32-Bit Microcontroller FM3
Peripheral Manual Communication Macro Part

Doc. No. 002-04843 Rev. *A

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone (USA): 800.858.1810
Phone (Intntl): 408.943.2600
http://www.cypress.com

How to Use This Manual

Finding a Function

The following methods can be used to search for the explanation of a desired function in this manual:

Search from the table of the contents

The table of the contents lists the manual contents in the order of description.

Search from the register

The address where each register is located is not described in the text. To verify the address of a register, see "A. Register Map" in "Appendixes".

About the Chapters

Basically, this manual explains Timer Part..

Terminology

This manual uses the following terminology.

Term	Explanation
Word	Indicates access in units of 32 bits.
Half word	Indicates access in units of 16 bits.
Byte	Indicates access in units of 8 bits.

Notations

The notations in bit configuration of the register explanation of this manual are written as follows.

bit:	bit number
Field:	bit field name
Attribute:	Attributes for read and write of each bit
R:	Read only
W:	Write only
R/W:	Readable/Writable
-:	Undefined
Initial value:	Initial value of the register after reset
0:	Initial value is 0
1:	Initial value is 1
X:	Initial value is undefined

The multiple bits are written as follows in this manual.

Example: bit7:0 indicates the bits from bit7 to bit0

The values such as for addresses are written as follows in this manual.

Hexadecimal number:	"0x" is attached in the beginning of a value as a prefix (example: 0xFFFF)
Binary number:	"0b" is attached in the beginning of a value as a prefix (example: 0b1111)
Decimal number:	Written using numbers only (example: 1000)

The target products in this manual

In this manual, the products are classified into the following groups and are described as follows.
For the descriptions such as "TYPE0", see the relevant items of the target product in the list below.

Table 1 TYPE0 Product list

Description in this manual	Flash memory size			
	512 Kbytes	**384 Kbytes**	**256 Kbytes**	**128 Kbytes**
TYPE0	MB9BF506N MB9BF506R MB9BF506NA MB9BF506RA MB9BF506NB MB9BF506RB	MB9BF505N MB9BF505R MB9BF505NA MB9BF505RA MB9BF505NB MB9BF505RB	MB9BF504N MB9BF504R MB9BF504NA MB9BF504RA MB9BF504NB MB9BF504RB	-
	MB9BF406N MB9BF406R MB9BF406NA MB9BF406RA	MB9BF405N MB9BF405R MB9BF405NA MB9BF405RA	MB9BF404N MB9BF404R MB9BF404NA MB9BF404RA	-
	MB9BF306N MB9BF306R MB9BF306NA MB9BF306RA MB9BF306NB MB9BF306RB	MB9BF305N MB9BF305R MB9BF305NA MB9BF305RA MB9BF305NB MB9BF305RB	MB9BF304N MB9BF304R MB9BF304NA MB9BF304RA MB9BF304NB MB9BF304RB	-
	MB9BF106N MB9BF106R MB9BF106NA MB9BF106RA	MB9BF105N MB9BF105R MB9BF105NA MB9BF105RA	MB9BF104N MB9BF104R MB9BF104NA MB9BF104RA	MB9BF102N MB9BF102R MB9BF102NA MB9BF102RA
	-	MB9AF105N MB9AF105R MB9AF105NA MB9AF105RA	MB9AF104N MB9AF104R MB9AF104NA MB9AF104RA	MB9AF102N MB9AF102R MB9AF102NA MB9AF102RA

Table 2 TYPE1 Product list

Description in this manual	Flash memory size				
	512 Kbytes	**384 Kbytes**	**256 Kbytes**	**128 Kbytes**	**64 Kbytes**
TYPE1	MB9AF316M MB9AF316N MB9AF316MA MB9AF316NA	MB9AF315M MB9AF315N MB9AF315MA MB9AF315NA	MB9AF314L MB9AF314M MB9AF314N MB9AF314L MB9AF314M MB9AF314N	MB9AF312L MB9AF312M MB9AF312N MB9AF312LA MB9AF312MA MB9AF312NA	MB9AF311L MB9AF311M MB9AF311N MB9AF311LA MB9AF311MA MB9AF311NA
	MB9AF116M MB9AF116N MB9AF116MA MB9AF116NA	MB9AF115M MB9AF115N MB9AF115MA MB9AF115NA	MB9AF114L MB9AF114M MB9AF114N MB9AF114LA MB9AF114MA MB9AF114NA	MB9AF112L MB9AF112M MB9AF112N MB9AF112LA MB9AF112MA MB9AF112NA	MB9AF111L MB9AF111M MB9AF111N MB9AF111LA MB9AF111MA MB9AF111NA

Table 3 TYPE2 Product list

Description in this manual	Flash memory size		
	1 Mbytes	768 Kbytes	512 Kbytes
TYPE2	MB9BFD18S MB9BFD18T	MB9BFD17S MB9BFD17T	MB9BFD16S MB9BFD16T
	MB9BF618S MB9BF618T	MB9BF617S MB9BF617T	MB9BF616S MB9BF616T
	MB9BF518S MB9BF518T	MB9BF517S MB9BF517T	MB9BF516S MB9BF516T
	MB9BF418S MB9BF418T	MB9BF417S MB9BF417T	MB9BF416S MB9BF416T
	MB9BF318S MB9BF318T	MB9BF317S MB9BF317T	MB9BF316S MB9BF316T
	MB9BF218S MB9BF218T	MB9BF217S MB9BF217T	MB9BF216S MB9BF216T
	MB9BF118S MB9BF118T	MB9BF117S MB9BF117T	MB9BF116S MB9BF116T

Table 4 TYPE3 Product list

Description in this manual	Flash memory size	
	128 Kbytes	64 Kbytes
TYPE3	MB9AF132K MB9AF132L	MB9AF131K MB9AF131L
	MB9AF132KA MB9AF132LA	MB9AF131KA MB9AF131LA
	MB9AF132KB MB9AF132LB	MB9AF131KB MB9AF131LB

Table 5 TYPE4 Product list

Description in this manual	Flash memory size			
	512 Kbytes	384 Kbytes	256 Kbytes	128 Kbytes
TYPE4	MB9BF516N MB9BF516R	MB9BF515N MB9BF515R	MB9BF514N MB9BF514R	MB9BF512N MB9BF512R
	MB9BF416N MB9BF416R	MB9BF415N MB9BF415R	MB9BF414N MB9BF414R	MB9BF412N MB9BF412R
	MB9BF316N MB9BF316R	MB9BF315N MB9BF315R	MB9BF314N MB9BF314R	MB9BF312N MB9BF312R
	MB9BF116N MB9BF116R	MB9BF115N MB9BF115R	MB9BF114N MB9BF114R	MB9BF112N MB9BF112R

Table 6 TYPE5 Product list

Description in this manual	Flash memory size	
	128 Kbytes	64 Kbytes
TYPE5	MB9AF312K	MB9AF311K
	MB9AF112K	MB9AF111K

●

Table 7 TYPE6 product list

Description in this manual	Flash memory size		
	256 Kbytes	128 Kbytes	64 Kbytes
TYPE6	MB9AFB44L MB9AFB44M MB9AFB44N MB9AFB44LA MB9AFB44MA MB9AFB44NA MB9AFB44LB MB9AFB44MB MB9AFB44NB	MB9AFB42L MB9AFB42M MB9AFB42N MB9AFB42LA MB9AFB42MA MB9AFB42NA MB9AFB42LB MB9AFB42MB MB9AFB42NB	MB9AFB41L MB9AFB41M MB9AFB41N MB9AFB41LA MB9AFB41MA MB9AFB41NA MB9AFB41LB MB9AFB41MB MB9AFB41NB
	MB9AFA44L MB9AFA44M MB9AFA44N MB9AFA44LA MB9AFA44MA MB9AFA44NA MB9AFA44LB MB9AFA44MB MB9AFA44NB	MB9AFA42L MB9AFA42M MB9AFA42N MB9AFA42LA MB9AFA42MA MB9AFA42NA MB9AFA42LB MB9AFA42MB MB9AFA42NB	MB9AFA41L MB9AFA41M MB9AFA41N MB9AFA41LA MB9AFA41MA MB9AFA41NA MB9AFA41LB MB9AFA41MB MB9AFA41NB
	MB9AF344L MB9AF344M MB9AF344N MB9AF344LA MB9AF344MA MB9AF344NA MB9AF344LB MB9AF344MB MB9AF344NB	MB9AF342L MB9AF342M MB9AF342N MB9AF342LA MB9AF342MA MB9AF342NA MB9AF342LB MB9AF342MB MB9AF342NB	MB9AF341L MB9AF341M MB9AF341N MB9AF341LA MB9AF341MA MB9AF341NA MB9AF341LB MB9AF341MB MB9AF341NB
	MB9AF144L MB9AF144M MB9AF144N MB9AF144LA MB9AF144MA MB9AF144NA MB9AF144LB MB9AF144MB MB9AF144NB	MB9AF142L MB9AF142M MB9AF142N MB9AF142LA MB9AF142MA MB9AF142NA MB9AF142LB MB9AF142MB MB9AF142NB	MB9AF141L MB9AF141M MB9AF141N MB9AF141LA MB9AF141MA MB9AF141NA MB9AF141LB MB9AF141MB MB9AF141NB

Table 8 TYPE7 product list

Description in this manual	Flash memory size	
	128 Kbytes	64 Kbytes
TYPE7	MB9AFA32L MB9AFA32M MB9AFA32N	MB9AFA31L MB9AFA31M MB9AFA31N
	MB9AF132M MB9AF132N	MB9AF131M MB9AF131N
	MB9AFAA2L MB9AFAA2M MB9AFAA2N	MB9AFAA1L MB9AFAA1M MB9AFAA1N
	MB9AF1A2M MB9AF1A2N	MB9AF1A1M MB9AF1A1N

Table 9 TYPE8 product list

Description in this manual	Flash memory size		
	512 Kbytes	384 Kbytes	256 Kbytes
TYPE8	MB9AF156M MB9AF156N MB9AF156R MB9AF156MA MB9AF156NA MB9AF156RA MB9AF156MB MB9AF156NB MB9AF156RB	MB9AF155M MB9AF155N MB9AF155R MB9AF155MA MB9AF155NA MB9AF155RA MB9AF155MB MB9AF155NB MB9AF155RB	MB9AF154M MB9AF154N MB9AF154R MB9AF154MA MB9AF154NA MB9AF154RA MB9AF154MB MB9AF154NB MB9AF154RB

Table 10 TYPE9 product list

Description in this manual	Flash memory size		
	256 Kbytes	128 Kbytes	64 Kbytes
TYPE9	MB9BF524K MB9BF524L MB9BF524M	MB9BF522K MB9BF522L MB9BF522M	MB9BF521K MB9BF521L MB9BF521M
	MB9BF324K MB9BF324L MB9BF324M	MB9BF322K MB9BF322L MB9BF322M	MB9BF321K MB9BF321L MB9BF321M
	MB9BF124K MB9BF124L MB9BF124M	MB9BF122K MB9BF122L MB9BF122M	MB9BF121K MB9BF121L MB9BF121M

Table 11 TYPE10 product list

Description in this manual	Flash memory size
	64 Kbytes
TYPE10	MB9BF121J

Table 12 TYPE11 product list

Description in this manual	Flash memory size
	64 Kbytes
TYPE11	MB9AF421K MB9AF421L
	MB9AF121K MB9AF121L

Table 13 TYPE12 product list

Description in this manual	Flash memory size	
	1.5 Mbytes	1 Mbytes
TYPE12	MB9BF529S MB9BF529T MB9BF529SA MB9BF529TA	MB9BF528S MB9BF528T MB9BF528SA MB9BF528TA
	MB9BF429S MB9BF429T MB9BF429SA MB9BF429TA	MB9BF428S MB9BF428T MB9BF428SA MB9BF428TA
	MB9BF329S MB9BF329T MB9BF329SA MB9BF329TA	MB9BF328S MB9BF328T MB9BF328SA MB9BF328TA
	MB9BF129S MB9BF129T MB9BF129SA MB9BF129TA	MB9BF128S MB9BF128T MB9BF128SA MB9BF128TA

Contents

CHAPTER 1-1: Multi-function Serial Interface

This chapter describes the overview of the multi-function serial interface.

CODE: 9BFMFS-E02.0

1. Overview of the Multi-function Serial Interface

This multi-function serial interface has the following characteristics.

■ Interface Mode

The following interface modes are selectable for the multi-function serial interface depending on the operation mode settings.

- UART0 (Asynchronous normal serial interface)
- UART1 (Asynchronous multi-processor serial interface)
- CSIO (Clock synchronous serial interface) (SPI can be supported)
- LIN(LIN bus interface)
- I^2C (I^2C bus interface)

<Note>

See explanations of “UART (Asynchronous normal serial interface), “CSIO (Clock synchronous serial interface)”, “LIN(LIN bus interface)”, and “I^2C (I^2C bus interface)” chapters for details about each interface.

■ Switching the Interface Mode

To communicate through each serial interface, the serial mode register (SMR) shown in Table 1-1 should be used to set the operation mode before starting the communication.

Table 1-1 Switching Interface Mode

MD2	MD1	MD0	Interface mode
0	0	0	UART0 (Asynchronous normal serial interface)
0	0	1	UART1 (Asynchronous multi-processor serial interface)
0	1	0	CSIO (Clock synchronization serial interface) (SPI can be supported)
0	1	1	LIN(LIN bus interface)
1	0	0	I^2C (I^2C bus interface)
Values other than the above			Setting is prohibited.

<Notes>

- Transmission and reception cannot be guaranteed when the operation mode is switched while one of the serial interfaces is still in use for transmission or reception operation.
- To switch the current operation mode, issue a programmable clear (SCR:UPCL=1) or disable the I^2C (ISMK:EN=0) , and switch the operation mode continuously. After the operation mode is set, set each register.
- The settings not listed in Table 1-1 are prohibited.

■ Transmission/Reception FIFO

This function has a 16 × 9 bits transmission FIFO and 16 × 9 bits reception FIFO. The FIFO steps should be converted to 16 × 9 bits when reading through this text.

■ LIN Sync field Detection: LSYN

If you are to use an ICU in the LIN bus interface mode, use the ICU of the multifunction timer.
For switching an input to an ICU, see the section for Extended Function Pin Setting Register in the chapter "I/O PORT" in "Peripheral Manual".

■ I²C Auxiliary Noise Filter

If the APB2 bus clock frequency exceeds 40 MHz when using the I²C bus interface, use an auxiliary noise filter. I²C standard input noise to a maximum of 50 ns is cut off.
For details, see the chapter "I²C Auxiliary Noise Filter".
The I²C auxiliary noise filter is built into only the product with the APB1 bus clock whose maximum frequency is 40 MHz or more.

<Notes>

- Since the maximum frequency of the APB1 bus clock varies depending on the product TYPE, refer to the internal operation clock frequency (F_{CP1}) of the "Data Sheet" of products used.
- When the I²C auxiliary noise filter is used, the calculation formula of the reload value that should be set to the Baud rate generator registers (BGR1, BGR0) is different. See chapter "I²C Auxiliary Noise Filter" for the calculation formula of reload value when using the I²C auxiliary noise filter.

■ Extended I²C Bus Control Register (EIBCR)

TYPE6 products and later equip the extended I²C bus control register (EIBCR). This register controls the following features. For details, see the chapter "I²C Interface (I²C Communication Control Interface)".

- Output control of SDA/SCL
- Continuity/non-continuity of I²C operation after a bus error occurs

---

<Note>

Because TYPE0 to TYPE5 products do not equip the EIBCR register, read the following explanation as EIBCR:BEC=0.

CHAPTER 1-2: UART (Asynchronous Serial Interface)

This chapter explains the UART (asynchronous serial interface) function supported in operation mode 0 and 1 of the multifunction serial interface.

CODE: 9BFUART-J02.0_FM15U-E05.6

1. Overview of UART (Asynchronous Serial Interface)

UART (asynchronous serial interface) is a general-purpose serial data communications interface for asynchronous communications (start/stop synchronization) with external devices. It supports a bi-directional communications function (normal mode) and a master/slave type communications function (multi-processor mode: both master and slave modes supported). It also has transmit /received FIFO installed.

■ Functions of UART (Asynchronous Serial Interface)

		Function
1	Data	· Full duplex double buffer (when FIFO is not used) · Transmit /received FIFO (size: max 128 × 9 bits each)[*1] (when FIFO is used)
2	Serial input	Run oversampling three times with the bus clock and determine the value of received data based on the majority sampling value.
3	Transfer system	Asynchronous
4	Baud rate	· A dedicated baud rate generator (constructed with a 15-bit reload counter) · The external clock input can be adjusted with the reload counter.
5	Data length	· 5 to 9 bits (in normal mode)/7 bits or 8 bits (in multiprocessor mode)
6	Signaling system	NRZ (Non Return to Zero), inverted NRZ
7	Start bit detection	· In synch with the falling edge of the start bit (in the NRZ system) · In synch with the rising edge of the start bit (in the inverted NRZ system)
8	Received error detection	· Framing error · Overrun error · Parity error[*2]
9	Hardware flow control	CTS/RTS-based automatic transmit /received control[*3]
10	Interrupt request	· Received interrupt (upon reception completed, framing error, overrun error or parity error[*2]) · Transmit interrupts (transmit data empty, transmit bus idle) · Transmit FIFO interrupt (when transmit FIFO is empty) · DMA(Transmit /Received) transferring support function is available.
11	Master/slave communications functions (in multiprocessor mode)	One (master)-to-n (slaves) communication is enabled. (Both master and slave systems are supported.)
12	FIFO options	· Transmit /received FIFO installed (maximum capacity: 128 × 9 bits for transmit FIFO, 128 bytes × 9 bits for received FIFO) [*1] · Transmit FIFO or received FIFO can be selected. · Transmit data can be resent. · Received FIFO interrupt timing can be changed via software. · FIFO resetting is supported independently.

*1: The FIFO capacity size varies depending on the product type.
*2: Parity errors are only generated in normal mode.
*3: The channel number, which the hardware flow control input/output (RTS/CTS) can be used, is dependent on the product type. See Data Sheet of the product used.

2. UART Interrupt

UART generates transmit or received interrupts. These interrupt requests can be generated if:
- Received data is set in the Received Data Register (RDR) or a data received error occurs.
- Transmit data is transferred from the Transmit Data Register (TDR) to the transmit shift register and the data transmission is started.
- The transmit bus is idle (No data transmission occurs).
- Transmit FIFO data is requested.

■ UART Interrupt

Table 2-1 shows the relationships between the UART interrupt control bits and the interrupt factors.

Table 2-1 UART interrupt control bits and interrupt factors

Interrupt type	Interrupt request flag bit	Flag register	Operation mode		Interrupt factor	Interrupt factor enable bit	Operation to clear interrupt request flag
			0	1			
Received	RDRF	SSR	○	○	A single-byte received	SCR:RIE	Reading from the received data register (RDR)
					Received of a data volume matching the value set for FBYTE.		Reading from the Received Data Register (RDR) until received FIFO is emptied
					While the FRIIE bit is "1" and the received FIFO contains valid data, a received idle state continues for 8 bits or longer period.		
	ORE	SSR	○	○	Overrun error		Setting the received error flag clear bit (SSR:REC) to "1"
	FRE	SSR	○	○	Framing error		
	PE	SSR	○	x	Parity error		
Transmit	TDRE	SSR	○	○	The Transmit Data Register is empty	SCR:TIE	Writing to the Transmit Data Register (TDR) or setting the transmit FIFO operation enable bit to "1" when the transmit FIFO operation enable bit is set to "0" and valid data are present in transmit FIFO (re-transmitting data) *1
	TBI	SSR	○	○	No data transmission	SCR:TBIE	Writing to the Transmit Data Register (TDR) or setting the transmit FIFO operation enable bit to "1" when the transmit FIFO operation enable bit is set to "0" and valid data are present in transmit FIFO (re-transmitting data) *1
	FDRQ	FCR1	○	○	Transmit FIFO is empty.	FCR1:FTIE	The FIFO transmit data request bit (FCR1:FDRQ) is set to "0" or transmit FIFO is full.

*1: Set the TIE bit to "1" only after the TDRE bit has been set to "0".

2.1. Received interrupt and flag set timing

Data reception can be interrupted by a Received Completion (SSR:RDRF=1) or a Received Error Occurrence (SSR:PE,ORE,FRE=1).

■ Received interrupt and flag set timing

Upon detection of the first stop bit, received data are stored in the Received Data Register (RDR). When the data received is completed (SSR:RDRF=1) or when a data received error occurs (SSR:PE, ORE, FRE=1), each flag is set. If received interrupts are enabled (SSR:RIE=1) then, a received interrupt occurs.

<Note>

If a received error occurs, data in the Received Data Register (RDR) becomes invalid.

Figure 2-1 RDRF (Received Data Register Full) flag bit set timing

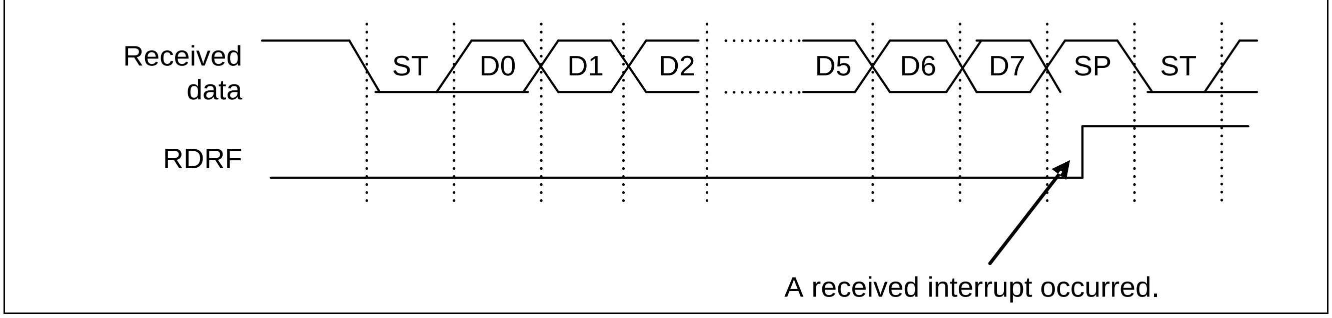
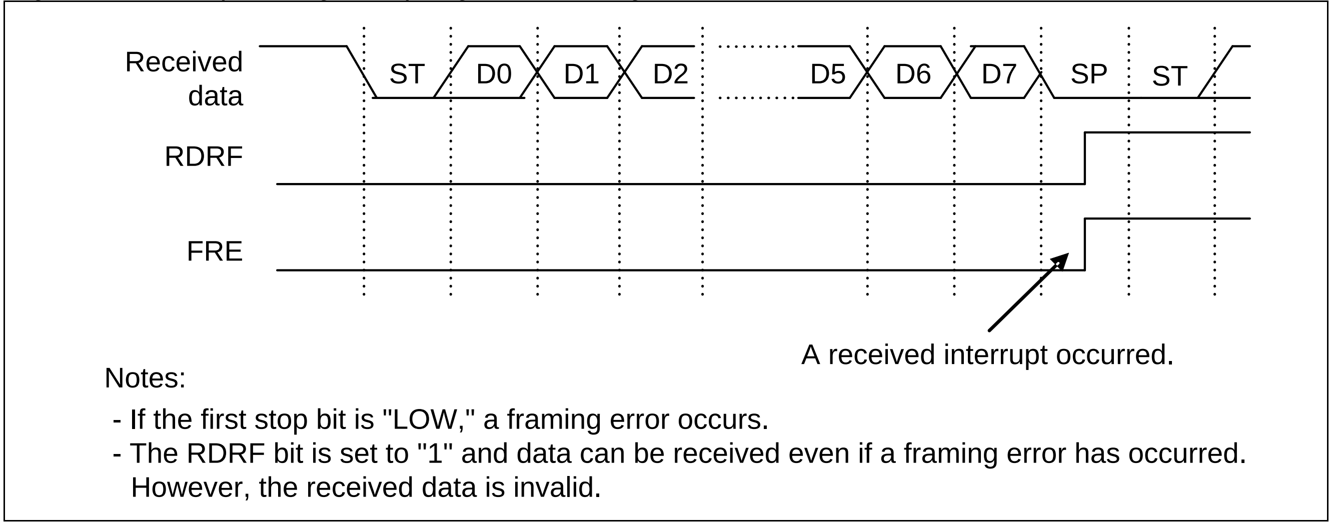


Figure 2-2 FRE (Framing Error) flag bit set timing

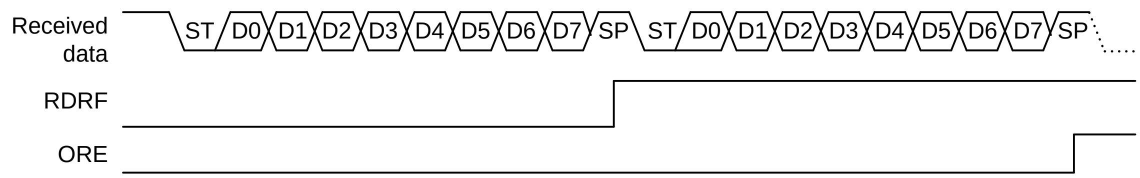


<Note>

During reception, if the following is detected at the same time as the stop bit sampling point or before the 1 to 2 bus clocks, the relevant edge becomes invalid, which may disable normal received of the next data. To output frames continuously, adequate intervals are required between frames.

- The falling edge of serial data (When ESCR:INV=0)
- The rising edge of serial data (When ESCR:INV=1)

Figure 2-3 ORE (Overrun Error) flag bit set timing



Note:

If the next data is transferred before the received data is read (RDRF=1), an overrun error occurs.

2.2. Interrupt and flag set timing when received FIFO is used

If the received FIFO is used, an interrupt occurs when the FBYTE data (preset for the FBYTE register) is received.

■ Interrupt and flag set timing when received FIFO is used

If the received FIFO is used, an interrupt occurs depending on the value set for the FBYTE register.

- When full FBYTE data is received, the received data full flag (SSR:RDRF) of the Serial Status register is set to "1". If received interrupts are enabled (SCR:RIE) during this time, a received interrupt occurs.
- If both of the following conditions are satisfied and if the received idle state continues for more than 8 baud rate clocks, the receive data full flag (SSR:RDRF) is set to "1".
  - The received FIFO idle detection enable bit (FCR:FRIIE) is "1".
  - The number of data sets stored in the received FIFO does not reach the transfer count.

  If the RDR data is read during counting of 8 clocks, this counter is reset to "0", and counting for 8 clocks is restarted. If received FIFO is disabled, this counter is reset to zero (0).If data remains in the received FIFO and if received FIFO is enabled, the data counting is restarted.
- When data is read from the Received Data Register (RDR) until received FIFO is emptied, the received data full flag (SSR:RDRF) is cleared.
- If the valid received data amount is the same as the FIFO capacity and if the next data is received, an overrun error (SSR:ORE=1) occurs.

Figure 2-4 Received interrupt timing when Received FIFO is used

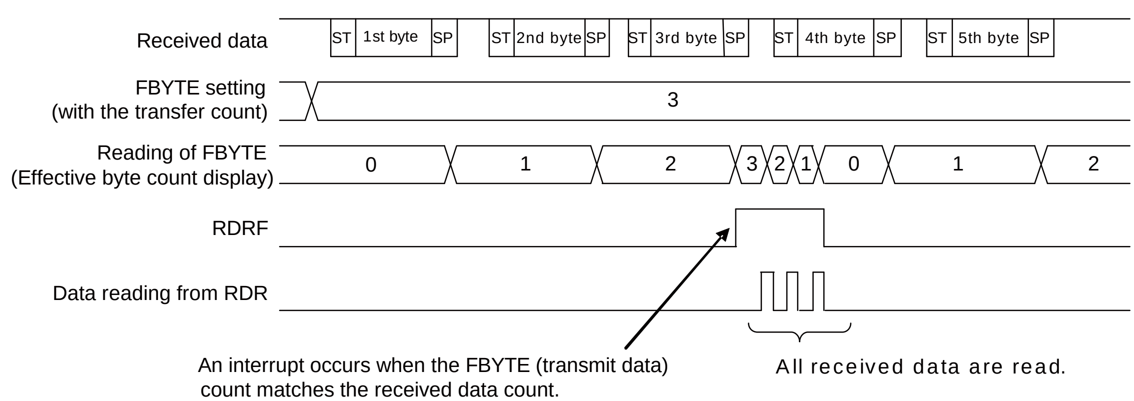
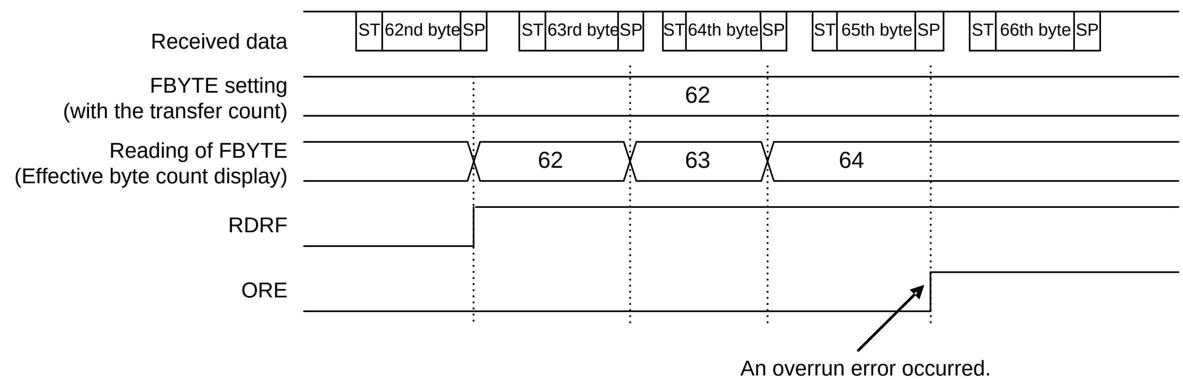


Figure 2-5 ORE (Overrun Error) flag bit set timing



Note:
If the next data set is received when the FBYTE reading is indicating the FIFO capacity, an overrun error occurs.
This figure shows a case where a 64-byte FIFO capacity is applied.

2.3. Transmit interrupt and flag set timing

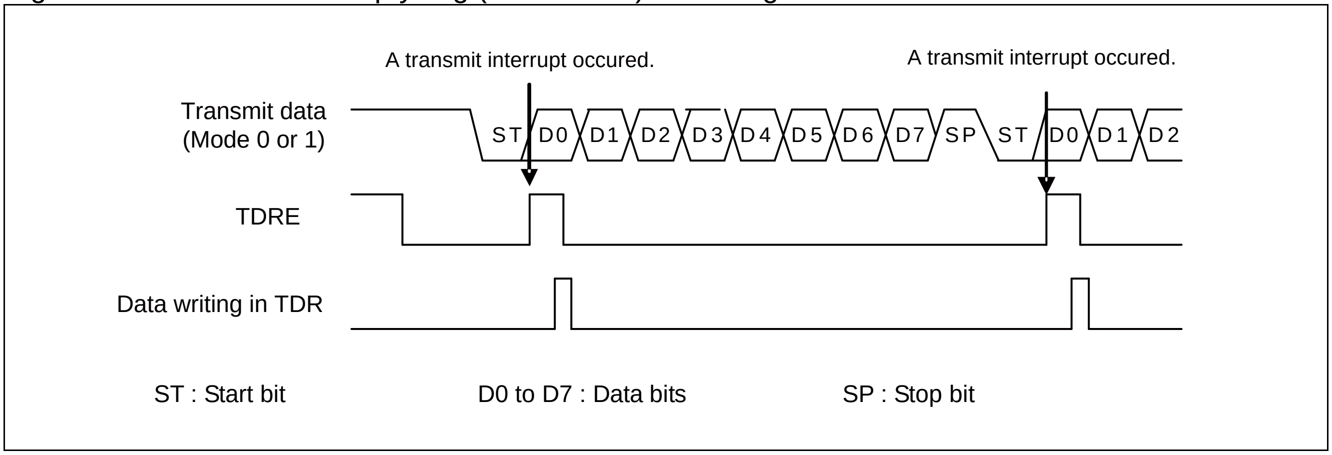
A transmit interrupt occurs when transmit data is transferred from the Transmit Data Register (TDR) to the transmit shift register (SSR:TDRE = 1) and transmission starts and when no transmission is performed (SSR:TBI = 1).

■ Transmit interrupt and flag set timing

● Transmit data empty flag (SSR:TDRE) set timing

After data has been transferred from the Transmit Data Register (TDR) to the transmit shift register, the next data can be written in the TDR (SSR:TDRE = 1). If transmit interrupts are enabled (SCR:TIE = 1) during this time, a transmit interrupt occurs. As the SSR:TDRE bit is read only, the SSR:TDRE bit is cleared to "0" when data is written to the Transmit Data Register (TDR).

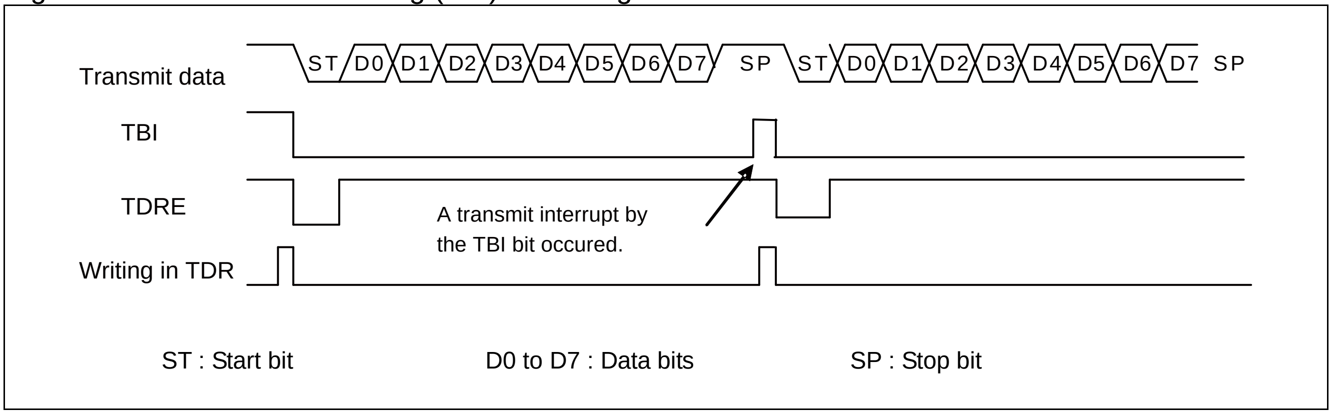
Figure 2-6 Transmit data empty flag (SSR:TDRE) set timing



● Transmit bus idle flag (SSR:TBI) set timing

If the Transmit Data Register is empty (SSR:TDRE=1) and no data is transmitted, the SSR:TBI bit is set to "1". If transmit bus idle interrupts are enabled (SCR:TBIE = 1) during this time, a transmit interrupt occurs. When transmit data is written to the Transmit Data Register (TDR), both the SSR:TBI bit and the transmit interrupt request are cleared.

Figure 2-7 Transmit bus idle flag (TBI) set timing



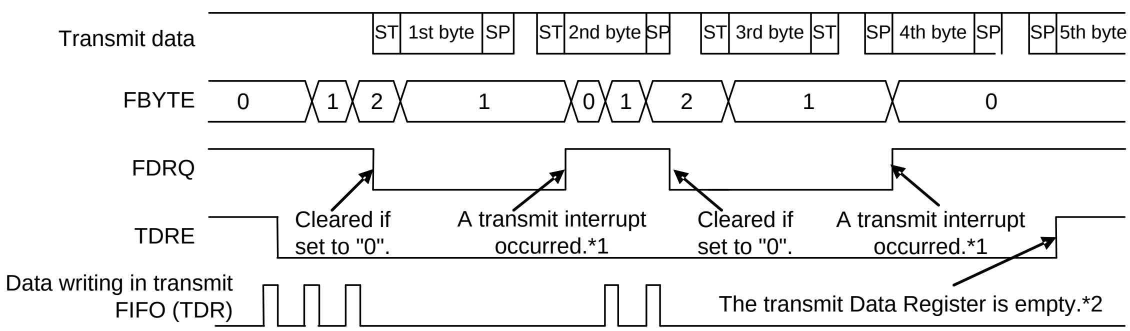
2.4. Interrupt and flag set timing when transmit FIFO is used

When the transmit FIFO is used, an interrupt occurs if the FIFO contains no data.

■ Transmit interrupt and flag set timing when transmit FIFO is used

- If the Transmit FIFO contains no data, the FIFO transmit data request bit (FCR1:FDRQ) is set to "1".
  If FIFO transmit interrupts are enabled (FCR1:FTIE=1), a transmit interrupt occurs.
- If a transmit interrupt has occurred and you have written the required data in transmit FIFO, clear the interrupt request by setting the FIFO transmit data request bit (FCR1:FDRQ) to "0".
- The FIFO transmit data request bit (FCR1:FDRQ) is set to "0" when transmit FIFO becomes full.
- To check to see if transmit FIFO contains any data, read from the FIFO Byte Register (FBYTE).
  If FBYTE=0x00, no data exists in the transmit FIFO.

Figure 2-8 Transmit interrupt timing when transmit FIFO is used



*1) The FDRQ bit is set to "1" as Transmit FIFO is empty.
*2) The TDRE bit is set to “1” as the Transmit Shift Register and the Transmit Buffer Register contain no data.

3. UART Operation

UART operates in bi-directional serial asynchronous communications in mode 0 and master/slave multiprocessor communications in mode 1.

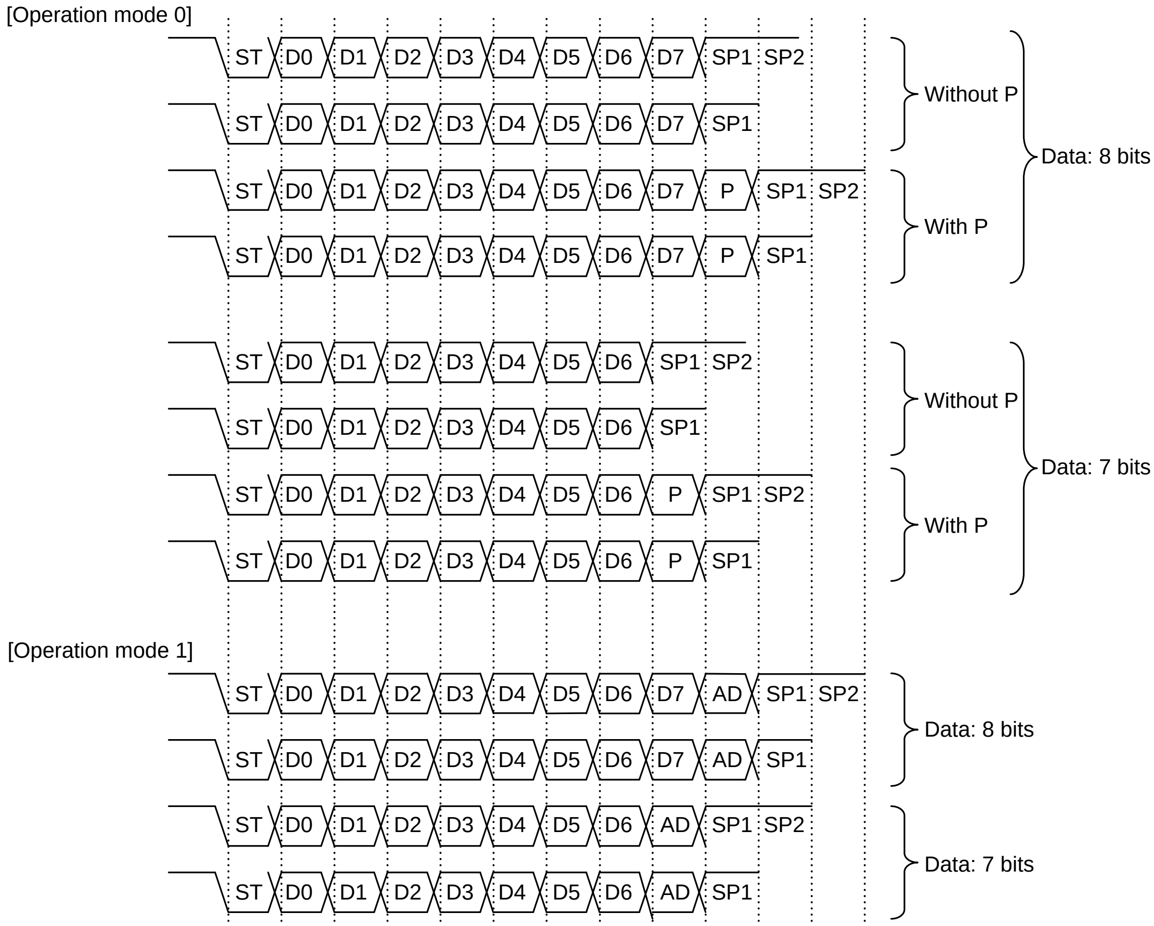
■ UART operation

● Transmit/received data format

- Transmit/received data always starts with a start bit, followed by transmit/received of data with the specified data bit length, and ends with at least one-bit long stop bit.
- The BDS bit of the Serial Mode Register (SMR) determines the data transmission direction (LSB first or MSB first). If parity is used, the parity bit is always placed between the last data bit and the first stop bit.
- In operation mode 0 (normal mode), selection is possible to use or not to use parity.
- In operation mode 1 (multiprocessor mode), no parity is added, and instead, the AD bit is added.

Figure 3-1 shows the transmit/received data formats for operation mode 0 and 1.

Figure 3-1 Example transmit/received data format (operation mode 0/1)



ST : Start bit
SP : Stop bit
P : Parity bit
AD : Address bit
D : Data bit

<Notes>

- The above figure shows formats when the data length is set to 7 or 8 bits. (In operation mode 0, the data length can be set between 5 and 9 bits.)
- If the BDS bit of the Serial Mode Register (SMR) is set to "1" (MSB first), the bits are processed from D7, and then D6, D5, ... D1, and D0 (P), in that order.
- If the data length is set to X bits, the lower X bit of the Transmit/Received Data Register (TDR/RDR) is enabled.

● Data transmission

- If the transmit data empty flag bit (TDRE) of the Serial Status Register (SSR) is "1", the transmit data can be written in the Transmit Data Register (TDR). (When transmit FIFO is enabled, transmit data can be written even if TDRE=0.)
- If transmit data is written in the Transmit Data Register (TDR), the transmit data empty flag bit (SSR:TDRE) is set to "0".
- Setting the transmission enable bit of the serial control register (SCR:TXE) to "1" causes transmit data to be loaded to the transmit shift register, followed by sequential transmission starting with the start bit.
- When transmission starts, the transmit data empty flag bit (SSR:TDRE) is set to "1" again. If transmit interrupts are then enabled (SCR:TIE=1), a transmit interrupt is generated. In the interrupt processing, the next transmit data set can be written in the Transmit Data Register,

---

<Notes>

- As the transmit data empty flag bit (SSR:TDRE) is initially set to "1", a transmit interrupt occurs as soon as transmit interrupts are enabled (SCR:TIE).
- As the FIFO transmit data request bit (FCR1:FDRQ) is initially set to "1", a transmit interrupt occurs as soon as FIFO transmit interrupts are enabled (FCR1:FTIE=1).

● Data reception

- When reception is enabled (SCR:RXE=1), the interface performs reception.
- Upon detection of the start bit, one-frame data reception takes place according to the data format set in the extended communications control register (ESCR:PEN, P, L2, L1, L0) and serial mode register (SMR:BDS). A start bit is detected when falling (ESCR:INV=0) is detected after passing the noise filter (with the majority value applied after sampling serial data input three times with the bus clock) or if rising (ESCR:INV=1) is detected and "LOW" is detected for the data passing the sampling point.
- When one-frame reception is completed, the received data full flag bit (SSR:RDRF) is set to "1". If received interrupts are then enabled (SCR:RIE=1), a received interrupt is generated.
- To read received data, perform reading of the received data after one-frame data received is completed and check the state of the error flag of the Serial Status Register (SSR). Handle the received error if it is occurring.
- Reading of the received data causes the received data full flag bit (SSR:RDRF) to be cleared to "0".
- If received FIFO is enabled, the received data full flag bit (SSR:RDRF) is set to "1" when the number of received frames has reached the value set for received FBYTE.
- If all of the following conditions are satisfied and if the received idle state continues for more than 8 baud rate clocks, the interrupt flag (RDRF) is set to "1".
    - The received FIFO idle detection enable bit (FRIIE) is "1".
    - The number of data sets stored in the received FIFO does not reach the transfer count.

  If the RDR data is read during counting of 8 clocks, this counter is reset to "0", and counting for 8 clocks is restarted. If received FIFO is disabled, this counter is reset to zero (0). If data remains in the received FIFO and if received FIFO is enabled, the data counting is restarted.
- If received FIFO is enabled, received FIFO does not store data in which an error has occurred when the error flag of the Serial Status Register (SSR) is set to "1". Also note that the received data full flag bit (SSR:RDRF) is not set to "1". (However, the RDRF flag is set to "1" in an overrun error.) What the received FBYTE indicates is the number of data sets received normally before the error occurred. Unless the error flag of the Serial Status Register (SSR) is cleared to "0", received FIFO is not enabled.
- If received FIFO is enabled, the received data full flag bit (SSR:RDRF) is cleared to "0" when all data in received FIFO is out.

<Notes>

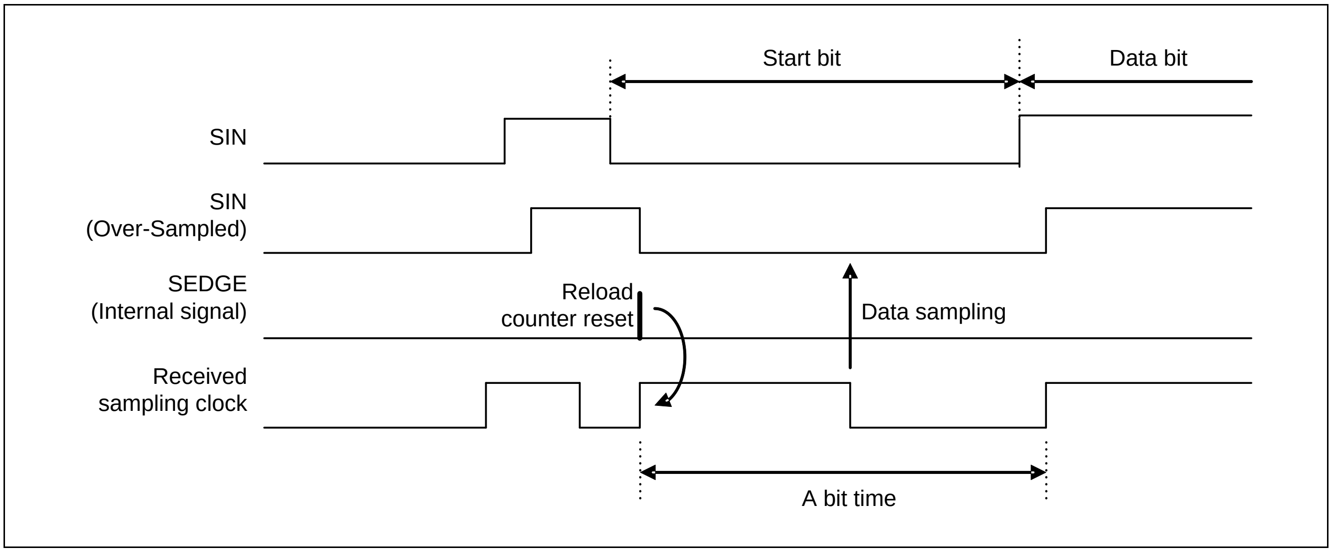
- Data in the Received Data Register (RDR) becomes valid when the received data register full flag bit (SSR:RDRF) is set to "1" and no received error occurs (SSR:PE, ORE, FRE=0).
- Although a noise filter is built in (with the majority value applied after sampling serial data input three times with the bus clock), wrong data may be received if any noise passes through the filter. As a countermeasures, you can design the board so as not to allow noise to pass through this filter or perform communications so that noise that has passed may not cause any problem (by adding check sum of data at the end and resending the data if any error occurs, for example).
- During reception, if the following is detected at the same time as the stop bit sampling point or before the 1 to 2 bus clocks, the relevant edge becomes invalid, which may disable normal reception of the next data. To output frames continuously, adequate intervals are required between frames.
    - The falling edge of serial data (When ESCR:INV=0)
    - The rising edge of serial data (When ESCR:INV=1)

● Clock selection

- You can use either an internal or external clock.
- To use the external clock, set SMR:EXT to "1". IN this case, the external clock is subject to frequency division by the baud rate generator. The external clock is input from SCK.

● Start bit detection

- In asynchronous mode, the start bit is recognized based on detection of the falling edge of the SIN signal.
  For that reason, reception is not started unless the falling edge of the SIN signal is input even if reception is enabled (SCR:RXE=1).
- Upon detection of the start bit's falling edge, the received reload counter of the baud rate generator is reset and reloaded to start countdown. Thus, sampling always takes place in the middle of data.



● Stop bit

- You can select the bit length to be between one and four.
- The received data full flag bit (SSR:RDRF) is set to "1" upon detection of the first stop bit.

● Error detection

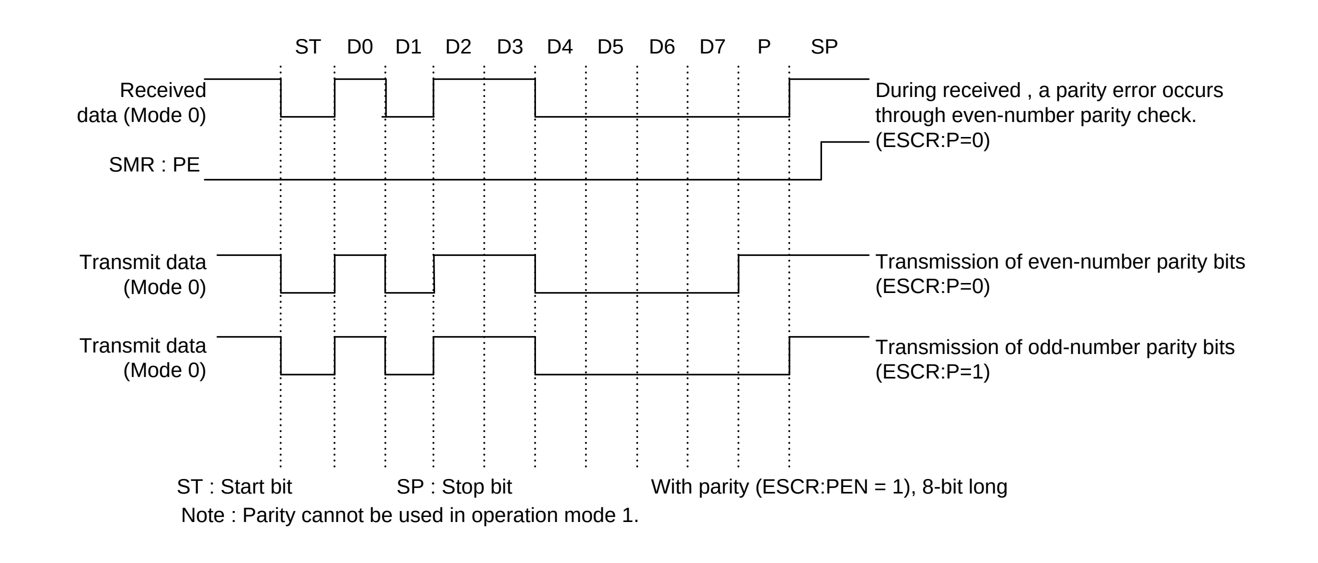
- In operation mode 0, parity, overrun and framing errors can be detected.
- In operation mode 1, overrun and framing errors can be detected but parity errors cannot be detected.

● Parity bit

- The parity bit can only be added in operation mode 0. The parity enable bit (ESCR:PEN) can be used to specify use or non-use of parity and the parity selection bit (ESCR:P) to set even-number parity or odd-number parity.
- Parity cannot be used in operation mode 1.

Figure 3-2 shows transmit/received data when parity is enabled.

Figure 3-2 Operation when parity is enabled

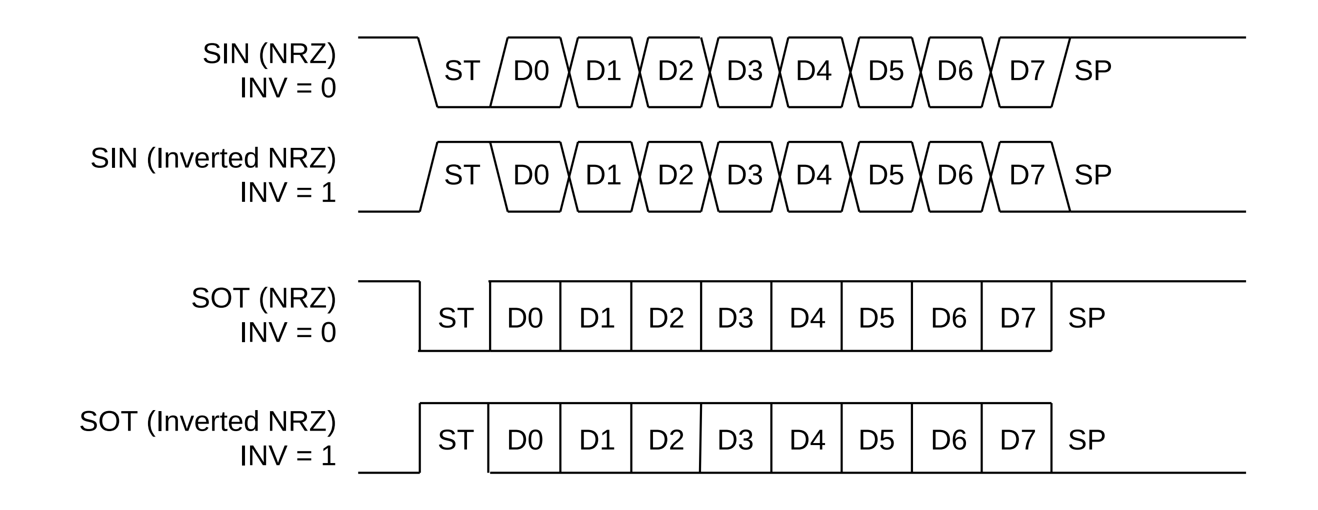


● Data signaling system

By setting up the INV bit of the extended communications control register, you can select either the NRZ (Non Return to Zero) signaling system (ESCR:INV=0) or inverted NRZ signaling system (ESCR:INV=1).

Figure 3-3 shows the NRZ and inverted NRZ signaling systems.

Figure 3-3 NRZ (Non Return to Zero) signaling system and inverted NRZ signaling system



● Data transfer system

As for the data bit transfer method, either LSB first or MSB first can be selected.

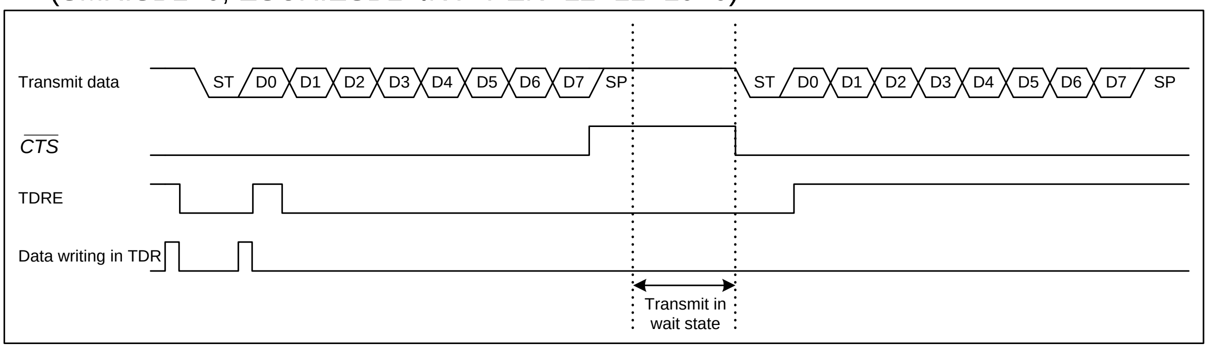
● Hardware flow control

When flow control is enabled (ESCR:FLWEN=1), UART performs hardware flow control.

· During data transmission

If $\overline{CTS}$ is "HIGH" after data is transmitted, the next data is not transmitted even if the transmit buffer contains data (TDRE=0) and the process waits until $\overline{CTS}$ is set to "LOW". To have transmission wait, input "HIGH" in $\overline{CTS}$ before the stop bit transmission is completed. Transmission continues up to the stop bit even if "HIGH" is input in $\overline{CTS}$ during transmission.

Figure 3-4 Hardware flow control during data transmission
(SMR:SBL=0, ESCR:ESBL=INV=PEN=L2=L1=L0=0)

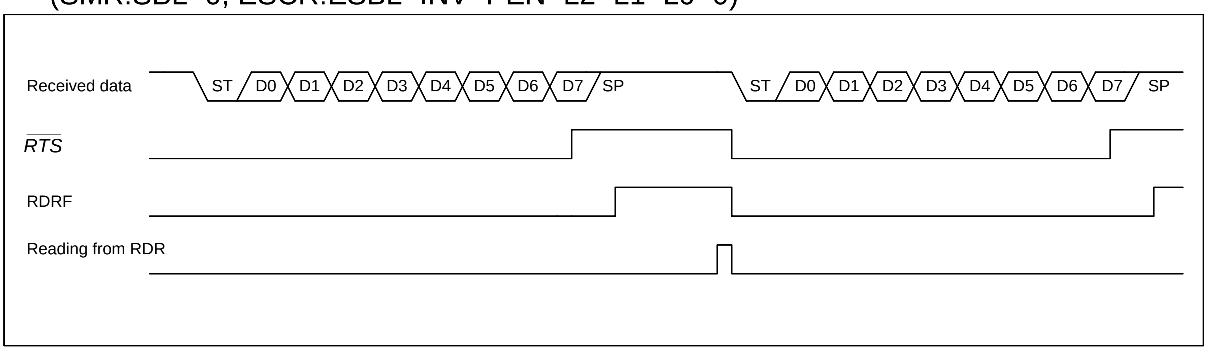


· During data reception

· If FIFO is not used

Upon reception of data one bit before the stop bit, "HIGH" is output to $\overline{RTS}$. After received data is read, "LOW" is output to $\overline{RTS}$.

Figure 3-5 Hardware flow control during data reception (with FIFO is unused.)
(SMR:SBL=0, ESCR:ESBL=INV=PEN=L2=L1=L0=0)

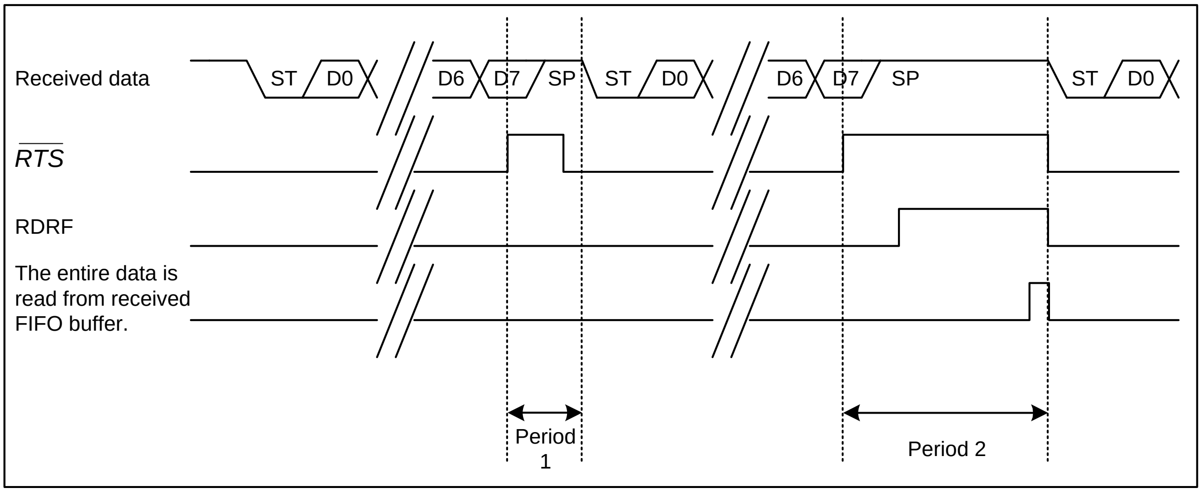


- If FIFO is used

  If SSR:RDRF is not set (the specified number of data sets are not received in received FIFO), $\overline{RTS}$ outputs "HIGH" upon reception of data one bit before the stop bit, but $\overline{RTS}$ outputs "LOW" upon detection of the stop bit. (For period 1)

  If SSR:RDRF is set (the specified number of data sets are received in received FIFO), $\overline{RTS}$ outputs "HIGH" upon reception of data one bit before the stop bit. $\overline{RTS}$ outputs "LOW" after all data is read from received FIFO. (For period 2)

Figure 3-6 Hardware flow control during data reception (with FIFO used)
(SMR:SBL=0, ESCR:ESBL=INV=PEN=L2=L1=L0=0)



<Notes>

- When reception operation is disabled (RXE=0), the $\overline{RTS}$ signal is fixed to "LOW".
- If both conditions below are satisfied when received FIFO is used and if the received idle state continues for more than 8 baud rate clocks, RDRF is set to "1" but "LOW" is maintained for the $\overline{RTS}$ signal.
  - The received FIFO idle detection enable bit (FCR1:FRIIE) is "1".
  - The preset data amount is not received and some data remains in received FIFO.
- Performing programmable resetting (SCR:UPCL=1) clears the $\overline{RTS}$ signal to "LOW".

---

4. Dedicated Baud Rate Generator

As for the UART transmit/received clock source, either of the following can be selected.
- Dedicated baud rate generator (reload counter)
- An external clock input to the baud rate generator (reload counter)

■ Selecting the UART baud rate

Select one of the following two baud rates.

● Baud rate obtained by dividing an internal clock using the dedicated baud rate generator (reload counter)

This generator provides two internal reload counters, which support transmitting and receiving serial clocks respectively. To select the baud rate, specify the 15-bit reload value using Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0).

Each reload counter divides an internal clock by the set value.

To set the clock source, select an internal clock (BGR1:EXT=0).

● Baud rate obtained by dividing an external clock using the dedicated baud rate generator (reload counter)

Use an external clock for the clock source of the reload counter. The external clock is input from SCK.

To select the baud rate, specify the 15-bit reload value using Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0).

Each reload counter divides an external clock by the set value.

To set the clock source, select use of an external clock and the baud rate generator clock (BGR1:EXT=1).

This mode is designed for cases where an oscillator with a divided non-standard frequency is used.

<Notes>

- Set the external clock (BGR1:EXT=1) while the reload counter is suspended (BGR1/0=15' h00).
- If an external clock is selected (BGR1:EXT=1), its HIGH and LOW signals must have a width at least of two bus clocks.

4.1. Baud rate settings

The following explains how to set the baud rate, and also a result of serial clock frequency calculation.

■ Calculating the baud rate

Two 15-bit reload counters are set using the Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0).
The baud rate is obtained in the following formulas.

(1) Reload value

$V = \phi / b - 1$

V : Reload value b : Baud rate ϕ: Bus clock frequency or external clock frequency

(2) Calculation example

To set the 16 MHz bus clock, use the internal clock, and set the 19200 bps baud rate, set the reload value as follows:

Reload value:

V = (16 × 1000000) / 19200 – 1 = 832

Therefore, the baud rate is:

b = (16 × 1000000) / (832 + 1) = 19208 bps

(3) Baud rate error

The baud rate error can be calculated by the following equation.

Error (%) = (Calculated value – Target value) / Target value × 100

Example: To set the 20 MHz bus clock and 153600 bps target baud rate:

Reload value =:(20 × 1000000) /153600-1 = 129

Buad rate (Calculated value) = (20 × 1000000) / (129 + 1) = 153846 (bps)

Error (%) = (153846 - 153600) / 153600 × 100= 0.16 (%):

<Notes>

- If the reload value is set to "0", the reload counter is stopped.
- If the reload value is an even number, in the received serial clock, the width of a "LOW" signal is longer than that of a "HIGH" signal by one bus clock cycle. If the value is odd, the serial clock has the same "HIGH" and "LOW" signal width.
- Set the reload value to 4 or more. Note that data may not be received normally due to the baud rate error and reload value setting.
- For allowable baud rate range, consider the effect by a jitter of the clock input to the macro.

■ Reload value and baud rate setting examples for each bus clock frequency

The following shows the reload values and baud rate setting examples.

Table 4-1 Reload values and baud rate setting examples

Baud rate (bps)	8 MHz		10 MHz		16 MHz		20 MHz		24 MHz	
	Value	ERR	Value	ERR	Value	ERR	Value	ERR	Value	ERR
4M	-	-	-	-	-	-	4	0	5	0
2.5M	-	-	-	-	-	-	7	0	-	-
2M	-	-	4	0	7	0	9	0	11	0
1M	7	0	9	0	15	0	19	0	23	0
500000	15	0	19	0	31	0	39	0	47	0
460800	-	-	-	-	-	-	-	-	51	0.16
250000	31	0	39	0	63	0	79	0	95	0
230400	-	-	-	-	-	-	86	-0.22	103	0.16
153600	51	0.16	64	0.16	103	0.16	129	0.16	155	0.16
125000	63	0	79	0	127	0	159	0	191	0
115200	-	-	86	-0.22	138	-0.08	173	-0.22	207	0.16
76800	103	0.16	129	0.16	207	0.16	259	0.16	312	-0.16
57600	138	-0.08	173	-0.22	277	-0.08	346	0.06	416	-0.08
38400	207	0.16	259	0.16	416	-0.08	520	-0.03	624	0
28800	277	-0.08	346	0.06	555	-0.08	693	0.06	832	0.04
19200	416	-0.08	520	-0.03	832	0.04	1041	-0.03	1249	0
10417	767	<0.01	959	<0.01	1535	<0.01	1919	<0.01	2303	<0.01
9600	832	0.04	1041	-0.03	1666	-0.02	2082	0.02	2499	0
7200	1110	0.01	1388	<0.01	2221	0.01	2777	<0.01	3332	0.01
4800	1666	-0.02	2082	0.02	3332	0.01	4166	<0.01	4999	0
2400	3332	0.01	4166	<0.01	6666	<0.01	8332	<0.01	9999	0
1200	6666	<0.01	8332	<0.01	13332	<0.01	16666	<0.01	19999	0
600	13332	<0.01	16666	<0.01	26666	<0.01	-	-	-	-
300	26666	<0.01	-	-	-	-	-	-	-	-

Value: BGR1/0 register set value (decimal)
ERR: Baud rate error (%)

Table 4-2 Reload values and baud rate setting examples (continued)

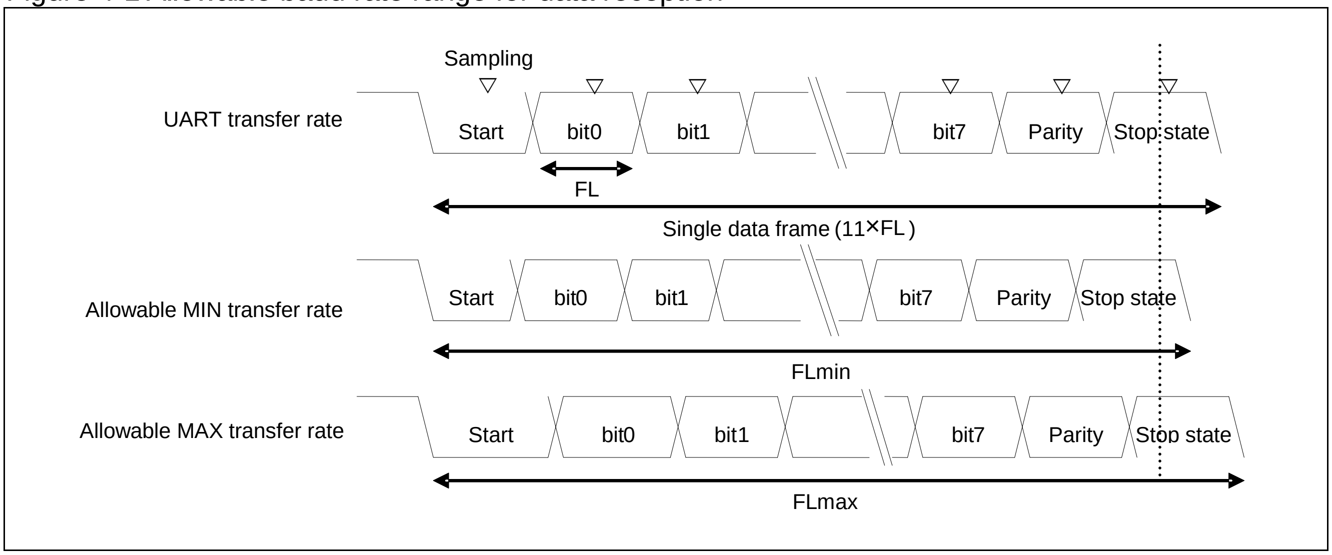
Baud rate (bps)	32 MHz		36 MHz		40 MHz		48 MHz		72 MHz	
	Value	ERR	Value	ERR	Value	ERR	Value	ERR	Value	ERR
4M	7	0	8	0	9	0	11	0	17	0
2.5M	-	-	-	-	15	0	-	-	-	-
2M	15	0	17	0	19	0	23	0	35	0
1M	31	0	35	0	39	0	47	0	71	0
500000	63	0	71	0	79	0	95	0	143	0
460800	-	-	77	0.16	86	-0.22	103	0.16	155	0.16
250000	127	0	143	0	159	0	191	0	287	0
230400	-	-	155	0.16	173	-0.22	207	0.16	312	-0.16
153600	207	0.16	233	0.16	259	0.16	312	-0.16	468	-0.05
125000	255	0	287	0	319	0	383	0	575	0
115200	277	-0.08	312	-0.16	346	0.06	416	-0.08	624	0
76800	416	-0.08	468	-0.05	520	-0.03	624	0	937	-0.05
57600	555	-0.08	624	0	693	0.06	832	0.04	1249	0
38400	832	0.04	937	-0.05	1041	-0.03	1249	0	1874	0
28800	1110	0.01	1249	0	1388	<0.01	1666	-0.02	2499	0
19200	1666	-0.02	1874	0	2082	0.02	2499	0	3749	0
10417	3071	<0.01	3455	<0.01	3839	<0.01	4607	<0.01	6911	<0.01
9600	3332	0.01	3749	0	4166	<0.01	4999	0	7499	0
7200	4443	0.01	4999	0	5555	<0.01	6666	<0.01	9999	0
4800	6666	<0.01	7499	0	8332	<0.01	9999	0	14999	0
2400	13332	<0.01	14999	0	16666	<0.01	19999	0	29999	0
1200	26666	<0.01	29999	0	-	-	-	-	-	-
600	-	-	-	-	-	-	-	-	-	-
300	-	-	-	-	-	-	-	-	-	-

Value: BGR1/0 register set value (decimal)
ERR: Baud rate error (%)

■ Allowable baud rate range for data reception

The following shows the range of baud rate error allowed for the destination to receive data.
Set the received baud rate error by using the following formulas to ensure that the value falls within the allowable range.

Figure 4-1 Allowable baud rate range for data reception



As shown in Figure 4-1, after detection of the start bit, the sampling timing of received data is determined by the counter set in the BGR1/0 register. Data can be received successfully if the bit sequence including the stop bit matches the sampling timing.

If this applies to a reception of 11 bits, a theoretical explanation can be given in the following.

Assuming that the sampling timing margin is one bus clock (ϕ), the minimum allowable transfer rate (FLmin) is determined as follows:

$FLmin = (11bits \times (V+1) - (V+1)/2 + 2)/\phi = (21V + 25)/2\ \phi$ (s) V: Reload value, ϕ: Bus clock

Thus, the maximum baud rate that allows the destination to receive data (BGmax) is determined as follows.

$BGmax = 11/FLmin = 22\phi/(21V+25)$ (bps) V: Reload value, ϕ: Bus clock

When data is received at the maximum allowable transfer rate (FLmax), the starting point of the received 11th bit is sampled.

Thus, the maximum allowable transfer rate (FLmax) is determined as follows:

$10/11 \times FLmax = (11bits \times (V+1) – (V+1)/2\ )/\phi$ V: Reload value, ϕ: Bus clock

$FLmax = (21/20 \times 11 \times (V+1)/\phi$

Assuming that the sampling timing margin (ϕ) is two clocks, the maximum allowable transfer rate (FLmax) is determined as follows:

$10/11\times FLmax = (11bits \times (V+1) – (V+1)/2 – 2)/\square\ \phi$ V: Reload value, ϕ: Bus clock

$FLmax = (21/20 \times 11 \times (V+1) – 44/20)/\phi = (231V + 187)/20\ \phi$ (s) V: Reload value, ϕ: Bus clock

Accordingly, the minimum baud rate that allows the destination to receive data (BGmin) is determined as follows:

$BGmin=11/FLmax=220\phi/(231V+187)$ (bps) V: Reload value, ϕ: Bus clock

From the above formulas for obtaining the minimum/maximum baud rate, the allowable error between UART and the destination is obtained as follows.

Reload value (V)	Maximum allowable baud rate error	Minimum allowable baud rate error
3	0%	0
10	+2.98%	-3.08%
50	+4.37%	-4.40%
100	+4.56%	-4.58%
200	+4.66%	-4.67%
32767	+4.76%	-4.76%

<Note>

Reception accuracy depends on the number of bits per frame, bus clock, and reload value. The higher the bus clock and frequency division ratio are, the higher the accuracy becomes.

■ External clock

Writing "1" to the EXT bit of the Baud Rate Generator Register (BGR) causes the baud rate generator to divide the external clock's frequency. The external clock is input from SCK.

<Note>

The external clock signal synchronizes with the internal clock on UART. Therefore, an external clock that does not allow synchronization causes unstable operation.

■ Functions of reload counter

There are two types of reload counters: The transmission reload counter and the received reload counter, both functioning as a dedicated baud rate generator. Each reload counter consists of a 15-bit register for the reload value, and generates transmitting and receiving clocks from the external or internal clock.

■ Starting counting

When the reload value is written to the Baud Rate Generator Register1, 0 (BGR1 or BGR0), the reload counter starts counting.

■ Restarting

The reload counter restarts counting in the following conditions.

● Common to transmit and received reload counters

· A programmable reset (SCR:UPCL bit)

● Received reload counter

· Detection of the start bit's falling edge in asynchronous mode

5. Setting Procedure and Program Flow in Operation Mode 0 (Asynchronous Normal Mode)

Operation mode 0 enables asynchronous bi-directional serial communications.

■ CPU-to-CPU connection

Select the bi-directional communication in operation mode 0 (normal mode). Connect two CPUs to each other as shown in Figure 5-1 and Figure 5-2.

Figure 5-1 A connection example of bi-directional communications in UART operation mode 0 (with flow control disabled)

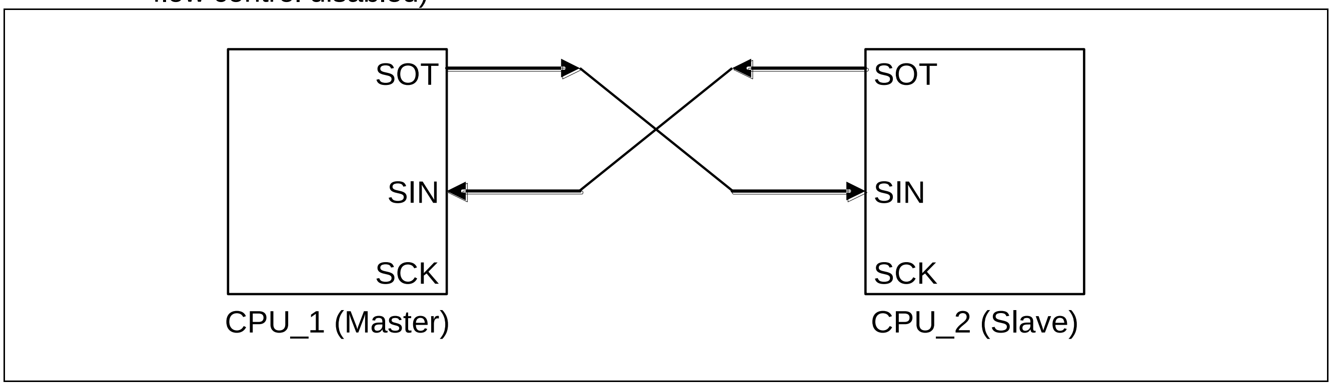
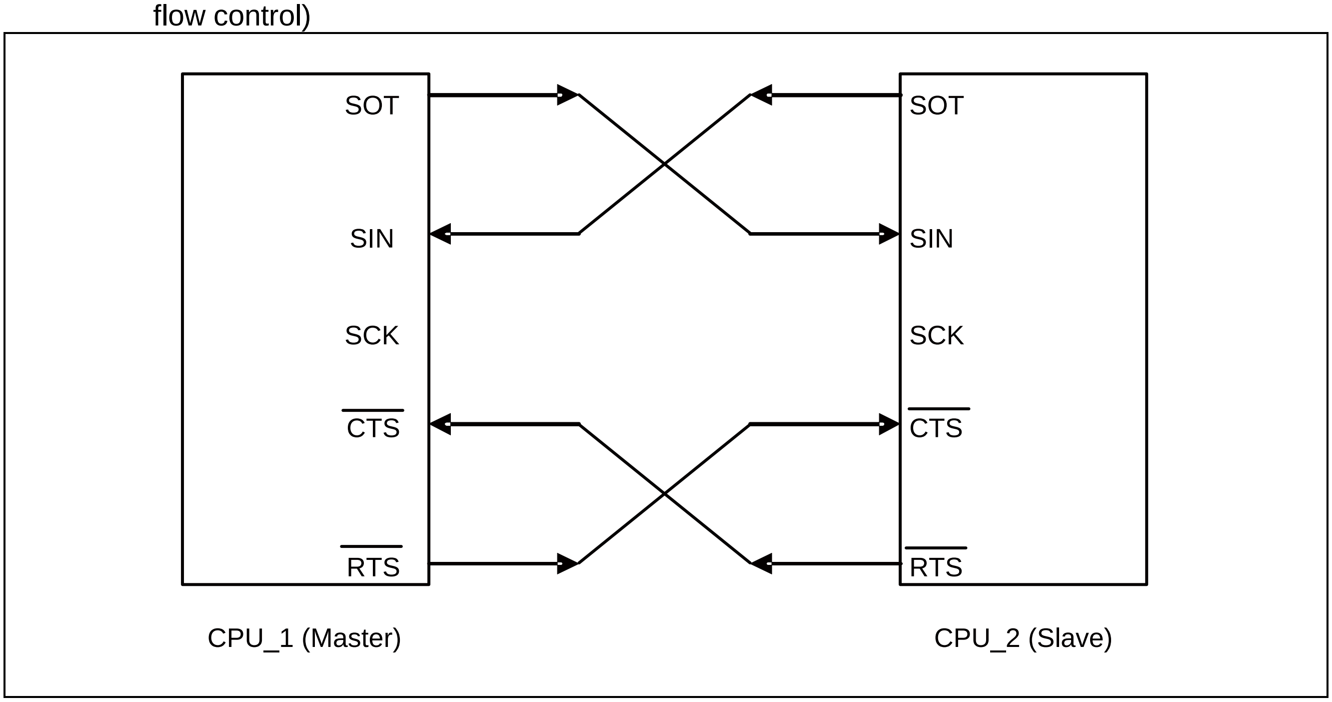


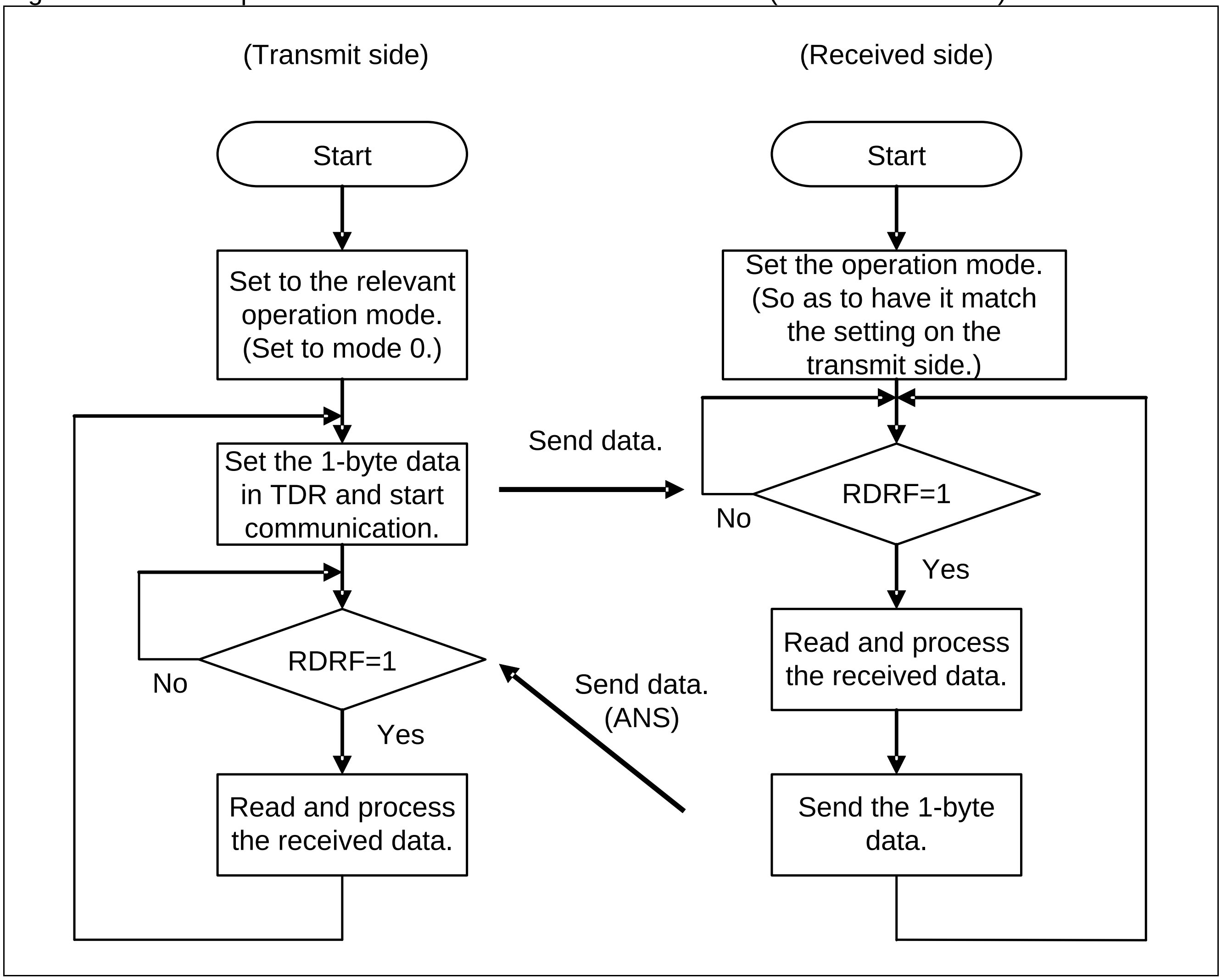
Figure 5-2 A connection example of bi-directional communications in UART operation mode 0 (with flow control)



■ Flowcharts

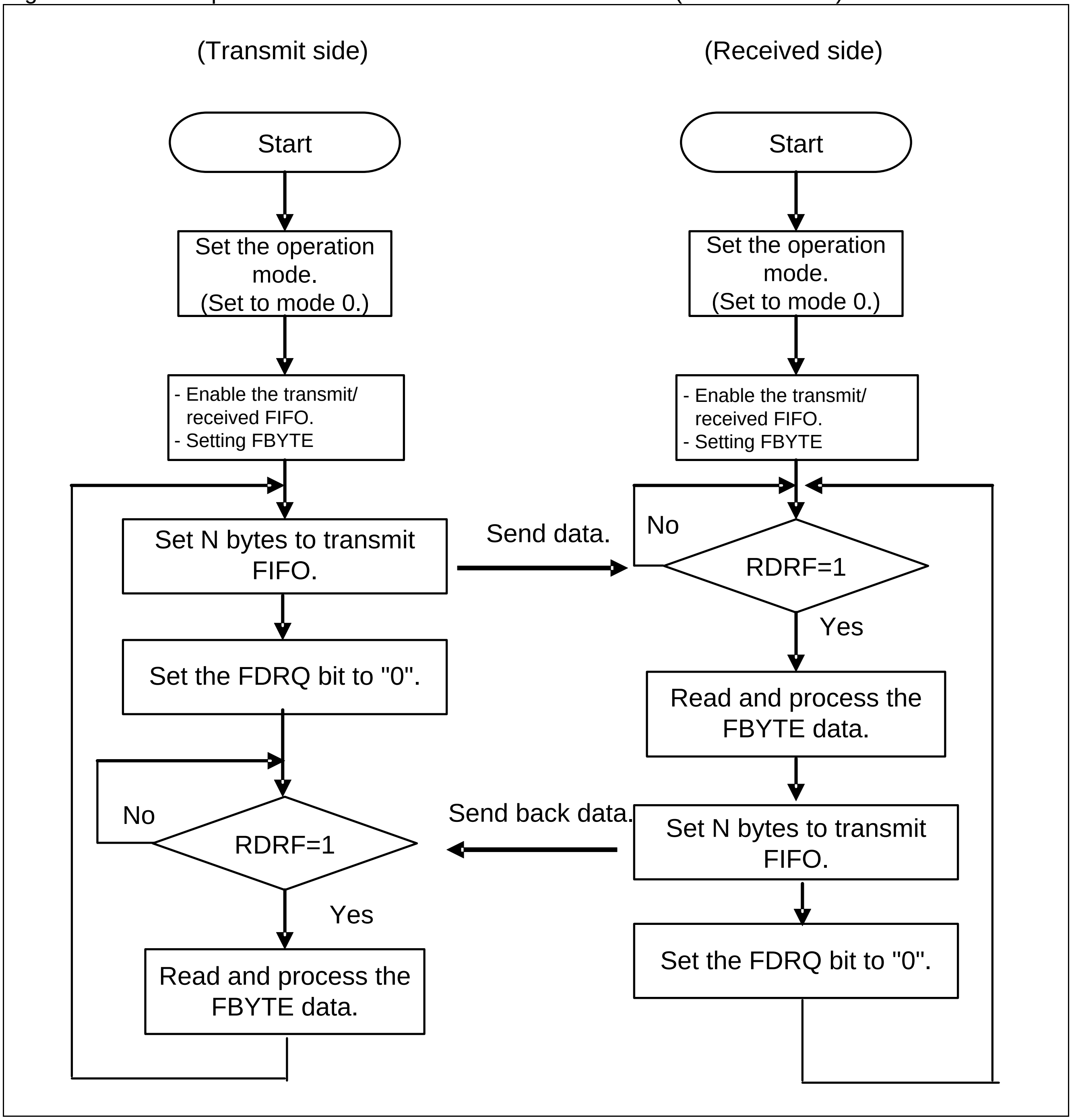
● If FIFO is not used

Figure 5-3 An example of bidirectional communication flowchart (if FIFO is not used)



● If FIFO is used

Figure 5-4 An example of bidirectional communication flowchart (if FIFO is used)



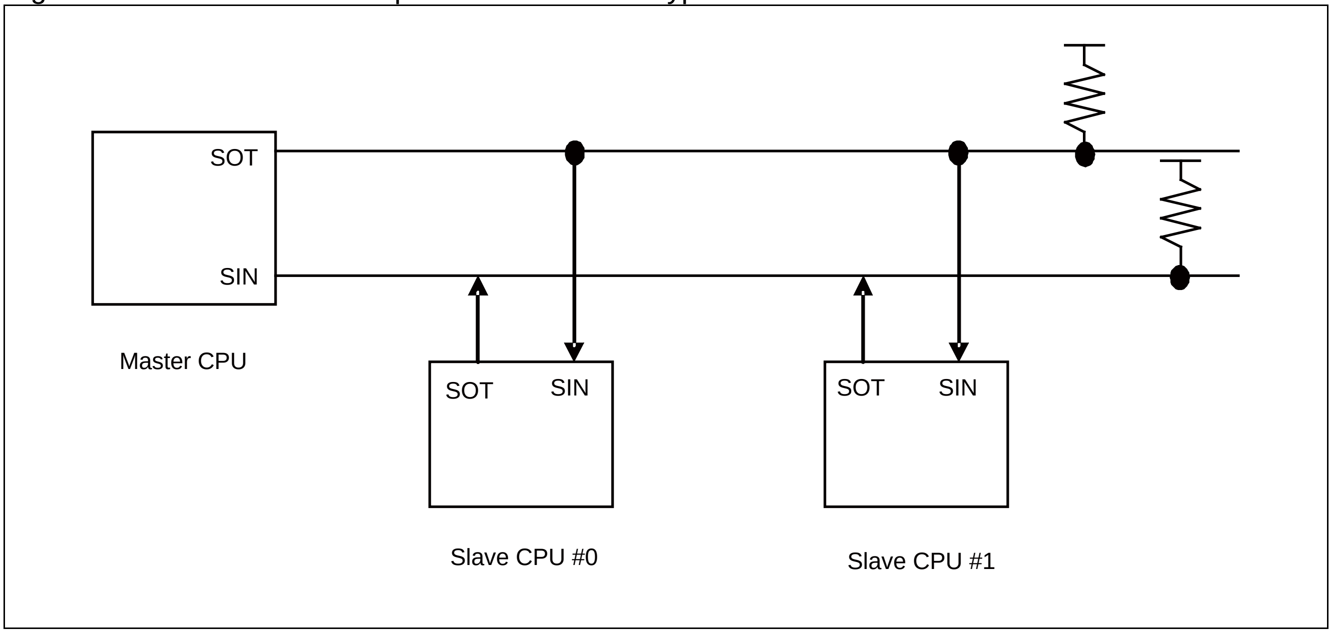
6. Setting Procedure and Program Flow in Operation Mode 1 (Asynchronous Multiprocessor Mode)

In operation mode 1 (multiprocessor mode), communications by master/slave connections with multiple CPUs are enabled. Either the master or slave function is available.

■ CPU-to-CPU connection

In a master/slave type communications, as shown in Figure 6-1, the communications system is configured with two common communication lines connected to the master CPU and multiple slave CPUs. UART can be used either as a master or a slave.

Figure 6-1 A connection example for master/slave type communications on UART



■ Function selection

In master/slave type communications, select the operation mode and data transfer system, as shown in Table 6-1.

Table 6-1 Selection of master/slave type communications functions

	Operation mode		Data	Parity	Stop state bit	bit direction
	Master mode CPU	Slave mode CPU				
Address transmit and reception	Mode 1 (A/D bit transmit)	Mode 1 (A/D bit reception)	AD=1 + 7 or 8 bits Address	OFF	One bit or 2 bits	LSB or MSB first
Data transmit and reception			AD=0 + 7 or 8 bits Data			

<Note>

In operation mode 1, operate in word access mode for transmit/received data (TDR/RDR).

● Communications procedure

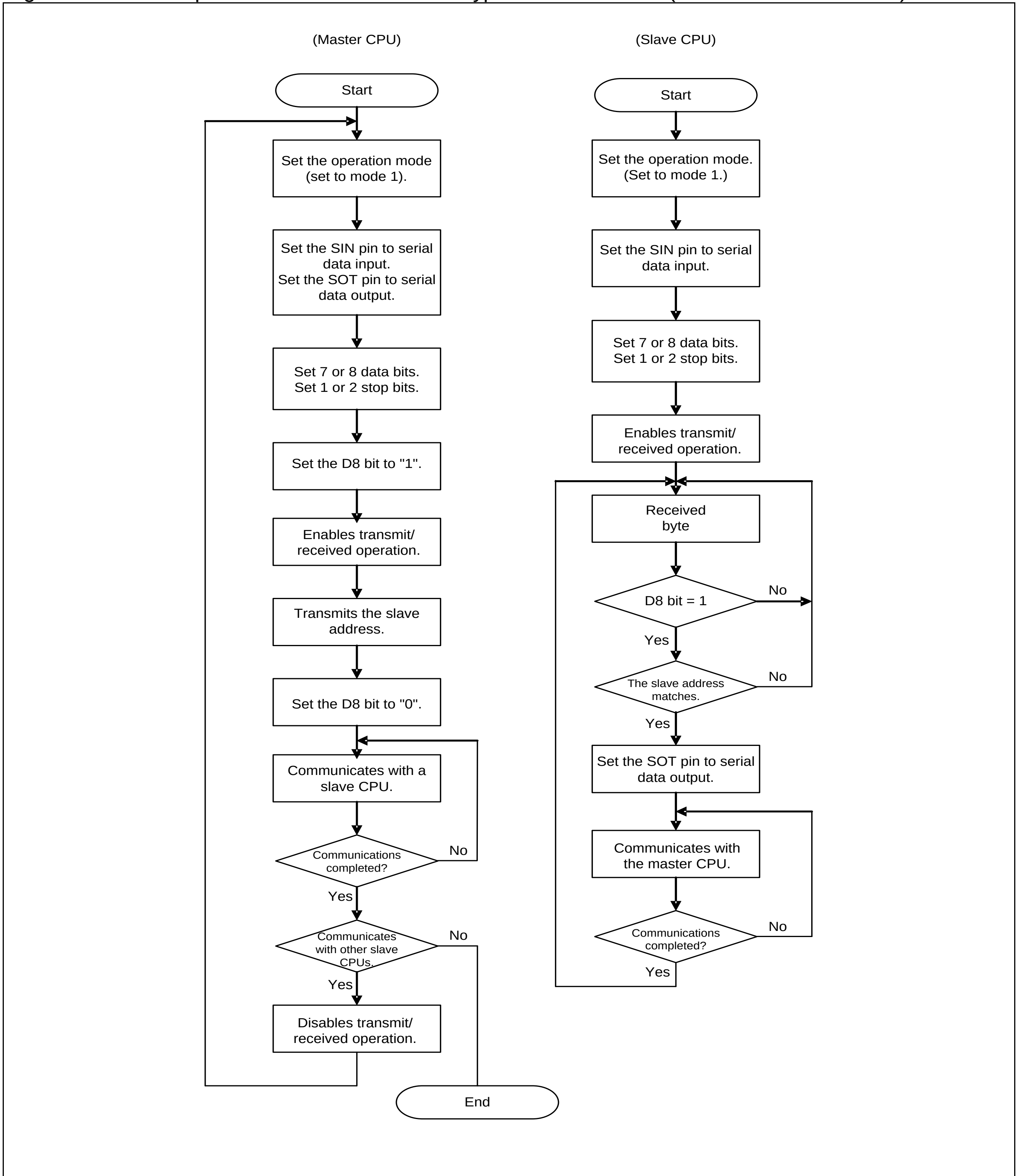
Communications start when the master CPU transmits address data. Address data is a data set whose D8 bit is "1", and used for selecting a slave CPU to communicate with. Each slave CPU judges the address as programmed, and communicates with the master CPU if that address matches the assigned address.

Figure 6-2 and Figure 6-3 show flowcharts of master/slave type communications (in multiprocessor mode).

■ Flowcharts

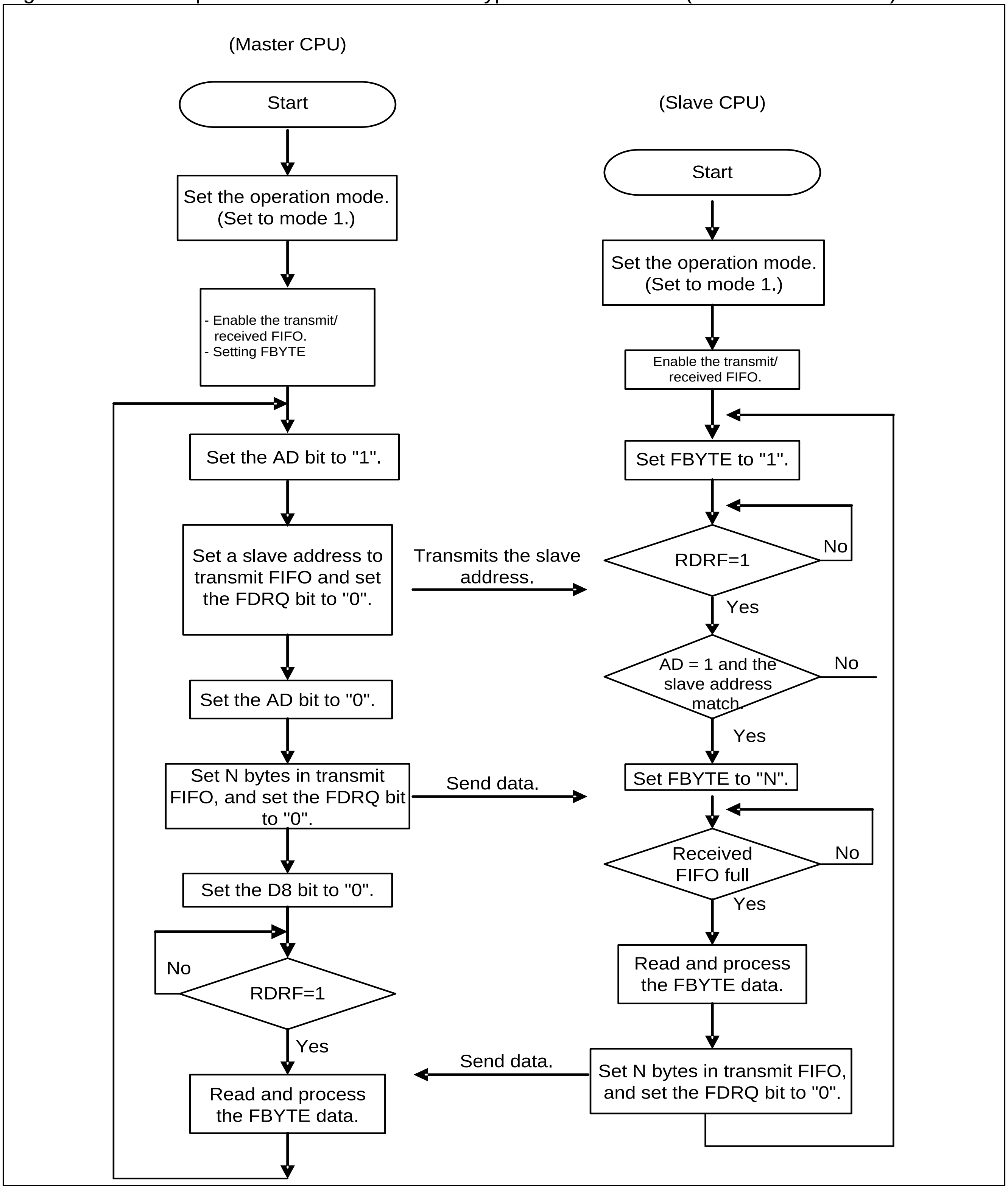
● If FIFO is not used

Figure 6-2 An example flowchart for master/slave type communications (if FIFO buffer is not used)



● If FIFO is used

Figure 6-3 An example flowchart for master/slave type communications (if FIFO buffer is used)



7. UART (Asynchronous Serial Interface) Registers

This section provides a list of UART (Asynchronous Serial Interface) registers.

■ UART (Asynchronous Serial Interface) registers list

Table 7-1 UART (Asynchronous Serial Interface) register list

	bit15 bit8	bit7 bit0
UART	SCR (Serial Control Register)	SMR (Serial Mode Register)
	SSR (Serial Status Register)	ESCR (Extended Communication Control Register)
	RDR/TDR (Transmit/Received Data Register)	
	BGR1 (Baud Rate Generator Register 1)	BGR0 (Baud Rate Generator Register 0)
FIFO	FCR1 (FIFO Control Register 1)	FCR0 (FIFO Control Register 0)
	FBYTE2 (FIFO2 Byte Register)	FBYTE1 (FIFO1 Byte Register)

Table 7-2 UART (Asynchronous Serial Interface) bit assignment

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR/ SMR	UPCL	-	-	RIE	TIE	TBIE	RXE	TXE	MD2	MD1	MD0	-	SBL	BDS	-	SOE
SSR/ ESCR	REC	-	PE	FRE	ORE	RDRF	TDRE	TBI	FLWEN	ESBL	INV	PEN	P	L2	L1	L0
TDR/ (RDR)	-							D8 (AD)	D7	D6	D5	D4	D3	D2	D1	D0
BGR1/ BGR0	EXT	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
FCR1/ FCR0	-	-	-	FLSTE	FRIIE	FDRQ	FTIE	FSEL	-	FLST	FLD	FSET	FCL2	FCL1	FE2	FE1
FBYTE2/ FBYTE1	FD15	FD14	FD13	FD12	FD11	FD10	FD9	FD8	FD7	FD6	FD5	FD4	FD3	FD2	FD1	FD0

■ Operation mode

UART (Asynchronous Serial Interface) operates in two different modes. The Serial Mode Register (SMR) determines the mode to be enabled, depending on its setting, MD2, MD1 or MD0.

