.

25.2.2 Temperature sensor calibration data register TSN85

Address: 0x0050_0668

symbol	15	0	After reset	R/W
TSN85	TSN85[11:0]		-	R

A read-only register that records calibration data for temperature sensors2 is automatically loaded when power is turned on or reset starts, with each chip having its own calibration data.

25.3 Instructions for use with the temperature sensor

25.3.1 How the temperature sensor is used

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 of the temperature sensor during temperature measurement (V)
T1: Temperature at the first point for experimental measurements (°C)
V1: Voltage output when the temperature sensor measures T1 (V)
T2: Temperature at the second point for experimental measurements (°C)
V2: Voltage output when the temperature sensor measures T2 (V)
Slope: The temperature slope of the temperature sensor (V/°C), slope = (V2 - V1) / (T2 - T1)

Different sensors have different characteristics, so we recommend measuring the following two different sample temperatures:

1、 The A/D converter is used to measure the voltage V1 output by the temperature sensor at temperature T1.

2、 The A/D converter is used to measure the voltage V2 output by the temperature sensor at the second temperature T2.

3、 The temperature slope (slope = (V2 - V1) / (T2 - T1)) is calculated from the two results

4、 Subsequently, the temperature is obtained by substituting the slope into the formula for the temperature characteristics (T = (Vs -V1) / slope + T1).

25.3.2 How to use the temperature sensor

Method 1: In this product, the TSN25 register stores the voltage conversion value (CAL25) of the temperature sensor measured under the conditions of Ta=Tj=25°C and AVCC0=3.0v. The TSN85 register stores the voltage conversion values of the temperature sensor measured at Ta=Tj=125°C and AVCC0=3.0v (CAL125). Using these two sets of values, the temperature slope can be calculated:

slope = (V2 – V1) / (125 - 25).

V1 = 3.0 × CAL25 / 256 [V]
V2 = 3. 0 × CAL125 / 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 (°C)
Vs: Output voltage of the temperature sensor obtained using an A/D converter at T temperature (V)

Method 2: If you use the temperature slope given in “Electrical Characteristics”, you can directly calculate the measured temperature using the following formula:

T = (Vs – V1) / slope + 25 [°C]

Note: This method produces a temperature that is less accurate than Method 1 measurements.

Chapter 26 Option Byte

26.1 Function of option byte

CMS32L051's flash memory 000C0H~000C3H, 500004H is the option byte area.

Option bytes consist of user option bytes (000C0H~000C2H) and flash data protection option bytes (000C3H, 500004H). When the power is turned on or reset starts, the specified function is automatically set by referring to the option byte. When using this product, the following functions must be set by option bytes. For bits that do not have a configured feature, you cannot change the initial value.

Note Regardless of whether or not to use each feature, the option byte must be set.

26.1.1 User option bytes (000C0H~000C2H)

(1) 000C0H

● Operation of the watchdog timer

- Allow or disallow the operation of the counter.
- Allow or stop the operation of the counter in sleep/deep sleep mode.

● The setting of the overflow time of the watchdog timer

- The watchdog timer is set during the window open

● The setting of the interval interrupt of the watchdog timer

- Use or do not use interval interrupts.

(2) 000C1H

● Setting of the LVD operating mode

- Interrupt & reset mode
- Reset mode
- Interrupt mode
- The LVD is OFF (using an external reset input from the RESETB pin).

● Setting of the LVD detection level (VLVDH, VLVDL, VLVD).

Note 1 When the supply voltage rises, the reset state must be maintained by voltage detection circuit or external reset before the supply voltage reaches the operating voltage range shown in the AC characteristics of the data sheet; When the supply voltage drops, it must be reset through deep sleep mode transfer, voltage detection circuitry, or external reset before the supply voltage drops below the operating voltage range.

The operating voltage range depends on the setting of the user option byte (000C2H).

(3) 000C2H

● Frequency setting of the high-speed internal oscillator

- Choose from 1MHz to 32MHz, 48MHz, 64MHz.

26.1.2 Flash data protection option bytes (000C3H, 500004H).

- Control of flash data protection during on-chip debugging

Level0: Allows read/write/erase operations on flash data via debugger

Level1: Allows chip full erase of flash data via debugger, not allowed to read or write operations.

Level2: Flash data is not allowed to be manipulated through debuggers.