Table 7-3 UART (Asynchronous Serial Interface) operation modes

Operation mode	MD2	MD1	MD0	Type
0	0	0	0	UART0 (asynchronous normal mode)
1	0	0	1	UART1 (asynchronous multiprocessor mode)

7.1. Serial Control Register (SCR)

The Serial Control Register (SCR) can perform transmit/received enable/disable, transmit/received interrupt enable/disable, transmit bus idle interrupt enable/disable and UART reset operations.

bit	15	14	13	12	11	10	9	8	7 ... 0
Field	UPCL	-	-	RIE	TIE	TBIE	RXE	TXE	(SMR)
Attribute	R/W	-	-	R/W	R/W	R/W	R/W	R/W	
Initial value	0	-	-	0	0	0	0	0	

[bit15] UPCL: Programmable Clear bit

Initializes the UART internal state.

If set to "1",

- UART is reset directly (software reset). However, the current register settings are maintained. The transmit or received state is disconnected immediately.
- The baud rate generator reloads the BGR1/0 register value and restarts operation.
- All of transmit/received interrupt factors (SSR:PE, FRE, ORE, RDRF, TDRE and TBI) are initialized (to 0b000011).
- $\overline{\text{RTS}}$ signal is cleared to "LOW".

If set to "0", it has no effect on operation.

"0" is always read during reading.

Value	Description	
	At writing	At reading
0	No effect on opreration.	"0" is always read.
1	Programmable clear	

<Notes>

- Disable an interrupt first, and then execute the programmable clear instruction.
- If the FIFO operation is used, disable it (FCR0:FE[2:1]=00) first and then execute Programmable Clear.

[bit14:13] - : Unused bits

These bits' values are undefined when read.
These bits have no effect when written.

[bit12] RIE: Received interrupt enable bit

- This bit enables or disables an output of received interrupt request to the CPU.
- If the RIE bit and the received data flag bit (SSR:RDRF) are "1", or if any of the error flag bits (SSR:PE, ORE or FRE) is "1", a received interrupt request is output.

Value	Description
0	Disables the received interrupt.
1	Enables the received interrupt.

[bit11] TIE: Transmit interrupt enable bit

- This bit enables or disables an output of Transmit Interrupt Request to the CPU.
- If the TIE bit and SSR:TDRE bit are "1", a Transmit Interrupt Request is output.

Value	Description
0	Disables a transmit interrupt.
1	Enables a transmit interrupt.

[bit10] TBIE: Transmit bus idle interrupt enable bit

- This bit enables or disables an output of transmit bus idle interrupt request to the CPU.
- If the TBIE bit and TBI bit are "1", a transmit bus idle interrupt request is output.

Value	Description
0	Disables the transmit bus idle interrupt.
1	Enables the transmit bus idle interrupt.

[bit9] RXE: Received operation enable bit

Enables or disables UART received operation.

Value	Description
0	Disables data received.
1	Enables data received.

<Notes>

- Reception is not started unless the falling edge of the start bit (in NRZ format, when ESCR:INV=0) is input even if reception is enabled (RXE=1). (In the inverted NRZ format (ESCR:INV=1), reception is not started unless the rising edge is input).
- If data reception is disabled (RXE=0) during the received operation, the current data reception is stopped immediately.
- When the received operation is disabled (RXE=0), the $\overline{\text{RTS}}$ signal is fixed to "LOW".

[bit8] TXE: Transmission operation enable bit

Enables or disables the UART transmission operation.

Value	Description
0	Disables the transmission.
1	Enables the transmission.

<Note>

If data transmission is disabled (TXE=0) during the transmission operation, the current data transmission is stopped immediately.

7.2. Serial Mode Register (SMR)

The Serial Mode Register (SMR) is used to set the operation mode, transfer direction, data length and to select the stop bit length as well as to enable/disable output of serial data to their pins.

bit	15 ... 8	7	6	5	4	3	2	1	0
Field	(SCR)	MD2	MD1	MD0	Reseved	SBL	BDS	Reserved	SOE
Attribute		R/W	R/W	R/W	-	R/W	R/W	-	R/W
Initial value		0	0	0	-	0	0	-	0

[bit7:5] MD2, MD1, MD0: Operation mode set bit

Set operation mode of the Asynchronous Serial Interface..

* This chapter explains the registers and their operation in operation mode 0 (asynchronous normal mode) and in operation mode 1 (asynchronous multiprocessor mode).

bit7	bit6	bit5	Description
0	0	0	Operation mode 0 (asynchronous normal mode)
0	0	1	Operation mode 1 (asynchronous multiprocessor mode)
0	1	0	Operation mode 2 (clock sync mode)
0	1	1	Operation mode 3 (LIN communication mode)
1	0	0	Operation mode 4 (I^2C mode)

<Notes>

- Any bit setting other than above is inhibited.
- To switch the current operation mode, issue a programmable clear instruction (SCR:UPCL=1) and switch the operation mode continuously.
- After the operation mode has been switched, set each register correctly.

[bit4] Reserved: Reserved bit

This bit value is undefined when read.
This bit has no effect when written.

[bit3] SBL: Stop bit length select bit

This bit sets a stop bit length (the frame end mark of the transmit data).

Value	Description	
0	ESCR:ESBL=0	1 bit
	ESCR:ESBL=1	3 bits
1	ESCR:ESBL=0	2 bits
	ESCR:ESBL=1	4 bits

<Notes>

- In the reception operation, only the first bit of the stop bit data is detected.
- Always set this bit when transmission is disabled (SCR:TXE=0).

[bit2] BDS: Transfer direction select bit

Specifies to transmit the least significant bit of the transmit serial data first (LSB first; BDS=0) or the most significant bit first (MSB first; BDS=1).

Value	Description
0	LSB first (The least significant bit is first transferred.)
1	MSB first (The most significant bit is first transferred.)

<Note>

Set this bit when transmission and reception are disabled (SCR:TXE=SCR:RXE=0).

[bit1] Reserved: Reserved bit

The read value is "0". Be sure to write "0".

[bit0] SOE: Serial data output enable bit

This bit enables or disables a serial data output.

Value	Description
0	Disables a serial data output.
1	Enables a serial data output.

<Note>

If this bit is used as the SOT pin, the GPIO must also be set.

7.3. Serial Status Register (SSR)

The Serial Status Register (SSR) is used to check the current transmit/received state, check the received error flag, and clears the received error flag.

bit	15	14	13	12	11	10	9	8	7 ... 0
Field	REC	-	PE	FRE	ORE	RDRF	TDRE	TBI	(ESCR)
Attribute	R/W	-	R	R	R	R	R	R	
Initial value	0	-	0	0	0	0	1	1	

[bit15] REC: Received error flag clear bit

This bit clears the PE, FRE and ORE flags of the Serial Status Register (SSR).

Value	Description	
	At writing	At reading
0	No effect on operation.	"0" is always read.
1	Clears the received error flag (PE, FRE, ORE).	

[bit14] - : Unused bit

This bit value is undefined when read.
This bit has no effect when written.

[bit13] PE: Parity error flag bit (only functions in operation mode 0)

- If a parity error occurs during data received with ESCR:PEN=1, this bit is set to "1". This is cleared if the REC bit of Serial Status Register (SSR) is set to "1".
- If the PE bit and SCR:RIE bit are "1", a received interrupt request is output.
- If this flag is set, data in the Received Data Register (RDR) is invalid.
- If this flag is set when received FIFO is used, the received FIFO enable bit is cleared and the received data is not stored in received FIFO.

Value	Description
0	No parity error occurred.
1	A parity error occurred.

[bit12] FRE: Framing error flag bit

- If a framing error occurs during data reception, this bit is set to "1". This is cleared if the REC bit of Serial Status Register (SSR) is set to "1".
- If the FRE bit and SCR:RIE bit are "1", a received interrupt request is output.
- If this flag is set, data in the Received Data Register (RDR) is invalid.
- If this flag is set when received FIFO is used, the received FIFO enable bit is cleared and the received data is not stored in received FIFO.

Value	Description
0	No framing error occurred.
1	A framing error occurred.

[bit11] ORE: Overrun error flag bit

- If an overrun occurs during data reception, this bit is set to "1". This is cleared if the REC bit of Serial Status Register (SSR) is set to "1".
- If the ORE and SCR:RIE bits are "1", a received interrupt request is output.
- If this flag is set, data in the Received Data Register (RDR) is invalid.
- If this flag is set when received FIFO is used, the received FIFO enable bit is cleared and the received data is not stored in received FIFO.

Value	Description
0	No overrun error occurred.
1	An overrun error occurred.

[bit10] RDRF: Received data full flag bit

- This flag shows the state of the Received Data Register (RDR).
- When the received data is loaded in the RDR, this bit is set to "1". When data is read from the Received Data Register (RDR), this bit is cleared to "0".
- If the RDRF bit and SCR:RIE bit are "1", a received interrupt request is output.
- If the received FIFO is used and if a certain count of data is received by the received FIFO, the RDRF bit is set to "1".
- If received FIFO is used, if both of the following conditions are satisfied, and if the Received Idle state continues more than 8 baud rate clocks, the RDRF bit is set to "1".
  - The received FIFO idle detection enable bit (FCR1:FRIIE) is "1".
  - The preset data amount is not received and some data remains in received FIFO.

  If the RDR data is read during counting of 8 clocks, this counter is reset to "0", and counting for 8 clocks is restarted.
- If the received FIFO is used and if this buffer is emptied, this bit is cleared to "0".

Value	Description
0	The Received Data Register (RDR) is empty.
1	The Received Data Register (RDR) contains data.

[bit9] TDRE: Transmit data empty flag bit

- This flag shows the state of Transmit Data Register (TDR).
- If transmit data is written in the TDR, this bit is set to "0" to indicate that the TDR contains valid data. When data is loaded to the transmit shift register and when the transmission is started, this bit is set to "1" to indicate that the TDR does not have the valid data.
- If the TDRE bit and SCR:TIE bit are "1", a transmit interrupt request is output.
- When the UPCL bit of the Serial Control Register (SCR) is set to "1", the TDRE bit is set to "1".
- For the TDRE bit set/reset timing when transmit FIFO is used, see "2.4 Interrupt and flag set timing when transmit FIFO is used".

Value	Description
0	The Transmit Data Register (TDR) contains data.
1	The Transmit Data Register is empty.

[bit8] TBI: Transmit bus idle flag

- This bit indicates that UART is not transmitting data.
- When transmit data is written in the Transmit Data Register (TDR), this bit is set to "0".
- If the Transmit Data Register is empty (TDRE=1) and not transmitting data, this bit is set to "1".
- When the UPCL bit of the Serial Control Register (SCR) is set to "1", the TBI bit is set to "1".
- If this bit is "1" and if the transmit bus idle interrupt is enabled (SCR:TBIE=1), a transmit interrupt request is output.

Value	Description
0	During data transmission
1	No data transmission

7.4. Extended Communication Control Register (ESCR)

The Extended Communication Control Register (ESCR) is used to set a transmit/received data length, enable/disable a parity bit, select a parity bit, invert the serial data format and set stop bit length selection.

bit	15 ... 8	7	6	5	4	3	2	1	0
Field	(SSR)	FLWEN	ESBL	INV	PEN	P	L2	L1	L0
Attribute		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value		0	0	0	0	0	0	0	0

[bit7] FLWEN: Flow control enable bit

Selects to enable or disable the hardware flow control operation.

Value	Description
0	Disables hardware flow control.
1	Enables hardware flow control.

<Notes>

- Set this bit when data transmission and reception is disabled (SCR:TXE=0, RXE=0).
- Set this bit to "1" only when the hardware flow control is desired.

[bit6] ESBL: Extension stop bit length select bit

This bit sets a stop bit length (the frame end mark of the transmit data).

Value	Description	
0	SMR:SBL=0	1 bit
	SMR:SBL=1	2 bits
1	SMR:SBL=0	3 bits
	SMR:SBL=1	4 bits

<Notes>

- In the reception operation, only the first bit of the stop bit data is detected.
- Always set this bit when transmission is disabled (SCR:TXE=0).

[bit5] INV: Inverted serial data format bit

Selects NRZ or inverted NRZ for the serial data format.

Value	Description
0	NRZ format
1	Inverted NRZ format

[bit4] PEN: Parity enable bit (only functions in operation mode 0)

Sets to add (for transmit) and detect (for reception) a parity bit or not to.

Value	Description
0	Disables parity.
1	Enables parity.

<Note>

In operation mode 1, this bit is internally fixed at "0".

[bit3] P: Parity select bit (only functions in operation mode 0)

When set to enable parity (ESCR:PEN=1, this bit is set to either odd-number parity "1" or even-number parity "0".

Value	Description
0	Even-number parity
1	Odd-number parity

[bit2:0] L2, L1, L0: Data length select bit

These bits set a length of transmit/received data.

- If set to "0b000", the data length is set to eight bits.
- If set to "0b001", the data length is set to five bits.
- If set to "0b010", the data length is set to six bits.
- If set to "0b011", the data length is set to seven bits.
- If set to "0b100", the data length is set to nine bits.

bit2	bit1	bit0	Description
0	0	0	8-bit length
0	0	1	5-bit length
0	1	0	6-bit length
0	1	1	7-bit length
1	0	0	9-bit length

<Notes>

- Any setting other than the above is inhibited.
- In operation mode 1, set the data length to seven or eight bits. Any other setting is inhibited.

7.5. Received Data Register/Transmit Data Register (RDR/TDR)

The Received and Transmit Data Registers are allocated at the same address. This register functions as the Received Data Register when data is read from it. This register operates as the Transmit Data Register when data is written in it.
When the FIFO operation is enabled, the RDR/TDR address functions as the FIFO read/write address.

■ Received Data Register (RDR)

bit	15 ... 9	8	7	6	5	4	3	2	1	0
Field		D8	D7	D6	D5	D4	D3	D2	D1	D0
Attribute		R	R	R	R	R	R	R	R	R
Initial value		0	0	0	0	0	0	0	0	0

The Received Data Register (RDR) is a 9-bit data buffer register for serial data reception.

- When serial data signals are sent to the Serial Input pin (SIN), they are converted by a shift register and stored in the Received Data Register (RDR).
- The upper bits are set to "0" according to the data length, as follows.

Data length	D8	D7	D6	D5	D4	D3	D2	D1	D0
9 bits	X	X	X	X	X	X	X	X	X
8 bits	0	X	X	X	X	X	X	X	X
7 bits	0	0	X	X	X	X	X	X	X
6 bits	0	0	0	X	X	X	X	X	X
5 bits	0	0	0	0	X	X	X	X	X

(X represents the received data bit.)

- When the received data is stored in the Received Data Register (RDR), the received data full flag bit (SSR:RDRF) is set to "1". If a received interrupt is enabled (SSR:RIE=1), a received interrupt request is generated.
- The Received Data Register (RDR) must be read only when the received data full flag bit (SSR:RDRF) is "1". When data is read from the Received Data Register (RDR), the received data full flag bit (SSR:RDRF) is cleared to "0" automatically.
- If a received error occurs (when SSR:PE, ORE or FRE is "1"), data in the Received Data Register (RDR) becomes invalid.
- In operation mode 1 (multiprocessor mode), 7-bit or 8-bit long operation takes place and the received AD bit is stored in the D8 bit.
- For 9-bit long data transfer and in operation mode 1, data must be read from RDR by 16-bit data accessing.

<Notes>

- If the Received FIFO is used and if the preset amount of data is received in the Received FIFO buffer, SSR:RDRF is set to "1".
- If the received FIFO is used and if this buffer is emptied, the SSR:RDRF bit is cleared to "0".
- If a received error occurs when received FIFO is used (SSR:PE, ORE, or FRE is "1"), the received FIFO enable bit is cleared and the received data is not stored in the received FIFO buffer.

■ Transmit Data Register (TDR)

bit	15 ... 9	8	7	6	5	4	3	2	1	0
Field		D8	D7	D6	D5	D4	D3	D2	D1	D0
Attribute		W	W	W	W	W	W	W	W	W
Initial value		1	1	1	1	1	1	1	1	1

The Transmit Data Register (TDR) is a 9-bit data buffer register for serial data transmission.

- If data transmission is enabled (SCR:TXE=1) and if the transmit data is written in the Transmit Data Register (TDR), the transmit data is transferred to the Transmit Shift Register. The transmit data is then converted into serial data and sent out from the serial data output pin (SOT).
- The upper bits are sequentially made invalid according to the data length as follows.

Data length	D8	D7	D6	D5	D4	D3	D2	D1	D0
9 bits	X	X	X	X	X	X	X	X	X
8 bits	Invalid	X	X	X	X	X	X	X	X
7 bits	Invalid	Invalid	X	X	X	X	X	X	X
6 bits	Invalid	Invalid	Invalid	X	X	X	X	X	X
5 bits	Invalid	Invalid	Invalid	Invalid	X	X	X	X	X

- When the transmit data is written in the Transmit Data Register (TDR), the transmit data empty flag (SSR:TDRE) is cleared to "0".
- When the transmit data is transferred to the transmit shift register and data transmission is started, and if transmit FIFO is disabled or if transmit FIFO is empty, the transmit data empty flag (SSR:TDRE) is set to "1".
- If the transmit data empty flag (SSR:TDRE) is "1", transmit data can be written. If a transmit interrupt is enabled, a transmit interrupt occurs. Perform transmit data write after a transmit interrupt is generated or when the transmit data empty flag (SSR:TDRE) is "1".
- If the transmit data empty flag (SSR:TDRE) is "0" and transmit FIFO is disabled or the transmit FIFO buffer is full, no transmit data can be written.
- In operation mode 1 (multiprocessor mode), 7-bit or 8-bit long operation takes place and the AD bit is sent by writing to the D8 bit.
- For 9-bit long data transfer and in operation mode 1, data must be written in TDR by 16-bit data accessing.

<Notes>

- The Transmit Data Register is a write-only register. While the Received Data Register is a read-only register. As the transmission and received registers are allocated at the same address, the write and read values differ from each other. Therefore, the INC/DEC instruction and other read-modify-write (RMW) instructions cannot be used.
- For the transmit data empty flag (SSR:TDRE) set timing when transmit FIFO is used, see "2.4 Interrupt and flag set timing when transmit FIFO is used".

7.6. Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0)

Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0) are used to set a frequency division ratio of serial clocks. Also, an external clock can be selected as the clock source of the reload counter.

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	EXT	(BGR1)							(BGR0)							
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- The Baud Rate Generator Registers are used to set a frequency division ratio of serial clocks.
- The BGR1 register corresponds to the upper bits, and the BGR0 register corresponds to the lower bits. The reload value to be counted can be written, and the BGR1/BGR0 set value can be read.
- When the reload value is written in Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0), the reload counter starts its counting.
- The EXT bit (bit15) specifies to use the clock source of reload counter as the internal clock or the external clock. If EXT=0 is set, an internal clock is used. If EXT=1 is set, an external clock is used.
  The external clock is input from SCK.

[bit15] EXT: External clock select bit

Value	Description
0	Uses the internal clock.
1	Uses an external clock.

[bit14:8] BGR1: Baud Rate Generator Register 1

Process	Description
Write	Writes data in bit8 to 14 of reload counter.
Read	Read the BGR1 set value.

[bit7:0] BGR0: Baud Rate Generator Register 0

Process	Description
Write	Write data in bit0 to 7 of reload counter.
Read	Read the BGR0 set value.

<Notes>

- Data must be written in the Baud Rate Generator Registers (BGR1 and BGR0) by 16-bit data accessing.
- If the current values of Baud Rate Generator Registers (BGR1, BGR0) are changed, the new values are reloaded only after the counter value has reached "15h00". In order to validate the new set values immediately, change the BGR1/BGR0 set values and execute the programmable clear (UPCL).
- If the reload value is an even number, in the received serial clock, the width of a "LOW" signal is longer than that of a "HIGH" signal by one bus clock cycle. If the value is an odd number, the width of a LOW signal is the same as that of a HIGH signal.
- Set a value "4" or higher to BGR1/BGR0. Note that data may not be received successfully depending on the baud rate error and reload value settings.
- To change the setting to an external clock (EXT=1) while the Baud Rate Generator is running, write "0" to the Baud Rate Generators 1 and 0 (BGR1, BGR0), execute Programmable Clear (UPCL) and then set for an external clock (EXT=1).

7.7. FIFO Control Register 1 (FCR1)

The FIFO Control Register (FCR1) is used to set the FIFO test, select the transmit or received FIFO, enable the transmit FIFO interrupt, and control the interrupt flag.

bit	15	14	13	12	11	10	9	8	7 ... 0
Field		Reserved		FLSTE	FRIIE	FDRQ	FTIE	FSEL	(FCR0)
Attribute		-		R/W	R/W	R/W	R/W	R/W	
Initial value		-		0	0	1	0	0	

[bit15:13] Reserved bits

The read value is "0". Be sure to write "0".

[bit12] FLSTE: Re-transmission data lost detect enable bit

This bit enables the FIFO re-transmission data lost flag (FLST) detection.

If set to "0", the FLST bit detection is disabled.
If set to "1", the FLST bit detection is enabled.

Value	Description
0	Disables the Data Lost detection.
1	Enables the Data Lost detection.

<Note>

If you wish to set this bit to "1", set the FSET bit to "1" first, and then set this bit to "1".

[bit11] FRIIE: Received FIFO idle detection enable bit

This bit sets to detect the received idle state if the received FIFO contains valid data and if it continues more than 8-bit hours. If the received interrupt is enabled (SCR:RIE=1), a received interrupt is generated when the received idle state is detected.

If set to "0", a detection of received idle state is disabled.
If set to "1", a detection of received idle state is enabled.

Value	Description
0	Disables the received FIFO idle detection.
1	Enables the received FIFO idle detection.

<Note>

In case of using Received FIFO, set this bit to "1".

[bit10] FDRQ: Transmit FIFO data request bit

This bit requests for the transmit FIFO data.
If this bit is "1", the transmit data is being requested. At this time, if a transmit FIFO interrupt is enabled (FTIE=1), a transmit FIFO interrupt request is output.

The FDRQ bit is set when:

· The FBYTE (for transmission) is "0" (Transmit FIFO is empty).

The FDRQ bit is reset when:

· This bit is set to "0".
· Transmit FIFO is filled with data.

Value	Description
0	Does not request for the transmit FIFO data.
1	Requests for the transmit FIFO data.

<Notes>

· "0" written when transmit FIFO is enabled is valid.
· If the FBYTE (for transmission) is "0", this bit cannot be set to "0".
· If this bit is set to "1", it has no effect on the operation.
· If a read-modify-write instruction is issued, "1" is read.

[bit9] FTIE: Transmit FIFO interrupt enable bit

This bit enables a transmit FIFO interrupt. If this bit is set to "1", an interrupt occurs when the FDRQ bit is set to "1".

Value	Description
0	Disables the transmit FIFO interrupt.
1	Enables the transmit FIFO interrupt.

[bit8] FSEL: FIFO select bit

This bit selects the transmit or received FIFO.

If set to "0", the transmit FIFO is assigned to FIFO1, and the received FIFO is assigned to FIFO2.
If set to "1", the transmit FIFO is assigned to FIFO2, and the received FIFO is assigned to FIFO1.

Value	Description
0	Transmit FIFO:FIFO1; Received FIFO:FIFO2
1	Transmit FIFO:FIFO2; Received FIFO:FIFO1

<Notes>

- This bit is not cleared by the FIFO Reset (FCR0:FCL[2:1]=11).
- To change this bit state, first disable the FIFO operation (FCR0:FE[2:1]=00).

7.8. FIFO Control Register 0 (FCR0)

The FIFO Control Register 0 (FCR0) is used to enable/disable the FIFO operation, reset FIFO, save the read pointer, and set the data re-transmission.

bit	15 ... 8	7	6	5	4	3	2	1	0
Field	(FCR1)	-	FLST	FLD	FSET	FCL2	FCL1	FE2	FE1
Attribute		-	R	R/W	R/W	R/W	R/W	R/W	R/W
Initial value		0	0	0	0	0	0	0	0

[bit7] - : Unused bit

When read, always "0" is read.

When written, always set this bit to "0".

[bit6] FLST: FIFO re- transmit data lost flag bit

This bit shows that the re- transmit data of transmit FIFO has been lost.

The FLST bit is set when:

- Data is written (overwritten) in the FIFO buffer when the FLSTE bit of FIFO Control Register 1 (FCR1) is "1" and the write pointer for transmit FIFO matches the read pointer which has been saved by the FSET bit.

The FLST bit is reset when:

- FIFO is reset (FCL bit is set to "1").
- The FSET bit is set to "1".

If this bit is set to "1", the data identified by the read pointer (saved by the FSET bit) is overwritten. Therefore, the FLD bit cannot set the data re-transmission even if an error has occurred. If this bit is set to "1" and if you wish to re-transmit data, first reset FIFO. Then, write data in the FIFO buffer again.

Value	Description
0	No Data Lost has occurred.
1	Data Lost has occurred.

[bit5] FLD: FIFO pointer reload bit

This bit reloads the data, being saved in transmit FIFO by the FSET bit, to the reload pointer. This bit can be used to re-transmit data after a communication error or others have occurred.

When the re-transmission setting has finished, this bit is set to "0".

Value	Description
0	Not reloaded
1	Reloaded

<Notes>

- If this bit is "1", data is being reloaded in the read pointer. Therefore, data writing except for FIFO reset is disabled.
- When FIFO is enabled or when data is being transmitted, this bit cannot be set to "1".
- After you have set the TIE bit and TBIE bit to "0", set this bit to "1". After you have enabled transmit FIFO, set the SCR:TIE bit and SCR:TBIE bit to "1".

[bit4] FSET: FIFO pointer save bit

This bit saves the transmit FIFO read pointer.

If the read pointer value is saved before being transmitted and if the FLST bit is "0", the data can be re-transmitted even if a communication error or others have occurred.

If set to "1", the current read pointer value is saved.

If set to "0", it has no effect.

Value	Description	
	At writing	At reading
0	Not saved	"0" is always read.
1	FIFO2 is reset.	

<Note>

This bit can be set to "1" only when the transmission byte count (FBYTE) is "0".

[bit3] FCL2: FIFO2 reset bit

This bit resets the FIFO2 value.
If this bit is set to "1", the FIFO2 internal state is initialized.
Only the FCR1:FLST bit is initialized, and the other bits of FCR1/FCR0 registers are kept.

Value	Description	
	At writing	At reading
0	No effect on operation.	"0" is always read.
1	FIFO2 is reset.	

<Notes>

- Disable the transmit and reception first, and then reset FIFO2.
- Set the transmit FIFO interrupt enable bit to "0" before the execution.
- The valid data count of the FBYTE2 register is set to "0".

[bit2] FCL1: FIFO1 reset bit

This bit resets the FIFO1 state.
If this bit is set to "1", the FIFO1 internal state is initialized.
Only the FCR1:FLST bit is initialized, and the other bits of FCR1/FCR0 registers are kept.

Value	Description	
	At writing	At reading
0	No effect on operation.	"0" is always read.
1	FIFO1 is reset.	

<Notes>

- Disable the transmit and reception first, and then reset FIFO1.
- Set the transmit FIFO interrupt enable bit to "0" before the execution.
- The valid data count of the FBYTE1 register is set to "0".

[bit1] FE2: FIFO2 operation enable bit

This bit enables or disables the FIFO2 operation.

- To use the FIFO2 operation, set this bit to "1".
- If FIFO2 is set as transmit FIFO (FCR1:FSEL=1) and if data exists in FIFO2 when this bit is set to "1", the data transmission starts immediately when the UART is enabled to transmit data (SCR:TXE=1). During this time, set both SCR:TIE bit and SCR:TBIE bit to "0". Then, set this bit to "1" and set both SCR:TIE bit and SCR:TBIE bit to "1".
- If received FIFO is selected by the FSEL bit and if a received error has occurred, this bit is cleared to "0". This bit cannot be set to "1" until the received error is cleared.
- If FIFO2 is used as transmit FIFO, this bit must be set to "1" or "0" when the transmit buffer is empty (SSR:TDRE=1).
- If FIFO2 is used as received FIFO, this bit must be set to "0" when the received buffer is empty (SSR:RDRF=0) and no valid data exists in received FIFO (FBYTE2=0) after reception is disabled (SCR:RXE=0).
- If FIFO2 is used as received FIFO, this bit must be set to "1" when the received buffer is empty (SSR:RDRF=0) after reception is disabled (SCR:RXE=0).
- The FIFO2 state is held even if the FIFO2 operation is disabled.

Value	Description
0	Disables the FIFO2 operation.
1	Enables the FIFO2 operation.

[bit0] FE1: FIFO1 operation enable bit

This bit enables or disables the FIFO1 operation.

- To use the FIFO1 operation, set this bit to "1".
- When the FIFO1 is set as transmit FIFO (FCR1:FSEL=0) and if data exists in FIFO1 when this bit is set to "1", the data transmission starts immediately when the UART is set to enable data transmission (SCR:TXE=1). During this time, set both SCR:TIE bit and SCR:TBIE bit to "0". Then, set this bit to "1" and set both TIE bit and SCR:TBIE bit to "1".
- If received FIFO is selected by the FSEL bit and if a received error has occurred, this bit is cleared to "0". This bit cannot be set to "1" until the received error is cleared.
- If FIFO1 is used as transmit FIFO, this bit must be set to "1" or "0" when the transmit buffer is empty (SSR:TDRE=1).
- If FIFO1 is used as received FIFO, this bit must be set to "0" when the received buffer is empty (SSR:RDRF=0) and no valid data exists in received FIFO (FBYTE2=0) after reception is disabled (SCR:RXE=0).
- If FIFO1 is used as received FIFO, this bit must be set to "1" when the received buffer is empty (SSR:RDRF=0) after reception is disabled (SCR:RXE=0).
- The FIFO1 state is held even if the FIFO1 operation is disabled.

Value	Description
0	Disables the FIFO1 operation.
1	Enables the FIFO1 operation.

7.9. FIFO Byte Register (FBYTE)

The FIFO Byte Register (FBYTE) indicates the effective data count in the FIFO buffer. Also, this register can be used to generate a received interrupt when certain number of data sets are received in the received FIFO.

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	(FBYTE2)								(FBYTE1)							
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The FBYTE register indicates the effective data count of data written from or received in FIFO. The following shows the settings of the FCR1:FSEL bit.

Table 7-4 Display of data count

FSEL	FIFO selection	Data count display
0	FIFO2: Received FIFO, FIFO1:Transmit FIFO	FIFO2:FBYTE2, FIFO1:FBYTE1
1	FIFO2: Transmit FIFO, FIFO1:Received FIFO	FIFO2:FBYTE2, FIFO1:FBYTE1

- The initial value of data transfer count is "0x08" for the FBYTE register.
- Set a data count to flag a received interrupt for the FBYTE register of received FIFO. If this specified transfer count matches the FBYTE register display, the receive data full flag bit (RDRF) is set to "1".
- If both conditions below are satisfied and if the received idle state continues for more than 8 baud rate clocks, the receive data full flag bit (RDRF) is set to "1".
  - The received FIFO idle detection enable bit (FRIIE) is "1".
  - The number of data sets stored in the received FIFO does not reach the transfer count.

  If the RDR data is read during counting of 8 clocks, this counter is reset to "0", and counting for 8 clocks is restarted. If received FIFO is disabled, this counter is reset to zero (0). If data remains in the received FIFO and if received FIFO is enabled, the data counting is restarted.

[bit15:8] FBYTE2: FIFO2 data count display bits

[bit7:0] FBYTE1: FIFO1 data count display bits

At writing	Sets the transfer data count.
At reading	Reads the effective count of data.

Read (Effective data count)

During transmit: The number of data sets already written in the FIFO buffer but not transmitted yet

During reception: The number of data sets reception in FIFO

Write (Transfer data count)

During transmit: Set "0x00".

During reception: Set the data count to generate a received interrupt.

<Notes>

- Set "0x00" in the FBYTE register of transmit FIFO.
- Set a data value equal to or greater than "1" in the FBYTE register of received FIFO.
- This state can be changed only after the data reception has been disabled.
- A read-modify-write instruction cannot be used for this register.
- Any setting exceeding the FIFO capacity is inhibited.

CHAPTER 1-3: CSIO (Clock Synchronous Serial Interface)

This chapter explains the Clock Synchronous Serial Interface (CSIO) function that is supported in Operation mode 2.

1. Overview of CSIO (Clock Synchronous Serial Interface)
2. CSIO (Clock Synchronous Serial Interface) interrupts
3. CSIO (Clock Synchronous Serial Interface) operations
4. Dedicated baud rate generator
5. CSIO (Clock Synchronous Serial Interface) registers

CODE: 9BFCSIO-E02.0_FM15C-E05.4

1. Overview of CSIO (Clock Synchronous Serial Interface)

The CSIO is a general-purpose serial data communication interface (supporting the SPI) to allow synchronous communication with an external device. It also has transmit/received FIFO (up to 128 × 9 bits each) [*1]installed.

■ CSIO (Clock Synchronous Serial Interface) functions

		Function
1	Data buffer	· Full duplex double buffer (when FIFO is not used) · Transmit/Received FIFO (up to 128 × 9 bits each) [*1] (if FIFO is used)
2	Transfer system	· Clock synchronization (without start/stop bit) · Master/slave function · SPI supported (for both master and slave modes)
3	Baud rate	· Dedicate baud rate generator provided (configured with a 15-bit reload counter; in master mode operation) · An external clock can be entered (in the slave mode operation).
4	Data length	Variable from 5 bits to 9 bits
5	Received error detection	Overrun error
6	Interrupt request	· Received interrupt (a received completion, an overrun error) · Transmit interrupt (a transmit data empty, a transmit bus idle) · Transmit FIFO interrupt (when transmit FIFO is empty) · DMA(Transmit/Received) transferring support function are available.
7	Synchronous mode	Master or slave function
8	Pin access	The serial data output pin can be set to "1".
9	FIFO options	· FIFO for transmit/received installed (maximum capacity: 128 × 9 bits for transmit FIFO, 128 × 9 bits for received FIFO) [*] · Transmit FIFO or received FIFO can be selected. · Transmit data can be resent. · Received FIFO interrupt timing can be changed via software. · FIFO resetting is supported independently.

* : The FIFO capacity size varies depending on the product type.

2. CSIO (Clock Synchronous Serial Interface) Interrupts

The CSIO interrupts contain the received interrupt and the transmit interrupt. These interrupt requests can be generated if:

- A received data is set in the Received Data Register (RDR) or a data received error occurs.
- A transmit data is transferred from the Transmit Data Register (TDR) to the transmit shift register and the data transmission is started
- The transmit bus is idle (No data transmission occurs).
- A transmit FIFO data is requested.

■ CSIO interrupts

Table 2-1 shows the CSIO interrupt control bits and the interrupt factors.

Table 2-1 CSIO interrupt control bits and interrupt factors

Interrupt type	Interrupt request flag bit	Flag register	Interrupt factor	Interrupt factor enable bit	Operation to clear interrupt request flag
Reception	RDRF	SSR	A single-byte reception	SCR:RIE	Reading from the received data register (RDR)
			Reception of a data volume matching the value set for FBYTE.		Reading from the Received Data Register (RDR) until received FIFO is emptied
			The FRIIE bit is "1", received FIFO contains valid data, and the Received Idle state continues more than 8 bits time hours.		
	ORE	SSR	Overrun error		Setting the Received Error Flag Clear bit (SSR:REC) to "1"
Transmission	TDRE	SSR	The Transmit Data Register is empty.	SCR:TIE	Writing to the Transmit Data Register (TDR) or setting the transmit FIFO operation enable bit to "1" when the transmit FIFO operation enable bit is set to "0" and valid data are present in transmit FIFO (re-transmitting data) *
	TBI	SSR	No data transmission	SCR:TBIE	Writing to the Transmit Data Register (TDR) or setting the transmit FIFO operation enable bit to "1" when the transmit FIFO operation enable bit is set to "0" and valid data are present in transmit FIFO (re-transmitting data) *
	FDRQ	FCR1	Transmit FIFO is empty.	FCR1:FTIE	The FIFO transmit data request bit (FCR1:FDRQ) is set to "0" or transmit FIFO is full.

* : Set the TIE bit to "1" only after the TDRE bit has been set to "0".

2.1. Received interrupt and flag set timing

Data reception can be interrupted by a Received Completion (SSR:RDRF=1) or a Received Error Occurrence (SSR:ORE=1).

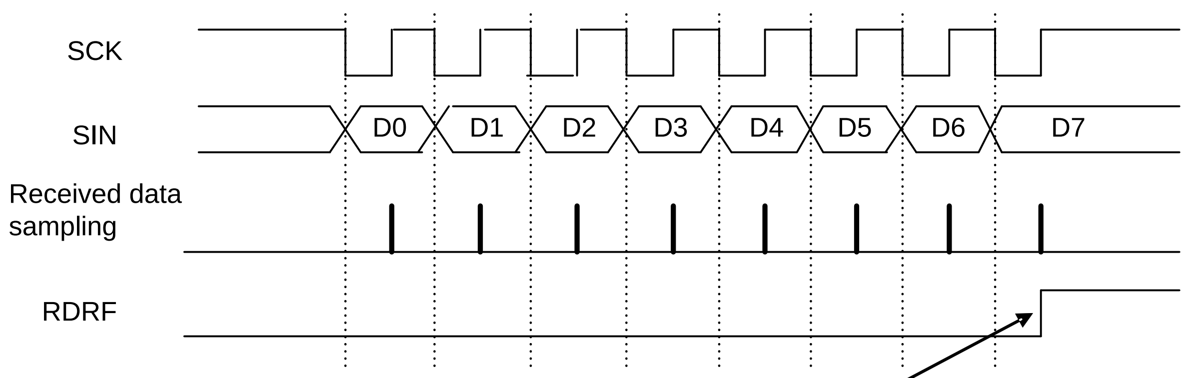
■ Received interrupt and flag set timing

When the last data bit is detected, the received data is stored in the Received Data Register (RDR). When the data reception is completed (SSR:RDRF=1) or when a data received error occurs (SSR:ORE=1), each flag is set. If a received interrupt is enabled (SCR:RIE=1) during this time, a received interrupt occurs.

<Note>

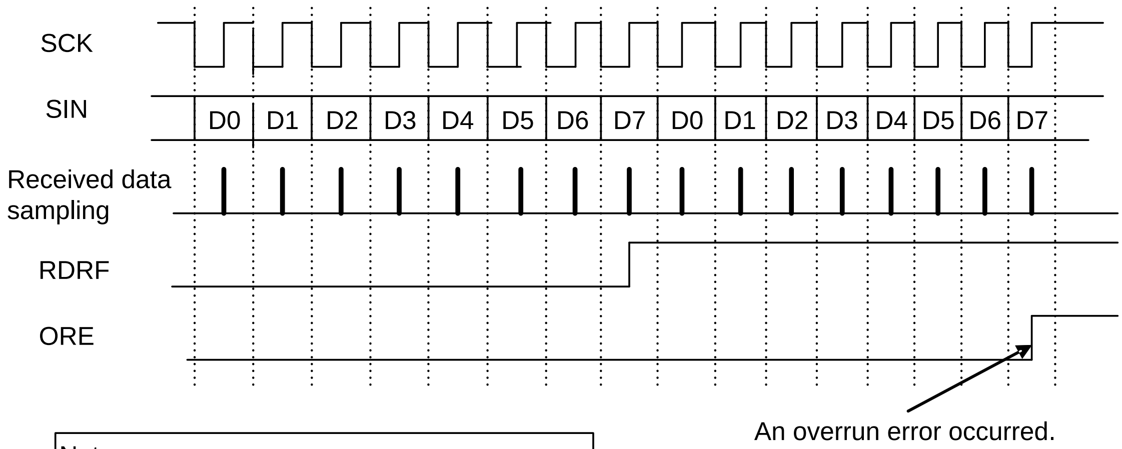
If a received error occurs, data in the Received Data Register (RDR) is invalidated.

Figure 2-1 Data receiving and flag set timing



Note:
This figure shows the signal timing under the following conditions.
SCR: MS=1, SPI=0
ESCR: L2 to L0=0b000
SMR:SCINV=0, BDS=0, SCKE=0, SOE=0

Figure 2-2 ORE (Overrun Error) flag set timing



Note:
This figure shows the signal timing under the following conditions.
SCR: MS=1, SPI=0
ESCR: L2 to L0=0b000
SMR:SCINV=0, BDS=0, SCKE=0, SOE=0

Note:
If the next data is transferred before the received data is read (RDRF=1), an overrun error occurs.

2.2. Interrupt and flag set timing when received FIFO is used

If received FIFO is used, an interrupt occurs when the FBYTE data (preset for the FBYTE register (FBYTE)) is received.

■ Received interrupt and flag set timing when received FIFO is used

If received FIFO is used, an interrupt occurs depending on the value set for the FBYTE register.

- When the amount of data set for transfer count in the FBYTE register is received, the received data full flag bit (SSR:RDRF) of the Serial Status Register is set to "1". If a received interrupt (SCR:RIE) is enabled during this time, a received interrupt occurs.
- If both all of the following conditions are satisfied and if the received idle state continues for more than 8 baud rate clocks, the received data full flag bit (RDRF) is set to "1".
  - The received FIFO idle detect enable bit (FRIIE) is "1".
  - The number of data sets stored in the received FIFO does not reach the transfer count.

  If the RDR data is read during counting of 8 clocks, this counter is reset to "0", and counting for 8 clocks is restarted. If received FIFO is disabled, this counter is reset to "0". If data remains in the received FIFO and if received FIFO is enabled, the data counting is restarted.
- When the received data (RDR) is all read and received FIFO is emptied, the received data full flag (SSR:RDRF) is cleared.
- If the display of the valid received data amount is the same as the FIFO capacity and if the next data is received, an overrun error (SSR:ORE=1) occurs.

Figure 2-3 Received interrupt occurrence timing when received FIFO is used

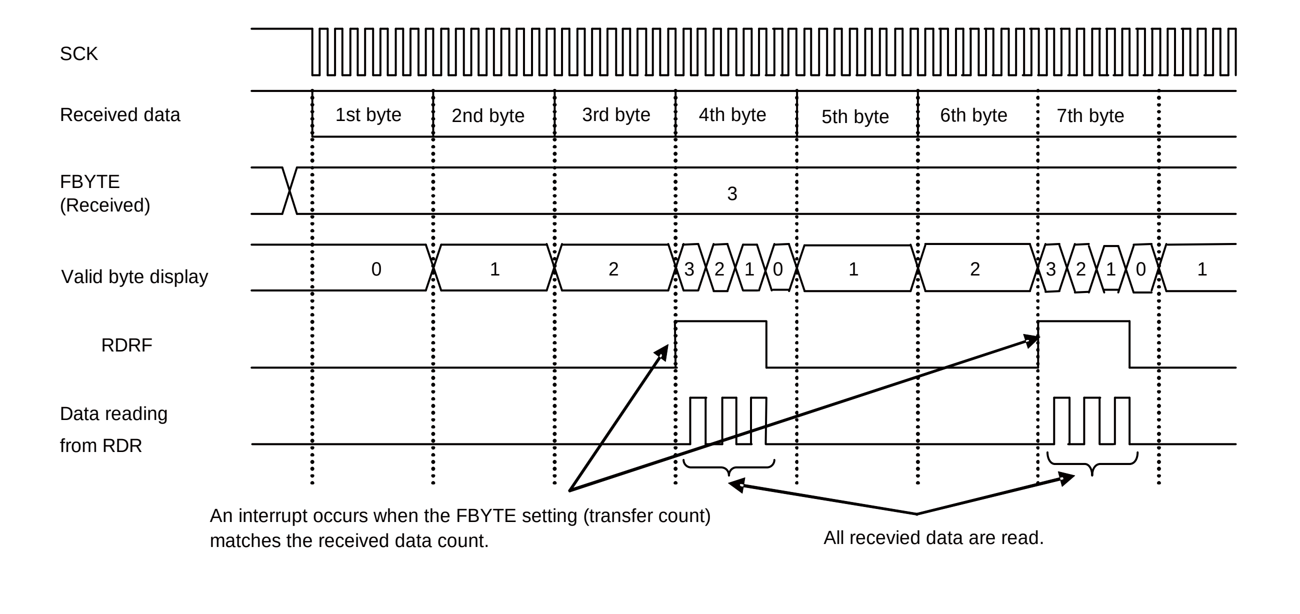
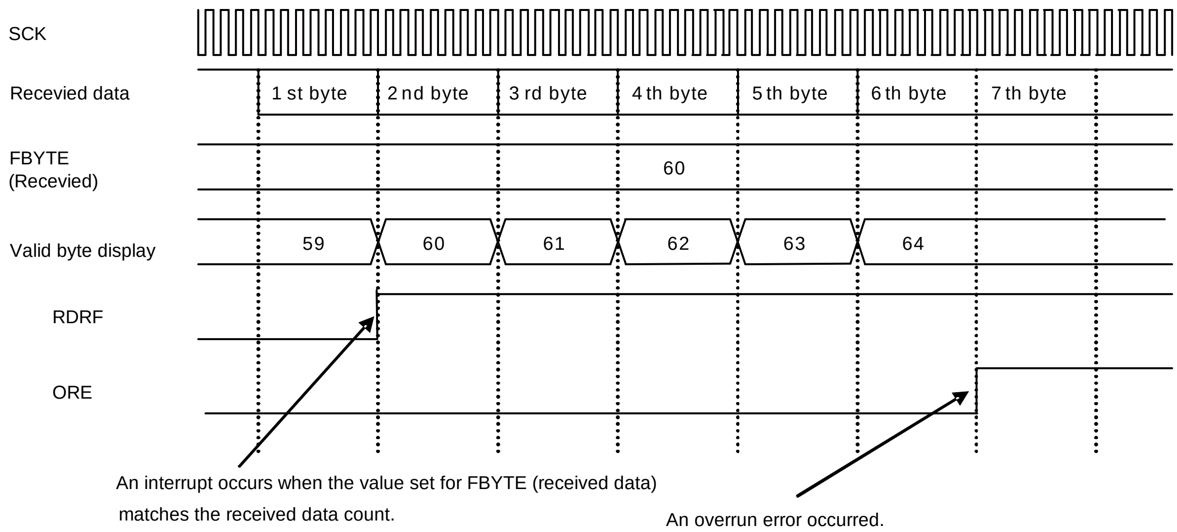


Figure 2-4 ORE (Overrun Error) flag bit set timing



Note:

If the FIFO buffer capcity is displayed by the FBYTE and if the next data is received, an overrun error occurs.

This figure shows acase where a 64-byte FIFO capacity is applied.

2.3. Transmit interrupt and flag set timing

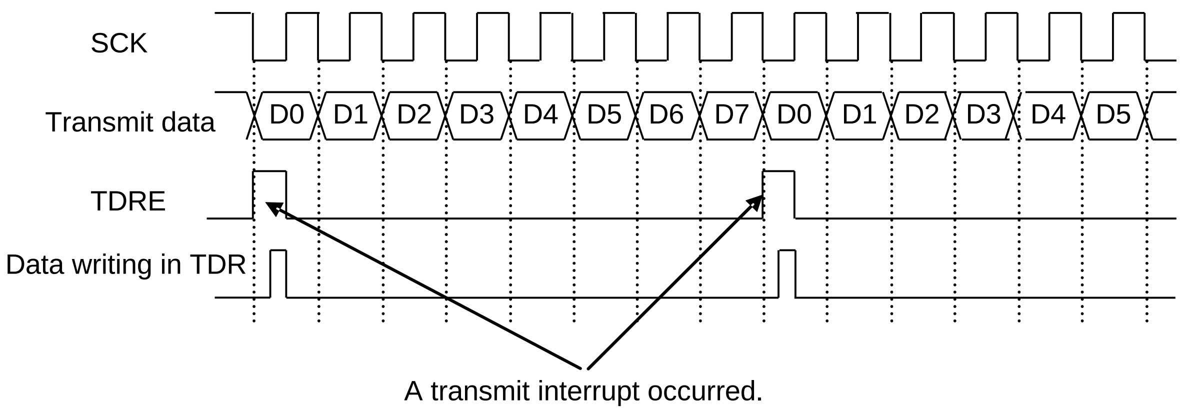
A transmit interrupt occurs if transmit data is transferred from the Transmit Data Register (TDR) to the transmit shift register (SSR:TDRE=1) and the data transmission is started, or if no data is transmitted (SSR:TBI=1).

■ Transmit interrupt and flag set timing

● Transmit data empty flag (SSR:TDRE) set timing

After data has been transferred from the Transmit Data Register (TDR) to the transmit shift register, the next data can be written in the TDR (SSR:TDRE=1). If a transmit interrupt is enabled (SCR:TIE=1) during this time, a transmit interrupt occurs. As the SSR:TDRE bit is read only, the SSR:TDRE bit is cleared to "0" when data is written to the Transmit Data Register (TDR).

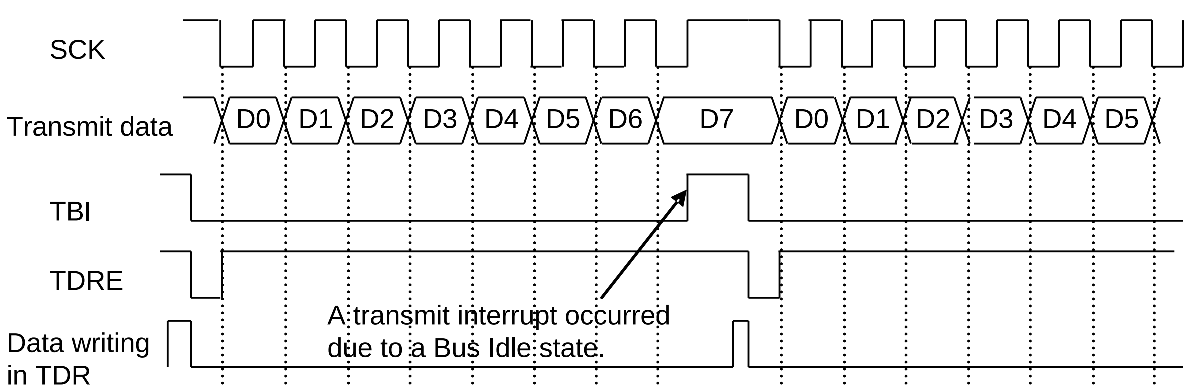
Figure 2-5 Transmit data empty flag (SSR:TDRE) set timing



● Transmit bus idle flag (SSR:TBI) set timing

If the Transmit Data Register is empty (SSR:TDRE=1) and no data is transmitted, the SSR:TBI bit is set to "1". If a transmit bus idle interrupt is enabled (SCR:TBIE=1) during this time, a transmit interrupt occurs. When transmit data is written to the Transmit Data Register (TDR), both the SSR:TBI bit and the transmit interrupt request are cleared.

Figure 2-6 Transmit bus idle flag (TBI) set timing



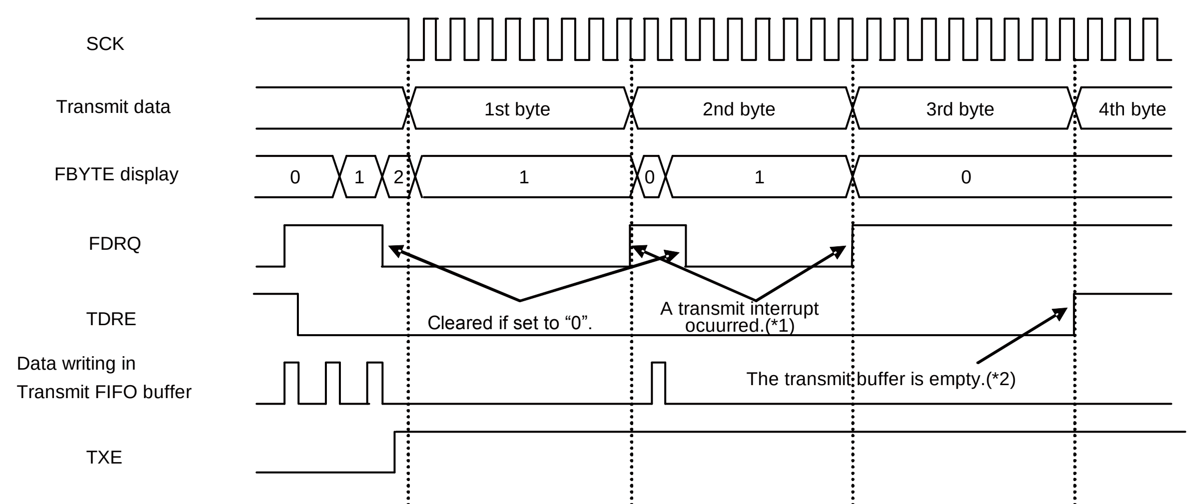
2.4. Interrupt and flag set timing when transmit FIFO is used

When transmit FIFO is used, an interrupt occurs if the buffer contains no data.

■ Transmit interrupt and flag set timing when transmit FIFO is used

- If transmit FIFO contains no data, the FIFO transmit data request bit (FCR1:FDRQ) is set to "1".
  If a FIFO transmit interrupt is enabled (FCR1:FTIE=1) during this time, a transmit interrupt occurs.
- If you have written the required data in transmit FIFO after occurrence of a transmit interrupt, clear the interrupt request by setting the FIFO transmit data request bit (FCR1:FDRQ) to "0".
- When transmit FIFO is filled with data, the FIFO transmit data request bit (FCR1:FDRQ) is set to "0".
- You can check a presence of data in transmit FIFO by reading the FIFO Byte Register (FBYTE).
  If FBYTE=0x00, no data exists in transmit FIFO.

Figure 2-7 Transmit interrupt occurrence timing when transmit FIFO is used



*1: The FDRQ bit is set to “1” as the Transmit FIFO buffer is empty.
*2: The TDRE bit is set to “1” as the Transmit Buffer register contains no data.

3. CSIO (Clock Synchronous Serial Interface) Operations

The clock synchronous data transfer is used.

3.1. Normal transfer (I)

■ Features

	Item	Description
1	Serial clock (SCK) signal mark level	"HIGH"
2	Transmit data output timing	SCK signal falling edge
3	Received data sampling	SCK signal rising edge
4	Data length	5 to 9 bits

■ Register settings

The register values required for normal transfer (I) are listed on the table below.

Table 3-1 Normal transfer (I) register settings

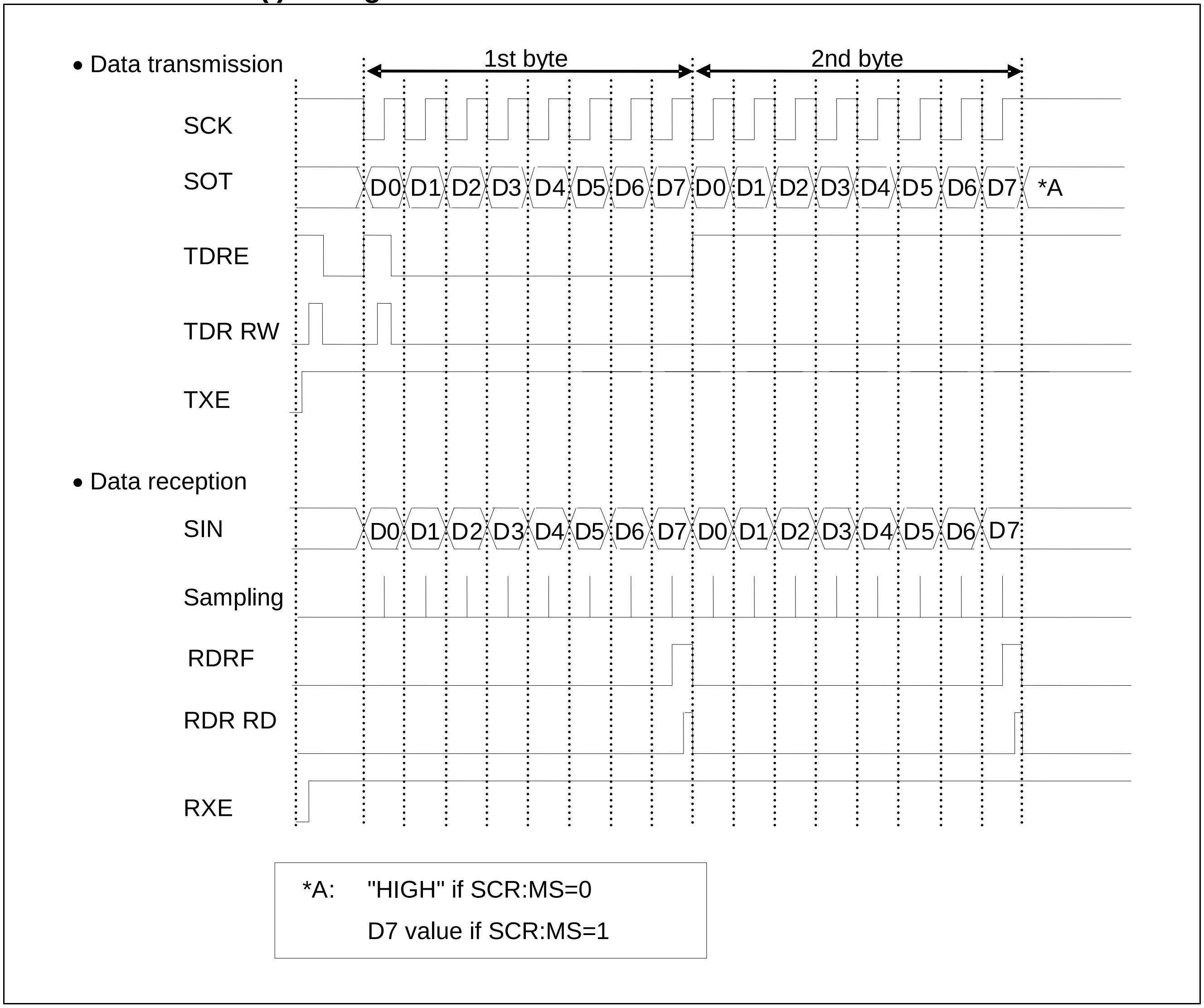
	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR/ SMR	UPCL	MS	SPI	RIE	TIE	TBIE	RXE	TXE	MD2	MD1	MD0	WUCR	SCINV	BDS	SCKE	SOE
	0	1/0	0	*	*	*	*	*	0	1	0	0	0	*	1/0	*
SSR/ ESCR	REC	-	-	-	ORE	RDRF	TDRE	TBI	SOP	-	-	WT1	WT0	L2	L1	L0
	0	-	-	-	-	-	-	-	0	-	-	*	*	*	*	*
TDR/ RDR								D8	D7	D6	D5	D4	D3	D2	D1	D0
								*	*	*	*	*	*	*	*	*
BGR1/ BGR 0	-	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
	-	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

1 : Set to "1".
0 : Set to "0".
* : User-dependent values

<Note>

The above bit setting (1/0) varies depending on the master or slave mode operation. Set as follows.
- During master mode operation: SCR:MS=0, SMR:SCKE=1
- During slave mode operation: SCR:MS=1, SMR:SCKE=0

■ Normal transfer (I) timing chart



■ Master mode operation (SCR:MS=0, SMR:SCKE=1)

● Data transmission

1. If serial data output is enabled (SMR:SOE=1), data transmission is enabled (SCR:TXE=1) and data reception is disabled (SCR:RXE=0), and when the transmit data is written in the TDR, the SSR:TDRE bit is set to "0". This causes the transmit data to be output in synchronization with a falling edge of the serial clock (SCK) output.
2. When the transmit data of the first bit is output, the SSR:TDRE bit is set to "1". Therefore, if the transmit interrupt is enabled (SCR:TIE=1), a transmit interrupt request is output. During this time, the transmit data of the 2nd byte can be written in the register.

● Data reception

1. If the serial data output is disabled (SMR:SOE=0), data transmission is enabled (SCR:TXE=1) and data reception is enabled (SCR:RXE=1), and when a dummy data is written in the TDR, the received data is sampled at a rising edge of serial clock (SCK) output.
2. When the last bit is received, the SSR:RDRF bit is set to "1". If a received interrupt is enabled (SCR:RIE=1) during this time, a received interrupt request is output.
   The received data (RDR) can be read during this time.
3. When the received data (RDR) is read, the SSR:RDRF bit is cleared to "0".

<Notes>

- To perform data reception only, write a dummy data in the TDR so that the serial clock (SCK) is output.
- If the FIFO transmission and reception are enabled, the serial clocks (SCK) for the preset number of frames are output when the frames to be transferred are set in the FBYTE register.

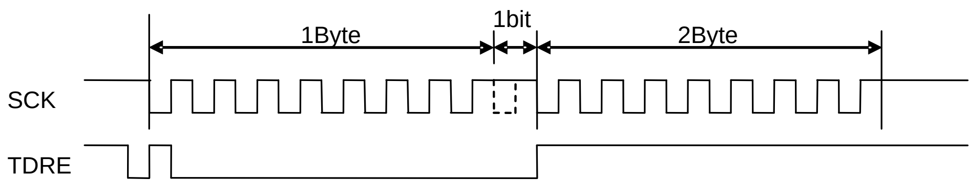
● Data transmission and reception

1. To perform data transmission and reception simultaneously, enable the serial data output (SMR:SOE=1) and enable the data transmission and reception (SCR:TXE, RXE=1).
2. When the transmit data is written in the TDR, the SSR:TDRE bit is set to "0" and the transmit data is output in synchronization with a falling edge of the serial clock (SCK) output. When the transmit data of the first bit is output, the SSR:TDRE bit is set to "1". If a transmit interrupt is enabled (SCR:TIE=1), a transmit interrupt request is output. During this time, the transmit data of the 2nd byte can be written in the register.
3. The received data is sampled at a rising edge of the serial clock (SCK) output. When the last bit of received data is received, the SSR:RDRF bit is set to "1". If a received interrupt is enabled (SCR:RIE=1), a received interrupt request is output. The received data (RDR) can be read during this time. When the received data is read, the SSR:RDRF bit is cleared to "0".

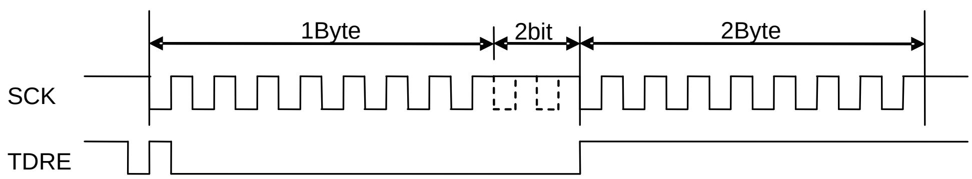
● Continuous data transmit or reception waiting

If anything other than ESCR:WT1, ESCR:WT0=00 is set for the continuous data transmission or reception, a wait is inserted between frames.

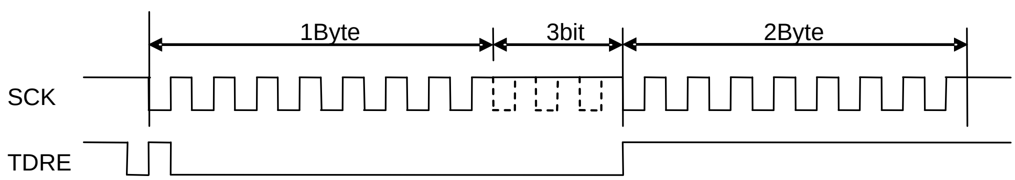
- ESCR:WT1, ESCR:WT0=01 (in master mode operation)



- ESCR:WT1, ESCR:WT0=10 (in master mode operation)



- ESCR:WT1, ESCR:WT0=11 (in master mode operation)



■ Slave mode operation (SCR:MS=1, SMR:SCKE=0)

● Data transmission

1. If serial data output is enabled (SMR:SOE=1) and data transmission is enabled (SCR:TXE=1) and when the transmit data is written in the TDR, the SSR:TDRE bit is set to "0". This causes the transmit data to be output in synchronization with a falling edge of the serial clock (SCK) input.
2. When the transmit data of the first bit is output, the SSR:TDRE bit is set to "1". If a transmit interrupt is enabled (SCR:TIE=1), a transmit interrupt request is output. During this time, the transmit data of the 2nd byte can be written in the register.

● Data reception

1. If the serial data output is disabled (SMR:SOE=0) and data reception is enabled (SCR:RXE=1), the received data is sampled at a rising edge of serial clock (SCK) input.
2. When the last bit is received, the SSR:RDRF bit is set to "1". If a received interrupt is enabled (SCR:RIE=1), a received interrupt request is output.
   The received data (RDR) can be read during this time.
3. When the received data (RDR) is read, the SSR:RDRF bit is cleared to "0".

● Data transmission and reception

1. To perform data transmission and reception simultaneously, enable the serial data output (SMR:SOE=1) and enable the data transmission and reception (SCR:TXE, RXE=1).
2. When the transmit data is written in the TDR, the SSR:TDRE bit is set to "0" and the transmit data is output in synchronization with a falling edge of the serial clock (SCK) input. When the transmit data of the first bit is output, the SSR:TDRE bit is set to "1". If a transmit interrupt is enabled (SCR:TIE=1), a transmit interrupt request is output. During this time, the transmit data of the 2nd byte can be written in the register.
3. The received data is sampled at a rising edge of the serial clock (SCK) input. When the last bit of received data is received, the SSR:RDRF bit is set to "1". If the received interrupt is enabled (SCR:RIE=1), a received interrupt request is output. The received data (RDR) can be read during this time. When the received data is read, the SSR:RDRF bit is cleared to "0".

3.2. Normal transfer (II)

■ Features

	Item	Description
1	Serial clock (SCK) signal mark level	"LOW"
2	Transmit data output timing	SCK signal rising edge
3	Received data sampling	SCK signal falling edge
4	Data length	5 to 9 bits

■ Register settings

The register values required for normal transfer (II) are listed on the table below.

Table 3-2 Normal transfer (II) register settings

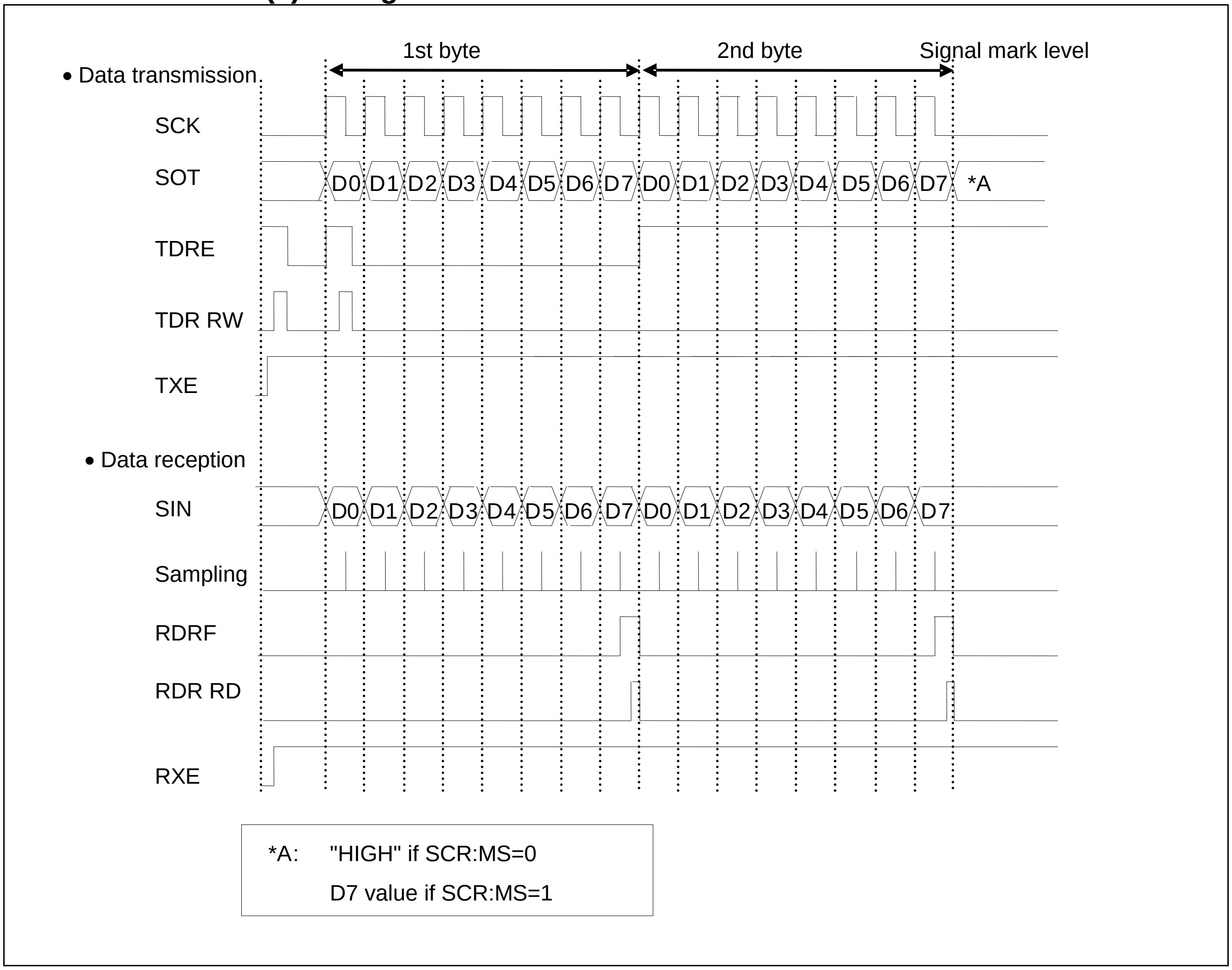
	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR/ SMR	UPCL	MS	SPI	RIE	TIE	TBIE	RXE	TXE	MD2	MD1	MD0	WUCR	SCINV	BDS	SCKE	SOE
	0	1/0	0	*	*	*	*	*	0	1	0	0	1	*	1/0	*
SSR/ ESCR	REC	-	-	-	ORE	RDRF	TDRE	TBI	SOP	-	-	WT1	WT0	L2	L1	L0
	0	-	-	-	-	-	-	-	0	-	-	*	*	*	*	*
TDR/ RDR								D8	D7	D6	D5	D4	D3	D2	D1	D0
								*	*	*	*	*	*	*	*	*
BGR1/ BGR0	-	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
	-	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

1 : Set to "1".
0 : Set to "0".
* : User-dependent values

<Note>

The above bit setting (1/0) varies depending on the master or slave mode operation. Set as follows.
- During master mode operation: SCR:MS=0, SMR:SCKE=1
- During slave mode operation: SCR:MS=1, SMR:SCKE=0

■ Normal transfer (II) timing chart



■ Master mode operation (SCR:MS=0, SMR:SCKE=1)

● Data transmission

1. If serial data output is enabled (SMR:SOE=1), data transmission is enabled (SCR:TXE=1) and data reception is disabled (SCR:RXE=0), and when the transmit data is written in the TDR, the SSR:TDRE bit is set to "0". This causes the transmit data to be output in synchronization with a rising edge of the serial clock (SCK) output.
2. When the transmit data of the first bit is output, the SSR:TDRE bit is set to "1". Therefore, if the transmit interrupt is enabled (SCR:TIE=1), a transmit interrupt request is output. During this time, the transmit data of the 2nd byte can be written in the register.

● Data reception

1. If the serial data output is disabled (SMR:SOE=0), data transmission is enabled (SCR:TXE=1) and data reception is enabled (SCR:RXE=1), and when a dummy data is written in the TDR, the received data is sampled at a falling edge of serial clock (SCK) output.
2. When the last bit is received, the SSR:RDRF bit is set to "1". If a received interrupt is enabled (SCR:RIE=1) during this time, a received interrupt request is output.
   The received data (RDR) can be read during this time.
3. When the received data (RDR) is read, the SSR:RDRF bit is cleared to "0".

<Notes>

· To perform data reception only, write a dummy data in the TDR so that the serial clock (SCK) is output.

· If the FIFO transmission and reception are enabled, the serial clocks (SCK) for the preset number of frames are output when the frames to be transferred are set in the FBYTE register.

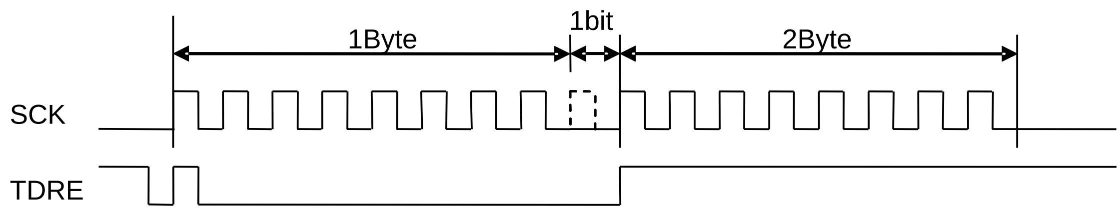
● Data transmission and reception

1. To perform data transmission and reception simultaneously, enable the serial data output (SMR:SOE=1) and enable the data transmission and reception (SCR:TXE, RXE=1).
2. When the transmit data is written in the TDR, the SSR:TDRE bit is set to "0" and the transmit data is output in synchronization with a rising edge of the serial clock (SCK) output. When the transmit data of the first bit is output, the SSR:TDRE bit is set to "1". If a transmit interrupt is enabled (SCR:TIE=1), a transmit interrupt request is output. During this time, the transmit data of the 2nd byte can be written in the register.
3. The received data is sampled at a falling edge of the serial clock (SCK) output. When the last bit of received data is received, the SSR:RDRF bit is set to "1". If a received interrupt is enabled (SCR:RIE=1), a received interrupt request is output. The received data (RDR) can be read during this time. When the received data is read, the SSR:RDRF bit is cleared to "0".

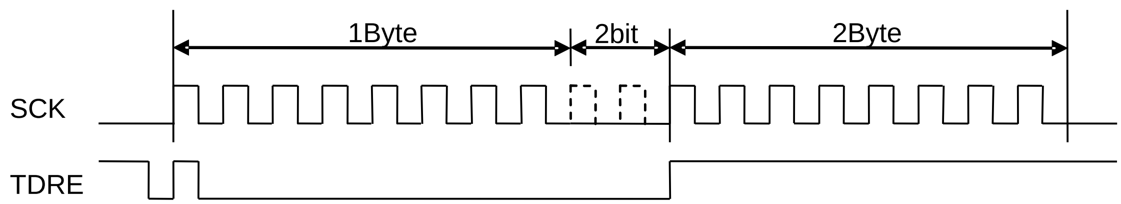
● Continuous data transmit or reception waiting

If anything other than ESCR:WT1, ESCR:WT0=00 is set for the continuous data transmission or reception, a wait is inserted between frames.

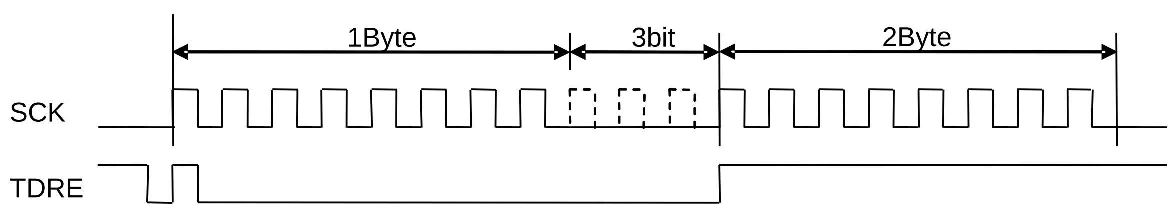
- ESCR:WT1, ESCR:WT0=01 (in master mode operation)



- ESCR:WT1, ESCR:WT0=10 (in master mode operation)



- ESCR:WT1, ESCR:WT0=11 (in master mode operation)



■ Slave mode operation (SCR:MS=1, SMR:SCKE=0)

● Data transmission

1. If serial data output is enabled (SMR:SOE=1) and data transmission is enabled (SCR:TXE=1) and when the transmit data is written in the TDR, the SSR:TDRE bit is set to "0". This causes the transmit data to be output in synchronization with a rising edge of the serial clock (SCK) input.
2. When the transmit data of the first bit is output, the SSR:TDRE bit is set to "1". If a transmit interrupt is enabled (SCR:TIE=1), a transmit interrupt request is output. During this time, the transmit data of the 2nd byte can be written in the register.

● Data reception

1. If the serial data output is disabled (SMR:SOE=0) and data reception is enabled (SCR:RXE=1), the received data is sampled at a falling edge of serial clock (SCK) input.
2. When the last bit is received, the SSR:RDRF bit is set to "1". If a received interrupt is enabled (SCR:RIE=1), a received interrupt request is output.
   The received data (RDR) can be read during this time.
3. When the received data (RDR) is read, the SSR:RDRF bit is cleared to "0".

● Data transmission and reception

1. To perform data transmission and reception simultaneously, enable the serial data output (SMR:SOE=1) and enable the data transmission and reception (SCR:TXE, RXE=1).
2. When the transmit data is written in the TDR, the SSR:TDRE bit is set to "0" and the transmit data is output in synchronization with a rising edge of the serial clock (SCK) input. When the transmit data of the first bit is output, the SSR:TDRE bit is set to "1". If a transmit interrupt is enabled (SCR:TIE=1), a transmit interrupt request is output. During this time, the transmit data of the 2nd byte can be written in the register.
3. The received data is sampled at a falling edge of the serial clock (SCK) input. When the last bit of received data is received, the SSR:RDRF bit is set to "1". If the received interrupt is enabled (SCR:RIE=1), a received interrupt request is output. The received data (RDR) can be read during this time. When the received data is read, the SSR:RDRF bit is cleared to "0".

3.3. SPI transfer (I)

■ Features

	Item	Description
1	Serial clock (SCK) signal mark level	"HIGH"
2	Transmit data output timing	SCK signal rising edge
3	Received data sampling	SCK signal falling edge
4	Data length	5 to 9 bits

■ Register settings

The register values required for SPI transfer (I) are listed on the table below.

Table 3-3 SPI transfer (I) register settings

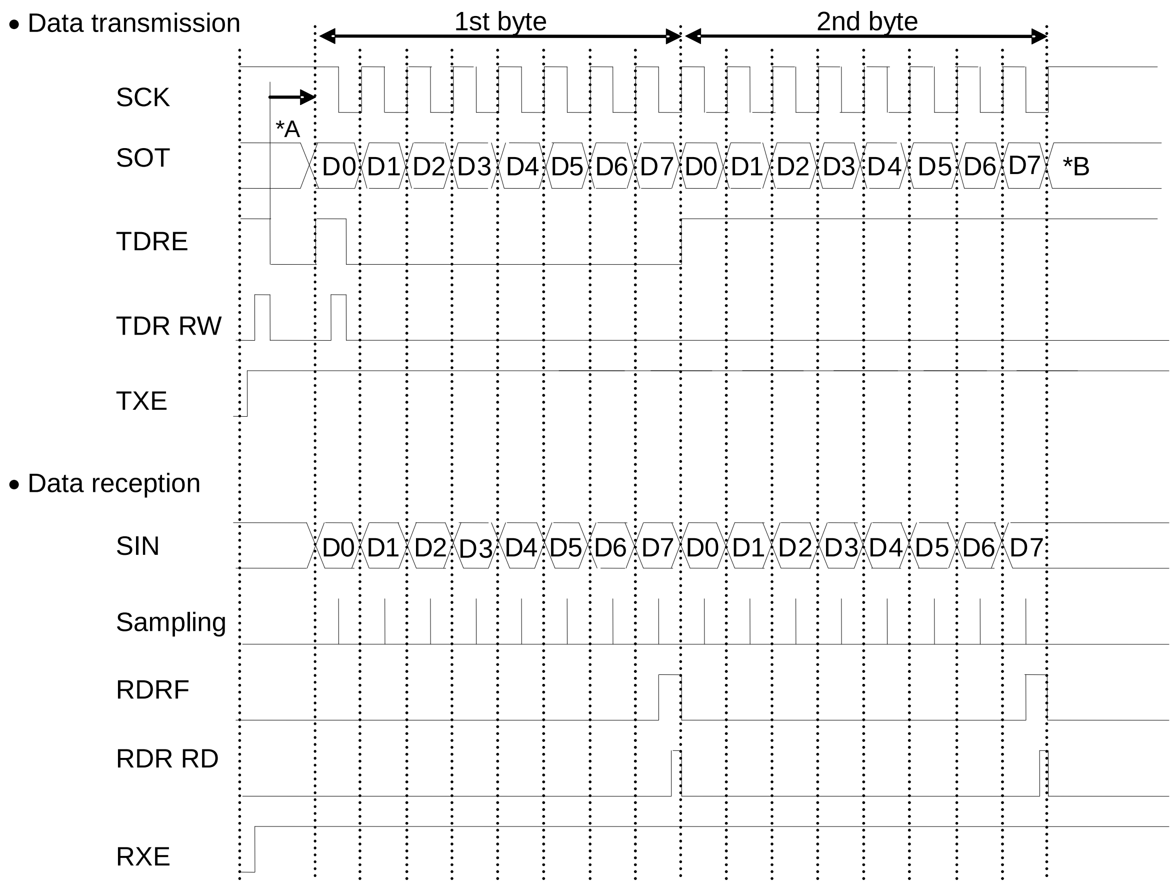
	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR/ SMR	UPCL	MS	SPI	RIE	TIE	TBIE	RXE	TXE	MD2	MD1	MD0	WUCR	SCINV	BDS	SCKE	SOE
	0	1/0	1	*	*	*	*	*	0	1	0	0	0	*	1/0	*
SSR/ ESCR	REC	-	-	-	ORE	RDRF	TDRE	TBI	SOP	-	-	WT1	WT0	L2	L1	L0
	0	-	-	-	-	-	-	-	0	-	-	*	*	*	*	*
TDR/ RDR								D8	D7	D6	D5	D4	D3	D2	D1	D0
								*	*	*	*	*	*	*	*	*
BGR1/ BGR0	-	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
	-	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

1 : Set to "1".
0 : Set to "0".
* : User-dependent values

<Note>

The above bit setting (1/0) varies depending on the master or slave mode operation. Set as follows.
- During master mode operation: SCR:MS=0, SMR:SCKE=1
- During slave mode operation: SCR:MS=1, SMR:SCKE=0

■ SPI transfer (I) timing chart



*A: During slave mode transmission (MS=1, SCKE=0, SOE=1), 4 machine cycles or more time is required after writing data in the TDR

*B: "HIGH" if SCR:MS=0

"D0" of the 3rd byte if SCR:MS=1 and TDRE is "LOW"

"HIGH" if SCR:MS=1 and TDRE is "HIGH"

■ Master mode operation (SCR:MS=0, SMR:SCKE=1)

● Data transmission

1. If serial data output is enabled (SMR:SOE=1), data transmission is enabled (SCR:TXE=1) and data reception is disabled (SCR:RXE=0), and when the transmit data is written in the TDR, the SSR:TDRE bit is set to "0". This causes the first bit to output. Then, the transmit data is output in synchronization with a rising edge of the serial clock (SCK) output.
2. The SSR:TDRE bit is set to "1" before a half cycle of a falling edge of serial clock (SCK) output. Therefore, if the transmit interrupt is enabled (SCR:TIE=1), a transmit interrupt request is output. During this time, the transmit data of the 2nd byte can be written in the register.

● Data reception

1. If the serial data output is disabled (SMR:SOE=0), data transmission is enabled (SCR:TXE=1) and data reception is enabled (SCR:RXE=1), and when a dummy data is written in the TDR, the received data is sampled at a falling edge of serial clock (SCK) output.
2. When the last bit is received, the SSR:RDRF bit is set to "1". If a received interrupt is enabled (SCR:RIE=1) during this time, a received interrupt request is output.
   The received data (RDR) can be read during this time.
3. When the received data (RDR) is read, the SSR:RDRF bit is cleared to "0".

<Notes>

- To perform data reception only, write a dummy data in the TDR so that the serial clock (SCK) is output.
- If the FIFO transmission and reception are enabled, the serial clocks (SCK) for the preset number of frames are output when the frames to be transferred are set in the FBYTE register.

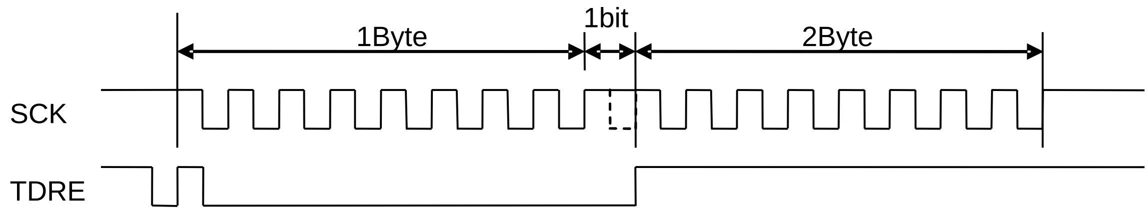
● Data transmission and reception

1. To perform data transmission and reception simultaneously, enable the serial data output (SMR:SOE=1) and enable the data transmission and reception (SCR:TXE, RXE=1).
2. When the transmit data is written in the TDR, the SSR:TDRE is set to "0" and the first bit is output. Then, the transmit data is output in synchronization with a rising edge of the serial clock (SCK) output. The SSR:TDRE bit is set to "1" before a half cycle of a falling edge of the first serial clock. If a transmit interrupt is enabled (SCR:TIE=1), a transmit interrupt request is output. During this time, the transmit data of the 2nd byte can be written in the register.
3. The received data is sampled at a falling edge of the serial clock (SCK) output. When the last bit of received data is received, the SSR:RDRF bit is set to "1". If a received interrupt is enabled (SCR:RIE=1), a received interrupt request is output. The received data (RDR) can be read during this time. When the received data is read, the SSR:RDRF bit is cleared to "0".

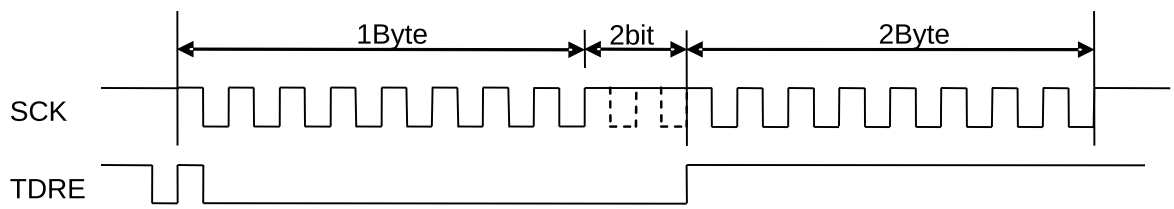
● Continuous data transmit or reception waiting

If anything other than ESCR:WT1, ESCR:WT0=00 is set for the continuous data transmission or reception, a wait is inserted between frames.

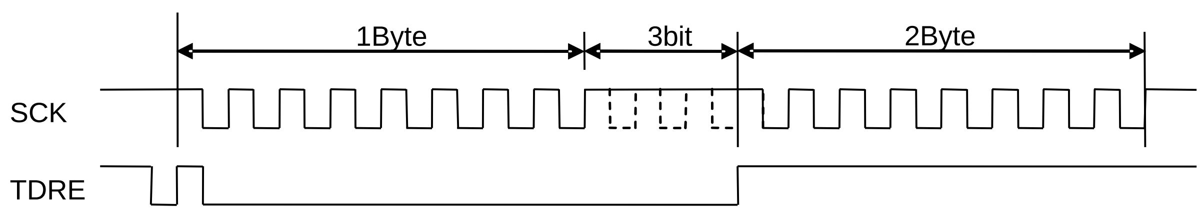
- ESCR:WT1, ESCR:WT0=01 (in master mode operation)



- ESCR:WT1, ESCR:WT0=10 (in master mode operation)



- ESCR:WT1, ESCR:WT0=11 (in master mode operation)



■ Slave mode operation (SCR:MS=1, SMR:SCKE=0)

● Data transmission

1. If serial data output is enabled (SMR:SOE=1) and data transmission is enabled (SCR:TXE=1) and when the transmit data is written in the TDR, the SSR:TDRE bit is set to "0". This causes the first bit to output. Then, the transmit data is output in synchronization with a rising edge of the serial clock (SCK) output.
2. When the first bit of transmit data is output, the SSR:TDRE bit is set to "1". If a transmit interrupt is enabled (SCR:TIE=1), a transmit interrupt request is output. During this time, the transmit data of the 2nd byte can be written in the register.

<Note>

If data transmission is enabled (SCR:TXE=1) and if the first transmit data is written in the TDR at a time other than the serial clock (SCK) signal mark level, the first data bit is not output and the data transmission may fail. After the data transmission is enabled (SCR:TXE=1), the first transmit data must be written in the TDR at a signal mark level of the serial clock (SCK).

● Data reception

1. If the serial data output is disabled (SMR:SOE=0) and data reception is enabled (SCR:RXE=1), the received data is sampled at a falling edge of serial clock (SCK) input.
2. When the last bit is received, the SSR:RDRF bit is set to "1". If a received interrupt is enabled (SCR:RIE=1), a received interrupt request is output.
   The received data (RDR) can be read during this time.
3. When the received data (RDR) is read, the SSR:RDRF bit is cleared to "0".

● Data transmission and reception

1. To perform data transmission and reception simultaneously, enable the serial data output (SMR:SOE=1) and enable the data transmission and reception (SCR:TXE, RXE=1).
2. When the transmit data is written in the TDR, the SSR:TDRE is set to "0" and the first bit is output. Then, the transmit data is output in synchronization with a rising edge of the serial clock (SCK) input. When the first bit of transmit data is output, the SSR:TDRE bit is set to "1". If a transmit interrupt is enabled (SCR:TIE=1), a transmit interrupt request is output. During this time, the transmit data of the 2nd byte can be written in the register.
3. The received data is sampled at a falling edge of the serial clock (SCK) input. When the last bit of received data is received, the SSR:RDRF bit is set to "1". If the received interrupt is enabled (SCR:RIE=1), a received interrupt request is output. The received data (RDR) can be read during this time. When the received data is read, the SSR:RDRF bit is cleared to "0".

● Continuous switching from data reception to transmission

1. Disable the serial data output (SMR:SOE=0), enable a received interrupt (SCR:RIE=1), enable data reception (SCR:RXE=1), and enable data transmission (SCR:TXE=1). If dummy data is written in the TDR at a signal mark level of serial clock (SCK), the received data is sampled at a falling edge of serial clock (SCK) input.
2. To continue data reception, write a dummy data in the TDR between the time when a received interrupt is requested and when the next serial clock (SCK) rises.
3. To switch the data reception to the data transmission, enable the serial data output (SMR:SOE=1), disable a received interrupt (SCR:RIE=0), and disable data reception (SCR:RXE=0) between the time when a received interrupt is requested and when the next serial clock (SCK) rises. Also, output the transmit data in synchronization with a rising edge of serial clock after the transmit data has been written in the TDR and the data reception has completed.

3.4. SPI transfer (II)

■ Features

	Item	Description
1	Serial clock (SCK) signal mark level	"LOW"
2	Transmit data output timing	SCK signal falling edge
3	Received data sampling	SCK signal rising edge
4	Data length	5 to 9 bits

■ Register settings

The register values required for SPI transfer (II) are listed on the table below.

Table 3-4 SPI transfer (II) register settings

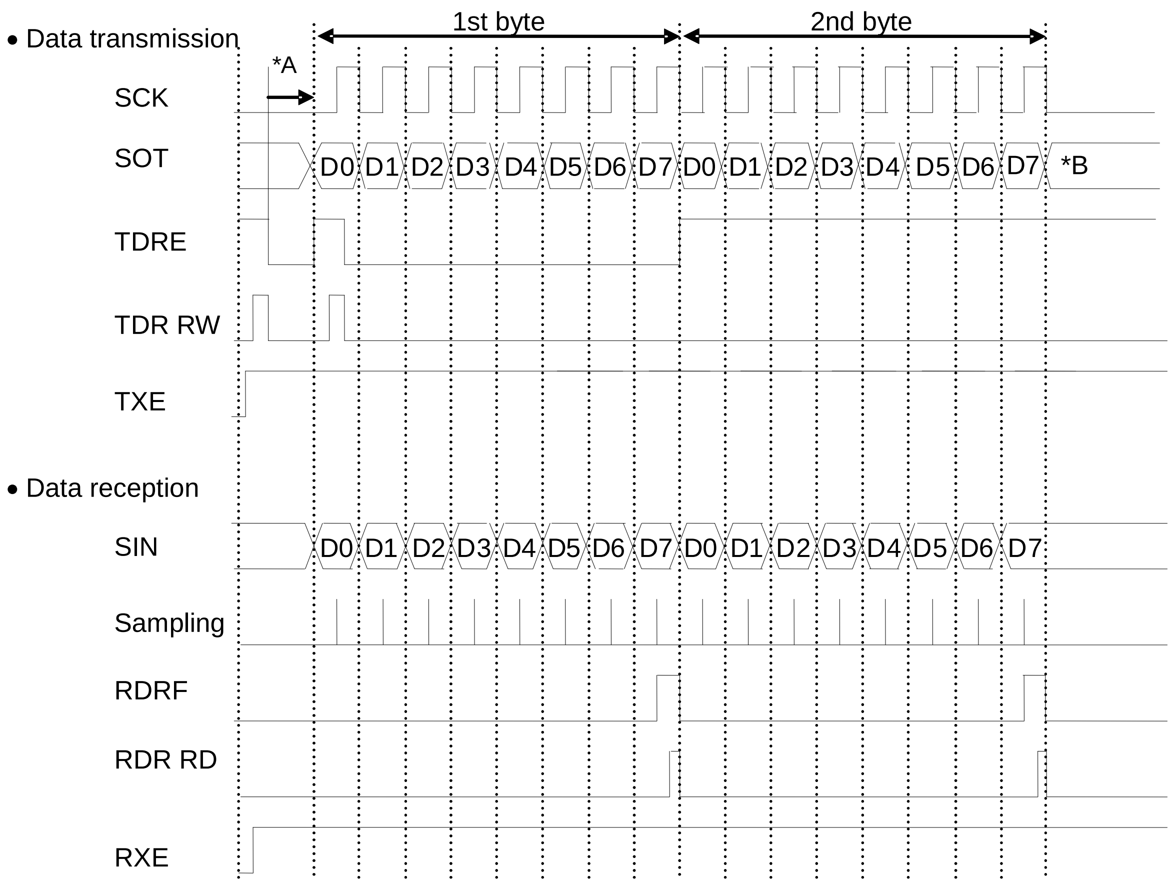
	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR/ SMR	UPCL	MS	SPI	RIE	TIE	TBIE	RXE	TXE	MD2	MD1	MD0	WUCR	SCINV	BDS	SCKE	SOE
	0	1/0	1	*	*	*	*	*	0	1	0	0	1	*	1/0	*
SSR/ ESCR	REC	-	-	-	ORE	RDRF	TDRE	TBI	SOP	-	-	WT1	WT0	L2	L1	L0
	0	-	-	-	-	-	-	-	0	-	-	*	*	*	*	*
TDR/ RDR								D8	D7	D6	D5	D4	D3	D2	D1	D0
								*	*	*	*	*	*	*	*	*
BGR1/ BGR0	-	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
	-	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

1 : Set to "1".
0 : Set to "0".
* : User-dependent values

<Note>

The above bit setting (1/0) varies depending on the master or slave mode operation. Set as follows.
- During master mode operation: SCR:MS=0, SMR:SCKE=1
- During slave mode operation: SCR:MS=1, SMR:SCKE=0

■ SPI transfer (II) timing chart



*A: During slave mode transmission (MS=1, SCKE=0, SOE=1), 4 machine cycles or more time is required after writing data in the TDR

*B: "HIGH" if SCR:MS=0
"D0" of the 3rd byte if SCR:MS=1 and TDRE is "LOW"
"HIGH" if SCR:MS=1 and TDRE is "HIGH"

■ Master mode operation (SCR:MS=0, SMR:SCKE=1)

● Data transmission

1. If serial data output is enabled (SMR:SOE=1), data transmission is enabled (SCR:TXE=1) and data reception is disabled (SCR:RXE=0), and when the transmit data is written in the TDR, the SSR:TDRE bit is set to "0". This causes the transmit data to be output in synchronization with a falling edge of the serial clock (SCK) output.
2. The SSR:TDRE bit is set to "1" before a half cycle of a rising edge of the first serial clock (SCK) output. Therefore, if the transmit interrupt is enabled (SCR:TIE=1), a transmit interrupt request is output. During this time, the transmit data of the 2nd byte can be written in the register.

● Data reception

1. If the serial data output is disabled (SMR:SOE=0), data transmission is enabled (SCR:TXE=1) and data reception is enabled (SCR:RXE=1), and when a dummy data is written in the TDR, the received data is sampled at a rising edge of serial clock (SCK) output.
2. When the last bit is received, the SSR:RDRF bit is set to "1". If a received interrupt is enabled (SCR:RIE=1) during this time, a received interrupt request is output.
   The received data (RDR) can be read during this time.
3. When the received data (RDR) is read, the SSR:RDRF bit is cleared to "0".

<Notes>

- To perform data reception only, write a dummy data in the TDR so that the serial clock (SCK) is output.
- If the FIFO transmission and reception are enabled, the serial clocks (SCK) for the preset number of frames are output when the frames to be transferred are set in the FBYTE register.

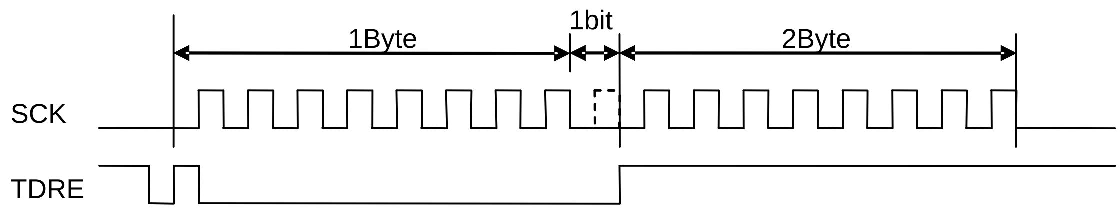
● Data transmission and reception

1. To perform data transmission and reception simultaneously, enable the serial data output (SMR:SOE=1) and enable the data transmission and reception (SCR:TXE, RXE=1).
2. When the transmit data is written in the TDR, the SSR:TDRE is set to "0" and the first bit is output. Then, the transmit data is output in synchronization with a falling edge of the serial clock (SCK) output. The SSR:TDRE bit is set to "1" before a half cycle of a rising edge of the first serial clock. If a transmit interrupt is enabled (SCR:TIE=1), a transmit interrupt request is output. During this time, the transmit data of the 2nd byte can be written in the register.
3. The received data is sampled at a rising edge of the serial clock (SCK) output. When the last bit of received data is received, the SSR:RDRF bit is set to "1". If a received interrupt is enabled (SCR:RIE=1), a received interrupt request is output. The received data (RDR) can be read during this time. When the received data is read, the SSR:RDRF bit is cleared to "0".

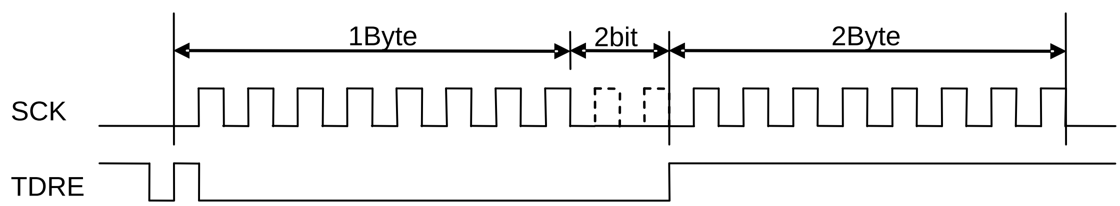
● Continuous data transmit or reception waiting

If anything other than ESCR:WT1, ESCR:WT0=00 is set for the continuous data transmission or reception, a wait is inserted between frames.

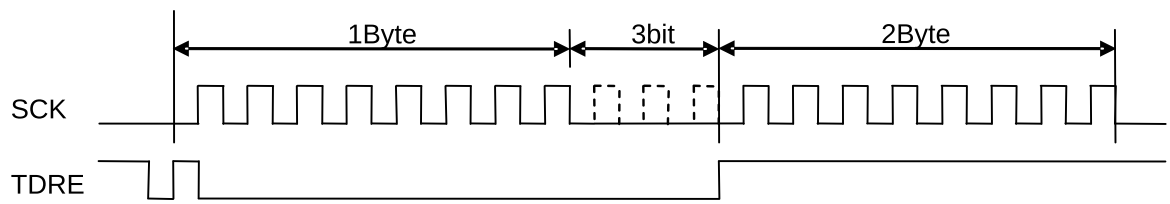
- ESCR:WT1, ESCR:WT0=01 (in master mode operation)



- ESCR:WT1, ESCR:WT0=10 (in master mode operation)



- ESCR:WT1, ESCR:WT0=11 (in master mode operation)



■ Slave mode operation (SCR:MS=1, SMR:SCKE=0)

● Data transmission

1. If serial data output is enabled (SMR:SOE=1) and data transmission is enabled (SCR:TXE=1) and when the transmit data is written in the TDR, the SSR:TDRE bit is set to "0". This causes the first bit to output. Then, the transmit data is output in synchronization with a falling edge of the serial clock (SCK) input.
2. When the first bit of transmit data is output, the SSR:TDRE bit is set to "1". If a transmit interrupt is enabled (SCR:TIE=1), a transmit interrupt request is output. During this time, the transmit data of the 2nd byte can be written in the register.

<Note>

If data transmission is enabled (SCR:TXE=1) and if the first transmit data is written in the TDR at a time other than the serial clock (SCK) signal mark level, the first data bit is not output and the data transmission may fail. After the data transmission is enabled (SCR:TXE=1), the first transmit data must be written in the TDR at a signal mark level of the serial clock (SCK).

● Data reception

1. If the serial data output is disabled (SMR:SOE=0) and data reception is enabled (SCR:RXE=1), the received data is sampled at a rising edge of serial clock (SCK) input.
2. When the last bit is received, the SSR:RDRF bit is set to "1". If a received interrupt is enabled (SCR:RIE=1), a received interrupt request is output.
   The received data (RDR) can be read during this time.
3. When the received data (RDR) is read, the SSR:RDRF bit is cleared to "0".

● Data transmission and reception

1. To perform data transmission and reception simultaneously, enable the serial data output (SMR:SOE=1) and enable the data transmission and reception (SCR:TXE, RXE=1).
2. When the transmit data is written in the TDR, the SSR:TDRE is set to "0" and the first bit is output. Then, the transmit data is output in synchronization with a falling edge of the serial clock (SCK) input. When the first bit of transmit data is output, the SSR:TDRE bit is set to "1". If a transmit interrupt is enabled (SCR:TIE=1), a transmit interrupt request is output. During this time, the transmit data of the 2nd byte can be written in the register.
3. The received data is sampled at a rising edge of the serial clock (SCK) input. When the last bit of received data is received, the SSR:RDRF bit is set to "1". If the received interrupt is enabled (SCR:RIE=1), a received interrupt request is output. The received data (RDR) can be read during this time. When the received data is read, the SSR:RDRF bit is cleared to "0".

● Continuous switching from data reception to transmission

1. Disable the serial data output (SMR:SOE=0), enable a received interrupt (SCR:RIE=1), enable data reception (SCR:RXE=1), and enable data transmission (SCR:TXE=1). If dummy data is written in the TDR at a signal mark level of serial clock (SCK), the received data is sampled at a falling edge of serial clock (SCK) input.
2. To continue data reception, write a dummy data in the TDR between the time when a received interrupt is requested and when the next serial clock (SCK) rises.
3. To switch the data reception to the data transmission, enable the serial data output (SMR:SOE=1), disable a received interrupt (SCR:RIE=0), and disable data reception (SCR:RXE=0) between the time when a received interrupt is requested and when the next serial clock (SCK) rises. Also, output the transmit data in synchronization with a rising edge of serial clock after the transmit data has been written in the TDR and the data reception has completed.

4. Dedicated Baud Rate Generator

The dedicated baud rate generator functions in the master mode operation only. However, if received FIFO is used, set the dedicated baud rate generator in the slave mode operation, too.

■ CSIO (Clock Synchronous Serial Interface) baud rate selection

The dedicated baud rate generator settings vary depending on the master or slave mode operation.

[1] During master mode operation

● Divide the internal clock frequency using the dedicated baud rate generator, and select a baud rate.

- This generator provides two internal reload counters, which support transmitting and receiving serial clocks respectively. To select the baud rate, specify the 15-bit reload value using Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0).
- The internal clock frequency is divided by the reload counter set value.

[2] During slave mode operation

- The dedicated baud rate generator does not function in the slave mode operation (SCR:MS=1).
  (An external clock, entered from the SCK clock input pin, is used directly.)

<Note>

If received FIFO is used, set the dedicated baud rate generator even in the slave mode operation.

4.1. Baud rate settings

This section explains how to set the baud rate. Also, the calculation result of serial clock frequency is shown.

■ Calculating the baud rate

Two 15-bit reload counters are set using the Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0).
The baud rate is obtained in the following formulas.

(1) Reload value

$V = \phi / b - 1$

V : Reload value; b : Baud rate; ϕ: Bus clock frequency

(2) Calculation example

To set the 16 MHz bus clock, use the internal clock, and set the 19200 bps baud rate, set the reload value as follows:

Reload value:

$V = (16 \times 1000000) / 19200 - 1 = 832$

Therefore, the baud rate is:

$b = (16 \times 1000000) / (832 + 1) = 19208$ bps

(3) Baud rate error

The baud rate error can be calculated by the following equation.

Error (%) = (Calculated value – Target value) / Target value 100

Example: To set the 20 MHz bus clock and 153600 bps target baud rate:

Reload value =:(20 × 1000000) / 1536009 - 1=129

Buad rate (Calculated value) = (20 × 1000000) / (129 + 1) = 153846 (bps)

Error (%) = (153846 - 153600) / 153600 × 100= 0.16 (%):

<Notes>

- If the reload value is set to "0", the reload counter is stopped.
- If the reload value is even, the "HIGH" and "LOW" width of serial clock changes as follows, depending on SMR:SCIN bit and SCR:SPI bit settings. If the value is odd, the serial clock has the same "HIGH" and "LOW" signal width.
  - When in normal transfer (SCR:SPI=0) and the mark level of the serial clock is "HIGH" (SMR:SCINV=0), or when in SPI transfer (SCR:SPI=1) and the mark level of the serial clock is "LOW" (SMR:SCINV=1), the "HIGH" width of serial clock is longer for 1 cycle of bus clock.
  - When in normal transfer (SCR:SPI=0) and the mark level of the serial clock is "LOW" (SMR:SCINV=1), or when in SPI transfer (SCR:SPI=1) and the mark level of the serial clock is "HIGH" (SMR:SCINV=0), the "LOW" width of serial clock is longer for 1 cycle of bus clock.
- Set the reload value to 3 or more.
- For the allowable baud rate range, consider the effect of a jitter of the clock input to a macro.

■ Reload values and baud rate setting examples for each bus clock frequency

The following shows the reload values and baud rate setting examples.

Table 4-1 Reload values and baud rate setting examples

Baud rate (bps)	8 MHz		10 MHz		16 MHz		20 MHz		24 MHz	
	Value	ERR	Value	ERR	Value	ERR	Value	ERR	Value	ERR
8M	-	-	-	-	-	-	-	-	-	-
6M	-	-	-	-	-	-	-	-	3	0
5M	-	-	-	-	-	-	3	0	-	-
4M	-	-	-	-	3	0	4	0	5	0
2.5M	-	-	3	0	-	-	7	0	-	-
2M	3	0	4	0	7	0	9	0	11	0
1M	7	0	9	0	15	0	19	0	23	0
500000	15	0	19	0	31	0	39	0	47	0
460800	-	-	-	-	-	-	-	-	51	0.16
250000	31	0	39	0	63	0	79	0	95	0
230400	-	-	-	-	-	-	86	-0.22	103	0.16
153600	51	0.16	64	0.16	103	0.16	129	0.16	155	0.16
125000	63	0	79	0	127	0	159	0	191	0
115200	-	-	86	-0.22	138	-0.08	173	-0.22	207	0.16
76800	103	0.16	129	0.16	207	0.16	259	0.16	312	-0.16
57600	138	-0.08	173	-0.22	277	-0.08	346	0.06	416	-0.08
38400	207	0.16	259	0.16	416	-0.08	520	-0.03	624	0
28800	277	-0.08	346	<0.01	555	-0.08	693	0.06	832	0.03
19200	416	-0.08	520	-0.03	832	0.03	1041	-0.03	1249	0
10417	767	<0.01	959	<0.01	1535	<0.01	1919	<0.01	2303	<0.01
9600	832	0.04	1041	-0.03	1666	-0.02	2082	0.02	2499	0
7200	1110	<0.01	1388	<0.01	2221	<0.01	2777	<0.01	3332	<0.01
4800	1666	-0.02	2082	0.02	3332	<0.01	4166	<0.01	4999	0
2400	3332	<0.01	4166	<0.01	6666	<0.01	8332	<0.01	9999	0
1200	6666	<0.01	8332	<0.01	13332	<0.01	16666	<0.01	19999	0
600	13332	<0.01	16666	<0.01	26666	<0.01	-	-	-	-
300	26666	<0.01	-	-	-	-	-	-	-	-

- Value: BGR1/0 register set value
- ERR: Baud rate error (%)

Table 4-2 Reload values and baud rate setting examples (continued)

Baud rate (bps)	32MHz		40 MHz		48 MHz		72 MHz	
	Value	ERR	Value	ERR	Value	ERR	Value	ERR
8M	3	0	4	0	5	0	8	0
6M	-	-	-	-	7	0	11	0
5M	-	-	7	0	-	-	-	-
4M	7	0	9	0	11	0	17	0
2.5M	-	-	15	0	-	-	-	-
2M	15	0	19	0	23	0	35	0
1M	31	0	39	0	47	0	71	0
500000	63	0	79	0	95	0	143	0
460800	-	-	86	-0.22	103	0.16	155	0.16
250000	127	0	159	0	191	0	287	0
230400	-	-	173	-0.22	207	0.16	312	-0.16
153600	207	-0.16	259	0.16	312	-0.16	468	-0.05
125000	255	0	319	0	383	0	575	0
115200	277	0.08	346	0.06	416	-0.08	624	0
76800	416	0.08	520	-0.03	624	0	937	-0.05
57600	555	0.08	693	0.06	832	0.04	1249	0
38400	832	-0.04	1041	-0.03	1249	0	1874	0
28800	1110	-0.01	1388	<0.01	1666	-0.02	2499	0
19200	1666	0.02	2082	0.02	2499	0	3749	0
10417	3071	<0.01	3839	<0.01	4607	<0.01	6911	<0.01
9600	3332	-0.01	4166	<0.01	4999	0	7499	0
7200	4443	-0.01	5555	<0.01	6666	<0.01	9999	0
4800	6666	<0.01	8332	<0.01	9999	0	14999	0
2400	13332	<-0.01	16666	<0.01	19999	0	29999	0
1200	26666	<0.01	-	-	-	-	-	-
600	-	-	-	-	-	-	-	-
300	-	-	-	-	-	-	-	-

■ Functions of reload counter

There are two types of reload counter: the transmit reload counter and the received reload counter. They function as the dedicated baud rate generators. Each reload counter consists of a 15-bit register for the reload value, and generates transmitting and receiving clocks from internal clocks.

■ Starting counting

When the reload value is written to the Baud Rate Generator Register (BGR1 or BGR0), the reload counter starts counting.

■ Restarting

The reload counter restarts counting in the following conditions.

● Common to transmit and received reload counters

A programmable reset (SCR:UPCL bit)

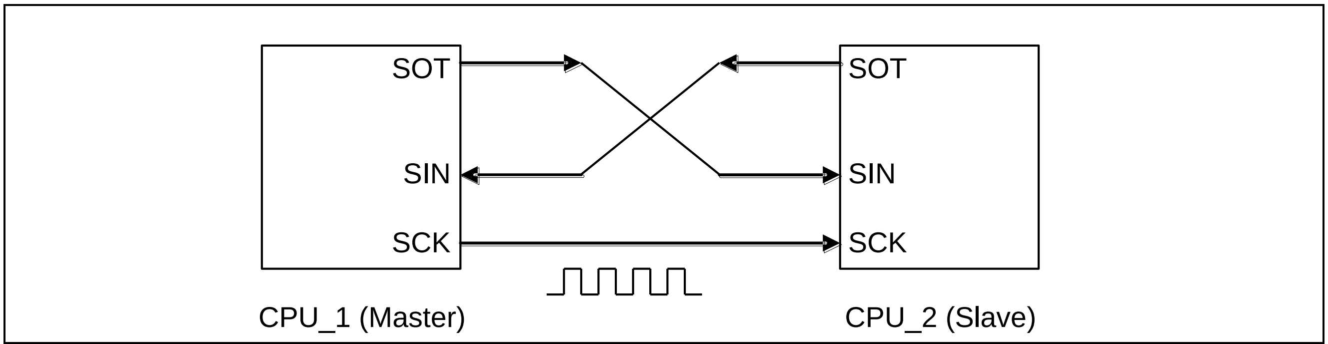
4.2. CSIO (Clock Synchronous Serial Interface) setup procedure and program flow

The CSIO (Clock Synchronous Serial Interface) allows bidirectional and synchronous serial data transmission.

● CPU-to-CPU connection

Select the bidirectional communication for the CSIO (Clock Synchronous Serial Interface). Connect two CPUs to each other as shown in Figure 4-1.

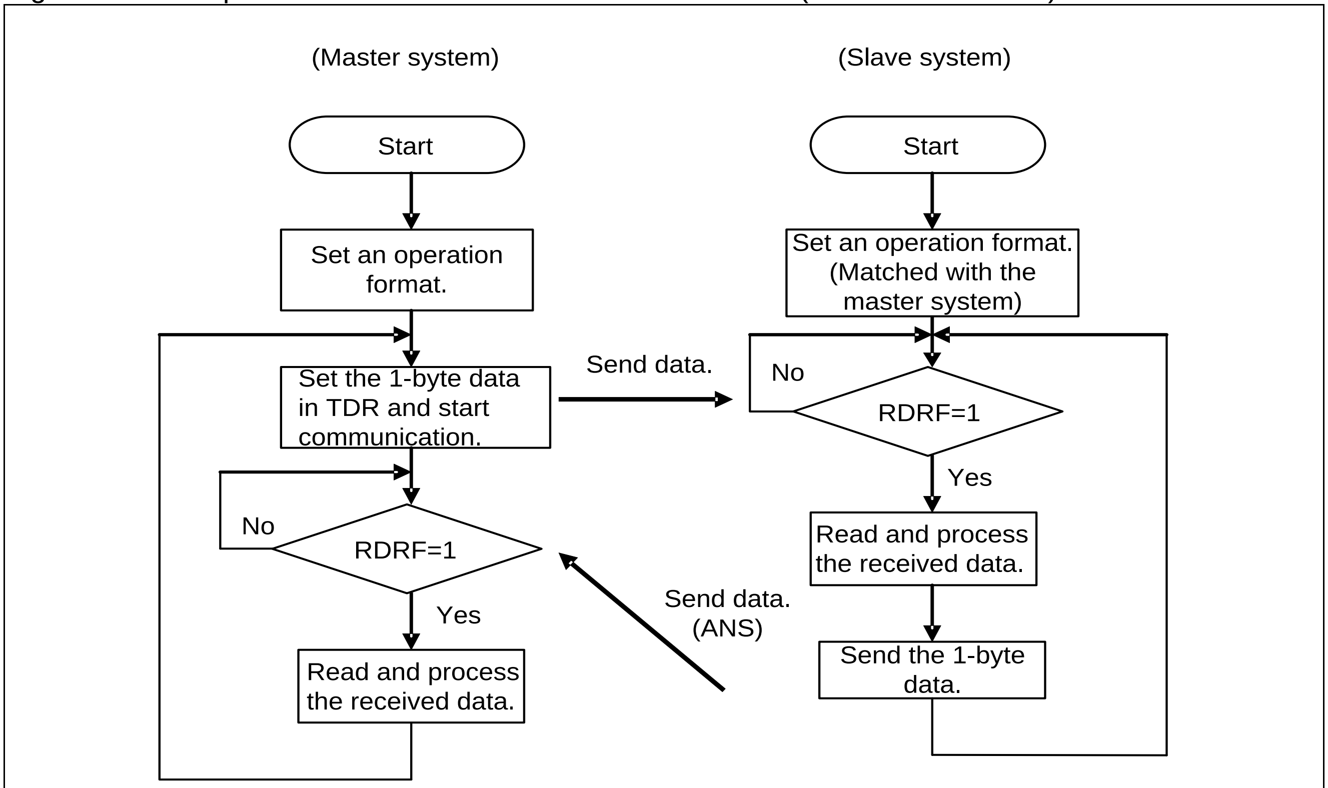
Figure 4-1 Connection example for CSIO (Clock Synchronous Serial Interface) bidirectional communication



■ Flowcharts

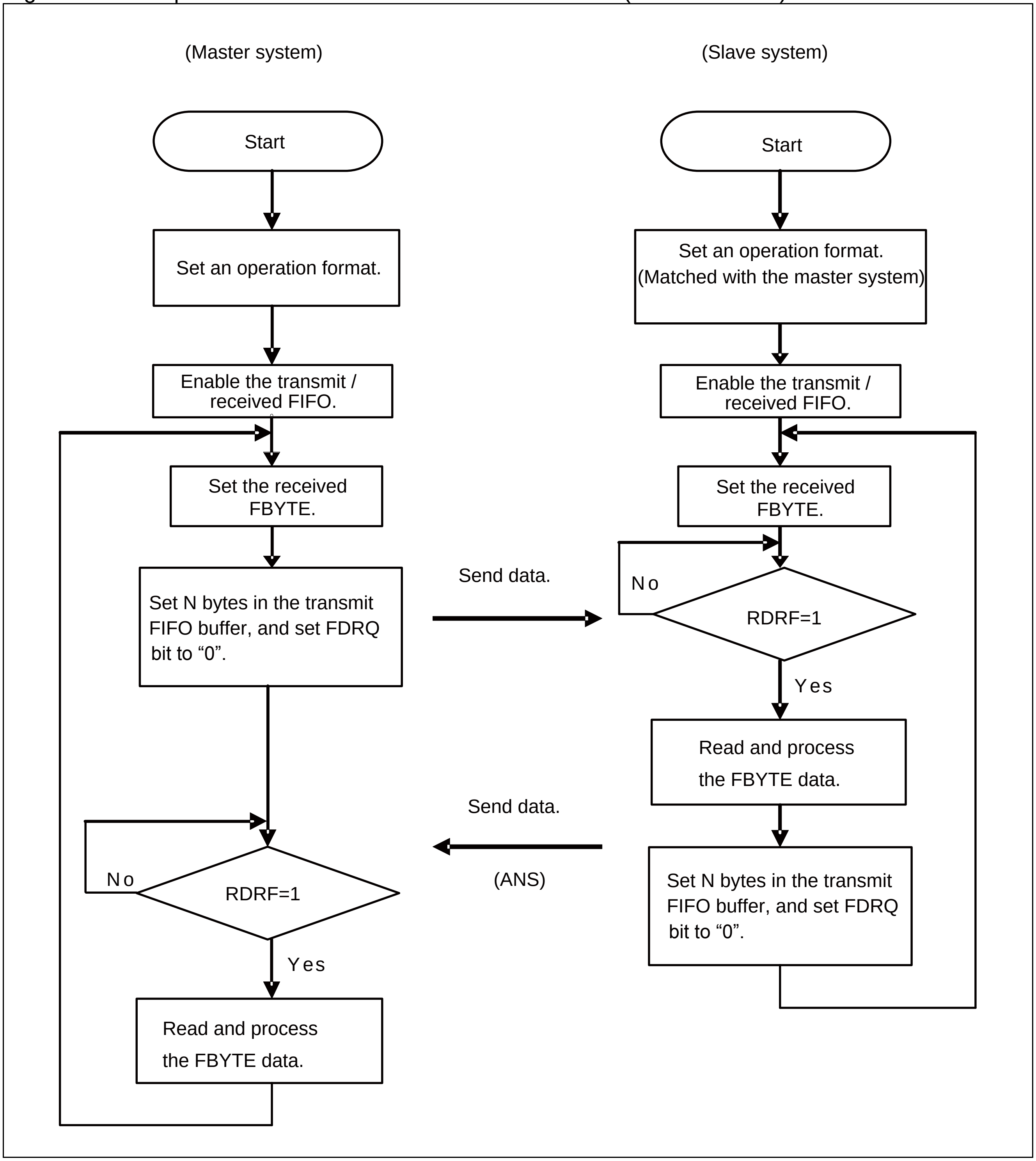
● If FIFO is not used

Figure 4-2 Example of bidirectional communication flowchart (if FIFO is not used)



● If FIFO is used

Figure 4-3 Example of bidirectional communication flowchart (if FIFO is used)



5. CSIO (Clock Synchronous Serial Interface) Registers

This section provides a list of CSIO (Clock Synchronous Serial Interface) registers.

■ CSIO (Clock Synchronous Serial Interface) register list

Table 5-1 CSIO (Clock Synchronous Serial Interface) register list

	bit15 bit8	bit7 bit0
CSIO	SCR (Serial Control Register)	SMR (Serial Mode Register)
	SSR (Serial Status Register)	ESCR (Extended Communication Control Register)
	RDR/TDR (Transmit/Received Data register)	
	BGR1 (Baud Rate Generator Register 1)	BGR0 (Baud Rate Generator Register 0)
FIFO	FCR1 (FIFO Control Register 1)	FCR0 (FIFO Control Register 0)
	FBYTE2 (FIFO2 Byte Register)	FBYTE1 (FIFO1 Byte Register)

Table 5-2 CSIO (Clock Synchronous Serial Interface) bit assignment

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR/ SMR	UPCL	MS	SPI	RIE	TIE	TBIE	RXE	TXE	MD2	MD1	MD0	-	SCINV	BDS	SCKE	SOE
SSR/ ESCR	REC	-	-	-	ORE	RDRF	TDRE	TBI	SOP	-	-	WT1	WT0	L2	L1	L0
TDR/ RDR	-							D8	D7	D6	D5	D4	D3	D2	D1	D0
BGR1/ BGR0	-	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
FCR1/ FCR0	-	-	-	FLSTE	FRIIE	FDRQ	FTIE	FSEL	-	FLST	FLD	FSET	FCL2	FCL1	FE2	FE1
FBYTE2/ FBYTE1	FD15	FD14	FD13	FD12	FD11	FD10	FD9	FD8	FD7	FD6	FD5	FD4	FD3	FD2	FD1	FD0

5.1. Serial Control Register (SCR)

The Serial Control Register (SCR) is used to enable/disable a transmit/received interrupt, enable/disable a transmit idle interrupt, and enable/disable data transmission and reception. Also, the register can set the SPI connection and reset the CSIO settings.

bit	15	14	13	12	11	10	9	8	7 ... 0
Field	UPCL	MS	SPI	RIE	TIE	TBIE	RXE	TXE	(SMR)
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

[bit15] UPCL: Programmable clear bit

Initializes the CSIO internal state.

If set to "1":

- The CSIO is reset directly (software reset). However, the current register settings are kept. The transmit or received state is disconnected immediately.
- The baud rate generator reloads the BGR1/0 register value and restarts operation.
- All of transmit/received interrupt factors (SSR:TDRE, TBI, RDRF, ORE) are initialized.

If set to "0":

No effect on the operation.

"0" is always read from this bit.

Value	Description	
	At writing	At reading
0	No effect on the operation.	"0" is always read.
1	Programmable clear	

<Notes>

- Disable an interrupt first, and then execute the programmable clear instruction.
- If the FIFO operation is used, disable it (FCR0:FE[2:1]=00) first and then execute the programmable clear instruction.

[bit14] MS: Master/Slave function select bit

Selects the master or slave mode.

Value	Description
0	Master mode
1	Slave mode

<Notes>

- If the slave mode is selected and if SMR:SCKE=0, the external clock is entered directly.
- After you have set the MS bit, enable data reception (RXE=1).

[bit13] SPI: SPI corresponding bit

This bit allows the SPI communication.

Value	Description
0	Normal synchronous transfer
1	SPI correspond

<Notes>

- Set this bit when the data transmisiion and reception is disabled (TXE=RXE=0).

[bit12] RIE: Received interrupt enable bit

- This bit enables or disables an output of received interrupt request to the CPU.
- If the RIE bit and the received data flag bit (SSR:RDRF) are "1", or if any of error flag bits (ORE) is "1", a received interrupt request is output.

Value	Description
0	Disables the received interrupt.
1	Enables the received interrupt.

[bit11] TIE: Transmit interrupt enable bit

- This bit enables or disables an output of transmit interrupt request to the CPU.
- If the TIE and SSR:TDRE bits are "1", a transmit interrupt request is output.

Value	Description
0	Disables a transmit interrupt.
1	Enables a transmit interrupt.

[bit10] TBIE: Transmit bus idle interrupt enable bit

- This bit enables or disables an output of transmit bus idle interrupt request to the CPU.
- If the TBIE bit and SSR:TBI bit are "1", a transmit bus idle interrupt request is output.

Value	Description
0	Disables the transmit bus idle interrupt.
1	Enables the transmit bus idle interrupt.

[bit9] RXE: Data received enable bit

Enables or disables a CSIO data reception.

Value	Description
0	Disables data reception.
1	Enables data reception.

<Notes>

- If data reception is disabled (RXE=0), the current data reception is stopped immediately.
- After you have set the MS bit and SMR:SCINV bit, enable the data reception (RXE=1).

[bit8] TXE: Data transmission enable bit

Enables or disables a CSIO data transmission.

Value	Description
0	Disables the transmission.
1	Enables the transmission.

<Note>

If data transmission is disabled (TXE=0), the current data transmission is stopped immediately.

5.2. Serial Mode Register (SMR)

The Serial Mode Register (SMR) is used to select an operation mode, to set a transmission direction, data length and serial clock inversion, and to enable or disable an output of serial data and clock to their pins.

bit	15 ... 8	7	6	5	4	3	2	1	0
Field	(SCR)	MD2	MD1	MD0	Reserved	SCINV	BDS	SCKE	SOE
Attribute		R/W	R/W	R/W	-	R/W	R/W	R/W	R/W
Initial value		0	0	0	-	0	0	0	0

[bit7:5] MD2, MD1, MD0: Operation mode set bits

These bits set an operation mode.

"0b000": Sets operation mode 0 (asynchronous normal mode).
"0b001": Sets operation mode 1 (asynchronous multiprocessor mode).
"0b010": Sets operation mode 2 (clock synchronous mode).
"0b011": Sets operation mode 3 (LIN communication mode).
"0b100": Sets operation mode 4 (I^2C mode).

*This chapter explains the registers and their operation in operation mode 2 (clock synchronous mode).

bit7	bit6	bit5	Description
0	0	0	Operation mode 0 (asynchronous normal mode)
0	0	1	Operation mode 1 (asynchronous multiprocessor mode)
0	1	0	Operation mode 2 (clock synchronous mode)
0	1	1	Operation mode 3 (LIN communication mode)
1	0	0	Operation mode 4 (I^2C mode)
Values other than the above			Setting is prohibited.

<Notes>

- Any bit setting other than above is inhibited.
- To switch the current operation mode, issue a programmable clear instruction (SCR:UPCL=1) and switch the operation mode continuously.
- After the operation mode has been set, set each register correctly.

[bit4] Reserved: Reserved bit

This bit value is undefined when read.
This bit has no effect on the operation when written

[bit3] SCINV: Serial clock invert bit

Inverts the serial clock format.

If set to "0":

- The signal mark level of serial clock output is set to "HIGH".
- The transmit data is output at a falling edge of serial clock during normal transfer, but it is output in synchronization with a rising edge of serial clock during SPI transfer.
- The received data is sampled at a rising edge of serial clock during normal transfer, but it is sampled at a falling edge of serial clock during SPI transfer.

If set to "1":

- The signal mark level of serial clock output is set to "LOW".
- The transmit data is output at a rising edge of serial clock during normal transfer, but it is output in synchronization with a falling edge of serial clock during SPI transfer.
- The received data is sampled at a falling edge of serial clock during normal transfer, but it is sampled at a rising edge of serial clock during SPI transfer.

Value	Description
0	Signal mark level "HIGH" format
1	Signal mark level "LOW" format

<Notes>

- Always set this bit when transmission and reception are disabled (TXE=RXE=0).
- After you have set the SCINV bit, enable data reception (SCR:RXE=1).

[bit2] BDS: Transfer direction select bit

Specifies to transfer the least significant bit of the transfer serial data first (LSB first; BDS=0) or the most significant bit first (MSB first; BDS=1).

Value	Description
0	LSB first (The least significant bit is first transferred.)
1	MSB first (The most significant bit is first transferred.)

<Note>

Always set this bit when transmission and reception are disabled (SCR:TXE=RXE=0).

[bit1] SCKE: Master mode serial clock output enable bit

This bit controls the serial clock I/O port.

Value	Description
0	Disables a serial clock output.
1	Enables a serial clock output.

<Note>

If this bit is used as the SCK pin, the GPIO must also be set.

[bit0] SOE: Serial data output enable bit

This bit enables or disables a serial data output.

Value	Description
0	Disables a serial data output.
1	Enables a serial data output.

<Note>

If this bit is used as the SOT pin, the GPIO must also be set.

5.3. Serial Status Register (SSR)

The Serial Status Register (SSR) is used to check the current transmission/reception state, check the Received Error flag, and clear the Received Error flag.

bit	15	14	13	12	11	10	9	8	7 ... 0
Field	REC	-	-	-	ORE	RDRF	TDRE	TBI	(ESCR)
Attribute	R/W	-	-	-	R	R	R	R	
Initial value	0	-	-	-	0	0	1	1	

[bit15] REC: Received error flag clear bit

This bit clears the ORE flag of the Serial Status Register (SSR).

- If this bit is set to "1", the error flag is cleared.
- This bit has no effect on the operation if set to "0".

"0" is always read.

Value	Description	
	At writing	At reading
0	No effect on the operation.	"0" is always read.
1	Clears the Received Error flag (FRE, ORE).	

[bit14:12] - : Unused bits

This bit value is undefined when read.

This bit has no effect on the operation when written.

[bit11] ORE: Overrun error flag bit

- If an overrun occurs during data reception, this bit is set to "1". This is cleared if the REC bit of Serial Status Register (SSR) is set to "1".
- If the ORE and SCR:RIE bits are "1", a received interrupt request is output.
- If this flag is set, data of the Received Data Register (RDR) is invalid.
- If this flag is set when received FIFO is used, the received FIFO enable bit is cleared and the received data is not stored in received FIFO.

Value	Description
0	No overrun error occurred.
1	An overrun error occurred.

[bit10] RDRF: Received data full flag bit

- This flag shows the state of Received Data Register (RDR).
- When the received data is loaded in the RDR, this bit is set to "1". When data is read from the Received Data Register (RDR), this bit is cleared to "0".
- If the RDRF bit and SCR:RIE bit are "1", a received interrupt request is output.
- If received FIFO is used and if the preset amount of data is received in received FIFO, the RDRF bit is set to "1".
- If received FIFO is used, if both of the following conditions are satisfied, and if the Received Idle state continues more than 8 baud rate clocks, the RDRF bit is set to "1".
  - The received FIFO idle detect enable bit (FCR1:FRIIE) is "1".
  - The preset data amount is not received and some data remains in received FIFO.

  If the RDR data is read during counting of 8 clocks, this counter is reset to "0", and counting for 8 clocks is restarted.
- If the received FIFO is used and if this buffer is emptied, this bit is cleared to "0".

Value	Description
0	The Received Data Register (RDR) is empty.
1	The Received Data Register (RDR) contains data.

[bit9] TDRE: Transmit data empty flag bit

- This flag shows the state of Transmit Data Register (TDR).
- If transmit data is written in the TDR, this bit is set to "0" to indicate that the TDR contains valid data. When data is loaded to the transmit shift register and when the transmission is started, this bit is set to "1" to indicate that the TDR does not have the valid data.
- If the TDRE bit and SCR:TIE bit are "1", a transmit interrupt request is output.
- When the UPCL bit of the Serial Control Register (SCR) is set to "1", the TDRE bit is set to "1".
- For the TDRE bit set/reset timing when transmit FIFO is used, see "2.4 Interrupt and flag set timing when transmit FIFO is used".

Value	Description
0	The Transmit Data Register (TDR) contains data.
1	The Transmit Data Register (TDR) is empty.

[bit8] TBI: Transmit bus idle flag bit

- This bit indicates that the CSIO is not transmitting data.
- When data is written in the Transmit Data Register (TDR), this bit is set to "0".
- If the Transmit Data Register (TDR) is empty (TDRE=1) and if no transmission is started, this bit is set to "1".
- When the UPCL bit of the Serial Control Register (SCR) is set to "1", the TDRE bit is set to "1".
- If this bit is "1" and if a transmit bus Idle interrupt is enabled (SCR:TBIE=1), a transmit interrupt request is output.

Value	Description
0	During data transmission
1	No data transmission

5.4. Extended Communication Control Register (ESCR)

The Extended Communication Control Register (ESCR) is used to set a transmit/received data length and to fix the serial data output to the "HIGH" state.

bit	15 ... 8	7	6	5	4	3	2	1	0
Field	-	SOP	-	-	WT1	WT0	L2	L1	L0
Attribute		R/W	-	-	R/W	R/W	R/W	R/W	R/W
Initial value		0	-	-	0	0	0	0	0

[bit7] SOP: Serial output pin set bit

- This bit sets the serial data output pin to the "HIGH" state. When this bit is set to "1", the SOT pin is set to "HIGH". After that, this bit needs not be set to "0".
- When it is read, "0" is always read.

Value	Description	
	At writing	At reading
0	No effect on the operation.	"0" is always read.
1	Sets the SOT pin to "HIGH" state.	

<Note>

Do not set this bit during serial data transmission.

[bit6:5] - : Unused bits

This bit value is undefined when read.
This bit has no effect when written.

[bit4:3] WT1, WT0: Data transmit/received wait select bits

In master operation mode, these bits set a wait count for continuous data transmission or reception. In slave operation mode, these bits are set to "00".

bit4	bit3	Description
0	0	0 bit
0	1	1 bit
1	0	2 bits
1	1	3 bits

[bit2:0] L2, L1, L0: Data length select bits

These bits set a length of transmit/received data.

bit2	bit1	bit0	Description
0	0	0	8-bit length
0	0	1	5-bit length
0	1	0	6-bit length
0	1	1	7-bit length
1	0	0	9-bit length
Values other than the above			Setting is prohibited.

<Note>

Any bit setting other than above is inhibited.

5.5. Received Data Register/Transmit Data Register (RDR/TDR)

The Received and Transmit Data Registers are allocated at the same address. This register functions as the Received Data Register when data is read from it. This register functions as the Transmit Data Register when data is written in it.

■ Received Data Register (RDR)

bit	15 ... 9	8	7	6	5	4	3	2	1	0
Field		D8	D7	D6	D5	D4	D3	D2	D1	D0
Attribute		R	R	R	R	R	R	R	R	R
Initial value		0	0	0	0	0	0	0	0	0

The Received Data Register (RDR) is a 9-bit data buffer register for serial data reception.

- When serial data signals are sent to the Serial input pin (SIN), they are converted by a shift register and stored in the Received Data Register (RDR).
- The high-order bits are sequentially set to "0" according to the data length as follows.

Data length	D8	D7	D6	D5	D4	D3	D2	D1	D0
9 bits	X	X	X	X	X	X	X	X	X
8 bits	0	X	X	X	X	X	X	X	X
7 bits	0	0	X	X	X	X	X	X	X
6 bits	0	0	0	X	X	X	X	X	X
5 bits	0	0	0	0	X	X	X	X	X

- When the received data is stored in the Received Data Register (RDR), the received data full flag bit (SSR:RDRF) is set to "1". If a received interrupt is enabled (SCR:RIE=1), a received interrupt request is generated.
- The Received Data Register (RDR) must be read only when the received data full flag bit (SSR:RDRF) is "1". When data is read from the Serial Received Data Register (RDR), the received data full flag bit (SSR:RDRF) is cleared to "0" automatically.
- If a received error occurs (SSR:ORE), data in the Received Data Register (RDR) is invalid.
- When the 9-bit length data is transferred, the RDR must be read in the 16-bit access mode.

<Notes>

- If the received FIFO is used and if a certain count of data is received by the received FIFO, the RDRF bit is set to "1".
- If received FIFO is used and if this buffer is emptied, the RDRF bit is cleared to "0".
- If received FIFO is used and if a received error occurs (SSR:ORE), the received FIFO enable bit is cleared and the received data is not stored in received FIFO.

■ Transmit Data Register (TDR)

bit	15 ... 9	8	7	6	5	4	3	2	1	0
Field		D8	D7	D6	D5	D4	D3	D2	D1	D0
Attribute		W	W	W	W	W	W	W	W	W
Initial value		1	1	1	1	1	1	1	1	1

The Transmit Data Register (TDR) is a 9-bit data buffer register for serial data transmission.

· If data transmission is enabled (SCR:TXE=1) and if the transmit data is written in the Transmit Data Register (TDR), the transmit data is transferred to the transmit shift register. Then, the data is converted into serial data, and output at the serial data output pin (SOT).

· The high-order bits are sequentially set to invalid data according to the data length as follows.

Data length	D8	D7	D6	D5	D4	D3	D2	D1	D0
9 bits	X	X	X	X	X	X	X	X	X
8 bits	Invalid	X	X	X	X	X	X	X	X
7 bits	Invalid	Invalid	X	X	X	X	X	X	X
6 bits	Invalid	Invalid	Invalid	X	X	X	X	X	X
5 bits	Invalid	Invalid	Invalid	Invalid	X	X	X	X	X

· When the transmit data is written in the Transmit Data Register (TDR), the transmit data empty flag (SSR:TDRE) is cleared to "0".

· When the transmit data is transferred to the transmit shift register and data transmission is started, and if transmit FIFO is disabled or if transmit FIFO is empty, the transmit data empty flag (SSR:TDRE) is set to "1".

· If the transmit data empty flag (SSR:TDRE) is "1", the next transmit data can be written in the buffer. If a transmit interrupt is enabled, a transmit interrupt occurs. The next transmit data must be written only after the transmit interrupt has occurred or when the transmit data empty flag (SSR:TDRE) is "1".

· If the transmit data empty flag (SSR:TDRE) is "0" and if transmit FIFO is disabled or transmit FIFO is full, the transmit data cannot be written in the Transmit Data Register (TDR).

· When the 9-bit length data is transferred, data must be written in the TDR in the 16-bit access mode.

<Notes>

· The Transmit Data Register is a write-only register. While the Received Data Register is a read-only register. As these two registers are allocated at the same address, the write and read values differ from each other. Therefore, the INC/DEC instruction and other read-modify-write (RMW) operation cannot be used.

· For the transmit data empty flag (SSR:TDRE) set timing when transmit FIFO is used, see "2.4 Interrupt and flag set timing when transmit FIFO is used".

5.6. Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0)

Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0) are used to set a frequency division ratio of serial clocks.

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	-	(BGR1)							(BGR0)							
Attribute	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- Set a clock frequency division to the Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0).
- The BGR1 register corresponds to the high-order bits, and the BGR0 register corresponds to the low-order bits. The reload value to be counted can be written, and the BGR1/BGR0 set value can be read.
- When the reload value is written in Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0), the reload counter starts its counting.

[bit15] - : Unused bit

This bit value is undefined when read.
This bit has no effect on the operation when written.

[bit14:8] BGR1: Baud Rate Generator Register 1

Process	Description
Write	Writes data in bit8 to 14 of reload counter.
Read	Reads the BGR1 set value.

[bit7:0] BGR0: Baud Rate Generator Register 0

Process	Description
Write	Write data in bit0 to 7 of reload counter.
Read	Reads the BGR0 set value.

<Notes>

- Data must be written in the Baud Rate Generator Register1, 0(BGR1 and BGR0) by 16-bit data accessing.
- If the reload value is even, the "HIGH" and "LOW" width of serial clock are as follows. If the value is odd, the serial clock has the same "HIGH" and "LOW" signal width.
  If SMR:SCINV="0", the "HIGH" width of serial clock is longer for 1 cycle of bus clock.
  If SMR:SCINV="1", the "LOW" width of serial clock is longer for 1 cycle of bus clock.
- Set the reload value to 3 or more.
- If the current values of Baud Rate Generator Register1, 0(BGR1, BGR0) are changed, the new values are reloaded only after the counter value has reached "15h00". In order to validate the new set values immediately, change the BGR1/BGR0 set values and execute the CSIO reset instruction (SCR:UPCL).
- If received FIFO is used and if you wish to set the received FIFO idle detect enable bit (FCR1:FRIIE) to "1" and starts the slave mode operation, set the desired baud rate in BGR1/BGR0.

5.7. FIFO Control Register 1 (FCR1)

The FIFO Control Register (FCR1) is used to set the FIFO test, select the transmit or received FIFO, enable the transmit FIFO interrupt, and control the interrupt flag.

bit	15	14	13	12	11	10	9	8	7 ... 0
Field		Reserved		FLSTE	FRIIE	FDRQ	FTIE	FSEL	(FCR0)
Attribute	-	-	-	R/W	R/W	R/W	R/W	R/W	
Initial value	-	-	-	0	0	1	0	0	

[bit15:13] Reserved : Reserved bits

The read value is "0". Be sure to write "0".

[bit12] FLSTE: Re-transmit data lost detect enable bit

This bit enables the FLST bit detection.

If set to "0": The FLST bit detection is disabled.
If set to "1": The FLST bit detection is enabled.

Value	Description
0	Disables the Data Lost detection.
1	Enables the Data Lost detection.

<Note>

If you wish to set this bit to "1", set the FSET bit to "1" first, and then set this bit to "1".

[bit11] FRIIE: Received FIFO idle detection enable bit

This bit sets to detect the received idle state if the received FIFO contains valid data and if it continues more than 8-bit hours. If the received interrupt is enabled (SCR:RIE=1), a received interrupt is generated when the received idle state is detected..

Value	Description
0	Disables the received FIFO idle detection.
1	Enables the received FIFO idle detection.

<Note>

In case of using Received FIFO, set this bit to "1".

[bit10] FDRQ: Transmit FIFO data request bit

This bit requests for the transmit FIFO data.
If this bit is "1", the transmit data is being requested. If the transmit FIFO interrupt is enabled (FTIE=1) during this time, a transmit FIFO interrupt request is output.

The FDRQ bit is set when:

- The FBYTE (for transmission) is "0" (Transmit FIFO is empty).
- Transmit FIFO is reset.

The FDRQ bit is reset when:

- This bit is set to "0".
- Transmit FIFO is filled with data.

Value	Description
0	Does not request for the transmit FIFO data.
1	Requests for the transmit FIFO data.

<Notes>

- If the FBYTE (for transmission) is "0", this bit cannot be set to "0".
- If this bit is "0", the FSEL bit state cannot be changed.
- If this bit is set to "1", it has no effect on the operation.
- If a read-modify-write instruction is issued, "1" is read.

[bit9] FTIE: Transmit FIFO interrupt enable bit

This bit enables a transmit FIFO interrupt. If this bit is set to "1", an interrupt occurs when the FDRQ bit is set to "1".

Value	Description
0	Disables the transmit FIFO interrupt.
1	Enables the transmit FIFO interrupt.

[bit8] FSEL: FIFO select bit

This bit selects the transmit or received FIFO.

Value	Description
0	Transmit FIFO:FIFO1; Received FIFO:FIFO2
1	Transmit FIFO:FIFO2; Received FIFO:FIFO1

<Notes>

- This bit is not cleared by FIFO reset (FCR0:FCL[2:1]=11).
- To change this bit state, first disable the FIFO operation (FCR0:FE[2:1]=00).

5.8. FIFO Control Register 0 (FCR0)

The FIFO Control Register 0 (FCR0) is used to enable/disable the FIFO operation, reset FIFO, save the read pointer, and set the data re-transmission.

bit	15 ... 8	7	6	5	4	3	2	1	0
Field	(FCR1)	-	FLST	FLD	FSET	FCL2	FCL1	FE2	FE1
Attribute		-	R	R/W	R/W	R/W	R/W	R/W	R/W
Initial value		0	0	0	0	0	0	0	0

[bit7] - : Unused bit

"0" is always read.

"0" must always be written.

[bit6] FLST: FIFO re-transmit data lost flag bit

This bit shows that the re-transmit data of transmit FIFO has been lost.

The FLST bit is set when:

· The FLSTE bit of FIFO Control Register 1 (FCR1) is "1", the write pointer of transmit FIFO matches the read pointer which has been saved by the FSET bit, and data is written in FIFO.

The FLST bit is reset when:

· FIFO is reset (FCL bit is set to "1").
· The FSET bit is set to "1".

If this bit is set to "1", the data identified by the read pointer (saved by the FSET bit) is overwritten. Therefore, the FLD bit cannot set the data re-transmission even if an error has occurred. If this bit is set to "1" and if you wish to re-transmit data, first reset FIFO. Then, write data in the FIFO buffer again.

Value	Description
0	No Data Lost has occurred.
1	Data Lost has occurred.

[bit 5] FLD: FIFO pointer reload bit

This bit reloads the data, being saved in transmit FIFO by the FSET bit, to the reload pointer. This bit can be used to re-transmit data after a communication error or others have occurred.
When the re-transmission setting has finished, this bit is set to "0".

Value	Description
0	Not reloaded
1	Reloaded

<Notes>

- If this bit is "1", data is being reloaded in the read pointer. Therefore, data writing except for FIFO reset is disabled.
- When FIFO is enabled or when data is being transmitted, this bit cannot be set to "1".
- After you have set the SCR:TIE bit and SCR:TBIE bit to "0", set this bit to "1". After you have enabled transmit FIFO, set the SCR:TIE bit and SCR:TBIE bit to "1".

[bit4] FSET: FIFO pointer save bit

This bit saves the transmit FIFO read pointer.
If the read pointer is saved before transmission and if the FLST bit is "0", data can be re-transmitted even when a communication error or others occur.

If set to "1": The current read pointer value is saved.
If set to "0": No effect on the operation.

Value	Description	
	At writing	At reading
0	Not saved	"0" is always read.
1	Saved	

<Note>

This bit can be set to "1" only when the transmit byte count (FBYTE) is "0".

[bit3] FCL2: FIFO2 reset bit

This bit resets the FIFO2 value.
When this bit is set to "1", the FIFO2 internal state is initialized.
Only the FCR1:FLST2 bit is initialized, but the other bits of FCR1/FCR0 registers are kept.

Value	Description	
	At writing	At reading
0	No effect on the operation.	"0" is always read.
1	FIFO2 is reset.	

<Notes>

- Disable the transmission and reception first, and then reset FIFO2.
- Set the transmit FIFO interrupt enable bit to "0" before the execution.
- The valid data count of the FBYTE2 register is set to "0".

[bit2] FCL1: FIFO1 reset bit

This bit resets the FIFO1 value.
When this bit is set to "1", the FIFO1 internal state is initialized.
Only the FCR1:FLST1 bit is initialized, but the other bits of FCR1/FCR0 registers are kept.

Value	Description	
	At writing	At reading
0	No effect on the opeartion.	"0" is always read.
1	FIFO1 is reset.	

<Notes>

- Disable the transmission and reception first, and then reset FIFO1.
- Set the transmit FIFO interrupt enable bit to "0" before the execution.
- The valid data count of the FBYTE1 register is set to "0".

[bit1] FE2: FIFO2 operation enable bit

This bit enables or disables the FIFO2 operation.

- To use the FIFO2 operation, set this bit to "1".
- If FIFO2 is set as transmit FIFO (FCR1:FSEL=1) and if data exists in FIFO2 when this bit is set to "1", the data transmission starts immediately when the UART is enabled to transmit data (SCR:TXE=1). During this time, set both SCR:TIE bit and SCR:TBIE bit to "0". Then, set this bit to "1" and set both SCR:TIE bit and SCR:TBIE bit to "1".
- If received FIFO is selected by the FSEL bit and if a received error has occurred, this bit is cleared to "0". This bit cannot be set to "1" until the received error is cleared.
- If FIFO2 is used as transmit FIFO, this bit must be set to "1" or "0" when the transmit buffer is empty (SSR:TDRE=1).
- If FIFO2 is used as received FIFO, this bit must be set to "0" when the received buffer is empty (SSR:RDRF=0) and no valid data exists in received FIFO (FBYTE2=0x00) after reception is disabled (SCR:RXE=0).
- If FIFO2 is used as received FIFO, this bit must be set to "1" when the received buffer is empty (SSR:RDRF=0) after reception is disabled (SCR:RXE=0).
- The FIFO2 state is held even if the FIFO2 operation is disabled.

Value	Description
0	Disables the FIFO2 operation.
1	Enables the FIFO2 operation.

[bit0] FE1: FIFO1 operation enable bit

This bit enables or disables the FIFO1 operation.

- To use the FIFO1 operation, set this bit to "1".
- If FIFO1 is set as transmit FIFO (FCR1:FSEL=0) and if data exists in FIFO1 when this bit is set to "1", the data transmission starts immediately when the UART is enabled to transmit data (SCR:TXE=1). During this time, set both SCR:TIE bit and SCR:TBIE bit to "0". Then, set this bit to "1" and set both TIE bit and TBIE bit to "1".
- If received FIFO is selected by the FSEL bit and if a received error has occurred, this bit is cleared to "0". This bit cannot be set to "1" until the received error is cleared.
- If FIFO1 is used as transmit FIFO, this bit must be set to "1" or "0" when the transmit buffer is empty (SSR:TDRE=1).
- If FIFO1 is used as received FIFO, this bit must be set to "0" when the received buffer is empty (SSR:RDRF=0) and no valid data exists in received FIFO (FBYTE2=0x00) after reception is disabled (SCR:RXE=0).
- If FIFO1 is used as received FIFO, this bit must be set to "1" when the received buffer is empty (SSR:RDRF=0) after reception is disabled (SCR:RXE=0).
- The FIFO1 state is held even if the FIFO1 operation is disabled.

Value	Description
0	Disables the FIFO1 operation.
1	Enables the FIFO1 operation.

5.9. FIFO Byte Register (FBYTE)

The FIFO Byte Register (FBYTE) indicates the effective data count in the FIFO buffer.

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	(FBYTE2)								(FBYTE1)							
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The FBYTE register indicates the effective data count of FIFO. The following shows the settings of the FCR1:FSEL bit.

Table 5-3 Display of data count

FCR1:FSEL	FIFO selection	Byte count display
0	FIFO2: Received FIFO, FIFO1: Transmit FIFO	FIFO2:FBYTE2, FIFO1:FBYTE1
1	FIFO2: Transmit FIFO, FIFO1: Received FIFO	FIFO2:FBYTE2, FIFO1:FBYTE1

- The initial value of data transfer count is "0x08" for the FBYTE register.
- Set a data count to generate a received interrupt flag for the FBYTE register of received FIFO. If this transfer data count matches the FBYTE register display, the received data full flag bit (RDRF) is set to "1".
- If both conditions below are satisfied and if the received idle state continues for more than 8 baud rate clocks, the received data full flag bit (RDRF) is set to "1".
  - The received FIFO idle detection enable bit (FRIIE) is "1".
  - The number of data sets stored in the received FIFO does not reach the transfer count.

  If the RDR data is read during counting of 8 clocks, this counter is reset to "0", and counting for 8 clocks is restarted. If received FIFO is disabled, this counter is reset to "0". If data remains in the received FIFO and if received FIFO is enabled, the data counting is restarted.
- To receive data in the master mode operation (master mode reception), set both SCR:TIE and SCR:TBIE bits to "0", set the received data count in the FBYTE register of transmit FIFO, and set the FCR1:FDRQ bit to "0". After set the SCR.RXE bit to "1", by setting the SCR:TXE to "1", the serial clock is output for the preset data amount, and the preset amount of data can be received. Set the SCR:TIE bit and SCR:TBIE bit to "1" only after the FCR1:FDRQ bit has been set to "1".

[bit15:8] FBYTE2: FIFO2 data count display bits

[bit7:0] FBYTE1: FIFO1 data count display bits

Writing	Sets the transfer data count.
Reading	Reads the effective count of data.

Read (Effective data count)

During transmission: The number of data sets already written in FIFO but not transmitted yet
During reception: The number of data sets received in FIFO

Write (Transfer data count)

During transmission: Set "0x00".
During reception: Set the data count to generate a received interrupt.

<Notes>

- The FBYTE register of transmit FIFO must be "0x00" except when data is received in the master mode operation.
- During the master mode data reception, the transmit data count must be set only when transmit FIFO is empty and both SCR:TIE bit and SSR:TBIE bit are "0".
- To disable the reception (SCR:RXE=0) when data is being received in the master mode operation, disable transmit FIFO first, and then disable the transmission and reception.
- The FBYTE bit of received FIFO must be set to "1" or larger.
- Change the FBYTE data of received FIFO only after you have disabled the data reception.
- A read-modify-write instruction cannot be used for this register.
- Any setting exceeding the FIFO capacity is inhibited.

CHAPTER 1-4: LIN Interface (Ver. 2.1) (LIN Communication Control Interface Ver. 2.1)

This chapter explains the LIN communication function, a part of multifunction serial interface functions and supported in Operation Mode 3.

CODE: 9BFLIN-E02.0 FM15L-E05.4

1. Overview of LIN Interface (Ver. 2.1) (LIN Communication Control Interface Ver. 2.1)

The LIN interface (ver. 2.1) (LIN communication control interface ver. 2.1) supports functions complying with the LIN bus. It also has transmit/received FIFO (up to 128 × 9 bits each)*1 installed.

■ Functions of LIN interface (ver. 2.1) (LIN communication control interface ver. 2.1)

		Function
1	Data buffer	· Full duplex double buffer (when FIFO is not used) · Transmit/received FIFO (max 128 × 9 bits each) * (when FIFO is used)
2	Serial input	Run oversampling three times with the bus clock and determine the value of received data based on the majority sampling value.
3	Transfer mode	Asynchronous
4	Baud rate	· A dedicated baud rate generator (constructed with a 15-bit reload counter) · The external clock can be adjusted with the reload counter.
5	Data length	8 bits
6	Signaling system	NRZ (Non Return to Zero)
7	Start bit detection	Synchronized with the falling edge of the start bit
8	Received error detection	· Framing error · Overrun error
9	Interrupt request	· Received interrupts (reception completed, framing error, overrun error) · Transmit interrupts (transmit data empty, transmit bus idle) · Status interrupts (LIN break field detection) · Interrupt request to ICU (LIN Sync field detection: LSYN) · Transmit FIFO interrupt (when transmit FIFO is empty) · DMA (Transmit/Received) transferring support function is available.
10	LIN bus option	· Supports LIN Protocol Revision 2.1 · Master device operations · Slave device operations · LIN break field generation (with variable bit length ranging from 13 to 16 bits) · LIN break delimiter generation (with variable data length ranging from 1 to 4 bits) · LIN break field detection · Detection of LIN sync field start/stop edges connected to input capture
11	FIFO options	· Transmit/received FIFO installed (maximum capacity: 128 × 9 bits for transmit FIFO, 128 × 9 bits for received FIFO) * · Transmit FIFO or received FIFO can be selected. · Transmit data can be resent. · Received FIFO interrupt timing can be changed via software. · FIFO resetting is supported independently.

* : The FIFO varies depending on the products type.

2. LIN Interface (Ver. 2.1) Interrupts

Received interrupts and transmit interrupts are provided for LIN interface (ver. 2.1). These interrupt requests can be generated if:
- Received data is set in the Received Data Register (RDR) or a data received error occurs.
- Transmit data is transferred from the Transmit Data Register (TDR) to the transmit shift register and the data transmission is started.
- The transmit bus is idle (No data transmission occurs).
- Transmit FIFO data is requested.
- A LIN break field is detected.

■ LIN interface (ver. 2.1) interrupts

Table 2-1 shows the interrupt control bits and the interrupt factors of LIN interface (ver. 2.1).

Table 2-1 LIN interface (ver. 2.1) interrupt control bits and interrupt factors

Interrupt type	Interrupt request flag bit	Flag register	Interrupt factor	Interrupt factor enable bit	Operation to clear interrupt request flag
Reception	RDRF	SSR	A single-byte reception	SCR:RIE	Reading from the received data register (RDR)
			Reception of a data volume matching the value set for FBYTE.		Reading from the Received Data Register (RDR) until received FIFO is emptied
			While the FRIIE bit is "1" and the received FIFO contains valid data, a received idle state continues for 8 bits or longer period.		
	ORE	SSR	Overrun error		Setting the Reception Error Flag Clear bit (SSR:REC) to "1"
	FRE	SSR	Framing error		
Transmission	TDRE	SSR	The Transmit Data Register is empty	SCR:TIE	Writing to the Transmit Data Register (TDR) or setting the transmit FIFO operation enable bit to "1" when the transmit FIFO operation enable bit is set to "0" and valid data are present in transmit FIFO (re-transmitting data) *1
	TBI	SSR	No data transmission	SCR:TBIE	Writing to the Transmit Data Register (TDR), setting the LIN break field setting bit (LBR) to "1", or setting the transmit FIFO operation enable bit to "1" when the transmit FIFO operation enable bit is set to "0" and valid data are present in transmit FIFO (re-transmitting data).*1
	FDRQ	FCR1	Transmit FIFO is empty.	FCR1:FTIE	The FIFO transmit data request bit (FCR1:FDRQ) is set to "0" or transmit FIFO is full.
Status	LBD	SSR	LIN break field is detected	ESCR:LBIE	The SSR:LBD bit is set to "0".
Input capture*2	ICP0/ ICP1	ICSA10/ICSA32	The first rising edge in the LIN Sync field	ICSA10.ICE0 ICSA10.ICE1 ICSA32.ICE0 ICSA32.ICE1	Disables ICP0 and ICP1
	ICP0/ ICP1	ICSA10/ICSA32	The fifth falling edge in the LIN Sync field		

*1: Set the TIE bit to "1" only after the TDRE bit has been set to "0".
*2: For the correspondace between the channel number of Input capture and that of LIN, see the descriptions of EPFR01/EPFR02/EPFR03 register.

2.1. Received interrupt and flag set timing

Data reception can be interrupted by a received completion (SSR:RDRF = 1), a received error occurrence (SSR:ORE, FRE = 1), or a LIN break field detection.

■ Received interrupt and flag set timing

Upon detection of the first stop bit, received data are stored in the Received Data Register (RDR). When the data reception is completed (SSR:RDRF = 1) or when a data received error occurs (SSR:ORE, FRE = 1), each flag is set. If received interrupts are enabled (SCR:RIE = 1) during this time, a received interrupt occurs.

<Note>

If a received error occurs, data in the Received Data Register (RDR) is invalidated.

Figure 2-1 RDRF (Received Data Full flag bit) set timing

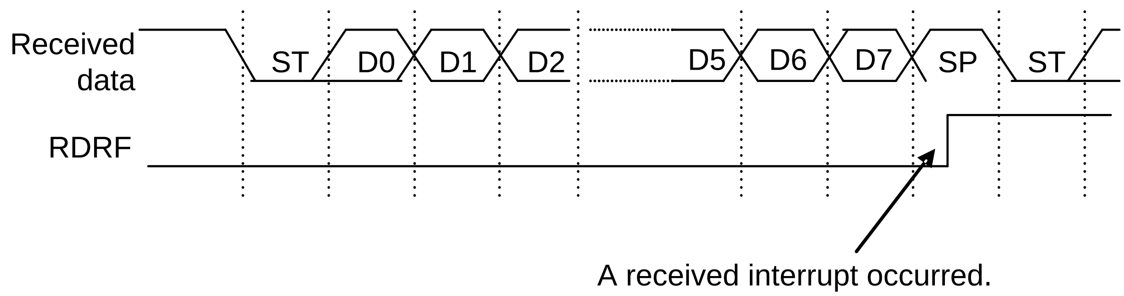
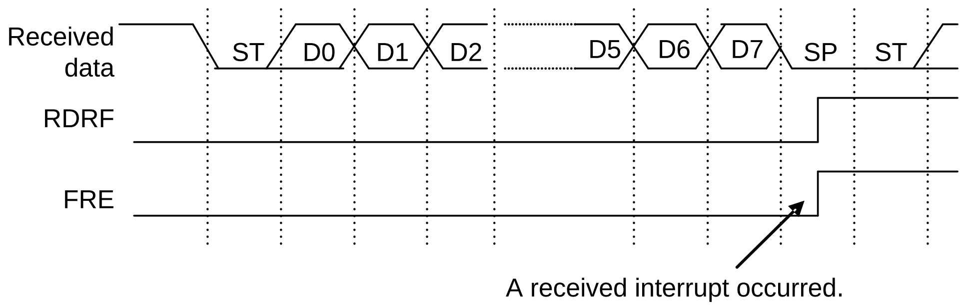


Figure 2-2 FRE (Framing Error flag bit) set timing



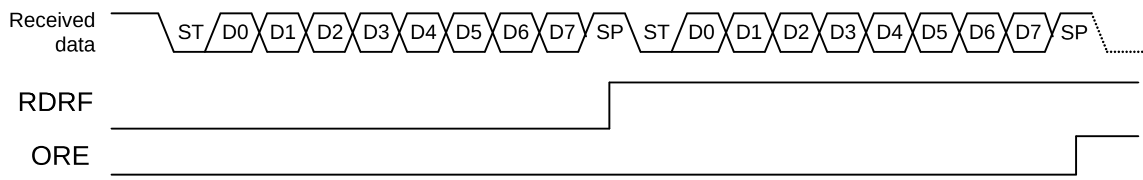
Notes:

- When the first stop bit is at "LOW" level, a framing error occurs.
- The RDRF bit is set to "1" and data can be received even if a framing error has occurred. However, the received data is invalid.

<Note>

During reception, if a falling edge of the serial data is detected concurrently with, or 1 to 2 bus clocks before the sampling point of the stop bit, the edge is ignored and the next data may not be received successfully. To output frames continuously, adequate intervals are required between frames.

Figure 2-3 ORE (Overrun Error flag bit) set timing



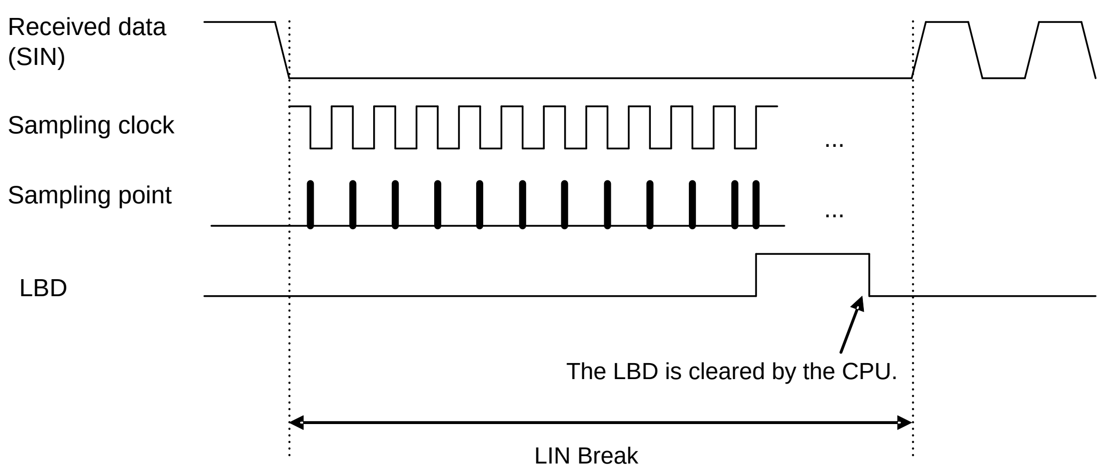
Notes:

If the next data is transferred before the received data is read (RDRF=1), an overrun error occurs.

■ LIN break field detection flag (LBD) set timing

If "0" is input for a width of 11 bits or more as serial input (SIN), the LBD bit is set to "1". If LIN break field interrupts are enabled (ESCR:LBIE = 1) then, a received interrupt occurs.

Figure 2-4 LBD (LIN Break field Detection flag) set timing



After 11 "LOW" state bits of the received data are detected at falling edge of the sampling clock, an LIN Break is detected at a rising edge of the sampling clock. When an LIN Break is detected, the LBD bit is set to "1".

2.2. Interrupt and flag set timing when received FIFO is used

If received FIFO is used, an interrupt occurs when the FBYTE data (preset for the FBYTE register (FBYTE)) is received.

■ Received interrupt and flag set timing when received FIFO is used

If the received FIFO is used, an interrupt occurs depending on the value set for the FBYTE register.

- When the amount of data set for transfer count in the FBYTE register is received, the received data full flag (SSR:RDRF) of the Serial Status register is set to "1". If received interrupts are enabled (SCR:RIE) during this time, a received interrupt occurs.
- If both of the following conditions are satisfied and if the received idle state continues for more than 8 baud rate clocks, the received data full flag (SSR:RDRF) is set to "1".
  - The received FIFO idle detection enable bit (FCR:FRIIE) is "1".
  - The number of data sets stored in the received FIFO does not reach the transfer count.

  If the RDR data is read during counting of 8 clocks, this counter is reset to 0 and counting for 8 clocks is restarted. If received FIFO is disabled, this counter is reset to "0". If data remains in the received FIFO and if received FIFO is enabled, the data counting is restarted.
- When the received data (RDR) is all read and received FIFO is emptied, the received data full flag (SSR:RDRF) is cleared.
- If the display of the valid received data amount is the same as the FIFO capacity and if the next data is received, an overrun error (SSR:ORE = 1) occurs.

Figure 2-5 Received interrupt occurrence timing when received FIFO is used

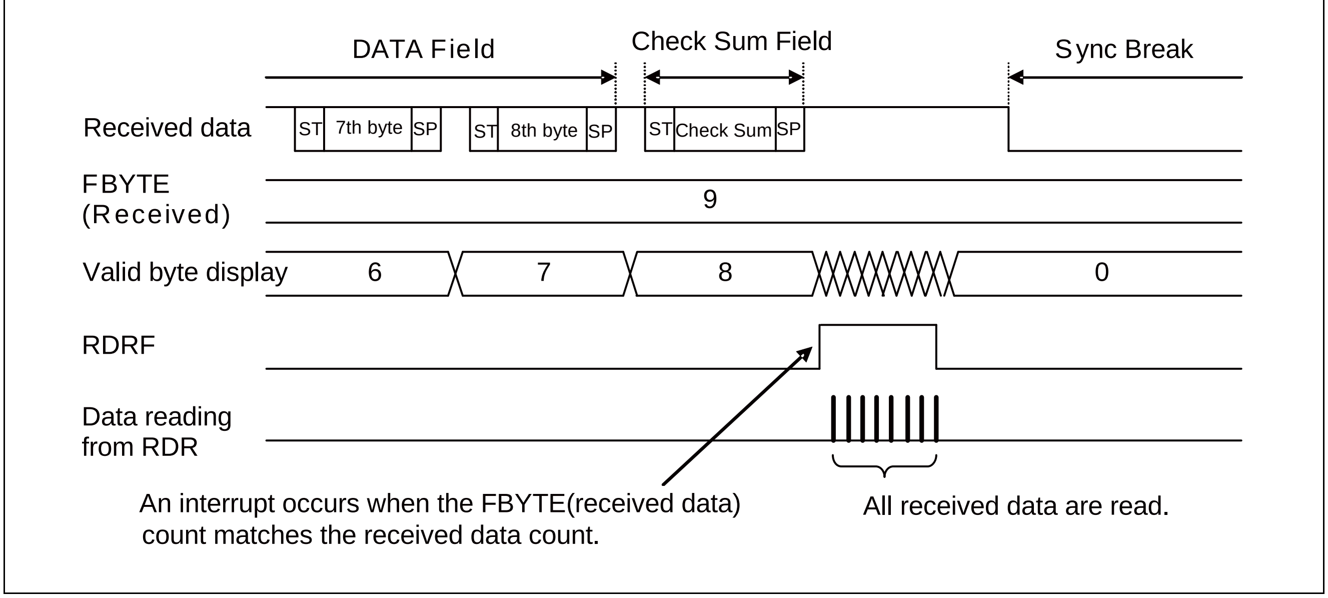
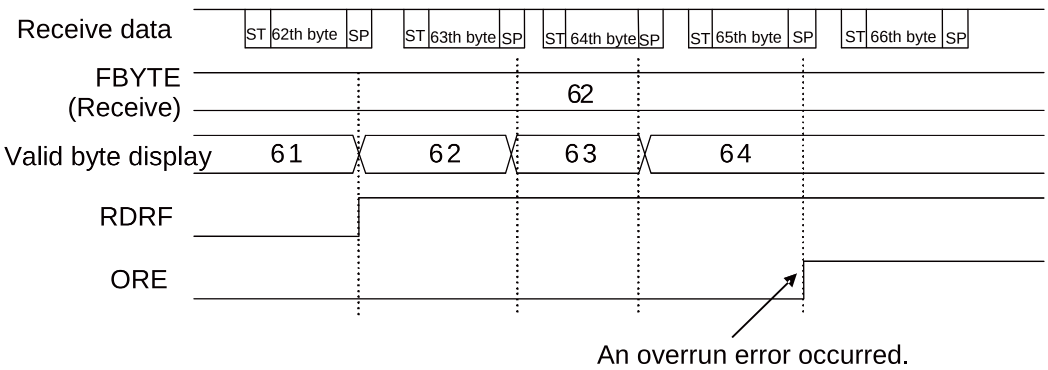


Figure 2-6 ORE (Overrun Error) flag bit set timing



Note:
If the FIFO capcity is displayed by the FBYTE and if the next data is received, an overrun error occurs.
This figure shows that the 64 bytes of FIFO capacity are used.

2.3. Transmit interrupt and flag set timing

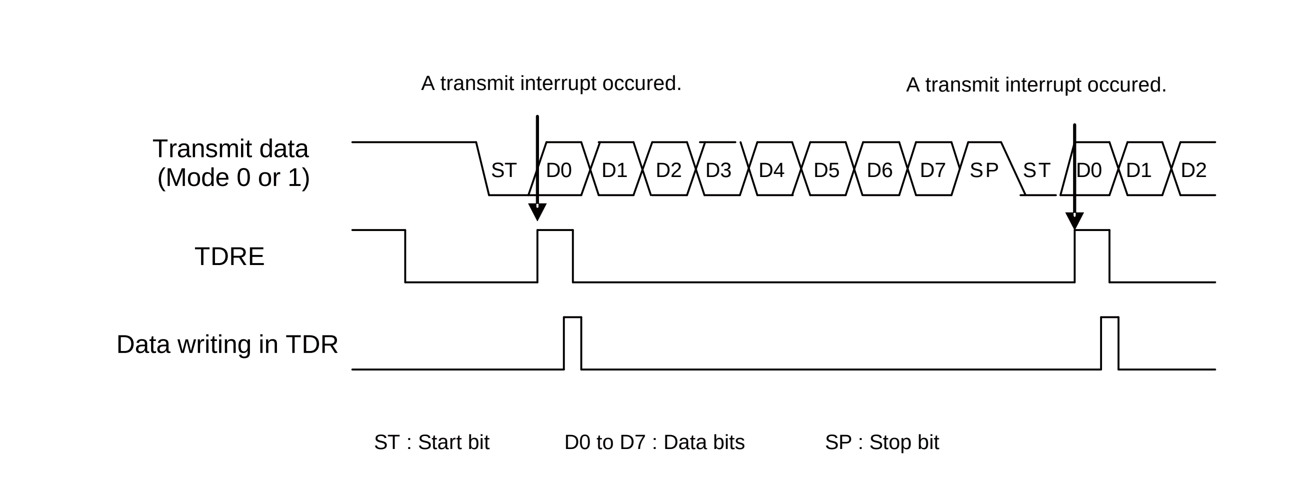
A transmit interrupt occurs when transmit data is transferred from the Transmit Data Register (TDR) to the transmit shift register (SSR:TDRE = 1) and transmission starts and when no transmission is performed (SSR:TBI = 1).

■ Transmit interrupt and flag set timing

● Transmit data empty flag (TDRE) set timing

After data has been transferred from the Transmit Data Register (TDR) to the transmit shift register, the next data can be written (SSR:TDRE=1). If transmit interrupts are enabled (SCR:TIE=1) during this time, a transmit interrupt occurs. As the TDRE bit is read only, the SSR:TDRE bit is cleared to "0" when data is written to the Transmit Data Register (TDR).

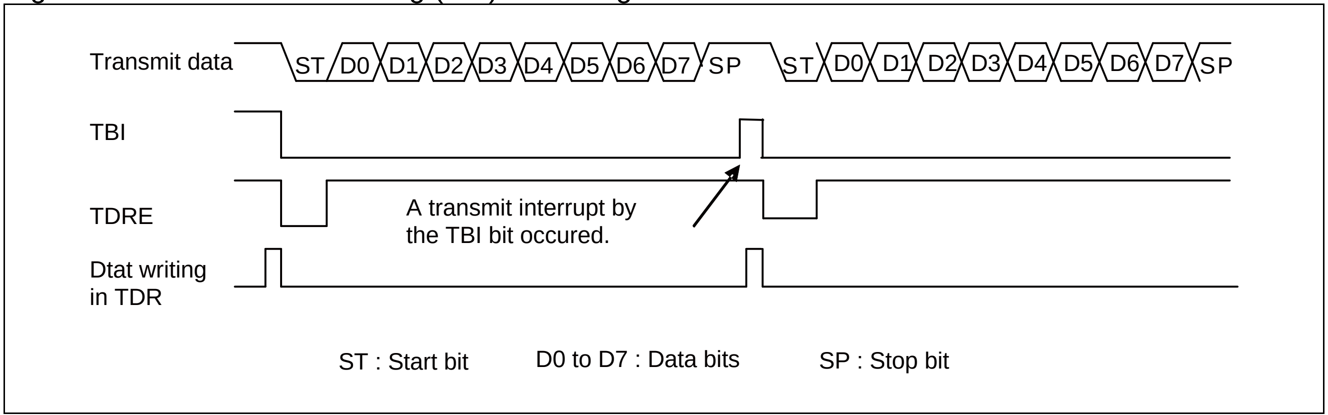
Figure 2-7 Transmit data empty flag (SSR:TDRE) set timing



● Transmit bus idle flag (TBI) set timing

If the Transmit Data Register is empty (TDRE=1) and no data is transmitted, the SSR:TBI bit is set to "1". If transmit bus idle interrupts are enabled (SCR:TBIE=1) during this time, a transmit interrupt occurs. When transmit data is written to the Transmit Data Register (TDR), both the TBI bit and the transmit interrupt request are cleared.

Figure 2-8 Transmit bus idle flag (TBI) set timing



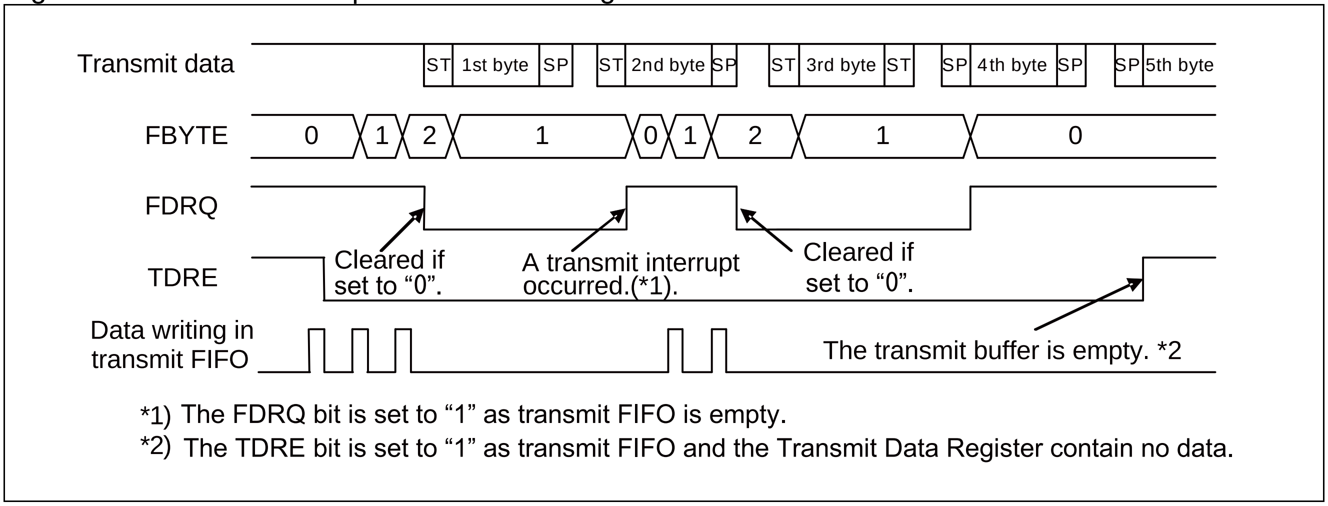
2.4. Interrupt and flag set timing when transmit FIFO is used

When the transmit FIFO is used, an interrupt occurs if the transmit FIFO contains no data.

■ Transmit interrupt and flag set timing when transmit FIFO is used

- If the transmit FIFO contains no data, the FIFO transmit data request bit (FCR1:FDRQ) is set to "1".
  If FIFO transmit interrupts are enabled (FCR1:FTIE=1) during this time, a transmit interrupt occurs.
- If a transmit interrupt has occurred and you have written the required data in transmit FIFO, clear the interrupt request by setting the FIFO transmit data request bit (FCR1:FDRQ) to "0".
- When transmit FIFO is filled with data, the FIFO transmit data request bit (FCR1:FDRQ) is set to "0".
- To check to see if transmit FIFO contains any data, read from the FIFO Byte Register (FBYTE).
  If FBYTE=0x00, no data exists in the transmit FIFO.

Figure 2-9 Transmit interrupt occurrence timing when transmit FIFO is used



3. Dedicated Baud Rate Generator

For the LIN interface (ver. 2.1) transmitting/receiving clock source, either of the following can be selected.
- Dedicated baud rate generator (reload counter)
- An external clock input to the baud rate generator (reload counter)

■ LIN interface (ver. 2.1) baud rate

Select one of the following two baud rates.

● Baud rate obtained by dividing an internal clock using the dedicated baud rate generator (reload counter)

This generator provides two internal reload counters, which support transmitting and receiving serial clocks respectively. To select the baud rate, specify the 15-bit reload value using Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0).

Each reload counter divides an internal clock by the set value.

To set the clock source, select an internal clock (SMR:EXT = 0).

● Baud rate obtained by dividing an external clock using the dedicated baud rate generator (reload counter)

Use an external clock for the clock source of the reload counter.

To select the baud rate, specify the 15-bit reload value using Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0).

Each reload counter divides an external clock by the set value.

To set the clock source, select use of an external clock and the baud rate generator clock (SMR:EXT = 1).

This mode is designed for cases where an oscillator with a divided non-standard frequency is used.

<Notes>

- Set the external clock (EXT = 1) while the reload counter is stopped (BGR1/BGR0 = 15h00).
- If an external clock is selected (EXT = 1), its HIGH and LOW signals must have a width at least of two bus clocks.

3.1. Baud rate settings

The following explains how to set the baud rate, and also a result of serial clock frequency calculation.

■ Calculating the baud rate

Two 15-bit reload counters are set using the Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0).
The baud rate is obtained in the following formulas.

(1) Reload value

$V = \phi / b - 1$

V : Reload value   b: Baud rate ϕ: Bus clock frequency or external clock frequency

(2) Calculation example

To set the 16 MHz bus clock, use the internal clock, and set the 19200 bps baud rate, set the reload value as follows:

Reload value:

V = (16 × 1000000) / 19200 – 1 = 832

Therefore, the baud rate is:

b = (16 × 1000000) / (832 + 1) = 19208 bps

(3) Baud rate error

The baud rate error can be obtained from the following equation.

Error (%) = (Calculated value – Target value) / Target value 100

Example: To set the 20 MHz bus clock and 153600 bps target baud rate:

Reload value = (20 × 1000000) /153600-1 = 129

Buad rate (Calculated value) = (20 × 1000000) / (129 + 1) = 153846 (bps)

Error (%) = (153846 - 153600) / 153600 × 100= 0.16 (%):

<Notes>

- If the reload value is set to "0", the reload counter is stopped.
- If the reload value is even, the "LOW" signal width of serial clock is longer than the "HIGH" signal width for a single cycle of bus clock. If the value is odd, the serial clock has the same "HIGH" and "LOW" signal width.
- Set the reload value to 3 or more. Note that data may not be received normally due to the baud rate error and reload value setting.
- For the allowable baud rate range, consider the jitter of the clock input to a macro.

■ Reload value and baud rate setting examples for each bus clock frequency

The following shows reload values and baud rate setting examples.

Table 3-1 Reload values and baud rate setting examples

Baud rate (bps)	8 MHz		10 MHz		16 MHz		20 MHz		24 MHz	
	Value	ERR	Value	ERR	Value	ERR	Value	ERR	Value	ERR
8M	-	-	-	-	-	-	-	-	-	-
6M	-	-	-	-	-	-	-	-	3	0
5M	-	-	-	-	-	-	3	0	-	-
4M	-	-	-	-	3	0	4	0	5	0
2.5M	-	-	3	0	-	-	7	0	-	-
2M	3	0	4	0	7	0	9	0	11	0
1M	7	0	9	0	15	0	19	0	23	0
500000	15	0	19	0	31	0	39	0	47	0
460800	-	-	-	-	-	-	-	-	51	0.16
250000	31	0	39	0	63	0	79	0	95	0
230400	-	-	-	-	-	-	86	-0.22	103	0.16
153600	51	0.16	64	0.16	103	0.16	129	0.16	155	0.16
125000	63	0	79	0	127	0	159	0	191	0
115200	-	-	86	-0.22	138	-0.08	173	-0.22	207	0.16
76800	103	0.16	129	0.16	207	0.16	259	0.16	312	-0.16
57600	138	-0.08	173	-0.22	277	-0.08	346	0.06	416	-0.08
38400	207	0.16	259	0.16	416	-0.08	520	-0.03	624	0
28800	277	-0.08	346	<0.01	555	-0.08	693	0.06	832	0.03
19200	416	-0.08	520	-0.03	832	0.03	1041	-0.03	1249	0
10417	767	<0.01	959	<0.01	1535	<0.01	1919	<0.01	2303	<0.01
9600	832	0.04	1041	-0.03	1666	-0.02	2082	0.02	2499	0
7200	1110	<0.01	1388	<0.01	2221	<0.01	2777	<0.01	3332	<0.01
4800	1666	-0.02	2082	0.02	3332	<0.01	4166	<0.01	4999	0
2400	3332	<0.01	4166	<0.01	6666	<0.01	8332	<0.01	9999	0
1200	6666	<0.01	8332	<0.01	13332	<0.01	16666	<0.01	19999	0
600	13332	<0.01	16666	<0.01	26666	<0.01	-	-	-	-
300	26666	<0.01	-	-	-	-	-	-	-	-

Value: BGR1/0 register set value
ERR: Baud rate error (%)

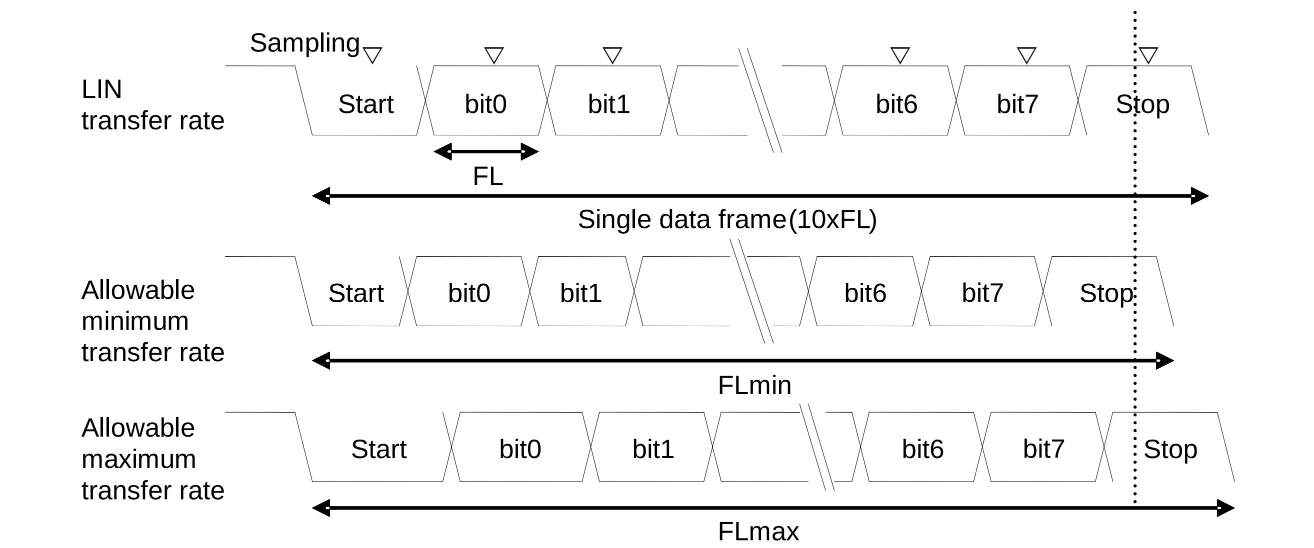
Table 3-2 Reload values and baud rate setting examples (Continued)

Baud rate (bps)	32MHz		40 MHz		48 MHz		72 MHz	
	Value	ERR	Value	ERR	Value	ERR	Value	ERR
8M	3	0	4	0	5	0	8	0
6M	-	-	-	-	7	0	11	0
5M	-	-	7	0	-	-	-	-
4M	7	0	9	0	11	0	17	0
2.5M	-	-	15	0	-	-	-	-
2M	15	0	19	0	23	0	35	0
1M	31	0	39	0	47	0	71	0
500000	63	0	79	0	95	0	143	0
460800	-	-	86	-0.22	103	0.16	155	0.16
250000	127	0	159	0	191	0	287	0
230400	-	-	173	-0.22	207	0.16	312	-0.16
153600	207	-0.16	259	0.16	312	-0.16	468	-0.05
125000	255	0	319	0	383	0	575	0
115200	277	0.08	346	0.06	416	-0.08	624	0
76800	416	0.08	520	-0.03	624	0	937	-0.05
57600	555	0.08	693	0.06	832	0.04	1249	0
38400	832	-0.04	1041	-0.03	1249	0	1874	0
28800	1110	-0.01	1388	<0.01	1666	-0.02	2499	0
19200	1666	0.02	2082	0.02	2499	0	3749	0
10417	3071	<0.01	3839	<0.01	4607	<0.01	6911	<0.01
9600	3332	-0.01	4166	<0.01	4999	0	7499	0
7200	4443	-0.01	5555	<0.01	6666	<0.01	9999	0
4800	6666	<0.01	8332	<0.01	9999	0	14999	0
2400	13332	<-0.01	16666	<0.01	19999	0	29999	0
1200	26666	<0.01	-	-	-	-	-	-
600	-	-	-	-	-	-	-	-
300	-	-	-	-	-	-	-	-

■ Allowable baud rate range for data reception

The following shows the range of baud rate error allowed for the destination to receive data.
Set the reception baud rate error by using the following formulas to ensure that the value falls within the allowable range.

Figure 3-1 Allowable baud rate range for data reception



As shown in Figure 3-1, after detection of the start bit, the sampling timing of received data data is determined by the counter set in the BGR1/BGR0 register. Data can be received successfully if the last data including the stop bit matches the sampling timing.

If this applies to a reception of 10 bits, a theoretical explanation can be given in the following.

Assuming that the sampling timing margin is one bus clock (ϕ), the minimum allowable transfer rate (FLmin) is determined as follows:

$FLmin = (10bit \times (V+1) - (V+1)/2 + 2)/\phi = (19V + 23)/2\phi$ (s) V: Reload value, ϕ: Bus clock

Thus, the maximum baud rate that allows the destination to receive data (BGmax) is determined as follows.

$BGmax = 10/FLmin = 20\phi/(19V+23)$ (bps) V: Reload value, ϕ: Bus clock

When data is received at the maximum allowable transfer rate (FLmax), the starting point of the received data 10th bit is sampled.

Thus, the maximum allowable transfer rate (FLmax) is determined as follows:

$9/10 \times FLmax = (10bit \times (V+1) - (V+1)/2)/\phi$ V: Reload value, ϕ: Bus clock

$FLmax = (19/18 \times 10 \times (V+1))/\phi$

Assuming that the sampling timing margin (ϕ) is two clocks, the maximum allowable transfer rate (FLmax) is determined as follows:

$9/10 \times FLmax = (10bit \times (V+1) - (V+1)/2 - 2)/\phi$ V: Reload value, ϕ: Bus clock

$FLmax = (19/18 \times 10 \times (V+1) - 40/18)/\phi = (190V + 150)/18\phi$ (s) V: Reload value, ϕ: Bus clock

Accordingly, the minimum baud rate that allows the destination to receive data (BGmin) is determined as follows:

$BGmin = 10/FLmax = 18\phi/(19V+15)$ (bps) V: Reload value, ϕ: Bus clock

From the above formulas that yields the minimum/maximum baud rates, the allowable baud rate errors between the LIN interface (ver. 2.1) and the destination can be obtained as shown in the following table.

Reload value (V)	Maximum allowable baud rate error	Minimum allowable baud rate error
3	0%	0
10	+3.28%	-3.41%
50	+4.83%	-4.87%
100	+5.04%	-5.07%
200	+5.15%	-5.16%
32767	+5.26%	-5.26%

<Note>

Reception accuracy depends on the number of bits per frame, bus clock, and reload value. The higher the bus clock and frequency division ratio are, the higher the accuracy becomes.

■ External clock

Writing "1" to the EXT bit of the Baud Rate Generator Register (BGR) causes the baud rate generator to divide the external clock's frequency.

<Note>

The external clock signal is synchronized with the internal clock on the LIN interface (ver. 2.1). Therefore, an external clock that does not allow synchronization causes unstable operation.

■ Functions of reload counter

There are two types of reload counters: The transmit reload counter and the received reload counter, both functioning as a dedicated baud rate generator. Each reload counter consists of a 15-bit register for the reload value, and generates transmitting and receiving clocks from the external or internal clock.

■ Starting counting

When the reload value is written to the Baud Rate Generator Register1, 0 (BGR1 or BGR0), the reload counter starts counting.

■ Restarting

The reload counter restarts counting in the following conditions.

● Common to transmit and received reload counters

A programmable reset (SCR:UPCL bit)

● Received reload counter

Detection of the start bit's falling edge in asynchronous mode

4. LIN Interface (Ver. 2.1) Operations

The LIN interface (ver. 2.1) performs bi-directional LIN communication of master and slave.

■ Master mode operations

● Selecting master mode

To operate the LIN interface as a master, set the SCR:MS bit to "0".

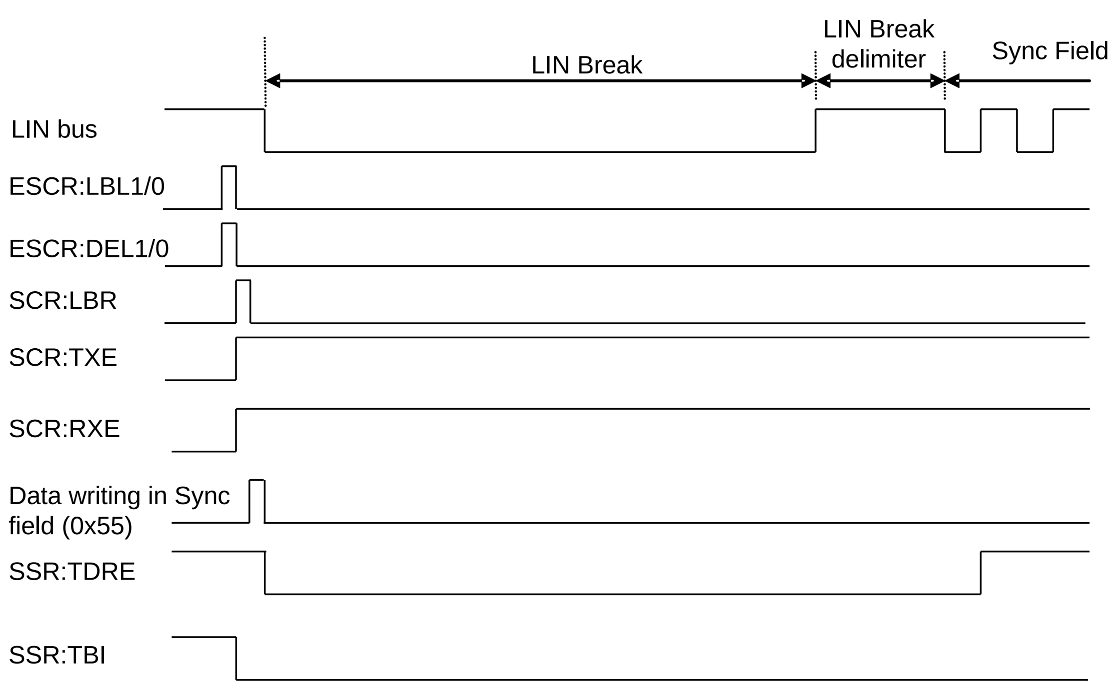
● Break field transmission-sync field transmission

- The break field length (ESCR:LBL1, LBL0) and the break field delimiter length (ESCR:DEL1, DEL0) can be selected.
- If transmission is enabled (SCR:TXE=1), and the SCR:LBR bit (LIN Break field setting bit) is set to "1", then the break field is transmitted.
- The sync field is transmitted when "0x55" is written to the Transmit Data Register (TDR).

<Notes>

- Before setting the Transmit Data Register (TDR) to "0x55", set the SCR:LBR bit (LIN break field setting bit) to "1".
- Setting the SCR:RXE bit (reception enable bit) to "1" does not enable the break field to perform reception.

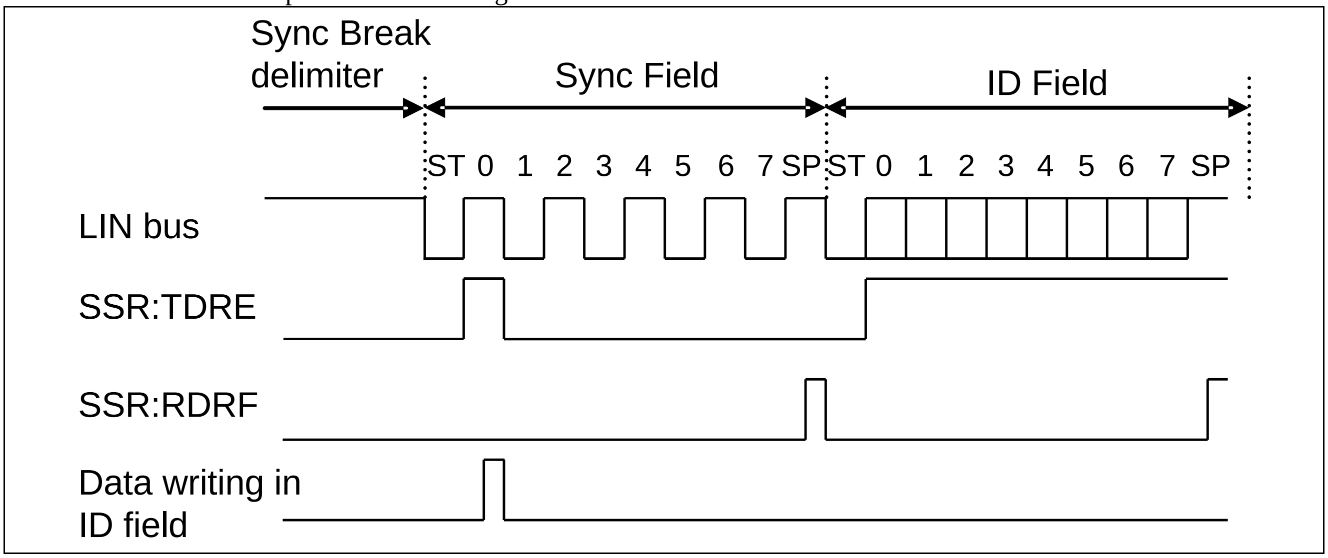
Figure 4-1 Break field-sync field transmission



Break field length – With ESCR:LBL[1:0] , it can be set between 13 and 16 bits long.
Break delimiter length – With ESCR:DEL[1:0], it can be set between 1 and 4 bits long.

● Sync field transmission-ID field transmission

- When the first bit of the sync field (0x55) is transmitted, the SSR:TDRE (transmit data empty) bit is set to "1". If transmit interrupts are enabled (SCR:TIE = 1) during this time, a transmit interrupt occurs.
- If a transmit interrupt occurs, the ID field can be written to the Transmit Data Register (TDR).
- If a received interrupt occurs, compare the received data with the transmit data to make sure that no error has occurred.
- The ID field is output in 8-bit data length and LSB-first order.



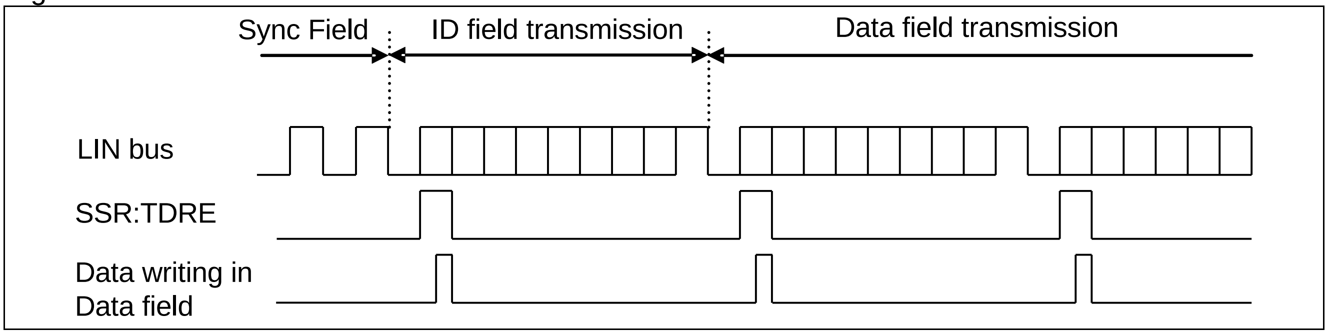
● ID field transmission-DATA field transmission/reception

Select whether to transmit the DATA field to a slave device or to receive the DATA field.

(To transmit the DATA field)

When the first bit of the ID field is transmitted, the SSR:TDRE bit is set to "1". Then data can be written to the DATA field.

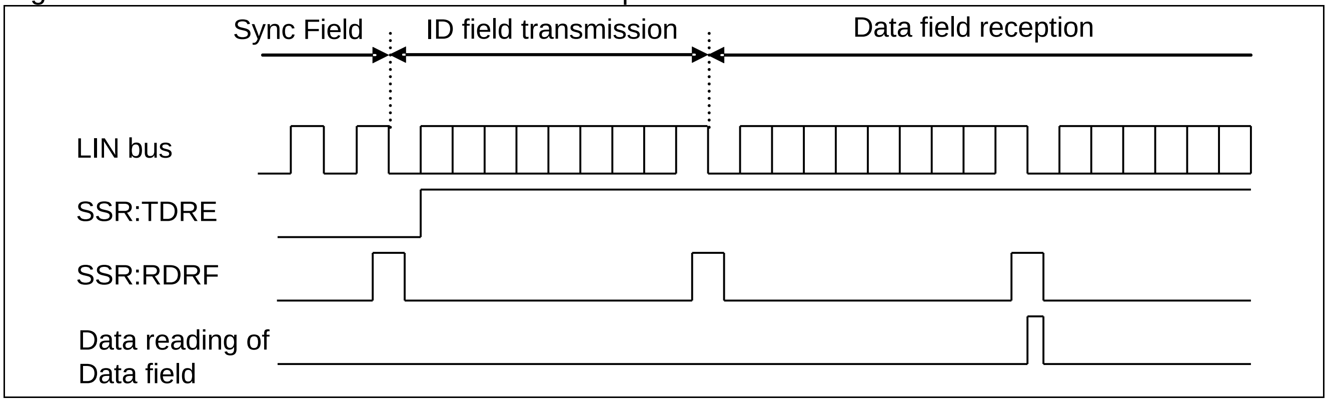
Figure 4-2 ID field transmission-DATA field transmission



(To receive the DATA field)

- When the first bit of the ID field is transmitted, the SSR:TDRE bit is set to "1". However, do not write any transmit data then. Also disable transmit interrupts (SCR:TIE = 0).
- When the DATA field is received, SSR:RDRF is set to "1". If received interrupts are enabled (SSR:RIE = 1) then, a received interrupt occurs.
- A start bit is detected when a falling edge is detected after data passes the noise filter (with the majority value applied after sampling serial data input three times with the bus clock) and a LOW level is detected for the data passing the sampling point.

Figure 4-3 ID field transmission-DATA field reception



<Notes>

- The LIN interface (Ver. 2.1) includes noise filter (with the majority value applied after sampling serial data input three times with the bus clock). However, design the board so as not to allow noise to pass through this filter or perform communications so that any noise that has passed does not cause any problems (e.g., by adding a data checksum to the end and resending the data if any error occurs).
- During reception, if a falling edge of the serial data is detected concurrently with, or 1 to 2 bus clocks before the sampling point of the stop bit, the edge is ignored and the next data cannot be received successfully. To output frames continuously, adequate intervals should be considered between frames.