26.2 Format of user option byte

Figure 26-1 Format of user option bytes (000C0H)

Address: 000C0H

Symbol	7	6	5	4	3	2	1	0
	WDTINT	WINDOW1	WINDOW0	WDTON	WDCS2	WDCS1	WDCS0	WDSTBYON

WDTINT	Interval interruption of watchdog timer use/non-use
0	Interval interrupts are not used.
1	When 75% of the overflow time is reached +1/2f_{IL}, an interval interrupt occurs.

WINDOW1	WINDOW0	The window of the watchdog timer opens during Note 2
0	-	Disable settings.
1	0	75%
1	1	100%

WDTON	Counter operation control of the watchdog timer
0	Disable the counter to operation (stop counting after resetting is released).
1	Enable the counter to operation (start counting after resetting is released).

WDCS2	WDCS1	WDCS0	Overflow time of the watchdog timer (f_{IL}=20kHz(MAX.))
0	0	0	$2^6/f_{IL}$(3.2ms)
0	0	1	$2^7/f_{IL}$(6.4ms)
0	1	0	$2^8/f_{IL}$(12.8ms)
0	1	1	$2^9/f_{IL}$(25.6ms)
1	0	0	$2^{11}/f_{IL}$(102.4ms)
1	0	1	$2^{13}/f_{IL}$(409.6ms)
1	1	0	$2^{14}/f_{IL}$(819.2ms)
1	1	1	$2^{16}/f_{IL}$(3276.8ms)

WDSTBYON	Counter operation control (sleep mode) for watchdog timers
0	In sleep mode, stop the operation of the counter Note 1.
1	In sleep mode, the operation of the counter is allowed.

Note 1 When the WDSTBYON bit is “0”, it has nothing to do with the values of the WINDOW1 bit and the WINDOW0 bit, which is 100% during window open.

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

Figure 26-2 Format of user option bytes (000C1H) (1/4)

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD setting (interrupt & reset mode)

Detection voltage			Setting value of the option byte						
INLVDH		INLVDL	VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rise	falling	falling						LVIMDS1	LVIMDS0
1.98V	1.94V	1.84V	0	0	1	1	0	1	0
2.09V	2.04V					0	1		
3.13V	3.06V					0	0		
2.61V	2.55V	2.45V		1	0	1	0		
2.71V	2.65V					0	1		
3.75V	3.67V					0	0		
2.92V	2.86V	2.75V		1	1	1	0		
3.02V	2.96V					0	1		
4.06V	3.98V					0	0		
—			Values other than those above are prohibited.						

Note that you must write “1” to bit4.

Note 1 For more information about LVD circuits, refer to Chapter 2 3, Voltage Detection Circuits.

2. The detection voltage is TYP Value. For details, please refer to the LVD circuit characteristics in the data sheet.

Figure 26-2 Format of User Option Bytes (000C1H) (2/4)

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD setting (reset mode)

Detection voltage		Setting value of the option byte						
V_{LVD}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rise	falling						LVIMDS1	LVIMDS0
1.88V	1.84V	0	0	1	1	1	1	1
1.98V	1.94V		0	1	1	0		
2.09V	2.04V		0	1	0	1		
2.50V	2.45V		1	0	1	1		
2.61V	2.55V		1	0	1	0		
2.71V	2.65V		1	0	0	1		
2.81V	2.75V		1	1	1	1		
2.92V	2.86V		1	1	1	0		
3.02V	2.96V		1	1	0	1		
3.13V	3.06V		0	1	0	0		
3.75V	3.67V		1	0	0	0		
4.06V	3.98V		1	1	0	0		
—		Values other than those above are prohibited.						

Note that you must write “1” to bit4.

Note 1 For more information about LVD circuits, refer to Chapter 2 3, Voltage Detection Circuits.

2. The detection voltage is TYP Value. For details, please refer to the LVD circuit characteristics in the data sheet.

Figure 26-2 Format of User Option Bytes (000C1H) (3/4)

Address: 000C1H Note

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD setting (interrupt mode).

Detection voltage		Setting value of the option byte						
VLVD		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rise	falling						LVIMDS1	LVIMDS0
1.88V	1.84V	0	0	1	1	1	0	1
1.98V	1.94V		0	1	1	0		
2.09V	2.04V		0	1	0	1		
2.50V	2.45V		1	0	1	1		
2.61V	2.55V		1	0	1	0		
2.71V	2.65V		1	0	0	1		
2.81V	2.75V		1	1	1	1		
2.92V	2.86V		1	1	1	0		
3.02V	2.96V		1	1	0	1		
3.13V	3.06V		0	1	0	0		
3.75V	3.67V		1	0	0	0		
4.06V	3.98V		1	1	0	0		
—		Values other than those above are prohibited.						