● Master mode operation timing chart (when FIFO is not used)

Figure 4-4 LIN bus timing (when DATA field is transmitted and FIFO is not used)

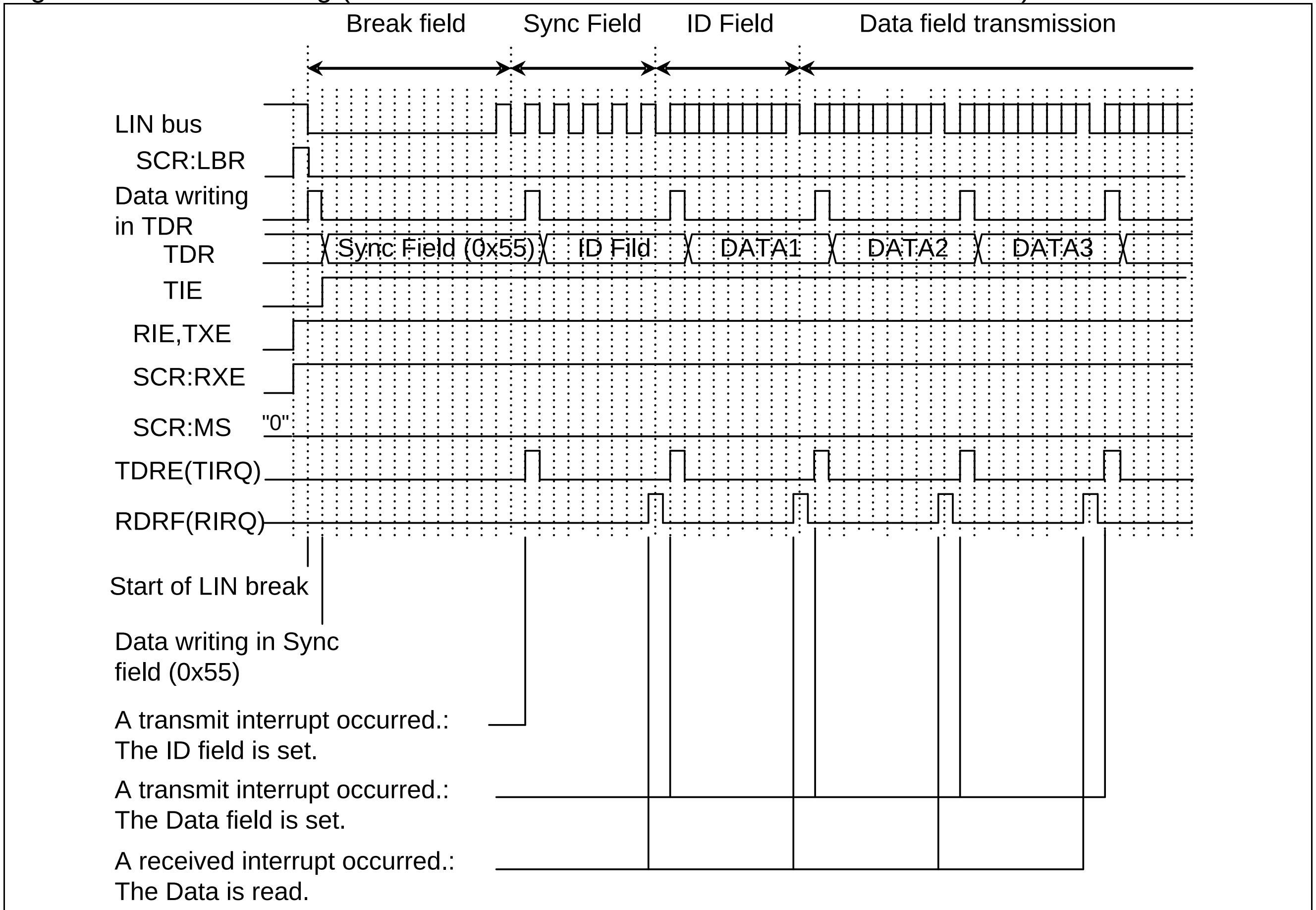
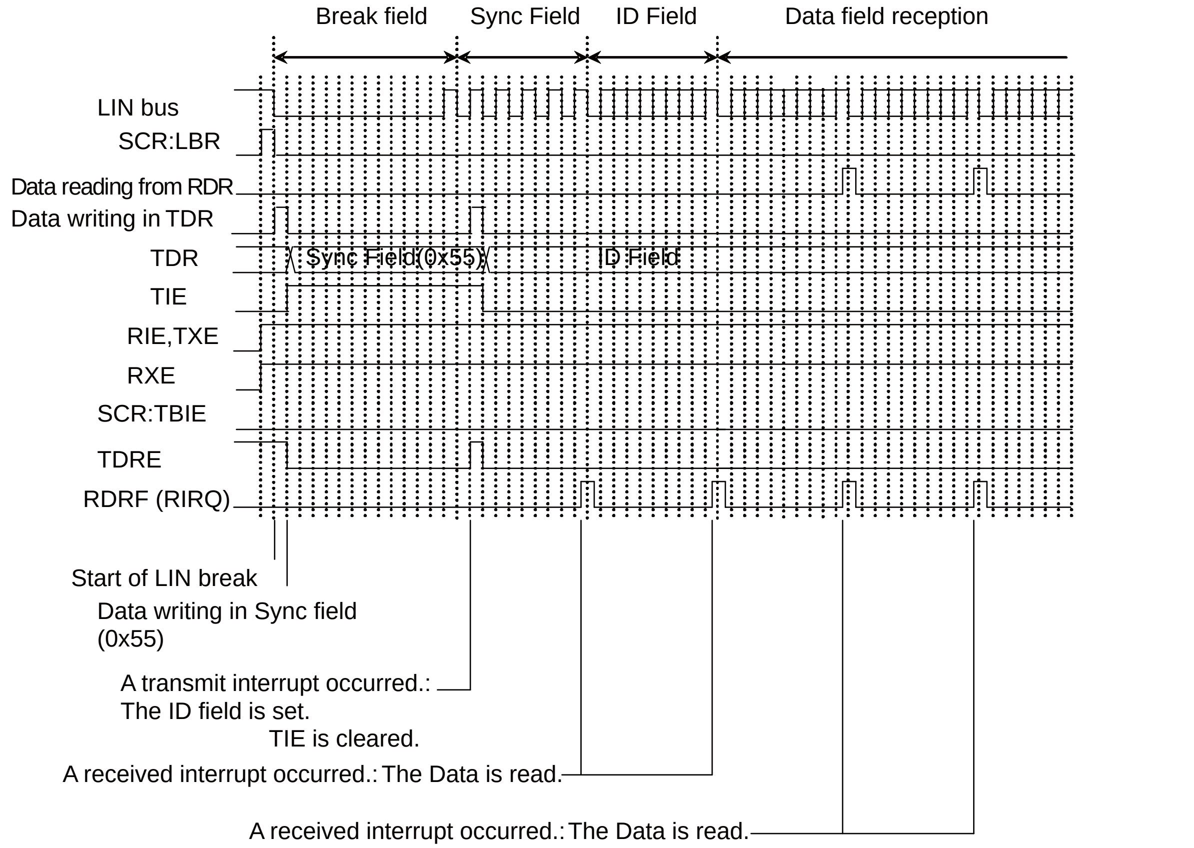


Figure 4-5 LIN bus timing (when DATA field is received and FIFO is not used)



● **Master mode operation timing chart (when FIFO is used)**

Figure 4-6 LIN bus timing (when DATA field is transmitted and FIFO is used)

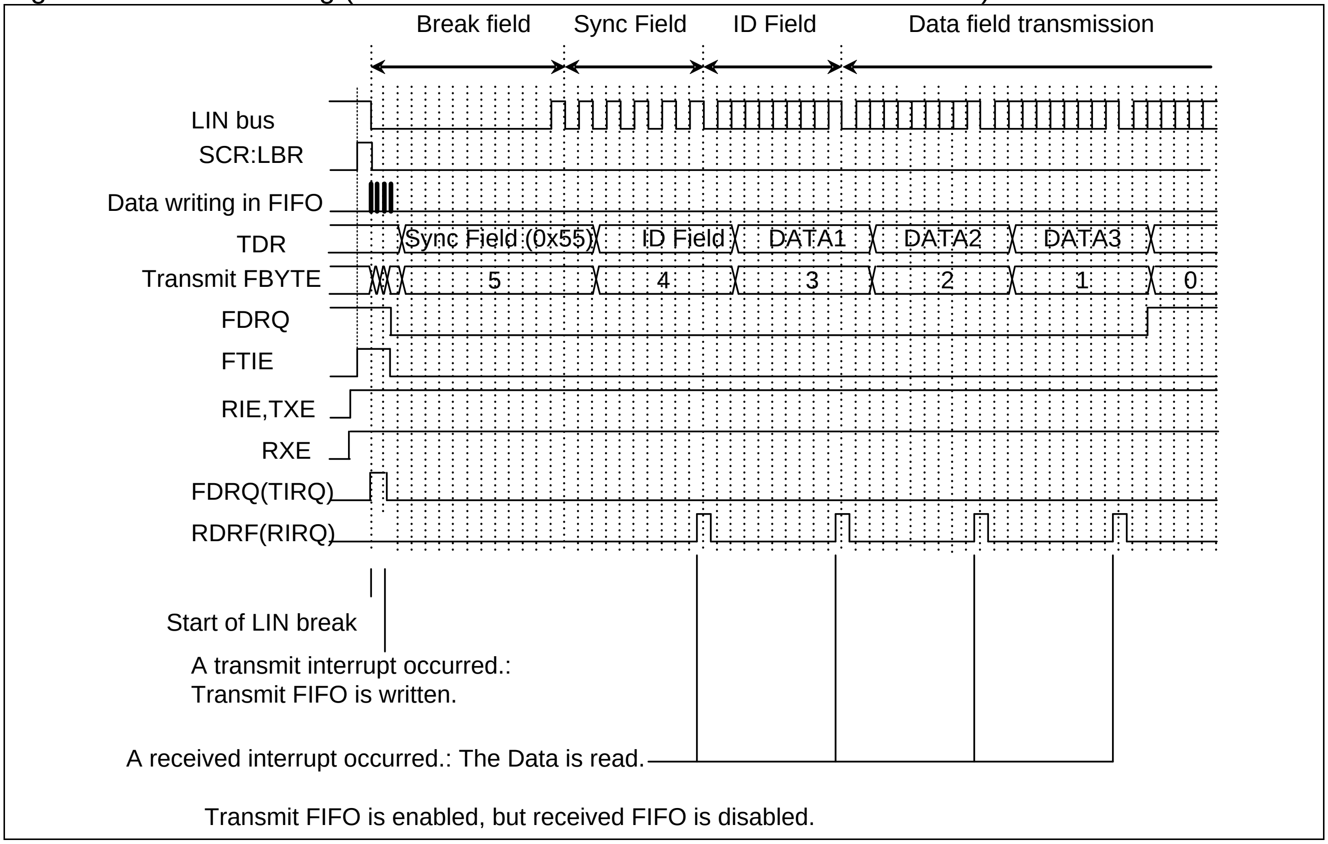
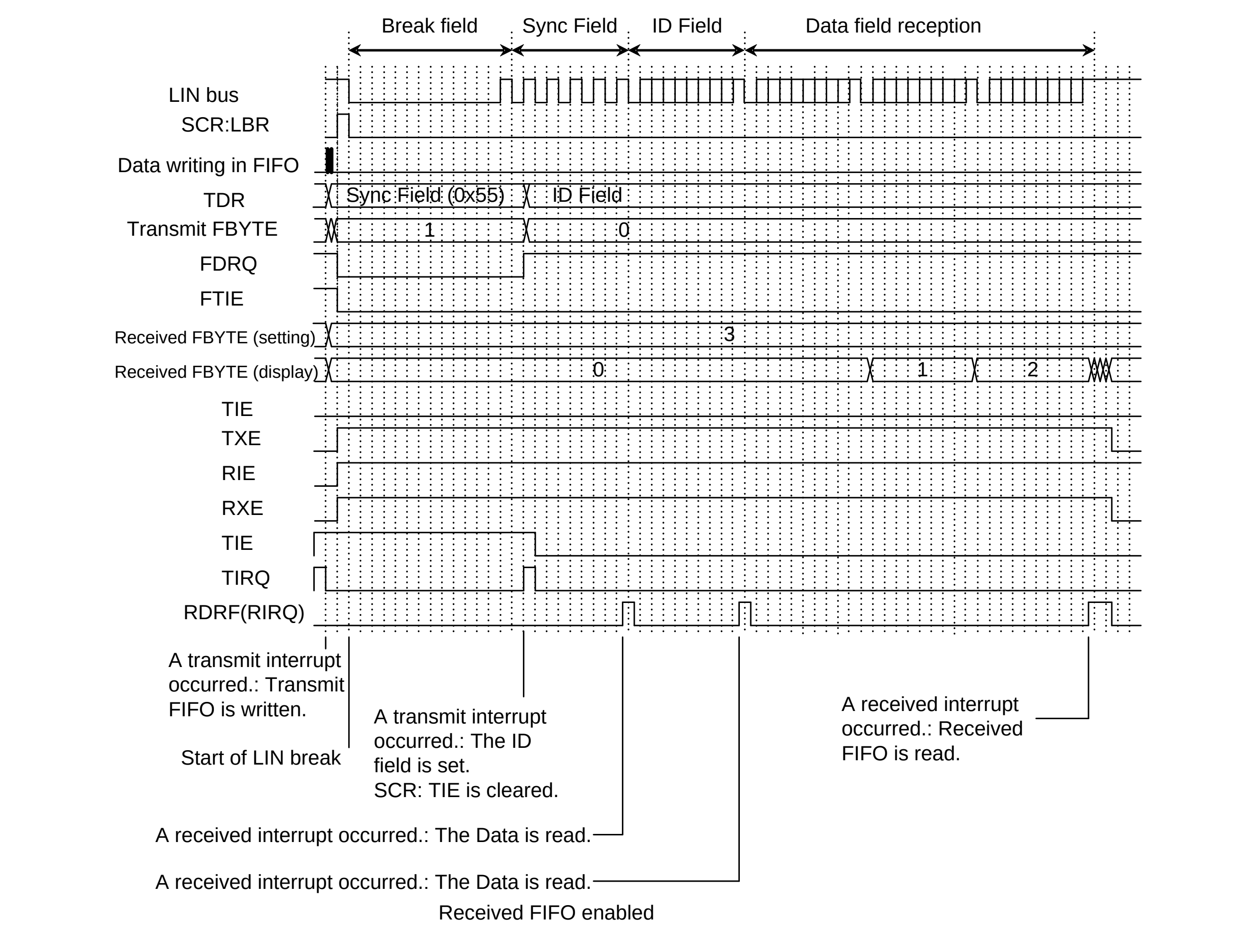


Figure 4-7 LIN bus timing (when DATA field is received and FIFO is used)



■ Slave mode operations

● Selecting slave mode

To operate the LIN interface as a slave, set the SCR:MS bit to "1".

● Break field reception-sync field reception

1. If the break field is input, the break field is detected (SSR:LBD = 1) at the 11th bit.
   If the ESCR:LBIE bit is set to "1" then, a received interrupt occurs.
2. Enable ICU interrupts then to detect both edges.
3. The LIN interface (ver. 2.1), upon the detection of the first falling edge in the sync field, sets the internal signal (LSYN) input to ICU to HIGH to start the ICU. This internal signal (LSYN) turns to LOW at the fifth falling edge.
4. The internal signal (LSYN) input to ICU is a value that the HIGH period multiplies the baud rate by eight. The baud rate set value is obtained as follows:

If the free run timer is not overflowed:

$$\text{BGR value} = (b - a) \times Fe/(8 \times \phi) - 1$$

If the free run timer is overflowed:

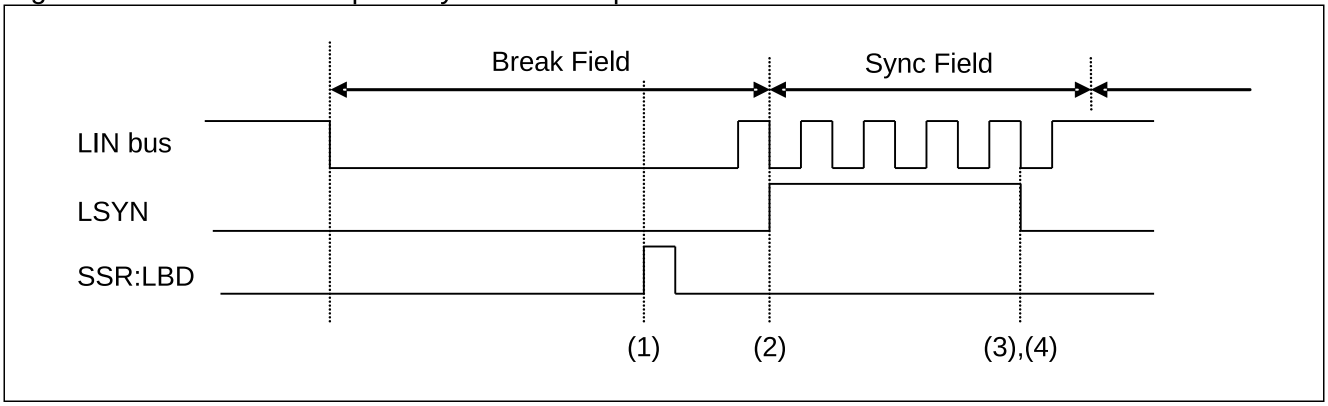
$$\text{BGR value} = (max + 1 + b - a) \times Fe/(8 \times \phi) - 1$$

max	:	Maximum value of the free run timer
a	:	The ICU data register value after the first interrupt
b	:	The ICU data register value after the second interrupt
ϕ	:	Bus clock frequency (MHz)
Fe	:	External clock frequency (MHz). When the internal clock is used (EXT = 0), Fe = ϕ is assumed.

<Note>

To operate the break field and the sync field, disable the reception (SCR:RXE = 0).

Figure 4-8 Break field reception-sync field reception



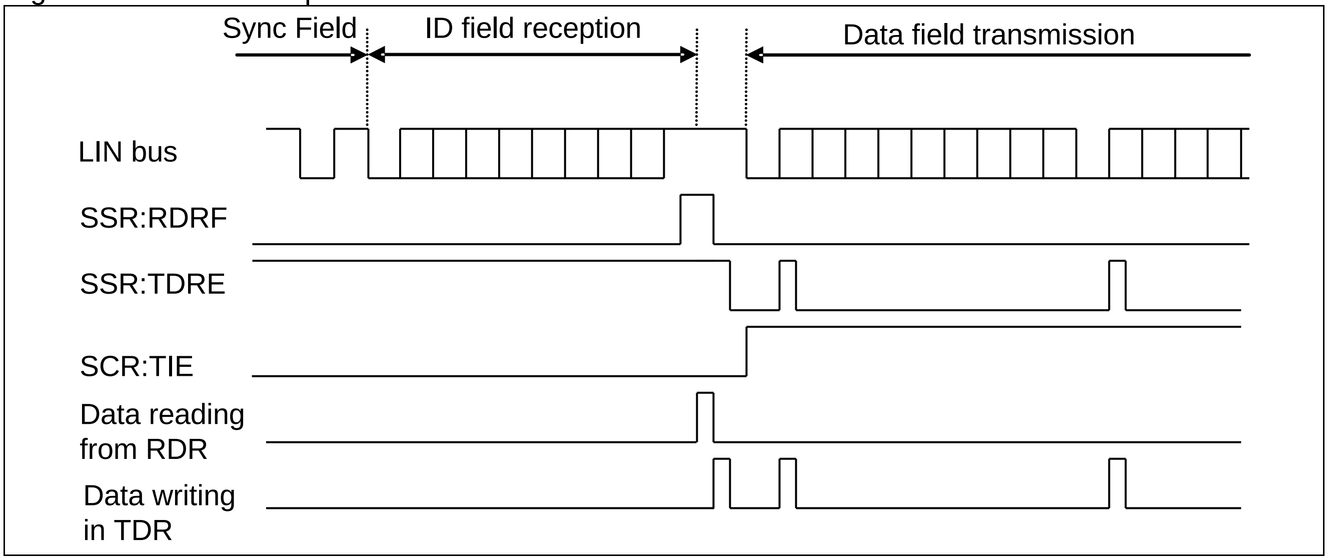
● ID field reception-DATA field transmission/reception

After reception of the ID field, whether to transmit or to receive the DATA field to master can be selected.

(To transmit the DATA field)

After reception of the ID field, write data to the Transmit Data Register (TDR). Enable transmit interrupts (SCR:TIE = 1) during this time.

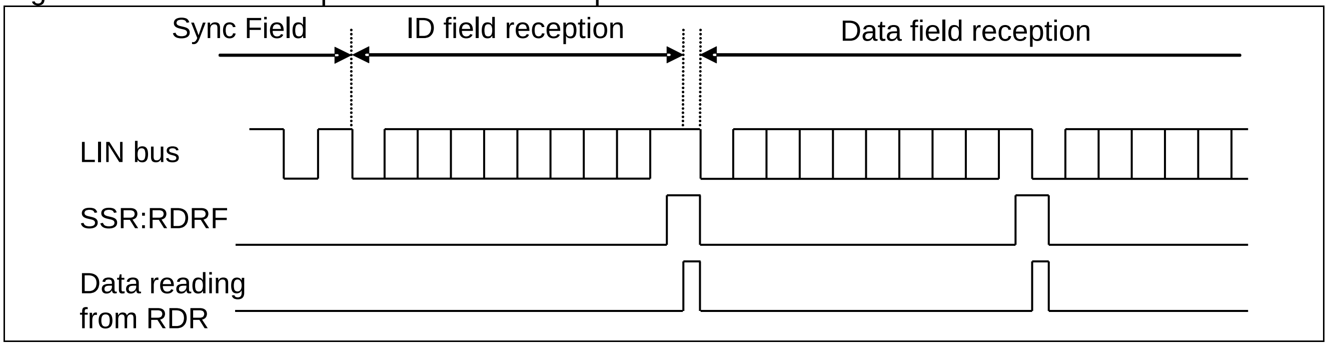
Figure 4-9 ID field reception-DATA field transmission



(To receive the DATA field)

- Every time the DATA field is received, SSR:RDRF is set to "1". If received interrupts are enabled (SCR:RDRF = 1) then, a received interrupt occurs.
- A start bit is detected when a falling edge is detected after data passes the noise filter (with the majority value applied after sampling serial data input three times with the bus clock) and a LOW level is detected for the data passing the sampling point.

Figure 4-10 ID field reception-DATA field reception



<Notes>

- The LIN interface (Ver. 2.1) includes noise filter (with the majority value applied after sampling serial data input three times with the bus clock). However, design the board so as not to allow noise to pass through this filter or perform communications so that any noise that has passed does not cause any problems (e.g., by adding a data checksum to the end and resending the data if any error occurs).
- During reception, if a falling edge of the serial data is detected concurrently with, or 1 to 2 bus clocks before the sampling point of the stop bit, the edge is ignored and the next data cannot be received successfully. To output frames continuously, adequate intervals should be considered between frames.

- **Slave mode operation timing chart**

Figure 4-11 LIN bus timing (when DATA field is transmitted and FIFO is not used)

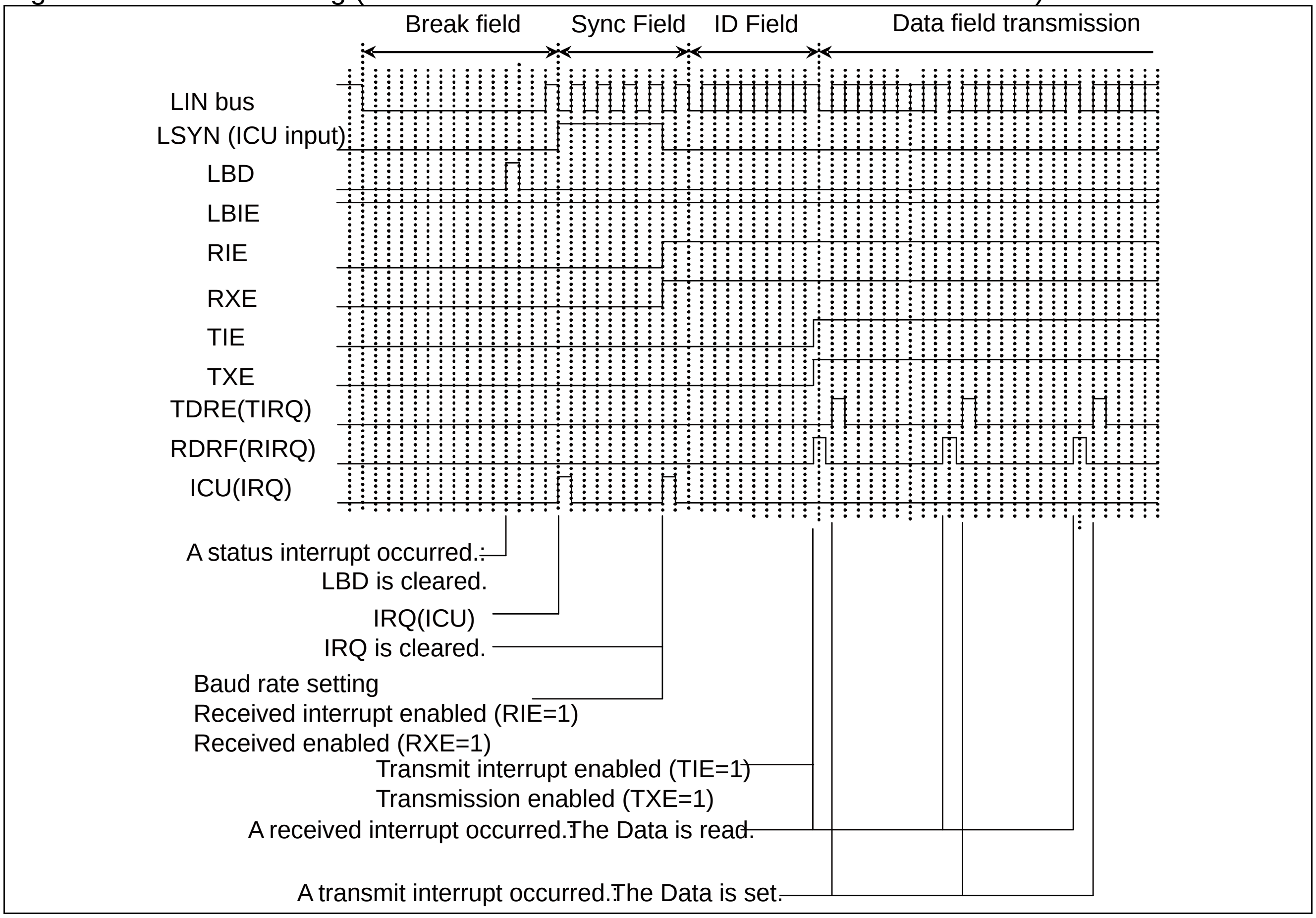
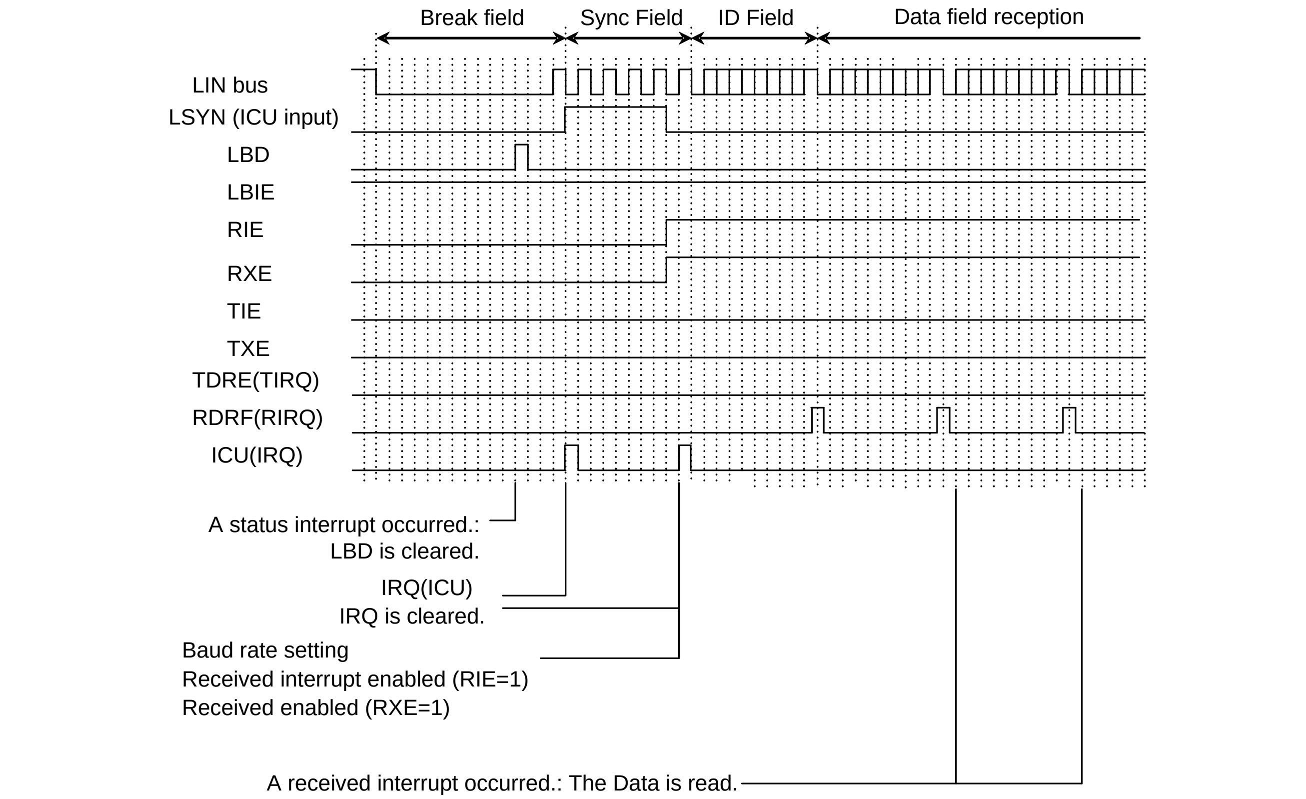


Figure 4-12 LIN bus timing (when DATA field is received and FIFO is not used)



● If FIFO is used

Figure 4-13 LIN bus timing (when DATA field is transmitted and FIFO is used)

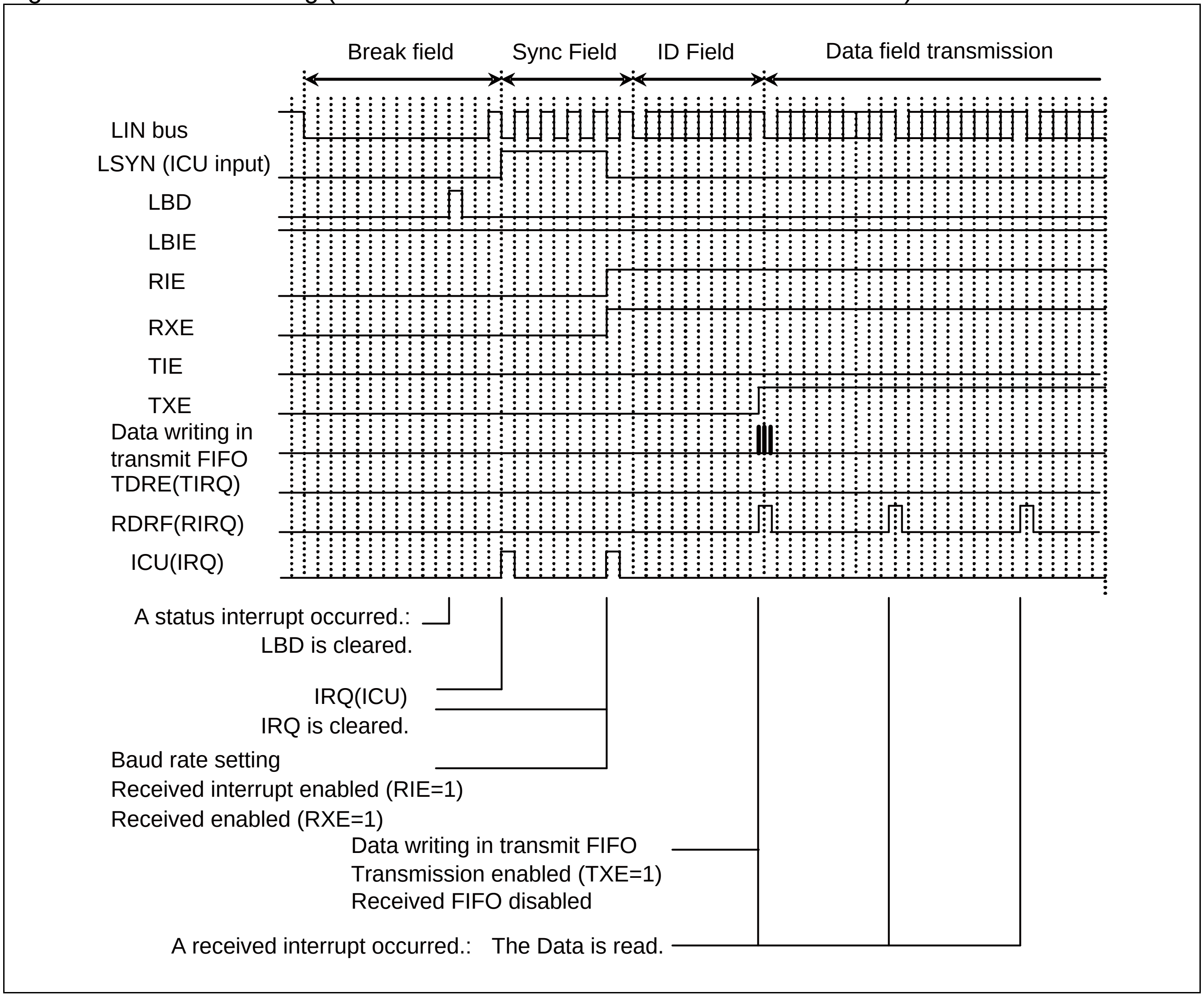
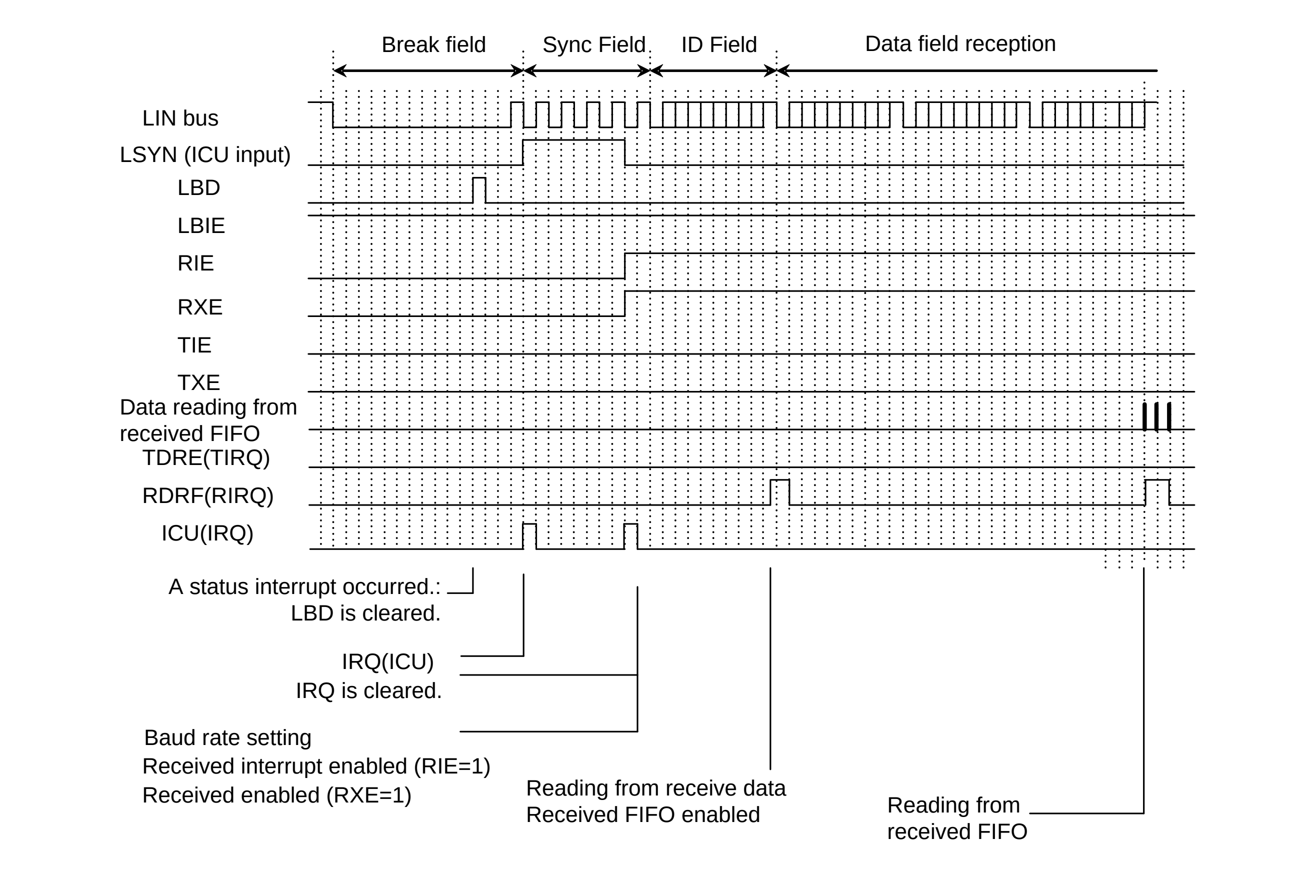


Figure 4-14 LIN bus timing (when DATA field is received and FIFO is used)



5. Operation Mode 3 (LIN Communication Mode) Setting Procedure and Program Flow

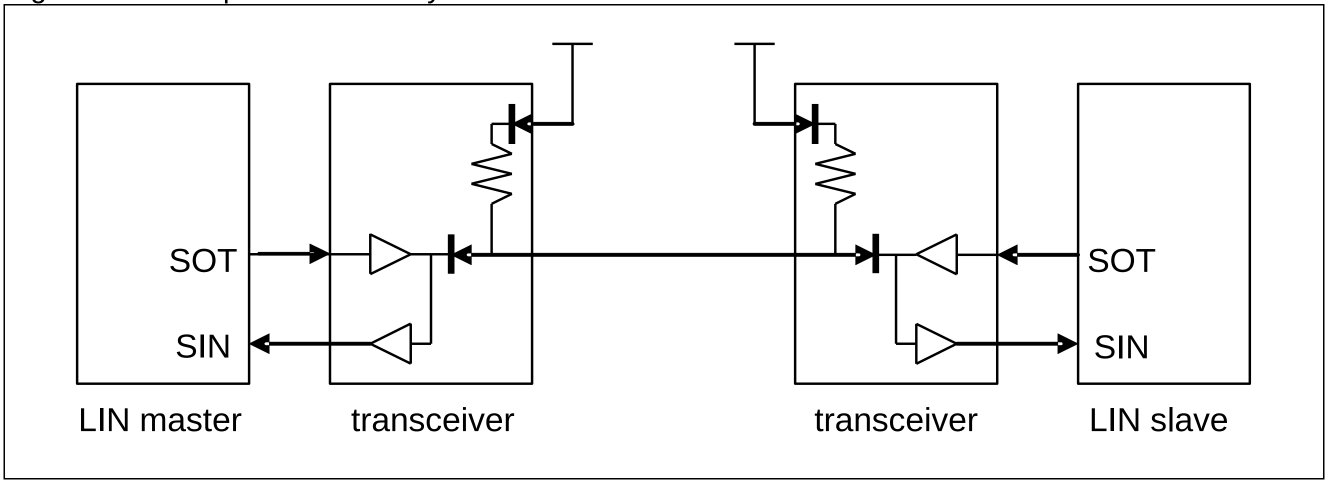
In Operation Mode 3 (LIN communication mode), the LIN interface (Ver. 2.1) can be used for a LIN master or LIN slave system.

■ Register settings

● CPU-to-CPU connection

Figure 5-1 shows a communication system consisting of one LIN master and one LIN slave. The LIN interface (ver. 2.1) can work as a LIN master or a LIN slave.

Figure 5-1 Example of LIN bus system communication



■ Example flowchart

● Master mode operations

Figure 5-2 Example flowchart of LIN communication in master mode (when FIFO is not used)

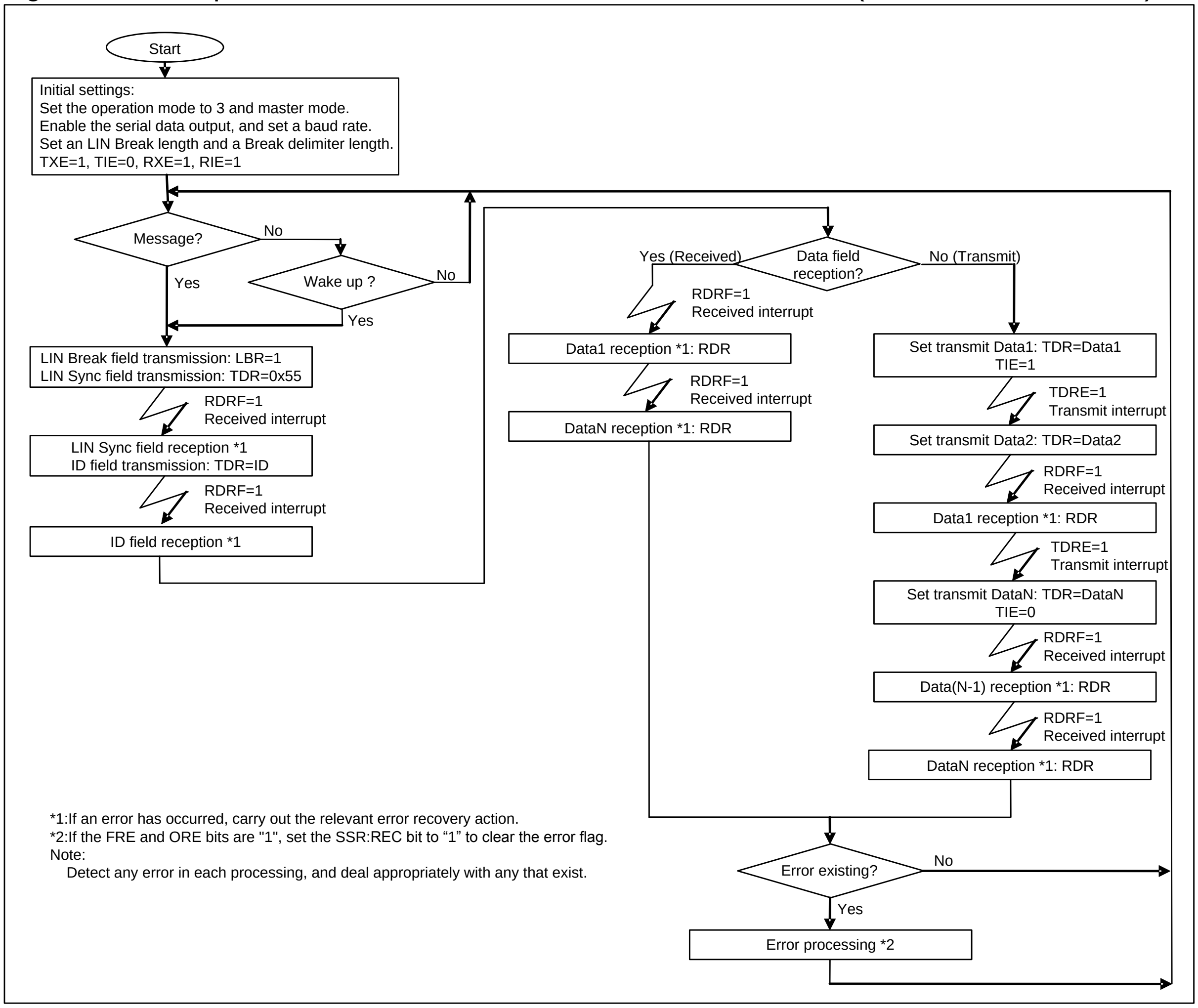
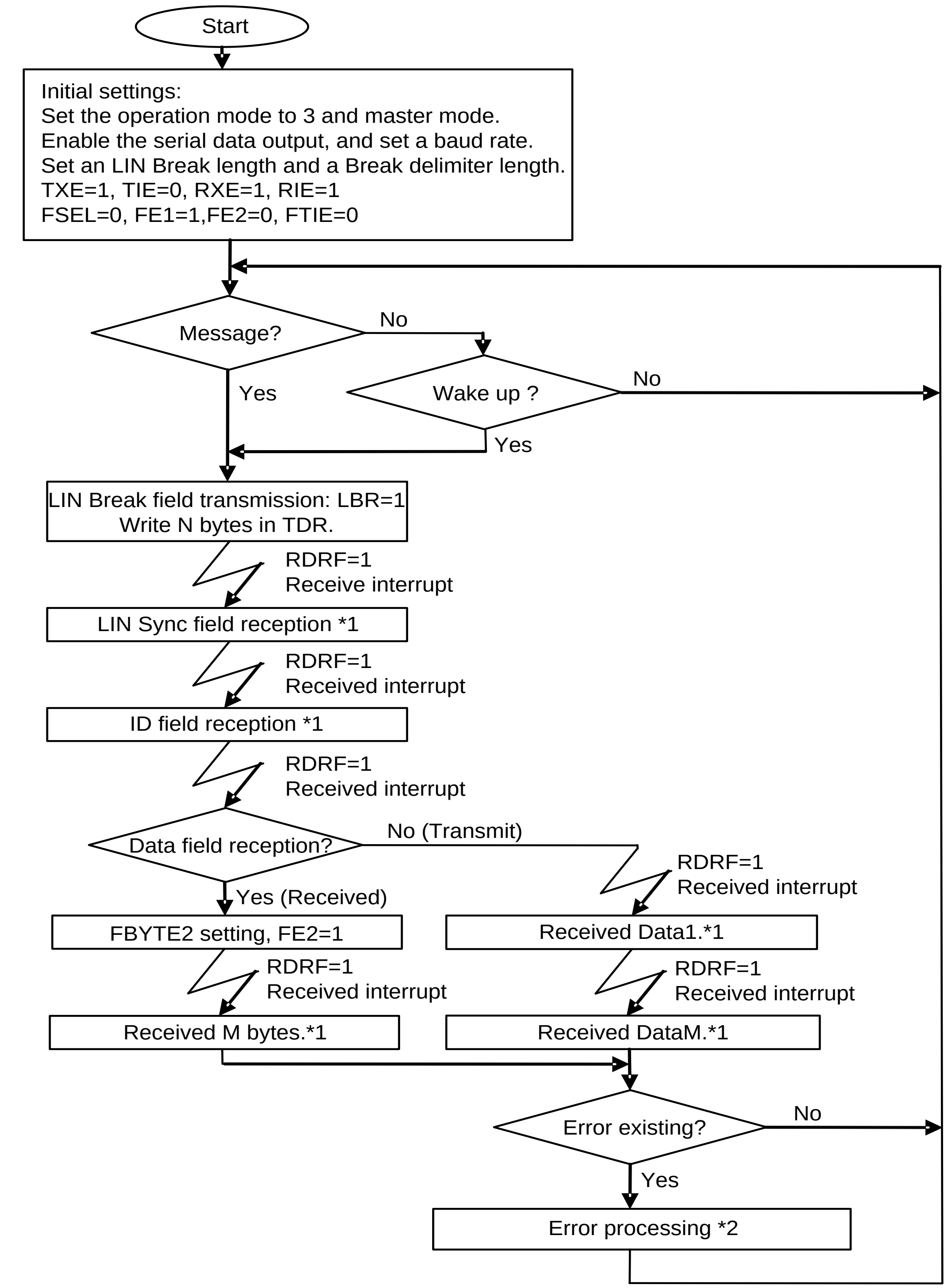


Figure 5-3 Example flowchart of LIN communication in master mode (when FIFO is used)



*1:If an error has occurred, carry out the relevant error recovery action.
*2:If the FRE and ORE bits are "1", set the SSR:REC bit to “1” to clear the error flag.
Note:
Detect any error in each processing, and deal appropriately with any that exist.

● Slave mode operations

Figure 5-4 Example flowchart of LIN communication in slave mode (when FIFO is not used)

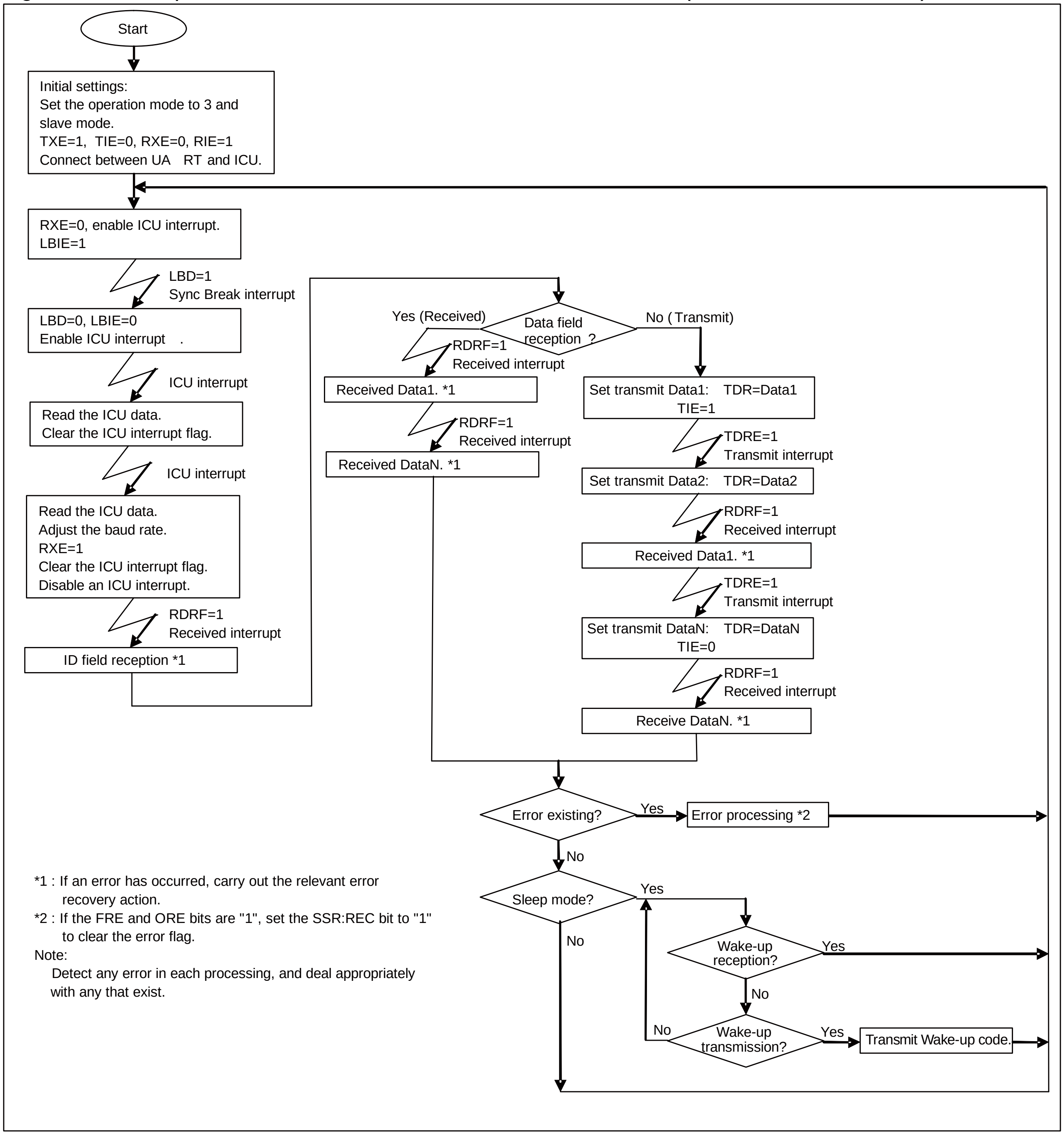
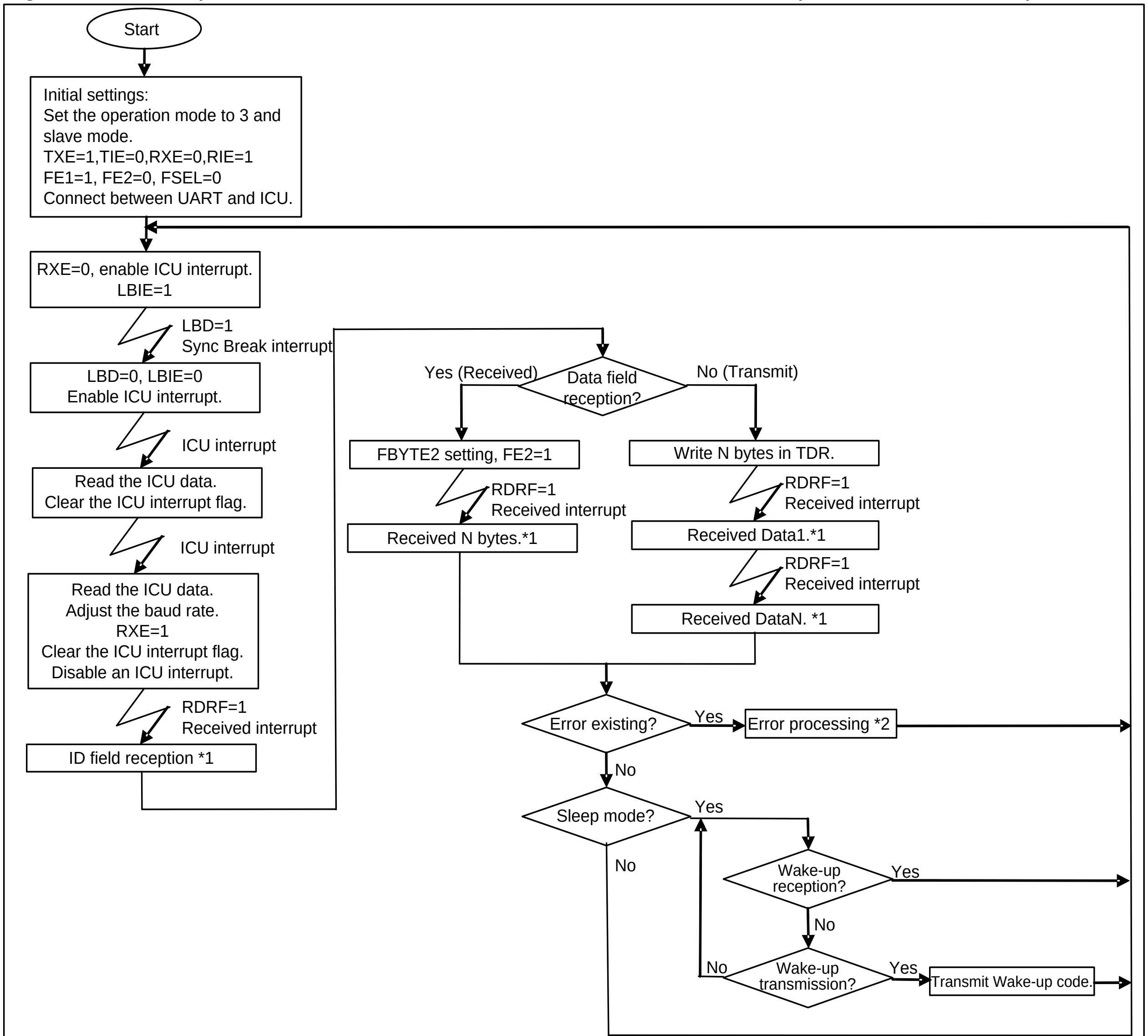


Figure 5-5 Example flowchart of LIN communication in slave mode (when FIFO is used)



*1:If an error has occurred, carry out the relevant error recovery action.
*2:If the FRE and ORE bits are “1”, set the SSR:REC bit to “1” to clear the error flag.
Note:
Detect any error in each processing, and deal appropriately with any that exist.

6. LIN Interface (ver. 2.1) Registers

The following shows a list of LIN interface (ver. 2.1) registers.

■ List of LIN interface (ver. 2.1) registers

Table 6-1 List of LIN interface (ver. 2.1) registers

	bit15 bit8	bit7 bit0
LIN interface (ver. 2.1)	SCR (Serial Control Register)	SMR (Serial Mode Register)
	SSR (Serial Status Register)	ESCR (Extended Communication Control Register)
	-	RDR/TDR (Transmit/Received Data Register)
	BGR1 (Baud Rate Generator Register 1)	BGR0 (Baud Rate Generator Register 0)
FIFO	FCR1 (FIFO Control Register 1)	FCR0 (FIFO Control Register 0)
	FBYTE2 (FIFO2 Byte Register)	FBYTE1 (FIFO1 Byte Register)

Table 6-2 LIN interface (ver. 2.1) bit assignment

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCR/ SMR	UPCL	MS	LBR	RIE	TIE	TBIE	RXE	TXE	MD2	MD1	MD0	-	SBL	-	-	SOE
SSR/ ESCR	REC	-	LBD	FRE	ORE	RDRF	TDRE	TBI	-	ESBL	-	LBIE	LBL1	LBL0	DEL1	DEL0
TDR/ RDR	-								D7	D6	D5	D4	D3	D2	D1	D0
BGR1	EXT	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
FCR1/ FCR0	-	-	-	FLSTE	FRIIE	FDRQ	FTIE	FSEL	-	FLST	FLD	FSET	FCL2	FCL1	FE2	FE1
FBYTE2/ FBYTE1	FD15	FD14	FD13	FD12	FD11	FD10	FD9	FD8	FD7	FD6	FD5	FD4	FD3	FD2	FD1	FD0

6.1. Serial Control Register (SCR)

The Serial Control Register (SCR) is used to enable/disable a transmit/received interrupt, enable/disable a transmit idle interrupt, and enable/disable data transmission and reception. Also, the SCR can be used to generate a LIN Break field and reset the LIN interface (ver. 2.1).

bit	15	14	13	12	11	10	9	8	7 ... 0
Field	UPCL	MS	LBR	RIE	TIE	TBIE	RXE	TXE	(SMR)
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	-	-	0	0	0	0	0	

[bit15] UPCL: Programmable clear bit

Initializes the internal state of LIN interface (ver. 2.1).

If set to "1":

- The LIN interface (ver. 2.1) is reset directly (Software reset). However, the current register settings are maintained. The transmit or received state is disconnected immediately.
- The baud rate generator reloads the BGR1/0 register value and restarts operation.
- All of transmit/received interrupt factors (SSR:TDRE, TBI, RDRF, FRE, ORE, LBD) are initialized.

If set to "0":

No effect on the operation.

"0" is always read.

Value	Description	
	At writing	At reading
0	No effect on the operation.	"0" is always read.
1	Programmable clear	

<Notes>

- Disable an interrupt first, and then execute the programmable clear instruction.
- If the FIFO operation is used, disable it (FCR0:FE[2:1:=00) first and then execute the programmable clear instruction.
- To switch from reception operation to transmit operation continuously, execute the programmable clear instruction after data is received and write transmit data to the Transmit Data Register (TDR).

[bit14] MS: Master/Slave function select bit

Selects the master or slave mode.

Value	Description
0	Master mode
1	Slave mode

[bit13] LBR: LIN Break Field setting bit (valid in master mode only)

If this bit is set to "1", a LIN Break field (having the length set by the ESCR:LBL1/LBL0 bit) is generated. Also, a LIN Break delimiter (set by the ESCR:DEL1/DEL0 bit) is generated.

When written:

When "0" is written: No effect on the operation.
When "1" is written: A LIN Break field is generated.

When read:

"0" is always read.

Value	Description	
	At writing	At reading
0	No effect on the operation.	"0" is always read.
1	A LIN Break field is generated.	

<Notes>

- This bit setting is valid in the master mode operation only (MS=0).
- Do not set this bit to "1" when a LIN Break field is being generated.

[bit12] RIE: Received interrupt enable bit

- This bit enables or disables an output of received interrupt request to the CPU.
- If the RIE bit and the received data flag bit (SSR:RDRF) are "1", or if any of the error flag bits (SSR:FRE, ORE) is "1", a received interrupt request is output.

Value	Description
0	Disables the received interrupt.
1	Enables the received interrupt.

[bit11] TIE: Transmit interrupt enable bit

- This bit enables or disables an output of transmit interrupt request to the CPU.
- If the TIE and SSR:TDRE bits are "1", a transmit interrupt request is output.

Value	Description
0	Disables a transmit interrupt.
1	Enables a transmit interrupt.

[bit10] TBIE: Transmit bus idle interrupt enable bit

- This bit enables or disables an output of transmit bus idle interrupt request to the CPU.
- If the TBIE bit and SSR:TBI bit are "1", a transmit bus idle interrupt request is output.

Value	Description
0	Disables the transmit bus idle interrupt.
1	Enables the transmit bus idle interrupt.

[bit9] RXE: Data reception enable bit

This bit enables or disables a data reception by the LIN interface (ver. 2.1).

Value	Description
0	Disables data frame reception.
1	Enables data frame reception.

<Notes>

- Data reception is not started unless a falling edge of the start bit is input even if the data reception is enabled (RXE=1).
- When a LIN Break field is being sent in the master mode operation, no data is received even if data reception is enabled (RXE=1).
- If data reception is disabled (RXE=0), the current data reception is stopped immediately.

[bit8] TXE: Data transmission enable bit

This bit enables or disables a data transmission by the LIN interface (ver. 2.1).

Value	Description
0	Disables data frame transmission.
1	Enables data frame transmission.

<Note>

If data transmission is disabled (TXE=0), the current data transmission is stopped immediately.

6.2. Serial Mode Register (SMR)

The Serial Mode Register (SMR) is used to set an operation mode, to select a transmission direction, data length, and stop bit length, and enable or disable an output of serial data to their pins.

bit	15 ... 8	7	6	5	4	3	2	1	0
Field	(SCR)	MD2	MD1	MD0	Reserved	SBL	Reserved		SOE
Attribute		R/W	R/W	R/W	-	R/W	-	-	R/W
Initial value		0	0	0	-	0	-	-	0

[bit7:5] MD2, MD1, MD0: Operation mode setting bits

These bits set an operation mode.

*This chapter explains the registers and their operation in operation mode 3 (LIN communication mode).

bit7	bit6	bit5	Description
0	0	0	Operation mode 0 (asynchronous normal mode)
0	0	1	Operation mode 1 (asynchronous multiprocessor mode)
0	1	0	Operation mode 2 (clock synchronous mode)
0	1	1	Operation mode 3 (LIN communication mode)
1	0	0	Operation mode 4 (I^2C mode)
Values other than the above			Setting is prohibited.

<Notes>

- Any bit setting other than above is inhibited.
- To switch the current operation mode, issue a programmable clear instruction (SCR:UPCL=1) and switch the operation mode continuously.
- After the operation mode has been set, set each register correctly.

[bit4] Rerved: Reserved bit

This bit value is undefined when read.

This bit has no effect on the operation when written.

[bit3] SBL: Stop bit length select bit

This bit sets a stop bit length (the frame end mark of the transmit data).

Value	Description	
0	ESCR:ESBL=0	Stop bit is set to 1 bit
	ESCR:ESBL=1	Stop bit is set to 3 bits
1	ESCR:ESBL=0	Stop bit is set to 2 bits
	ESCR:ESBL=1	Stop bit is set to 4 bits

<Notes>

- In reception operation, only the first bit of the stop bit data is detected.
- Always set this bit when transmission is disabled (SCR:TXE=0).

[bit2:1] Reserved: Reserved bits

The read value is "0". Be sure to write "0".

[bit0] SOE: Serial data output enable bit

This bit enables or disables a serial data output.

Value	Description
0	Disables a serial data output.
1	Enables a serial data output.

<Note>

If this bit is used as the SOT pin, the GPIO must also be set.

6.3. Serial Status Register (SSR)

The Serial Status Register (SSR) is used to check the current transmission/reception state, check the Received Error flag, detect a LIN Break field, and clear the Received Error flag.

bit	15	14	13	12	11	10	9	8	7 ... 0
Field	REC	-	LBD	FRE	ORE	RDRF	TDRE	TBI	(ESCR)
Attribute	R/W	-	R/W	R	R	R	R	R	
Initial value	0	-	0	0	0	0	1	1	

[bit15] REC: Received Error flag clear bit

This bit clears the FRE and ORE flags of the Serial Status Register (SSR).

Value	Description	
	Writing	Reading
0	No effect on the operation.	"0" is always read.
1	Clears the Received Error flag (FRE, ORE).	

[bit14] - : Unused bit

This bit value is undefined when read.
This bit has no effect on the operation when written.

[bit13] LBD: LIN Break field detection flag bit

This bit shows a detection of LIN Break field.
When 11-bit wide or more of serial input (SIN) are "LOW", the LBD bit is set to "1". If the LIN Break field interrupt enable bit (LBIE) is "1" during this time, a status interrupt occurs.

Value	Description	
	At writing	At reading
0	Clears the LBD flag.	A Break field was not detected.
1	No effect on the operation.	A Break field was detected.

<Note>

If a read-modify-write instruction is issued, "1" is read.

[bit12] FRE: Framing error flag bit

- If a framing error occurs during data reception, this bit is set to "1". If the REC bit of Serial Status Register (SSR) is set to "1", this flag is cleared.
- If the FRE and RIE bits are "1", a received interrupt request is output.
- If this flag is set, data of the Received Data Register (RDR) is invalid.
- If this flag is set when received FIFO is used, the received FIFO enable bit is cleared and the received data is not stored in received FIFO.

Value	Description
0	No framing error occurred.
1	A framing error occurred.

[bit11] ORE: Overrun error flag bit

- If an overrun occurs during data reception, this bit is set to "1". If the REC bit of Serial Status Register (SSR) is set to "1", this flag is cleared.
- If the ORE and RIE bits are "1", a received interrupt request is output.
- If this flag is set, data in the Received Data Register (RDR) is invalid.
- If this flag is set when received FIFO is used, the received FIFO enable bit is cleared and the received data is not stored in received FIFO.

Value	Description
0	No overrun error occurred.
1	An overrun error occurred.

[bit10] RDRF: Received data full flag bit

- This flag shows the state of Received Data Register (RDR).
- When the received data is loaded in the RDR, this bit is set to "1". When the Received Data Register (RDR) is read, this bit is cleared to "0".
- If the RDRF and RIE bits are "1", a received interrupt request is output.
- If received FIFO is used, the RDRF bit is set to "1" when the preset amount of data is received in received FIFO.
- If received FIFO is used, this bit is cleared to "0" when received FIFO is emptied.

Value	Description
0	The Received Data Register (RDR) is empty.
1	The Received Data Register (RDR) contains data.

[bit9] TDRE: Transmit data empty flag bit

- This flag shows the state of Transmit Data Register (TDR).
- If the transmit data is written in the TDR, this bit is set to "0" to indicate that the TDR contains valid data. When the data is loaded to the transmit shift register and when the transmission is started, this bit is set to "1" to indicate that the TDR does not contain the valid data.
- If the TDRE and TIE bits are "1", a transmit interrupt request is output.
- When the UPCL bit of Serial Control Register (SCR) is set to "1", the TDRE bit is set to "1".
- For the TDRE bit set/clear timing when transmit FIFO is used, see "2.4 Interrupt and flag set timing when transmit FIFO is used".

Value	Description
0	The Transmit Data Register (TDR) contains data.
1	The Transmit Data Register (TDR) is empty.

[bit8] TBI: Transmit bus idle flag bit

- This bit indicates that the LIN interface (ver. 2.1) is not transmitting data.
- When transmit data is written in the Transmit Data Register (TDR), this bit is set to "0".
- When the LIN Break field is set (SMR:LBR=1), this bit is set to "0".
- If the Transmit Data register (TDR) is empty (TDRE=1) and if no transmission is started, this bit is set to "1".
- If the Transmit Data Register is emptied after the LIN Break field has been transmitted, this bit is set to "1".
- If this bit is "1" and if a transmit bus idle interrupt is enabled (SCR:TBIE=1), a transmit interrupt request is output.

Value	Description
0	Data being transmitted
1	No data transmission

6.4. Extended Communication Control Register (ESCR)

The Extended Communication Control Register (ESCR) is used to enable/disable a LIN Break field interrupt, detect a LIN Break field, set a LIN Break field length and a Break delimiter length, and select a stop bit length.

bit	15 ... 8	7	6	5	4	3	2	1	0
Field	(SSR)	Reserved	ESBL	-	LBIE	LBL1	LBL0	DEL1	DEL0
Attribute		-	R/W	-	R/W	R/W	R/W	R/W	R/W
Initial value		0	0	-	0	0	0	0	0

[bit7] Reserved: Reserved bit

The read value is "0". Be sure to write "0".

[bit6] ESBL: Extended stop bit length select bit

This bit sets a stop bit length (the frame end mark of the transmit data).

Value	Description	
0	SMR:SBL=0	Stop bit length is set to 1 bit
	SMR:SBL=1	Stop bit length is set to 2 bits
1	SMR:SBL=0	Stop bit length is set to 3 bits
	SMR:SBL=1	Stop bit length is set to 4 bits

<Notes>

- In reception operation, only the first bit of the stop bit data is detected.
- Always set this bit when transmission is disabled (TXE=0).

[bit5] - : Unused bit

This bit value is undefined when read.

This bit has no effect on the operation when written.

[bit4] LBIE: LIN Break field detect interrupt enable bit

This bit enables or disables a LIN Break field detect interrupt.

If the LIN Break field detect flag (LBD) is "1", a received interrupt occurs when an interrupt is enabled (LBIE=1).

Value	Description
0	Disables a LIN Break field detect interrupt.
1	Enables a LIN Break field detect interrupt.

[bit3:2] LBL1/LBL0: LIN Break field length select bits (valid in master mode only)

- These bits set a LIN Break field generation time (in number of bits).
- This bit must be set before the LBR bit of Serial Control Register (SCR) is set to "1" (for LIN Break field transmission).
- A LIN Break field is always detected at the 11th bit in the slave mode operation regardless of this bit setting.

bit3	bit2	Description
0	0	13 bits long
0	1	14 bits long
1	0	15 bits long
1	1	16 bits long

<Note>

This bit setting is valid in the master mode operation only (SMR:MS="0").

[bit1:0] DEL1/DEL0: LIN Break delimiter length select bits (valid in master mode only)

- These bits set a LIN Break delimiter length (in number of bits).
- These bits must be set before the LBR bit of Serial Control Register (SCR) is set to "1" (for LIN Break field transmission).

bit1	bit0	Description
0	0	1 bit long
0	1	2 bits long
1	0	3 bits long
1	1	4 bits long

<Note>

This bit setting is valid in the master mode operation only (SMR:MS=0).

6.5. Received Data Register/Transmit Data Register (RDR/TDR)

The Received and Transmit Data Registers are allocated at the same address. This register functions as the Received Data Register when data is read from it. This register functions as the Transmit Data Register when data is written in it.

■ Received Data Register (RDR)

bit	15 ... 8	7	6	5	4	3	2	1	0
Field		D7	D6	D5	D4	D3	D2	D1	D0
Attribute		R	R	R	R	R	R	R	R
Initial value		0	0	0	0	0	0	0	0

The Received Data Register (RDR) is a data buffer register for serial data reception.

- When serial data signals are sent to the Serial Input pin (SIN), they are converted by a shift register and stored in the Received Data Register (RDR).
- When the received data is stored in the Received Data Register (RDR), the received data full flag bit (SSR:RDRF) is set to "1". If a received interrupt is enabled (SSR:RIE=1), a received interrupt request is generated.
- The Received Data Register (RDR) must be read only when the received data full flag bit (SCR:RDRF) is "1". When data is read from the Serial Received Data Register (RDR), the received data full flag bit (SSR:RDRF) is cleared to "0" automatically.
- If a received error occurs (when SSR:ORE or FRE is "1"), data in the Received Data Register (RDR) becomes invalid.

<Notes>

- If received FIFO is used and if the preset amount of data is received in received FIFO, the RDRF bit is set to "1".
- If received FIFO is used and if this buffer is emptied, the RDRF bit is cleared to "0".
- If a received error occurs when received FIFO is used (SSR:ORE or FRE is "1"), the received FIFO enable bit is cleared and the received data is not stored in received FIFO.

■ Transmit Data Register (TDR)

bit	15 ... 8	7	6	5	4	3	2	1	0
Field		D7	D6	D5	D4	D3	D2	D1	D0
Attribute		W	W	W	W	W	W	W	W
Initial value		1	1	1	1	1	1	1	1

The Transmit Data Register (TDR) is a data buffer register for serial data transmission.

- If data transmission is enabled (SCR:TXE=1) and if the transmit data is written in the Transmit Data Register (TDR), the transmit data is transferred to the transmit shift register. Then, the data is converted into serial data, and output at the serial data output pin (SOT).
- When the transmit data is written in the Transmit Data Register (TDR), the transmit data empty flag (SSR:TDRE) is cleared to "0".
- When the transmit data is transferred to the serial transmit shift register and data transmission is started, and if transmit FIFO is disabled or if transmit FIFO is empty, the transmit data empty flag (SSR:TDRE) is set to "1".
- If the transmit data empty flag (SSR:TDRE) is "1", the next transmit data can be written in the buffer. If a transmit interrupt is enabled, a transmit interrupt occurs. The next transmit data must be written only after the transmit interrupt has occurred or when the transmit data empty flag (SSR:TDRE) is "1".
- If the transmit data empty flag (SSR:TDRE) is "0" and transmit FIFO is disabled or transmit FIFO is full, no transmit data can be written in the Transmit Data Register (TDR).

<Notes>

- The Transmit Data Register is a write-only register. While the Received Data Register is a read-only register. As these two registers are allocated at the same address, the write and read values differ from each other. Therefore, the INC/DEC instruction and other read-modify-write (RMW) operation cannot be used.
- For the transmit data empty flag (SSR:TDRE) set timing when transmit FIFO is used, see "2.4 Interrupt and flag set timing when transmit FIFO is used".

6.6. Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0)

Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0) are used to set a frequency division ratio of serial clocks. Also, an external clock can be selected as the clock source of the reload counter.

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	EXT	(BGR1)							(BGR0)							
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- The Baud Rate Generator Registers are used to set a frequency division ratio of serial clocks.
- The BGR1 register corresponds to the high-order bits, and the BGR0 register corresponds to the low-order bits. The reload value to be counted can be written, and the BGR1/BGR0 set value can be read.
- When the reload value is written in Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0), the reload counter starts its counting.
- The EXT bit (bit15) specifies to use the clock source of reload counter as the internal clock or the external clock. If EXT=0 is set, an internal clock is used. If EXT=1 is set, an external clock is used.

[bit15] EXT: External clock select bit

Value	Description
0	Uses the internal clock.
1	Uses an external clock.

[bit14:8] BGR1: Baud Rate Generator Register 1

Process	Description
Write	Writes data in bit8 to bit14 of reload counter.
Read	Reads the BGR1 set value.

[bit7:0] BGR0: Baud Rate Generator Register 0

Process	Description
Write	Writes data in bit0 to bit7 of reload counter.
Read	Reads the BGR0 set value.

<Notes>

- Data must be written in the Baud Rate Generator Register1, 0 (BGR1 and BGR0) in 16-bit data access mode.
- If the current values of Baud Rate Generator Register1, 0 (BGR1, BGR0) are changed, the new values are reloaded only after the counter value has reached "15h00". In order to validate the new set values immediately, change the BGR1/BGR0 set values and execute the programmable clear (UPCL).
- If the reload value is even, the "LOW" signal width of serial clock is longer than the "HIGH" signal width for a single cycle of bus clock. If the value is odd, the serial clock has the same "HIGH" and "LOW" signal width.
- Set the reload value to 3 or more. Note that data may not be received normally due to the baud rate error and reload value setting.
- When the baud rate generator is operating and if you need to switch to the external clock (EXT=1), first set the baud rate generators 1 and 0 (BGR1 and BGR0) to "0". Then, execute the programmable clear instruction (UPCL) and select the external clock (EXT=1).

6.7. FIFO Control Register 1 (FCR1)

The FIFO Control Register (FCR1) is used to set the FIFO test, select transmit or received FIFO, enable transmit FIFO interrupt, and control the interrupt flag.

bit	15	14	13	12	11	10	9	8	7 ... 0
Field	Reserved			FLSTE	FRIIE	FDRQ	FTIE	FSEL	(FCR0)
Attribute	-	-	-	R/W	R/W	R/W	R/W	R/W	
Initial value	-	-	-	0	0	1	0	0	

[bit15:13] Reserved: Reserved bits

The read value is "0". Be sure to write "0".

[bit12] FLSTE: Re-transmit data lost detect enable bit

This bit enables the FLST bit detection.

Value	Description
0	Disables the Data Lost detection.
1	Enables the Data Lost detection.

<Note>

If you wish to set this bit to "1", set the FSET bit to "1" first, and then set this bit to "1".

[bit11] FRIIE: Received FIFO idle detect enable bit

This bit sets to detect the received idle state if received FIFO contains valid data for more than 8-bit hours. If the received interrupt is enabled (SCR:RIE=1), a received interrupt is generated when the received idle state is detected.

Value	Description
0	Disables the received FIFO idle detection.
1	Enables the received FIFO idle detection.

<Note>

In case of using Received FIFO, set this bit to "1".

[bit10] FDRQ: Transmit FIFO data request bit

This bit requests for the transmit FIFO data.
If this bit is "1", the transmit data is being requested. If the Transmit Interrupt is enabled (FTIE=1) during this time, a transmit FIFO interrupt request is output.

The FDRQ bit is set when:

- The FBYTE (for transmission) is "0" (Transmit FIFO is empty).
- Transmit FIFO is reset.

The FDRQ bit is cleared when:

- This bit is set to "0".
- Transmit FIFO is filled with data.

Value	Description
0	Does not request for the transmit FIFO data.
1	Requests for the transmit FIFO data.

<Notes>

- If the FBYTE (for transmission) is "0", this bit cannot be set to "0".
- If this bit is "0", the FSEL bit state cannot be changed.
- If this bit is set to "1", it has no effect on the operation.
- If a read-modify-write instruction is issued, "1" is read.

[bit9] FTIE: Transmit FIFO interrupt enable bit

This bit enables a transmit FIFO interrupt. If this bit is set to "1", an interrupt occurs when the FDRQ bit is set to "1".

Value	Description
0	Disables the transmit FIFO interrupt.
1	Enables the transmit FIFO interrupt.

[bit8] FSEL: FIFO select bit

This bit selects the transmit or received FIFO.

Value	Description
0	Transmit FIFO:FIFO1; Received FIFO:FIFO2
1	Transmit FIFO:FIFO2; Received FIFO:FIFO1

<Notes>

- This bit is not cleared by FIFO reset (FCR0:FCL[2:1]=11).
- To change this bit state, first disable the FIFO operation (FCR0:FE[2:1]=00).

6.8. FIFO Control Register 0 (FCR0)

FIFO Control Register 0 (FCR0) is used to enable/disable the FIFO operation, reset FIFO, save the read pointer, and set the data re-transmission.

bit	15 ... 8	7	6	5	4	3	2	1	0
Field	(FCR1)	-	FLST	FLD	FSET	FCL2	FCL1	FE2	FE1
Attribute		-	R	R/W	W	R/W	R/W	R/W	R/W
Initial value		-	0	0	0	0	0	0	0

[bit7] - : Unused bit

This bit value is undefined when read.
This bit has no effect on the operation when written.

[bit6] FLST: FIFO re-transmit data lost flag bit

This bit shows that the re-transmit data of transmit FIFO has been lost.

The FLST bit is set when:

- The FLSTE bit of FIFO Control Register 1 (FCR1) is "1", the write pointer of transmit FIFO matches the read pointer which has been saved by the FSET bit, and data is written in FIFO.

The FLST bit is cleared when:

- FIFO is reset (FCL bit is set to "1").
- The FSET bit is set to "1".

If this bit is set to "1", the data identified by the read pointer (saved by the FSET bit) is overwritten. Therefore, the FLD bit cannot set the data re-transmission even if an error has occurred. If this bit is set to "1" and if you wish to re-transmit data, first reset FIFO. Then, write data in FIFO again.

Value	Description
0	No Data Lost has occurred.
1	Data Lost has occurred.

[bit5] FLD: FIFO pointer reload bit

This bit reloads the data, being saved in transmit FIFO by the FSET bit, to the reload pointer. This bit can be used to re-transmit data after a communication error or others have occurred.
When the re-transmission setting has finished, this bit is set to "0".

Value	Description
0	Not reloaded
1	Reloaded

<Notes>

- If this bit is "1", data is being reloaded in the read pointer. Therefore, data writing except for FIFO reset is disabled.
- When FIFO is enabled or when data is being transmitted, this bit cannot be set to "1".
- After you have set the TIE and TBIE bits to "0", set this bit to "1". After you have enabled transmit FIFO, set the TIE and TBIE bits to "1".

[bit4] FSET: FIFO pointer save bit

This bit saves the transmit FIFO read pointer.
If the read pointer is saved before transmission and if the FLST bit is "0", data can be re-transmitted even when a communication error or others occur.

Value	Description
0	Not saved
1	Saved

<Note>

This bit can be set to "1" only when the transmit byte count (FBYTE) is "0".

[bit3] FCL2: FIFO2 reset bit

This bit resets the FIFO2 value.
If this bit is set to "1", the FIFO2 internal state is initialized.
Only the FCR1:FLST2 bit is initialized, but the other bits of FCR1/FCR0 registers are kept.

Value	Description	
	Writing	Reading
0	No effect on the operaion.	"0" is always read.
1	FIFO2 is reset.	

<Notes>

- Disable the transmission and reception first, and then reset FIFO2.
- Set the transmit FIFO interrupt enable bit to "0" before the execution.
- The valid data count of the FBYTE2 register is set to "0".

[bit2] FCL1: FIFO1 reset bit

This bit resets the FIFO1 value.
If this bit is set to "1", the FIFO1 internal state is initialized.
Only the FCR1:FLST1 bit is initialized, but the other bits of FCR1/FCR0 registers are kept.

Value	Description	
	Writing	Reading
0	No effect on the operation.	"0" is always read.
1	FIFO1 is reset.	

<Notes>

- Disable the transmission and reception first, and then reset FIFO1.
- Set the transmit FIFO interrupt enable bit to "0" before the execution.
- The valid data count of the FBYTE1 register is set to "0".

[bit1] FE2: FIFO2 operation enable bit

This bit enables or disables the FIFO2 operation.

- To use the FIFO2 operation, set this bit to "1".
- If FIFO2 is set as transmit FIFO and if data exists in FIFO2 when this bit is set to "1", the data transmission starts immediately when the LIN interface (ver. 2.1) is enabled to transmit data (TXE=1). During this time, set both TIE and TBIE bits to "0". Then, set this bit to "1" and set both TIE and TBIE bits to "1".
- If received FIFO is selected by the FSEL bit and if a received error has occurred, this bit is cleared to "0". This bit cannot be set to "1" until the received error is cleared.
- If FIFO2 is used as transmit FIFO, this bit must be set to "1" or "0" when the transmit buffer is empty (TDRE=1).
- If FIFO2 is used as received FIFO, this bit must be set to "0" when the received buffer is empty (SSR:RDRF=0) and no valid data exists in received FIFO (FBYTE2=0x00) after reception is disabled (SCR:RXE=0).
- If FIFO2 is used as received FIFO, this bit must be set to "1" when the received buffer is empty (SSR:RDRF=0) after reception is disabled (SCR:RXE=0).
- The FIFO2 state is held even if the FIFO2 operation is disabled.

Value	Description
0	Disables the FIFO2 operation.
1	Enables the FIFO2 operation.

[bit0] FE1: FIFO1 operation enable bit

This bit enables or disables the FIFO1 operation.

- To use the FIFO1 operation, set this bit to "1".
- If FIFO1 is set as transmit FIFO and if data exists in FIFO1 when this bit is set to "1", the data transmission starts immediately when the LIN interface (ver. 2.1) is enabled to transmit data (TXE=1). During this time, set both TIE and TBIE bits to "0". Then, set this bit to "1" and set both TIE and TBIE bits to "1".
- If received FIFO is selected by the FSEL bit and if a received error has occurred, this bit is cleared to "0". This bit cannot be set to "1" until the received error is cleared.
- If FIFO1 is used as transmit FIFO, this bit must be set to "1" or "0" when the transmit buffer is empty (TDRE=1).
- If FIFO1 is used as received FIFO, this bit must be set to "0" when the received buffer is empty (SSR:RDRF=0) and no valid data exists in received FIFO (FBYTE2=0x00) after reception is disabled (SCR:RXE=0).
- If FIFO1 is used as received FIFO, this bit must be set to "1" when the received buffer is empty (SSR:RDRF=0) after reception is disabled (SCR:RXE=0).
- The FIFO1 state is held even if the FIFO1 operation is disabled.

Value	Description
0	Disables the FIFO1 operation.
1	Enables the FIFO1 operation.

6.9. FIFO Byte Register (FBYTE)

The FIFO Byte Register (FBYTE) indicates the effective data count in the FIFO buffer. Also, this register can be used to generate a received interrupt when a certain number of data sets is received in the received FIFO.

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	(FBYTE2)								(FBYTE1)							
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The FBYTE register indicates the effective data count of FIFO. The following shows the settings of the FCR1:FSEL bit.

Table 6-3 Display of data count

FCR1:FSEL	FIFO selection	Data count display
0	FIFO2:Received FIFO, FIFO1:Transmit FIFO	FIFO2:FBYTE2, FIFO1:FBYTE1
1	FIFO2:Transmit FIFO, FIFO1:Received FIFO	FIFO2:FBYTE2, FIFO1:FBYTE1

- The initial value of data transfer count is "0x08" for the FBYTE register.
- Set a data count to the FBYTE register of received FIFO to generate a received interrupt flag. If this transfer data count matches the FBYTE register display, the received data full flag bit (RDRF) is set to "1".
- If both conditions below are satisfied and if the received idle state continues for more than 8 baud rate clocks, the received data full flag (SSR:RDRF) is set to "1".
  - The received FIFO idle detect enable bit (FRIIE) is "1".
  - The number of data sets stored in the received FIFO does not reach the transfer count.

  If the RDR data is read during counting of 8 clocks, this counter is reset to "0", and counting for 8 clocks is restarted. If received FIFO is disabled, this counter is reset to "0". If data remains in received FIFO and if received FIFO is enabled, the data counting is restarted.

[bit15:8] FBYTE2: FIFO2 data count display bits

[bit7:0] FBYTE1: FIFO1 data count display bits

Writing	Sets the transfer data count.
Reading	Reads the effective count of data.

Read (Effective data count)

During transmission: The number of data sets already written in FIFO but not transmitted yet
During reception: The number of data sets received in FIFO

Write (Transfer data count)

During transmission: Set "0x00".
During reception: Set the data count to generate a received interrupt.

<Notes>

- Set "0x00" in the FBYTE register of transmit FIFO.
- Set data equal to or greater than "1" in the FBYTE register of received FIFO.
- This state can be changed only after the data transmission or reception has been disabled.
- A read-modify-write instruction cannot be used for this register.
- Any setting exceeding the FIFO capacity is inhibited.
- After setting FIFO select bit (FCR1:FSEL), set FIFO byte register (FBYTE).
- FIFO select bit (FCR1:FSEL) and FIFO byte register (FBYTE) cannot be set at the same time.
- In the FIFO data count display at transmit, the data count which is made by subtracting "1" from transmit data written count is displayed. This is because data transmitted is written to be saved in transmit FIFO when the data not transmitted to TDR register exists. When data in TDR register is transmitted, the data not transmitted in transmit FIFO is transferred to TDR register.
- In the FIFO data count display at receive, the count of data which is received but not read is displayed. The data under receiving at TDR register is no included.

CHAPTER 1-5: I^{2}C Interface (I^{2}C Communications Control Interface)

This chapter explains the I^{2}C function supported in operation mode 4 of the multifunction serial interface.

CODE: 9BFI2C-E02.0 FM15I-E05.5

1. Overview of I^{2}C Interface (I^{2}C Communications Control Interface)

The I^{2}C interface (I^{2}C communications control interface) supports the I^{2}C bus and operates as a master/slave device on the I^{2}C bus. It also has transmit/received FIFO (up to 128 × 9 bits each) [*1]installed.

■ Functions of I^{2}C interface (I^{2}C communications control interface)

		Function
1	Data buffer	· Full duplex double buffer (when FIFO is not used) · Transmit/received FIFO (max 128 × 9 bits each) * (when FIFO is used)
2	Serial input	Removes noise up to 2 clocks in the bus clock for serial clock/serial data input.
3	Transfer mode	Synchronous
4	Baud rate	· A dedicated baud rate generator (constructed with a 15-bit reload counter) · The external clock can be adjusted with the reload counter.
5	Data length	8 bits
6	Signaling system	NRZ (Non Return to Zero)
7	Interrupt request	· Received interrupt · Transmit interrupt · Request of status interrupt/interrupt to ICU · Transmit FIFO interrupt (when transmit FIFO is empty) · DMA(Transmit/Received) transferring support function is available.
8	I^{2}C	· Master/slave transmission and reception functions · Arbitration function · Clock synchronization function · Transmission direction detection function · Function to generate and detect iteration start condition · Bus error detection function · General call addressing function · 7-bit addressing as master/slave · Generation of interrupt enabled during transmission or a bus error · The 10-bit addressing function can be programmatically enabled.
9	FIFO	· Transmit/received FIFO installed (maximum capacity: 128 × 9 bits for transmit FIFO, 128 × 9 bits for received FIFO) * · Transmit FIFO or received FIFO can be selected. · Transmit data can be resent. · Received FIFO interrupt timing can be changed via software. · FIFO resetting is supported independently.

* : The FIFO capacity size varies depending on the product type.

2. I²C Interface Interrupt

I²C interface interrupt request is generated due to the following factors.
- After transmission/reception of the first byte and after data transmission/reception is completed
- Stop condition
- Iteration start condition
- FIFO transmit data request
- FIFO received data completed

■ I²C Interface Interrupt

Table 2-1 shows the interrupt control bits and interrupt factors for the I²C interface.

Table 2-1 Interrupt control bits and interrupt factors for the I²C interface

Interrupt type	Interrupt request flag bit	Flag register	Interrupt factor	Interrupt factor enable bit	Operation to clear interrupt request flag
Status	INT	IBCR	The first byte has been transmitted/received[*1] (except for master operation when SSR:DMA=1)	IBCR:INTE	Setting the interrupt flag bit (IBCR:INT) to "0"
			Data has been transmitted/received[*1] (When SSR:DMA=0)		
			Bus Error detection (EIBCR:BCE=0)		
			Detection of arbitration lost		
			Detection of reserved address		
			Reception of NACK		
			Received FIFO being full during reception as a slave (When SSR:DMA=0)		Setting IBCR:INT to "0" after reading received data until received FIFO is emptied
	SPC	IBSR	Stop condition	IBCR:CNDE	Setting SPC to "0"
	RSC		Detection of iteration start		Setting RSC to "0"
Reception	RDRF	SSR	Reception of reserved address	SMR:RIE	Reading from the received data register (RDR)
			Completion of data reception		
			Reception of a data volume matching the value set for FBYTE.		Reading from the Received Data Register (RDR) until received FIFO is emptied
			Detection of reception idling when FRIIE=1		
	ORE	SSR	Overrun error		Setting the reception error flag bit (SSR:REC) to "1"

Interrupt type	Interrupt request flag bit	Flag register	Interrupt factor	Interrupt factor enable bit	Operation to clear interrupt request flag
Transmission	TDRE	SSR	The Transmit Data Register is empty.	SMR:TIE	Writing to the Transmit Data Register (TDR) or setting the transmit FIFO operation enable bit to "1" when the transmit FIFO operation enable bit is set to "0" and valid data are present in transmit FIFO (re-transmitting data) *2
			Setting the transmit buffer empty flag set bit (SSR:TSET) to "1"		
	FDRQ	FCR1	Transmit FIFO is empty.	FCR1:FTIE	The FIFO transmit data request bit is set to "0" or transmit FIFO is full.
	TBI (SSR: DMA=1)	SSR	No transmission operation	SCR:TBIE	Writing to the Transmit Data Register (TDR) or setting the transmit FIFO operation enable bit to "1" when the transmit FIFO operation enable bit is set to "0" and valid data are present in transmit FIFO (re-transmitting data) *3
			Setting the transmit buffer empty flag set bit (SSR:TSET) to "1"		

*1 : If normal data can be transmitted/received and SSR:TDRE is "0", no interrupt is generated. This is to support DMA transfers.
To generate the IBCR:INT bit at a time of data transmission/reception, the SSR:TDRE bit needs to be set to "1" before the IBCR:INT bit is set.

*2 : Be sure to check that the SSR:TDRE bit is set to "0" and then set the SMR:TIE bit to "1".

*3 : Be sure to check that the SSR:TBI bit is set to "0" and then set the SSR:TBIE bit to "1".

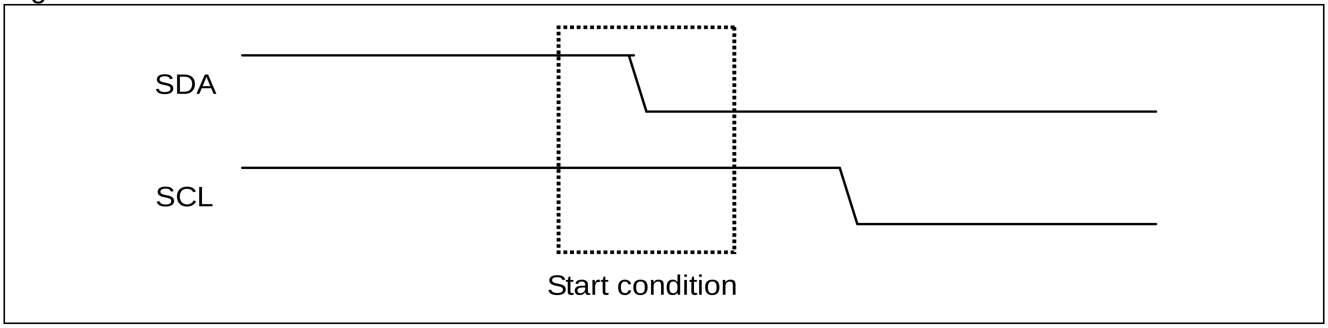
2.1. I²C interface operation

The I²C interface performs communications using two two-way bus lines, a serial data line (SDA) and a serial clock line (SCL).

■ I²C bus start condition

The following shows the I²C bus start condition.

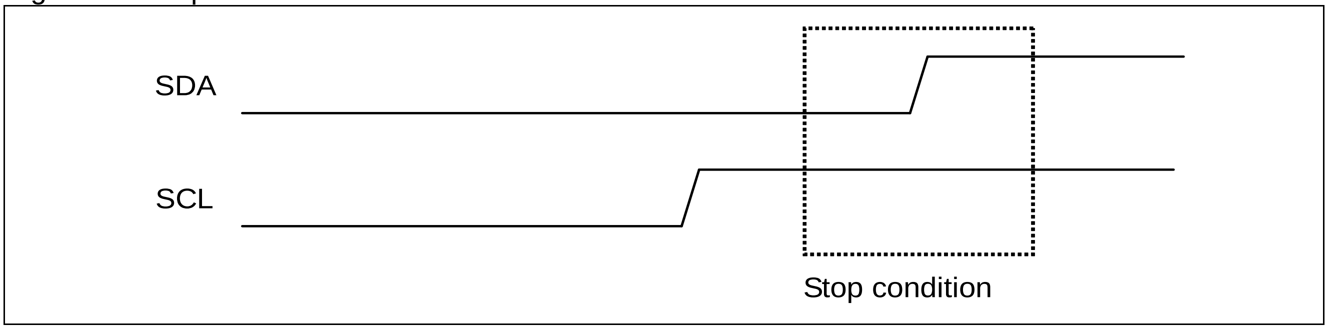
Figure 2-1 Start condition



■ I²C bus stop condition

The following shows the I²C bus stop condition.

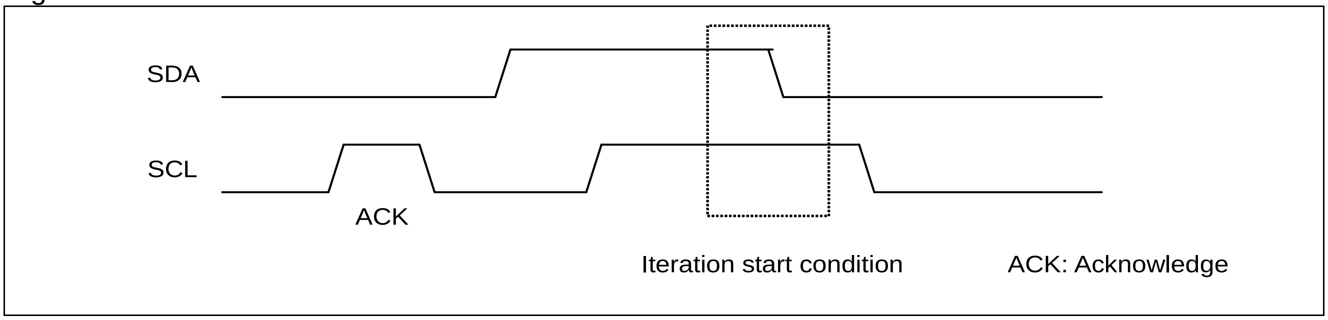
Figure 2-2 Stop condition



■ I²C bus iteration start condition

The following shows the I²C bus iteration start condition.

Figure 2-3 Iteration start condition



2.2. Master mode

Master mode generates the start condition on the I^{2}C bus and outputs clocks to the I^{2}C bus. When the MSS bit in the IBCR register is set to "1" while the I^{2}C bus is in idle state (SCL=HIGH, SDA=HIGH), master mode is activated, causing the ACT bit in the IBCR register to be set to "1".

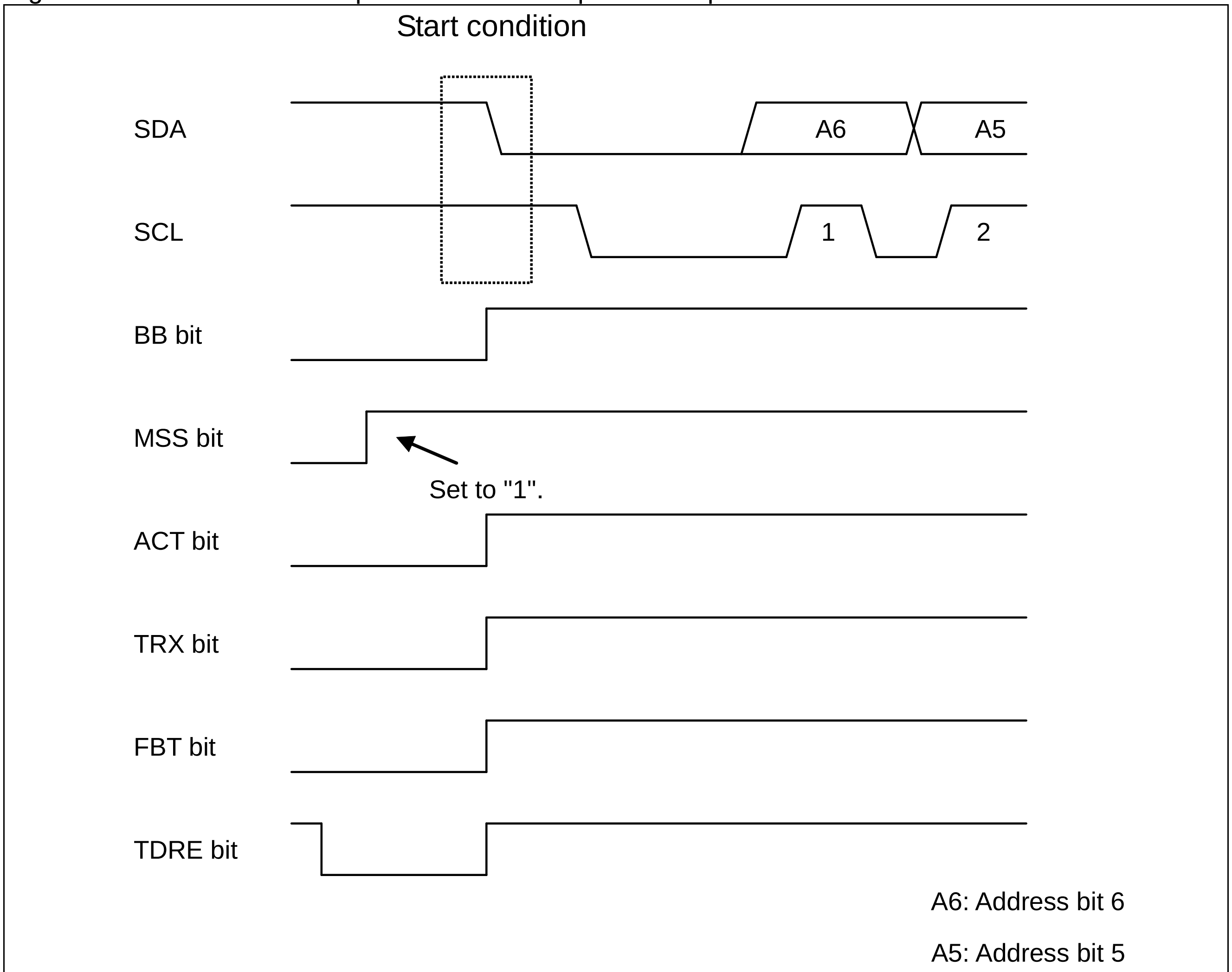
■ Generating start condition

The start condition is generated under the following condition.

- When SDA="H", SCL="H", ISMK:EN=1 and IBSR:BB=0, the IBCR:MSS bit is set to "1".

Outputting the start condition to the I^{2}C bus causes the IBCR:ACT bit to be set to "1". After that, when the start condition is received, the IBSR:BB bit is set to "1" to indicate that the I^{2}C bus is carrying out communications. (See Figure 2-4.)

Figure 2-4 Start condition output and relationships with respective bits



<Note>

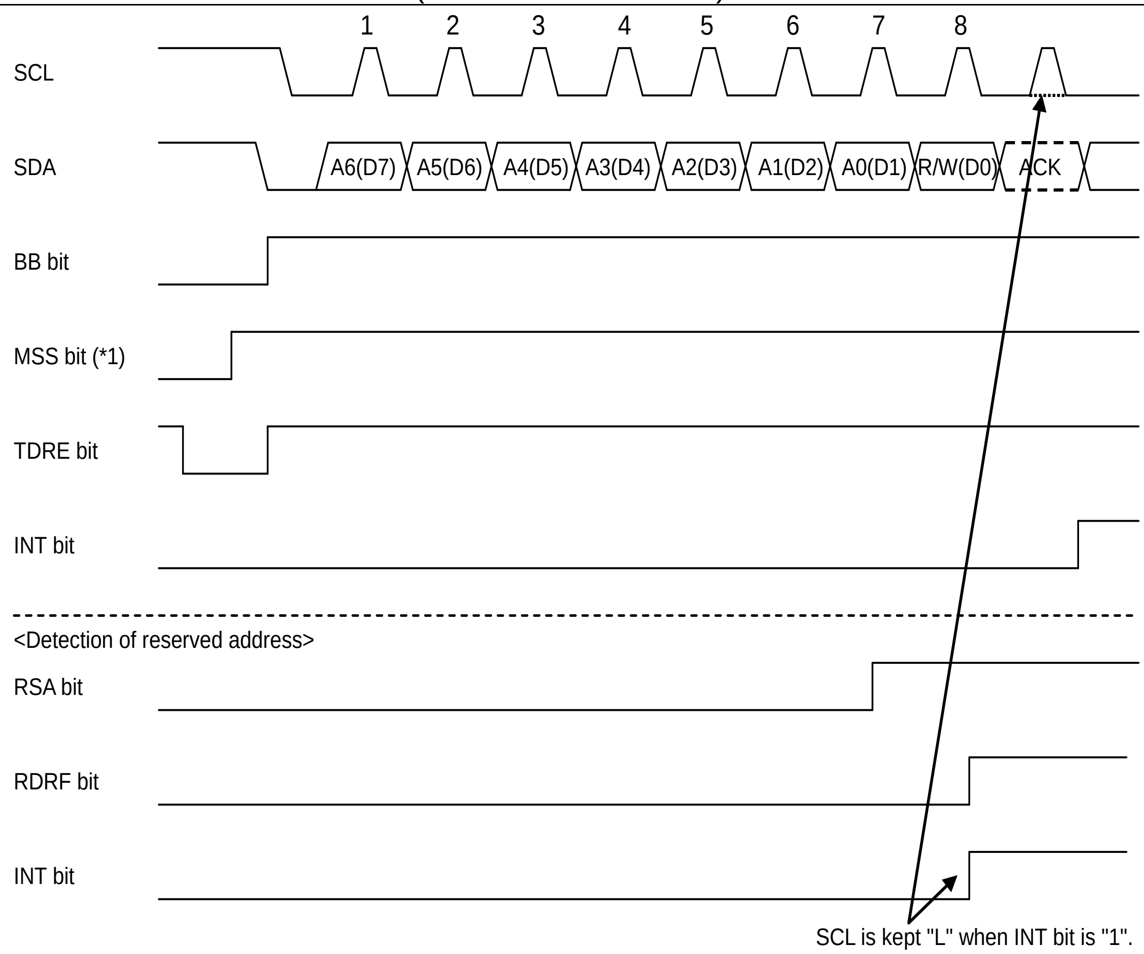
In operation mode 4 (I^2C mode), the bus clock is used at a frequency no lower than 8 MHz. Also note that setting of a baud rate generator that exceeds 400 kbps is prohibited.

■ Slave address output

Outputting the start condition causes data that are set in the TDR register to be output as the address, starting with bit 7. When FIFO is enabled, the data in the TDR register that is written the earliest is output. bit 0 is used as the data direction bit (R/W). When the data direction bit (R/W) is "0", it indicates that data flow in the write direction (from the master to a slave). Set the address to the TDR register before setting the IBCR:MSS bit to "1" or IBCR:SCC bit to "1".

For the output timing of the address and the data direction, see Figure 2-5, Figure 2-6.

Figure 2-5 Address and data direction (when FIFO is disabled)



A6 to A0: Address bits

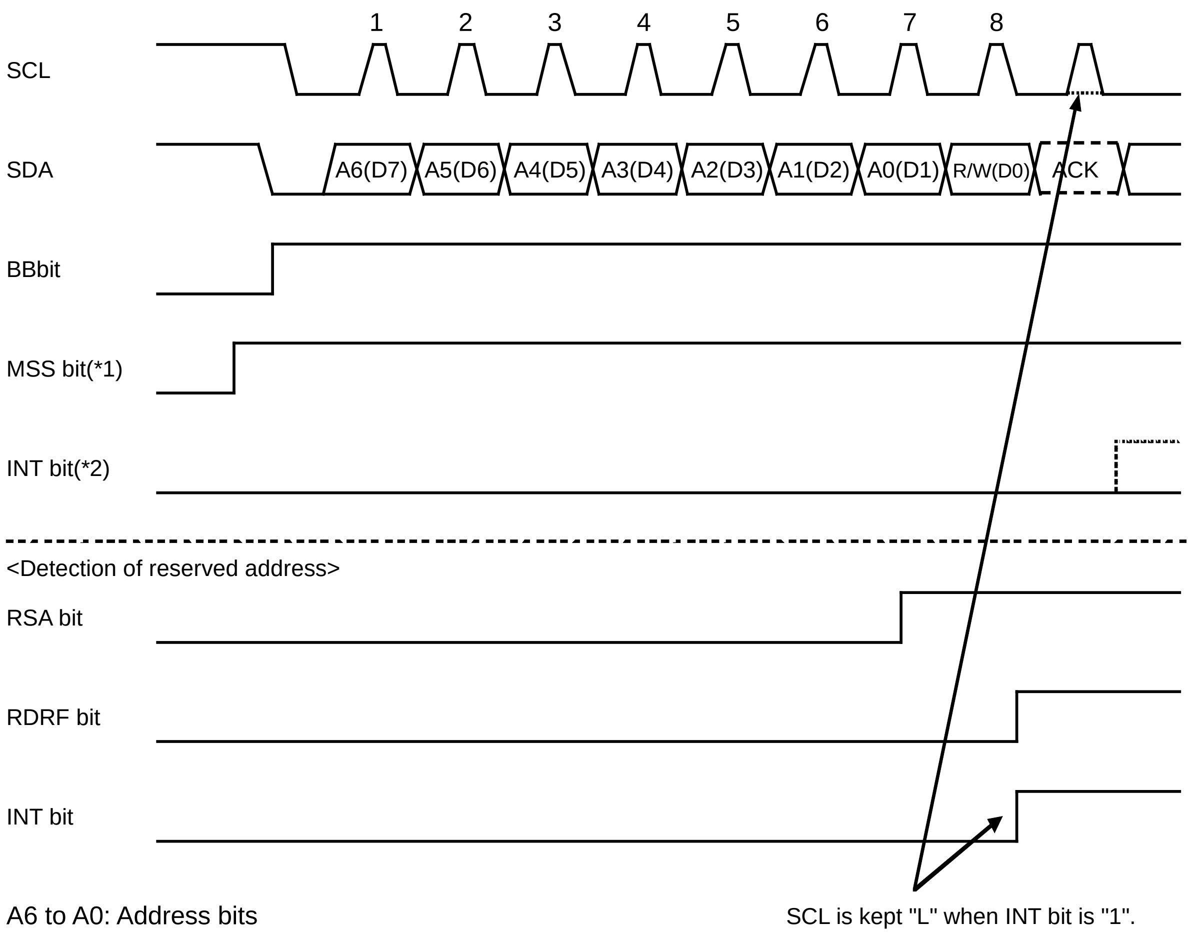
D7 to D0: TDR register bits

R/W: Data direction (write direction if "L")

ACK: Acknowledge (Acknowledged if "L" and output in Slave mode)

*1 : An address must be set in the TDR register before setting the MSS bit to "1".

Figure 2-6 Address and data direction (when transmit/received FIFO is enabled)



A6 to A0: Address bits

D7 to D0: TDR register bits

R/W: Data direction (write direction if "L")

ACK: Acknowledge (Acknowledged if "L" and output in Slave mode)

*1 : An address must be set in the TDR register before setting the MSS bit to “1”.

*2 : If acknowledged with "L" and if R/W="L", the Send FIFO buffer has data. If acknowledged with "L" and if R/W="H", the Receive FIFO buffer has no data, the INT bit is not set to "1".

■ Acknowledgement reception by first byte transmission

When the data direction bit (R/W) is output, the I^{2}C interface receives acknowledgement from a slave. The following lists operations to enable/disable FIFO.

Table 2-2 Operations after acknowledgement reception with DMA mode disabled (IBSR:RSA="0", SSR:DMA="0")

Transmit FIFO	Received FIFO	Transmit FIFO status	Received FIFO status	Data direction bit (R/W)	Operation immediately after receiving acknowledgement: Acknowledgement: ACK	Acknowledgement: NACK
Disable	Disable	-	-	0	If the SSR:TDRE bit is set to "1", the interface sets the IBCR:INT bit to "1" and waits. If the SSR:TDRE bit is set to "0", IBCR:INT bit stays "0" without the wait state.	Sets the IBCR:INT bit to "1" with the wait state.
				1		
Disable	Enable	-	Without data	0	If the SSR:TDRE bit is set to "1", the interface sets the IBCR:INT bit to "1" and waits. If the SSR:TDRE bit is set to "0", IBCR:INT bit stays "0" without the wait state.	Sets the IBCR:INT bit to "1" with the wait state.
			With data		Sets the IBCR:INT bit to "1" with the wait state.	
			-	1	If the SSR:TDRE bit is set to "1", the interface sets the IBCR:INT bit to "1" and waits. If the SSR:TDRE bit is set to "0", IBCR:INT bit stays "0" without the wait state.	
Enable	Disable	-	-	0	If the SSR:TDRE bit is set to "1", the interface sets the IBCR:INT bit to "1" and waits. If the SSR:TDRE bit is set to "0", IBCR:INT bit stays "0" without the wait state.	Sets the IBCR:INT bit to "1" with the wait state.
				1		
Enable	Enable	-	Without data	0	If the SSR:TDRE bit is set to "1", the interface sets the IBCR:INT bit to "1" and waits. If the SSR:TDRE bit is set to "0", IBCR:INT bit stays "0" without the wait state.	Sets the IBCR:INT bit to "1" with the wait state.
			With data		Sets the IBCR:INT bit to "1" with the wait state.	
			-	1	If the SSR:TDRE bit is set to "1", the interface sets the IBCR:INT bit to "1" and waits. If the SSR:TDRE bit is set to "0", IBCR:INT bit stays "0" without the wait state.	

Table 2-3 Operations after acknowledgement reception with DMA mode enabled (IBSR:RSA="0", SSR:DMA="1")

Transmit FIFO	Received FIFO	Transmit FIFO status	Received FIFO status	Data direction bit (R/W)	Operation immediately after receiving acknowledgement: Acknowledgement: ACK	Operation immediately after receiving acknowledgement: Acknowledgement: NACK
Disable	Disable	-	-	0	If the SSR:TDRE bit is set to "1", the interface sets the SSR:TBI bit to "1" and waits. If the SSR:TDRE bit is set to "0", SSR:TBI bit stays "0" without the wait state.	Sets the IBCR:INT bit to "1" with the wait state.
				1		
Disable	Enable	-	Without data	0	If the SSR:TDRE bit is set to "1", the interface sets the SSR:TBI bit to "1" and waits. If the SSR:TDRE bit is set to "0", SSR:TBI bit stays "0" without the wait state.	Sets the IBCR:INT bit to "1" with the wait state.
			With data		Sets the IBCR:INT bit to "1" with the wait state.	
			-	1	If the SSR:TDRE bit is set to "1", the interface sets the SSR:TBI bit to "1" and waits. If the SSR:TDRE bit is set to "0", SSR:TBI bit stays "0" without the wait state.	
Enable	Disable	-	-	0	If the SSR:TDRE bit is set to "1", the interface sets the SSR:TBI bit to "1" and waits. If the SSR:TDRE bit is set to "0", SSR:TBI bit stays "0" without the wait state.	Sets the IBCR:INT bit to "1" with the wait state.
				1		
Enable	Enable	-	Without data	0	If the SSR:TDRE bit is set to "1", the interface sets the SSR:TBI bit to "1" and waits. If the SSR:TDRE bit is set to "0", SSR:TBI bit stays "0" without the wait state.	Sets the IBCR:INT bit to "1" with the wait state.
			With data		Sets the IBCR:INT bit to "1" with the wait state.	
			-	1	If the SSR:TDRE bit is set to "1", the interface sets the SSR:TBI bit to "1" and waits. If the SSR:TDRE bit is set to "0", SSR:TBI bit stays "0" without the wait state.	

● When DMA mode is disabled (SSR:DMA=0)

To disable FIFO (To disable both transmit FIFO and received FIFO)

- When the IBSR:RSA bit is set to "0", after receiving acknowledgement, the interface sets the interrupt flag (IBCR:INT) to "1" if the SSR:TDRE bit is set to "1" and waits while maintaining SCL at LOW. Writing "0" to the interrupt flag sets the interrupt flag to "0", which releases wait. If the SSR:TDRE bit is set to "0", the interface generates a clock on SCL upon reception of ACK without setting the interrupt flag to "1".
- When the IBSR:RSA bit is set to "1", after receiving a reserved address (before acknowledgement), the interface sets the interrupt flag (IBCR:INT) to "1" and waits while maintaining SCL at LOW. After reading from the RDR register, setting the IBCR:ACKE bit and transmit data and writing "0" to the interrupt flag causes the interrupt flag to be set to "0", which releases wait.
- The received acknowledgement is set to the IBSR:RACK bit. The interface checks the IBSR:RACK bit during wait, and, in case of NACK, it writes "0" to the IBCR:MSS bit or "1" to the IBCR:SCC bit to generate a stop condition or iteration start condition. At this time, the IBCR:INT bit is cleared to "0" automatically.

To enable FIFO

- Before setting "1" to the IBCR:MSS bit, it is needed to set the following for FIFO.
  - When transmitting to a slave (the data direction bit=0), data including the slave address must be set to transmit FIFO.
  - When receiving data from a slave (the data direction bit=1), the FIFO Byte Register must be set with the number of data sets to be received, and dummy data must be written to the Transmit Data Register for the slave address, data direction bit and the data volume for the number of bytes to be received.
- When the IBSR:RSA bit is set to "0", after receiving acknowledgement and if it is ACK, the interface transmits/receives data according to the data direction bit without setting the interrupt flag (IBCR:INT) to "1" (with no wait occurring). If it is NACK, the interface sets the interrupt flag (IBCR:INT) to "1", and waits while maintaining SCL at LOW.
- The received acknowledgement is stored in the IBSR:RACK bit. The interface checks the IBSR:RACK bit during wait, and, in case of NACK, it writes "0" to the IBCR:MSS bit or "1" to the IBCR:SCC bit to generate a stop condition or iteration start condition. At this time, the IBCR:INT bit is cleared to "0" automatically.

● When DMA mode is enabled (SSR:DMA=1)

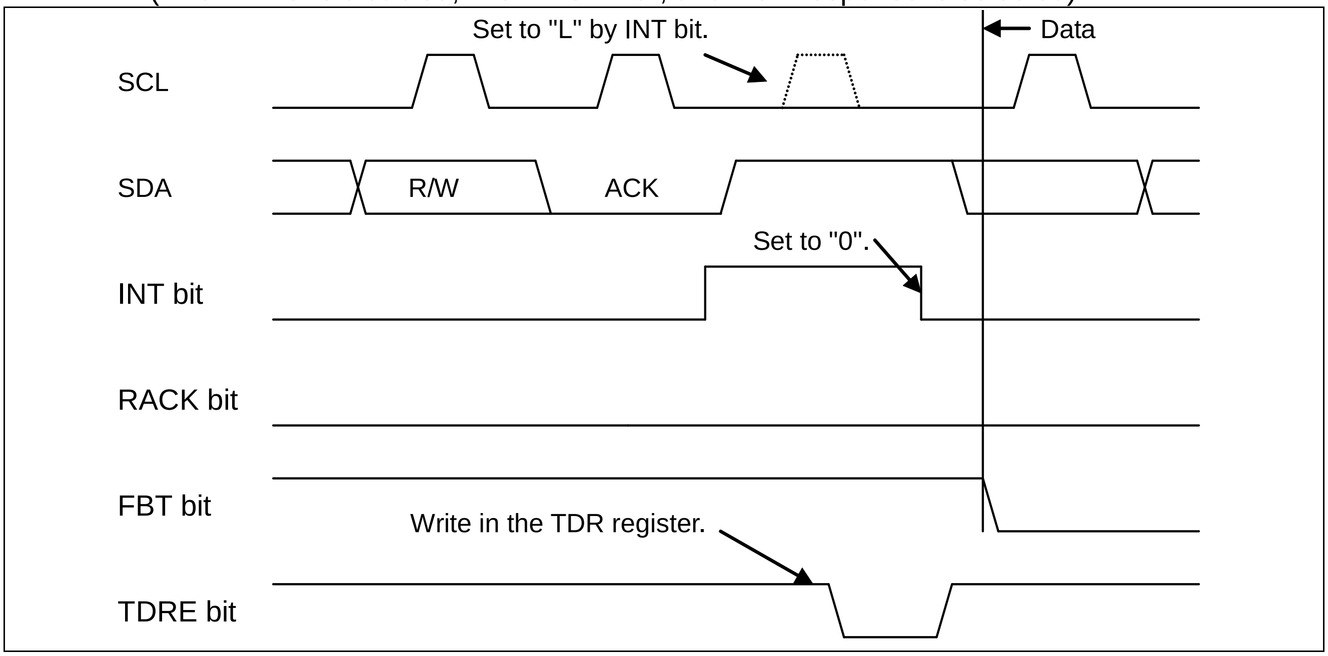
To disable FIFO (To disable both transmit FIFO and received FIFO)

- When the IBSR:RSA bit is set to "0", after receiving acknowledgement, the interface sets the transmit bus idle flag (SSR:TBI) to "1" if the SSR:TDRE bit is set to "1" and waits while maintaining SCL at LOW. Writing data to be transmitted to the TDR register causes the transmit bus idle flag to be set to "0", which releases wait. If the SSR:TDRE bit is set to "0", the interface generates a clock on SCL upon reception of ACK without setting the transmit bus idle flag (SSR:TBI) to "1".
- When the IBSR:RSA bit is set to "1", after receiving a reserved address (before acknowledgement), the interface sets the interrupt flag (IBCR:INT) to "1" and waits while maintaining SCL at LOW. After reading from the RDR register, setting the IBCR:ACKE bit and transmit data and writing "0" to the interrupt flag causes the interrupt flag to be set to "0", which releases wait.
- The received acknowledgement is set to the IBSR:RACK bit. The interface checks the IBSR:RACK bit during wait, and, in case of NACK, it writes "0" to the IBCR:MSS bit or "1" to the IBCR:SCC bit to generate a stop condition or iteration start condition. At this time, the IBCR:INT bit is cleared to "0" automatically.

To enable FIFO

- Before setting "1" to the IBCR:MSS bit, it is needed to set the following for FIFO.
  - When transmitting to a slave (the data direction bit=0), data including the slave address must be set to transmit FIFO.
  - When receiving data from a slave (the data direction bit=1), the FIFO Byte Register must be set with the number of data sets to be received, and dummy data must be written to the Transmit Data Register for the slave address, data direction bit and the data volume for the number of bytes to be received.
- When the IBSR:RSA bit is set to "0", after receiving acknowledgement and if it is ACK, the interface transmits/receives data according to the data direction bit without setting the interrupt flag (IBCR:INT) to "1" (with no wait occurring). If it is NACK, the interface sets the interrupt flag (IBCR:INT) to "1", and waits while maintaining SCL at LOW.
- The received acknowledgement is stored in the IBSR:RACK bit. The interface checks the IBSR:RACK bit during wait, and, in case of NACK, it writes "0" to the IBCR:MSS bit or "1" to the IBCR:SCC bit to generate a stop condition or iteration start condition. At this time, the IBCR:INT bit is cleared to "0" automatically.

Figure 2-7 Acknowledgement
(when FIFO is disabled, IBSR:RSA="0", and ACK response is selected)



The following describes the wait timing for an address.
- After receiving acknowledgment if the IBSR:RSA bit is "0".
- Before receiving acknowledgment if the IBSR:RSA bit is "1".
Not dependent on the setting of the IBCR:WSEL.

Figure 2-8 Acknowledgement (when FIFO is disabled, IBSR:RSA="0", and NACK response is selected)

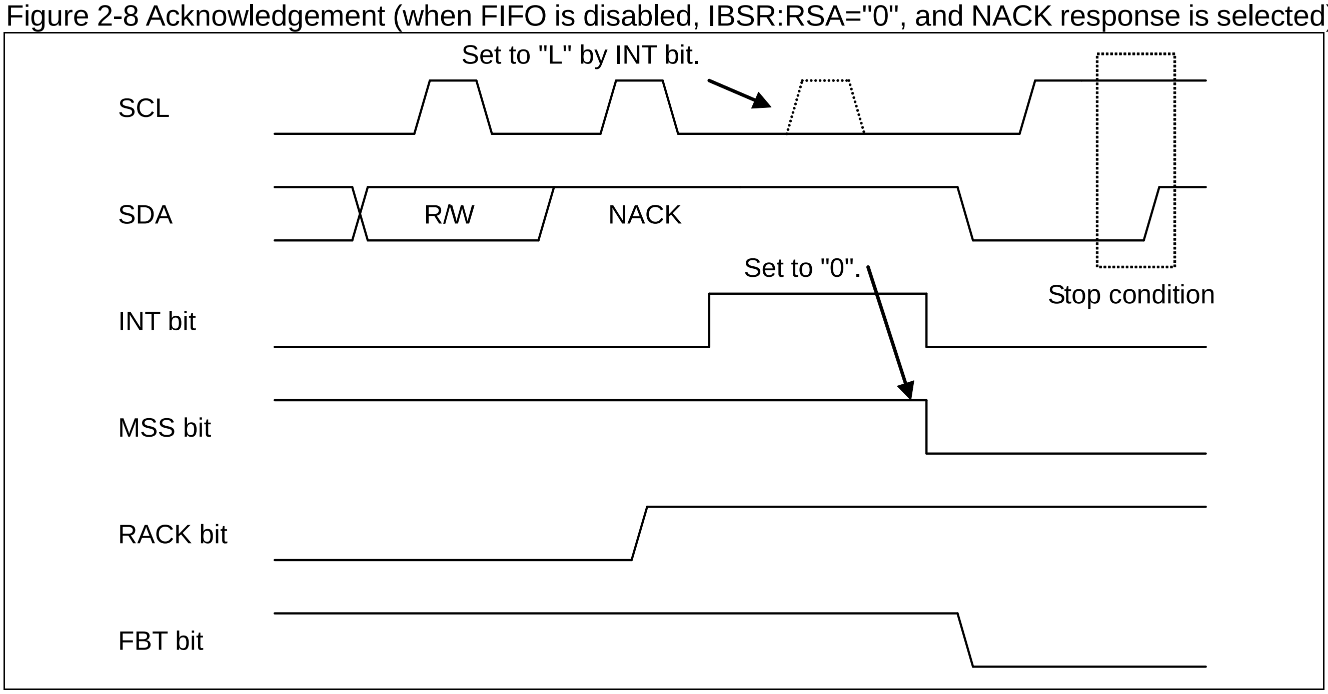


Figure 2-9 Acknowledgement (when FIFO is disabled, IBSR:RSA="1", and ACK response is selected)

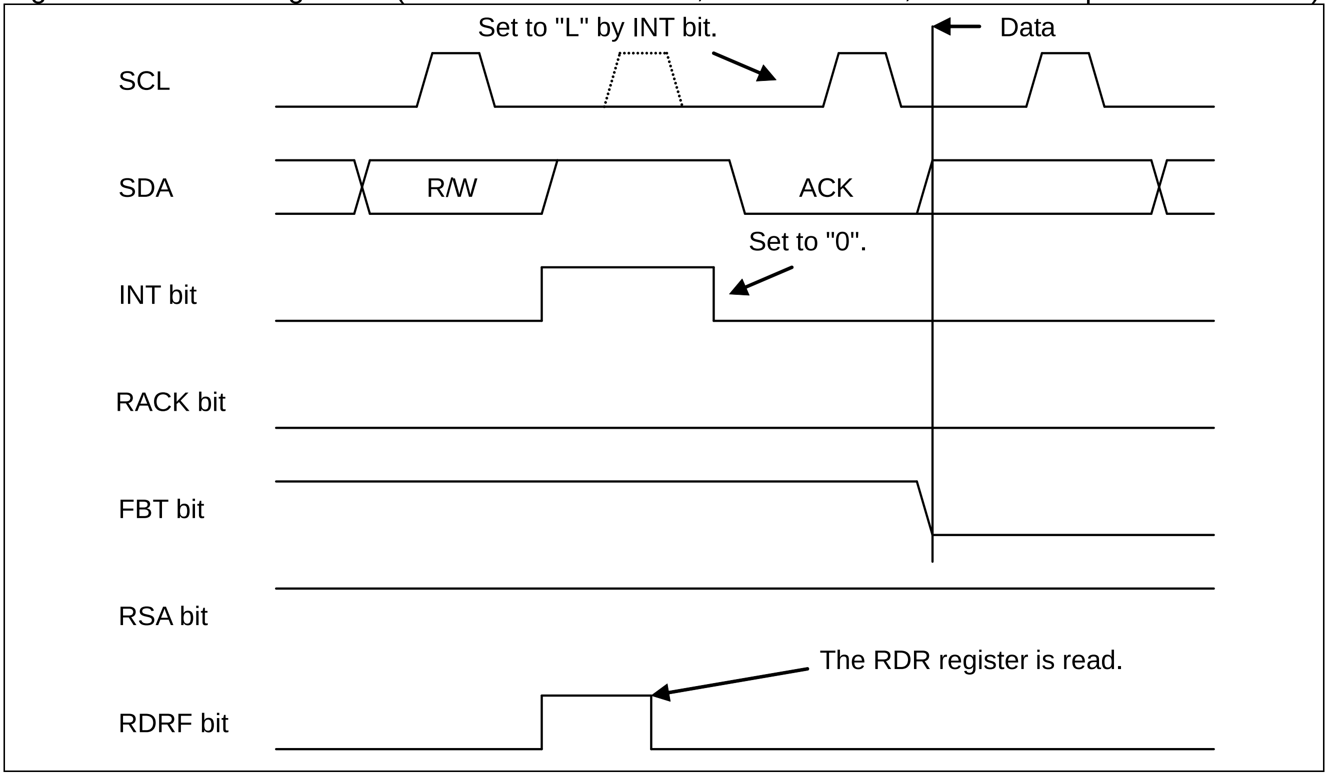


Figure 2-10 Acknowledgement (when FIFO is disabled, IBSR:RSA="1", and NACK response is selected)

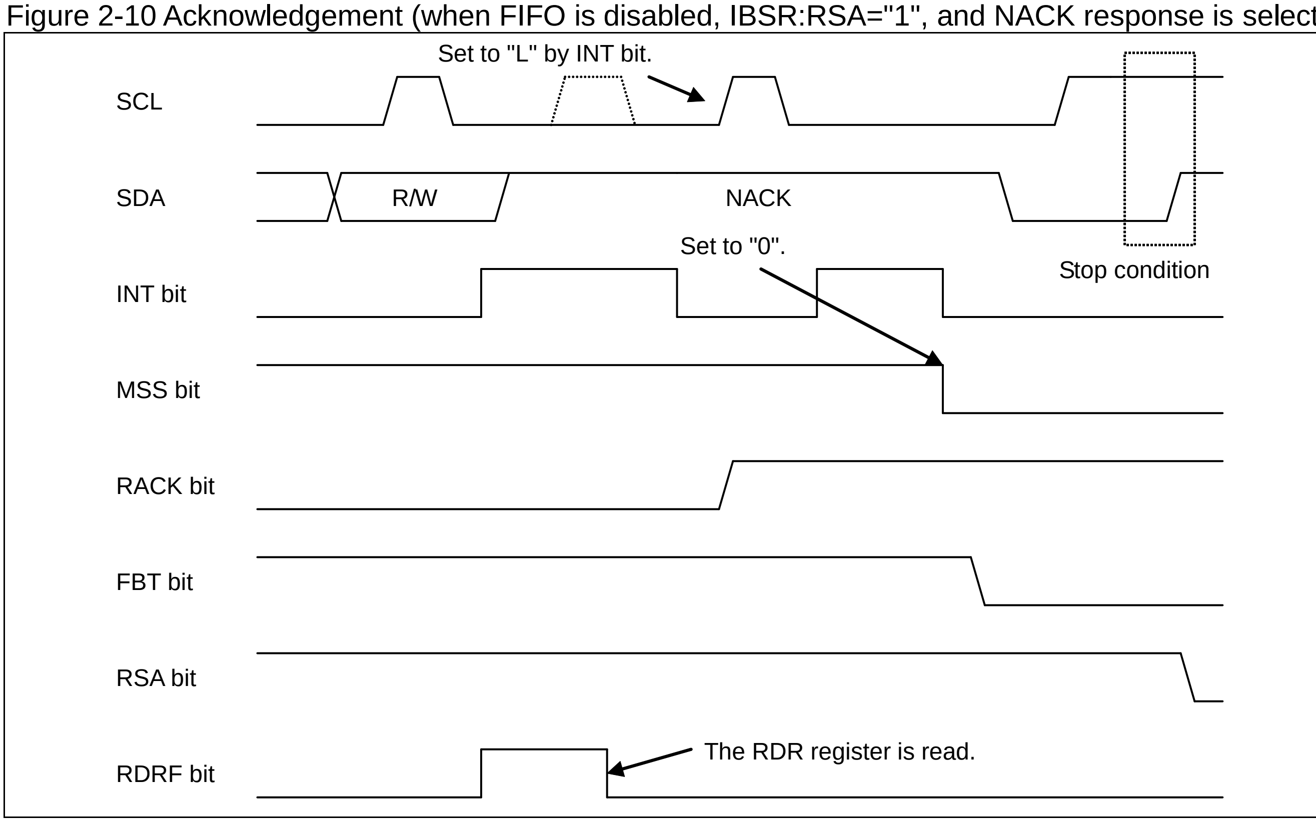
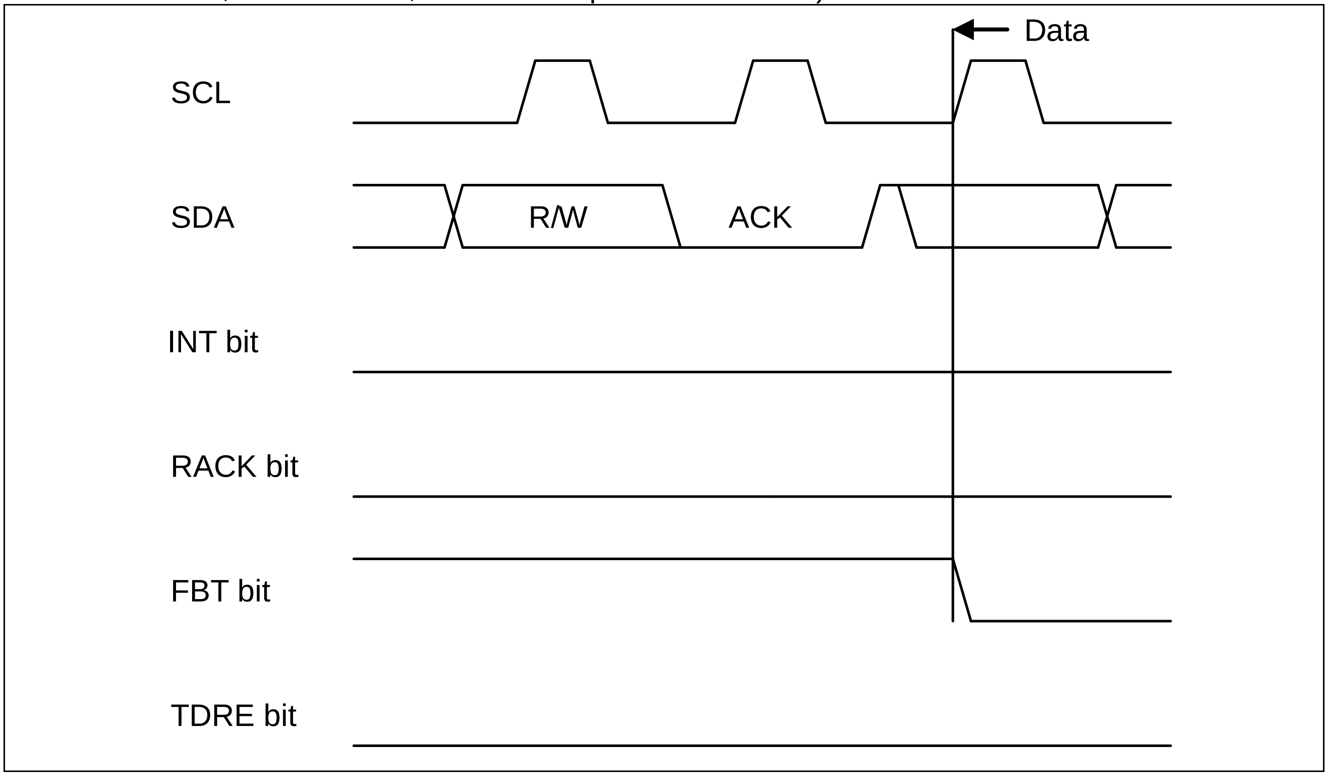


Figure 2-11 Acknowledgement (when FIFO is enabled, transmit FIFO has data, received FIFO has no data, IBSR:RSA=0, and ACK response is selected)



■ Data transmission by the master

When the data direction bit (R/W) is set to "0", data are transmitted from the master. The slave gives response either with ACK or NACK for each one-byte transmission.

The following shows the wait timing by IBCR:WSEL setting.

Table 2-4 IBCR:WSEL bit status for master data transmission when DMA mode is disabled (SSR:DMA=0)

WSEL bit	Operation
0	<When FIFO is not used> After the second byte, after acknowledgement with "1" set for the SSR:TDRE bit or upon detection of arbitration lost, the interrupt flag (IBCR:INT) is set to "1" and SCL to LOW for the wait state. <When FIFO is used> Starts the wait state by setting the interrupt flag (IBCR:INT) to "1" after acknowledgement upon detection of arbitration lost or when no more valid data remain in the Transmit Data Register (SSR:TDRE=1).
1	<When FIFO is not used> After the second byte, after the master has transmitted one-byte data with "1" set for the SSR:TDRE bit or upon detection of arbitration lost, the interrupt flag (IBCR:INT) is set to "1" and SCL to LOW for the wait state. <When FIFO is used> Starts the wait state by setting the interrupt flag (IBCR:INT) to "1" when data transmission has taken place after detection of arbitration lost or no more valid data in the Transmit Data Register (SSR:TDRE=1).

Table 2-5 IBCR:WSEL bit status for master data transmission when DMA mode is enabled (SSR:DMA=1)

WSEL bit	Operation
0	<When FIFO is not used> After the second byte, after acknowledgement with "1" set for the SSR:TDRE bit, the transmit bus idle flag (SSR:TBI) is set to "1" and SCL to LOW for the wait state. <When FIFO is used> Starts the wait state by setting the transmit bus idle flag (SSR:TBI) to "1" after acknowledgment when no more valid data remain in the Transmit Data Register (SSR:TDRE=1).
1	<When FIFO is not used> After the second byte, after the master has transmitted one-byte data with "1" set for the SSR:TDRE bit, the transmit bus idle flag (SSR:TBI) is set to "1" and SCL to LOW for the wait state. <When FIFO is used> Starts the wait state by setting the transmit bus idle flag (SSR:TBI) to "1" after the master has transmitted one-byte data when no more valid data remain in the Transmit Data Register (SSR:TDRE=1).

In the following case, however, the interrupt flag (IBCR:INT) is set after acknowledgement, regardless of the IBCR:WSEL setting:

· If NACK is received when the stop condition (IBCR:MSS=0, ACT=1) is not set.

The following shows an example procedure for transmitting data to a slave.

● Data Transmission to slave when DMA mode is disabled (SSR:DMA=0)

1. To transmit data to an address other than the reserved:

· When transmit FIFO is disabled:

1. Sets Slave Address (including the data direction bit) to the TDR register and writes "1" to the IBCR:MSS bit.
2. ACK is received after the Slave Address setting is transmitted, and then the interrupt flag (IBCR:INT) is set to "1".
3. Writes transmit data to the TDR register.
4. Writes "0" to the interrupt flag (IBCR:INT) upon updating of the IBCR:WSEL bit and releases the wait state of the I^2C bus.
5. After transmitting one byte, the interrupt flag is set to "1", which puts the I^2C bus in the wait state after receiving acknowledgment in case of IBCR:WSEL=0, and directly after transmitting one byte in case IBCR:WSEL=1. Repeats steps 3 to 5 until all the specified number of data sets have been transmitted. However, if NACK is received after the wait state is released when IBCR:WSEL=1, another interrupt is generated after receiving acknowledgement and the bus enters the wait state.
6. Sets the IBCR:MSS bit to "0" or sets the IBCR:SCC bit to "1" to generate the stop condition or iteration start condition.

· When transmit FIFO is enabled:

1. Writes Slave Address (including the data direction bit) and transmit data to the TDR register.
2. Writes "1" to the IBCR:MSS bit upon setting of the IBCR:WSEL bit.
3. If NACK is received during transmission, sets the interrupt flag (IBCR:INT) to "1" immediately after that to put the I^2C bus in the wait state. If ACK responses are received for all bytes, sets the interrupt flag to "1" according to the setting of IBCR:WSEL after the last byte is transmitted to put the I^2C bus in the wait state.
4. Sets the IBCR:MSS bit to "0" or sets the IBCR:SCC bit to "1" to generate the stop condition or iteration start condition.

2. To transmit data to a reserved address:

- When transmit FIFO is disabled:
  1. Sets the reserved address for Slave Address in the TDR register and writes "1" to the IBCR:MSS bit.
  2. After the Slave Address setting is transmitted, the interrupt flag (IBCR:INT) is set to "1".
  3. Reads from the RDR register and confirms the reserved address.(*1)
  4. Writes transmit data to the TDR register.
  5. Writes "0" to the interrupt flag (IBCR:INT) upon updating of the IBCR:WSEL bit and releases the wait state of the I^2C bus.
  6. After transmitting one byte, the interrupt flag is set to "1", which puts the I^2C bus in the wait state after receiving acknowledgment in case of IBCR:WSEL=0, and directly after transmitting one byte in case IBCR:WSEL=1. Repeats steps 4 to 6 until all the specified number of data sets have been transmitted. However, if NACK is received after the wait state is released when IBCR:WSEL=1, another interrupt is generated after receiving acknowledgement and the bus enters the wait state.
  7. Sets the IBCR:MSS bit to "0" or sets the IBCR:SCC bit to "1" to generate the stop condition or iteration start condition.

- When transmit FIFO is enabled:
  1. Sets the reserved address for Slave Address in the TDR register and writes "1" to the IBCR:MSS bit.
  2. After the Slave Address setting is transmitted, the interrupt flag (IBCR:INT) is set to "1".
  3. Reads from the RDR register and confirms the reserved address.(*1)
  4. Writes all transmit data to the TDR register (until transmit FIFO becomes full if it is the case).
  5. If NACK is received during transmission, the interrupt flag (IBCR:INT) is set to "1" immediately after that to put the I^2C bus in the wait state.
     If ACK responses are received for all bytes, sets the interrupt flag to "1" according to the setting of IBCR:WSEL after the last byte is transmitted to put the I^2C bus in the wait state.
  6. Sets the IBCR:MSS bit to "0" or sets the IBCR:SCC bit to "1" to generate the stop condition or iteration start condition.

*1 : When any one of the following conditions is met, the IBCR:ACKE and IBCR:WSEL bits must be set to "1" and to check which is needed for the next data, operation as a master or operation as a slave.

- Multi-master mode is activated and the reserved address is a general call.
- Arbitration lost has been detected and the interface may operate as a slave.

● Data Transmission to slave when DMA mode is enabled (SSR:DMA=1)

1. To transmit data to an address other than the reserved:

- When transmit FIFO is disabled:
  1. Sets Slave Address (including the data direction bit) to the TDR register and writes "1" to the IBCR:MSS bit.
  2. ACK is received after the Slave Address setting is transmitted, and then the transmit bus idle flag (SSR:TBI) is set to "1".
  3. Writes data to be transmitted to the TDR register to release the wait state of the I^2C bus.
  4. After transmitting one byte, sets the transmit bus idle flag (SSR:TBI) to "1" to put the I^2C bus in the wait state after receiving acknowledgment in case of IBCR:WSEL=0, and directly after transmitting one byte in case of IBCR:WSEL=1.
  5. Writes data to be transmitted to the TDR register to release the wait state of the I^2C bus.
  6. After transmitting one byte, sets the transmit bus idle flag to "1" to put the I^2C bus in the wait state after receiving acknowledgment in case of IBCR:WSEL=0, and directly after transmitting one byte in case of IBCR:WSEL=1. Repeats steps 5 to 6 until all the specified number of data sets have been transmitted. However, if NACK is received after the wait state is released when IBCR:WSEL=1, the interrupt flag (IBCR:INT) is set to "1" after receiving acknowledgement and the bus enters the wait state.
  7. Sets the IBCR:MSS bit to "0" or sets the IBCR:SCC bit to "1"[*2] to generate the stop condition or iteration start condition.

- When transmit FIFO is enabled:
  1. Writes Slave Address (including the data direction bit) and transmit data to the TDR register.
  2. Writes "1" to the IBCR:MSS bit upon setting of the IBCR:WSEL bit.
  3. If NACK is received during transmission, sets the interrupt flag (IBCR:INT) to "1" immediately after that to put the I^2C bus in the wait state. If ACK responses are received for all bytes, sets the transmit bus idle flag (SSR:TBI) to "1" according to the setting of IBCR:WSEL after the last byte is transmitted to put the I^2C bus in the wait state.
  4. Sets the IBCR:MSS bit to "0" or sets the IBCR:SCC bit to "1"[*2] to generate the stop condition or iteration start condition.

2. To transmit data to a reserved address:

- When transmit FIFO is disabled:
  1. Sets the reserved address for Slave Address in the TDR register and writes "1" to the IBCR:MSS bit.
  2. After the Slave Address setting is transmitted, the interrupt flag (IBCR:INT) is set to "1".
  3. Reads from the RDR register and confirms the reserved address.(*1)
  4. Writes transmit data to the TDR register.
  5. Writes "0" to the interrupt flag (IBCR:INT) upon updating of the IBCR:WSEL bit and releases the wait state of the I^2C bus.
  6. After transmitting one byte, the interrupt flag is set to "1", which puts the I^2C bus in the wait state after receiving acknowledgment in case of IBCR:WSEL=0, and directly after transmitting one byte in case IBCR:WSEL=1.
  7. Writes data to be transmitted to the TDR register to release the wait state of the I^2C bus.
  8. After transmitting one byte, sets the transmit bus idle flag to "1" to put the I^2C bus in the wait state after receiving acknowledgment in case of IBCR:WSEL=0, and directly after transmitting one byte in case of IBCR:WSEL=1. Repeats steps 7 to 8 until all the specified number of data sets have been transmitted. However, if NACK is received after the wait state is released when IBCR:WSEL=1, the interrupt flag (IBCR:INT) is set to "1" after receiving acknowledgement and the bus enters the wait state.
  9. Sets the IBCR:MSS bit to "0" or sets the IBCR:SCC bit to "1"[*2] to generate the stop condition or iteration start condition.

- When transmit FIFO is enabled:

1. Sets the reserved address for Slave Address in the TDR register and writes "1" to the IBCR:MSS bit.
2. After the Slave Address setting is transmitted, the interrupt flag (IBCR:INT) is set to "1".
3. Reads from the RDR register and confirms the reserved address.(*1)
4. Writes all transmit data to the TDR register (until transmit FIFO becomes full if it is the case).
5. If NACK is received during transmission, sets the interrupt flag (IBCR:INT) to "1" immediately after that to put the I^2C bus in the wait state. If ACK responses are received for all bytes, sets the interrupt flag (IBCR:INT) to "1" according to the setting of IBCR:WSEL after the last byte is transmitted, which puts the I^2C bus in the wait state.
6. Sets the IBCR:MSS bit to "0" or sets the IBCR:SCC bit to "1"[*2] to generate the stop condition or iteration start condition.

*1 : When any one of the following conditions is met, the IBCR:ACKE and IBCR:WSEL bits must be set to "1" and to check which is needed for the next data, operation as a master or operation as a slave.

- Multi-master mode is activated and the reserved address is a general call.
- Arbitration lost has been detected and the interface may operate as a slave.

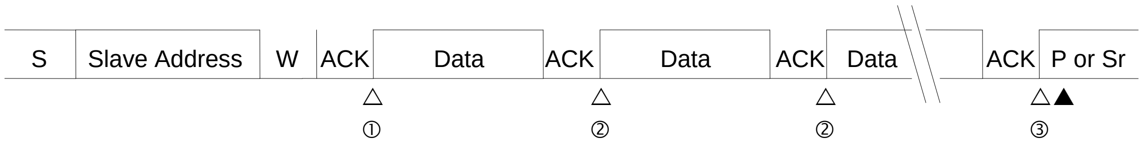
*2 : When DMA is enabled (SSR:DMA=1), the SSR:TBI bit is "1" and the IBCR:INT bit is "0", follow the steps below to issue the iteration start condition.

1. Set the IBCR:INT bit to "1".
2. Check that the IBCR:INT bit is set to "1".
3. Write the slave address in the TDR.
4. Set the IBCR:SCC bit to "1".

<Notes>

- When seven-bit slave address detection is enabled (ISBA:SAEN=1), it is prohibited to specify a seven-bit slave address in master mode.
- To change the IBCR register during transmission/reception, do so when the interrupt flag (IBCR:INT) is "1".
- If the IBCR:WSEL bit is changed, the update is used as a condition for generating the transmit bus idle flag (SSR:TBI) when the interrupt flag (IBCR:INT) is enabled and DMA mode is also enabled (SSR:DMA=1) for the next data.
- The master operates as follows when transmit data are written to the TDR register during data transmission with SSR:TDRE set to "1" and an ACK response is detected.
  - When DMA mode is disabled (SSR:DMA=0), the interrupt flag (IBCR:INT) does not attain "1", and the written data are transmitted.
  - When DMA mode is enabled (SSR:DMA=1), the transmit bus idle flag (SSR:TBI) does not attain "1", and the written data are transmitted.
- The master operates as follows when transmit data are written to the TDR register during data reception with SSR:TDRE set to "1" and an ACK response is detected.
  - When DMA mode is disabled (SSR:DMA=0), the interrupt flag (IBCR:INT) does not attain "1" and only SSR:RDRF attains "1" (when received FIFO is enabled, and the number of bytes set in the FBYTE register have been received).
  - When DMA mode is enabled (SSR:DMA=1), the transmit bus idle flag (SSR:TBI) does not attain "1" and only SSR:RDRF attains "1" (when received FIFO is enabled, and the number of bytes set in the FBYTE register have been received).

Figure 2-12 Master mode interrupt 1 by disabling FIFO (SSR:DMA="0", IBCR:WSEL="0", IBSR:RSA="0")



S: Start condition

W: Data direction bit (write direction)

P: Stop condition

Sr: Iteration start condition

△: Interrupt by INTE="1"

▲: Interrupt by CNDE="1"

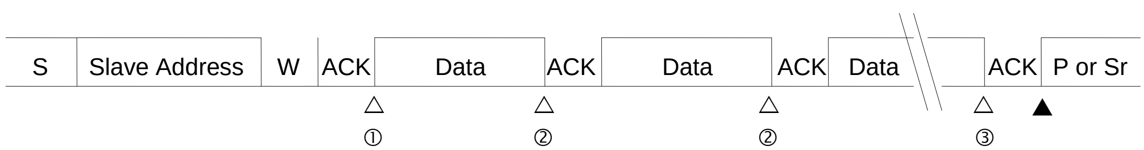
① An interrupt occurs when the slave address is sent, the direction bit is sent, and an ACK is received.
- The send data is written in the TDR register, and the INT bit is set to "0".

② An interrupt occurs when a single byte is sent and an ACK is received.
- The send data is written in the TDR register, and the INT bit is set to "0".

③ An interrupt occurs when a single byte is sent and an ACK is received.
- MSS bit is set to "0", or MSS and SCC bits are set to "1".

*) If an interrupt flag (INT) is set, the TDRE bit is set to "1".

Figure 2-13 Master mode transmit interrupt 2 by disabling FIFO (SSR:DMA="0", IBCR:WSEL="1", IBSR:RSA="0", ACK response)



S: Start condition

W: Data direction bit (write direction)

P: Stop condition

Sr: Iteration start condition

△: Interrupt by INTE="1"

▲: Interrupt by CNDE="1"

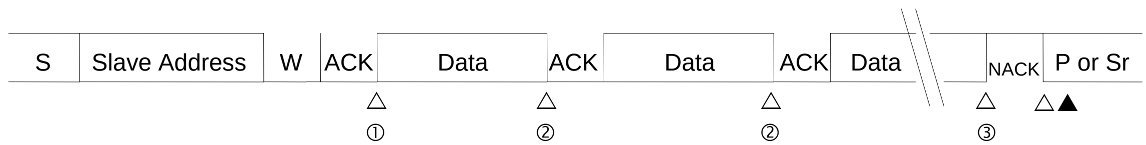
① An interrupt occurs when the slave address is sent, the direction bit is sent, and an ACK is received.
- The send data is written in the TDR register, and the INT bit is set to "0".

② An interrupt occurs when a single byte is sent.
- The send data is written in the TDR register, and the INT bit is set to "0".

③ An interrupt occurs when a single byte is sent.
- MSS bit is set to "0", or MSS and SCC bits are set to "1".

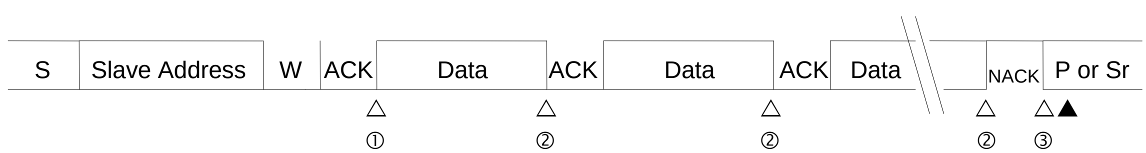
*) If an interrupt flag (INT) is set, the TDRE bit is set to "1".

Figure 2-14 Master mode transmit interrupt 3 by disabling FIFO (SSR:DMA="0", IBCR:WSEL="1", IBSR:RSA="0", NACK response)



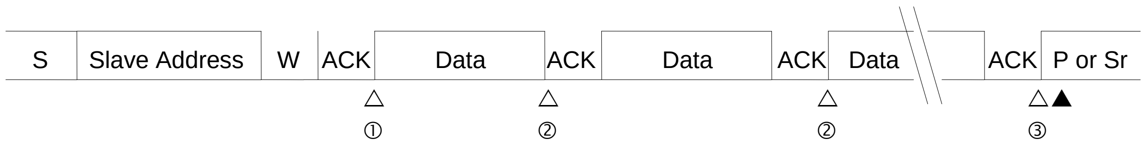
S: Start condition
W: Data direction bit (write direction)
P: Stop condition
Sr: Iteration start condition
△: Interrupt by INTE="1"
▲: Interrupt by CNDE="1"
① An interrupt occurs when the slave address is sent, the direction bit is sent, and an ACK is received.
- The send data is written in the TDR register, and the INT bit is set to "0".
② An interrupt occurs when a single byte is sent.
- The send data is written in the TDR register, and the INT bit is set to "0".
③ An interrupt occurs when a single byte is sent.
- MSS bit is set to "0", or MSS and SCC bits are set to "1".
*) If an interrupt flag (INT) is set, the TDRE bit is set to "1".

Figure 2-15 Master mode transmit interrupt 4 by disabling FIFO (SSR:DMA="0", IBCR:WSEL="1", IBSR:RSA="0", NACK response during transmission)



S: Start condition
W: Data direction bit (write direction)
P: Stop condition
Sr: Iteration start condition
△: Interrupt by INTE="1"
▲: Interrupt by CNDE="1"
① An interrupt occurs when the slave address is sent, the direction bit is sent, and an ACK is received.
- The send data is written in the TDR register, and the INT bit is set to "0".
② An interrupt occurs when a single byte is sent.
- The send data is written in the TDR register, and the INT bit is set to "0".
③ An interrupt occurs when a NACK is responded.
- MSS bit is set to "0", or MSS and SCC bits are set to "1".
*) If an interrupt flag (INT) is set, the TDRE bit is set to "1".

Figure 2-16 Master mode transmit interrupt 5 by disabling FIFO (SSR:DMA="0", IBCR:WSEL="1" -> "0", IBSR:RSA="0", ACK response)



S: Start condition

W: Data direction bit (write direction)

P: Stop condition

Sr: Iteration start condition

△: Interrupt by INTE="1"

▲: Interrupt by CNDE="1"

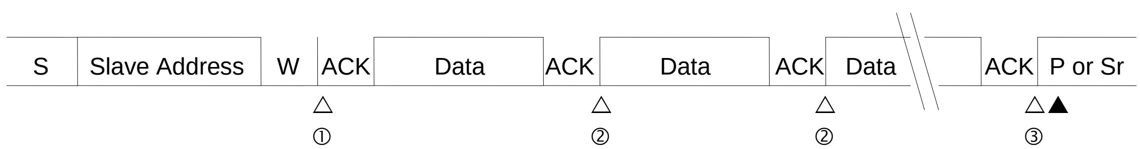
① An interrupt occurs when the slave address is sent, the direction bit is sent, and an ACK is received.
- The send data is written in the send buffer, and the INT bit is set to "0".

② An interrupt occurs when a single byte is sent.
- The send data is written in the send buffer, and both WSEL and INT bits are set to "0".

③ An interrupt occurs when a single byte is sent.
- MSS bit is set to "0", or MSS and SCC bits are set to "1".

*) If an interrupt flag (INT) is set, the TDRE bit is set to "1".

Figure 2-17 Master mode interrupt 6 by disabling FIFO (SSR:DMA="0", IBCR:WSEL="0", IBSR:RSA="1")



S: Start condition

W: Data direction bit (write direction)

P: Stop condition

Sr: Iteration start condition

△: Interrupt by INTE="1"

▲: Interrupt by CNDE="1"

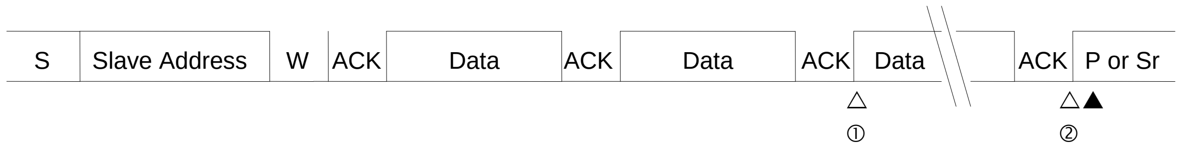
① An interrupt occurs when the slave address (reserved address) is sent, a direction bit is sent, and an ACK is received.
- The send data is written in the TDR register, and the INT bit is set to "0".

② An interrupt occurs when a single byte is sent and an ACK is received.
- The send data is written in the TDR register, and the INT bit is set to "0".

③ An interrupt occurs when a single byte is sent and an ACK is received.
- MSS bit is set to "0", or MSS and SCC bits are set to "1".

*) If an interrupt flag (INT) is set, the TDRE bit is set to "1".

Figure 2-18 Master mode transmit interrupt 7 by enabling FIFO (SSR:DMA="0", IBCR:WSEL="0", IBSR:RSA="0", ACK response)



S: Start condition

W: Data direction bit (write direction)

P: Stop condition

Sr: Iteration start condition

△: Interrupt by INTE="1"

▲: Interrupt by CNDE="1"

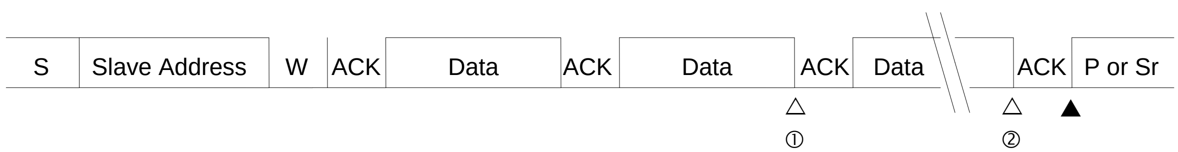
① An interrupt occurs if the Send FIFO buffer is emptied.

- The send data is written in the Send FIFO buffer, and INT bit is set to "0".

② An interrupt occurs when the last byte is sent (the Send FIFO buffer is emptied) and an ACK is received.

- MSS bit is set to "0", or MSS and SCC bits are set to "1".

Figure 2-19 Master mode transmit interrupt 8 by enabling FIFO (SSR:DMA="0", IBCR:WSEL="1", IBSR:RSA="0")



S: Start condition

W: Data direction bit (write direction)

P: Stop condition

Sr: Iteration start condition

△: Interrupt by INTE="1"

▲: Interrupt by CNDE="1"

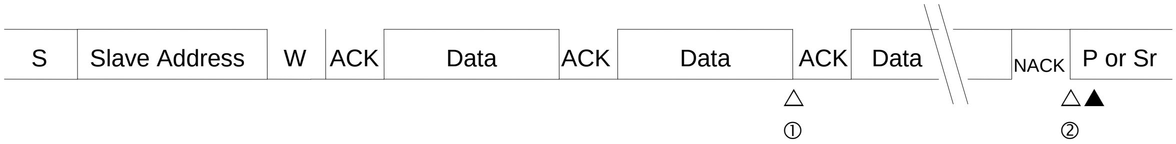
① An interrupt occurs if the Send FIFO buffer is emptied.

- The send data is written in the Send FIFO buffer, and INT bit is set to "0".

② An interrupt occurs when the last byte is sent (the Send FIFO buffer is emptied).

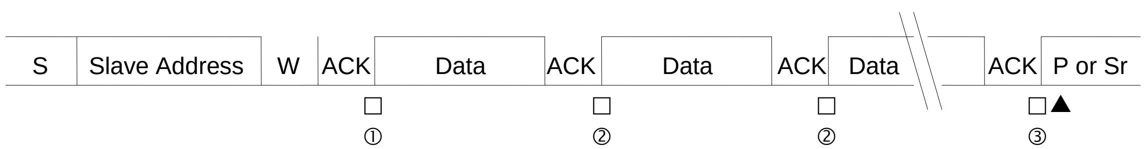
- MSS bit is set to "0", or MSS and SCC bits are set to "1".

Figure 2-20 Master mode transmit interrupt 9 by enabling FIFO (SSR:DMA="0", IBCR:WSEL="1", IBSR:RSA="0", NACK response)



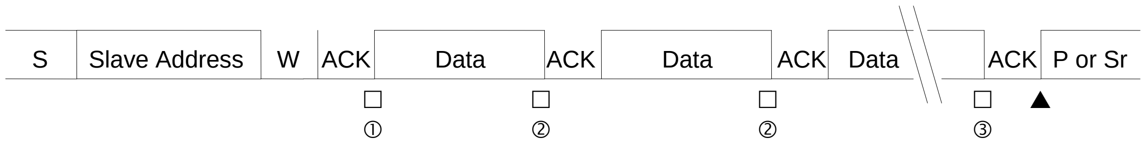
S: Start condition
W: Data direction bit (write direction)
P: Stop condition
Sr: Iteration start condition
△: Interrupt by INTE="1"
▲: Interrupt by CNDE="1"
① An interrupt occurs if the Send FIFO buffer is emptied.
- The send data is written in the Send FIFO buffer, and INT bit is set to "0".
② An interrupt occurs when a NACK is responded.
- MSS bit is set to "0", or MSS and SCC bits are set to "1".

Figure 2-21 Master mode interrupt 10 by disabling FIFO (SSR:DMA="1", IBCR:WSEL="0", IBSR:RSA="0")



S: Start condition
W: Data direction bit (write direction)
P: Stop condition
Sr: Iteration start condition
▲: Interrupt by CNDE="1"
□: Interrupt by TBIE="1"
① An interrupt occurs when the slave address is sent, the direction bit is sent, and an ACK is received.
- The send data is written in the TDR register.
② An interrupt occurs when a single byte is sent and an ACK is received.
- The send data is written in the TDR register.
③ An interrupt occurs when a single byte is sent and an ACK is received.
- MSS bit is set to "0", or MSS and SCC bits are set to "1".
*) If an interrupt flag (TBI) is set, the TDRE bit is set to "1".

Figure 2-22 Master mode transmit interrupt 11 by disabling FIFO (SSR:DMA="1", IBCR:WSEL="1", IBSR:RSA="0", ACK response)



S: Start condition

W: Data direction bit (write direction)

P: Stop condition

Sr: Iteration start condition

▲: Interrupt by CNDE="1"

□: Interrupt by TBIE="1"

① An interrupt occurs when the slave address is sent, the direction bit is sent, and an ACK is received.

- The send data is written in the TDR register.

② An interrupt occurs when a single byte is sent.

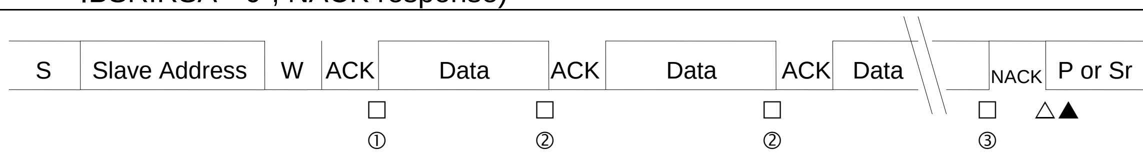
- The send data is written in the TDR register.

③ An interrupt occurs when a single byte is sent.

- MSS bit is set to "0", or MSS and SCC bits are set to "1".

*) If an interrupt flag (TBI) is set, the TDRE bit is set to "1".

Figure 2-23 Master mode transmit interrupt 12 by disabling FIFO (SSR:DMA="1", IBCR:WSEL="1", IBSR:RSA="0", NACK response)



S: Start condition

W: Data direction bit (write direction)

P: Stop condition

Sr: Iteration start condition

△: Interrupt by INTE="1"

▲: Interrupt by CNDE="1"

□: Interrupt by TBIE="1"

① An interrupt occurs when the slave address is sent, the direction bit is sent, and an ACK is received.

- The send data is written in the TDR register.

② An interrupt occurs when a single byte is sent.

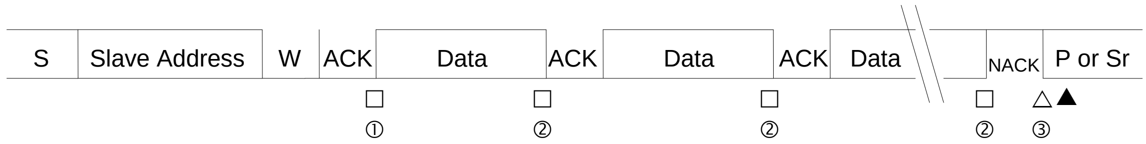
- The send data is written in the TDR register.

③ An interrupt occurs when a single byte is sent.

- MSS bit is set to "0", or MSS and SCC bits are set to "1".

*) If an interrupt flag (INT or TBI) is set, the TDRE bit is set to "1".

Figure 2-24 Master mode transmit interrupt 13 by disabling FIFO (SSR:DMA="1", IBCR:WSEL="1", IBSR:RSA="0", NACK response during transmission)



S: Start condition

W: Data direction bit (write direction)

P: Stop condition

Sr: Iteration start condition

△: Interrupt by INTE="1"

▲: Interrupt by CNDE="1"

□: Interrupt by TBIE="1"

① An interrupt occurs when the slave address is sent, the direction bit is sent, and an ACK is received.

- The send data is written in the TDR register.

② An interrupt occurs when a single byte is sent.

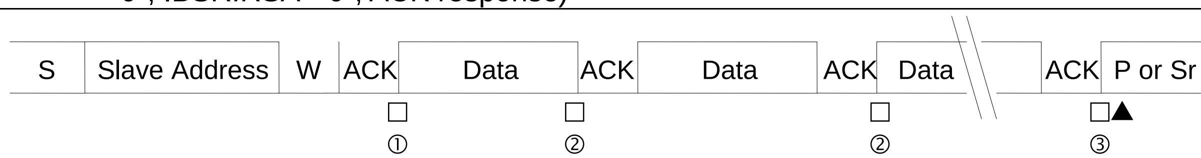
- The send data is written in the TDR register.

③ An interrupt occurs when a NACK is responded.

- MSS bit is set to "0", or MSS and SCC bits are set to "1".

*) If an interrupt flag (INT or TBI) is set, the TDRE bit is set to "1".

Figure 2-25 Master mode transmit interrupt 14 by disabling FIFO (SSR:DMA="1", IBCR:WSEL="1" -> "0", IBSR:RSA="0", ACK response)



S: Start condition

W: Data direction bit (write direction)

P: Stop condition

Sr: Iteration start condition

▲: Interrupt by CNDE="1"

□: Interrupt by TBIE="1"

① An interrupt occurs when the slave address is sent, the direction bit is sent, and an ACK is received.

- The send data is written in the send buffer.

② An interrupt occurs when a single byte is sent.

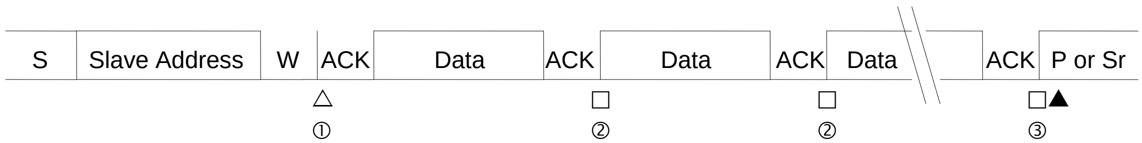
- The WSEL bit is set to "0" and the send data is written in the send buffer.

③ An interrupt occurs when a single byte is sent.

- MSS bit is set to "0", or MSS and SCC bits are set to "1".

*) If an interrupt flag (TBIE) is set, the TDRE bit is set to "1".

Figure 2-26 Master mode interrupt 15 by disabling FIFO (SSR:DMA="1", IBCR:WSEL="0", IBSR:RSA="1")



S: Start condition

W: Data direction bit (write direction)

P: Stop condition

Sr: Iteration start condition

△: Interrupt by INTE="1"

▲: Interrupt by CNDE="1"

□: Interrupt by TBIE="1"

① An interrupt occurs when the slave address (reserved address) is sent, a direction bit is sent, and an ACK is received.

- The send data is written in the TDR register, and the INT bit is set to "0".

② An interrupt occurs when a single byte is sent and an ACK is received.

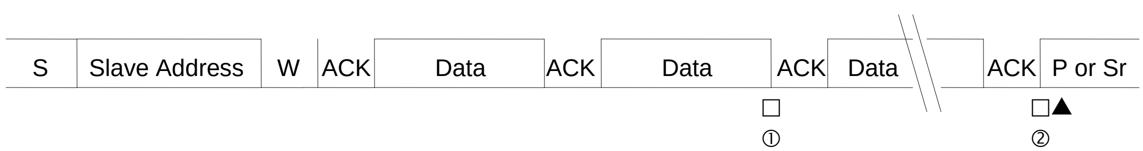
- The send data is written in the TDR register.

③ An interrupt occurs when a single byte is sent and an ACK is received.

- MSS bit is set to "0", or MSS and SCC bits are set to "1".

*) If an interrupt flag (INT or TBI) is set, the TDRE bit is set to "1".

Figure 2-27 Master mode transmit interrupt 16 by enabling FIFO (SSR:DMA="1", IBCR:WSEL="0", IBSR:RSA="0", ACK response)



S: Start condition

W: Data direction bit (write direction)

P: Stop condition

Sr: Iteration start condition

△: Interrupt by INTE="1"

▲: Interrupt by CNDE="1"

□: Interrupt by TBIE="1"

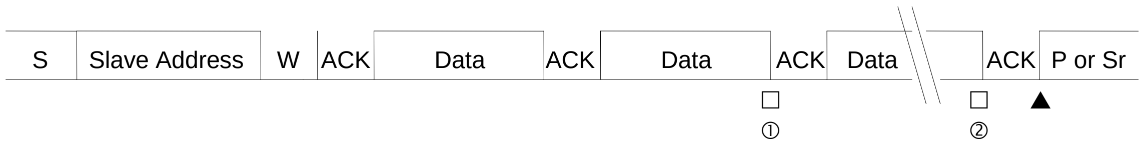
① An interrupt occurs if the Send FIFO buffer is emptied.

- The send data is written in the Send FIFO buffer.

② An interrupt occurs when the last byte is sent (the Send FIFO buffer is emptied) and an ACK is received.

- MSS bit is set to "0", or MSS and SCC bits are set to "1".

Figure 2-28 Master mode transmit interrupt 17 by enabling FIFO (SSR:DMA="1", IBCR:WSEL="1", IBSR:RSA="0")



S: Start condition

W: Data direction bit (write direction)

P: Stop condition

Sr: Iteration start condition

△: Interrupt by INTE="1"

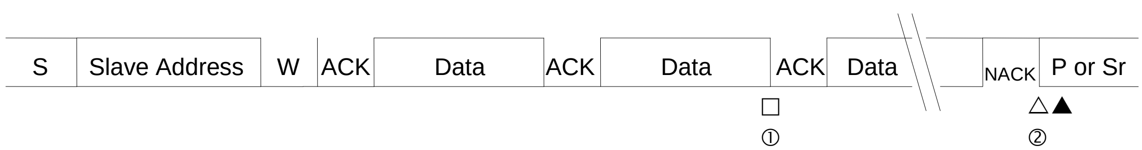
▲: Interrupt by CNDE="1"

□: Interrupt by TBIE="1"

① An interrupt occurs if the Send FIFO buffer is emptied.
- The send data is written in the Send FIFO buffer.

② An interrupt occurs when the last byte is sent (the Send FIFO buffer is emptied).
- MSS bit is set to "0", or both MSS and SCC bits are set to "1".

Figure 2-29 Master mode transmit interrupt 18 by enabling FIFO (SSR:DMA="1", IBCR:WSEL="1", IBSR:RSA="0", NACK response)



S: Start condition

W: Data direction bit (write direction)

P: Stop condition

Sr: Iteration start condition

△: Interrupt by INTE="1"

▲: Interrupt by CNDE="1"

□: Interrupt by TBIE="1"

① An interrupt occurs if the Send FIFO buffer is emptied.
- The send data is written in the Send FIFO buffer.

② An interrupt occurs when a NACK is responded.
- MSS bit is set to "0", or both MSS and SCC bits are set to "1".

■ Data reception by the master

● When DMA mode is disabled (SSR:DMA=0)

When the data direction bit (R/W) is set to "1", the master receives data transmitted from a slave.

When FIFO is disabled, the master operates as follows.

- If the SSR:TDRE bit is set to "1", wait is generated (IBCR:INT=1, SSR:RDRF=1) each time one byte is received . At this time, an ACK or NACK response is returned, according to the setting of the ACKE bit in the IBCR register, before wait if the IBCR:WSEL bit is "1", and after wait if the IBCR:WSEL bit is "0".
- If the SSR:TDRE bit is set to "0", the next data is received without generating wait (IBCR:INT=0) when an ACK response is set for the ACKE bit in the IBCR register while wait is generated when the NACK response is set (IBCR:INT=1).

When FIFO is enabled, the SSR:RDRF bit is set to "1" upon reception of data in the same number of bytes set for the number of bytes to be received. The interrupt flag is set to "1" when the SSR:TDRE bit is "1", which puts the I^2C bus in the wait state. At this time, acknowledgement operates as follows. Even if NACK is output, it is stored in received FIFO as received data.

- In case of IBCR:WSEL=0, an NACK response is returned when the SSR:TDRE bit is set to "1" if NACK is set for the ACKE bit.
- In case of IBCR:WSEL=1, the interrupt flag is set to "1" after receiving the final byte, which generates wait. During that wait, an ACK or NACK response is returned according to the IBCR:ACKE setting after the IBCR:ACKE bit is set and the interrupt flag is cleared to "0".

For interrupt-generated wait, refer to the following.

Table 2-6 IBCR:WSEL bit status for master data reception when DMA mode is disabled (SSR:DMA=0)

WSEL bit	Operation
0	After the second byte, after acknowledgement with "1" set for the SSR:TDRE bit, the interrupt flag (IBCR:INT) is set to "1" and SCL to LOW for the wait state.
1	After the second byte, after the master has received one-byte data with "1" set for the SSR:TDRE bit, the interrupt flag (IBCR:INT) is set to "1" and SCL to LOW for the wait state.

The following shows an example procedure for receiving data from a slave.

- When received FIFO is disabled:
  1. Sets Slave Address (including the data direction bit) to the TDR register and writes "1" to the IBCR:MSS bit.
  2. ACK is received after the Slave Address setting is transmitted, and then the interrupt flag (IBCR:INT) is set to "1".
  3. Writes "0" to the interrupt flag bit (IBCR:INT) upon updating of the IBCR:WSEL bit to release the wait state of the I^2C bus.
  4. After receiving one byte, sets the interrupt flag to "1" to set the I^2C bus in the wait state after transmitting acknowledgment in case of IBCR:WSEL=0 and directly after receiving one byte in case of IBCR:WSEL=1. Repeats steps 3 to 4 until all the specified number of data sets have been received.
  5. After receiving the last data, outputs NACK and sets the IBCR:MSS bit to "0" or sets the IBCR:SCC bit to "1" to generate the stop condition or iteration start condition.

· When transmit/received FIFO is enabled:
1. Sets the number of bytes to be received to the FBYTE register.
2. Writes Slave Address (including the data direction bit) and dummy data in the number of bytes to be received to the TDR register.
3. Writes "1" to the IBCR:MSS bit.
4. An ACK response is returned and data reception continues as long as the SSR:TDRE bit stays "0". During that reception operation, SSR:RDRF is set to "1" when the number of bytes set up in FBYTE have been received. When SSR:RDRF is set to "1", starts reading from the RDR register.
5. When SSR:TDRE bit is "1", sets the interrupt flag to "1" to set the I^2C bus in the wait state after outputting NACK if IBCR:WSEL=0, and directly after one-byte reception if IBCR:WSEL=1.
6. In case of IBCR:WSEL=1, sets the IBCR:ACKE bit to "0". In case of IBCR:WSEL=0, no setting is needed for the IBCR:ACKE bit, Setting the IBCR:MSS bit to "0" or setting the IBCR:SCC bit to "1" generates the stop condition or iteration start condition.

● When DMA mode is enabled (SSR:DMA=1)

When the data direction bit (R/W) is set to "1", the master receives data transmitted from a slave.

When FIFO is disabled, the master operates as follows.

· If the SSR:TDRE bit is set to "1", wait is generated (SSR:TBI=1, SSR:RDRF=1) each time one byte is received. At this time, an ACK or NACK response is returned, according to the setting of the ACKE bit in the IBCR register, before wait if the IBCR:WSEL bit is "1", and after wait if the IBCR:WSEL bit is "0".
· If the SSR:TDRE bit is set to "0", wait is generated (SSR:RDRF=1) each time one byte is received. At this time, an ACK or NACK response is returned, according to the setting of the ACKE bit in the IBCR register, before wait if the IBCR:WSEL bit is "1", and after wait if the IBCR:WSEL bit is "0".

When FIFO is enabled, the SSR:RDRF bit is set upon reception of data in the same number of bytes set for the number of bytes to be received. The transmit bus idle flag (SSR:TBI) is set when the SSR:TDRE bit is "1", which puts the I^2C bus in the wait state. At this time, acknowledgement operates as follows. Even if NACK is output, it is stored in received FIFO as received data.

· In case of IBCR:WSEL=0, an NACK response is returned when the SSR:TDRE bit is set to "1" if NACK is set for the ACKE bit.
· In case of IBCR:WSEL=1, wait is generated (SSR:TBI=1) after receiving the last byte. During that wait, the master sets the IBCR:ACKE bit and returns ACK or NACK response, according to the IBCR:ACKE setting, after clearing the transmit bus idle flag (SSR:TBI).

For interrupt-generated wait, refer to the following.

Table 2-7 IBCR:WSEL bit status for master data reception when DMA mode is enabled (SSR:DMA=1)

WSEL bit	Operation
0	After the second byte, after acknowledgement with "1" set for the SSR:TDRE bit, the transmit bus idle flag (SSR:TBI) is set to "1" and SCL to LOW for the wait state. After the second byte, after acknowledgement with received FIFO is unused, if the received data full flag (SSR:RDRF) is set to "1", SCL is set to LOW for the wait state.
1	After the second byte, after the master has received one-byte data with "1" set for the SSR:TDRE bit, the interrupt flag (SSR:TBI) is set to "1" and SCL to LOW for the wait state. After the second byte, after the received data full flag (SSR:RDRF) is set to "1" when received FIFO is not used, SCL is set to LOW for the wait state.

The following shows an example procedure for receiving data from a slave.

When received FIFO is disabled:

1. Sets Slave Address (including the data direction bit) to the TDR register and writes "1" to the IBCR:MSS bit.
2. ACK is received after the Slave Address setting is transmitted, and then the transmit bus idle flag (SSR:TBI) is set to "1".
3. Writes data to be transmitted to the TDR register to release the wait state of the I^2C bus.
4. After one byte is received, sets the transmit bus idle flag (SSR:TBI) and the received data full flag (SSR:RDRF)*2 to "1" under the following conditions to put the I^2C bus in the wait state.
   - In case of IBCR:WSEL=0, after transmitting acknowledgement
   - In case of IBCR:WSEL=1, after receiving one byte
5. Updates the IBCR:WSEL bit, reads from the RDR register and writes dummy data to the TDR register.
6. After one byte is received, sets the transmit bus idle flag (SSR:TBI) and the received data full flag (SSR:RDRF)*2 to "1" under the following conditions to put the I^2C bus in the wait state.
   - In case of IBCR:WSEL=0, after transmitting acknowledgement
   - In case of IBCR:WSEL=1, after receiving one byte

   Repeats steps 5 to 6 until all the specified number of data sets have been received.
7. After receiving the last data, outputs NACK and sets the IBCR:MSS bit to "0" or sets the IBCR:SCC*1 bit to "1" to generate the stop condition or iteration start condition.

- When transmit/received FIFO is enabled:

1. Sets the number of bytes to be received to the FBYTE register.
2. Writes Slave Address (including the data direction bit) and dummy data in the number of bytes to be received to the TDR register.
3. In case of IBCR:WSEL=0, sets NACK for the ACKE bit, and writes "1" to the IBCR:MSS bit.
4. An ACK response is returned and data reception continues as long as the SSR:TDRE bit stays "0". During that reception operation, SSR:RDRF is set to "1" when the number of bytes set up in FBYTE have been received. When SSR:RDRF is set to "1", starts reading from the RDR register.
5. When the SSR:TDRE bit is set to "1", sets the interrupt flag to "1" to set the I^2C bus in the wait state after outputting NACK if IBCR:WSEL=0. In case of IBCR:WSEL=1, directly after one byte is received, sets the transmit bus idle flag (SSR:TBI) to "1" to put the I^2C bus in the wait state.
6. In case of IBCR:WSEL=1, sets the IBCR:ACKE bit to "0". In case of IBCR:WSEL=0, no setting is needed for the IBCR:ACKE bit, Set the IBCR:MSS bit to "0" or set the IBCR:SCC*1 bit to "1" to generate the stop condition or iteration start condition.

*1 : When DMA is enabled (SSR:DMA=1), the SSR:TBI bit is "1" and the IBCR:INT bit is "0", follow the steps below to issue the iteration start condition.

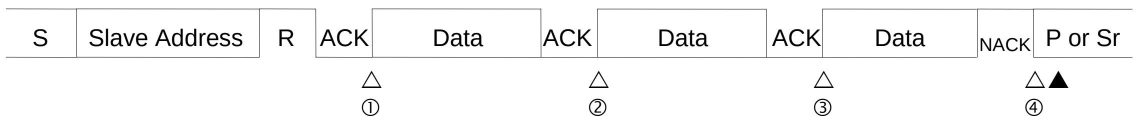
1. Set the IBCR:INT bit to "1".
2. Check that the IBCR:INT bit is set to "1".
3. Write the slave address in the TDR.
4. Set the IBCR:SCC bit to "1".

*2 : Directly after receiving one byte, the received data full flag (SSR:RDRF) is set to "1" regardless of the setting for IBCR:WSEL. When the received data full flag (SSR:RDRF) is set to "1" in the second byte or later, put the I^2C bus in the wait state after transmitting acknowledgment in case of IBCR:WSEL=0, and directly after receiving one byte in case of IBCR:WSEL=1.

<Notes>

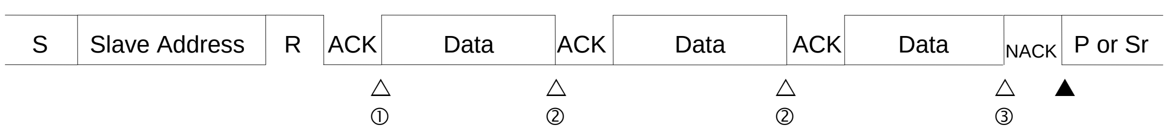
- When seven-bit slave address detection is enabled (ISBA:SAEN=1), it is prohibited to specify a seven-bit slave address in master mode.
- When SSR:TDRE is "0", even if an overrun error occurs, acknowledgement is output according to the setting for the IBCR:ACKE bit, and then the next process should follow.
- To change the IBCR register during transmission/reception, do so when the interrupt flag (IBCR:INT) is "1" or when the transmit bus idle flag (SSR:TBI) is "1" during DMA mode being enabled (SSR:DMA=1).
- In the master mode reception with DMA disabled (SSR:DMA=0), write dummy data to the TDR register, and then, if the SSR:TDRE bit is "0" when the interrupt flag (IBCR:INT) is turned to "1", receive the next data with the interrupt flag (IBCR:INT) kept at "0".
- In the master mode reception with DMA enabled (SSR:DMA=1), write dummy data to the TDR register, and then, if the SSR:TDRE bit is "0" when the transmit bus idle flag (SSR:TBI) is turned to "1", receive the next data with the transmit bus idle flag (SSR:TBI) kept at "0".
- To receive data when received FIFO is enabled and IBCR:WSEL=0, the SSR:RDRF bit is set to "1" after receiving the last bit and the interrupt flag (IBCR:INT) is set to "1" after transmitting ACK.

Figure 2-30 Master mode received interrupt 1 by disabling FIFO (SSR:DMA="0", IBCR:WSEL="0", IBSR:RSA="0")



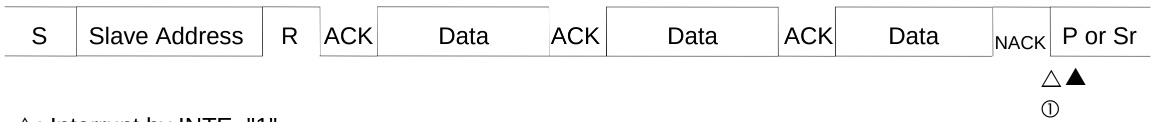
△: Interrupt by INTE="1"
▲: Interrupt by CNDE="1"
① An interrupt occurs when the slave address is sent, the direction bit is sent, and an ACK is received.
- If the INT bit is set to "0", the interrupt flag is cleared to "0".
② An interrupt occurs when a single byte is received and an ACK is sent.
- After the received data has been read, the INT bit is set to "0".
③ An interrupt occurs when a single byte is received and an ACK is sent.
- After the received data has been read, both ACKE and INT bits are set to "0".
④ An interrupt occurs when a single byte is received and an ACK is sent.
- MSS bit is set to "0", or both MSS and SCC bits are set to "1".
*) If an interrupt flag (INT) is set, the TDRE bit is set to "1".

Figure 2-31 Master mode received interrupt 2 by disabling FIFO (SSR:DMA="0", IBCR:WSEL="1", IBSR:RSA="0")



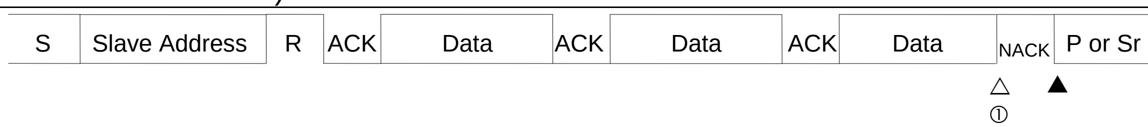
△: Interrupt by INTE="1"
▲: Interrupt by CNDE="1"
① An interrupt occurs when the slave address is sent, the direction bit is sent, and an ACK is received.
- If the INT bit is set to "0", the interrupt flag is cleared to "0".
② An interrupt occurs when a single byte is received.
- After the received data has been read, the INT bit is set to "0".
③ An interrupt occurs when a single byte is received.
- After the received data has been read, ACKE bit are set to "0". MSS bit is set to "0", or both MSS and SCC bits are set to "1".
*) If an interrupt flag (INT) is set, the TDRE bit is set to "1".

Figure 2-32 Master mode received interrupt 3 by enabling FIFO (SSR:DMA="0", IBCR:WSEL="0", IBCR:ACKE="0", IBSR:RSA="0")



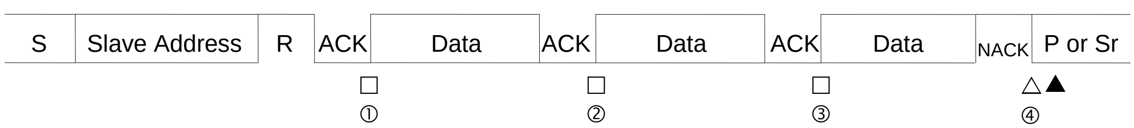
△: Interrupt by INTE="1"
▲: Interrupt by CNDE="1"
① An interrupt occurs if TDRE bit is set to "1".
- The entire data is read from the Received FIFO buffer, and MSS bit is set to "0" or both MSS and SCC bits are set to "1".

Figure 2-33 Master mode received interrupt 4 by enabling FIFO (SSR:DMA="0", IBCR:WSEL="1", IBSR:RSA="0")



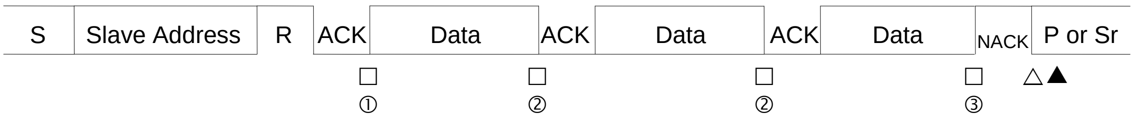
△: Interrupt by INTE="1"
▲: Interrupt by CNDE="1"
① An interrupt occurs if TDRE bit is set to "1".
- After the entire data has been read from the Received FIFO buffer, ACKE bit is set to "0" and MSS bit is set to "0" or both MSS and SCC bits are set to "1".

Figure 2-34 Master mode received interrupt 5 by disabling FIFO (SSR:DMA="1", IBCR:WSEL="0", IBSR:RSA="0")



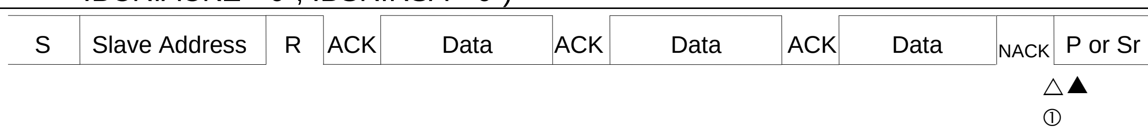
△: Interrupt by INTE="1"
▲: Interrupt by CNDE="1"
□: Interrupt by TBIE="1"
① An interrupt occurs when the slave address is sent, the direction bit is sent, and an ACK is received.
- Dummy data is written in the TDR register.
② An interrupt occurs when a single byte is received and an ACK is sent.
- After the received data has been read, the dummy data is written in the TDR register.
③ An interrupt occurs when a single byte is received and an ACK is sent.
- After the received data has been read, the ACKE bit are set to "0" and the dummy data is written in the TDR register.
④ An interrupt occurs when a single byte is received and an ACK is sent.
- MSS bit is set to "0", or both MSS and SCC bits are set to "1".
*) If an interrupt flag (INT or TBI) is set, the TDRE bit is set to "1".

Figure 2-35 Master mode received interrupt 6 by disabling FIFO (SSR:DMA="1", IBCR:WSEL="1", IBSR:RSA="0")



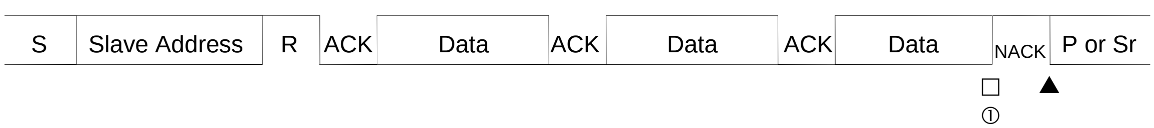
△: Interrupt by INTE="1"
▲: Interrupt by CNDE="1"
□: Interrupt by TBIE="1"
① An interrupt occurs when the slave address is sent, the direction bit is sent, and an ACK is received.
- Dummy data is written in the TDR register.
② An interrupt occurs when a single byte is received.
- After the received data has been read, the dummy data is written in the TDR register.
③ An interrupt occurs when a single byte is received.
- After the received data has been read, ACKE bit are set to "0" MSS bit is set to "0", or both MSS and SCC bits are set to "1".
*) If an interrupt flag (INT or TBI) is set, the TDRE bit is set to "1".

Figure 2-36 Master mode received interrupt 7 by enabling FIFO (SSR:DMA="1", IBCR:WSEL="0", IBCR:ACKE="0", IBSR:RSA="0")



△: Interrupt by INTE="1"
▲: Interrupt by CNDE="1"
① An interrupt occurs if TDRE bit is set to "1".
- The entire data is read from the Received FIFO buffer, and MSS bit is set to "0" or both MSS and SCC bits are set to "1".

Figure 2-37 Master mode received interrupt 8 by enabling FIFO (SSR:DMA="1", IBCR:WSEL="1", IBSR:RSA="0")



□: Interrupt by TBIE="1"
▲: Interrupt by CNDE="1"
① An interrupt occurs if TDRE bit is set to "1".
- After the entire data has been read from the Received FIFO buffer, ACKE bit is set to "0" and MSS bit is set to "0" or both MSS and SCC bits are set to "1".

■ Arbitration lost

If the master receives the data different from sent data, due to collision of data from another master, the master judges the situation as arbitration lost. At this time, the IBCR:MSS bit is set to "0" and the IBSR:AL bit to "1", enabling operation in slave mode.

The IBSR:AL bit can be cleared to "0" under the following conditions:

- The IBCR:MSS bit is set to "1".
- The IBCR:INT bit is set to "0".
- The IBSR:SPC bit is set to "0" when the IBSR:AL bit and IBSR:SPC bit are "1".
- The I^2C interface operation is disabled (ISMK:EN=0).

Upon an occurrence of arbitration lost, the interrupt flag (IBCR:INT) is set to "1" according to the setting of the IBCR:WSEL bit, and sets SCL of the I^2C bus to LOW.

■ Wait state for master mode

When both conditions below are satisfied, master mode is put in the wait state while the IBSR:BB bit stays "1". After the IBSR:BB bit attains "0", start condition is transmitted.

- When the IBCR:MSS is set to "1" while the IBSR:BB bit is "1"
- When the interface is not operating as a slave

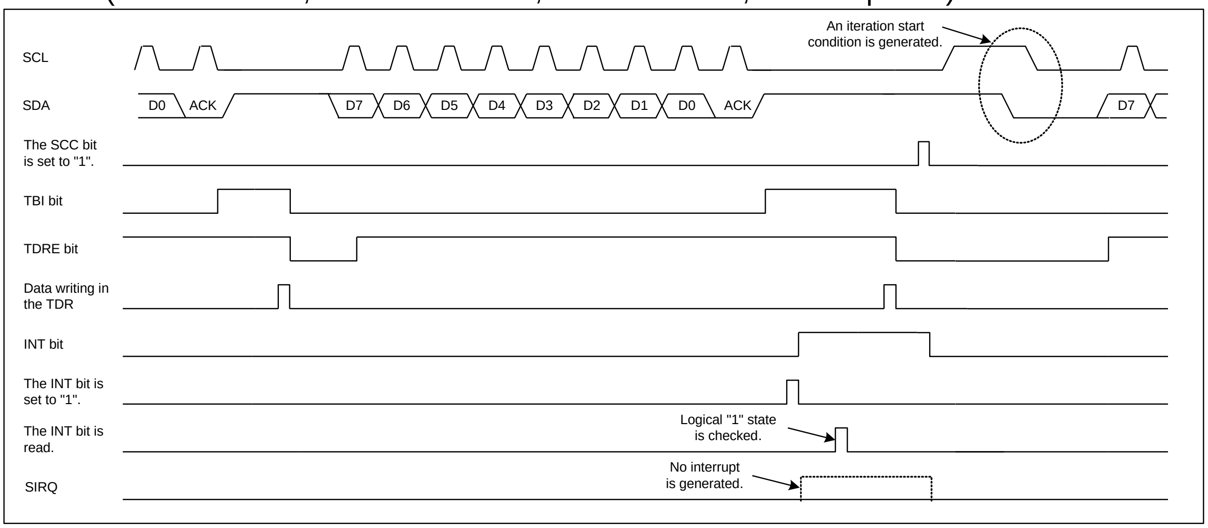
Refer to the IBCR:MSS bit and IBCR:ACT bit to check if master mode is in the wait state or not (in the wait state if the IBCR:MSS=1 and IBCR:ACT=0). After setting the IBCR:MSS bit to "1" and to operate in slave mode, set the IBSR:AL bit to "1", the IBCR:MSS bit to "0", and the IBCR:ACT bit to "1".

■ Issuing iteration start condition when DMA mode is enabled (SSR:DMA=1)

When writing a slave address to the TDR register while the transmit bus is idle (SSR:TBI=1) and the interrupt flag (IBCR:INT) is "0", transmission starts and the iteration start condition cannot be issued.
Therefore, to issue the iteration start condition while the transmit bus is idle (SSR:TBI=1) and the interrupt flag (IBCR:INT) is "0", follow the steps below.

1. Set the IBCR:INT bit to "1". At this time, no SIRQ interrupt is generated.
2. Check that the IBCR:INT bit is set to "1".
3. Write the slave address in the TDR.
4. Issue the iteration start condition (IBCR:SCC=1).

Figure 2-38 Issuing iteration start condition when DMA mode is enabled (SSR:DMA="1", IBCR:WSEL="0", IBSR:RSA="0", ACK response)



2.3. Slave mode

If the (iteration) start condition is detected and a combination of the ISBA and ISMK registers matches the received address, the interface outputs an ACK response and acts in slave mode.

<Note>

When EIBCR:BEC set to "0", If a start condition is detected again while transferring address data after a start condition is detected or while transferring bit2 to bit19 (acknowledge bits), the next data cannot be received since a bus error (IBCR:BER = 1) is detected and reception is stopped. In such a case, a start condition must be retransmitted from the master after clearing the interrupt flag (IBCR:INT).

■ Slave address match detection

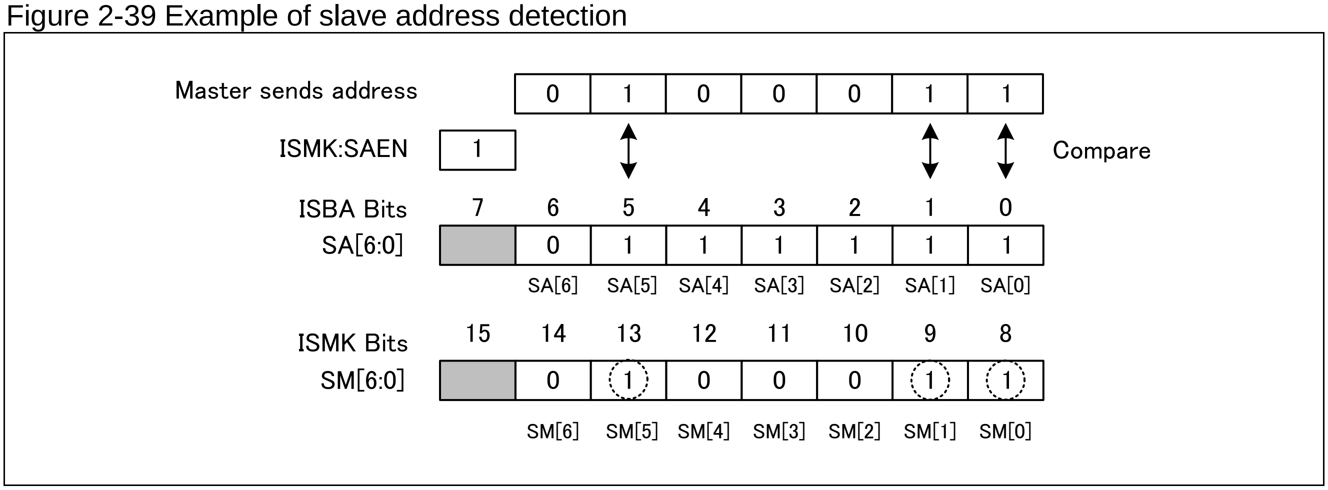
The 7-bit slave address and the direction of a data transfer is contained in the first byte after detection of a start or repeated start condition. The ISMK becomes the value to mask the slave address: a zero mask value designates a don’t care, and a 1 must be a direct match. In other words, if a mask bit is set to 0 in the ISMK register, the address bit is not compared.

The SAEN is the enables the slave address detection when set. The address that is sent from master is compared with the slave address bits (SA[6:0]) that sets the mask bits (SM[6:0]) to 1. If they match, an ACK is output. If there is no match, or SAEN is 0, no ACK is output.

● Example of a slave address detection

Master addresses slave address 0x23.

Figure 2-39 Example of slave address detection



Only SA5, SA1, and SA0 are compared to the address sent by master because the SM[6] and SM[4:2] are zero and therefore are don't care. the multifunction serial interface outputs an ACK response.

Table 2-8 Operation immediately after outputting acknowledgement to a slave address

Transmit FIFO	Received FIFO	Transmit FIFO status	Received FIFO status	Data direction bit (R/W)	Operation immediately after receiving acknowledgement	
					Acknowledgement: ACK	Acknowledgement: NACK
Disable	Disable	-	-	0	If the SSR:TDRE bit is set to "1", the interface sets the IBCR:INT bit to "1" and waits. If the SSR:TDRE bit is set to "0", IBCR:INT bit stays "0" without the wait state.	Holds the IBCR:INT bit to "0" without the wait state.
				1		
Disable	Enable	-	Without data	0	Holds the IBCR:INT bit to "0" without the wait state.	Holds the IBCR:INT bit to "0" without the wait state.
			With data		Sets the IBCR:INT bit to "1" with the wait state.	
			-	1	If the SSR:TDRE bit is set to "1", the interface sets the IBCR:INT bit to "1" and waits. If the SSR:TDRE bit is set to "0", IBCR:INT bit stays "0" without the wait state.	
Enable	Disable	-	-	0	If the SSR:TDRE bit is set to "1", the interface sets the IBCR:INT bit to "1" and waits. If the SSR:TDRE bit is set to "0", IBCR:INT bit stays "0" without the wait state.	Holds the IBCR:INT bit to "0" without the wait state.
				1		
Enable	Enable	-	Without data	0	Holds the IBCR:INT bit to "0" without the wait state.	Holds the IBCR:INT bit to "0" without the wait state.
			With data		Sets the IBCR:INT bit to "1" with the wait state.	
			-	1	If the SSR:TDRE bit is set to "1", the interface sets the IBCR:INT bit to "1" and waits. If the SSR:TDRE bit is set to "0", IBCR:INT bit stays "0" without the wait state.	

- Detection of reserved address

  If the first byte matches the reserved address ("0000xxxx" or "1111xxxx"), the value of 8th bit is received regardless of whether or not transmit/received FIFO is enabled, and the IBCR:INT bit is set to "1", causing the I^2C bus to be placed into the wait state. After the received data has been read, configure the following settings.

  - To run the interface as a slave device, set the IBCR:ACKE bit to "1" and check the value of the data direction bit (IBSR:TRX). If the transmitting direction is set, write the transmit data to TDR, and clear the IBCR:INT bit. The interface then acts as a slave device.
  - When not running the interface as a slave device, set the IBCR:ACKE bit to "0", and clear the IBCR:INT bit. After acknowledgement has been output, the interface does not act as a slave device.

■ Data direction bit

After receiving the address, the interface receives the data direction bit to determine whether to transmit or receive data. If this bit is "0", it means that data is transmitted from the master device, and the interface receives data as a slave device.

■ Reception in slave mode

If the received data matches the slave address and the data direction bit is "0", it means that data is received in slave mode. The following shows a procedure example to receive data in slave mode.

● When DMA mode is disabled (SSR:DMA=0)

· When received FIFO is disabled:

1. After transmitting ACK, set the interrupt flag (IBCR:INT) to "1", and place the I^2C bus into the wait state. Based on the IBCR:MSS, IBCR:ACT, and IBSR:FBT bits, judge that the event is an interrupt by a slave address match. Then write "1" to the IBCR:ACKE bit and "0" to the interrupt flag (IBCR:INT), and release the wait state of the I2C bus (see Table 2-8).
2. After receiving 1-byte data, set the interrupt flag (IBCR:INT) to "1" according to setting of the IBCR:WSEL bit, and place the I^2C bus into the wait state.
3. Read the data received from the RDR register, set the IBCR:ACKE bit, write "0" to the interrupt flag (IBCR:INT), and release the wait state of the I^2C bus.
4. Repeat steps 2 and 3 to detect the stop or iteration start condition.