Note You must write “1” to bit4.

Note 1 For more information about LVD circuits, refer to Chapter 2 3, Voltage Detection Circuits.

2. The detection voltage is TYP Value. For details, please refer to the LVD circuit characteristics in the data sheet.

Figure 26-2 Format of user option bytes (000C1H) (4/4)

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD is OFF (Use RESETB External reset input for the pin)

Detection voltage		The setting value of the option byte						
V_{LCEOH}		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	Mode setting	
rise	decline						LVIMDS1	LVIMDS0
—	—	1	×	×	×	×	×	1

Note 1 You must write “1” to bit4.

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 working voltage range shown in the AC characteristics of the data sheet; When the supply voltage drops, it must be reset through sleep mode transfer, voltage detection circuitry, or external reset before the supply voltage drops below the operating voltage range.

   The operating voltage range depends on the setting of the user option byte (000C2H).

Note 1 ×: Ignore

2. For details of the LVD circuit, please refer to “Chapter 2 3 Voltage Detection Circuit”.

3. The detection voltage is TYP Value. For details, please refer to the LVD circuit characteristics in the data sheet.

Figure 26-3 Format of User Option Bytes (000C2H)

Address: 000C2H

7	6	5	4	3	2	1	0
1	1	1	FRQSEL4	FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0

FRQSEL4	FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0	The clock frequency of the high-speed internal oscillator	
					f_{HOCO}	f_{IH}
0	1	0	0	0	64MHz	64MHz
0	0	0	0	0	48MHz	48MHz
0	1	0	0	1	64MHz	32MHz
0	0	0	0	1	48MHz	24MHz
0	1	0	1	0	64MHz	16MHz
0	0	0	1	0	48MHz	12MHz
0	1	0	1	1	64MHz	8MHz
0	0	0	1	1	48MHz	6MHz
0	1	1	0	0	64MHz	4MHz
0	0	1	0	0	48MHz	3MHz
0	1	1	0	1	64MHz	2MHz
Beyond the above					Disable settings.	

Note 1 You must write “1” for bits7 to 5.

2. The operating frequency range and operating voltage range vary depending on the operating mode of the flash. For details, please refer to the AC characteristics in the data sheet.

26.3 Format of flash data protection option bytes

The format of the Flash Data Protection Options byte is as follows.

Figure 26-4 Format of flash data protection option bytes (000C3H)

Address: 000C3H

Symbol	7	6	5	4	3	2	1	0
	OCD[7:0]							

Address: 500004H

Symbol	7	6	5	4	3	2	1	0
	OCDM[7:0]							

OCDM	OCDEN	Control of flash data protection
3C	C3	It is not allowed to manipulate flash data throughd-ebuggers.
A value other than 3C	C3	Allows chip full erase operation of flash data through debugger, and does not allow read and write operations.
Beyond the above		Allows read/write/erase operations on flash data via debugger