· When received FIFO is enabled:

1. If NACK is detected or received FIFO becomes full, the interrupt flag (IBCR:INT) is set to "1", and the I^2C bus is placed into the wait state. If the stop or iteration start condition is detected, the interrupt flag (IBCR:INT) is not set to "1" (the I^2C bus is not placed into the wait state) by setting the IBSR:SPC and IBSR:RSC bits to "1". Received FIFO sets the SSR:RDRF bit to "1" when the set value of the FBYTE register matches the number of data sets received. If the SMR:RIE bit is then "1", a received interrupt is generated.
2. When the interrupt flag (IBCR:INT) is set to "1", read the received data from the RDR register. After all data has been read, write "0" to the interrupt flag to release the wait state of the I^2C bus. If the stop or iteration start condition is detected, read all the received data from the RDR register, and clear the IBSR:SPC or IBSR:RSC bit to "0".

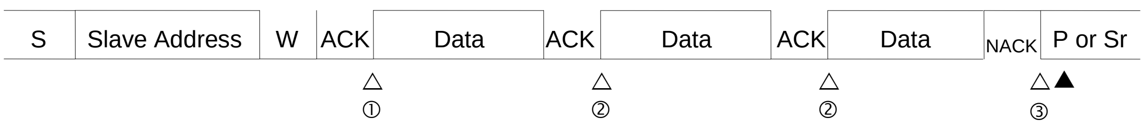
● When DMA mode is enabled (SSR:DMA=1)

· When received FIFO is disabled:

1. After transmitting ACK, set the interrupt flag (IBCR:INT) to "1", and place the I^2C bus into the wait state. Based on the IBCR:MSS, IBCR:ACT, and IBSR:FBT bits, judge that the event is an interrupt by a slave address match. Then write "1" to the IBCR:ACKE bit and "0" to the interrupt flag (IBCR:INT), and release the wait state of the I^2C bus (see Table 2-8).
2. Set "1" to the received data full flag (SSR:RDRF) immediately after receiving 1-byte data. When the received data full flag (SSR:RDRF) is set to "1", if IBCR:WSEL=0, place the I^2C bus into the wait state after transmitting acknowledgement. If IBCR:WSEL=1, place the I^2C bus into the wait state immediately after receiving the 1-byte data.
3. After setting the IBCR:ACKE bit, read the data received from the RDR register, and clear the received data full flag (SSR:RDRF) to "0" to release the wait state of the I^2C bus.
4. Repeat steps 2 and 3 to detect the stop or iteration start condition.

- When received FIFO is enabled:
  1. If NACK is detected, the interrupt flag (IBCR:INT) is set to "1", and the I^2C bus is placed into the wait state. When received FIFO becomes full, place the I^2C bus into the wait state. If the stop or iteration start condition is detected, the IBSR:SPC and IBSR:RSC bits are set to "1", and the interrupt flag (IBCR:INT) is not set to "1" (the I^2C bus is not placed into the wait state). Received FIFO sets the SSR:RDRF bit to "1" when the set value of the FBYTE register matches the number of data sets received. If the SMR:RIE bit is then "1", a received interrupt is generated.
  2. When the interrupt flag (IBCR:INT) is set to "1", read the received data from the RDR register. After all data has been read, write "0" to the interrupt flag to release the wait state of the I^2C bus. When received FIFO is full, release the wait state of the I^2C bus if the received data is read from the RDR register even once. If the stop or iteration start condition is detected, read all the received data from the RDR register, and clear the IBSR:SPC or IBSR:RSC bit to "0".

Figure 2-40 Slave mode received interrupt 1 by disabling FIFO (SSR:DMA="0", IBCR:WSEL="0", IBSR:RSA="0")



△ :Interrupt by INTE="1"

▲ :Interrupt by CNDE="1"

① As the slave address matches , an ACK is output and an interrupt is generated .

- ACKE bit is set to "1" and INT bit is set to "0".

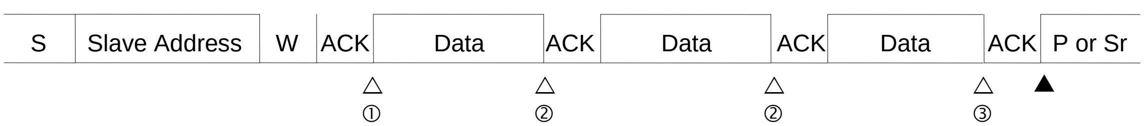
② An interrupt occurs when a single byte is received and an ACK is responded .

- After the received data has been read from the received buffer, the INT bit is set to "0".

③ An interrupt occurs when a single byte is received and a NACK is responded .

- After the received data has been read from the received buffer, the INT bit is set to "0".

Figure 2-41 Slave mode received interrupt 2 by disabling FIFO (SSR:DMA="0", IBCR:WSEL="1", IBSR:RSA="0")



△ :Interrupt by INTE="1"

▲ :Interrupt by CNDE="1"

① As the slave address matches , an ACK is output and an interrupt is generated .

- ACKE bit is set to "1" and INT bit is set to "0".

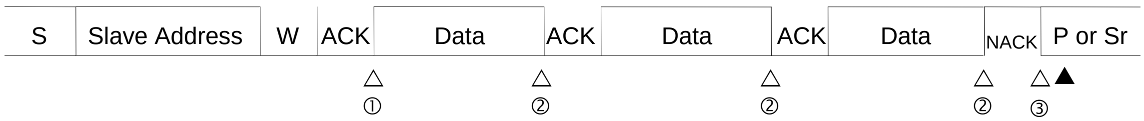
② An interrupt occurs when a single byte is received .

- After the received data has been read from the received buffer, the INT bit is set to "0".

③ An interrupt occurs when a single byte is received .

- After the received data has been read from the received buffer, the INT bit is set to "0".

Figure 2-42 Slave mode received interrupt 3 by disabling FIFO (SSR:DMA="0", IBCR:WSEL="1", IBSR:RSA="0")



△ :Interrupt by INTE="1"

▲ :Interrupt by CNDE="1"

① As the slave address matches , an ACK is output and an interrupt is generated .

- ACKE bit is set to "1" and INT bit is set to "0".

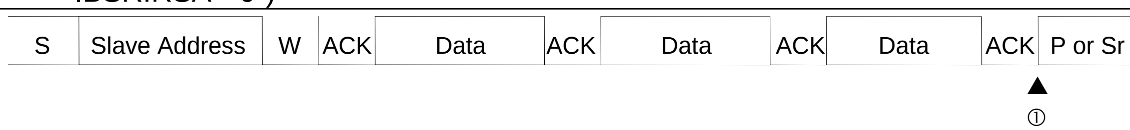
② An interrupt occurs when a single byte is received.

- After the received data has been read from the received buffer, the INT bit is set to "0".

③ An interrupt occurs when a NACK is responded.

- INT bit is set to "0".

Figure 2-43 Slave mode received interrupt 4 by enabling received FIFO (SSR:DMA="0", IBSR:RSA="0")



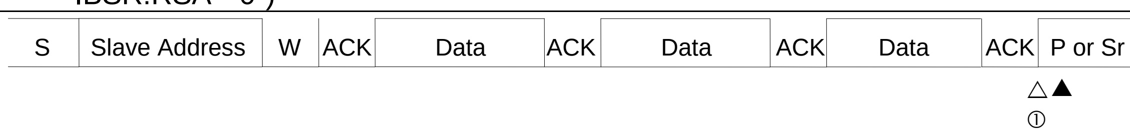
△ :Interrupt by INTE="1"

▲ :Interrupt by CNDE="1"

① An interrupt occurs when the stop condition or the iteration start condition is detected .

- The entire data is read from the Received FIFO buffer

Figure 2-44 Slave mode received interrupt 5 by enabling received FIFO (SSR:DMA="0", IBSR:RSA="0")



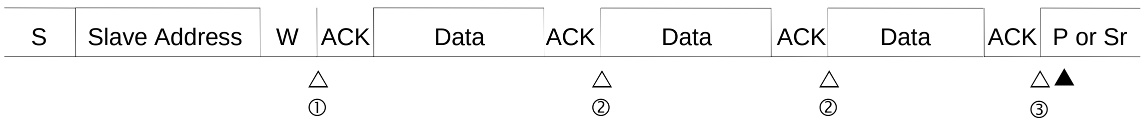
△ :Interrupt by INTE="1"

▲ :Interrupt by CNDE="1"

① An interrupt occurs when the Received FIFO buffer is filled with data

- The entire data is read from the Received FIFO buffer and the INT bit is set to "0".

Figure 2-45 Slave mode received interrupt 6 by disabling FIFO (SSR:DMA="0", IBCR:WSEL="0", IBSR:RSA="1")



△ :Interrupt by INTE="1"

▲ :Interrupt by CNDE="1"

① An interrupt occurs as the reserved address ("0000xxxx" or "1111xxxx") matches.

- The received data is read , and ACKE bit is set to "1" and INT bit is set to "0".

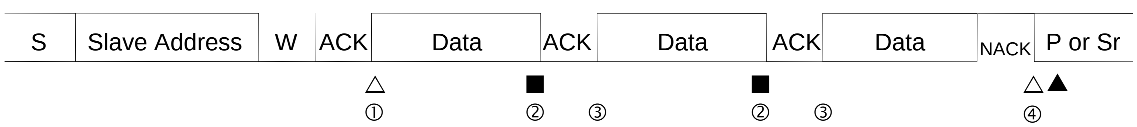
② An interrupt occurs when a single byte is received and an ACK is output .

- INT bit is set to "0".

③ An interrupt occurs when a single byte is received and an ACK is output .

- An interrupt occurs if INT bit is set to "0".

Figure 2-46 Slave mode received interrupt 7 by disabling FIFO (SSR:DMA="1", IBCR:WSEL="0", IBSR:RSA="0")



△ :Interrupt by INTE="1"

▲ :Interrupt by CNDE="1"

■ :Interrupt by RIE="1"

① As the slave address matches , an ACK is output and an interrupt is generated .

- ACKE bit is set to "1" and INT bit is set to "0".

② An interrupt occurs (but the I^2C bus is not waited) when a single byte is received.

- The received data is read from the received buffer

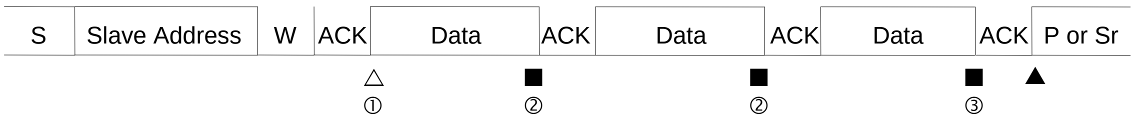
③ The I^2C bus is waited when an ACK is responded.

- The received data is read from the received buffer

④ An interrupt occurs when a single byte is received and a NACK is responded.

- After the received data has been read from the received buffer, the INT bit is set to "0".

Figure 2-47 Slave mode received interrupt 8 by disabling FIFO (SSR:DMA="1", IBCR:WSEL="1", IBSR:RSA="0")



△ :Interrupt by INTE="1"

▲ :Interrupt by CNDE="1"

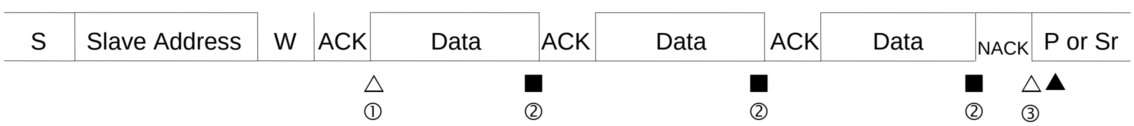
■ :Interrupt by RIE="1"

(1) As the slave address matches, an ACK is output and an interrupt is generated.
- ACKE bit is set to "1" and INT bit is set to "0".

(2) An interrupt occurs when a single byte is received.
- The received data is read from the received buffer

(3) An interrupt occurs when a single byte is received.
- The received data is read from the received buffer

Figure 2-48 Slave mode received interrupt 9 by disabling FIFO (SSR:DMA="1", IBCR:WSEL="1", IBSR:RSA="0")



△ :Interrupt by INTE="1"

▲ :Interrupt by CNDE="1"

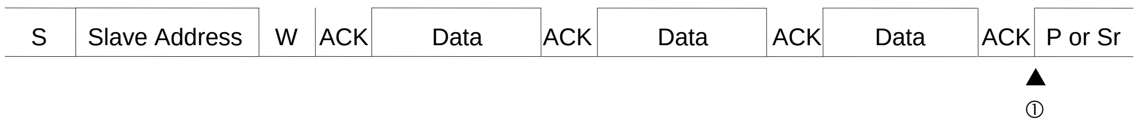
■ :Interrupt by RIE="1"

(1) As the slave address matches, an ACK is output and an interrupt is generated.
- ACKE bit is set to "1" and INT bit is set to "0".

(2) An interrupt occurs when a single byte is received.
- The received data is read from the received buffer

(3) An interrupt occurs when a NACK is responded.
- INT bit is set to "0".

Figure 2-49 Slave mode received interrupt 10 by enabling received FIFO (SSR:DMA="1", IBSR:RSA="0")



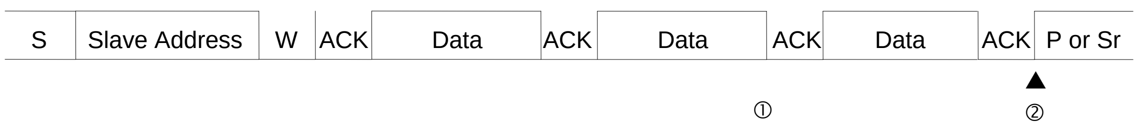
△ :Interrupt by INTE="1"

▲ :Interrupt by CNDE="1"

① An interrupt occurs when the stop condition or the iteration start condition is detected .

- The entire data is read from the Received FIFO buffer

Figure 2-50 Slave mode received interrupt 11 by enabling received FIFO (SSR:DMA="1", IBSR:RSA="0")



▲ :Interrupt by CNDE="1"

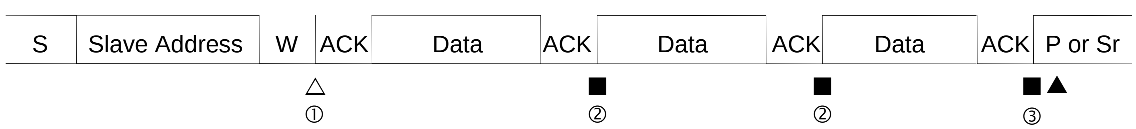
① As the Received FIFO buffer is filled with data, the I^{2}C bus is waited.

- The waiting is released when data is read even once from the Received FIFO buffer

② An interrupt occurs when the stop condition or the iteration start condition is detected .

- The entire data is read from the Received FIFO buffer

Figure 2-51 Slave mode received interrupt 12 by disabling FIFO (SSR:DMA="1", IBCR:WSEL="0", IBSR:RSA="1")



△ :Interrupt by INTE="1"

▲ :Interrupt by CNDE="1"

■ :Interrupt by RIE="1"

① An interrupt occurs as the reserved address ("0000xxxx" or "1111xxxx") matches.

- The received data is read, and ACKE bit is set to "1" and INT bit is set to "0".

② An interrupt occurs when a single byte is received and an ACK is output.

- The received data is read.

③ An interrupt occurs when a single byte is received and an ACK is output.

- The received data is read.

■ Transmission in slave mode

If the received data matches the slave address and the data direction bit is "1", it means that data is transmitted in slave mode. If FIFO is disabled, set the interrupt flag (IBCR:INT) to "1" after transmitting one byte or outputting an acknowledgement response depending on setting of the IBCR:WSEL bit. Then place the I^2C bus into the wait state (see Table 2-8).

Using the IBSR:RACK bit, check the acknowledgement output from the master device. If NACK response is returned from the master device, it means that the master device could not receive data correctly or data receiving was ended. If NACK is detected at IBCR:WSEL=1, an interrupt is generated to place the I^2C bus into the wait state.

2.4. Bus error

If the stop or (iteration) start condition is detected while transmitting or receiving data on the I^2C bus, it is handled as a bus error.

■ Bus error occurrence condition

If a bus error occurs, the IBCR:BER bit is set to "1" in the following conditions.

- The (iteration) start or stop condition is detected while transferring the first byte.
- The (iteration) start condition or stop condition is detected at bit2 to bit9 (acknowledgement) of data.

■ Bus error operation

● EIBCR:BEC=0

If the interrupt flag (IBCR:INT) is set to "1" by transmitting or receiving data, check the IBCR:BER bit. When the IBCR:BER bit is "1", perform error processing. The IBCR:BER bit is cleared by writing "0" to the IBCR:INT bit.

If a bus error occurs, the IBCR:INT bit is set to "1"; however, the I^2C bus is not placed into the wait state by setting its SCL to LOW.

● EIBCR:BEC=1

If the interrupt flag (IBCR:SPC or IBCR:RSC) is set to "1" by transmitting or receiving data, check the IBCR:BER bit. When the IBCR:BER bit is "1", perform error processing. The IBCR:BER bit is cleared by flowing operations.

- When IBCR:INT=1, write "0" in IBCR:INT.
- When IBCR:SPC=1, write "0" in IBCR:SPC.
- When IBCR:RSC=1, write "0" in IBCR:RSC.

3. Dedicated Baud Rate Generator

The dedicated baud rate generator configures the setting of the serial clock frequency.

■ Selecting the baud rate

● Baud rate obtained by dividing an internal clock using the dedicated baud rate generator (reload counter)

This generator provides two internal reload counters, which support transmitting and receiving serial clocks respectively. To select the baud rate, specify the 15-bit reload value using Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0).

Each reload counter divides an internal clock by the set value.

■ Calculating the baud rate

Two 15-bit reload counters are set using the Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0). The baud rate is obtained in the following formulas.

(1) Reload value

$V = \phi / b - 1$

V: Reload value   b: Baud rate ϕ : Bus clock frequency or external clock frequency

Note that the preset baud rate may not be generated at a rising edge of signal on I^2C bus. In such case, adjust the reload value.

(2) Calculation example

To set the 16 MHz bus block and 400 kbps baud rate, set the reload value as follows.

Reload value:

$V = (16 \times 1000000)/400000 - 1 = 39$

Therefore, the baud rate is:

$b=(16 \times 100000) / (39+1) = 400$ kbps

<Notes>

- Write Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0) by 16-bit access operation.
- When the ISMK:EN bit in the ISMK register is "0", set the value of each Baud Rate Generator Register.
- In operation mode 4 (I^2C mode), operate the bus clock at a frequency no lower than 8 MHz. Also note that setting of a baud rate generator that exceeds 400 kbps is prohibited.
- If the reload value is set to "0", the reload counter is stopped.

■ Reload values and baud rates for each bus clock frequency

The following shows the reload values and baud rate setting examples.

Table 3-1 Reload values and baud rate setting examples 1

Baud rate [bps]	8 MHz	10 MHz	16 MHz	20 MHz	24 MHz
	Value	Value	Value	Value	Value
400000	19	24	39	49	59
200000	39	49	79	99	119
100000	79	99	159	199	239

The numeric values above are available when the SCL rising timing of the I^2C bus is 0s. If the SCL rising timing of the I^2C bus is late, the baud rate is set to the value later than the numeric values above.

Table 3-2 Reload values and baud rate setting examples 2

Baud rate [bps]	32 MHz	40 MHz	48 MHz	72 MHz	80 MHz
	Value	Value	Value	Value	Value
400000	79	99	119	179	199
200000	159	199	239	359	399
100000	319	399	479	719	799

The numeric values above are available when the SCL rising timing of the I2C bus is 0 s. If the SCL rising timing of the I2C bus is late, the baud rate is set to the value later than the numeric values above.

■ Functions of reload counter

Each reload counter consists of a 15-bit register for the reload value, and generates transmitting and receiving clocks from internal clocks. The count value of the transmit reload counter can be read from the Baud Rate Generator Registers (BGR1 and BGR0).

■ Starting counting

When the reload value is written to the Baud Rate Generator Register (BGR1 or BGR0), the reload counter starts counting.

4. I²C communication operation flowchart examples

This section shows I²C communication operation flowchart examples.

■ I²C flowchart example (FIFO not used) when DMA mode is disabled (SSR:DMA=0)

Figure 4-1 I²C flowchart example (FIFO not used) when DMA mode is disabled (SSR:DMA=0) 1/3

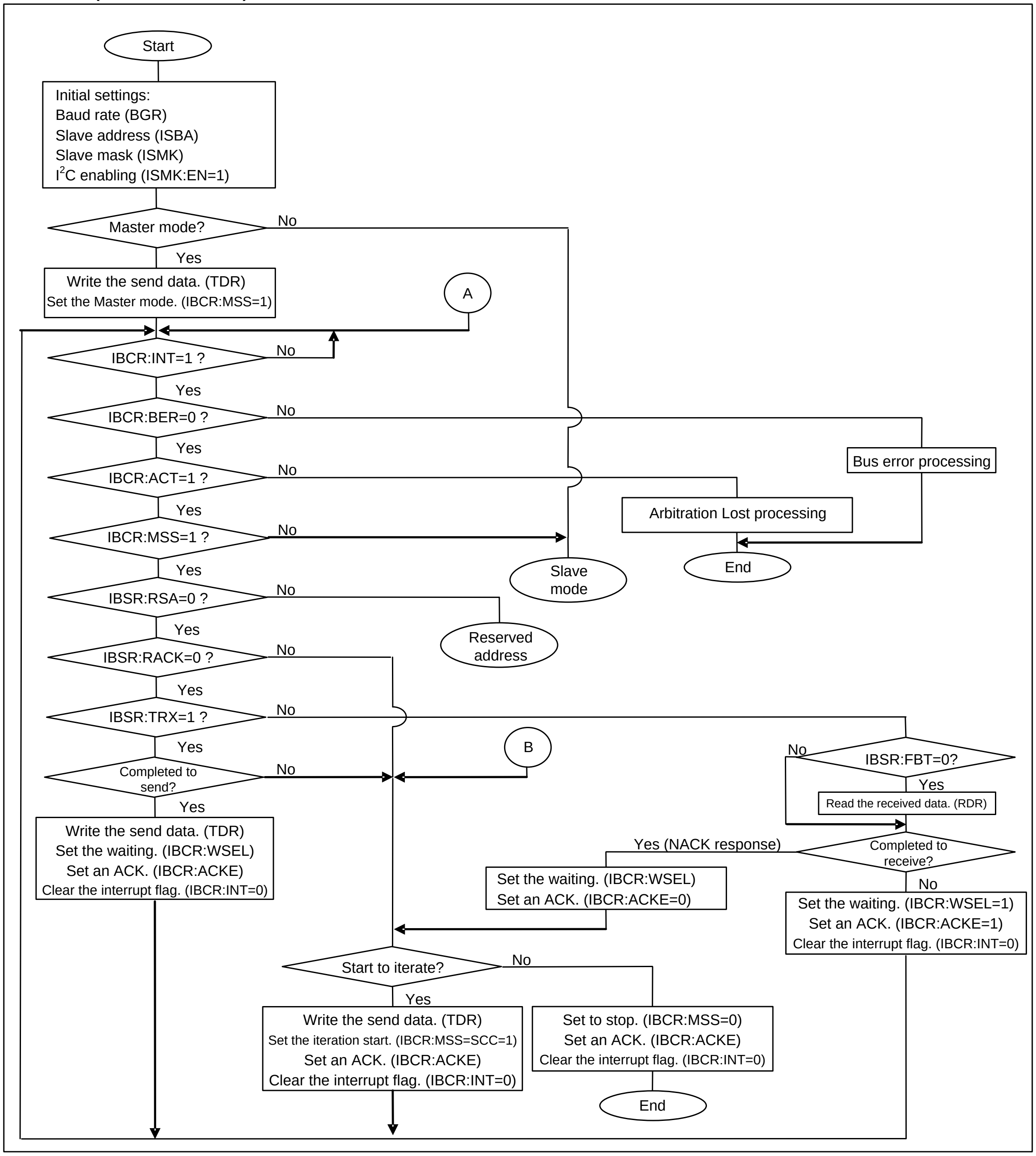


Figure 4-2 I²C flowchart example (FIFO not used) when DMA mode is disabled (SSR:DMA=0) 2/3

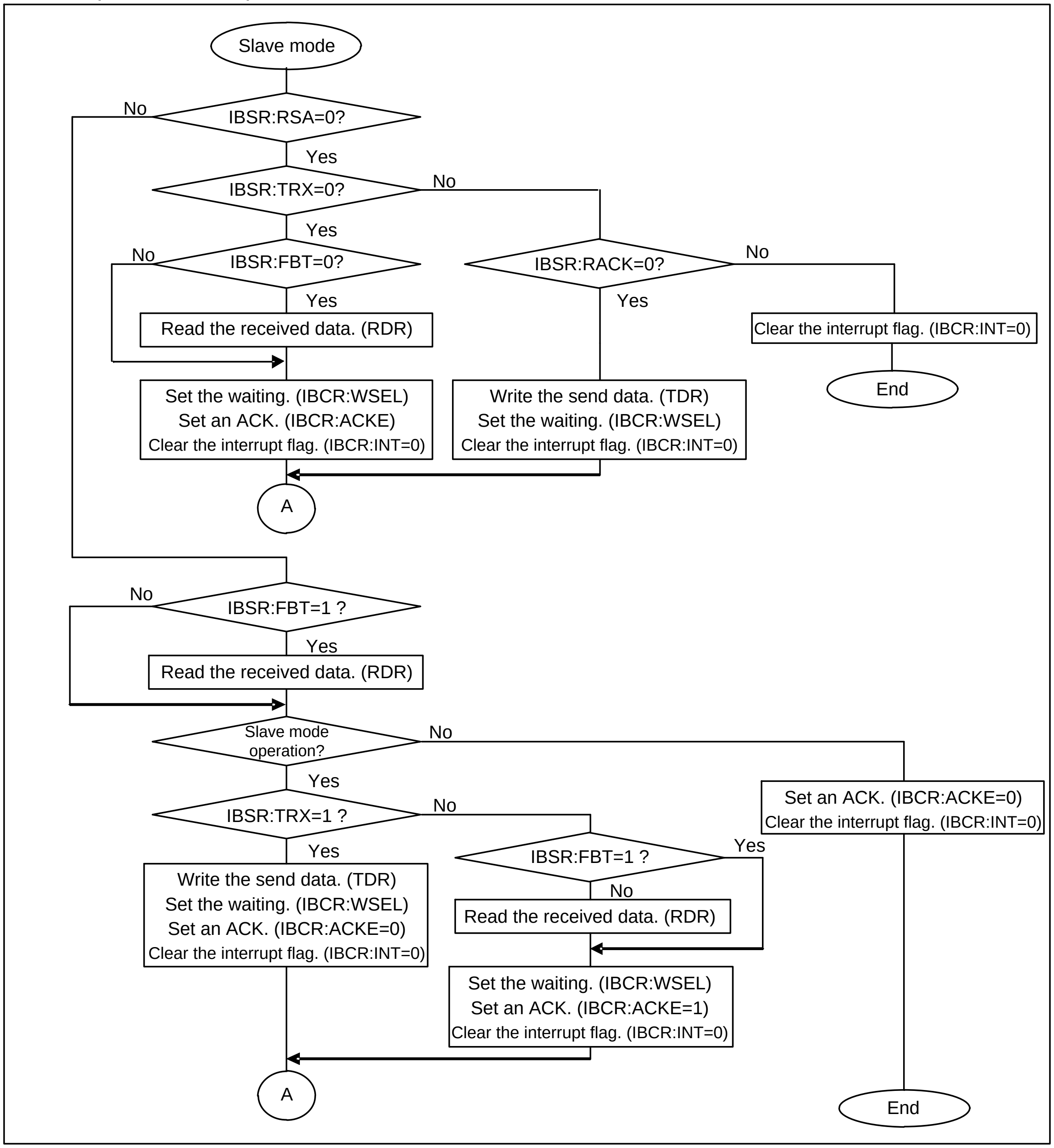
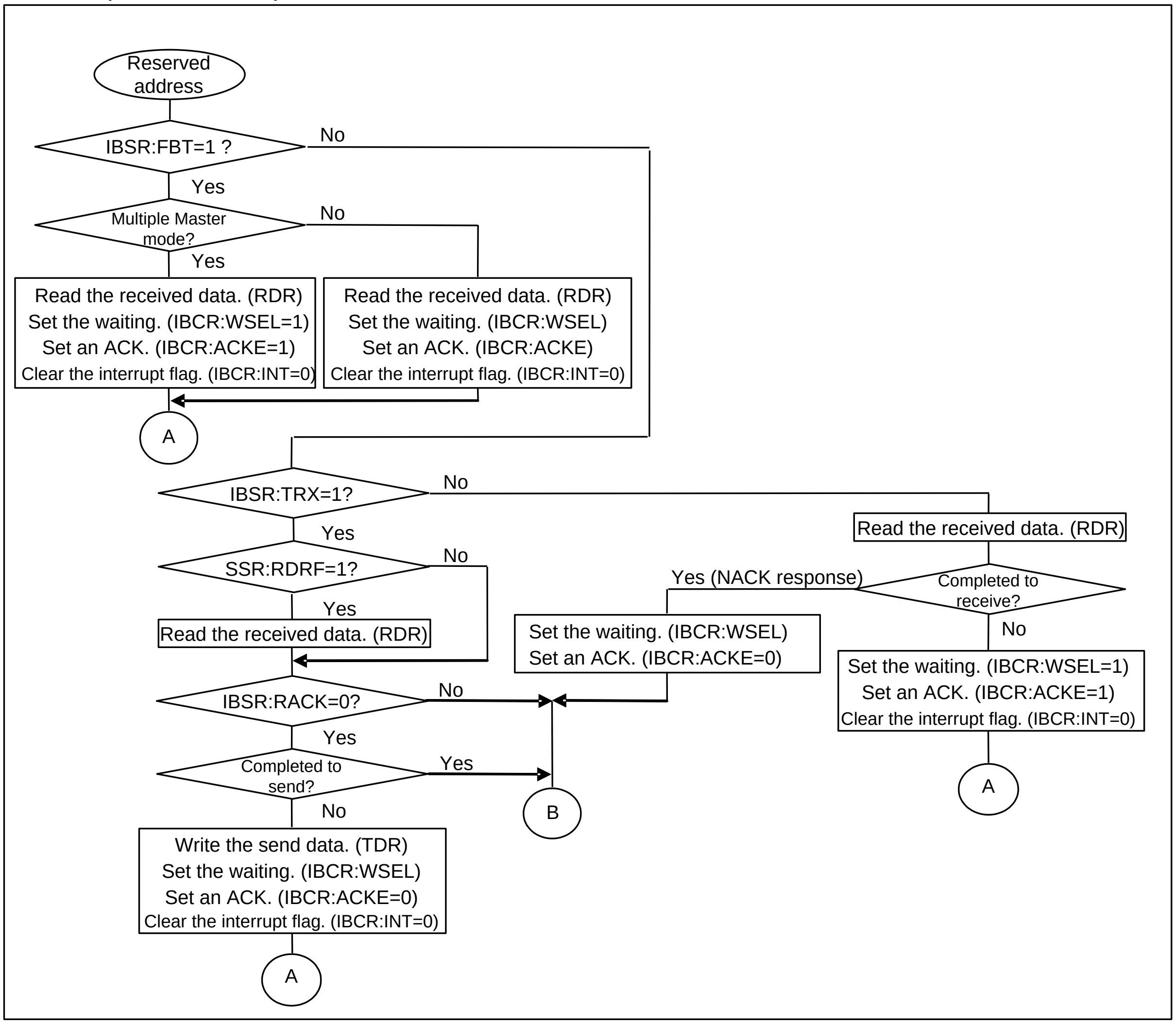


Figure 4-3 I^{2}C flowchart example (FIFO not used) when DMA mode is disabled (SSR:DMA=0) 3/3



■ I²C flowchart examples (FIFO not used) when DMA mode is enabled (SSR:DMA=1)

Figure 4-4 I²C flowchart example (FIFO not used) when DMA mode is enabled (SSR:DMA=1) 1/4

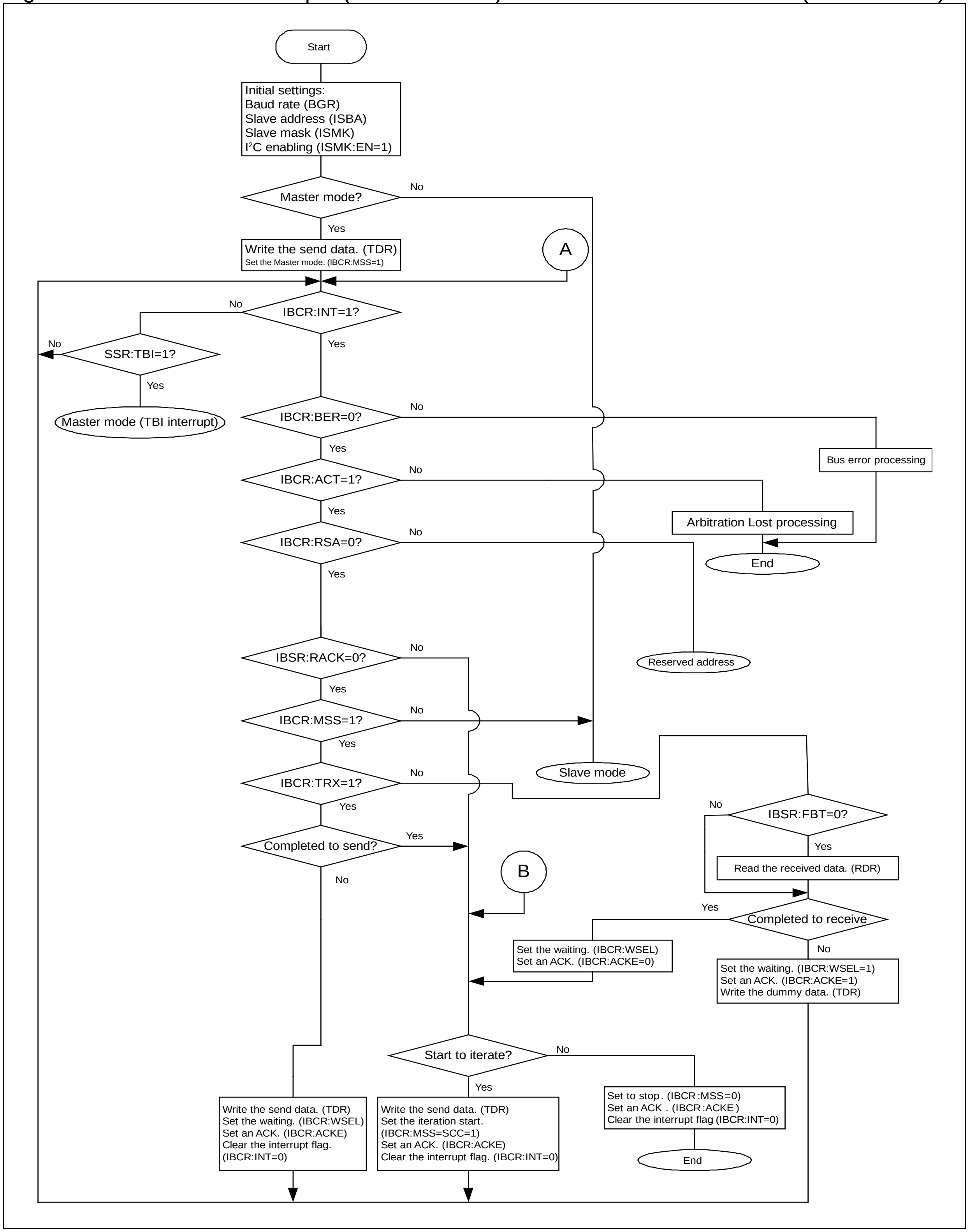


Figure 4-5 I²C flowchart example (FIFO not used) when DMA mode is enabled (SSR:DMA=1) 2/4

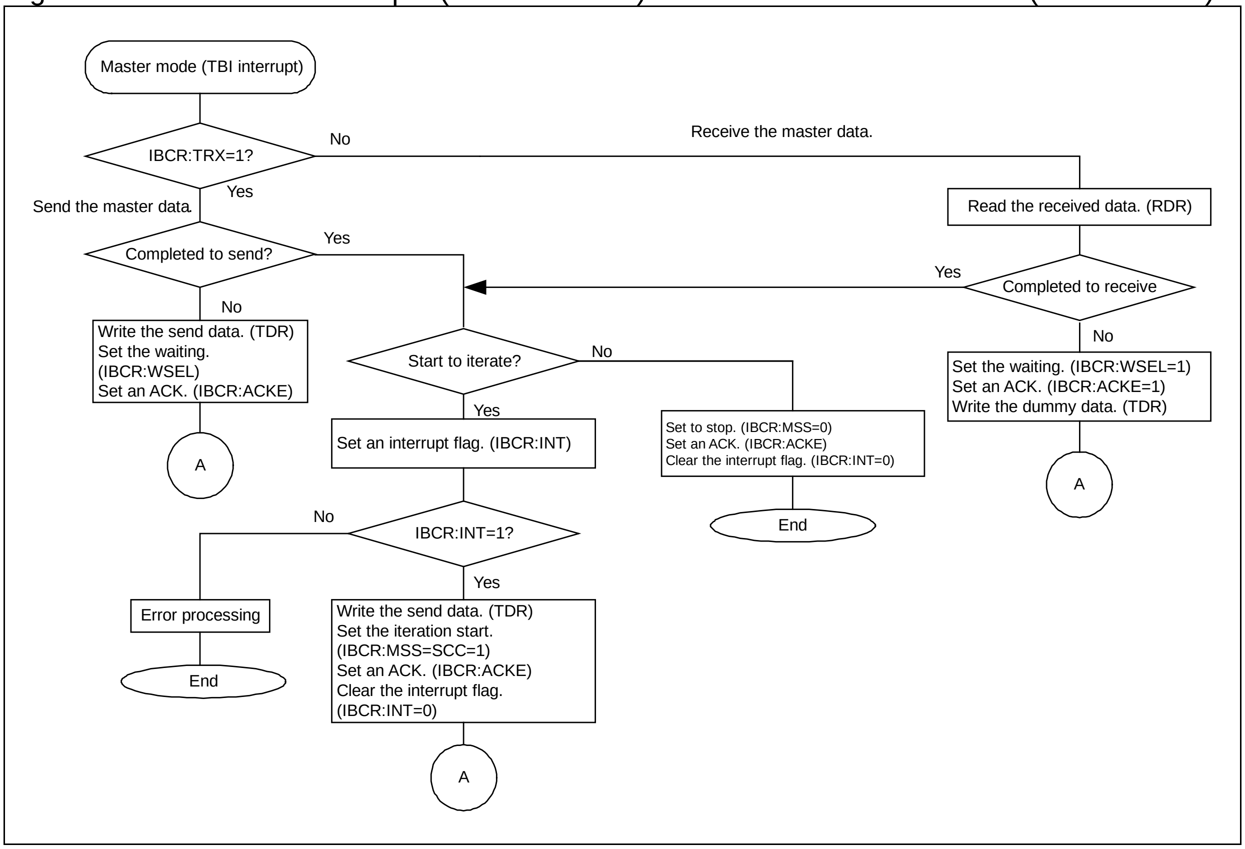


Figure 4-6 I²C flowchart example (FIFO not used) when DMA mode is enabled (SSR:DMA=1) 3/4

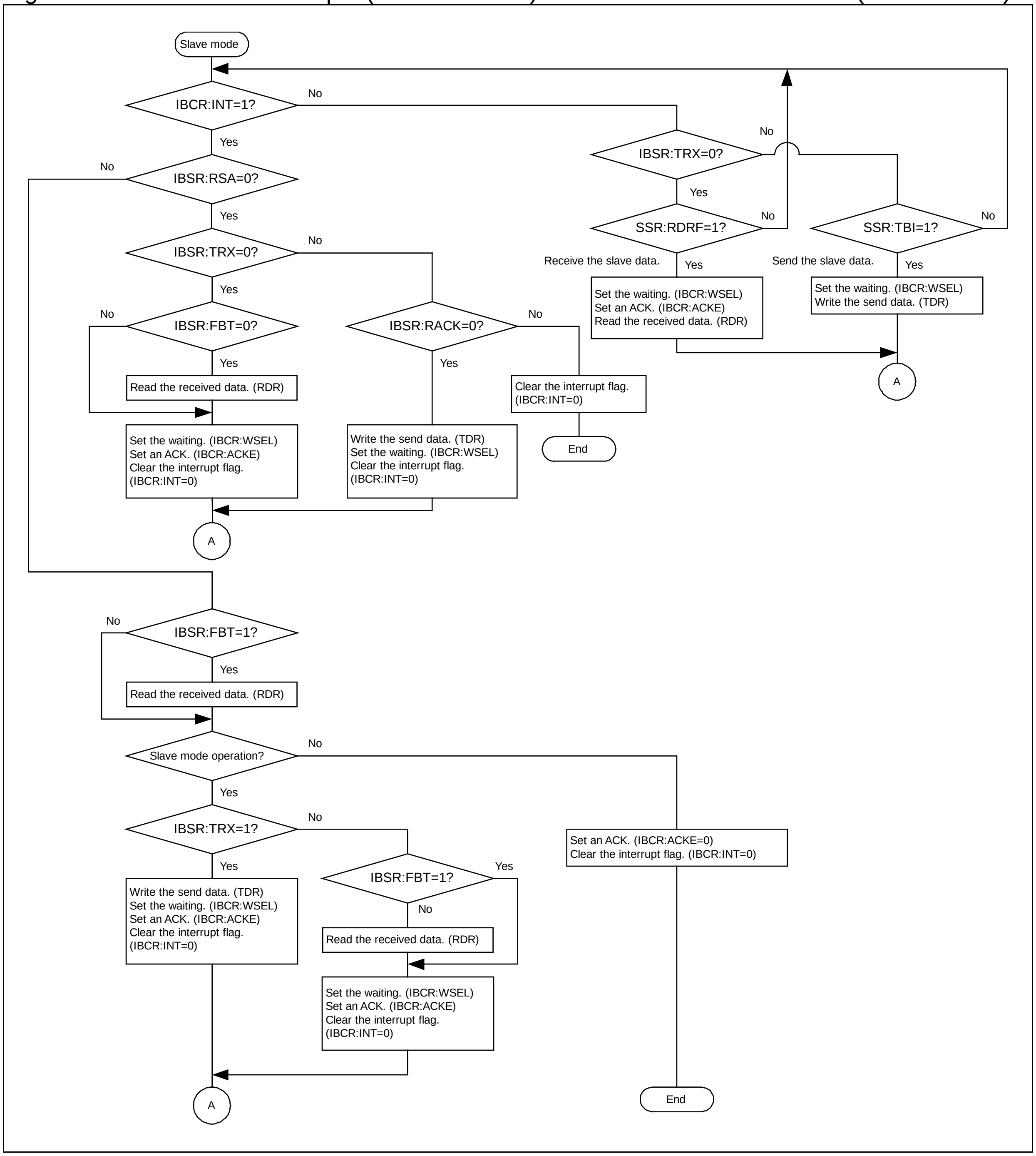
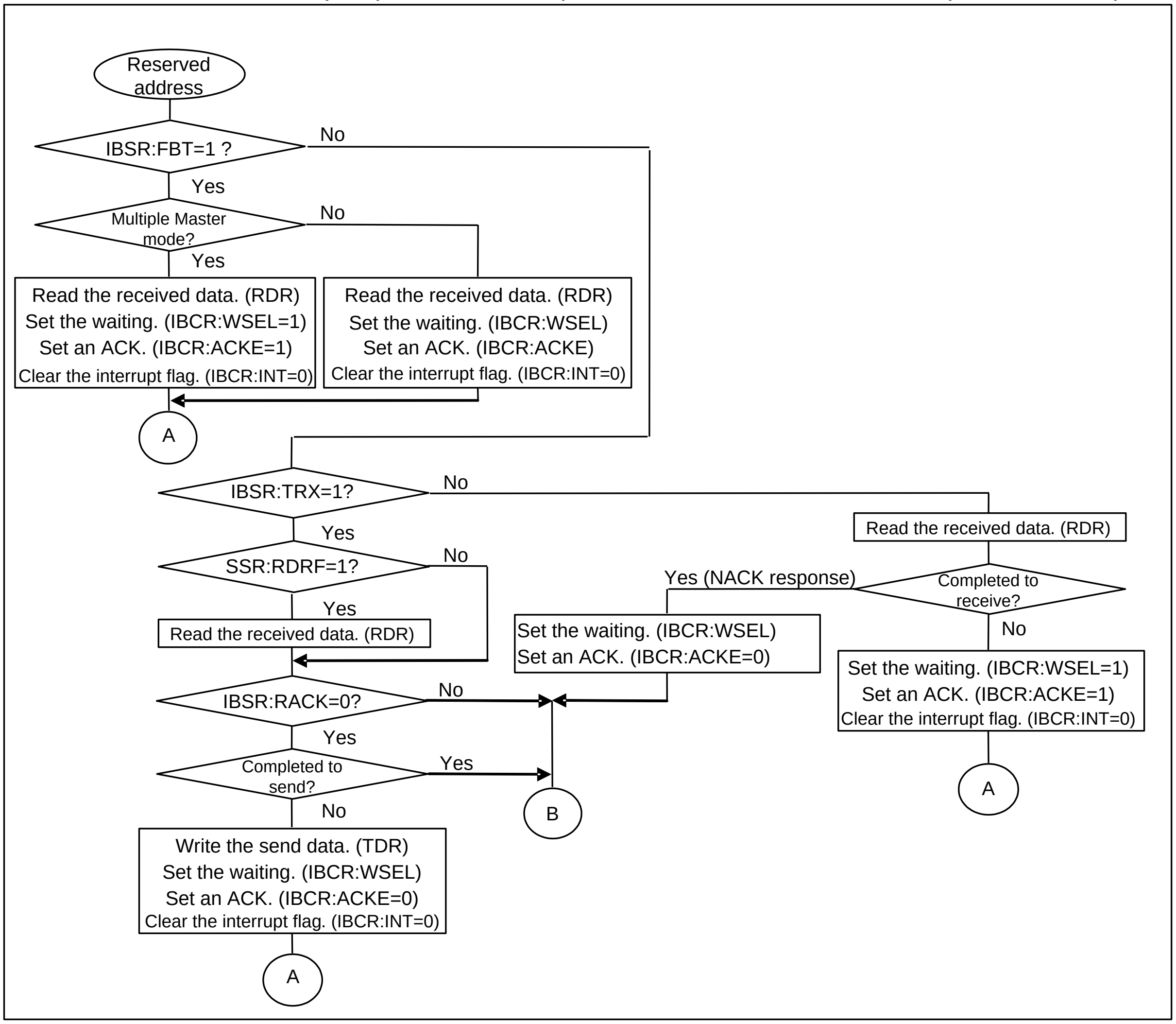


Figure 4-7 I²C flowchart example (FIFO not used) when DMA mode is enabled (SSR:DMA=1) 4/4



<Note>

The flow shows an outline of operation settings in I²C mode. To perform the appropriate operations, take into account error processing based on applications.

5. I^{2}C Interface Registers

The following lists the I^{2}C interface registers.

■ List of I^{2}C interface registers

Table 5-1 List of I^{2}C interface registers

	bit15 bit8	bit7 bit0
I^{2}C	IBCR (I^{2}C Bus Control Register)	SMR (Serial Mode Register)
	SSR (Serial Status Register)	IBSR (I^{2}C Bus Status Register)
	-	RDR/TDR (Transmit/Received Data Register)
	EIBCR (Extension I^{2}C Bus control Register)	-
	BGR1 (Baud Rate Generator Register 1)	BGR0 (Baud Rate Generator Register 0)
	ISMK (7-bit Slave Address Mask Register)	ISBA (7-bit Slave Address Register)
FIFO	FCR1 (FIFO Control Register 1)	FCR0 (FIFO Control Register 0)
	FBYTE2 (FIFO2 Byte Register)	FBYTE1 (FIFO1 Byte Register)

Table 5-2 I^{2}C Interface bit assignment

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
IBCR/ SMR	MSS	ACT/ SCC	ACKE	WSEL	CNDE	INTE	BER	INT	MD2	MD1	MD0	-	RIE	TIE	-	-
SSR/ IBSR	REC	TSET	DMA	TBIE	ORE	RDRF	TDRE	TBI	FBT	RACK	RSA	TRX	AL	RSC	SPC	BB
TDR1/ TDR0	-	-	-	-	-	-	-	-	D7	D6	D5	D4	D3	D2	D1	D0
EIBCR/ -	-	-	SDAS	SCLS	SDAC	SCLC	SOCE	BEC	-	-	-	-	-	-	-	-
BGR1/ BGR0	-	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
ISMK/ ISBA	EN	SM6	SM5	SM4	SM3	SM2	SM1	SM0	SAEN	SA6	SA5	SA4	SA3	SA2	SA1	SA0
FCR1/ FCR0	-	-	-	FLSTE	FRIIE	FDRQ	FTIE	FSEL	-	FLST	FLD	FSET	FCL2	FCL1	FE2	FE1
FBYTE2/ FBYTE1	FD15	FD14	FD13	FD12	FD11	FD10	FD9	FD8	FD7	FD6	FD5	FD4	FD3	FD2	FD1	FD0

5.1. I^{2}C Bus Control Register (IBCR)

The I^{2}C Bus Control Register (IBCR) is used to select master or slave mode, generate an iteration start condition, enable an acknowledgement, enable an interrupt, and display an interrupt flag.

bit	15	14	13	12	11	10	9	8	7 ... 0
Field	MSS	ACT/ SCC	ACKE	WSEL	CNDE	INTE	BER	INT	(SMR)
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial value	0	0	0	0	0	0	0	0	

[bit15] MSS: Master/slave select bit

- If this bit is set to "1" when the I^{2}C bus is in idle state (ISMK:EN=1, IBSR:BB=0), master mode is selected.
- If this bit is set to "1" when the BB bit of IBSR register is "1", the occurrence of start condition is waited until the IBSR:BB bit is set to "0". If the slave address matches and the slave operation is started during waiting, this bit is set to "0" and the AL bit of IBSR register is set to "1".
- When master mode is selected (MSS=1, ACT=1) and the interrupt flag (INT) is "1", a stop condition is generated when this bit is set to "0".

The MSS bit is cleared in any of the following conditions.

1. When the I^{2}C interface operation is disabled (ISMK:EN=0)
2. When an arbitration lost occurs
3. When a bus error is detected (BER=1) and when EIBCR:BEC=0.
4. When the MSS bit is set to "0" if INT=1
5. When DMA mode is enabled (SSR:DMA=1), SSR:TBI=1, and when the MSS bit is set to "0"

The following provides the relation between MSS and ACT bits.

MSS bit	ACT bit	State
0	0	Idle
0	1	The slave address matching or ACK is responded to the reserved address (*1), and slave mode is in operation (in slave mode).
1	0	The master mode operation is waited.
1	1	During master mode operation (in master mode)

*1) ACK response: The SDA is LOW on the I^{2}C bus during acknowledgement.

Value	Description
0	Selects slave mode.
1	Selects master mode.

<Notes>

- If DMA mode is disabled (SSR:DMA=0) and the MSS bit is set to "1", the MSS bit must be set to "0" only when the MSS bit is "1" and the INT bit is "1". If the MSS bit is set to "0" when the ACT bit is "1", the INT bit is also cleared to "0".
- If DMA mode is enabled (SSR:DMA=1) and the MSS bit is set to "1", the MSS bit must be set to "0" only when the MSS bit is "1" and the INT bit is "1", or the SSR:TBI bit is "1". If the MSS bit is set to "0" when the ACT bit is "1", the INT bit is also cleared to "0".
- When master mode is selected, the MSS bit is read to be "1" even when it is set to "0" while the ACT bit is "1".

[bit14] ACT/SCC : Operation flag/iteration start condition generation bit

This bit setting has a different meaning when it is written and read.

Reading	Writing
ACT bit	SCC bit

The ACT bit indicates the current operation in master or slave mode.

The ACT bit is set when:

1. The start condition is output onto the I^2C bus (master mode)
2. The slave address matches the address sent from the master device (slave mode)
3. The reserved address is detected and it is acknowledged (If MSS is "0", slave mode is selected.)

The ACT bit is reset when:

<Master mode>

1. The stop condition is detected.
2. An arbitration lost is detected.
3. When a bus error is detected and when EIBCR:BEC=0.
4. The I^2C interface operation is disabled (ISMK:EN=0)

<Slave mode>

1. The (iteration) start condition is detected
2. The stop condition is detected.
3. The reserved address is detected (IBSR:RSA=1) but not acknowledged
4. The I^2C interface operation is disabled (ISMK:EN=0)
5. When a bus error is detected (BER=1) and when EIBCR:BEC=0.

If this bit is set to "1" in master mode, the iteration start is executed. This bit is disabled to set to "0".

Value	Description	
	At writing	At reading
0	No effect	No operation
1	Generates an iteration start condition.	During the I^2C operation

<Notes>

- The SCC bit must be set to "1" during an interrupt of master mode (when MSS=1, ACT=1 and INT=1) only. If the SCC bit is set to "1" when the ACT bit is "1", the INT bit is cleared to "0".
- This bit must not be set to "1" in slave mode (when MSS=0 and ACT=1).
- If the SCC bit is set to "1" and if the MSS bit is set to "0" simultaneously, the MSS bit setting is preceded.
- When data is read by a read-modify-write instruction, the SCC bit is read.
- If both of the following conditions are satisfied, the INT bit is set to "1" and the I^2C bus is waited (SCL=LOW). To generate an iteration start condition, clear the INT bit by setting the SCC bit to "1" again.
  - The SCC bit is set to "1" during master mode interrupt at 8th bit (MSS=1, ACT=1, INT=1 and WSEL=1).
  - A negative acknowledgement (NACK) is received at 9th bit.
- When DMA mode is enabled (SSR:DMA=1), the SSR:TBI bit is "1" and the IBCR:INT bit is "0", follow the steps below to issue the iteration start condition.
  1. Set the IBCR:INT bit to "1".
  2. Check that the IBCR:INT bit is set to "1".
  3. Write the slave address in the TDR.
  4. Set this bit to "1".

[bit13] ACKE: Data byte acknowledge enable bit

- If this bit is set to "1", LOW is output when acknowledged.
- This bit must be changed if any of the following conditions has occurred:
  - If DMA mode is disabled (SSR:DMA=0), the ACT bit is "1", and the INT bit is "1"
  - If DMA mode is enabled (SSR:DMA=1), the ACT bit is "1", and the SSR:TBI bit is "1"
  - If DMA mode is enabled (SSR:DMA=1), the ACT bit is "1", the slave mode reception is selected, and the SCR:RDRF is "1"
  - If the ACT bit is "0"

This bit is invalid in the following conditions.

1. During acknowledgement to an address field other than the reserved address (automatic generation)
2. During data transmission (IBSR:RSA=0, IBSR:TRX=1, IBSR:FBT=0)
3. If the received FIFO is enabled and the slave mode reception is selected (FCR0:FE=1, MSS=0, ACT=1), an ACK is returned.
4. If the received FIFO is enabled, the WSEL bit is "0", the master mode reception is selected (FCR0:FE=1, MSS=1, ACT=1, WSEL=0), and the SSR:TDRE bit is "0", an ACK is always returned. If the SSR:TDRE bit is "1", a NACK is returned.
5. If the received FIFO is enabled, WSEL=0, the reserved address is detected and the slave transmission is selected (IBSR:RSA=1, IBSR:TRX=1, IBSR:FBT=1), an ACK is always returned. To respond with a NACK, disable the received FIFO and set the ACKE bit to "0" during interrupt after detection of the reserved address.
6. The received FIFO is enabled, the WSEL bit is "1", the master mode reception is selected, and the Transmit Data Register has data (FCR0:FE=1, MSS=1, ACT=1, WSEL=1, SSR:TDRE=0).

Value	Description
0	Disables acknowledgment.
1	Enables acknowledgement.

[bit12] WSEL: Wait selection bit

- If DMA mode is disabled (SSR:DMA=0), this bit selects a generation time of interrupt before or after acknowledgement (INT=1) and selects to wait the I^2C bus or not.
- If DMA mode is enabled (SSR:DMA=1), this bit selects a generation time of interrupt before or after acknowledgement (INT=1, and SSR:TBI=1 for transmission or SSR:RDRF=1 for reception) and selects to wait the I^2C bus or not.
- The WSEL bit is invalid in the following conditions.

1. An interrupt occurs (INT=1) for the first byte. (*1)
2. The reserved address is detected (IBSR:FBT=1, IBSR:RSA=1).
3. The NACK response is detected during FIFO data transfer (FCR0:FE=1, IBSR:RACK=1, ACT=1). (*2)
4. The received FIFO is filled with data during FIFO reception.

*1) The first byte indicates data after the (iteration) start condition.

*2) NACK response: The SDA bit of I^2C bus is HIGH during acknowledgement.

Value	Description
0	Waits (9 bits) after acknowledgement.
1	Waits (8 bits) after data transmission or reception.

[bit11] CNDE: Condition detection interrupt enable bit

This bit enables an interrupt if a stop condition or an iteration start condition is detected in master or slave mode (ACT=1). An interrupt occurs if the RSC or SPC bit of IBSR register is "1" and if this bit is set to "1".

Value	Description
0	Disables an interrupt due to the iteration start or stop condition.
1	Enables an interrupt due to the iteration start or stop condition.

[bit10] INTE: Interrupt enable bit

This bit enables an interrupt (INT=1) due to a data transmission and reception or bus error in master or slave mode.

Value	Description
0	Disables an interrupt.
1	Enables an interrupt.

[bit9] BER: Bus error flag bit

This bit indicates that an error has been detected on the I^2C bus.

The BER bit is set when:

1. The start or stop condition is detected during transfer of the first byte. (*1)
2. The (iteration) start condition or the stop condition is detected at bit2 to bit9 (acknowledgement) of data after the 2nd or subsequent byte.

The BER bit is reset when:

1. The INT bit is set to "0" if EIBCR:BEC=0 and BER=1.
2. The I^2C interface operation is disabled (ISMK:EN=0).
3. The IBCR:INT bit is set to "0" when EIBCR:BEC=1 and IBCR:INT=1.
4. The IBCR:SPC bit is set to "0" when EIBCR:BEC=1 and IBCR:SPC=1.
5. The IBCR:RSC bit is set to "0" when EIBCR:BEC=1 and IBCR:RSC=1.

*1) The first byte indicates data after the (iteration) start condition.

Value	Description
0	No error
1	An error was detected.

<Note>

In the following cases, check this bit state if the interrupt flag (INT bit) is "1". If it is "1", the normal data transmission and reception fail. Retransmit the data.

· The interrupt flag(INT bit) is "1" when EIBCR:BEC=0
· The iteration start condition confirmation bit(IBSR:RSC bit) is "1" when EIBCR:BEC=1
· The stop condition confirmation bit(IBSR:SPC bit) is "1" when EIBCR:BEC=1

[bit8] INT: interrupt flag bit

The interrupt flag bit is set to "1" after 8 or 9 bits (ACK) of data have been transmitted and received or when a bus error has occurred in master or slave mode. During operation other than bus error, if the INT bit is set to "1", the SCL flag is set to LOW. If the INT bit is set to "0", the SCL is released from the LOW state.

The INT bit is set when:

<8th bit>

<If DMA mode is not related>

1. The reserved address is detected in the first byte.
2. The WSEL bit is "1" and an arbitration lost is detected in the 2nd or subsequent byte.

<If DMA mode is disabled (SSR:DMA=0)>

1. If DMA mode is disabled (SSR:DMA=0), WSEL bit is "1", master mode is selected, and the SSR:TDRE bit is "1" in the 2nd or subsequent byte.
2. If DMA mode is disabled (SSR:DMA=0), WSEL bit is "1", slave mode is selected, the received FIFO is disabled, and the SSR:TDRE bit is "1" in the 2nd or subsequent byte.
3. If DMA mode is disabled (SSR:DMA=0), WSEL bit is "1", the slave mode transmission is selected, and the SSR:TDRE bit is "1" in the 2nd or subsequent byte.

4. If DMA mode is disabled (SSR:DMA=0), WSEL bit is "1", the received FIFO is disabled, and the slave mode reception is selected.

<If DMA mode is enabled (SSR:DMA=1)>

1. If DMA mode is enabled (SSR:DMA=1), WSEL bit is "1", master mode is selected , the SSR:TBI bit is "1" in the 2nd or subsequent byte, and the INT bit is set to "1".

<9th bit>

<If DMA mode is not related>

1. An arbitration lost is detected in the first byte.
2. The NACK signal is received during the time other than stop condition output setting (the MSS bit is set to "0" during the master mode operation).
3. The WSEL bit is "0" and an arbitration lost is detected in the 2nd or subsequent byte.
4. The reserved address is not detected in the 1st byte, and data is found in the received FIFO when the received FIFO is enabled and data is received in master or slave mode (IBSR:TRX=0).
5. EIBCR:BEC=1 and IBSR:BER=1

<If DMA mode is disabled (SSR:DMA=0)>

1. If DMA mode is disabled (SSR:DMA=0), the reserved address is not detected in the 1st byte, and the SSR:TDRE bit is "1" when data is transmitted (IBSR:TRX=1) in master or slave mode.
2. If DMA mode is disabled (SSR:DMA=0), the reserved address is not detected in the 1st byte, and the SSR:TDRE bit is "1" when the received FIFO is disabled for data reception (IBSR:TRX=0) in master or slave mode.
3. If DMA mode is disabled (SSR:DMA=0), WSEL bit is "0", and the SSR:TDRE bit is "1" in the 2nd or subsequent byte during the master mode operation.
4. If DMA mode is disabled (SSR:DMA=0), WSEL bit is "0", and the SSR:TDRE bit is "1" in the 2nd or subsequent byte during the slave mode transmission.
5. If DMA mode is disabled (SSR:DMA=0), WSEL bit is "0", the received FIFO is disabled, and the slave mode reception is selected. However, if the reserved address is detected in the 1st byte during the slave mode reception, no interrupt is generated by bit 9.
6. If DMA mode is disabled (SSR:DMA=0), the received FIFO is enabled, data is received in slave mode, and the received FIFO is filled with data.

<If DMA mode is enabled (SSR:DMA=1)>

1. If DMA mode is enabled (SSR:DMA=1), the reserved address is not detected in the 1st byte, and the SSR:TDRE bit is "1" when data is transmitted (IBSR:TRX=1) in slave mode.
2. If DMA mode is enabled (SSR:DMA=1), the reserved address is not detected in the 1st byte, and the SSR:TDRE bit is "1" when the received FIFO is disabled for data reception (IBSR:TRX=0) in slave mode.
3. If DMA mode is enabled (SSR:DMA=1), WSEL bit is "0", the SSR:TBI bit is "1" in the 2nd or subsequent byte during the master mode operation, and the INT bit is set to "1".

<Others>

1. When a bus error is detected and EIBCR:BEC=0.

The INT bit is reset when:

1. The INT bit is set to "0".
2. The INT bit is "1" and the ACT bit is "1", the MSS bit is set to "0".
3. The INT bit is "1" and the ACT bit is "1", the SCC bit is set to "1".

If the DMA mode is disabled (SSR:DMA=0), it is invalid to set the INT bit to "1".

Value	Description	
	At writing	At reading
0	Clears the INT bit.	Does not issue an interrupt request.
1	No effect	Issues an interrupt request.

<Notes>

- When DMA mode is enabled (SSR:DMA=1) and the SSR:TBI bit is "1" in the 2nd or subsequent byte during the master mode operation, a status interrupt (SIRQ=1) is not generated even when the INT bit is set to "1".
- When DMA is enabled (SSR:DMA=1), the SSR:TBI bit is "1" and the IBCR:INT bit is "0", follow the steps below to issue the iteration start condition.
  1. Set the IBCR:INT bit to "1".
  2. Check that the IBCR:INT bit is set to "1".
  3. Write the slave address in the TDR.
  4. Set the IBCR:SCC bit to "1".
- If the INT flag is changed from "1" to "0", the I^2C bus is released from waiting.
- If the ISMK:EN bit is set to "0", the SSR:RDRF and INT bits may be set to "1" in certain received timing. If so, read the received data and clear the INT bit.
- When a read-modify-write instruction is issued, "1" is read.
- If the received FIFO is enabled, the INT bit is not set to "1" even when the received FIFO is filled with data during the master mode reception.
- Set this bit to "1" when the start condition is issued (IBCR:MSS=1).

5.2. Serial Mode Register (SMR)

The Serial Mode Register (SMR) is used to set an operation mode, and to enable or disable the transmit/received interrupt.

bit	15 ... 8	7	6	5	4	3	2	1 ... 0
Field	(SCR)	MD2	MD1	MD0	Reseved	RIE	TIE	Reserved
Attribute		R/W	R/W	R/W	-	R/W	R/W	-
Initial value		0	0	0	0	0	0	-

[bit7:5] MD2, MD1, MD0: operation mode set bits

These bits set an operation mode.

* This chapter explains the registers and their operation in operation mode 4 (I^{2}C mode).

bit7	bit6	bit5	Description
0	0	0	Operation mode 0 (async normal mode)
0	0	1	Operation mode 1 (async multiprocessor mode)
0	1	0	Operation mode 2 (clock sync mode)
0	1	1	Operation mode 3 (LIN communication mode)
1	0	0	Operation mode 4 (I^{2}C mode)
Values other than the above			Setting disabled.

<Notes>

- Any bit setting other than above is inhibited.
- To switch the current operation mode, disable the I^{2}C (ISMK:EN=0) and change the operation mode continuously.
- After the operation mode has been set, set each register correctly.

[bit4] Reserved: Reserved bit

The read value is "0". Be sure to write "0".

[bit3] RIE: Received interrupt enable bit

- This bit enables or disables an output of received interrupt request to the CPU.
- If the RIE bit and the received data flag bit (SSR:RDRF) are "1", or if any of error flag bits (SSR:ORE) is "1", a received interrupt request is output.

Value	Description
0	Disables the received interrupt.
1	Enables the received interrupt.

<Note>

To receive data using the INT bit of I^2C Bus Control Register (IBCR) when DMA mode is disabled (SSR:DMA=0), set this bit to "0".

[bit2] TIE: Transmit interrupt enable bit

- This bit enables or disables an output of transmit interrupt request to the CPU.
- If the TIE and SSR:TDRE bits are "1", a transmit interrupt request is output.

Value	Description
0	Disables the transmit interrupt.
1	Enables the transmit interrupt.

<Note>

To transmit data using the INT bit of I^2C Bus Control Register (IBCR) when DMA mode is disabled (SSR:DMA=0), set this bit to "0".

[bit1:0] Reserved: Reserved bits

The read value is "0". Be sure to write "0".

5.3. I^{2}C Bus Status Register (IBSR)

The I^{2}C Bus Status Register (IBSR) shows the iteration start, acknowledgement, data direction, arbitration lost, stop condition, I^{2}C bus status, and bus error detection.

bit	15 ... 8	7	6	5	4	3	2	1	0
Field	(SSR)	FBT	RACK	RSA	TRX	AL	RSC	SPC	BB
Attribute		R	R	R	R	R	R/W	R/W	R
Initial value		0	0	0	0	0	0	0	0

[bit7] FBT: First byte bit

This bit indicates the first byte.

The FBT bit is set when:

1. The (iteration) start condition is detected.

The FBT bit is cleared when:

1. The second byte is sent or received.
2. The stop condition is detected.
3. The I^{2}C interface operation is disabled (ISMK:EN=0).
4. When a bus error is detected (IBCR:BER=1) and EIBCR:BEC=0.

Value	Description
0	Other than 1st byte
1	The 1st byte is being sent or received.

[bit6] RACK: Acknowledge flag bit

This bit shows acknowledgement being received in the 1st byte or in master or slave mode.

The RACK bit is updated when:

1. Acknowledged in the 1st byte.
2. Data is acknowledged in master or slave mode.

The RACK bit is cleared (RACK=0) when:

1. The (iteration) start condition is detected.
2. The I^{2}C interface operation is disabled (ISMK:EN=0).
3. When a bus error is detected (IBCR:BER=1) and EIBCR:BEC=0.

bit	Description
0	LOW is received.
1	HIGH is received.

[bit5] RSA: Reserved address detection bit

This bit shows that the reserved address has been detected.

The RSA bit is set (RSA=1) when:

1. The 1st byte is "0000xxxx" or "1111xxxx". Where "x" can be "0" or "1".

The RSA bit is reset (RSA=0) when:

1. The (iteration) start condition is detected.
2. The stop condition is detected.
3. The I^2C interface operation is disabled (ISMK:EN=0).
4. When a bus error is detected (IBCR:BER=1) and EIBCR:BEC=0.

If the RSA bit is set to "1" in the 1st byte, the interrupt flag (IBCR:INT) is set to "1" and the SCL flag is set to "L" at the falling edge of SCL (8th bit) of the 1st byte regardless of FIFO enable or disable state. To read the received data and start the slave mode operation during this time, set the IBCR:ACKE bit to "1" and clear the interrupt flag (IBCR:INT) to "0". If the TRX bit is "0" after that, data is received in slave mode. To stop the data reception, set the IBCR:ACKE bit to "0". No data is received after that.

Value	Description
0	The reserved address is not detected.
1	The reserved address is detected.

<Notes>

- If the IBCR:ACKE bit is set to "0" during data transfer, this IBCR:ACKE bit cannot be set to "1" until the stop condition or the iteration start condition is detected.
- If the slave mode transmission is detected during an interrupt by reserved address detection and if the received FIFO is enabled, an ACK response is returned. In this case, disable the received FIFO and set the IBCR:ACKE bit to "0".

[bit4] TRX: Data direction bit

This bit indicates the data direction.

The TRX bit is set when:

1. The (iteration) start condition is sent in master mode.
2. 8th bit of the 1st byte is "1" in slave mode (in the slave mode transmission direction).

The TRX bit is reset when:

1. An arbitration lost occurs (AL=1).
2. 8th bit of the 1st byte is "0" in slave mode (in the slave mode reception direction).
3. 8th bit of the 1st byte is "1" in master mode (in the master mode reception direction).
4. The stop condition is detected.
5. The (iteration) start condition is detected in any mode other than master mode.
6. The I^2C interface operation is disabled (ISMK:EN=0).
7. When a bus error is detected (IBCR:BER=1) and EIBCR:BEC=0.

Value	Description
0	Received direction
1	Transmission direction

[bit3] AL: Arbitration lost bit

This bit indicates an arbitration lost.

The AL bit is set when:

1. The output data does not match the received data in master mode.
2. The IBCR:MSS bit is set to "1" but the slave mode operation is selected.
3. The iteration start condition is detected by 1st bit of the 2nd or subsequent byte data in master mode when EIBCR:BEC=0.
4. The iteration start condition is detected in master mode and when EIBCR:BEC=0.
5. The stop condition is detected by 1st bit of the 2nd or subsequent byte data in master mode when EIBCR:BEC=1.
6. The stop condition is detected in master mode when EIBCR:BEC=1 (except the case where the stop condition is detected in the acknowledge field.)
7. The iteration start condition cannot be generated in master mode.
8. The stop condition cannot be generated in master mode.

The AL bit is reset when:

1. The IBCR:MSS bit is set to "1".
2. The IBCR:INT bit is set to "0".
3. The SPC bit is set to "0" when both AL and SPC bits are "1".
4. The I^2C interface operation is disabled (ISMK:EN=0).
5. When a bus error is detected (IBCR:BER=1) and EIBCR:BEC=0.

Value	Description
0	No arbitration lost has occurred.
1	An arbitration lost has occurred.

[bit2] RSC: Iteration start condition check bit

This bit shows that an iteration start condition is detected in master or slave mode.

The RSC bit is set when:

1. When an iteration start condition is detected after acknowledgement, during the master or slave mode operation when EIBCR:BEC=0.
2. When an iteration start condition is detected in the first byte, during the master or slave mode, in the first bit when EIBCR:BEC=1.

The RSC bit is reset when:

1. The RSC bit is set to "0".
2. The IBCR:MSS bit is set to "1".
3. The I^2C interface operation is disabled (ISMK:EN=0).

It is invalid to set this bit to "1".

Value	Description
0	No iteration start condition has been detected.
1	An iteration start condition has been detected.

<Notes>

- If no acknowledgement response is sent while data is received in slave mode due to the reserved address being detected, slave mode is released. In this case, this bit is not set to "1" even if the next iteration start condition is detected.
- When a read-modify-write instruction is issued, "1" is read.

[bit1] SPC: Stop condition check bit

This bit shows that a stop condition is detected in master or slave mode.

The SPC bit is set when:

1. When the stop condition is detected in the master or slave mode operation, when EIBCR:BEC=0.
2. The stop condition is detected in the one of the following cases when EIBCR:BEC=1.
   - In the first byte when IBCR:ACT=0
   - In the slave operation mode
   - In the master mode(except the case where the stop condition is detected in the acknowledge field)
3. In master mode, the stop condition has occurred and, therefore, an arbitration lost has occurred.

The SPC bit is reset when:

1. This bit is set to "0".
2. The IBCR:MSS bit is set to "1".
3. The I^2C interface operation is disabled (ISMK:EN=0).

It is invalid to set this bit to "1".

Value	Description	
0	No stop condition is detected.	
1	Master mode	An arbitration lost has occurred when the stop condition is detected or when it is output.
	Slave mode	The stop condition is detected.

<Notes>

- If no acknowledgement response is sent while data is received in slave mode due to the reserved address being detected, slave mode is released. In this case, this bit is not set to "1" even if the next stop condition is detected.
- When a read-modify-write instruction is issued, "1" is read.
- When all the following conditions are met, this bit is not set to"1" and the master operation is continued even if the stop condition is detected:
  - When EIBCR:BEC=1
  - In the master operation
  - In the acknowledge field

[bit0] BB: Bus state bit

This bit shows the bus state.

The BB bit is set when:

1. LOW is detected in SDA or SCL of the I^{2}C bus.

The BB bit is reset when:

1. The stop condition is detected.
2. The I^{2}C interface operation is disabled (ISMK:EN=0).
3. When a bus error is detected (IBCR:BER=1) and EIBCR:BEC=0.

Value	Description
0	The bus is in idle state.
1	The bus is in transmission and reception state.

5.4. Serial Status Register (SSR)

The Serial Status Register (SSR) is used to check the transmission or reception state.

bit	15	14	13	12	11	10	9	8	7 ... 0
Field	REC	TSET	DMA	TBIE	ORE	RDRF	TDRE	TBI	(IBSR)
Attribute	R/W	R/W	R/W	R/W	R	R	R	R	
Initial value	0	0	0	0	0	0	1	1	

[bit15] REC: Received error flag clear bit

This bit clears the ORE bit of Serial Status Register (SSR).

- If this bit is set to "1", the ORE bit is cleared.
- This bit has no effect on the operation if set to "0".

When it is read, "0" is always read.

Value	Description	
	At writing	At reading
0	No effect on operation.	"0" is always read.
1	Clears the Received Error flag (ORE).	

[bit14] TSET: Transmit empty flag set bit

This bit sets the TDRE bit of Serial Status Register (SSR).

- If it is set to "1" and if the TDRE bit and DMA mode are enabled (DMA=1), the TBI bit is set.
- This bit has no effect on the operation if set to "0".

When it is read, "0" is always read.

Value	Description	
	At writing	At reading
0	No effect on operation.	"0" is always read.
1	The TDRE bit is set.	

<Note>

Set this bit to "1" only when the IBCR:INT bit is "1".

[bit13] DMA: DMA mode enable bit

This bit enables or disables DMA mode.

- If this bit is set to "1", an interrupt condition is generated during DMA transfer.
- If this bit is set to "0", an interrupt condition is generated during normal data transfer.

For details, see Table 2-1.

Value	Description
0	Disables DMA mode.
1	Enables DMA mode.

<Note>

This bit state can be changed only when the ISMK:EN bit is "0".

[bit12] TBIE: Transmit bus idle interrupt enable bit (Effective only when DMA mode is enabled)

- This bit enables or disables an output of transmit bus idle interrupt request to the CPU.
- If DMA mode is enabled (DMA=1) and both TBIE and TBI bits are "1", a transmit bus idle interrupt request is output.
- If DMA mode is disabled (DMA=0), this bit is set to "0". If data is written, this writing is ignored and the "0" is maintained.

Value	Description
0	Disables the transmit bus idle interrupt.
1	Enables the transmit bus idle interrupt.

[bit11] ORE: Overrun error flag bit

- If an overrun occurs during data reception, this bit is set to "1". This is cleared if the REC bit of Serial Status Register (SSR) is set to "1".
- If the ORE and SMR:RIE bits are "1", a received interrupt request is output.
- If this flag is set, the Received Data Register (RDR) is invalid.
- If the received FIFO is used and if this flag is set, the received data is not stored in the received FIFO.

Value	Description
0	No overrun error occurred.
1	An overrun error occurred.

[bit10] RDRF: Received data full flag bit

- This flag shows the state of Received Data Register (RDR).
- If the SMR:RIE bit and the received data flag bit (RDRF) are "1", a received interrupt request is issued.
- When the received data is loaded in the RDR, this bit is set to "1". When data is read from the Received Data Register (RDR), this bit is cleared to "0".
- This bit is set at the falling edge of SCL signal (8th bit of data).
- This bit is also set even when a NACK is responded. (*1)
- If the received FIFO is used and if a certain count of data is received by the received FIFO, the RDRF bit is set to "1".
- If the received FIFO is used and if received FIFO is emptied, this bit is cleared to "0".
- If all of the following conditions are satisfied and if the received idle state continues for more than 8 baud rate clocks, the interrupt flag (SSR:RDRF) is set to "1".
  - The received FIFO idle detection enable bit (FCR:FRIIE) is "1".
  - The number of data sets stored in the received FIFO does not reach the transfer count.
  - The IBCR:BER bit is "0".

If the RDR data is read during counting of 8 clocks, this counter is reset to 0 and counting for 8 clocks is restarted.

*1) NACK response: The SDA bit of I^2C bus is "H" during acknowledgement.

Value	Description
0	The Received Data Register (RDR) is empty.
1	The Received Data Register (RDR) contains data.

<Notes>

- If all of the following conditions are satisfied, the SCL flag is set to LOW after ACK is transmitted was transmitted. If the RDRF bit is set to "0", the SCL flag is released from the LOW state.
  - The received FIFO is not used.
  - DMA mode is enabled (SSR:DMA=1).
  - Data is received in the 2nd or subsequent byte (IBSR:TRX=0), and the RDRF bit is "1".
  - The IBCR:WSEL bit is "0".
- If all of the following conditions are satisfied, the SCL flag is set to LOW immediately after single-byte data reception. If the RDRF bit is set to "0", the SCL flag is released from the LOW state.
  - The received FIFO is not used.
  - DMA mode is enabled (SSR:DMA=1).
  - Data is received in the 2nd or subsequent byte (IBSR:TRX=0), and the RDRF bit is "1".
  - The IBCR:WSEL bit is "1".
- If the received FIFO is used and DMA mode is enabled for data reception (DMA=1), the SCL flag is set to LOW when the received FIFO is filled with data. If data is read from the RDR even once, the SCL flag is released from the LOW state.

[bit9] TDRE: Transmit data empty flag bit

- This flag shows the state of Transmit Data Register (TDR).
- If the SMR:TIE and TDRE bits are "1", a Transmit Interrupt Request is output.
- If transmit data is written in the TDR, this bit is set to "0" to indicate that the TDR contains valid data. When data is loaded to a shift register for transmission and its transmission is started, this bit is set to "1" to indicate that the TDR does not have the valid data.
- If the TSET bit of Serial Status Register (SSR) is set to "1", this flag is set. If an arbitration lost or a bus error is detected, use this flag to set the TDRE bit to "1".

Value	Description
0	The Transmit Data Register (TDR) contains data.
1	The Transmit Data Register is empty.

[bit8] TBI: Transmit bus idle flag bit (Effective only when DMA mode is enabled)

This bit shows that no data is sent by the I^2C when DMA mode is enabled (DMA=1). If DMA mode is enabled (DMA=1) and the TBI bit is set to "1" in the 2nd or subsequent byte, the SCL flag is set to LOW. If the TBI bit is set to "0", the SCL flag is cleared from the LOW state.

The TBI bit is set when:

<8th bit>

1. The WSEL bit is "1", master mode is selected, and the TDRE bit is "1" in the 2nd or subsequent byte.
2. The WSEL bit is "1", the slave mode transmission is selected, and the SSR:TDRE bit is "1" in the 2nd or subsequent byte.

<9th bit>

1. Master mode is selected, the reserved address is not detected in the 1st byte, and the SSR:TDRE bit is "1".
2. The WSEL bit is "0", master mode is selected, and the TDRE bit is "1" in the 2nd or subsequent byte.
3. The WSEL bit is "0", the slave mode transmission is selected, and the SSR:TDRE bit is "1" in the 2nd or subsequent byte.

<Others>

The transmit buffer empty flag set bit (TSET) is set to "1".

The TBI bit is reset when:

1. The transmit data is written in the Transmit Data Register (TDR).

If this bit is "1" and if the transmit bus idle interrupt is enabled (SCR:TBIE=1), a transmit interrupt request is output.

- If DMA mode is disabled (DMA=0), this bit is undefined.

Value	Description
0	During data transmission
1	No data transmission

5.5. Received Data Register/Transmit Data Register (RDR/TDR)

The Received and Transmit Data Registers are allocated at the same address. This register functions as the Received Data Register when data is read from it. This register functions as the Transmit Data Register when data is written in it.

■ Received Data Register (RDR)

bit	15 ... 8	7	6	5	4	3	2	1	0
Field		D7	D6	D5	D4	D3	D2	D1	D0
Attribute		R	R	R	R	R	R	R	R
Initial value		0	0	0	0	0	0	0	0

The Received Data Register (RDR) is a data buffer register for serial data reception.

- When a serial data signal is sent to the serial data line (SDA pin), it is converted by a shift register and stored in the Received Data Register (RDR).
- When the first byte (*1) is received, a received address is not stored in the Received Data Register (RDR). However, when the first byte is a reserved address, a received address is stored in the Received Data Register (RDR). In this case, the least significant bit (RDR:D0) is the data direction bit.
- When the received data is stored in the Received Data Register (RDR), the received data full flag bit (SSR:RDRF) is set to "1".
- When data is read from the Received Data Register (RDR), the received data full flag bit (SSR:RDRF) is cleared to "0" automatically.

  *1) The first byte indicates data after the (iteration) start condition.

<Notes>

- If the received FIFO is used and if a certain count of data is received by the received FIFO, the SSR:RDRF bit is set to "1".
- If the received FIFO is used and if received FIFO is emptied, the SSR:RDRF bit is cleared to "0".

■ Transmit Data Register (TDR)

bit	15	...	8	7	6	5	4	3	2	1	0
Field				D7	D6	D5	D4	D3	D2	D1	D0
Attribute				W	W	W	W	W	W	W	W
Initial value				1	1	1	1	1	1	1	1

The Transmit Data Register (TDR) is a data buffer register for serial data transmission.

- Data of the Transmit Data register (TDR) is output to the serial data line (SDA pin) with the MSB first order.
- When the first byte is transmitted, the least significant bit (TDR:D0) indicates the data direction.
- When the transmit data is written in the Transmit Data Register (TDR), the transmit data empty flag (SSR:TDRE) is cleared to "0".
- When data is transferred to a shift register for transmission, the transmit data empty flag (SSR:TDRE) is set to "1".
- If transmit FIFO is disabled and if the data empty flag (SSR:TDRE) is "0", the transmit data cannot be written in the Transmit Data Register (TDR).
- If transmit FIFO is used, the transmit data can be written until transmit FIFO is filled with it even if the transmit data empty flag (SSR:TDRE) is "0".

<Note>

The Transmit Data Register is a write-only register. While the Received Data Register is a read-only register. As these two registers are allocated at the same address, the write and read values differ from each other. Therefore, the INC/DEC instruction and other read-modify-write (RMW) operation cannot be used.

5.6. Extension I²C Bus Control Register (EIBCR)

The Extension I²C Bus Control Register (EIBCR) is used to control the output of SDA/SCL and set the operation continuity after a bus error occurs.
This register is not available for TYPE0 to TYPE5 products.

bit	15	14	13	12	11	10	9	8	7 ... 0
Field	Reserved		SDAS	SCLS	SDAC	SCLC	SOCE	BEC	-
Attribute	-		R	R	R/W	R/W	R/W	R/W	
Initial value	-		0	0	1	1	0	0	

[bit15:14] Reserved: Reserved bits

The read value is "0". Be sure to write "0".

[bit13] SDAS: SDA status bit

This bit indicates the signal level of SDA line after a noise filter.

Value	Description
0	SDA line is in "Low" level.
1	SDA line is in "High" level.

<Note>

This bit is valid only when I²C is enabled (ISMK:EN=1). When I²C is disabled (ISMK:EN=0), "0" is always read from this bit.

[bit12] SCLS: SCL status bit

This bit indicates the signal level of SCL line after a noise filter.

Value	Description
0	SCL line is in "Low" level.
1	SCL line is in "High" level.

<Note>

This bit is valid only when I²C is enabled (ISMK:EN=1). When I²C is disabled (ISMK:EN=0), "0" is always read from this bit.

[bit11] SDAC: SDA output control bit

When the serial output control is enabled (SOCE=1), this bit controls SDA output.

Value	Description
0	SDA output is in "Low" level.
1	SDA output is in "High" level.

[bit10] SCLC: SCL output control bit

When the serial output control is enabled (SOCE=1), this bit controls SCL output.

Value	Description
0	SCL output is in "Low" level.
1	SCL output is in "High" level.

[bit9] SOCE: Serial output enabled bit

This bit enables the serial output.

When this bit is set to "1", the following operations are executed:

- SDA output is controlled with SDA output control bit (SDAC).
- SCL output is controlled with SCL output control bit (SCLC)

Value	Description
0	Serial output control is disabled.
1	Serial output control is enabled.

<Note>

Only when IBCR:MSS=0 and IBCR:ACT=0, this bit must be set to "1".

[bit8] BEC: Bus error control bit

After a bus error occurs (IBSR:BER=1), this bit selects the continuity or abortion of I^2C operation.

Value	Description
0	I^2C operation is aborted.
1	I^2C operation is continued.

<Note>

When EIBCR:BEC=0, if the restart condition is detected while the address data is being transferred or bit2 to bit9(acknowledge bits) are being transferred after the start condition is detected, a bus error is detected(IBCR:BER=1) and reception is aborted. So, the next data is not received. In this case, after clearing the interrupt flag (IBCR:INT), the re-processing of the start condition from master is required.

5.7. 7-bit Slave Address Mask Register (ISMK)

The 7-bit Slave Address Mask Register (ISMK) is used to compare or set each bit of the slave address.

bit	15	14	13	12	11	10	9	8	7 ... 0
Field	EN	SM6	SM5	SM4	SM3	SM2	SM1	SM0	(ISBA)
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	1	1	1	1	1	1	1	

[bit15] EN: I^2C interface operation enable bit

This bit enables or disables the I^2C interface operation.

If set to "0": The I^2C interface operation is disabled.
If set to "1": The I^2C interface operation is enabled.

Value	Description
0	Disable
1	Enable

<Notes>

- This bit is not cleared to "0" even if the BER bit of IBSR register is set to "1".
- The baud rate generator must be set only when this bit is "0".
- When this bit is "0", set both the 7-bit Slave Address Register and the 7-bit Slave Address Mask Register.
- If the I^2C interface operation is disabled (EN=0), data transmission and reception is inhibited immediately.
- If you have set the IBCR:MSS bit to "0" to generate a Stop condition and if you wish to disable the I^2C interface operation, make sure that the stop condition has occurred. Then, disable the operation (EN=0).
- If the EN bit is set to "0" during data transmission, a pulse may be generated on the SDA/SCL signal of the I^2C bus.

[bit14:8] SM6 to SM0: Slave address mask bits

These bits specify to exclude the 7-bit slave address and the received address from comparison.

If set to "1", the address is compared.
If set to "0", the address matching is assumed.

Value	Description
0	Does not compare the bits.
1	Compares the bits.

<Note>

This register must be set only when the EN bit is "0".

5.8. 7-bit Slave Address Register (ISBA)

The 7-bit Slave Address Register (ISBA) is used to set the slave address.

bit	15 ... 8	7	6	5	4	3	2	1	0
Field	(ISMK)	SAEN	SA6	SA5	SA4	SA3	SA2	SA1	SA0
Attribute		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value		0	0	0	0	0	0	0	0

[bit7] SAEN: Slave address enable bit

This bit enables the slave address detection.

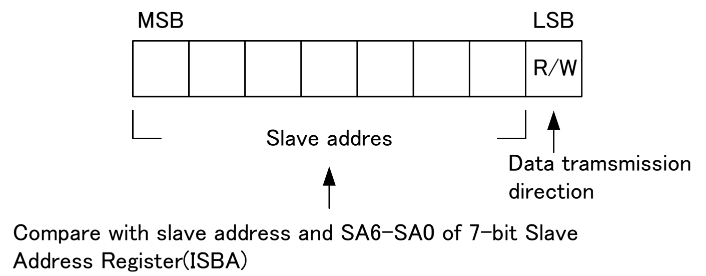
If set to "0": The slave address is not detected.
If set to "1": The ISBA and ISMK settings and the received 1st byte are compared.

Value	Description
0	Disable
1	Enable

[bit6:0] SA6 to SA0: 7-bit slave address

- If the slave address detection is enabled (SAEN=1), the 7-bit Slave Address Register (ISBA) compares the 7-bit data, which has been received after detection of (iteration) start condition, with this register value. If all bits match each other, slave mode is selected and an ACK is output. At this time, the received slave address is set in this register (if SAEN=0, no ACK is output).
- The 7-bit slave address and the direction of a data transfer is contained in the first byte after detection of (iteration) start condition. The slave address which are contained in the received data and these bits are compared.

Figure 5-1 The first byte format after detection of (iteration) start condition



- If an address bit is set to "0" in the ISMK register, it is not compared.

bit6:0	Description
	7-bit slave address

<Notes>

- The reserved address cannot be set.
- This register must be set only when the EN bit of ISMK register is "0".

5.9. Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0)

Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0) are used to set a frequency division ratio of serial clocks.

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	-	(BGR1)							(BGR0)							
Attribute	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The Baud Rate Generator Registers are used to set a frequency division ratio of serial clocks.
The BGR1 register corresponds to the high-order bits, and the BGR0 register corresponds to the low-order bits. The reload value to be counted can be written, and the BGR1/BGR0 set value can be read.
When the reload value is written in Baud Rate Generator Registers 1 and 0 (BGR1 and BGR0), the Reload counter starts its counting.

[bit15] -: Unused bit

This bit value is undefined when read.
This bit has no effect on the operation when written.

[bit14:8] BGR1: Baud Rate Generator Register 1

Process	Description
Write	Writes data in bit8 to bit14 of reload counter.
Read	Reads the BGR1 set value.

[bit7:0] BGR0: Baud Rate Generator Register 0

Process	Description
Write	Writes data in bit0 to bit7 of reload counter.
Read	Reads the BGR0 set value.

<Notes>

- Data must be written in the Baud Rate Generator Registers (BGR1 and BGR0) by 16-bit data accessing.
- The Baud Rate Generator Registers must be set when the EN bit of ISMK register is "0".
- The baud rate must be set regardless of master or slave mode selection.
- In operation mode 4 (I^2C mode), operate the bus clock at a frequency no lower than 8 MHz. Also note that setting of a baud rate generator that exceeds 400 kbps is prohibited.

5.10. FIFO Control Register 1 (FCR1)

The FIFO Control Register (FCR1) is used to select the transmit or received FIFO, enable the transmit FIFO interrupt, and control the interrupt flag.

bit	15	14	13	12	11	10	9	8	7 ... 0
Field		Reserved		FLSTE	FRIIE	FDRQ	FTIE	FSEL	(FCR0)
Attribute		-		R/W	R/W	R/W	R/W	R/W	
Initial value		-		0	-	1	0	0	

[bit15:13] Reserved: Reserved bits

The read value is "0". Be sure to write "0".

[bit12] FLSTE: Re-transmit data lost detection enable bit

This bit enables the FCR0:FLST bit detection.

If set to "0", the FCR0:FLST bit detection is disabled.
If set to "1", the FCR0:FLST bit detection is enabled.

Value	Description
0	Disables the Data Lost detection.
1	Enables the Data Lost detection.

<Note>

If you wish to set this bit to "1", set the FSET bit to "1" first, and then set this bit to "1".

[bit11] FRIIE: Received FIFO idle detection enable bit

This bit sets to detect the received idle state if the received FIFO contains valid data and if it continues more than 8-bit hours. If the received interrupt is enabled (SCR:RIE=1), a received interrupt is generated when the received idle state is detected.

Value	Description
0	Disables the received FIFO idle detection.
1	Enables the received FIFO idle detection.

<Note>

In case of using Received FIFO, set this bit to "1".

[bit10] FDRQ: Transmit FIFO data request bit

This bit requests for the transmit FIFO data.
If this bit is "1", the transmit data is being requested. If the Transmit Interrupt is enabled (FTIE=1) during this time, a transmit FIFO interrupt request is output.

The FDRQ bit is set when:

- The FBYTE (for transmission) is "0" (Transmit FIFO is empty).
- Transmit FIFO is reset.

The FDRQ bit is reset when:

- This bit is set to "0".
- Transmit FIFO is filled with data.

Value	Description
0	Does not request for the transmit FIFO data.
1	Requests for the transmit FIFO data.

<Notes>

- If the FBYTE (for transmission) is "0", this bit cannot be set to "0".
- If this bit is "0", the FSEL bit state cannot be changed.
- If this bit is set to "1", it has no effect on the operation.
- If a read-modify-write instruction is issued, "1" is read.
- If a transmit interrupt has occurred and you have written the required data in transmit FIFO, clear the interrupt request by setting the FIFO transmit data request bit (FCR1:FDRQ) to "0".

[bit9] FTIE: Transmit FIFO interrupt enable bit

This bit enables a transmit FIFO interrupt. If this bit is set to "1", an interrupt occurs when the FDRQ bit is set to "1".

Value	Description
0	Disables the transmit FIFO interrupt.
1	Enables the transmit FIFO interrupt.

[bit8] FSEL: FIFO buffer selection bit

This bit selects the transmit or received FIFO.

Value	Description
0	Set transmit FIFO as FIFO1, and the received FIFO as FIFO2.
1	Set transmit FIFO as FIFO2, and the received FIFO as FIFO1.

<Notes>

- This bit is not cleared by FIFO reset (FCR0:FCL[2:1]=11).
- To change this bit state, first disable the FIFO operation (FCR0:FE[2:1]=00).

5.11. FIFO Control Register 0 (FCR0)

The FIFO Control Register 0 (FCR0) is used to enable/disable the FIFO operation, reset FIFO, save the read pointer, and set the data re-transmission.

bit	15 ... 8	7	6	5	4	3	2	1	0
Field	(FCR1)	-	FLST	FLD	FSET	FCL2	FCL1	FE2	FE1
Attribute		-	R	R/W	R/W	R/W	R/W	R/W	R/W
Initial value		0	0	0	0	0	0	0	0

[bit7] - : Unused bit

When read, "0" is always read.
When writing, always set to "0".

[bit6] FLST: FIFO re-transmit data lost flag bit

This bit shows that the re-transmit data of transmit FIFO has been lost.

The FLST bit is set when:

· If the FLSTE bit of FIFO Control Register 1 (FCR1) is "1", the write pointer of transmit FIFO matches the read pointer which has been saved by the FSET bit, and data is written in the FIFO buffer.

The FLST bit is reset when:

· FIFO is reset (FCL bit is set to "1").
· The FSET bit is set to "1".

If this bit is set to "1", the data which has been saved by the FSET bit and identified by the read pointer is overwritten. The data re-transmission cannot be set by the FLD bit even if an error has occurred. If this bit is set to "1" and if you wish to re-transmit data, first reset FIFO. Then, write data in the FIFO buffer again.

Value	Description
0	No Data Lost has occurred.
1	Data Lost has occurred.

[bit5] FLD: FIFO pointer reload bit

This bit reloads the data, being saved in transmit FIFO by the FSET bit, to the reload pointer. This bit can be used to re-transmit data after a communication error or others have occurred.
When the re-transmission setting has finished, this bit is set to "0".

Value	Description
0	Not reloaded
1	Reloaded

<Notes>

- If this bit is "1", data is being reloaded in the read pointer. Therefore, data writing except for FIFO reset is disabled.
- When FIFO is enabled or when data is being transmitted, this bit cannot be set to "1".
- Set the SMR:TIE bit to "0" first, and set this bit to "1". Then, enable transmit FIFO and set the SMR:TIE bit to "1".

[bit4] FSET: FIFO pointer save bit

This bit saves the read pointer value of transmit FIFO.
If the read pointer value is saved before being transmitted and if the FLST bit is "0", the data can be re-transmitted even if a communication error or others have occurred.

If set to "1", the current read pointer value is saved.
If set to "0", it has no effect on the operation.

Value	Description
0	Not saved
1	Saved

<Note>

This bit can be set to "1" only when the transmit byte count (FBYTE) is "0".

[bit3] FCL2: FIFO2 reset bit

This bit resets the FIFO2 value.
If this bit is set to "1", the FIFO2 buffer is initialized.
Only the FCR0:FLST bit is initialized, but the other bits of FCR1/0 registers are kept.

Value	Description	
	At writing	At reading
0	No effect on operation.	"0" is always read.
1	FIFO2 is reset.	

<Notes>

- Disable the FIFO2 operation first, and then reset the FIFO2 buffer.
- Set the transmit FIFO interrupt enable bit to "0" before the execution.
- The FBYTE2 register has the significant data count of "0".

[bit2] FCL1: FIFO1 reset bit

This bit resets the FIFO1 value.
If this bit is set to "1", the FIFO1 buffer is initialized.
Only the FCR0:FLST bit is initialized, but the other bits of FCR1/0 registers are kept.

Value	Description	
	At writing	At reading
0	No effect on operation.	"0" is always read.
1	FIFO1 is reset.	

<Notes>

- Disable the FIFO1 operation first, and then reset FIFO1.
- Set the transmit FIFO interrupt enable bit to "0" before the execution.
- The FBYTE1 register has the significant data count of "0".

[bit1] FE2: FIFO2 operation enable bit

This bit enables or disables the FIFO2 operation.

- To use the FIFO2 operation, set this bit to "1".
- If received FIFO is selected by the FCR1:FSEL bit and if a received error has occurred, this bit is cleared to "0". This bit cannot be set to "1" until the received error is cleared.
- To use FIFO2 as transmit FIFO, this bit must be set to "1" or "0" when the transmit data is empty (SSR:TDRE=1).
- To use FIFO2 as received FIFO, this bit must be set to "0" when the received buffer is empty (SSR:RDRF=0) and received FIFO contains no valid data (FBYTE2=0) while the I^2C interface operation is disabled (ISMK:EN=0), the operation flag (IBCR:ACT) is "0", or the interrupt flag (IBCR:INT) is "1".
- To use FIFO2 as received FIFO, this bit must be set to "1" when the received buffer is empty (SSR:RDRF=0) while the I^2C interface operation is disabled (ISMK:EN=0), the operation flag (IBCR:ACT) is "0", or the interrupt flag (IBCR:INT) is "1".
- The FIFO2 state is held even if the FIFO2 operation is disabled.

Value	Description
0	Disables the FIFO2 operation.
1	Enables the FIFO2 operation.

<Notes>

- The enable or disable state must be switched only when the IBSR:BB bit is "0" or when the IBCR:INT bit is "1".
- If received FIFO is selected and the reserved address is detected, and if you wish to select the slave mode transmission, set this bit to "0" and set IBCR:ACKE bit to "0" with an interrupt of reserved address detection.
- If received FIFO is selected and if the SSR:RDRF bit of SSR is "1" when this bit is changed from "1" to "0", received FIFO is not disabled until the bit is set to "0".
- If transmit FIFO is selected, FIFO2 contains data, and you wish to change this bit from "0" to "1", set the SMR:TIE bit to "0" first. Then, set this bit to "1", and set the SMR:TIE bit to "1".

[bit0] FE1: FIFO1 operation enable bit

This bit enables or disables the FIFO1 operation.

- To use the FIFO1 operation, set this bit to "1".
- If received FIFO is selected by the FCR1:FSEL bit and if a received error has occurred, this bit is cleared to "0". This bit cannot be set to "1" until the received error is cleared.
- To use FIFO1 as transmit FIFO, this bit must be set to "1" or "0" when the transmit data is empty (SSR:TDRE=1).
- To use FIFO1 as received FIFO, this bit must be set to "0" when the received buffer is empty (SSR:RDRF=0) and received FIFO contains no valid data (FBYTE2=0) while the I^2C interface operation is disabled (ISMK:EN=0), the operation flag (IBCR:ACT) is "0", or the interrupt flag (IBCR:INT) is "1".
- To use FIFO1 as received FIFO, this bit must be set to "1" when the received buffer is empty (SSR:RDRF=0) while the I^2C interface operation is disabled (ISMK:EN=0), the operation flag (IBCR:ACT) is "0", or the interrupt flag (IBCR:INT) is "1".
- The FIFO1 state is held even if the FIFO1 operation is disabled.

Value	Description
0	Disables the FIFO1 operation.
1	Enables the FIFO1 operation.

<Notes>

- The enable or disable state must be switched only when the IBSR:BB bit is "0" or when the IBCR:INT bit is "1".
- If received FIFO is selected and the reserved address is detected, and if you wish to select the slave mode transmission, set this bit to "0" and set IBCR:ACKE bit to "0" with an interrupt of reserved address detection.
- If received FIFO is selected and the SSR:RDRF bit is "1" when this bit is changed from "1" to "0", received FIFO is not disabled until the bit is set to "0".
- If transmit FIFO is selected, FIFO1 contains data, and if you wish to change this bit from "0" to "1" state, set the SMR:TIE bit to "0" first. Then, set this bit to "1", and set the SMR:TIE bit to "1".

5.12. FIFO Byte Register (FBYTE)

The FIFO Byte Register (FBYTE) indicates the effective data count in the FIFO buffer. Also, this register can be used to generate a received interrupt when certain number of data sets are received in the received FIFO.

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	(FBYTE2)								(FBYTE1)							
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The FBYTE register indicates the effective data count in the FIFO buffer. The following table shows the relation between the FCR1:FSEL bit state and FBYTE.

Table 5-3 Display of data count

FSEL	FIFO selection	Data count display
0	FIFO2:Received FIFO, FIFO1: Transmit FIFO	FIFO2:FBYTE2, FIFO1:FBYTE1
1	FIFO2:Transmit FIFO, FIFO1:Received FIFO	FIFO2:FBYTE2, FIFO1:FBYTE1

- The initial value of data transfer count is "0x08" for the FBYTE register.
- Set a data count to generate a received interrupt flag for the FBYTE register of received FIFO. If this transfer data count matches the FBYTE register display, the received data full flag bit (SSR:RDRF) is set to "1".
- If both of the following conditions are satisfied and if the received idle state continues for more than 8 baud rate clocks, the received data full flag bit (SSR:RDRF) is set to "1".
  - The received FIFO idle detection enable bit (FCR:FRIIE) is "1".
  - The number of data sets stored in the received FIFO does not reach the transfer count.

  If the RDR data is read during counting of 8 clocks, this counter is reset to 0 and counting for 8 clocks is restarted. If received FIFO is disabled, this counter is reset to 0. If data remains in the received FIFO and if received FIFO is enabled, the data counting is restarted.
- To receive data in the master mode operation (master mode reception), set the SMR:TIE bit to "0", set the received data count for the FBYTE register of transmit FIFO, and set the FCR1:FDRQ bit to "0". The SCL clocks are output for the specified data count, and then IBCR:INT bit is set to "1". The SMR:TIE bit must be set to "1" only after the FCR1:FDRQ bit is set to "1".

[bit15:8] FBYTE2: FIFO2 data count display bits

[bit7:0] FBYTE1: FIFO1 data count display bits

Writing	Sets the transfer data count.
Reading	Reads the effective count of data.

Read (Effective data count)

During transmission: The number of data sets already written in the FIFO buffer but not transmitted yet
During reception: The number of data sets received in FIFO

Write (Transfer data count)

During transmission: Set "0x00".
During reception: Set the data count to generate a received interrupt.

<Notes>

- The FBYTE value of transmit FIFO must be "0x00" except when data is received in the master mode operation.
- During the master mode data reception, the transmit data count must be set only when transmit FIFO is empty and the SMR:TIE bit is "0".
- When data is being received in the master mode operation, the I^2C interface operation can be disabled (ISMK:EN=0) only after transmit/received FIFO has been disabled.
- Setting of a send data number when receiving the data by master operation must be executed when the transmit FIFO is empty and SMR:TIE bit is "0".
- The FBYTE bit of received FIFO must be set to "1" or larger.
- Change this register under one of the following conditions:
  - When the I^2C interface operation is disabled (ISMK:EN=0)
  - When IBCR:INT=1 in case of SSR:DMA=0 and master mode reception
  - When SSR:TBI=1 in case of SSR:DMA=1 and master mode reception
- A read-modify-write instruction cannot be used for this register.
- Any setting exceeding the FIFO capacity is inhibited.
- To receive data in the master mode operation (master mode reception), do not write dummy data to the Transmit Data Register (TDR) when setting the SMR:TIE bit to "0" and setting the received data count for the FBYTE register of transmit FIFO.

CHAPTER 1-6: I^2C Auxiliary Noise Filter

This chapter explains the I^2C auxiliary noise filter.

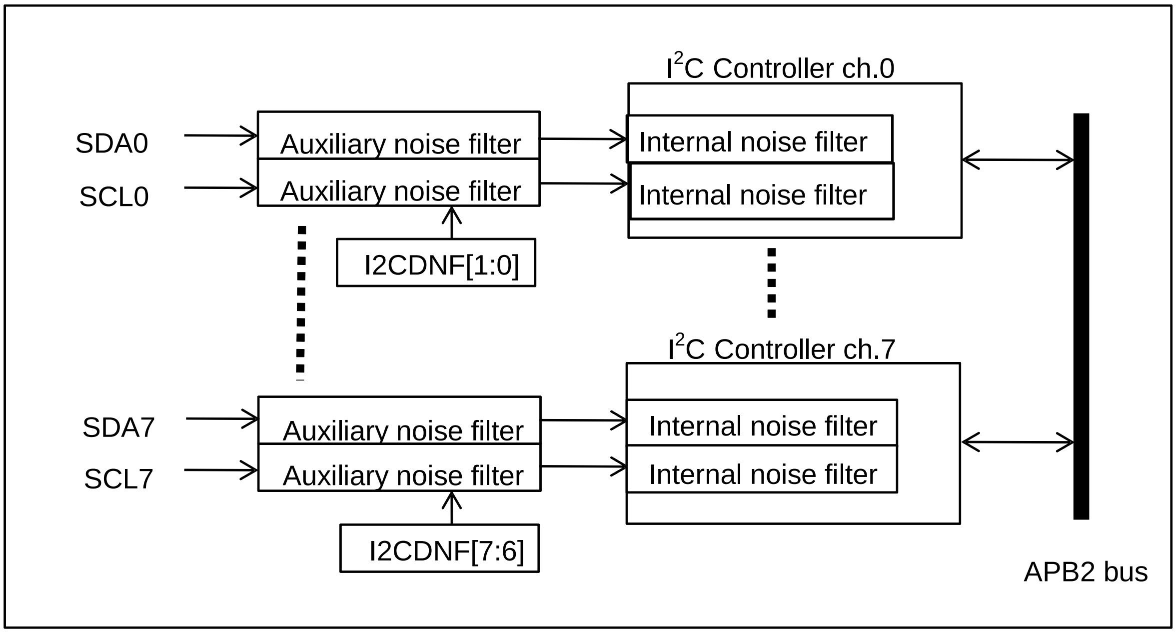
CODE: 9BFBDNF-E01.4

1. Overview and Configuration

This section describes the overview of the I^2C auxiliary noise filter.

If the APB2 bus clock frequency exceeds 40 MHz when using the I^2C bus interface, you must insert a noise filter in the SDA/SCL input path. (In addition to this noise filter, two steps of the noise filters are incorporated in the I^2C controller)

Figure 1-1 Block diagram of I^2C input



■ Description of operation

Set the digital noise filter control register (I2CDNF) according to the APB2 bus clock frequency. The noise filter for the selected number of steps in I2CDNF is added in the Digital Noise Filter circuit and eliminate the input noise at a maximum of 50 ns.

■ Calculation of the baud rate

When this auxiliary noise filter is used, the calculation formula of the reload value that should be set to the Baud rate generator registers (BGR1, BGR0) is different from that in chapter "I^2C Interface (I^2C Communications Control Interface)". Use the following calculation formula to calculate the reload value when using the auxiliary noise filter.

Reload value:

$$V = \phi / b - (nf + 5)$$

V: Reload value b: Baud rate ϕ : Bus clock frequency or external clock frequency
nf: The number of steps of the added noise filter (that is selected with the I2CDNF register)

Note that the preset baud rate may not be generated at a rising edge of signal on I^2C bus.
In such case, adjust the reload value.

Consequently, Baud rate is as follows;

$$b \text{ [bps]} = \phi \text{ [Hz]} / (V + nf + 5)$$

2. Register of I^2C Auxiliary Noise Filter

This section describes the register of the I^2C auxiliary noise filter.

Register abbreviation	Register name	Reference
I2CDNF	I^2C auxiliary noise filter setting register	2.1

2.1. I^{2}C Auxiliary Noise Filter Setting Register (I2CDNF)

Set the number of steps of the auxiliary noise filter according to the APB2 bus clock frequency.

■ Register configuration

bit	15 14	13 12	11 10	9 8
Field	I2CDNF7	I2CDNF6	I2CDNF5	I2CDNF4
Attribute	R/W	R/W	R/W	R/W
Initial value	00	00	00	00

bit	7 6	5 4	3 2	1 0
Field	I2CDNF3	I2CDNF2	I2CDNF1	I2CDNF0
Attribute	R/W	R/W	R/W	R/W
Initial value	00	00	00	00

■ Register function

[bit15:14] I2CDNF7: Auxiliary noise filter additional step select bits for I^{2}C ch.7

Value	Description
00	Does not add auxiliary noise filter. APB2 bus clock ≤ 40 MHz [initial value]
01	Add 1 step of auxiliary noise filter. 40 MHz < APB2 bus clock ≤ 60 MHz
10	Add 2 steps of auxiliary noise filter. 60 MHz < APB2 bus clock ≤ 80 MHz
11	Add 3 steps of auxiliary noise filter. 80 MHz < APB2 bus clock ≤ 100 MHz

[bit13:12] I2CDNF6: Auxiliary noise filter additional step select bits for I^{2}C ch.6

Value	Description
00	Does not add auxiliary noise filter. APB2 bus clock ≤ 40 MHz [initial value]
01	Add 1 step of auxiliary noise filter. 40 MHz < APB2 bus clock ≤ 60 MHz
10	Add 2 steps of auxiliary noise filter. 60 MHz < APB2 bus clock ≤ 80 MHz
11	Add 3 steps of auxiliary noise filter. 80 MHz < APB2 bus clock ≤ 100 MHz

[bit11:10] I2CDNF5: Auxiliary noise filter additional step select bits for I^{2}C ch.5

Value	Description
00	Does not add auxiliary noise filter. APB2 bus clock ≤ 40 MHz [initial value]
01	Add 1 step of auxiliary noise filter. 40 MHz < APB2 bus clock ≤ 60 MHz
10	Add 2 steps of auxiliary noise filter. 60 MHz < APB2 bus clock ≤ 80 MHz
11	Add 3 steps of auxiliary noise filter. 80 MHz < APB2 bus clock ≤ 100 MHz

[bit9:8] I2CDNF4: Auxiliary noise filter additional step select bits for I^{2}C ch.4

Value	Description
00	Does not add auxiliary noise filter. APB2 bus clock ≤ 40 MHz [initial value]
01	Add 1 step of auxiliary noise filter. 40 MHz < APB2 bus clock ≤ 60 MHz
10	Add 2 steps of auxiliary noise filter. 60 MHz < APB2 bus clock ≤ 80 MHz
11	Add 3 steps of auxiliary noise filter. 80 MHz < APB2 bus clock ≤ 100 MHz

[bit7:6] I2CDNF3: Auxiliary noise filter additional step select bits for I^{2}C ch.3

Value	Description
00	Does not add auxiliary noise filter. APB2 bus clock ≤ 40 MHz [initial value]
01	Add 1 step of auxiliary noise filter. 40 MHz < APB2 bus clock ≤ 60 MHz
10	Add 2 steps of auxiliary noise filter. 60 MHz < APB2 bus clock ≤ 80 MHz
11	Add 3 steps of auxiliary noise filter. 80 MHz < APB2 bus clock ≤ 100 MHz

[bit5:4] I2CDNF2: Auxiliary noise filter additional step select bits for I^{2}C ch.2

Value	Description
00	Does not add auxiliary noise filter. APB2 bus clock ≤ 40 MHz [initial value]
01	Add 1 step of auxiliary noise filter. 40 MHz < APB2 bus clock ≤ 60 MHz
10	Add 2 steps of auxiliary noise filter. 60 MHz < APB2 bus clock ≤ 80 MHz
11	Add 3 steps of auxiliary noise filter. 80 MHz < APB2 bus clock ≤ 100 MHz

[bit3:2] I2CDNF1: Auxiliary noise filter additional step select bits for I^{2}C ch.1

Value	Description
00	Does not add auxiliary noise filter. APB2 bus clock ≤ 40 MHz [initial value]
01	Add 1 step of auxiliary noise filter. 40 MHz < APB2 bus clock ≤ 60 MHz
10	Add 2 steps of auxiliary noise filter. 60 MHz < APB2 bus clock ≤ 80 MHz
11	Add 3 steps of auxiliary noise filter. 80 MHz < APB2 bus clock ≤ 100 MHz

[bit1:0] I2CDNF0: Auxiliary noise filter additional step select bits for I^{2}C ch.0

Value	Description
00	Does not add auxiliary noise filter. APB2 bus clock ≤ 40 MHz [initial value]
01	Add 1 step of auxiliary noise filter. 40 MHz < APB2 bus clock ≤ 60 MHz
10	Add 2 steps of auxiliary noise filter. 60 MHz < APB2 bus clock ≤ 80 MHz
11	Add 3 steps of auxiliary noise filter. 80 MHz < APB2 bus clock ≤ 100 MHz

<Note>

- The AC characteristics (tSP) described in the Data Sheet are (noise filter total step number) × $t_{CYCP.}$ Since the I^2C controller incorporates 2 steps of the noise filter, the tSP noise filter is added is as follows:
- I2CDNFx = 0b00: tSP = (2) × t_{CYCP}
- I2CDNFx = 0b01: tSP = (2 + 1) × t_{CYCP}
- I2CDNFx = 0b10: tSP = (2 + 2) × t_{CYCP}
- I2CDNFx = 0b11: tSP = (2 + 3) × t_{CYCP}

CHAPTER 2-1: USB/Ethernet Clock Generation Block

This chapter explains the USB/Ethernet clock generation.

CODE: 9BFBSPLL-E01.3

1. Overview and Configuration

Generating USB clock and Ethernet clock

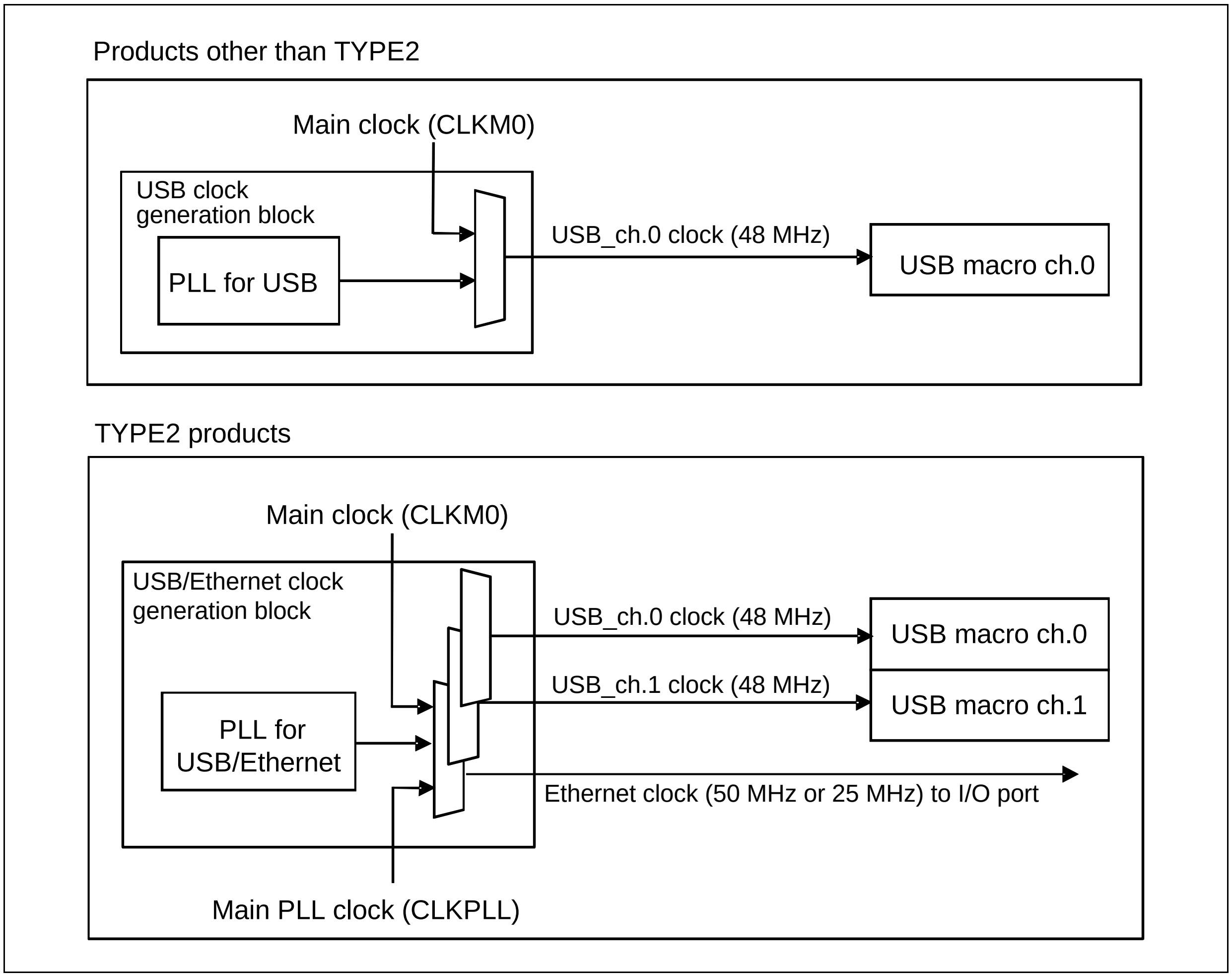
This block generates a 48 MHz USB clock used in USB macro communication and a 50 MHz (RMII)/25 MHz (MII) Ethernet clock used in Ethernet communication.

Since the function and configuration differ by products, see the chapter "USB Clock Generation" for the products other than TYPE2, and see the chapter "USB/Ethernet Clock Generation" for TYPE2 products.

Figure 1-1 shows a block diagram of a USB clock and a USB/Ethernet clock generation block.

■ Block diagram of USB clock and USB/Ethernet clock generation block

Figure 1-1 USB clock and USB/Ethernet clock generation block



■ Difference between USB/Ethernet clock generation block (TYPE2) and USB clock generation block (other than TYPE2)

In the USB/Ethernet clock block, the following functions are added from the USB clock generation block.

- Outputting 2 channels of a USB clock
- Outputting an Ethernet clock
- The Main PLL clock (CLKPLL) can be used as a USB/Ethernet clock.
- A USB-PLL/Ethernet control function is added in timer mode.

CHAPTER 2-2: USB Clock Generation

This chapter explains the USB clock generation.

CODE: 9BFUSBPRE-E03.0

1. Overview

This section provides an overview of the USB clock generation.

The USB clock runs at 48 MHz and is used by USB macro for communication.

An external 48 MHz main clock (hereinafter CLKMO) can be used for the USB clock, or a 48 MHz clock can be generated using USB PLL (hereinafter USB-PLL).

The USB clock generation unit is responsible for the following functions:

- Enables or stops output of the USB clock.
- Selects the USB clock.
- Enables or stops oscillation of USB-PLL.
- Selects the input clock of USB-PLL.
- Sets the input clock frequency division of USB-PLL.
- Sets the output clock multiplication of USB-PLL.
- Sets the stabilization wait time of USB-PLL.
- Stops the USB clock in standby mode.

2. Configuration and Block Diagram

This section explains the configuration and block diagram of the USB clock generation unit.

Figure 2-1 Block diagram of USB clock generation unit (TYPE0 products)

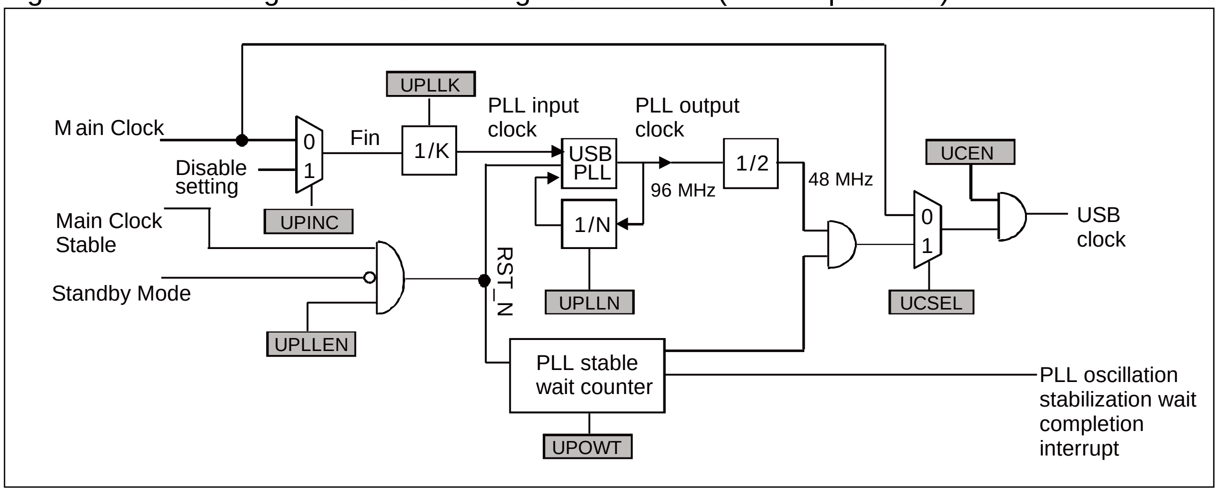
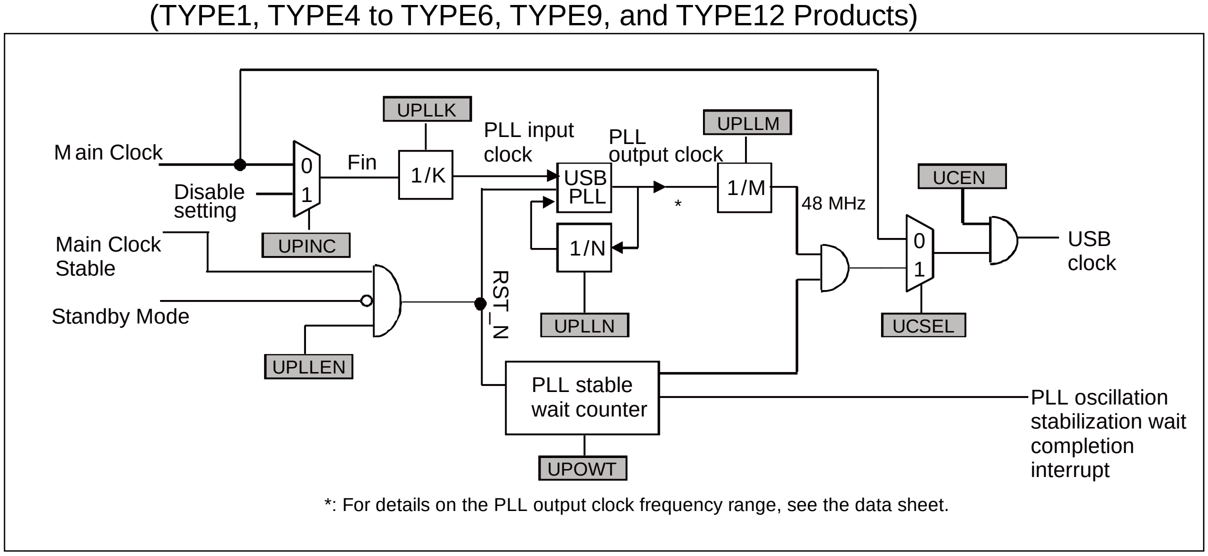


Figure 2-2 Block diagram of USB clock generation unit (TYPE1, TYPE4 to TYPE6, TYPE9, and TYPE12 Products)



■ USB-PLL Control Register (UPLLEN)

- The control register can enable USB-PLL oscillation.

■ Input Clock Select Register (UPINC)

- Be sure to select the CLKMO.

■ USB-PLL

- Frequency division setting register (UPLLK, UPLLN, UPLLM)
  To generate 48 MHz as USB clock, the settings of K frequency division, N frequency division and M frequency division are required (M frequency division is not available for TYPE0 products).

  For the specification range of the USB-PLL input clock frequency, output clock frequency, and PLL macro multiplier (N division setting value), refer to the PLL use conditions of "PLL input clock frequency", "PLL macro oscillation clock frequency", and "PLL multiplier" in "Data Sheet" of the product used.

- Oscillation stabilization wait time setting (UPOWT)
  Oscillation stabilization wait time for USB-PLL can be specified.

■ Output clock

- Output Clock Select Register (UCSEL)
  Can be selected from CLKMO or USB-PLL clock.

- PLL Clock Output Enable Register (UCEN)
  Can set the USB clock output enable.

■ Standby mode setting

- The Standby-Mode signal shown in Figure 2-1 and Figure 2-2 turns to be active in the following modes.
  The USB clock stops in the following standby modes.
  - Stop mode
  - TIMER mode
- The Main Clock stable signal shown in Figure 2-1 and Figure 2-2 is an oscillation stabilization signal for each mode.

3. Explanation of Operation

This section explains the operation of the USB clock generation unit.

■ Selecting the USB clock

The following two types of clocks can be selected for the USB clock.

● CLKMO

CLKMO can be used directly as the USB clock. In this case, CLKMO must be input externally at 48 MHz, or must oscillate at 48 MHz. Enable the output of the USB clock after confirming stabilization of the CLKMO oscillation.

● Selecting the USB-PLL output clock

The USB-PLL output clock can be used as the USB clock.

· TYPE0 products
The USB-PLL output clock must be output at 96 MHz. The USB-PLL output clock is divided by two to generate a 48 MHz clock.

· TYPE1/TYPE4/TYPE5 products
The USB-PLL output clock must be output at 240 MHz or 288 MHz to generate a 48 MHz clock after M division.

· TYPE6/TYPE9/TYPE12 products
The USB-PLL output clock must be output at 96 MHz or 144 MHz to generate a 48 MHz clock after M division.

Table 3-1 below shows the setting example of the division ratio.

Table 3-1 Example of PLL frequency division ratio settings

Porducts TYPE	Fin [MHz]	K	N	M	PLL clock frequency [MHz]
TYPE0	4	1	24	-	96
	8	1	12	-	
	8	2	24	-	
	16	1	6	-	
	16	2	12	-	
	16	4	24	-	
	24	2	8	-	
	24	4	16	-	
	24	6	24	-	
TYPE1, TYPE4, TYPE5	4	1	60	5	240
	8	1	30	5	
	8	2	60	5	
	16	1	15	5	
	16	2	30	5	
	16	4	60	5	
	24	2	20	5	
	24	4	40	5	
	24	6	60	5	
TYPE6, TYPE9, TYPE12	4	1	24	2	96
	8	1	12	2	
	8	2	24	2	
	16	1	6	2	
	16	2	12	2	
	16	4	24	2	
	24	2	8	2	
	24	4	16	2	
	24	3	8	2	

■ Changing to standby mode

● When changing to standby mode

Before changing to standby mode (Stop mode, RTC mode, or TIMER mode), set UCCR:UCEN to "0" to stop the USB clock supply.

1. Set UCCR:UCEN to "0".
2. Read the UCCR Register to check that UCEN is set to "0".
3. Changing to standby mode.

When returning from standby mode, set UCEN to "1". The supply starts when the USB clock oscillation has been stabilized. Take either of the following actions to confirm whether or not the USB clock oscillation has been stabilized.

a) When USB-PLL is used
Check that UPRDY is "1", or use the USB-PLL oscillation stabilization wait interrupt.

b) When CLKMO (48 MHz) is used
After the CLKMO oscillation has been stabilized, supply the USB clock.

■ USB-PLL oscillation stabilization wait settings

● Oscillation stabilization wait time for USB-PLL can be specified

After CLKMO oscillation has been stabilized, the oscillation stabilization wait time for USB-PLL begins to be counted.

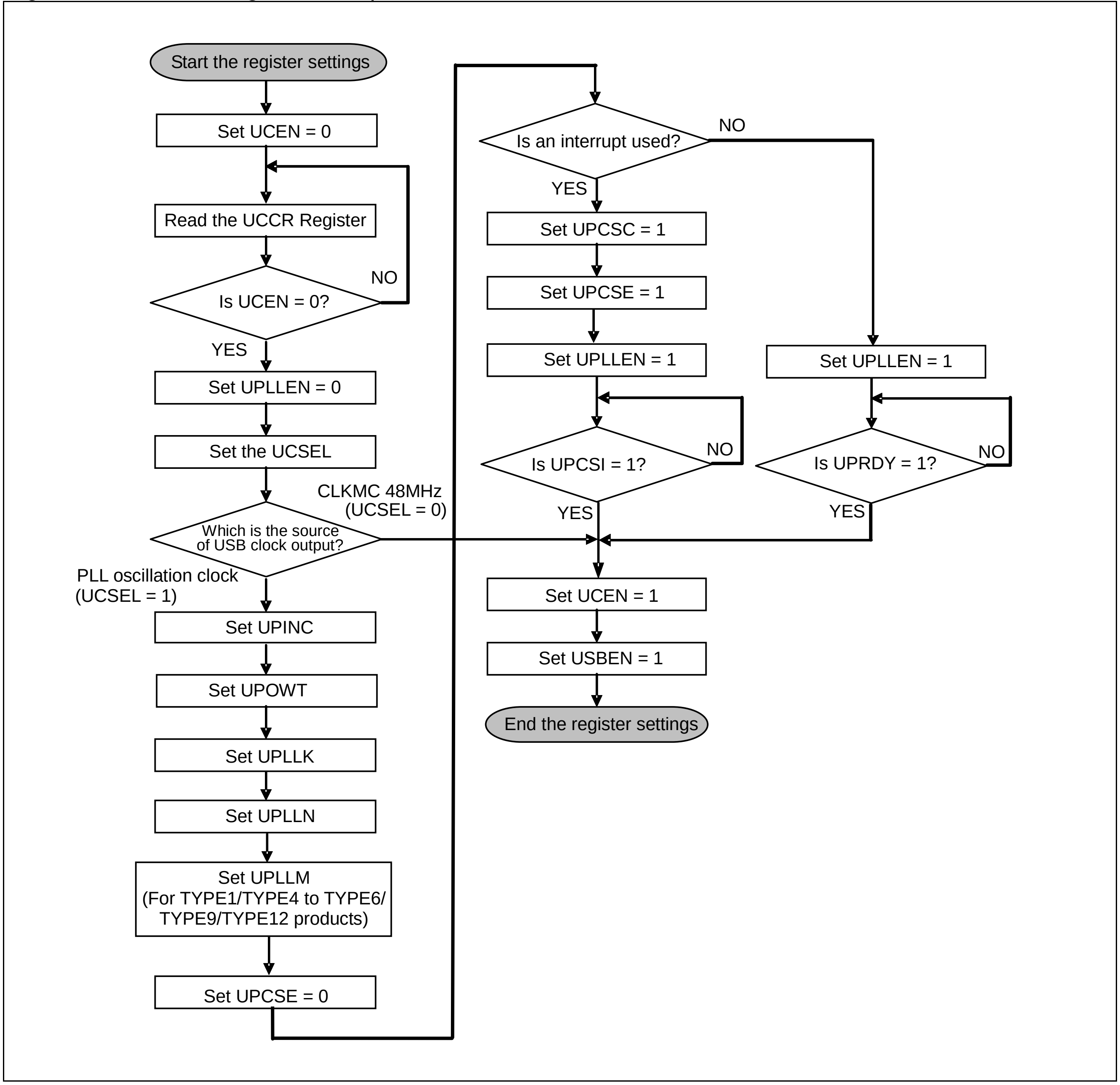
Before enabling the USB-PLL oscillation, configure the oscillation stabilization wait time for USB-PLL and the oscillation stabilization complete interrupt.Do not change the oscillation stabilization wait time while waiting for oscillation to stabilize.

4. Setup Procedure Example

This section explains an example of setting up the USB clock generation unit.

Figure 4-1 shows an example of setting up the USB clock.

Figure 4-1 USB clock generation procedure



5. Register List

This section explains the register list of the USB clock generation unit.

■ The register list of the USB clock generation unit

Abbreviation	Register name	Reference
UCCR	USB Clock Control Register	5.1
UPCR1	USB-PLL Control Register 1	5.2
UPCR2	USB-PLL Control Register 2	5.3
UPCR3	USB-PLL Control Register 3	5.4
UPCR4	USB-PLL Control Register 4	5.5
UPCR5	USB-PLL Control Register 5	5.6
UP_STR	USB-PLL Status Register	5.7
UPINT_ENR	USB-PLL Interrupt factor Enable Register	5.8
UPINT_CLR	USB-PLL Interrupt factor Clear Register	5.10
UPINT_STR	USB-PLL Interrupt factor Status Register	5.9
USBEN	USB Enable Register	5.11

5.1. USB Clock Control Register (UCCR)

The UCCR selects the USB clock and enables/disables the USB clock output.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved						UCSEL	UCEN
Attribute	-						R/W	R/W
Initial value	-						0	0

■ Register functions

[bit7:2] Reserved: Reserved bits

"0b000000" is read from these bits.
Set these bits to "0b000000" when writing.

[bit1] UCSEL: USB clock selection bit

Value	Description
0	CLKMO [Initial value]
1	USB-PLL oscillation clock

[bit0] UCEN: USB clock output enable bit

Value	Description
0	Disables the USB clock output [Initial value]
1	Enables the USB clock output

<Notes>

- When selecting the main clock with UCSEL, the 48 MHz frequency must be input from an external main oscillation.
- This register is not initialized by software reset.

5.2. USB-PLL Control Register1 (UPCR1)

The UPCR1 sets USB-PLL.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved						UPINC	UPLLEN
Attribute	-						R/W	R/W
Initial value	-						0	0

■ Register functions

[bit7:2] Reserved: Reserved bits

"0b000000" is read from these bits.

Set these bits to "0b000000" when writing.

[bit1] UPINC: USB-PLL input clock selection bit

Value	Description
0	CLKMO [Initial value]
1	Setting is prohibited.

[bit0] UPLLEN: USB-PLL oscillation enable bit

Value	Description
0	Stops USB-PLL [Initial value]
1	Enables the USB-PLL oscillation

<Notes>

- Be sure to set UPINC to "0". Operation is not guaranteed when UPINC is set to "1".
- This register is not initialized by software reset.

5.3. USB-PLL Control Register2 (UPCR2)

The UPCR2 sets the oscillation stabilization wait time of USB-PLL.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved					UPOWT		
Attribute	-					R/W		
Initial value	-					000		

■ Register functions

[bit7:3] Reserved: Reserved bits

"0b00000" is read from these bits.
Set these bits to "0b00000" when writing.

[bit2:0] UPOWT: USB-PLL oscillation stabilization wait time setting bits

bit2	bit1	bit0	Description
0	0	0	2^9/Fin : Approx. 128 µs * [Initial value]
0	0	1	2^{10}/Fin : Approx. 256 µs *
0	1	0	2^{11}/Fin : Approx. 512 µs *
0	1	1	2^{12}/Fin : Approx. 1.02 ms *
1	0	0	2^{13}/Fin : Approx. 2.05 ms *
1	0	1	2^{14}/Fin : Approx. 4.10 ms *
1	1	0	2^{15}/Fin : Approx. 8.20 ms *
1	1	1	2^{16}/Fin : Approx. 16.4 ms *

*: When Fin = 4 MHz

<Notes>

- F_{in} is the clock (CLKMO) selected by UPINC.
- This register is not initialized by software reset.
- Since the oscillation stabilization wait time for PLL macro differs by products, refer to the use conditions of "PLL oscillation stabilization wait time" in "Data Sheet" of the product used.

5.4. USB-PLL Control Register 3 (UPCR3)

The UPCR3 sets the frequency division ratio (K) of USB-PLL macro.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved			UPLLK				
Attribute	-			R/W				
Initial value	-			00000				

■ Register functions

[bit7:5] Reserved: Reserved bits

"0b000" is read from these bits.
Set these bits to "0b000" when writing.

[bit4:0] UPLLK: Frequency division ratio (K) setting bits of the USB-PLL clock

Value	Description
00000	Divides the frequency by (UPLLK+1) (Example) UPLLK = "00000" => 1/1 frequency [Initial value]
00001	
•	
•	
11111	

<Note>

This register is not initialized by software reset.

5.5. USB-PLL Control Register 4 (UPCR4)

The UPCR4 Register sets the frequency division ratio (N) of USB-PLL (specification differs by TYPE products).

■ TYPE0/TYPE6/TYPE9/TYPE12 products

● Register configuration

bit	7	6	5	4	3	2	1	0
Field		Reserved				UPLLN		
Attribute		-				R/W		
Initial value		-				10111		

● Register functions

[bit7:5] Reserved: Reserved bits

"0b000" is read from these bits.
Set these bits to "0b000" when writing.

[bit4:0] UPLLN: Frequency division ratio (N) setting bits of the USB-PLL clock

Value	Description
00000	Setting is prohibited.
00001	Divides the frequency by (UPLLN+1) (Example) UPLLN = "10111" => 1/24 frequency [Initial value]
•	
•	
11111	

<Note>

This register is not initialized by software reset.

■ TYPE1/TYPE4/TYPE5 products

● Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved	UPLLN						
Attribute	-	R/W						
Initial value	-	0111011						

● Register functions

[bit7] Reserved: Reserved bit

"0b0" is read from this bit.
Set this bit to "0b0" when writing.

[bit6:0] UPLLN: Frequency division ratio (N) setting bits of the USB-PLL clock

Value	Description
0000000	Setting is prohibited.
•	
0001100	
0001101	Divides the frequency by (UPLLN+1) (Example) UPLLN = "0111011" => 1/60 frequency [Initial value]
•	
•	
1100011	
1100100	Setting is prohibited.
•	
1111111	

<Note>

This register is not initialized by software reset.

5.6. USB-PLL Control Register 5 (UPCR5)

The UPCR5 sets the frequency division ratio (M) of USB-PLL (not available for TYPE0 products. specification differs by TYPE products).

■ TYPE1/TYPE4/TYPE5 products

● Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved				UPLLM			
Attribute	-				R/W			
Initial value	-				0100			

● Register functions

[bit7:4] Reserved: Reserved bits

"0b0000" is read from these bits.
Set these bits to "0b0000" when writing.

[bit3:0] UPLLM: Frequency division ratio (M) setting bits of the USB-PLL clock

Value	Description
0000	Divides the frequency by (UPLLM+1) (Example) UPLLM = "0100" => 1/5 frequency [Initial value]
0001	
•	
•	
1111	

<Note>

This register is not initialized by software reset.

■ TYPE6/TYPE9/TYPE12 products

● Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved				UPLLM			
Attribute	-				R/W			
Initial value	-				0001			

● Register functions

[bit7:4] Reserved: Reserved bits

"0b0000" is read from these bits.
Set these bits to "0b0000" when writing.

[bit3:0] UPLLM: Frequency division ratio (M) setting bits of the USB-PLL clock

Value	Description
0000	Divides the frequency by (UPLLM+1) (Example) UPLLM = "0001" => 1/2 frequency [Initial value]
0001	
•	
•	
1111	

<Note>

This register is not initialized by software reset.

5.7. USB-PLL Status Register (UP_STR)

The UP_STR indicates the macro status of USB-PLL.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved							UPRDY
Attribute	-							R
Initial value	-							0

■ Register functions

[bit7:1] Reserved: Reserved bits

"0b0000000" is read from these bits.
Set these bits to "0b0000000" when writing.

[bit0] UPRDY: USB-PLL oscillation stabilization bit

Value	Description
0	In a stabilization wait or an oscillation stop state [Initial value]
1	In a stabilized state

<Note>

This register is not initialized by software reset.

5.8. USB-PLL Interrupt Factor Enable Register (UPINT_ENR)

The UPINT_ENR enables/disables the USB-PLL oscillation stabilization wait complete interrupt.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved							UPCSE
Attribute	-							R/W
Initial value	-							0

■ Register functions

[bit7:1] Reserved: Reserved bits

"0b0000000" is read from these bits.

Set these bits to "0b0000000" when writing.

[bit0] UPCSE: USB-PLL oscillation stabilization wait complete interrupt enable bit

Value	Description
0	Disables the interrupt [Initial value]
1	Enables the interrupt

5.9. USB-PLL Interrupt Source Status Register (UPINT_STR)

The UPINT_STR indicates the status of USB-PLL oscillation stabilization wait interrupts.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved							UPCSI
Attribute	-							R
Initial value	-							0

■ Register functions

[bit7:1] Reserved: Reserved bits

"0b0000000" is read from these bits.

Set these bits to "0b0000000" when writing.

[bit0] UPCSI: USB-PLL interrupt factor status bit

Value	Description
0	No interrupt has occurred [Initial value]
1	An interrupt has occurred

5.10. USB-PLL Interrupt Factor Clear Register (UPINT_CLR)

The UPINT_CLR is used to clear the USB-PLL interrupt factor.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved							UPCSC
Attribute	-							W
Initial value	-							0

■ Register functions

[bit7:1] Reserved: Reserved bits

"0b0000000" is read from these bits.

Set these bits to "0b0000000" when writing.

[bit0] UPCSC: USB-PLL oscillation stabilization interrupt factor clear bit

Value	Description
0	Disabled [Initial value]
1	Clears the USB-PLL oscillation stabilization wait interrupt.

<Note>

Writing "1" to this register to clear the UPINT_STR Register.

5.11. USB Enable Register (USBEN)

The USBEN enables/disables USB controller operation.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved							USBEN
Attribute	-							R/W
Initial value	-							0

■ Register functions

[bit7:1] Reserved: Reserved bits

"0b0000010" is read from these bits.

Set these bits to "0b0000010" when writing.

[bit0] USBEN: USB enable bit

Value	Description
0	Disables the USB operation (Resets the USB controller) [Initial value]
1	Enables the USB operation

<Notes>

- When using USB, set this bit to "1" previously.
- Supply at least five cycles of USB clocks to the USB controller before setting this bit to "1".

6. Usage Precautions

This section explains the precautions for using the clock generation unit.

- USB clock output setting and USB clock selection
  Do not disable the USB clock output (UCEN = 0) and select the USB clock (UCSEL) at the same time.
  Be sure to disable the USB clock output before selecting the USB clock.

- Setting the frequency division ratio of USB-PLL oscillation
  When the PLL frequency division ratio is changed after stabilization of PLL oscillation, stop the PLL oscillation once, change the frequency division ratio, and then enable the PLL oscillation again.

- Selecting CLKMO
  By writing "0" to the UCSEL bit, CLKMO is selected as the USB clock.
  The main clock should be selected when CLKMO oscillates at 48 MHz.

- Setting the PLL oscillation stabilization wait time
  Set the oscillation stabilization wait time with the PLL Oscillation Stabilization Wait Time Setting Register, and then enable PLL. Do not change the oscillation stabilization wait time while waiting for oscillation to stabilize.

- Selecting the USB-PLL input clock
  By writing "1" to the UCSEL bit, the USB-PLL oscillation clock is selected as the USB clock.
  Write "0" to the UPINC bit of the USB-PLL Control Register 1 (UPCR1), and be sure to select CLKMO as the USB-PLL input clock.

  The following Table 6-1 shows relationship between the USB clock and UCSEL/UPLLEN/UPINC.

Table 6-1 USB clock and register settings

		UCSEL	UPLLEN	UPINC
When using the 48 MHz main clock		0	0	-
When using the PLL macro oscillation clock	Main clock oscillation input	1	1	0
	Setting is prohibited.	1	1	1

- Standby mode and the USB-PLL oscillation stabilization wait counter
  If the mode changes to TIMER/RTC/STOP mode while waiting for the USB-PLL oscillation to stabilize, USB-PLL stops and the stabilization wait counter is cleared.

- Setting the USB enable bit and USB controller
  To use the USB controller, enable the USB enable bit. Supply the USB clock to the USB controller before enabling the USB enable bit. For details on USB controller settings, see Chapters "USB Function" and "USB Host".

CHAPTER 2-3: USB/Ethernet Clock Generation

This chapter explains the USB/Ethernet clock generation.

CODE: 9BFUSBETHERPLL-E02.0

1. Overview

This section explains the overview of the USB/Ethernet clock generation.

The USB clock is a 48 MHz clock used by USB macro to communicate. The Ethernet clock is a 50 MHz (RMII)/25 MHz (MII) clock used for Ethernet communication.

By using this function, a USB (48 MHz) clock and Ethernet (50 MHz/25 MHz) clock can be generated simultaneously.

The following three methods are used to generate a USB/Ethernet clock:

- Using a 48 MHz or 50 MHz/25 MHz main clock (hereafter CLKMO) without change
- Using PLL for USB/Ethernet (hereafter USB/Ethernet-PLL) as a clock source
- Using a main PLL clock (hereafter CLKPLL) as a clock source

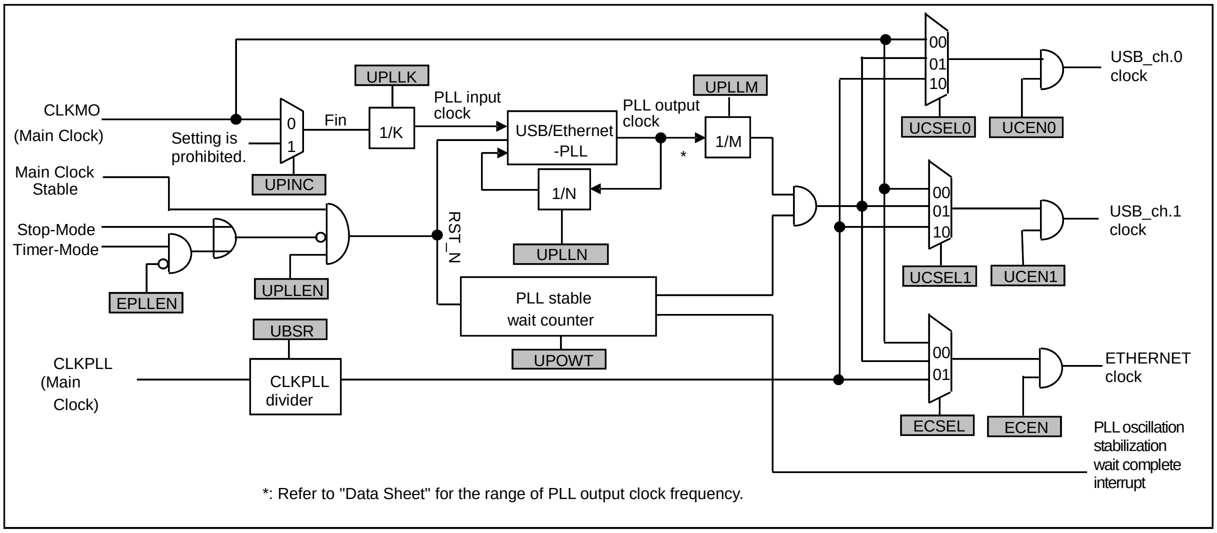
USB/Ethernet clock generation block has the following functions:

- USB/Ethernet clock output enable/disable setting
- Selection of USB/Ethernet clock
- USB/Ethernet-PLL oscillation enable/disable setting
- Selection of USB/Ethernet-PLL input clock
- USB/Ethernet-PLL input clock division setting
- USB/Ethernet-PLL output clock multiplication setting
- USB/Ethernet-PLL stabilization wait time setting
- USB/Ethernet clock stop in standby mode

2. Configuration and Block Diagram

This section describes the configuration of the USB/Ethernet clock generation block and block diagram.

Figure 2-1 Block diagram of USB/Ethernet clock generation block



■ USB/Ethernet-PLL control register (UPLLEN)

USB/Ethernet-PLL oscillation enable can be set by the control register.

■ Input clock selection register (UPINC)

CLKMO must be selected.

■ USB/Ethernet-PLL

· Division setting register (UPLLK, UPLLN, UPLLM)

To generate 48 MHz as a USB clock or 50 MHz/25 MHz as an Ethernet clock, settings of K division, N division, and M division are required.

Refer to the use conditions of "PLL input clock frequency", "PLL macro oscillation clock frequency", and "PLL multiplier" in "Data Sheet" of the product used for the specification range of input clock frequency, output clock frequency, and multiplier (N division setting value) of USB/Ethernet-PLL.

· Oscillation stabilization wait time setting register (UPOWT)

Oscillation stabilization wait time of the USB/Ethernet-PLL can be set.

■ CLKPLL input

· Division setting register (UBSR)

Division setting of CLKPLL must be executed.

■ Output clock

· Output clock selection register (UCSEL0, UCSEL1, ECSEL)

It can be selected from a CLKMO, USB/Ethernet-PLL output clock, or CLKPLL division clock.

· USB/Ethernet clock output enable register (UCEN0, UCEN1, ECEN)

USB/Ethernet clock output enable can be set.

■ Standby mode setting

- Oscillation of USB/Ethernet-PLL stops in TIMER mode or STOP mode. However, if USB/Ethernet-PLL is used as an Ethernet clock (ECSEL[1:0] = 01) and is set to EPLLEN = 1, oscillation stop of USB/Ethernet-PLL will not be executed in TIMER mode.

- The Main Clock stable signals described in Figure 2-1 are oscillation stabilization signals.

3. Description of Operation

This section explains the operation of the USB/Ethernet clock generation block.

■ USB/Ethernet clock selection

A source clock of the USB/Ethernet clock can be selected from the following two types.

● CLKMO

CLKMO can be directly used as a USB clock or Ethernet clock. In this case, CLKMO needs to be externally input in 48 MHz or 50 MHz/25 MHz, or it needs to oscillate in 48 MHz or 50 MHz/25 MHz. Also, wait for output enable of the USB clock or Ethernet clock after confirming the oscillation stabilization of CLKMO.

● USB/Ethernet-PLL output clock

The USB/Ethernet-PLL output clock can be used as the source clock of the USB/Ethernet clock.

· When used as USB clock
USB/Ethernet-PLL output clock must be output in 240 MHz or 288 MHz to generate a 48 MHz clock by M division.

· When used as an Ethernet clock
The USB/Ethernet-PLL output clock must be output from 200 MHz to 300 MHz to generate a 50 MHz clock or 25 MHz clock by M division.

<Note>

If it is used as an Ethernet clock, the output clock of USB/Ethernet-PLL must not be divided by three (UPLLM = 0b0010) due to the specification restriction of Ethernet communication clock duty.

Table 3-1 shows the setting example of the PLL division ratio.

Table 3-1 Setting example of PLL division ratio

Fin (MHz)	Ethernet clock output 50 MHz			Ethernet clock output 25 MHz			USB clock output 48 MHz		
	PLL output frequency 200 MHz			PLL output frequency 200 MHz			PLL output frequency 240 MHz		
	K	N	M	K	N	M	K	N	M
4	1	50	4	1	50	8	1	60	5
8	1	25	4	1	25	8	1	30	5
16	2	25	4	2	25	8	1	15	5
24	3	25	4	6	50	8	2	20	5
25	5	40	4	*			5	48	5
48	6	25	4	6	25	8	*		
50	*			5	20	8	10	48	5

*: Use CLKMO directly as a USB clock or Ethernet clock without using USB/Ethernet-PLL.

● CLKPLL

CLKPLL can be divided to be used as a USB clock or Ethernet clock if needed.

<Note>

If this clock generation block is used as an Ethernet clock, CLKPLL must not be divided by three (UBSR = 0b0010) due to the specification restriction of Ethernet communication clock duty.

■ Transition to standby mode

● When executing a transition to standby mode

Before executing a transition to standby mode (STOP mode or TIMER mode), set "0" to all UCEN0, UCEN1, and ECEN bits of UCCR register to stop supplying the USB clock and Ethernet clock.

1. Set UCCR:UCEN = 0, UCCR:UCEN1 = 0, and UCCR:ECEN = 0
2. Read UCCR register and confirm that UCEN0, UCEN1, and ECEN bits are "0".
3. Transition to the standby mode

When returning from standby mode, set UCEN0, UCEN1, and ECEN back to "1" if needed. When oscillation of the USB/Ethernet clock stabilizes, it starts supplying. Check the following to know if oscillation of the USB/Ethernet clock stabilizes.

a) When USB/Ethernet-PLL is used

Check if UPRDY = 1, or use USB/Ethernet-PLL oscillation stabilization wait interrupt.

b) When CLKMO (50 MHz/25 MHz or 48 MHz) is used

After stabilization of CLKMO oscillation, the USB/Ethernet clock is provided.

c) When CLKPLL is used

Check if SCM_STR:PLRDY = 1, or use PLL oscillation stabilization wait interrupt (see the chapter "Clock" in "PERIPHERAL MANUAL").

■ USB/Ethernet-PLL oscillation stabilization wait

● USB/Ethernet-PLL oscillation stabilization wait time setting

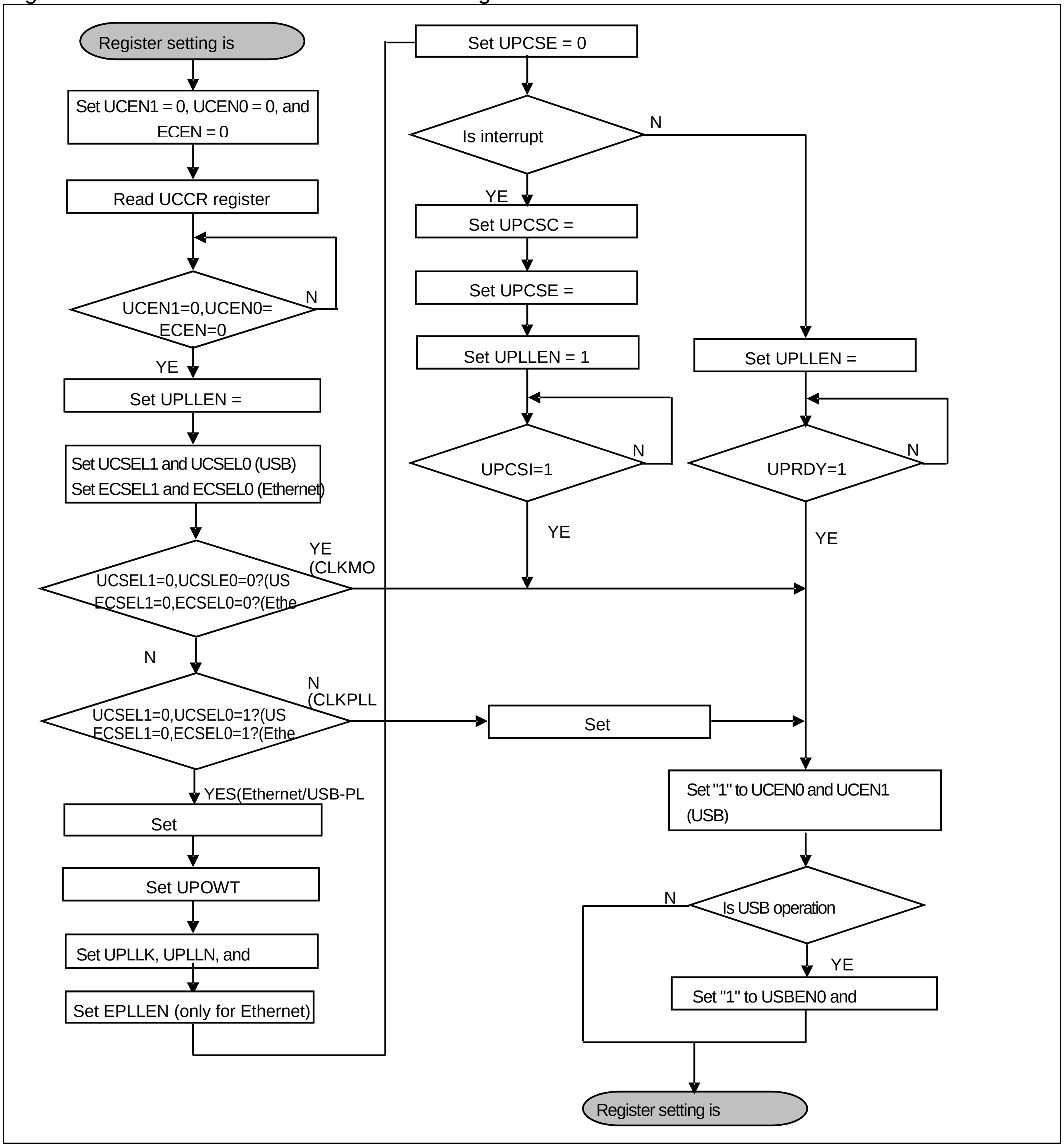
After stabilization of CLKMO oscillation, start counting USB/Ethernet-PLL oscillation stabilization wait time.
Before executing USB/Ethernet-PLL oscillation enable, set the USB/Ethernet-PLL oscillation stabilization wait time and oscillation stabilization complete interrupt. Do not change the oscillation stabilization wait time during the oscillation stabilization wait.

4. Example of Setting Procedure

This section describes the example of the setting procedure for the USB/Ethernet clock generation block.

Figure 4-1 shows the example of the setting procedure for the USB/Ethernet clock.

Figure 4-1 Procedure for USB/Ethernet clock generation



5. List of Registers

This section describes the list of registers for the USB/Ethernet clock generation block.

■ List of registers for the USB/Ethernet clock generation block

Register abbreviation	Register Name	Reference
UCCR	USB/Ethernet clock control register	5.1
UPCR1	USB/Ethernet-PLL control register1	5.2
UPCR2	USB/Ethernet-PLL control register2	5.3
UPCR3	USB/Ethernet-PLL control register3	5.4
UPCR4	USB/Ethernet-PLL control register4	5.5
UPCR5	USB/Ethernet-PLL control register5	5.6
UPCR6	USB/Ethernet-PLL control register6	5.7
UPCR7	USB/Ethernet-PLL control register7	5.8
UP_STR	USB/Ethernet-PLL state register	5.9
UPINT_ENR	USB/Ethernet-PLL interrupt factor enable register	5.10
UPINT_CLR	USB/Ethernet-PLL interrupt factor clear register	5.11
UPINT_STR	USB/Ethernet-PLL interrupt factor state register	5.12
USBEN0	USB (ch.0) enable register	5.13
USBEN1	USB (ch.1) enable register	5.14

5.1. USB/Ethernet Clock Setting Register (UCCR)

The UCCR register sets selection for the USB/Ethernet clock and output enable for the USB/Ethernet clock.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved	ECSEL1	ECSEL0	ECEN	UCEN1	UCSEL1	UCSEL0	UCEN0
Attribute	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	-	0	0	0	0	0	0	0

■ Register function

[bit7] Reserved: Reserved bit

From this bit, "0" is read.
When writing, set "0".

[bit6:5] ECSEL1/ECSEL0: Ethernet clock selection bits

Value	Description
00	CLKMO [initial value]
01	USB/Ethernet-PLL oscillation clock
10	CLKPLL division clock
11	Reserved

[bit4] ECEN: Ethernet clock output enable bit

Value	Description
0	Disable Ethernet clock output [initial value]
1	Enable Ethernet clock output

[bit3] UCEN1: USB (ch.1) clock output enable bit

Value	Description
0	Disable USB (ch.1) clock output [initial value]
1	Enable USB (ch.1) clock output

[bit2:1] UCSEL1/UCSEL0: USB clock selection bits

Value	Description
00	CLKMO [initial value]
01	USB/Ethernet-PLL oscillation clock
10	CLKPLL division clock
11	Reserved

[bit0] UCEN0: USB (ch.0) clock output enable bit

Value	Description
0	Disable USB (ch.0) clock output [initial value]
1	Enable USB (ch.0) clock output

<Notes>

· If CLKMO is selected as the USB clock in UCSEL[1:0] bits, 48 MHz input is required from the external main oscillation. Also, if it is selected as the Ethernet clock, 50 MHz or 25 MHz input is required from the external main oscillation.

· This register is not initialized in software reset.

5.2. USB/Ethernet-PLL Setting Register1 (UPCR1)

The UPCR1 register sets PLL for USB/Ethernet.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved						UPINC	UPLLEN
Attribute	-						R/W	R/W
Initial value	-						0	0

■ Register function

[bit7:2] Reserved: Reserved bits

From these bits, "0b000000" is read.
When writing, set "0b000000".

[bit1] UPINC: USB/Ethernet-PLL input clock selection bit

Value	Description
0	CLKMO [initial value]
1	Setting is disabled

[bit0] UPLLEN: USB/Ethernet-PLL oscillation enable bit

Value	Description
0	Stop USB/Ethernet-PLL [initial value]
1	Enable USB/Ethernet-PLL oscillation

<Notes>

- "0" must be set in UPINC. If "1" is set, the operation will not be guaranteed.
- This register is not initialized in software reset.

5.3. USB/Ethernet-PLL Setting Register2 (UPCR2)

The UPCR2 register sets the oscillation stabilization wait time of PLL for USB/Ethernet.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved					UPOWT		
Attribute	-					R/W		
Initial value	-					000		

■ Register function

[bit7:3] Reserved: Reserved bits

From these bits, "0b00000" is read.
When writing, set "0b00000".

[bit2:0] UPOWT: USB/Ethernet-PLL oscillation stabilization wait time setting bits

bit2	bit1	bit0	Description
0	0	0	2^{9}/Fin : Approx. 128 μs* [initial value]
0	0	1	2^{10}/Fin : Approx. 256 μs*
0	1	0	2^{11}/Fin : Approx. 512 μs*
0	1	1	2^{12}/Fin : Approx. 1.02 ms*
1	0	0	2^{13}/Fin : Approx. 2.05 ms*
1	0	1	2^{14}/Fin : Approx. 4.10 ms*
1	1	0	2^{15}/Fin : Approx. 8.20 ms*
1	1	1	2^{16}/Fin : Approx. 16.4 ms*

*: When Fin = 4 MHz.

<Notes>

- Fin is the clock selected in UPINC.
- This register is not initialized in software reset.
- Since the oscillation stabilization wait time for PLL macro differs by products, refer to the use conditions of "PLL oscillation stabilization wait time" in "Data Sheet" of the product used.

5.4. USB/Ethernet-PLL Setting Register3 (UPCR3)

The UPCR3 register sets the division ratio (K) of PLL for Ethernet/USB.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved			UPLLK				
Attribute	-			R/W				
Initial value	-			00000				

■ Register function

[bit7:5] Reserved: Reserved bits

From these bits, "0b000" is read.
When writing, set "0b000".

[bit4:0] UPLLK: USB/Ethernet-PLL clock division ratio (K) setting bits

Value	Description
00000	Divided by (UPLLK + 1). The division ratio of 1 to 32 can be set by using the UPLIK value. (example) UPLLK = "00000" ⇒ 1 division [initial value]
00001	
.	
.	
11111	

<Note>

This register is not initialized in software reset.

5.5. USB/Ethernet-PLL Setting Register4 (UPCR4)

The UPCR4 register sets the division ratio (N) of PLL for USB/Ethernet.

● Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved	UPLLN						
Attribute	-	R/W						
Initial value	-	0111011						

● Register function

[bit7] Reserved: Reserved bit

From this bit, "0" is read.
When writing, set "0".

[bit6:0] UPLLN: USB/Ethernet-PLL clock division ratio (N) setting bits

Value	Description
0000000	Setting is prohibited.
.	
0001100	
0001101	Divided by (UPLLN + 1). The division ratio of 14 to 100 can be set by using the UPLLN value. (example) UPLLN = "0111011" ⇒ 60 division [initial value]
.	
.	
1100011	
1100100	Setting is prohibited.
.	
1111111	

<Note>

This register is not initialized in software reset.

5.6. USB/Ethernet-PLL Setting Register5 (UPCR5)

The UPCR5 register sets the division ratio (M) of PLL for USB/Ethernet.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved				UPLLM			
Attribute	-				R/W			
Initial value	-				0100			

■ Register function

[bit7:4] Reserved: Reserved bits

From these bits, "0b0000" is read.
When writing, set "0b0000".

[bit3:0] UPLLM: USB/Ethernet-PLL clock division ratio (M) setting bits

Value	Description
0000	Divided by (UPLLM + 1). The division ratio of 1 to 16 can be set by using the UPLLM value. (example) UPLLM = "0100" ⇒ 5 division [initial value]
0001	
.	
.	
1111	

<Note>

This register is not initialized in software reset.

5.7. USB/Ethernet-PLL Setting Register6 (UPCR6)

The UPCR6 register sets the division ratio of CLKPLL.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved				UBSR			
Attribute	-				R/W			
Initial value	-				0010			

■ Register function

[bit7:4] Reserved: Reserved bits

From these bits, "0b0000" is read.
When writing, set "0b0000".

[bit3:0] UBSR: CLKPLL division ratio setting bits

Value	Description
0000	Divided by (UBSR + 1). The division ratio of 1 to 16 can be set by using the USBR value. (example) UBSR = "0010" ⇒ 3 division [initial value]
0001	
.	
.	
1111	

<Note>

This register is not initialized in software reset.

5.8. USB/Ethernet-PLL Setting Register7 (UPCR7)

The UPCR7 register controls USB/Ethernet-PLL in TIMER mode.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved							EPLLEN
Attribute	-							R/W
Initial value	-							0

■ Register function

[bit7:1] Reserved: Reserved bits

From these bits, "0b0000000" is read.
When writing, set "0b0000000".

[bit0] EPLLEN: USB/Ethernet-PLL control bit in Timer mode

Value	Description
0	Stop USB/Ethernet-PLL in TIMER mode.
1	Does not stop USB/Ethernet-PLL in TIMER mode.

<Note>

This register is not initialized in software reset.

5.9. USB/Ethernet-PLL State Register (UP_STR)

The UP_STR register indicates the state of USB/Ethernet-PLL.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved							UPRDY
Attribute	-							R
Initial value	-							0

■ Register function

[bit7:1] Reserved: Reserved bits

From these bits, "0b0000000" is read.
When writing, set "0b0000000".

[bit0] UPRDY: USB/Ethernet-PLL oscillation stabilization bit

Value	Description
0	Stabilization wait or oscillation stop state [initial value]
1	Stabilization state

<Note>

This register is not initialized in software reset.

5.10. USB/Ethernet-PLL Interrupt Factor Enable Register (UPINT_ENR)

The UPINT_ENR register sets enable/disable of USB/Ethernet-PLL oscillation stabilization wait complete interrupt.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved							UPCSE
Attribute	-							R/W
Initial value	-							0

■ Register function

[bit7:1] Reserved: Reserved bits

From these bits, "0b0000000" is read.
When writing, set "0b0000000".

[bit0] UPCSE: USB/Ethernet-PLL oscillation stabilization wait complete interrupt enable bit

Value	Description
0	Disable generation of interrupt [initial value]
1	Enable generation of interrupt

5.11. USB/Ethernet-PLL Interrupt Factor State Register (UPINT_STR)

The UPINT_ENR register indicates the state of USB/Ethernet-PLL oscillation stabilization wait interrupt.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved							UPCSI
Attribute	-							R
Initial value	-							0

■ Register function

[bit7:1] Reserved: Reserved bits

From these bits, "0b0000000" is read.
When writing, set "0b0000000".

[bit0] UPCSI: USB/Ethernet-PLL interrupt factor state bit

Value	Description
0	Interrupt does not occur [initial value]
1	Interrupt occurs

5.12. USB/Ethernet-PLL Interrupt Factor Clear Register (UPINT_CLR)

The UPINT_ENR register sets USB/Ethernet-PLL interrupt factor clear.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved							UPCSC
Attribute	-							W
Initial value	-							0

■ Register function

[bit7:1] Reserved: Reserved bits

From these bits, "0b0000000" is read.
When writing, set "0b0000000".

[bit0] UPCSC: USB/Ethernet-PLL oscillation stabilization interrupt generation factor clear bit

Value	Description
0	Null [initial value]
1	Clear the USB/Ethernet-PLL oscillation stabilization wait interrupt

<Note>

If this register is written and cleared, the UPINT_STR register will be cleared.

5.13. USB (ch.0) Enable Register (USBEN0)

The USBEN register sets operation enable of USB (ch.0) controller.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved							USBEN0
Attribute	-							R/W
Initial value	-							0

■ Register function

[bit7:1] Reserved: Reserved bits

From these bits, "0b0000010" is read.
When writing, set "0b0000010".

[bit0] USBEN0: USB (ch.0) enable bit

Value	Description
0	Disable USB (ch.0) operation (reset USB controller block) [initial value]
1	Enable USB (ch.0) operation

<Notes>

- To use a USB (ch.0), firstly set "1" to this bit.
- Set "1" after supplying more than 5 cycles of the USB clock to the USB controller.

5.14. USB (ch.1) Enable Register (USBEN1)

The USBEN register sets operation enable of USB (ch.1) controller.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved							USBEN1
Attribute	-							R/W
Initial value	-							0

■ Register function

[bit7:1] Reserved: Reserved bits

From these bits, "0b0000010" is read.
When writing, set "0b0000010".

[bit0] USBEN1: USB (ch.1) enable bit

Value	Description
0	Disable USB (ch.1) operation (reset USB controller block) [initial value]
1	Enable USB (ch.1) operation

<Notes>

- To use a USB (ch.1), firstly set "1" to this bit.
- Set "1" after supplying more than 5 cycles of the USB clock to the USB controller.

6. Usage Precautions

This section describes precautions for the clock generation block.

- USB clock output setting and selection of USB clock
  Do not execute USB (ch.0) clock output disable (UCEN0 = 0) and USB clock selection (UCSEL0, UCSEL1), or USB (ch.1) clock output disable (UCEN1 = 0) and USB clock selection (UCSEL0, UCSEL1) simultaneously. Make sure to execute first USB clock output disable and then USB clock selection.

- Division ratio setting of USB/Ethernet-PLL oscillation
  To change PLL division ratio after stabilization of PLL oscillation, firstly stop PLL oscillation. After changing the division ratio, enable PLL oscillation again.

- Selection of CLKMO
  If UCSEL0 = 0 and UCSEL1 = 0 are set, CLKMO will be selected for the USB/Ethernet clock. Select CLKMO only when CLKMO is oscillating at 48 MHz (used in USB) or 50 MHz/25 MHz (used in Ethernet).

- USB/Ethernet-PLL oscillation stabilization wait time setting
  Enable PLL after setting the oscillation stabilization wait time in the PLL oscillation stabilization wait time setting register. Do not change the oscillation stabilization wait time while in oscillation stabilization wait.

- Selection of USB/Ethernet-PLL input clock
  A source clock of the USB clock and Ethernet clock can be selected by UCSEL0 and UCSEL1 settings and ECSEL0 and ECSEL1 settings. Also, a separate source clock can be specified for the USB clock and Ethernet clock.

  Table 6-1 shows the setting values of registers related to source clock selection.

Table 6-1 List of register settings for each USB/Ethernet clock source selection

USB clock source	CLKMO (48 MHz)		USB/Ethernet-PLL output clock		CLKPLL	
Ethernet clock source	USB/Ethernet-PLL output clock	CLKPLL	CLKMO (50 MHz/ 25 MHz)	CLKPLL	CLKMO (50MHz/ 25 MHz)	USB/Ethernet-PLL output clock
Setting value	UCSEL1 = 0 UCSEL0 = 0 ECSEL1 = 0 ECSEL0 = 1 UPLLEN = 1	UCSEL1 = 0 UCSEL0 = 0 ECSEL1 = 1 ECSEL0 = 0 UPLLEN = 1	UCSEL1 = 0 UCSEL0 = 1 ECSEL1 = 0 ECSEL0 = 0 UPLLEN = 1	UCSEL1 = 0 UCSEL0 = 1 ECSEL1 = 1 ECSEL0 = 0 UPLLEN = 1	UCSEL1 = 1 UCSEL0 = 0 ECSEL1 = 0 ECSEL0 = 0 UPLLEN = 1	UCSEL1 = 1 UCSEL0 = 0 ECSEL1 = 0 ECSEL0 = 1 UPLLEN = 1

- Standby mode and USB/Ethernet-PLL oscillation stabilization wait counter
  By executing a transition to TIMER/STOP mode during USB/Ethernet-PLL oscillation stabilization wait time, PLL stops and the stabilization wait counter is cleared (except for TIMER mode when EPLLEN = 1 and ECSEL[1:0] = 01).

- Settings of USB enable bit and USB controller
  When using the USB controller, enable USB enable bit (USBEN). Also, enable USB enable bit (USBEN) after supplying the USB clock to the USB controller. As for the details on the USB controller setting, see the chapters "USB Function" and "USB Host".

CHAPTER 3-1: USB Device (USB Function)

This chapter explains the USB device.

1. Overview of USB Device
2. Configuration of USB Device
3. Operations of USB Device
4. Examples of USB Device Setting Procedures
5. USB Device Registers

CODE: FW03F-E19.5

1. Overview of USB Device

The USB device is an interface supporting the USB (Universal Serial Bus) communication protocol. It supports full-speed transfer mode (12 Mbps), and has the following features.

1.1. Features of USB device

- Full-speed (12 Mbps) transfer supported.
- Auto answered device status.
- Automatic generation and check of bit stripping, bit stuffing, CRC5, and CRC16.
- Toggle check by data synchronization bit.
- Auto-answer to all standard commands other than the Get/SetDescriptor and SynchFrame commands (these three commands can be processed similarly as class vendor commands).
- The class vendor commands can be received as data and responded by firmware.
- Up to 6 Endpoints supported. (Endpoint 0 is fixed to control transfer)
- Each Endpoint includes 2 buffers for data transfer.
  (Endpoint 0 includes each buffer exclusively for IN and OUT directions)
- Automatic data transfer via DMA supported (except Endpoint 0 buffers).

<Note>

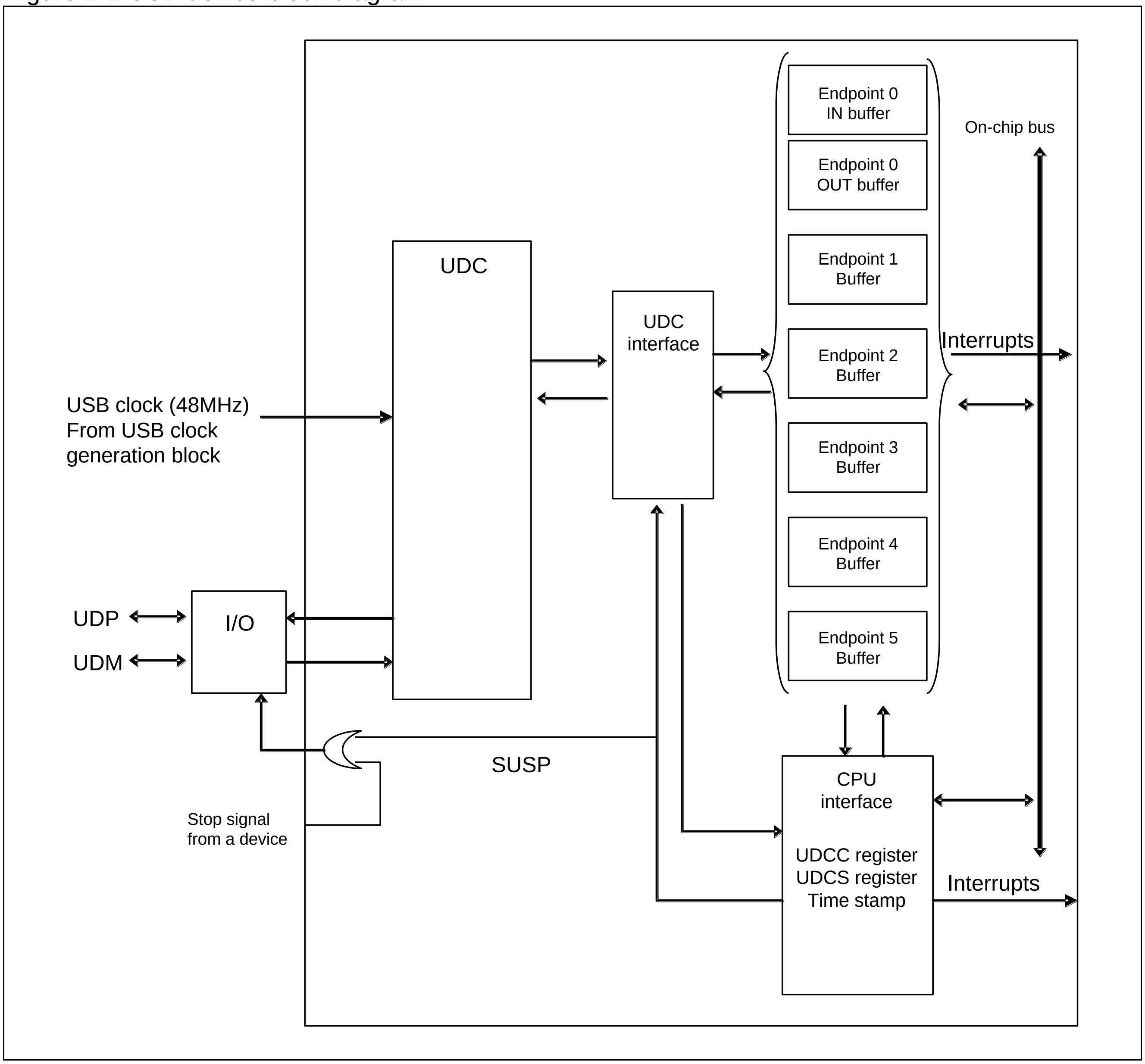
Set the base clock (HCLK) to 13 MHz or higher when using the USB device.

2. Configuration of USB Device

Figure 2-1 shows the block diagram of the USB device.

■ USB device block diagram

Figure 2-1 USB device block diagram



■ Configuration of endpoint for USB device

Configuration combination	Configuration	Interface	Alternate	Endpoint	Type
Comb1	-	-	-	0	CTRL
	1	0	0	1	Bulk/Interrupt
		0	0	2	Bulk/Interrupt
		0	0	3	Bulk/Interrupt
		0	0	4	Bulk/Interrupt
		0	0	5	Bulk/Interrupt
Comb2	-	-	-	0	CTRL
	1	1	0	-	- (*1)
		1	1	1	ISO
		0	0	2	Bulk/Interrupt
		0	0	3	Bulk/Interrupt
		0	0	4	Bulk/Interrupt
		0	0	5	Bulk/Interrupt
Comb3	-	-	-	0	CTRL
	1	1	0	-	- (*1)
		1	1	1	ISO
		2	0	-	- (*1)
		2	1	2	ISO (*2)
		0	0	3	Bulk/Interrupt
		0	0	4	Bulk/Interrupt
		0	0	5	Bulk/Interrupt

Comb1: Configuration when ISO is not set to Types of Endpoint1 and Endpoint2
Comb2: Configuration when ISO is set to Type of Endpoint1
Comb3: Configuration when ISO is set to Types of Endpoint1 and Endpoint2

*1: When isochronous is set, the endpoint does not exist for Alternate=0.
Set "0" for the number of interface descriptor endpoints for Alternate=0.
*2: When ISO is set to Type of Endpoint2, ISO must be also set to Type of Endpoint1.

3. Operations of USB Device

The USB device supports the USB (Universal Serial Bus) communication protocol. Its hardware supports the basic protocol operation (handshake). Therefore, USB communication can be implemented by processing only transfer data.

3.1. USB device operation

To use the USB device, take the following steps for setup.

1. Configure the USB clock generation block while the USB Enable Register (USBEN) disables USB operation (USBEN = 0).
2. Enable the USB clock output.
3. Enable USB operation (USBEN = 1).

The USB device transfers packets bi-directionally to/from a host controller that supports the USB protocol. Connection with the host and devices, and configuration are enumerated. Communications are implemented subsequently in different transfer types using device drivers.

The following explains the operation of USB communication between the host and devices by taking an enumeration for example.

Behaviors of registers and USB packets are shown here to provide details of the entire process.

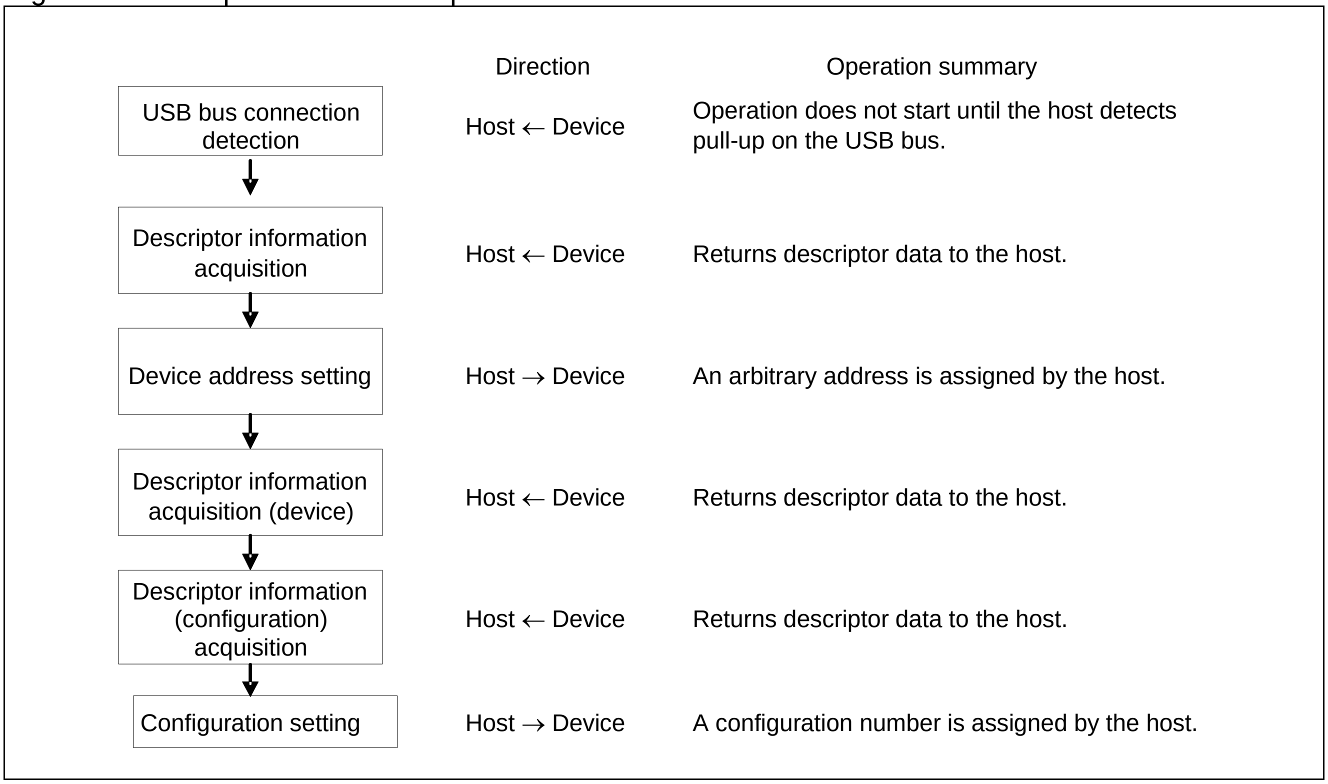
■ Enumeration

Enumeration is the first process for USB operation to establish connection between the host and devices. The host investigates what types of devices are connected on the USB bus by using USB control transfer (a USB transfer type). (Defined in the USB specification) This process uses EP0 (Endpoint 0) from the six Endpoints (as defined in the USB specification).

To use EP1 to EP5, reception and processing on the USB bus are required in the following order:

1. Resetting the USB bus
2. Setting the address by SET_Address
3. Setting configuration by SET_Config

Figure 3-1 Example of USB cable pin connection



· USB bus connection detection

The connection is reported from a device to the host.
The host monitors two signal lines (D+ and D-) on the USB bus, and finds the connection of a device if either of the signals turns to HIGH level.

For a detailed procedure explaining how to use the device in self-powered mode, see "3.2 Detection of connection and disconnection". To use the device as bus-powered, follow the procedure given in "l Initial register setting and operation start procedures".

● Initial register setting and operation start procedures

The following shows an example initial setting procedure of USB device registers.

1. Set EP0 configuration (such as packet size) by the EP0C register.
2. Set EPEN, DIR, or TYPE of each Endpoint by the EP1C to EP5C registers.
3. Clear the RST bit in the UDCC register.
4. Clear BFINI in the EP0IS, EP0OS, and EP1S to EP5S registers.
5. Clear the HCONX bit in the UDCC register.

● USB bus reset

The USB device core is initialized when the host executes a bus reset on the device, but register and buffer states are not initialized.
Take the following steps to process the device. (The process is not required in the initial bus reset after USB connection.)

1. Initialize the buffer by the BFINI bit in the EP0I Status Register (EP0IS), the BFINI bit in the EP0O Status Register (EP0OS), and the BFINI bit in the EP1 to EP5 Status Registers (EP1S to EP5S).
2. Return firmware control to the state before the enumeration.

● Descriptor acquisition

When the host requests a device, the device reports data to the host in reply to the request.
The communication is broken up into the following three stages.

Figure 3-2 Communication stages

Setup stage -> Data stage -> Status stage

The setup stage checks whether the device has received the packets from the host successfully and decodes the command. The descriptor information to be returned in the next data stage is prepared in the send buffer in this stage. The data stage checks whether the host has sent data successfully. In the status stage, the host sends a packet without data to end the transfer.

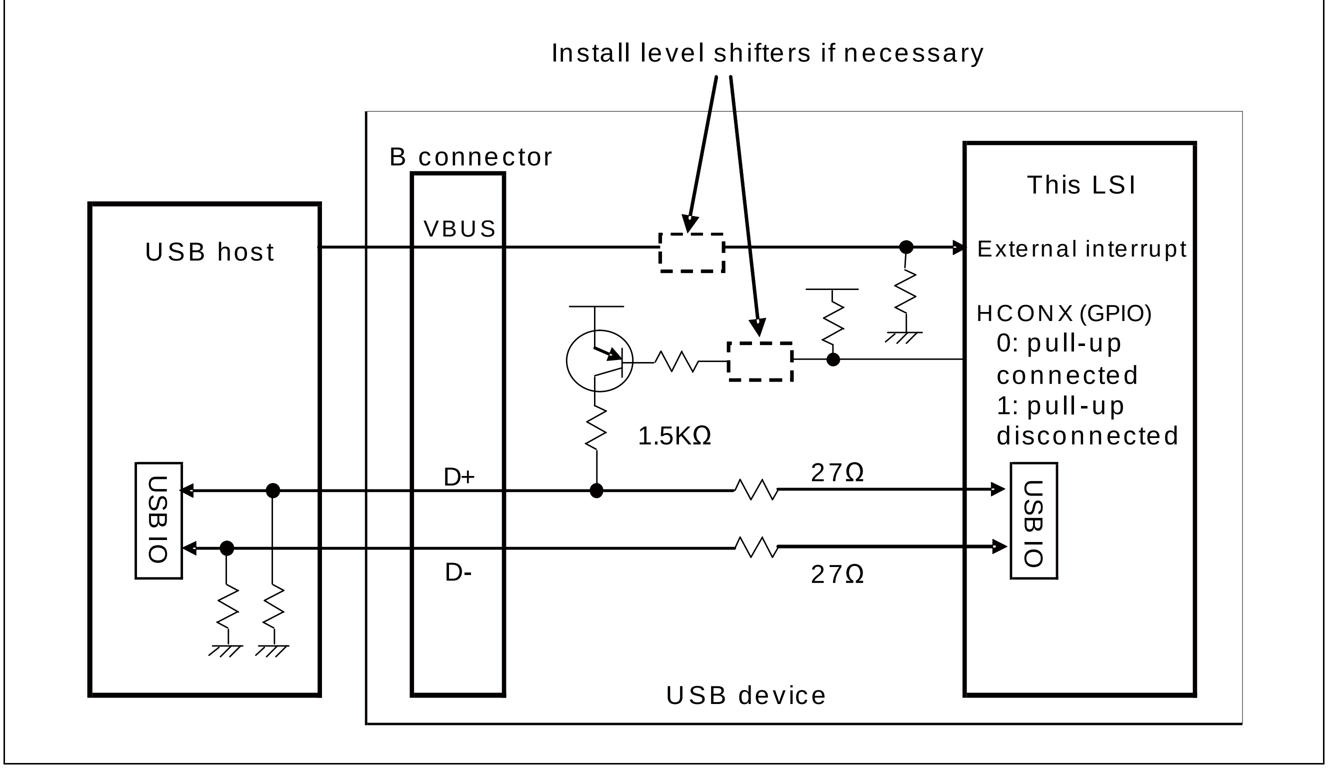
3.2. Detection of connection and disconnection

The following explains about detecting connection and disconnection to/from the USB host.

■ Example of USB system connection

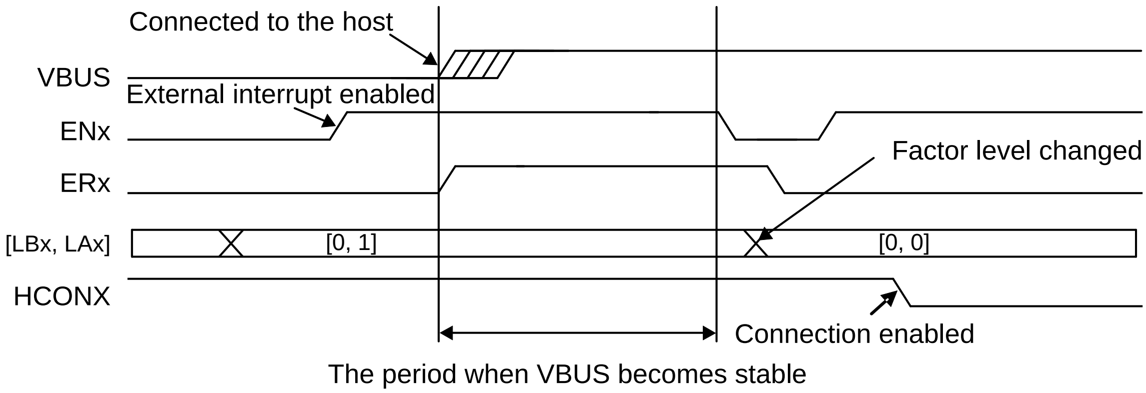
By connecting an external interrupt pin to the VBUS pin of the USB connector, and installing a pull-down resistor onto the VBUS signal, disconnection from the USB host can be detected. Figure 3-3 shows an example connection of USB connector with D+, D- and VBUS.

Figure 3-3 Example of USB system configuration



● Connection detection

Figure 3-4 Connection detecting operation



A device finds and processes the connection with the host in the following sequence:

1. The HCONX bit in the UDCC register must be set to "1". (When controlling a pull-up resistor on a general-purpose port, set the port to the pull-up resistor disconnection.)
2. Set the source level of external interrupts connected with VBUS to HIGH level detection to enable interrupts.
3. Find the USB host connection by the detection of HIGH level of the external interrupt pin, and waits for the period the VBUS becomes stable.
4. Disable external interrupts once. Change the external interrupt factor level to LOW to clear the interrupt source, and enable external interrupts again.
5. Configure the initial settings (Initialize all components including the USB device registers.)See "Initial register setting and operation start procedures" in this section.
6. Connect the pull-up resistor to D+ by clearing[*1] the HCONX bit in the UDCC register.[*2]

*1 : When control the pull-up resistor on a general-purpose port, clear the HCONX bit in the UDCC register, and set the pull-up resistor control general-purpose port to the pull-up resistor connection.

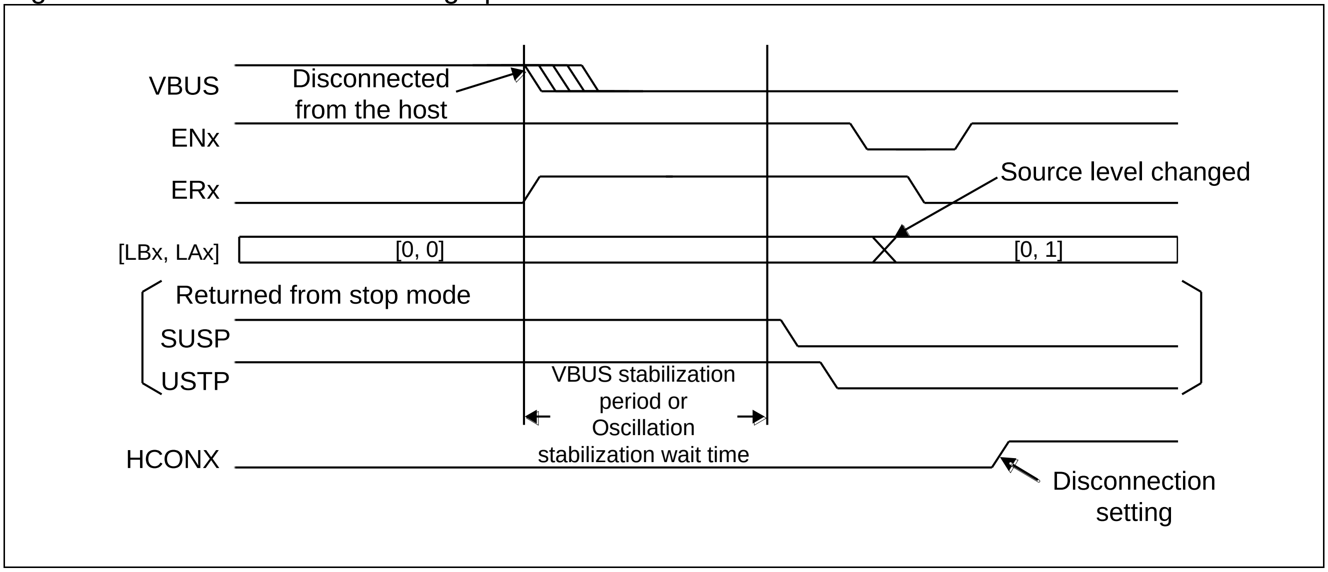
*2 : Clear the HCONX bit even if the pull-up resistor is not controlled.

<Note>

If an external noise filter is installed on the external interrupt pin, the above VBUS stabilization period does not need to be set by the program.

● Disconnection detection

Figure 3-5 Disconnection detecting operation



A device finds and processes the disconnection from the host in the following sequence:

1. Find the disconnection of the USB host by detecting LOW level of the external interrupt pin connected to VBUS.
2. When returned from stop mode or timer mode
   After the oscillation stabilization wait time, clear in the order of SUSP in the UDCS register and USTP in the UDCC register.
   In other than stop mode and timer mode wait for the period the VBUS becomes stable.
3. Disable external interrupts once. Change the external interrupt factor level to HIGH to clear the interrupt factor, and enable external interrupts again.
4. Disconnect the pull-up resistor from D+ by setting*1 the HCONX bit in the UDCC register.*2

*1: When controlling the pull-up resistor on a general-purpose port, set the HCONX bit in the UDCC register, and set the pull-up resistor control general-purpose port to the pull-up resistor disconnection.

*2: Set the HCONX bit even if the pull-up resistor is not controlled.

<Note>

If an external noise filter is installed on the external interrupt pin, the above VBUS stabilization period does not need to be set by the program.

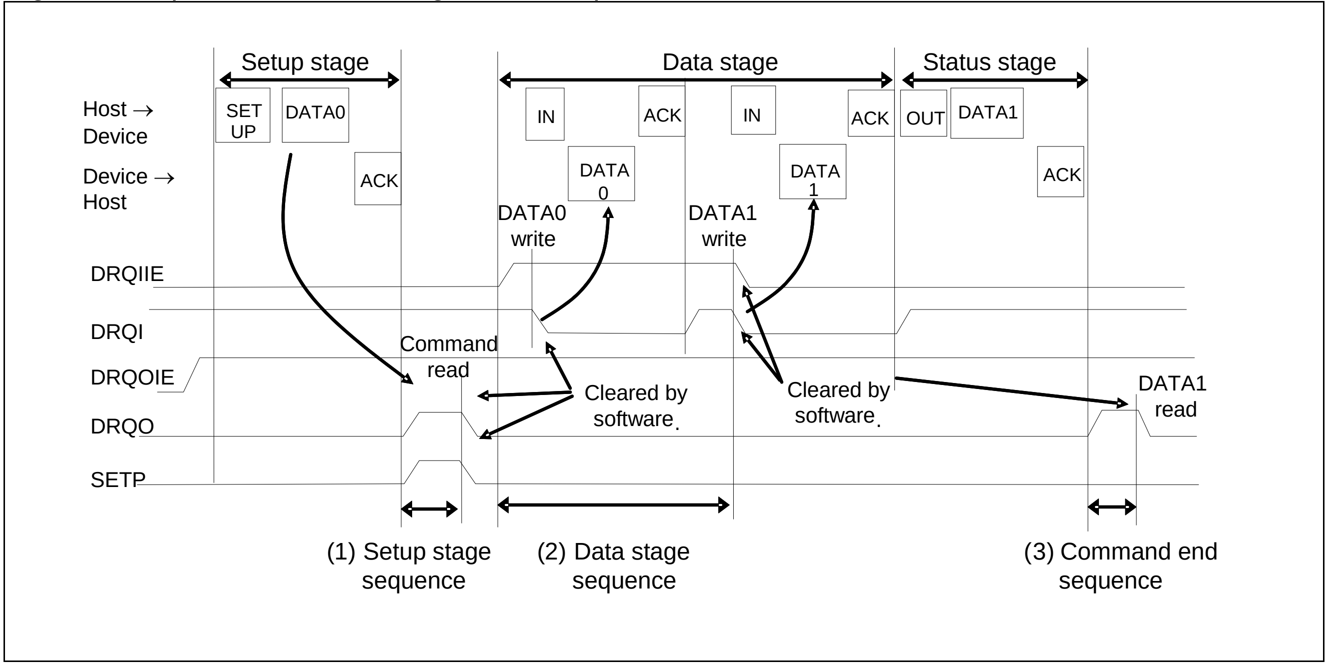
3.3. Operation of each register in response to a command

The following explains the method (architecture) to process USB packets. Responding to CPU interrupts, the firmware sequence is processed for each handshake. This is equivalent to the processing of each packet on the stage basis.

■ Operation of each register in response to a read command

The following explains the case of GetDescripter, SynchFrame, and class vendor commands.

Figure 3-6 Operation of Each Register in Response to a Read Command



(1) Setup stage sequence

Upon the receipt of the setup stage, DRQO changes to "1". Immediately when DRQO has changed, enter the CPU interrupt and check the SETP flag. If the flag is "1", read required bits of the command in the receive buffer. (Not necessarily read all the eight bytes.) Subsequently, decode the command, configure required settings, clear the SETP flag and the DRQO interrupt factor, and return.

(2) Data stage sequence

If the command decoding concludes that the data stage is in the IN direction, enable DRQIE,* and transfer outgoing data to the send buffer by the CPU interrupt. When the transfer has finished, clear the DRQI interrupt factor, and return.

*: The DRQI interrupt factor is initially set to "1", and is only used to enable interrupts.

DRQI is set when the data packet to the IN direction has finished. The CPU interrupt is entered immediately when DRQI has been set, and outgoing data is transferred to the send buffer in preparation for the next data packet. When the transfer has finished, clear the interrupt source DRQI, and return.

(3) Command end sequence

DRQO is set when the status stage to OUT direction has finished. Immediately when DRQO is set, enter the CPU interrupt and check that the number of received data units is 0. In preparation for the next setup stage, clear the interrupt factor DRQO, and return.

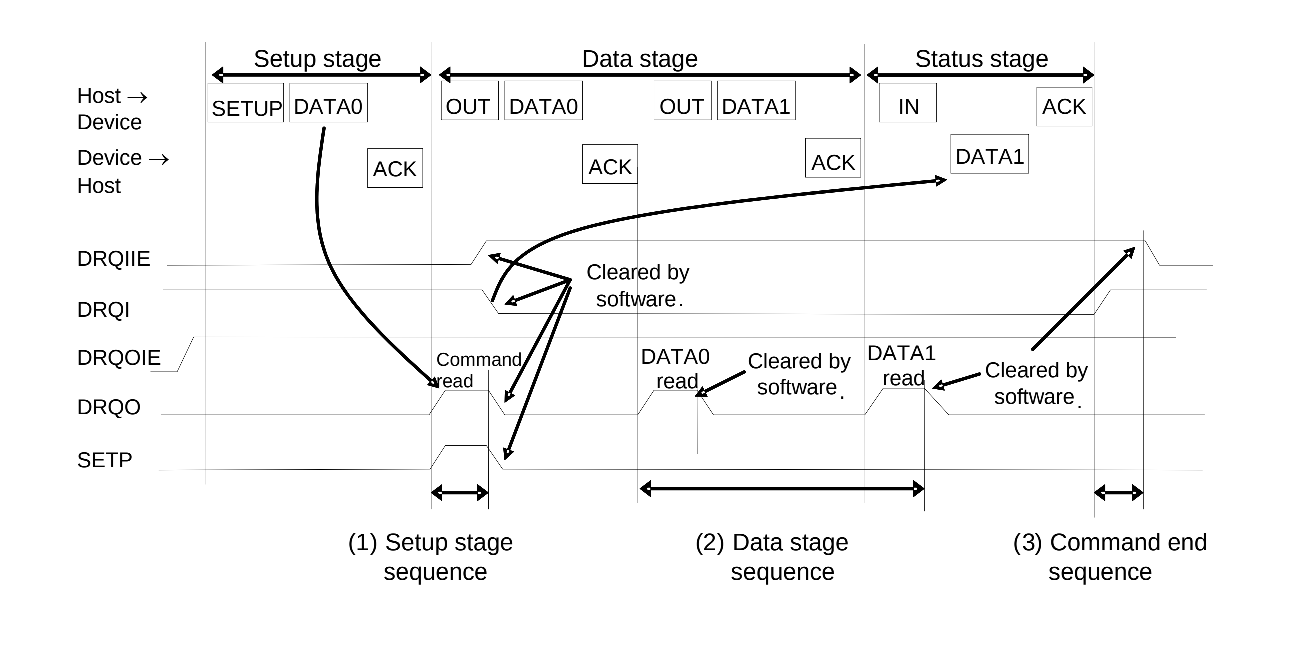
<Note>

When next setup stage is received without (3) Command end sequence being carried out due to the process of an interrupt which has higher priority than USB, the device makes no response to the next setup stage.
In order to avoid this phenomenon, carry out any of the following.
- Increase the interrupt priority of the Setup stage, Data stage and Command end sequence.
- Continue the process of the IN transfer interrupt in the Data stage sequence until DRQO is cleared in Command end sequence.