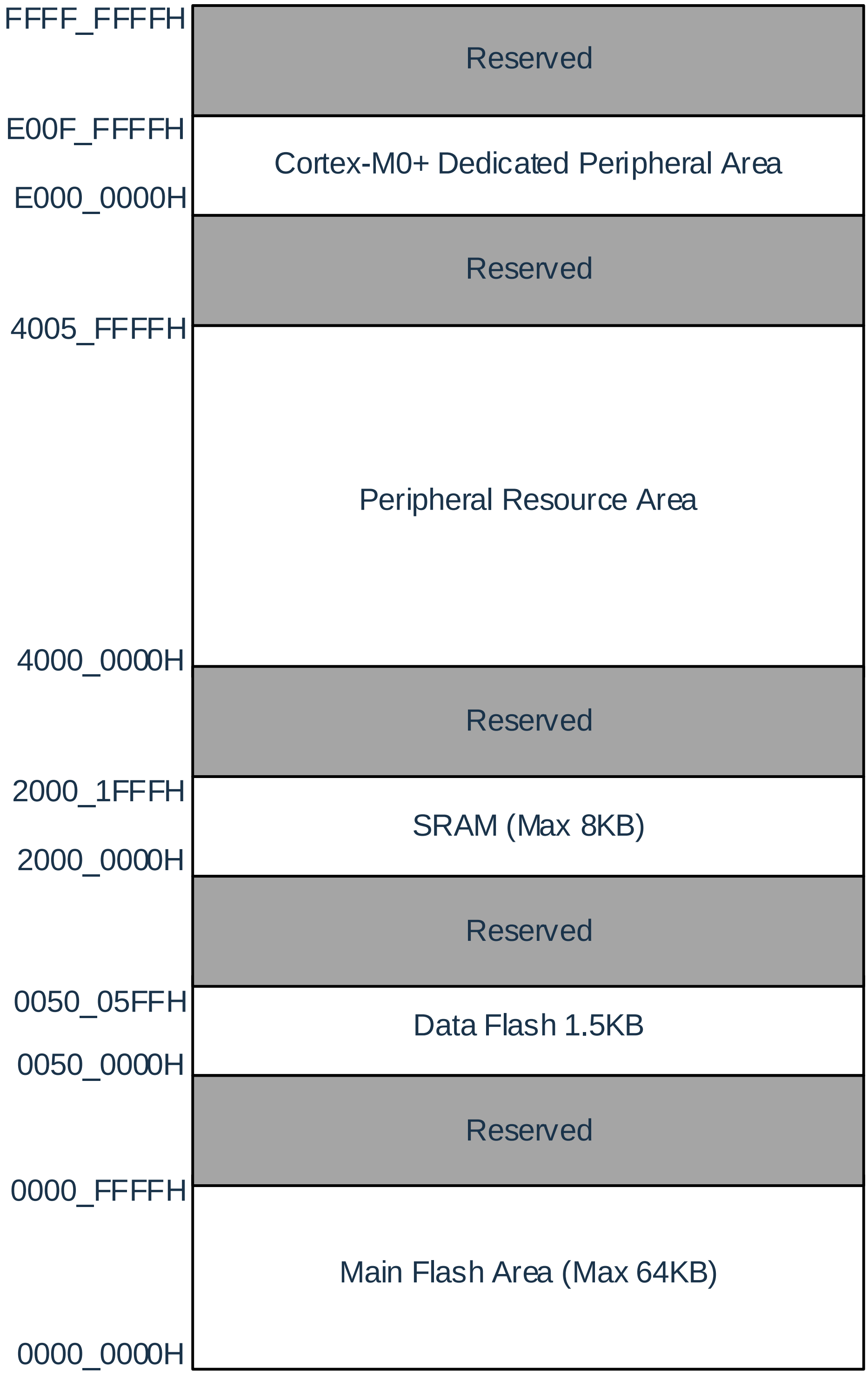
Notice The address 50_0004H belongs to the data flash area. If you use this address for data storage, you need to make sure that the value will not cause the protection option to be set by mistake first.

Chapter 27 FLASH Control

27.1 Description of FLASH control

This product contains a 64KByte capacity FLASH memory, divided into 128 sectors, each with a capacity of 512-byte. It can be used as program memory, data memory. This module supports erasing, programming, and reading of this memory.

27.2 Structure of FLASH memory



27.3 Registers for controlling FLASH

The registers that control FLASH are as follows:

- Flash write protection register (FLPROT).
- Flash operation control register (FLOPMD1, FLOPMD2).
- Flash erase mode control register (FLERMD)
- Flash status register (FLSTS).
- Flash full-chip erase time control register (FLCERCNT).
- Flash sector erasure time control register (FLSERCNT).
- Flash write time control register (FLPROCNT).
- Flash mode time control register (FLNVSCNT/FLPRVCNT/FLLERVCNT).

27.3.1 Flash write protection register (FLPROT)

Flash protection registers are used to protect FLASH operating control registers.

Address: 0x40020020 After reset: 00000000H R/W

symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLPROT	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	-	-	PRKEY[7:1]							WRP

WRP	Operation registers (FLOPMD1/FLOPMD2) are write-protected
0	Overwriting of FLOPMD1/FLOPMD2 is not allowed
1	Rewriting of FLOPMD1/FLOPMD2 is allowed

PRKEY[7:1]	WRP write protection
78h	Rewriting of WRP is allowed
Beyond the above	Overriding WRP is not allowed

27.3.2 FLASH operation control registers (FLOPMD1, FLOPMD2)

Flash operation control registers for setting the erase and write operations of FLASH.

Address: 0x40020004 After reset: 000000000H 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: 00HR/W

symbol	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLOPMD2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	-	-	FLOPMD2[7:0]							

FLOPMD1	FLOPMD2	OPERATE
55	AA	Erase
AA	55	write
00	00	Read out
Beyond the above		Set Prohibited

27.3.3 Flash erase control register (FLERMD)

Flash erasure control register to set the type of FLASH erase operation.

Address: 0x4002000C after reset: 00HR/W

Symbol	7	6	5	4	3	2	1	0
FLERMD	0	0	0	ERMD1	ERMD0	0	0	0

ERMD1	ERMD0	OPERATE
0	0	sector erases, no hardware verification is performed after wiping
1	0	sector erase, hardware verification after wiping
0	1	chip erase note
1	1	Set Prohibited

Note: chip wipe erases only the code flash area, not the data flash area. And chip erasure does not support hardware verification.

27.3.4 Flash status register (FLSTS)

The status register allows you to query the status of the FLASH controller.

Address: 0x40020000 after reset: 00HR/W

Symbol	7	6	5	4	3	2	1	0
FLSTS	0	0	0	0	0	EVF	0	OVF Note

OVF	The FLASH erasing operation is finished with the flag
0	The FLASH erase operation did not complete
1	The FLASH erasing operation is complete

Note: OVF requires software to write “1” to clear it. If it is not cleared, the next erase and write operation cannot occur.

EVF	FLASH erases the hardware check error flag
0	After FLASH erasing, no errors occur in hardware verification
1	After FLASH erasing, an error occurred in hardware verification

Note: EVF requires software to write “1” to clear it.

27.3.5 Flash full-chip erase time control register (FLCERCNT)

The FLCERCNT register allows the time of FLASH full film erasure to be set.

Address: 0x40020010 After reset: Indefinite R/W

symbol

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
load	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

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

FLCERCNT

Load	Erase the selection of the time setting Note
0	Use the erase time set by the hardware
1	Use the erase time set by the software (FLCERCNT [9:0]).

Note: When the master clock is an internal high-speed OCO or an external input clock <=20M, the hardware setting time can be used and FLCERCNT is not set.

FLCERCNT[9:0]	Software erase time setting
Chip erasure time = (CERCNT*2048*Tfclk), which requires a hardware requirement of >20ms	

27.3.6 Flash sector erase time control register (FLSERCNT)

The FLSERCNT register allows the time of the FLASH full film erase to be set.

Address: 0x40020014 After reset: Indefinite R/W

symbol

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
load	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

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

FLSERCNT

Load	Erase the selection of the time setting Note
0	Use the erase time set by the hardware
1	Use the erase time set by the software (FLSERCNT [9:0]).

Note: When the master clock is internal high-speed OCO or the external input clock <=20M, the hardware setting time can be used and FLSERCNT is not set.

FLSERCNT[9:0]	Software erase time setting
sector erasure time = (SERCNT*256*Tfclk), which meets the hardware requirements of >4ms	

27.3.7 Flash write time control register (FLPROCNT).

The FLPROCNT register allows you to set the FLASH WORD write time.

Address: 0x4002001C After reset: Indefinite R/W

symbol

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Load1	-	-	-	-	-	-	FLPGSCNT[8:0]								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Load0	-	-	-	-	-	-	FLPROCNT[8:0]								

FLPROCNT

Load0	Write time (Tprog) setting Note
0	Use the write time set by the hardware
1	Use the erase time set by the software (FLPROCNT [9:0]).

Note: When the master clock is an internal high-speed OCO or an external input clock <=20M, the hardware setting time can be used without setting FLPROCNT.

FLPROCNT[8:0]	Software erase time setting
Write time = (PROCNT*4*Tfclk), subject to hardware requirements >24us	

Load1	Write Action Settling Time (Tpgs) setting Note
0	Use the hardware-set write action settling time
1	Use the erase time set by the software (FLPGSCNT8:0]).

Note: When the master clock is an internal high-speed OCO or an external input clock <=20M, the hardware setting time can be used and the FLPGSCNT is not set.

FLPGSCNT[8:0]	Software erase time setting
Write action settling time = (PGSCNT*Tfclk), which meets the hardware requirements of >5 us	

27.4 FLASH operation method

27.4.1 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 is 1'b0, select sector erase mode, and set the value of ERMD 1 according to whether hardware verification is required;

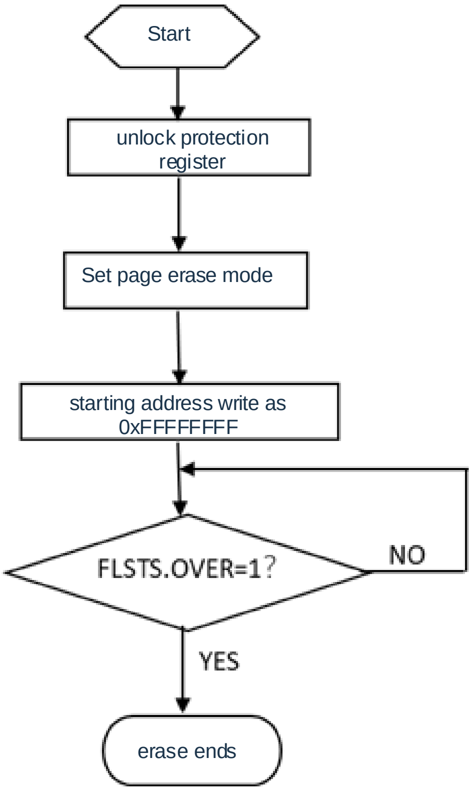
2) Set FLPROT to 0xF1 to unprotect FLOPMD. Then set FLOPMD1 to 0x55 and FLOPMD2 to 0xAA

3) Write arbitrary data to the first address of the erasure target sector. Example: *((unsigned long *)0x00000200) = 0xffffffff.

4) Software query status register FLSTS. OVF, OVF=1, indicates that the erase operation is complete.

5) If the hardware check after erasing is set (ERMD1=1), FLSTS.EV F can be determined by the software and whether the check is correct.

6) Before proceeding with the next operation, the software sets “1” to clear the FLSTS.


Start
unlock protection register
Set page erase mode
starting address write as 0xFFFFFFFF
FLSTS.OVER=1?
NO
YES
erase ends

27.4.2 Chip erase

Chip erase, and the erase time are implemented by hardware and can also be configured via FLCERCNT. The operation process is as follows:

1) Set FLERMD. ERMD0 is 1'b 1, 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 complete.
5) Before proceeding with the next operation, the software sets “1” to clear the FLSTS.

27.4.3 Programming (word program).

WORD programming, write time is implemented by hardware or can be configured via PROCNT. The operation process is as follows:

1) Set FLPROT to 0xF1 to unprotect FLOPMD. Then set FLOPMD1 to 0x AA and FLOPMD2 to 0x55
2) Writes the appropriate data to the destination address.
3) Software query status register FLSTS. OVF, OVF=1, indicates that the write operation is complete.
4) Before proceeding with the next operation, the software sets “1” to clear the FLSTS.

27.5 Flash read

The fastest finger frequency supported by the built-in FLASH of this device is 32 MHz. When the HCLK frequency exceeds 32MHz, the hardware inserts a 1 wait period when the CPU accesses the FLASH.

27.6 Cautions for FLASH operation

- FLASH memory has strict time requirements for the control signal of erasing and programming operation, and the timing of the control signal is not qualified will cause the erase operation and programming operation to fail. The setting of erasing and writing parameters can be implemented by hardware, or it can be modified by software by modifying parameter registers; When using internal high-speed OCO, MAINOSC/external input clock = 20M, it is recommended to use the hardware-set erasing parameters without setting parameter registers.

- If the erase and write operation is performed from within FLASH, the C PU stops taking the finger and the hardware automatically waits for the operation to complete before proceeding to the next command. If the operation is performed from the RAM, the CPU does not stop pointing and can now proceed to the next command.

- When FLASH is in programmatic operation, if the CPU executes the command to go to deep sleep, the system will wait for the programming action to end before entering deep sleep.

Appendix Revision History

Version	Date	Revised content
V1.0	2021/8/2	Initial version
V1.1	2021/12/20	24.3.8: Modified the unique product identification address
V1.2	2022/05/19	20.4.2: Modified deep sleep mode release conditions and added some notes
V1.2.1	2023/03/08	2.3: Add 24 pins and 20 pins in Table 2-1
V1.2.2	Jun 2023	1) Correct the multiplier description in section 1.2 2) Chapter 20 removes "Deep Sleep mode with partial power failure". 3) Delete sections 4.3.11 and 4.3.12