■ Operation of each register in response to a write command

The following explains the case of SetDescripter and class vendor commands.

Figure 3-7 Operation of Each Register in Response to a write Command



(1) Setup stage sequence

Upon the receipt of the setup stage, DRQO changes to "1". Immediately when DRQO has changed to "1", enter the CPU interrupt and check the SETP flag. If the flag is "1", read required bits of the command in the receive buffer. (Not necessarily read all the eight bytes.) Subsequently, decode the command, configure required settings.

In preparation of 0-byte response in the status stage, do not write data to the send buffer, and set DRQI to "0" (as the DRQI interrupt factor is initially set to "1"). Set the DRQIIE to "1" to check a successful completion of the status stage. Clear the SETP flag and the DRQO interrupt factor to return from the interrupt.

(2) Data stage sequence

DRQO is set when the data packed to OUT direction has finished. Immediately when DRQO is set, enter the CPU interrupt and check SIZE in the EP0 Status Register. Use DMA limited to received data, or use CPU read access to read data from the receive buffer. Subsequently, clear interrupt factor DRQO to return from the interrupt.

(3) Command end sequence

DRQI is set when the status stage to the IN direction has finished. Immediately when DRQI is set, enter the CPU interrupt and check that the status stage has finished successfully. Subsequently, clear interrupt factor DRQI, and return.

3.4. Suspend function

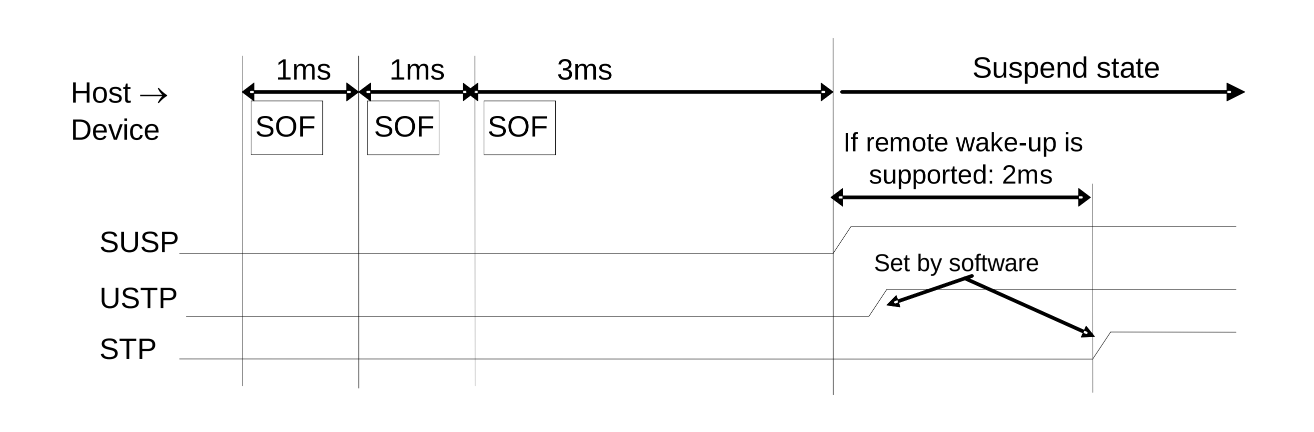
Depending on the bus power configuration, USB devices must drop the power consumption to 500 μA or less in suspend state. The following explains the sequence the USB device makes transition to suspend state, and then stop mode or timer mode.

■ Suspend sequence

When the USB device core detects a suspend state, SUSP bit in the UDCS register is enabled.

The following provides an example sequence.

Figure 3-8 Suspend operation



- Suspend sequence
  When there is a 3 ms or longer period of inactivity on the USB bus, the USB device detects a suspend state, and sets the SUSP bit interrupt factor in the UDCS register. For devices supporting remote wake-up function, the USB device waits 2 ms more * and sets stop mode or timer mode.

  *: This period is required to block remote wake-up.

<Note>

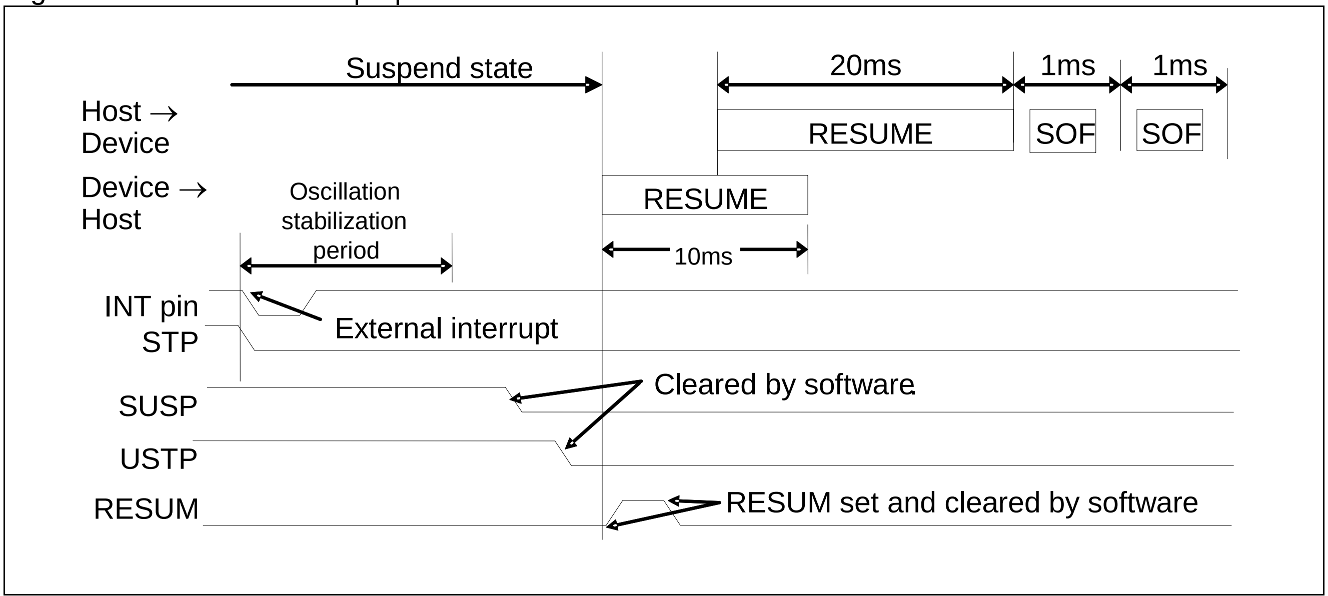
Before stop mode or timer mode is entered, set UDCIE:SUSPIE = 0 and UDCC:USTP = 1 in this order.

3.5. Wake-up function

To recover a USB device from suspend state to wake-up state, the USB protocol provides two ways.
- Remote wake-up from the device
- Wake-up from the host

■ Remote wake-up

Figure 3-9 Remote wake-up operation

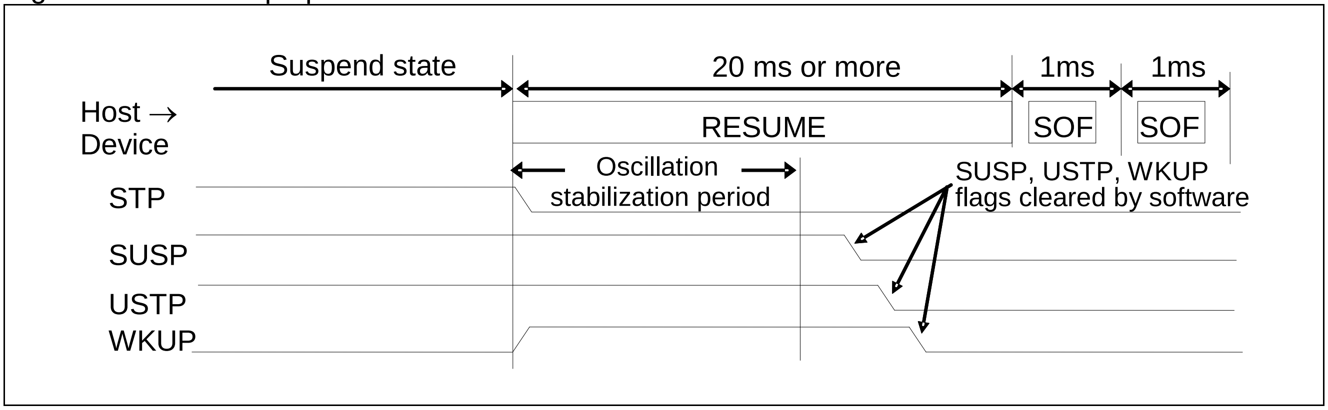


The device must be processed in the following sequence:

1. Recover the device from stop mode or timer mode by an external interrupt.
2. Check that the USB generation clock is stable.
3. Clear the SUSP bit in the UDCS register to "0".
4. Perform a dummy read from the UDCS register.
5. Clear the USTP bit of the UDCC register to "0".
6. Perform a dummy read from the UDCC register.
7. Set the RESUM bit in the UDCC register to "0".
8. Clear the RESUM bit in the UDCC register to "0".

■ Wake-up from the host

Figure 3-10 Wake-up operation from the host



Process the USB device in the following sequence.

1. Set the oscillation stabilization time so that it will not exceeds 10 ms.
2. Check that the USB clock is stable.
3. Clear SUSP bit in the UDCS register, and USTP bit in the UDCC register to "0" in this order.
4. Clear WKUP bit in the UDCS register to "0".

3.6. DMA transfer function

Data handled by the USB device can be transferred via DMA between the send/receive buffer and embedded RAM. The following two modes are available for the DMA transfer.
- Packet transfer mode, in which CPU starts DMA for each packet.
- Automatic data size transfer mode, in which DMA is automatically started for every packet.

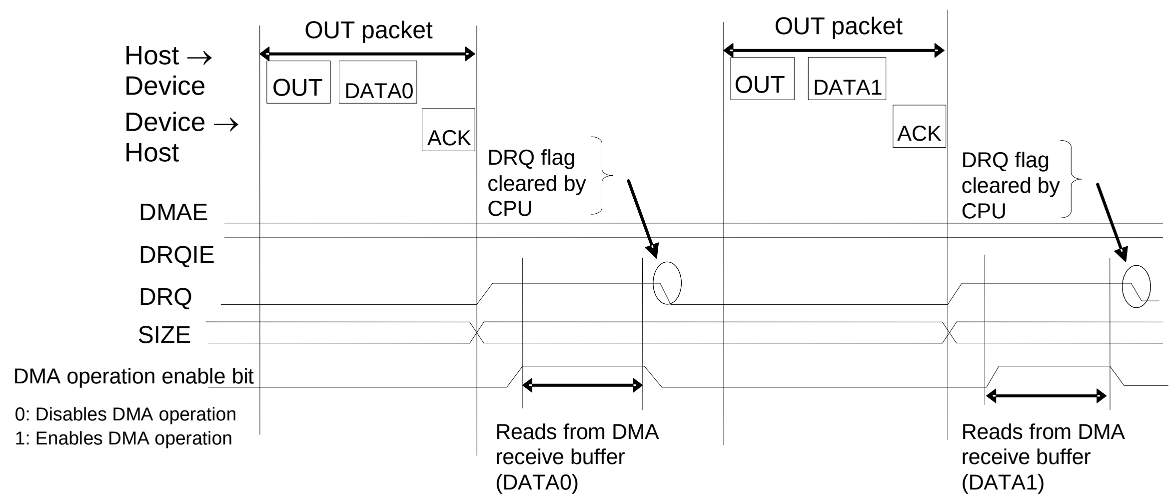
■ Packet transfer mode

The packet transfer mode transfers each packet according to the data size set in DMA and, each time the transfer of a packet finished, clears the interrupt factor (DRQ) for the next packet transfer. This transfer mode can access buffers of Endpoint 1 to Endpoint 5. Before using DMA, set the interrupt output destination by the DREQ Select Register. (Connect the interrupt output to CPU.NVIC.)

Figure 3-11 and Figure 3-12 show the timing to access buffers in each OUT direction and IN direction.

● Transfer in the OUT direction (Host -> Device)

Figure 3-11 OUT packet transfer

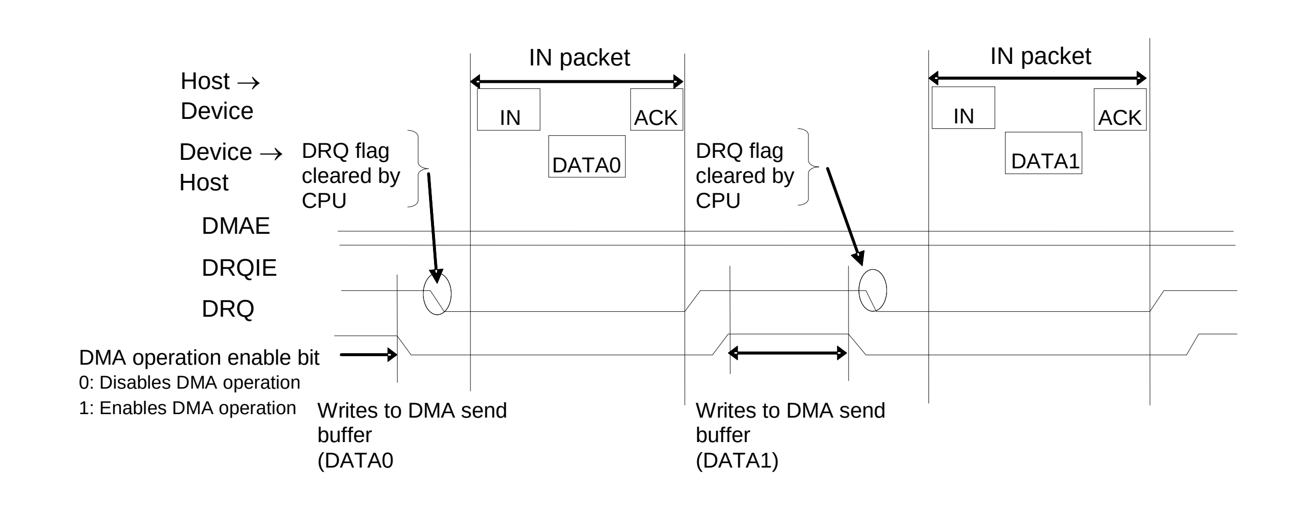


In the OUT direction transfer, the device must be processed in the following sequence:

1. Once the DRQ flag is set and the interrupt handling is entered, check the transfer data size.
2. Configure the DMA register setting relevant to the number of transfers and block size corresponding to the transfer data size, and then enable DMA to start the transfer.
3. After the transfer, clear the pertinent DRQ flag in the EP1S to EP5S registers and the pertinent interrupt factor flag in the DMAC status register, and return from the interrupt handling.

● Transfer in the IN direction (Device -> Host)

Figure 3-12 IN packet transfer



In the IN direction transfer, the device must be processed in the following sequence:

1. Once the DRQ flag is set and the interrupt handling is entered, configure the DMA register settings relevant to the number of transfers and block size corresponding to the data size to be transferred in the next IN packet, and then enable DMA to start the transfer.
2. After the DMA transfer, clear the pertinent DRQ flag in the EP1S to EP5S registers and the pertinent interrupt factor flag in the DMAC status register, and return from the interrupt handling.

■ Automatic data size transfer mode

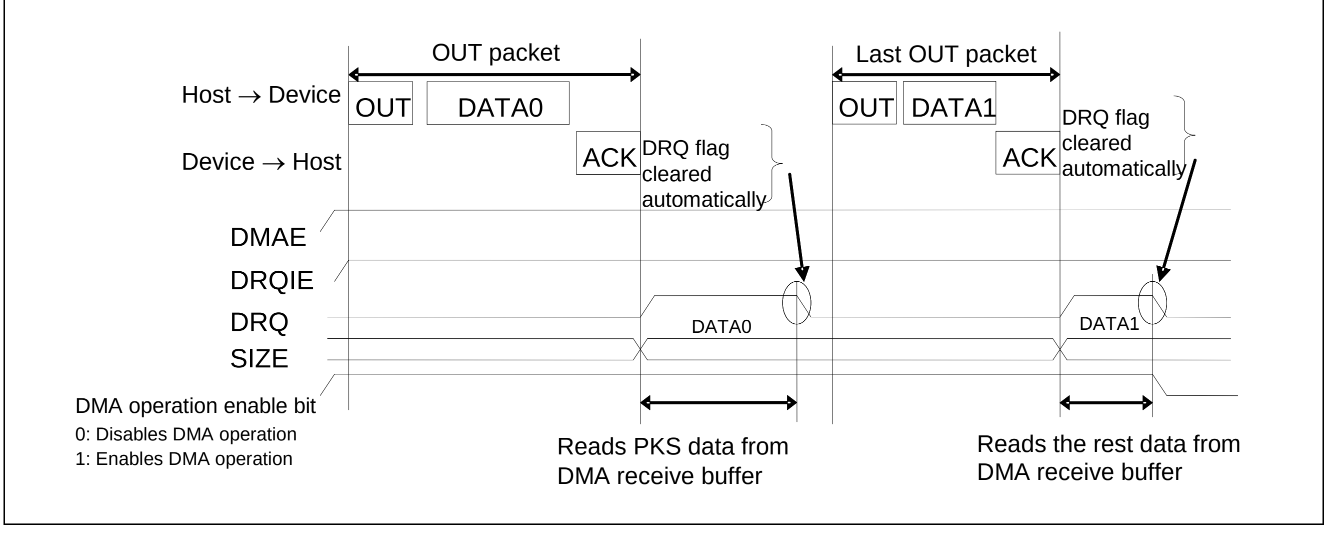
This mode can transfer even bytes. To transfer odd bytes in the OUT direction transfer, a CPU transfer sequence is required. (See Figure 3-14.) To transfer odd bytes in the IN direction transfer, see the following information.

- For TYPE0 products
  Odd bytes cannot be transferred in the IN direction transfer.
- For products other than TYPE0
  To transfer odd bytes in the IN direction transfer via DMA, set the ODDPKS register.(See chapter "Interrupts (A)".)

Before using DMA, set the interrupt output destination by the DREQ Select Register.(Connect the interrupt output to DMAC.) Configure in DMA the total data size to transfer, and also set the transfer enable bit previously. If DRQ is set after transfer from the host while DMAE is enabled, the interrupt factor (DRQ) is automatically cleared when the data size corresponding to PKS in the EP1 to EP5 Control Registers (EPxC) has been transferred. Afterward, the same sequence is repeated after transfer from the host until the transfer data size configured previously in DMA is reached. Meanwhile, configuration by the CPU is not required at all. Thus this mode can transfer data automatically by a single setting. The CPU interrupt is entered after the transfer of the last data. To perform the next transfer, therefore, reconfigure DMAC then to enable DMA and return from the interrupt. The automatic data size transfer mode uses DMAE as "1", buffer access to Endpoints 1 to 5 is only enabled. The following shows the timing to access the buffer in each of the OUT and IN directions.

● Transfer in the OUT direction (Host -> Device)

Figure 3-13 Transfer in the OUT direction (Host -> Device)



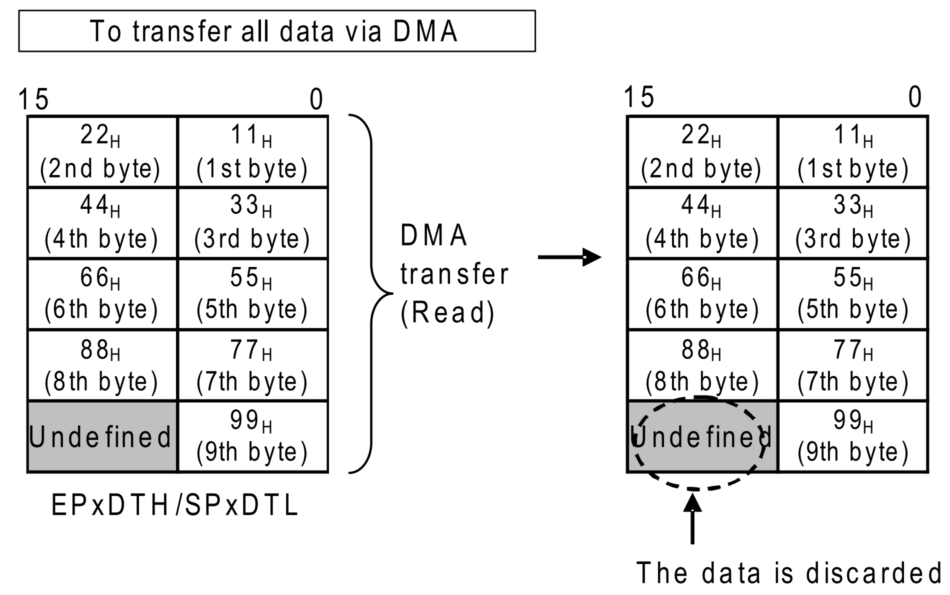
In the OUT direction transfer, the device must be processed in the following sequence:

1. Configure the DMA register setting relevant to the number of transfers and block size corresponding to the total data size, and then enable DMA to start the transfer.
2. Enable DMAE and DRQIE.
3. After the transfer, reconfigure the DMAC using an interrupt generated by the interrupt factor pertinent to the DMAC status register, and clear the flag to return from the interrupt handling.

To transfer the data size corresponding to the odd bytes via DMA, the following methods are available:

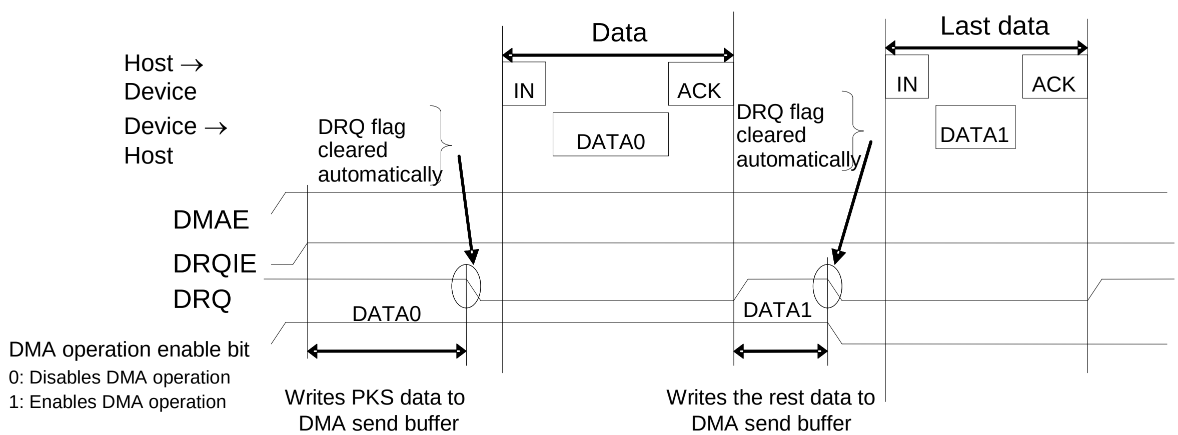
- Transfer all the data + 1 byte via DMA, and discard the last data after an endian conversion.

Figure 3-14 Example odd bytes transfer in the OUT direction



● Transfer in the IN direction (Device -> Host)

Figure 3-15 Transfer in the IN direction (Device -> Host)



In the IN direction transfer, the device must be processed in the following sequence:

1. Configure the DMA register setting relevant to the number of transfers and block size corresponding to the total data size, and then enable DMA to start the transfer.
2. Enable DMAE and DRQIE.
3. After the transfer, reconfigure the DMAC using an interrupt generated by the interrupt factor pertinent to the DMAC status register, and clear the flag to return from the interrupt handling.

3.7. NULL transfer function

If data sent from the USB device is the last packet and satisfies the maximum packet size, then the 0-byte can be automatically transferred via the next packet transfer. DMAE must be enabled to use this function. This function is valid only in IN transfer.

■ NULL transfer mode

NULL transfer mode sends 0-byte in reply to the next host's data request in the IN direction after the last data in the IN direction has been transferred.

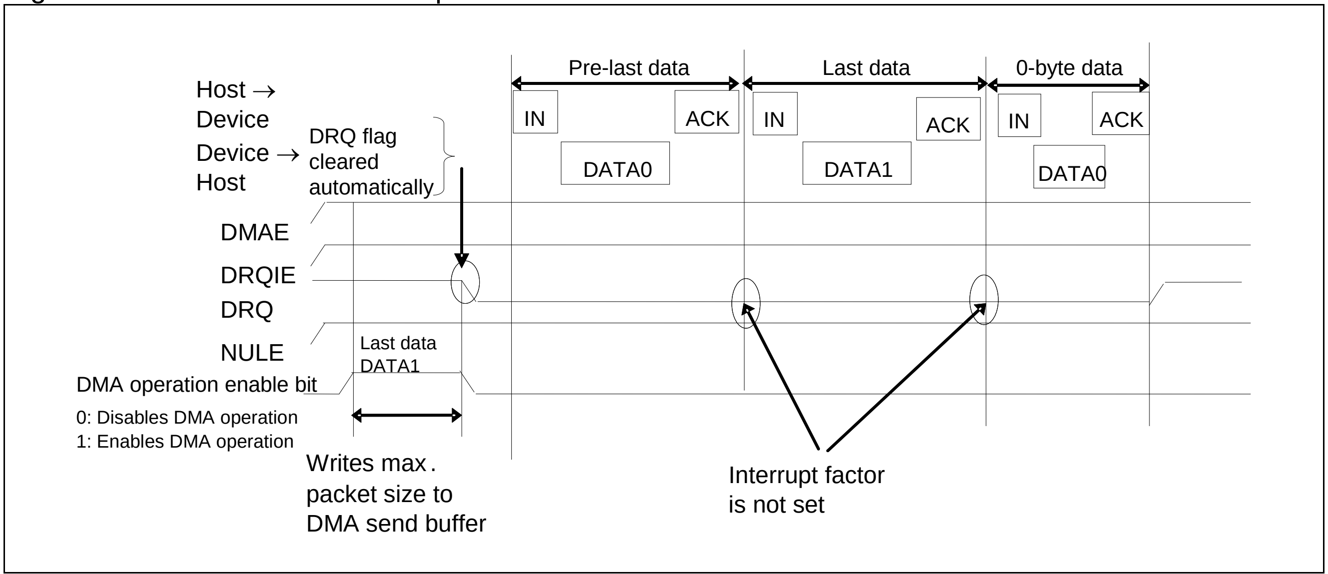
NULL transfer mode works when the following conditions are met:

- Automatic buffer transfer mode is set (DMAE = 1)
- The last data transfer writes the maximum packet size to the DMA buffer
- DMA data units are counted as 0 by writing the last data

After the last data has been written to buffer via DMA, the DRQ interrupt flag is not set until the 0-byte data is read from the host. The following shows the timing to access the buffer.

Only the transfer in the IN direction (Device -> Host) is explained.

Figure 3-16 NULL data transfer operation



The device must be processed as follows:

1. Enable EPxC:DMAE, EPxS:DRQIE, and EPxC:NULE bits by setting to "1".

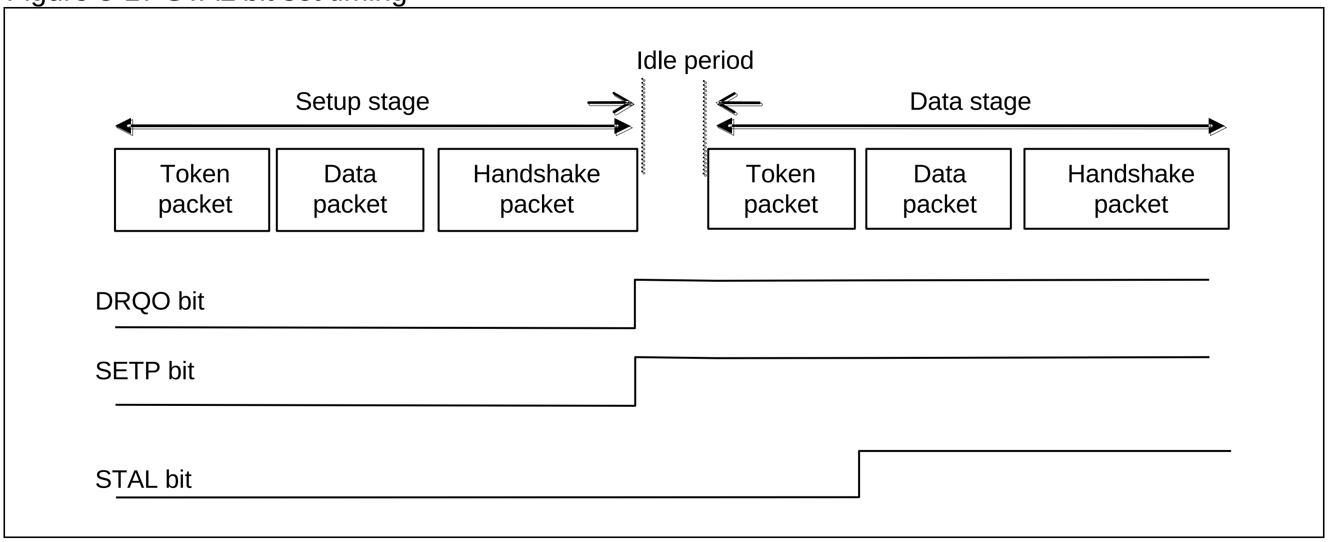
3.8. STALL response/release of Endpoint 0

The STAL bit in the EP0 Control Register (EP0C) controls the STALL response and release of Endpoint 0.

■ STAL bit set timing

To perform the STALL response, interpret the command at the setup stage (SETP = 1 detection) of control transfer. If the STALL response is required, set the STAL bit. (See Figure 3-17) After setting the STAL bit, clear the interrupt factor (DRQO bit).

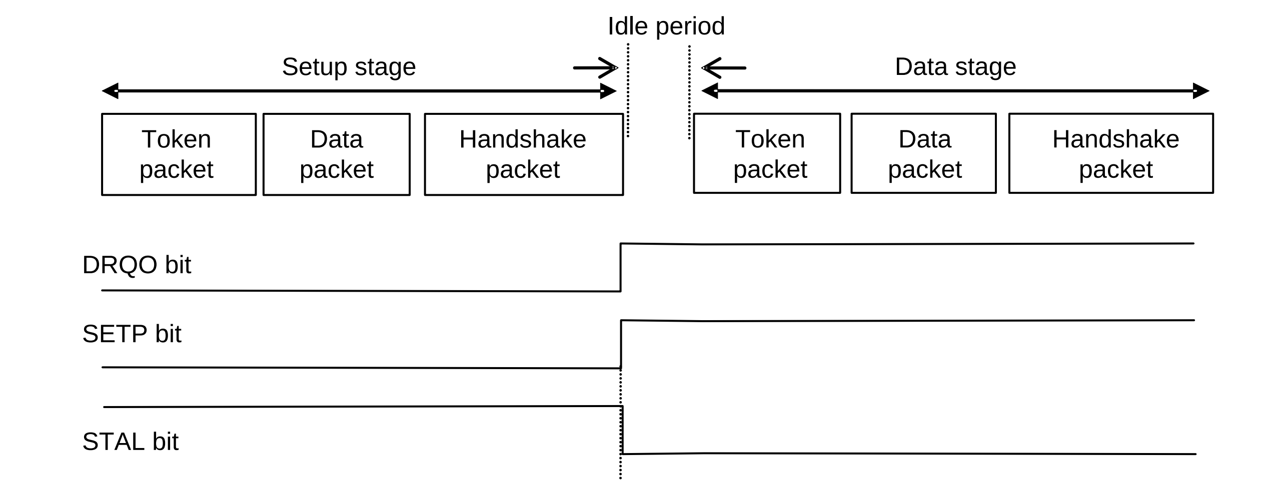
Figure 3-17 STAL bit set timing



■ STAL bit clear timing

Upon the detection of SETP = 1, pointing to the setup stage of control transfer, the STAL bit is automatically cleared and the STALL state is released. (See Figure 3-18)

Figure 3-18 STAL bit clear timing



<Note>

Upon the detection of SETP = 1 (DRQO = 1 interrupt), the STAL bit is cleared to "0". To enable the STALL response again, set the STAL bit to "1".

3.9. STALL response/release of Endpoint 1 to Endpoint 5

The STAL bit and the internal status bit in the EP1 to EP5 Control Registers (EP1C to EP5C) controls the STALL response and release of Endpoints 1 to 5.

■ STALL response processed by software

Figure 3-19 and Figure 3-20 show the procedures to process the STALL response by software. To perform the STALL response, configure the STAL bit of relevant Endpoint by software. The internal status bit does not change then.

When a transaction occurs from the host to the Endpoint to which the STAL bit is set, the hardware automatically sets the internal status bit of the relevant Endpoint to perform the STALL response to the host. Once the internal status bit is set, it remains set even when the STAL bit cleared. As the internal state bit remains set until the host issues the Clear Feature command, the STALL response remains running. While the STALCLREN bit of the UDC Control Register (UDCC) is set to "0", the STALL response also remains running in the following condition:

The STAL bit remains set even after the internal status bit is cleared by the Clear Feature command.

This is because the internal status bit is set each time a transaction occurs to the relevant Endpoint. To release the STALL response, therefore, the STAL bit must be cleared, and the internal status bit must be cleared by the Clear Feature command. If the STALCLREN bit in the UDC Control Register (UDCC) is set to "1", the STAL bit is cleared at the same time the internal status bit is cleared by the Clear Feature command, and the STALL response is not performed for the next transaction.

Figure 3-19 To process the STALL response by software (the STAL bit is cleared by software) UDCC.STALCLREN=0

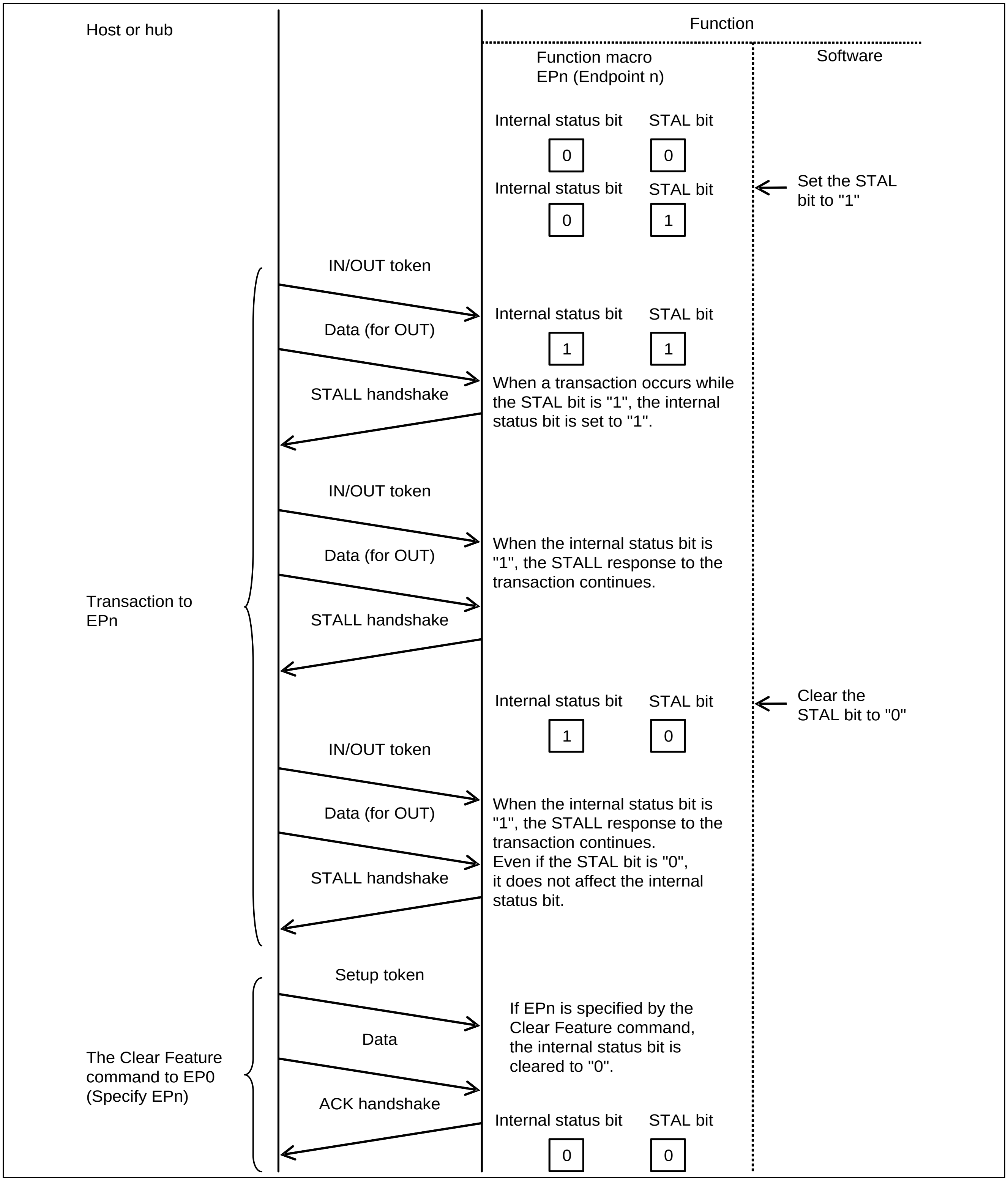
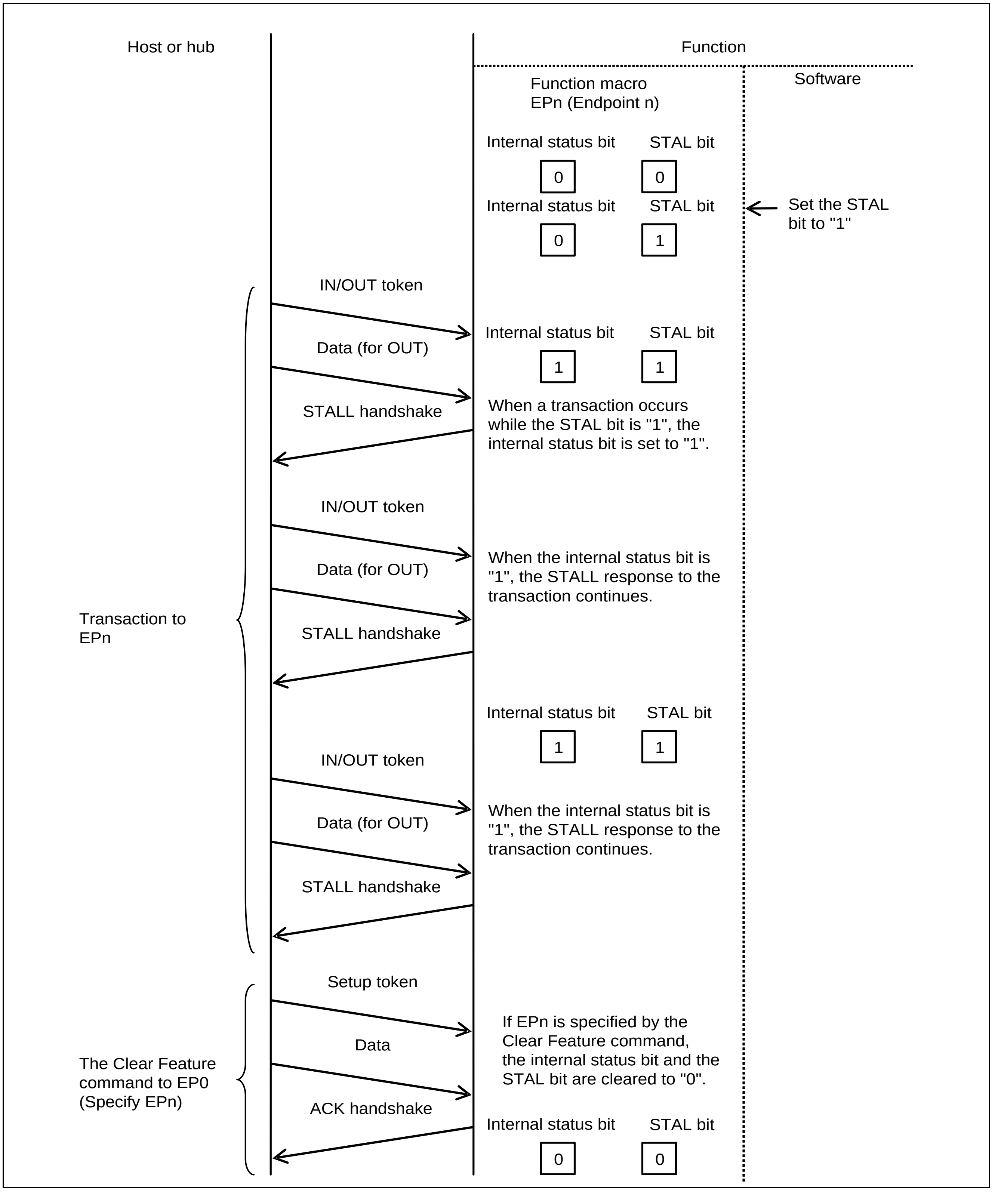


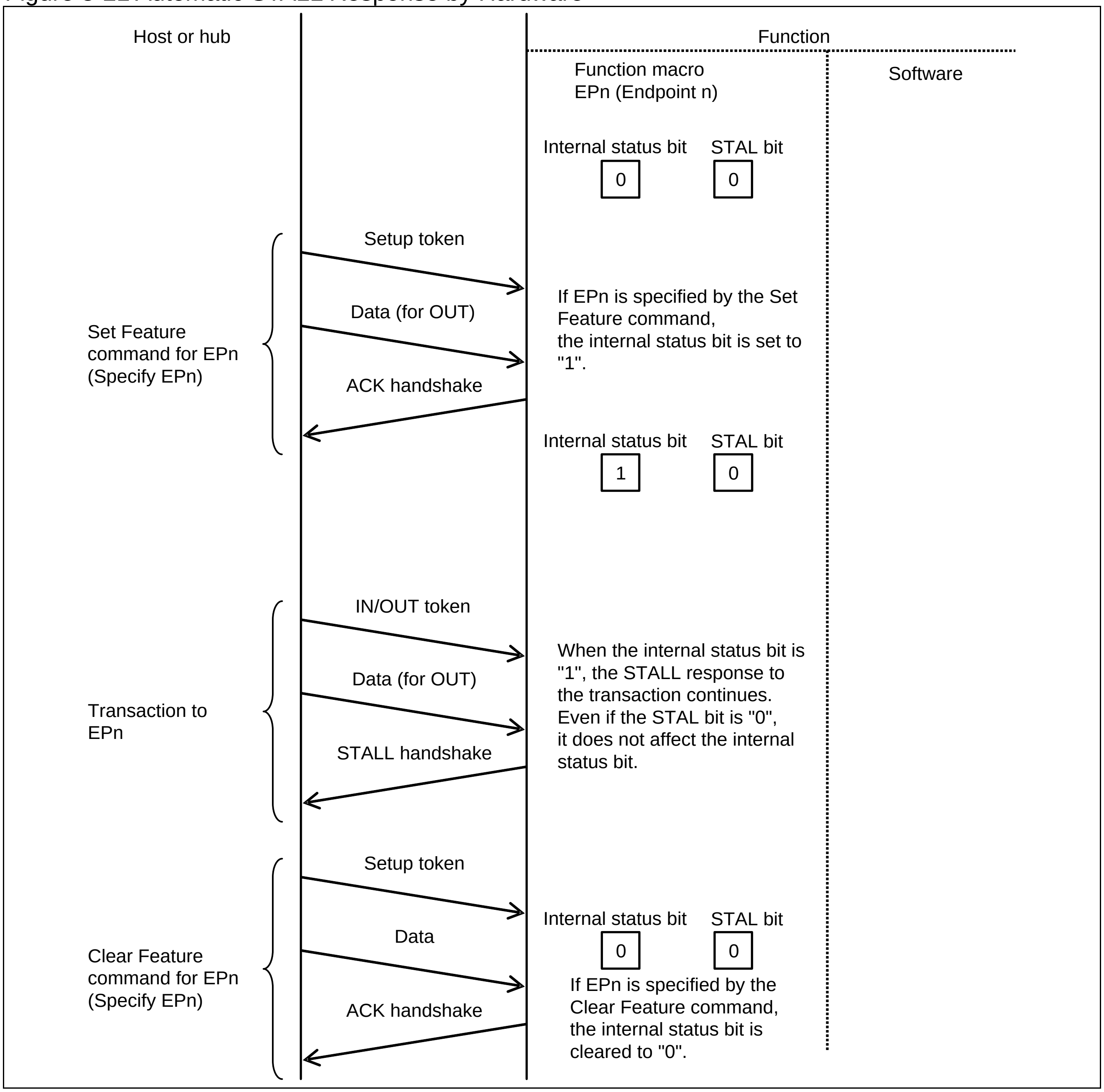
Figure 3-20 To process the STALL response by software (the STAL bit is cleared by hardware) UDCC.STALCLREN=1



■ Automatic STALL response by hardware

Figure 3-21 shows the procedure for the automatic STALL response by hardware.
When the STALL response is set by the Set Feature command, the hardware automatically set the internal status bit of the relevant Endpoint, irrespective of the STAL bit setting, and perform the STALL response. Once the internal bit is set, the value is retained until cleared by the Clear Feature command from the host irrespective of the STAL bit setting.
The STAL bit is referred to even after the internal status bit is cleared by the Clear Feature command. To release the STALL response, therefore, the internal status bit must be cleared by the Clear Feature command.

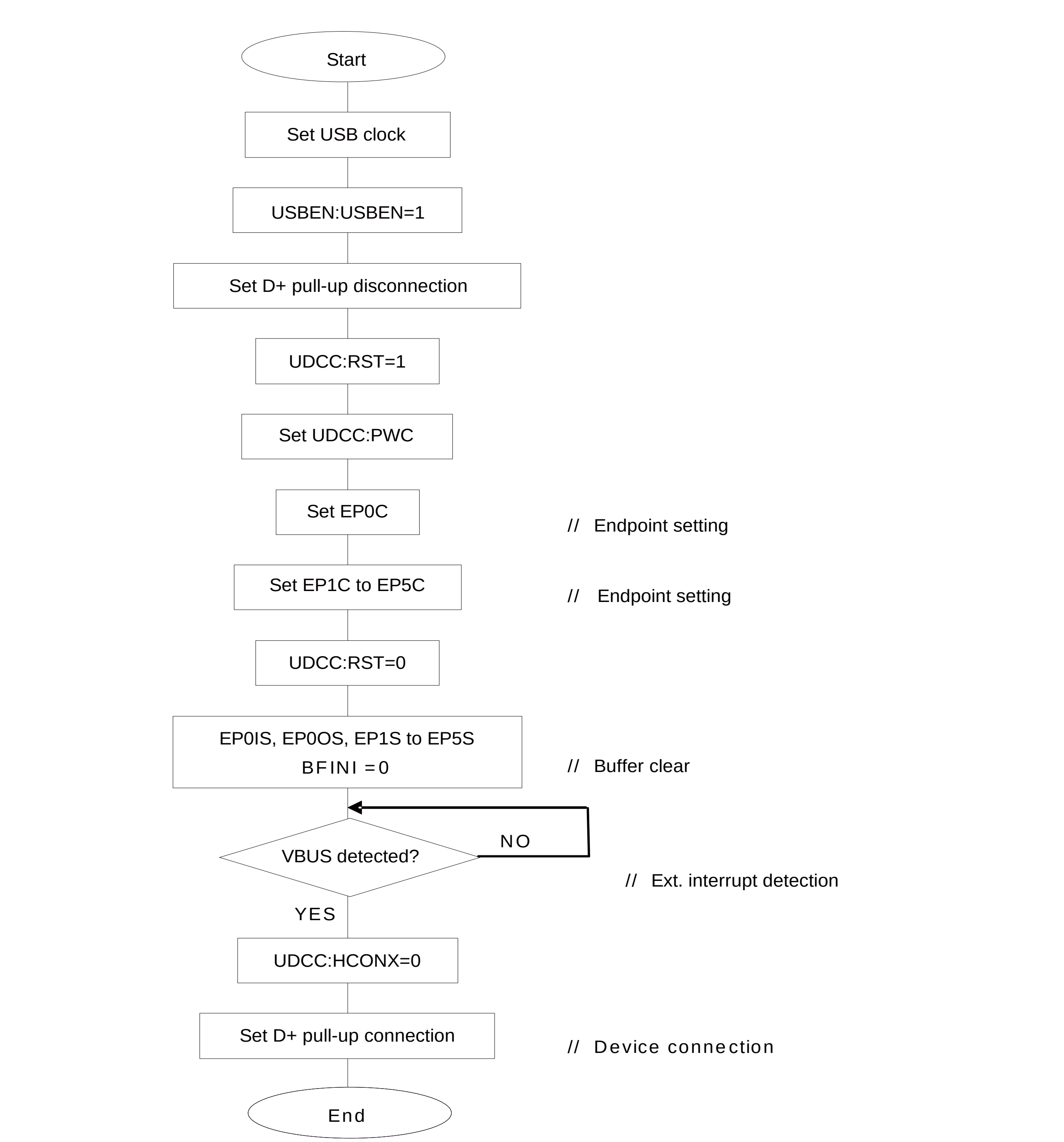
Figure 3-21 Automatic STALL Response by Hardware



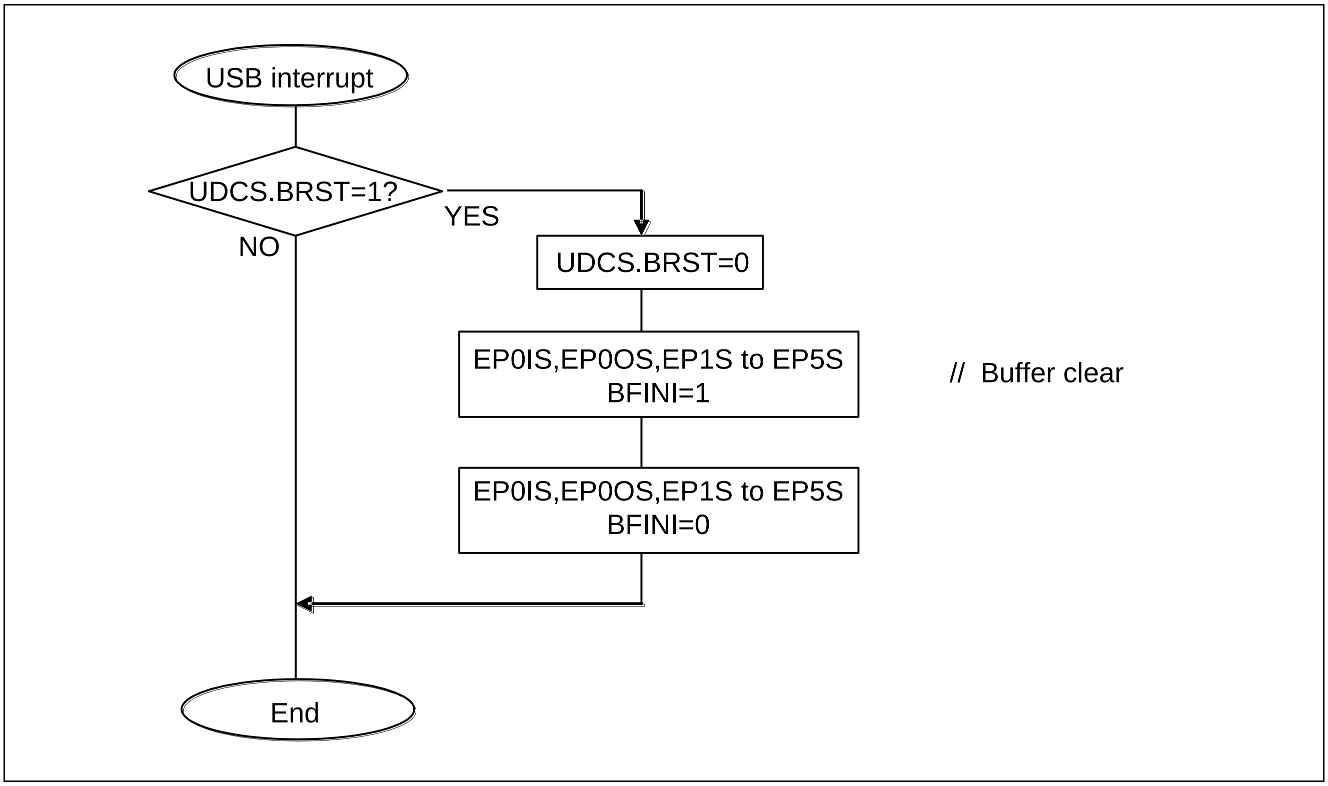
4. Examples of USB Device Setting Procedures

This section provides flowcharts for initialization, bus reset, CPU transfer, packet transfer (IN/OUT) and automatic data size transfer (IN/OUT).

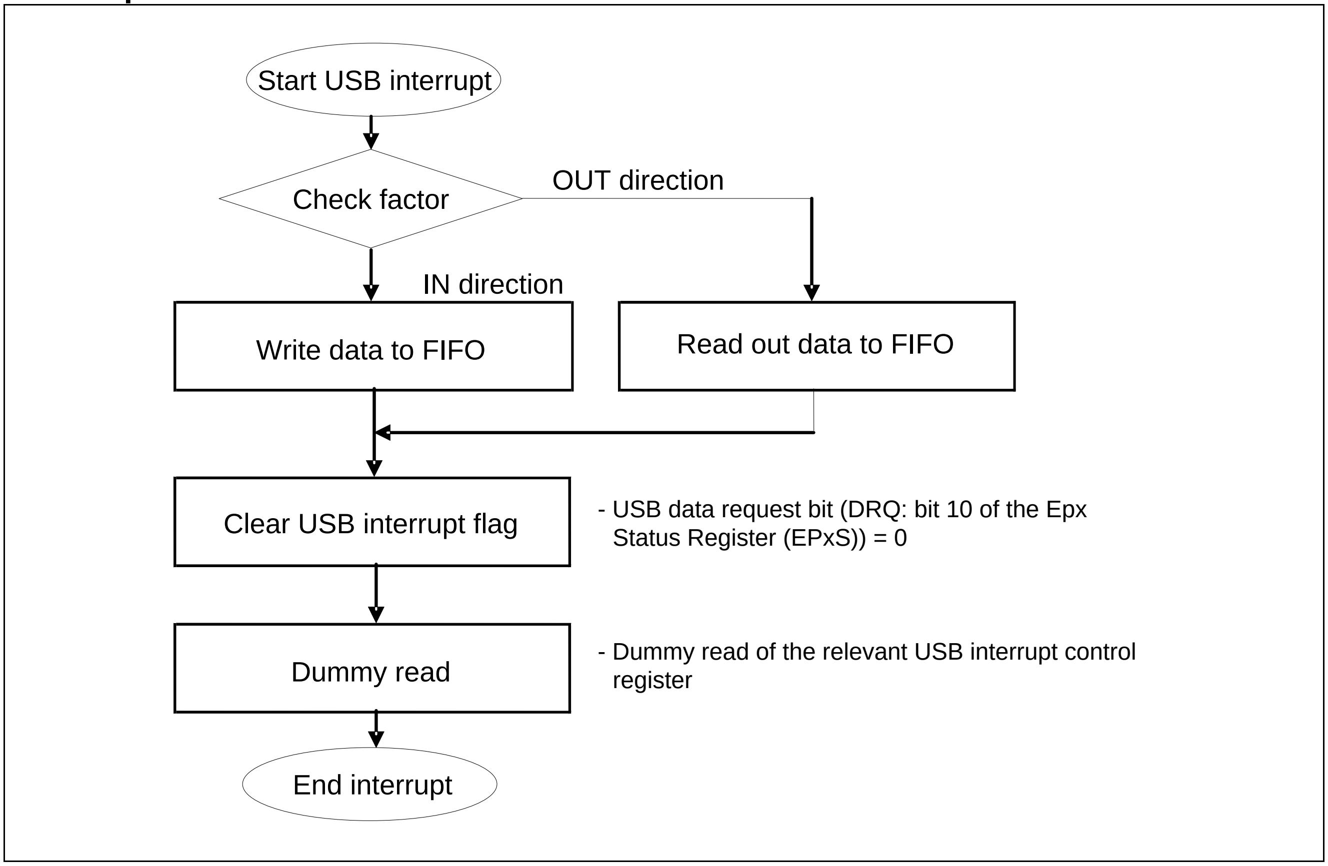
■ Initialization



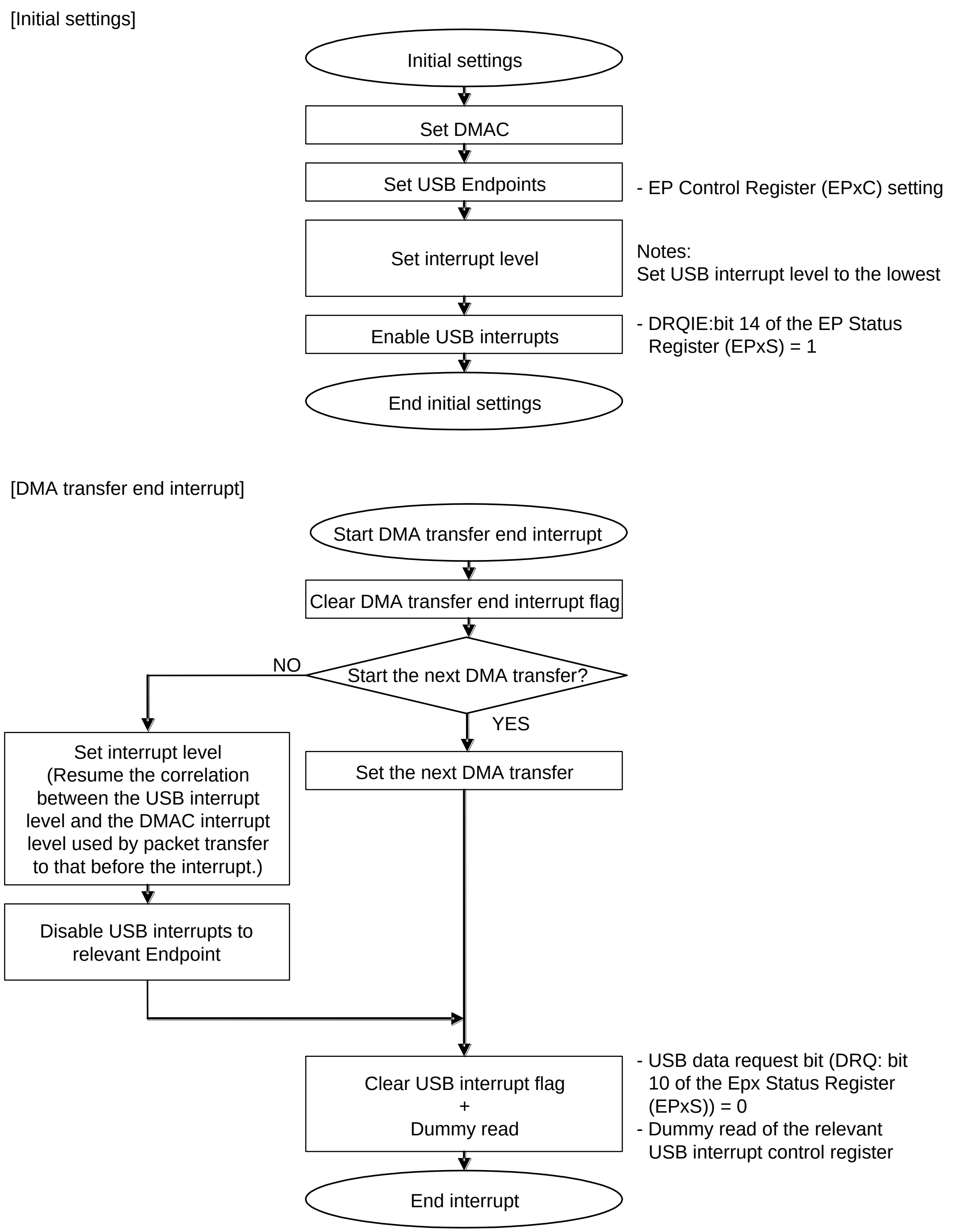
■ Bus reset


USB interrupt
UDCS.BRST=1?
YES
NO
UDCS.BRST=0
EP0IS,EP0OS,EP1S to EP5S
BFINI=1
// Buffer clear
EP0IS,EP0OS,EP1S to EP5S
BFINI=0
End

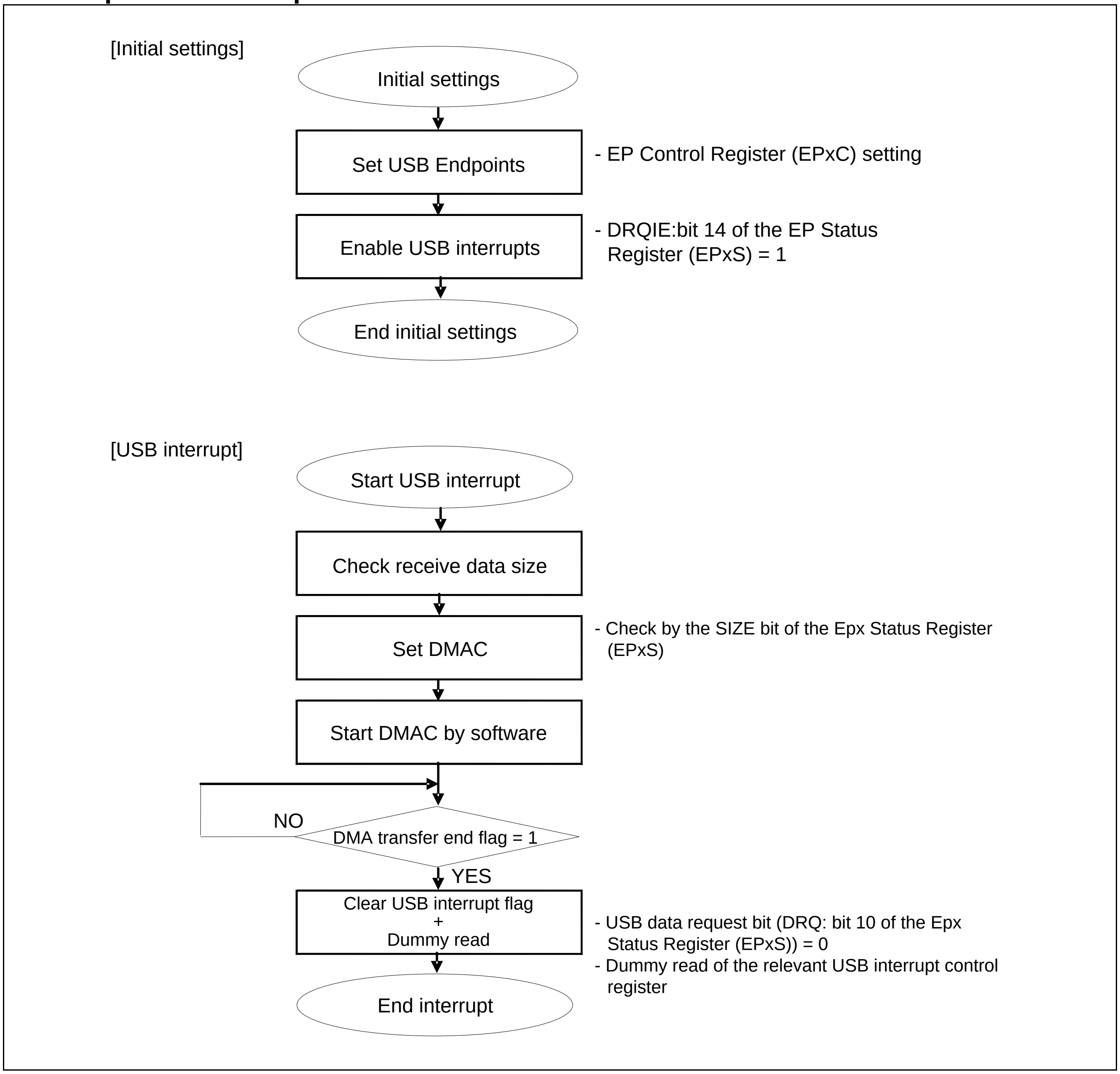
■ Example control for CPU transfer



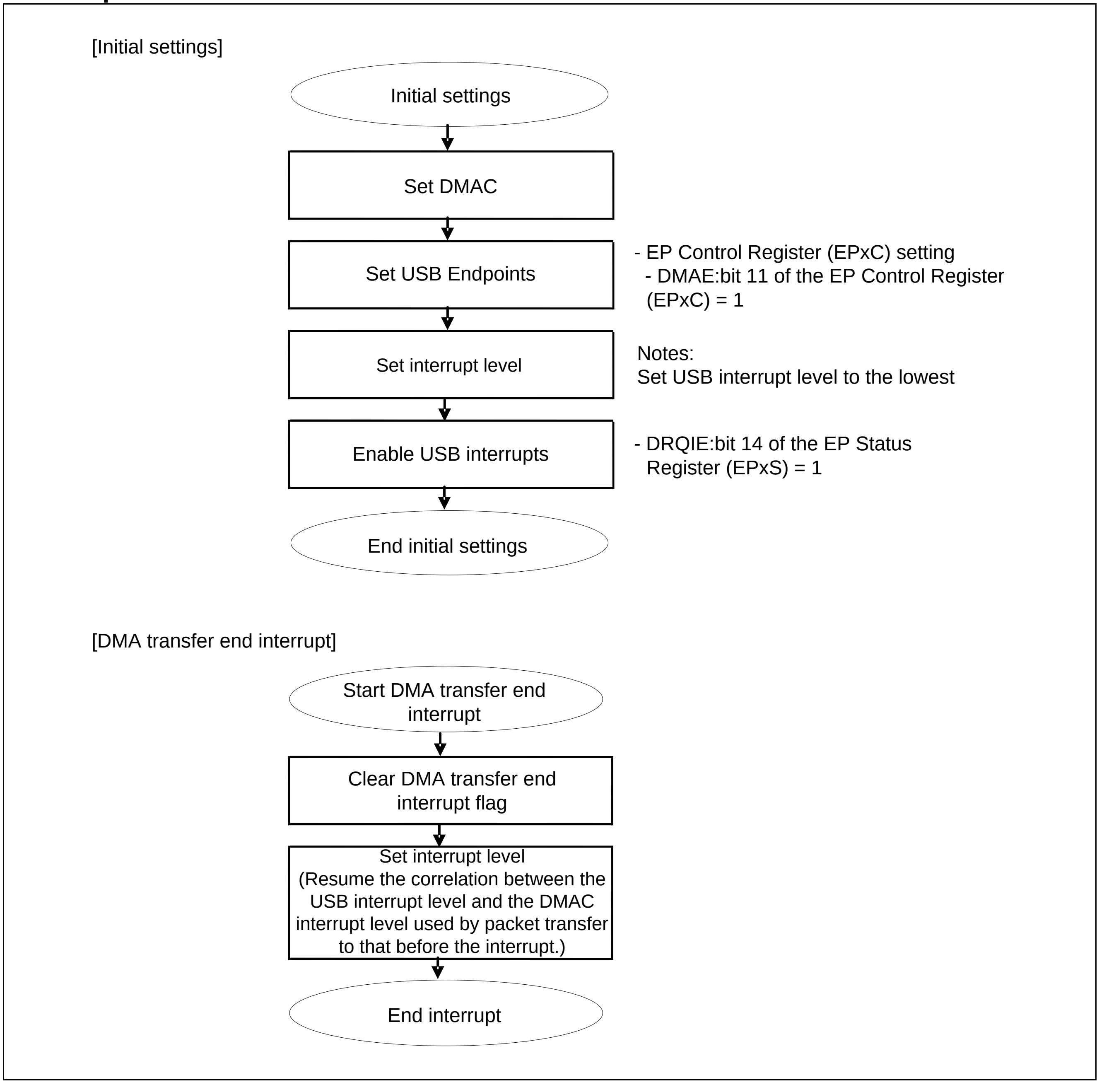
■ Example control for packet transfer in IN direction



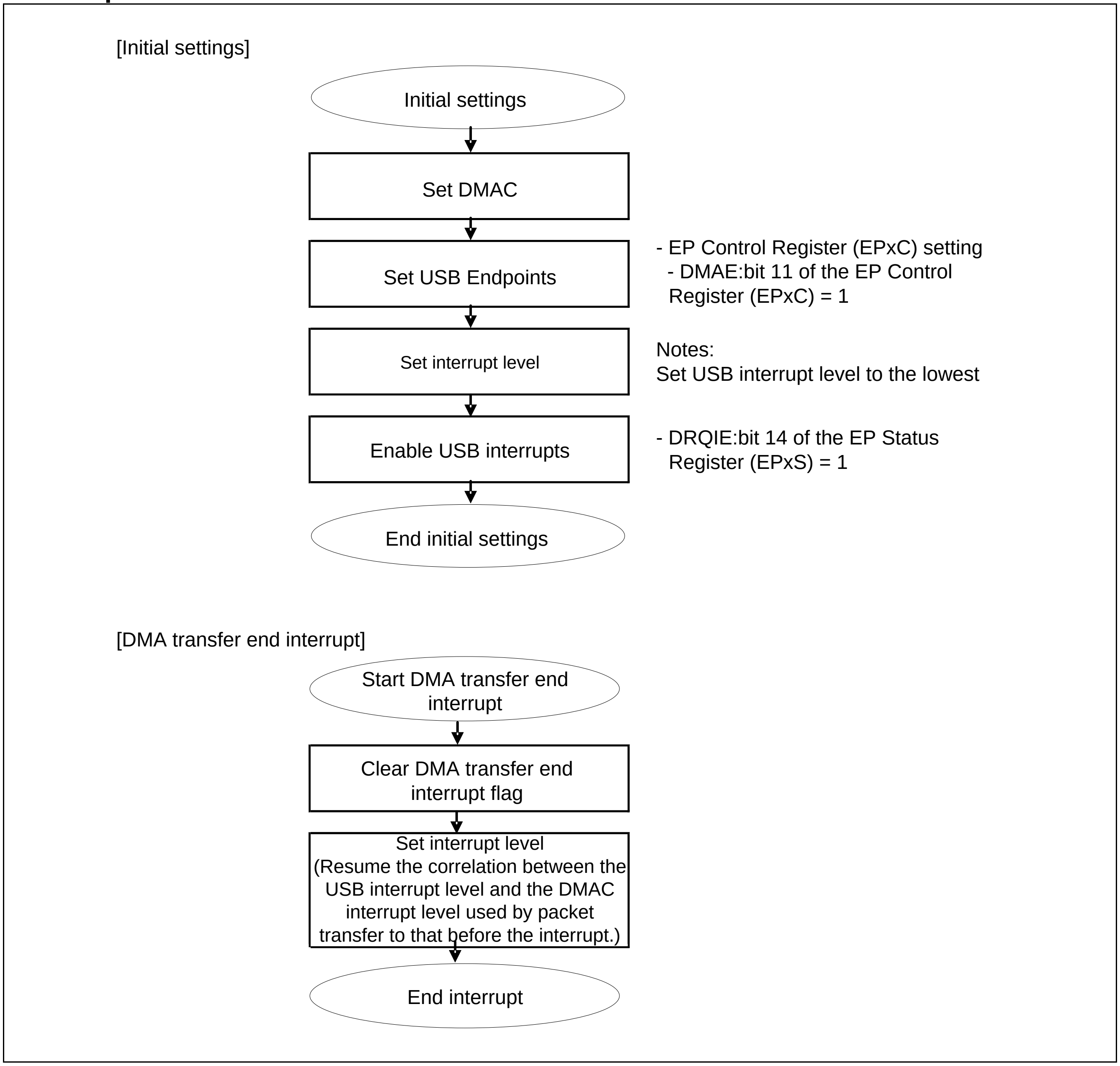
■ Example control for packet transfer in OUT direction



■ Example control for automatic data size transfer in IN direction



■ Example control for automatic data size transfer in OUT direction



5. USB Device Registers

This section explains the configurations and functions of the registers used for the USB device.

■ USB device register list

Abbreviation	Register name	Reference
UDCC	UDC Control Register	5.1
EP0C	EP0 Control Register	5.2
EP1C	EP1 Control Register	5.3
EP2C	EP2 Control Register	
EP3C	EP3 Control Register	
EP4C	EP4 Control Register	
EP5C	EP5 Control Register	
TMSP	Time Stamp Register	5.4
UDCS	UDC Status Register	5.5
UDCIE	UDC Interrupt Enable Register	5.6
EP0IS	EP0I Status Register	5.7
EP0OS	EP0O Status Register	5.8
EP1S	EP1 Status Register	5.9
EP2S	EP2 Status Register	
EP3S	EP3 Status Register	
EP4S	EP4 Status Register	
EP5S	EP5 Status Register	
EP0DTH	EP0 Data Register high-order	5.10
EP0DTL	EP0 Data Register low-order	
EP1DTH	EP0 Data Register high-order	
EP1DTL	EP0 Data Register low-order	
EP2DTH	EP0 Data Register high-order	
EP2DTL	EP0 Data Register low-order	
EP3DTH	EP0 Data Register high-order	
EP3DTL	EP0 Data Register low-order	
EP4DTH	EP0 Data Register high-order	
EP4DTL	EP0 Data Register low-order	
EP5DTH	EP0 Data Register high-order	
EP5DTL	EP0 Data Register low-order	

■ UDCC:RST dependent register bit update timing list

	Register	bit
Register bits to be updated when UDCC:RST=1	UDCC	HCONTX, PFBK, PWC
	EP0C	PKS0
	EP1C	EPEN, TYPE, DIR, PKS1
	EP2C	EPEN, TYPE, DIR, PKS2
	EP3C	EPEN, TYPE, DIR, PKS3
	EP4C	EPEN, TYPE, DIR, PKS4
	EP5C	EPEN, TYPE, DIR, PKS5
Register bits initialized when UDCC:RST=1 (Update when UDCC:RST=0)	EP0IS	BFINI, DRQI
	EP0OS	BFINI, DRQ, SPK
	EP1S	BFINI, DRQ, SPK
	EP2S	BFINI, DRQ, SPK
	EP3S	BFINI, DRQ, SPK
	EP4S	BFINI, DRQ, SPK
	EP5S	BFINI, DRQ, SPK
	TMSP	TMSP
	UDCS	SUSP, SOF, BRST, WKUP, SETP, CONF
	UDCIE	SUSPIE, SOFIE, BRSTIE, WKUPIE, CONFN, CONFIE
Register bits unaffected by UDCC:RST	UDCC	RESUME, USTP
	EP0C	STAL
	EP1C	DMAE, NULE, STAL
	EP2C	DMAE, NULE, STAL
	EP3C	DMAE, NULE, STAL
	EP4C	DMAE, NULE, STAL
	EP5C	DMAE, NULE, STAL
	EP1DTH/L	BFDT
	EP2DTH/L	BFDT
	EP3DTH/L	BFDT
	EP4DTH/L	BFDT
	EP5DTH/L	BFDT

5.1. UDC Control Register (UDCC)

The UDC Control Register (UDCC) controls the UDC core circuit.

The following figure shows the bit configuration of the UDC Control Register (UDCC).

bit	15	14	13	12	11	10	9	8
Field	Reserved							
Attribute	-							
Initial value	0x00							

bit	7	6	5	4	3	2	1	0
Field	RST	RESUM	HCONX	USTP	STALCLREN	Reserved	RFBK	PWC
Attribute	R/W	R/W	R/W	R/W	R/W	-	R/W	R/W
Initial value	1	0	1	0	0	0	0	0

<Note>

The UDC Control Register (UDCC), except bit6 RESUM and bit4 USTP, should be configured while bit7 RST = 1, and should not be rewritten while USB is running. bit6 RESUM must be set or reset in USB suspend mode and while the remote wake-up is enabled by the following command.

Set bit4 USTP to "1" before stop mode or timer mode is entered.

When those modes have been released, set the SUSP of UDCS and USTP of UDCC to "0" in this order after confirmation of stabilized USB supply clock.

The following explains the function of each bit in the UDC Control Register (UDCC).

[bit15:8] Reserved: Reserved bits

Always write "0" to these bits. They are always read as "0".

[bit7] RST: Device Reset bit (device ReSeT)

This bit is ORed with the chip system reset to individually reset the USB device. The USB device is reset by the RST bit when connected with the host via cable. As the initial value is "1", reset enabled, write "0" to release the state.

Value	Description
0	Releases USB Device reset
1	Resets the USB device

<Note>

This bit initializes the relevant bit of the Time Stamp Register (TMSP), UDC Status Register (UDCS), UDC Interrupt Enable Register (UDCIE) at the same time. It also sets the BFINI of the EP0I, EP0O, and EP1 to EP5 Status Register concurrently. After the initial settings, therefore, clear the RST bit (BFINI bit is not cleared) and clear BFINI bit of the Endpoints used in this order.

[bit6] RESUM: Resume Setting bit (RESUMe set)

In suspend state while remote wake-up is enabled *, the resume is started when writing "1" to the RESUM bit. To instruct to resume, set the RESUM bit to "1", and then write "0" to it to clear.

*: The DEVICE_REMOTE_WAKEUP bit is set by the SET_FEATURE command from the host.

Value	Description
0	Resets the USB resume start instruction bit
1	Instructs to start the USB resume

[bit5] HCONX: Host Connection bit (Host CONnection)

This bit controls the switch between an external pull-up resistor and the USB data line to make the connection with the host or HUB recognized.

Value	Description
0	Connected to the host or HUB
1	Disconnected from the host or HUB

<Note>

Even if the connection is found by the host or HUB while the external pull-up resistor is kept ON, the bus reset command on the USB bus is ignored while this bit is "1".

[bit4] USTP: USB Operating Clock Stop bit (Udc SToP)

Setting this bit stops the clock for the USB operating unit. When USB is not operated, power consumption can be reduced by configuring this bit.

Value	Description
0	Normal mode
1	Stops the clock for the USB operating unit

<Note>

If stop mode and timer mode is not set, the USTP bit must be configured after setting RST to "1", and also after 3 cycles at full speed or 43 cycles at low speed (supported only in host mode) so that the reset can be ensured. This bit can be cleared at the same time RST is cleared.

[bit3] STALCLREN: Endpoint 1 to Endpoint 5 STAL bit Clear Select bit (STALl CLeaR Enable)

This bit selects the method to clear the STAL bit of Endpoint 1 to Endpoint 5 using the Clear Feature command. The STALCLREN bit sets whether to automatically clear the STAL bit to "0" by hardware,
a bit of EP1 to EP5 Control Registers (EP1C to EP5C) for Endpoints (1 to 5) specified by the Clear Feature command.
This bit selects the method to clear the STAL bit of the Endpoint Control Registers (EP1C to EP5C), either by software or hardware.

Value	Description
0	Clears the STAL bit of the EP1 to EP5 Control Registers (EP1C to EP5C) by software.
1	Automatically clears the STAL bit of the EP1 to EP5 Control Registers (EP1C to EP5C) by hardware.

<Note>

The STALCLREN bit should be configured while the RST of the UDC Control Register (UDCC) is "1", and should not be rewritten while USB is running.

[bit2] Reserved: Reserved bit

Always write "0" to this bit. It is always read as "0".

[bit1] RFBK: Data Toggle Mode Select bit (Rate Feed BacK mode)

This bit selects the data toggle mode for USB interrupt transfer.

Value	Description
0	Selects the alternating data toggle mode. Toggles data PID when the transfer has finished successfully.
1	Selects the data toggle mode. Unconditionally toggles data PID.

[bit0] PWC: Power Control bit (PoWer Control)

This bit specifies the operating power mode (self power or bus power) of the USB device.
(Configuration of this bit applies to standard command GetStatus.)

Value	Description
0	Bus power
1	Self power

5.2. EP0 Control Register (EP0C)

The EP0 Control Register (EP0C) controls Endpoint 0.

The following figure shows the bit configuration of the EP0 Control Register (EP0C).

bit	15	14	13	12	11	10	9	8
Field	-				Reserved		STAL	Reserved
Attribute	-				-		R/W	-
Initial value	XXXX				00		0	0

bit	7	6	5	4	3	2	1	0
Field	Reserved	PKS0						
Attribute	-	R/W						
Initial value	0	1000000						

<Note>

Except bit9 STAL, the EP0 Control Register (EP0C) must be configured while both of the bit7 RST in the UDC Control Register (UDCC) and bit7 BFINI in the EP0I/O Status Register (EP0I/EP0OS) are "1".
It must not be rewritten while USB is running.

The following explains the function of each bit in the EP0 Control Register (EP0C).

[bit15:12] -: Undefined bits

The written value has no effect. The read value is undefined.

[bit11:10] Reserved: Reserved bits

Always write "0" to these bits.
They are always read as "0".

[bit9] STAL: Endpoint 0 STALL Setting bit (STALl ep0 set)

This bit can set Endpoint 0 to the STALL state (STALL response).
This bit is automatically cleared by hardware. If a SETUP packet is received by Endpoint 0 after the STALL response to Endpoint 0 is performed, this bit is cleared to "0". For the timing to clear this bit, see "n STAL bit clear timing" of "3.8 STALL response/release of Endpoint 0".

Value	Description
0	Ignored
1	Sets the STALL state (STALL response)

<Notes>

- If the STALCLREN bit of UDC Control Register (UDCC) is "0", the STALL response remains operating to the host while the STAL bit is set to "1". Upon the receipt of a normal SETUP packet after STAL bit reset, Endpoint 0 resumes from the STALL state.
- A read-modify-write instruction reads this bit as "0".

[bit8:7] Reserved: Reserved bits

Write value should always be "0".
They are always read as "0".

[bit6:0] PKS0: Packet Size Endpoint 0 Setting bits (PacKet Size ep0 set)

These bits specify the maximum number of bytes transferred by one packet. For Endpoint 0, the maximum number of bytes is 64, and the set value is valid both for IN and OUT directions.

Example: "0x08" => 8 bytes, "0x40" => 64 bytes (maximum value)

<Notes>

- These bits must be configured when both of the RST bit in the UDC Control Register (UDCC) and the BFINI bit in the EP0I/O Status Register (EP0IS/EP0OS) are "1". Do not rewrite while USB is running.
- A value exceeding the maximum number of transferable bytes (0x40), and "0x00" must not be written.

5.3. EP1 to EP5 Control Registers (EP1C to EP5C)

The EP1 to EP5 Control Registers (EP1C to EP5C) control Endpoint 1 to Endpoint 5.

The following figure shows the bit configuration of the EP1 to EP5 Control Registers (EP1C to EP5C).

■ EP1 Control Register (EP1C)

bit	15	14	13	12	11	10	9	8
Field	EPEN	TYPE		DIR	DMAE	NULE	STAL	PSK1
Attribute	R/W	R/W		R/W	R/W	R/W	R/W	R/W
Initial value	0	11		0	0	0	0	1

bit	7	6	5	4	3	2	1	0
Field	PSK1							
Attribute	R/W							
Initial value	0x00							

■ EP2 to EP5 Control Registers (EP2C to EP5C)

bit	15	14	13	12	11	10	9	8
Field	EPEN	TYPE		DIR	DMAE	NULE	STAL	Reserved
Attribute	R/W	R/W		R/W	R/W	R/W	R/W	-
Initial value	0	11		0	0	0	0	0

bit	7	6	5	4	3	2	1	0
Field	Reserved	PKS5 to PKS2						
Attribute	-	R/W						
Initial value	0	1000000						

<Note>

Except DMAE, NULE, and STAL bits, the EP1 to EP5 Control Registers (EP1C to EP5C) must be configured while both of the bit 7 RST in the UDC Control Register (UDCC) and bit 15 BFINI in the EP0 to EP5 Status Registers (EP1S to EP5S) are "1". They must not be rewritten while USB is running.

The following explains the function of each bit in the EP1 to EP5 Control Registers (EP1C to EP5C).

[bit15] EPEN: Endpoint 1 to Endpoint 5 Enable bits (EndPoint1 to EndPoint5 ENable)

This bit enables the Endpoint. Based on the EPEN bit setting, the Endpoint is configured by the host as those used by the function. TYPE, DIR and PKS bits in the EP1 to EP5 Control Registers are valid as the configuration information.

Value	Description
0	Disables the Endpoint
1	Enables the Endpoint

[bit14:13] TYPE: Endpoint Transfer Type Select bits (Endpoint TYPE)

These bits specify the transfer type that the Endpoint supports.

Value	Description
00	Setting is prohibited.
01	Iso transfer (Device operating mode)
10	Bulk transfer
11	Interrupt transfer

<Note>

Iso transfer can be set in function operating mode for Endpoint 1 only or for both Endpoint 1 and Endpoint 2. Setting for Endpoint 2 only, setting for other than Endpoint 1/ Endpoint 2 or setting in host operating mode is disabled.

[bit12] DIR: Endpoint Transfer Direction Select bit (endpoint DIRection)

This bit specifies the transfer direction that the Endpoint supports.

Value	Device operating mode	Host operating mode (EP1 and EP2 only)
0	OUT Endpoint	IN Endpoint
1	IN Endpoint	OUT Endpoint

[bit11] DMAE: DMA Automatic Transfer Enable bit (DMA Enable)

This bit sets a mode that uses DMA for writing or reading transfer data to/from send/receive buffer, and automatically transfers the send/receive data synchronized with an data request in the IN or OUT direction by the host. Until the data size set in the DMA is reached, the data is transferred.

Value	Description
0	Releases the automatic buffer transfer mode
1	Sets the automatic buffer transfer mode

<Note>

The CPU must not access the send/receive buffer while the DMAE bit is set to "1".

[bit10] NULE: NULL Automatic Transfer Enable bit (NULl Enable set)

When a data transfer request in IN direction is received while automatic buffer transfer mode is set (DMAE = 1), this bit sets a mode that transfers 0-byte data automatically upon the detection of the last packet transfer.

Value	Description
0	Releases the NULL automatic transfer mode
1	Sets the NULL automatic transfer mode

<Note>

For data transfer in the OUT direction or when automatic buffer transfer mode is not set, the NULL bit configuration does not affect communication.

[bit9] STAL: Endpoint 1 to Endpoint 5 Stall Setting bit (STALl set)

This bit can set Endpoint to the STALL state (STALL response).

- When the STALCLREN bit of the UDC Control Register (UDCC) is "0"
  This bit is not cleared to "0" by the Clear Feature command. This bit must be cleared by software. For the timing to clear this bit, see "*n* STALL response processed by software" of "3.9 STALL response/release of Endpoint 1 to Endpoint 5".

Value	Description
0	Release the STALL state
1	Sets the STALL state (STALL response)

- When the STALCLREN bit of the UDC Control Register (UDCC) is "1"
  This bit is cleared by hardware. It is cleared to "0" for the Endpoint specified by the Clear Feature command. For the timing to clear this bit, see "*n* STALL response processed by software" of "3.9 STALL response/release of Endpoint 1 to Endpoint 5".

Value	Description
0	Ignored
1	Sets the STALL state (STALL response)

<Notes>

- If the STALCLREN bit of the UDC Control Register (UDCC) is "0", the STALL response remains operating to the host while the STAL bit is set to "1". Return from the STALL state is possible by the Clear Feature command after resetting the STAL bit.
- The value read by a read-modify-write instruction differs depending on the value set in STALCLREN.
  - When STALCLREN = 0, the value at that time is read.
  - When STALCLREN = 1,"0" is read.

[EP2 to EP5: bit8:7] EP2 to EP5 reserved bits

In EP2 to EP5, these bits are reserved. Write value should always be "0". They are always read as "0".

[(EP1: bit8:7) bit6:0] PKS: Packet Size Setting bits (PacKet Size ep1 set)

These bits specify the maximum size transferred by one packet. The following shows the maximum packet size that can be specified for Endpoint 1 to Endpoint 5.

EndPoint	Maximum transfer size	Configurable range
1	256 bytes (Odd numbers allowed)	0x001 to 0x100
2 to 5	64 bytes (Odd numbers allowed)	0x01 to 0x40

<Notes>

- A value exceeding the maximum number of transferable bytes (0x100 or 0x40), and "0x00" must not be written. For Endpoint 2 to Endpoint 5, write "00" to bit8 and bit 7. Also when automatic buffer transfer mode (DMAE = 1) is used, 0 to 2 must not be written to the relevant Endpoint.
- Set even bytes for PKS.

5.4. Time Stamp Register (TMSP)

The Time Stamp Register (TMSP) indicates the frame number upon the receipt of SOF packets.

The following figure shows the bit configuration of the Time Stamp Register (TMSP).

bit	15	14	13	12	11	10	9	8
Field	Reserved	Reserved		Reserved			TMSP	
Attribute	-	-		-		R	R	R
Initial value	X	X		XXX		0	0	0
RST reset	0	0		Irrelevant		0	0	0

bit	7	6	5	4	3	2	1	0
Field				TMSP				
Attribute	R	R	R	R	R	R	R	R
Initial value	0	0	0	0	0	0	0	0
RST reset	0	0	0	0	0	0	0	0

The following explains the function of each bit in the Time Stamp Register (TMSP).

[bit15:11] Reserved: Reserved bits

The written value has no effect on operation. The read value is undefined.

[bit10:0] TMSP: Time Stamp bits (TiMe StamP)

These bits indicate the frame number of a received SOF packet. The frame number is updated upon the receipt of a SOF packet.

5.5. UDC Status Register (UDCS)

The UDC Status Register (UDCS) indicates the bus status during USB communication or the reception of specific commands. Each bit except the SETP bit is an interrupt factor, and so can generate an interrupt to the CPU if the correspondent interrupt enable bit is enabled.

The following figure shows the bit configuration of the UDC Status Register (UDCS).

bit	7	6	5	4	3	2	1	0
Field	-	-	SUSP	SOF	BRST	WKUP	SETP	CONF
Attribute	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	X	X	0	0	0	0	0	0
RST reset	X	X	0	0	0	0	0	0

The following explains the function of each bit in the UDC Status Register (UDCS).

[bit7:6] -: Undefined bits

The written value has no effect on operation. The read value is undefined.

[bit5] SUSP: Suspend detection bit (SUSPend)

This bit indicates that the USB device makes transition to suspend state. The SUSP bit is an interrupt factor, and writing "1" is ignored. Clear it by writing "0". A read-modify-write access reads the bit as "1".

Value	Description
0	No suspend has been detected or interrupt factor has been cleared.
1	Suspend has been detected

[bit4] SOF: SOF Detection bit (Start Of Freame)

This bit indicates that a SOF packet has been received, and then the Time Stamp Register value is updated. The SOF bit is an interrupt factor, and writing "1" is ignored. Clear it by writing "0". A read-modify-write access reads the bit as "1".

Value	Description
0	No SOF has been received or interrupt factor has been cleared.
1	SOF packet has been received

[bit3] BRST: Bus Reset Detection bit (Bus ReSeT)

This bit indicates the detection of a USB bus reset. The BRST bit is an interrupt factor, and writing "1" is ignored. Clear it by writing "0". A read-modify-write access reads the bit as "1".

Value	Description
0	No USB bus reset has been detected or interrupt factor has been cleared.
1	USB bus reset has been detected

<Note>

When this bit is detected, initialize the buffer by the BFINI bit in the EP0I Status Register (EP0IS), the BFINI bit in the EP0O Status Register (EP0OS), and the BFINI bit in the EP1 to EP5 Status Registers (EP1S to EP5S).

[bit2] WKUP: Wake-up Detection bit (WaKe UP)

This bit indicates that the USB device has resumed from suspend state. Remote wake-up caused by the RESUM bit setting, and wake-up caused by a request from the host are the resume factors, but the WKUP bit is automatically set only by a resume request by the host. The WKUP bit is an interrupt factor, and writing "1" is ignored. Clear it by writing "0".A read-modify-write access reads the bit as "1".

Value	Description
0	No host-caused resume has been detected or interrupt factor has been cleared.
1	Host caused resume has been detected

<Note>

Even when wake-up caused by a host request occurs, this bit is not set if the RESUM bit in the UDCC register has been set.

[bit1] SETP: Setup Stage Detection bit (SETuP)

This bit indicates that the received data is the setup stage of USB control transfer. Writing "1" to this bit is ignored. Clear it by writing "0". A read-modify-write access reads the bit as "1".

Value	Description
0	No SETUP stage has been received or factor has been cleared.
1	Setup stage of control transfer has been received

<Note>

The SETP bit is not set during standard command automatic response. This bit is not an interrupt factor.

[bit0] CONF: Configuration Detection bit (CONFigration)

This bit indicates that the USB device has been configured. The CONF bit is set when SetConfig of a USB command is received successfully. The CONF bit is an interrupt factor, and writing "1" is ignored. Clear it by writing "0". A read-modify-write access reads the bit as "1".

Value	Description
0	No SetConfig has been detected or interrupt factor has been cleared.
1	SetConfig has been detected

5.6. UDC Interrupt Enable Register (UDCIE)

The UDC Interrupt Enable Register (UDCIE) enables interrupts generated by the factors of the UDC Status Register with respective bits (except for CONFN bit).

The following figure shows the bit configuration of the UDC Interrupt Enable Register (UDCIE).

bit	15	14	13	12	11	10	9	8
Field	Reserved	Reserved	SUSPIE	SOFIE	BRSTIE	WKUPIE	CONFN	CONFIE
Attribute	-	-	R/W	R/W	R/W	R/W	R	R/W
Initial value	0	0	0	0	0	0	0	0
RST reset	0	Irrelevant	0	0	0	0	0	0

The following explains the function of each bit in the UDC Interrupt Enable Register (UDCIE).

[bit15:14] Reserved: Reserved bits

Always write "0" to these bits. They are always read as "0".

[bit13] SUSPIE: Suspend Interrupt Enable bit (SUSP Interrupt Enable)

This bit enables interrupts generated by the "SUSP" interrupt factor of the UDC Status Register.

Value	Description
0	Disables interrupts generated by the SUSP factor
1	Enables interrupts generated by the SUSP factor

[bit12] SOFIE: SOF Reception Interrupt Enable bit (SOF Interrupt Enable)

This bit enables interrupts generated by the "SOF" interrupt factor of the UDC Status Register.

Value	Description
0	Disables interrupts generated by the SOF factor
1	Enables interrupts generated by the SOF factor

[bit11] BRSTIE: Bus Reset Enable bit (BRST Interrupt Enable)

This bit enables interrupts generated by the "BRST" interrupt factor of the UDC Status Register.

Value	Description
0	Disables interrupts generated by the BRST factor
1	Enables interrupts generated by the BRST factor

[bit10] WKUPIE: Wake-up Interrupt Enable bit (WKUP Interrupt Enable)

This bit enables interrupts generated by the "WKUP" interrupt factor of the UDC Status Register.

Value	Description
0	Disables interrupts generated by the WKUP factor
1	Enables interrupts generated by the WKUP factor

[bit9] CONFN: Configuration Number Indication bit (CONFiguration Number)

This bit indicates the configuration number. The information is updated when the CONF interrupt factor of the UDC Status Register is set.

Value	Description
0	CONFIG number 0
1	CONFIG number 1

[bit8] CONFIE: Configuration Interrupt Enable bit (CONFiguration)

This bit enables interrupts generated by the "CONF" interrupt factor of the UDC Status Register.

Value	Description
0	Disables interrupts generated by the CONF factor.
1	Enables interrupts generated by the CONF factor.

5.7. EP0I Status Register (EP0IS)

The EP0I Status Register (EP0IS) indicates the status of the Endpoint 0 transfer in the IN direction.

The following figure shows the bit configuration of the EP0I Status Register (EP0IS).

bit	15	14	13	12	11	10	9	8
Field	BFINI	DRQIIE	-	-	-	DRQI	-	-
Attribute	R/W	R/W	-	-	-	R/W	-	-
Initial value	1	0	X	X	X	1	X	X
BFINI reset	1	Irrelevant	X	X	X	1	X	X

bit	7	6	5	4	3	2	1	0
Field	-	-	-	-	-	-	-	-
Attribute	-	-	-	-	-	-	-	-
Initial value	X	X	X	X	X	X	X	X
BFINI reset	X	X	X	X	X	X	X	X

The following explains the function of each bit in the EP0I Status Register (EP0IS).

[bit15] BFINI: Send Buffer Initialization bit (BuFfer INItial)

This bit initializes the send buffer of transfer data. In addition, this bit is automatically set to "1" when the RST bit in the UDC Control Register (UDCC) is set to "1". If the RST bit was used for resetting, therefore, set the RST bit to "0" before clearing this bit.

Value	Description
0	Clears the initialization
1	Initializes the send buffer

<Note>

Initialization by the BFINI bit initializes the buffer and the DRQI bit. Before initializing the buffer, make sure that the DRQI or DRQO bit is set, and there is no access from the host, and then configure the STAL bit if necessary.

[bit14] DRQIIE: Send Data Interrupt Enable bit (Data ReQuest In Interrupt Enable)

This bit enables interrupts generated by the "DRQI" interrupt factor of the EP0I Status Register.

Value	Description
0	Disables interrupts generated by the DRQI factor.
1	Enables interrupts generated by the DRQI factor.

[bit13:11] -: Undefined bits

The written value has no effect on operation. The read value is undefined.

[bit10] DRQI: Send/Receive Data Interrupt Request bit (Data ReQuest In)

This bit indicates that the IN packet transfer from the EP0 host normally ended and data was read out from the send buffer, so that the next send data can be written. The DRQI bit is an interrupt factor, and writing "1" is ignored. Clear it by writing "0". A read-modify-write access reads the bit as "1".

Value	Description
0	Clears the interrupt factor
1	Send data can be written to the send buffer

<Note>

This bit must be cleared after data has been written to the send buffer. Also while this bit is not set, "0" must not be written.

Data can be written to the send buffer when DRQI bit is "1". Also when the DRQI bit is cleared, data has been set to the send buffer. When an IN packet request is received while the DRQI bit is "1", therefore, NAK is sent automatically to the host.

[bit9:0] -: Undefined bits

The written value has no effect onoperation. The read value is undefined.

5.8. EP0O Status Register (EP0OS)

The EP0O Status Register (EP0OS) indicates the status of the Endpoint 0 transfer in the OUT direction.

The following figure shows the bit configuration of the EP0O Status Register (EP0OS).

bit	15	14	13	12	11	10	9	8
Field	BFINI	DRQOIE	SPKIE	-	-	DRQO	SPK	Reserved
Attribute	R/W	R/W	R/W	-	-	R/W	R/W	-
Initial value	1	0	0	X	X	0	0	0
BFINI reset	1	Irrelevant	Irrelevant	X	X	0	0	0

bit	7	6	5	4	3	2	1	0
Field	Reserved	SIZE						
Attribute	-	R	R	R	R	R	R	R
Initial value	X	X	X	X	X	X	X	X
BFINI reset	X	X	X	X	X	X	X	X

The following explains the function of each bit in the EP0O Status Register (EP0OS).

[bit15] BFINI: Receive Buffer Initialization bit (BuFfer INItial)

This bit initializes the receive buffer for transfer data. This bit is also automatically set by setting the RST bit of the UDC Control Register (UDCC). If the RST bit was used for resetting, therefore, set the RST bit to "0" before clearing this bit.

Value	Description
0	Clears the initialization
1	Initializes the receive buffer

<Note>

Initialization by the BFINI bit initializes the DRQO and SPK bits. Before initializing the buffer, make sure that the DRQI or DRQO bit is set, and there is no access from the host, and then configure the STAL bit if necessary.

[bit14] DRQOIE: Receive Data Interrupt Enable bit (Data ReQuest Out Interrupt Enable)

This bit enables interrupts generated by the "DRQO" interrupt factor of the EP0O Status Register.

Value	Description
0	Disables interrupts generated by the DRQO factor
1	Enables interrupts generated by the DRQO factor

[bit13] SPKIE: Short Packet Interrupt Enable bit (SPK Interrupt Enable)

This bit enables interrupts generated by the "SPK" interrupt factor of the EP0O Status Register.

Value	Description
0	Disables interrupts generated by the SPK factor
1	Enables interrupts generated by the SPK factor

[bit12:11] -: Undefined bits

The written value has no effect on operation. The read value is undefined.

[bit10] DRQO: Receive Data Interrupt Request bit (Data ReQuest Out)

This bit indicates that the OUT packet transfer from the EP0 host normally ended, and data has been written to the receive buffer, which can be read out. This bit is an interrupt factor, and writing "1" is ignored. Clear it by writing "0". A read-modify-write access reads the bit as "1".

Value	Description
0	Clears the interrupt factor
1	Received data can be read from the receive buffer

<Note>

This bit must be cleared after data has been read from the receive buffer. Also while this bit is not set, "0" must not be written.

The receive buffer is not updated when DRQO is "1". The update is allowed when DRQO is cleared. When an OUT packet request is received while the DRQO bit is "1", therefore, NAK is sent automatically to the host.

[bit9] SPK: Short Packet Interrupt Request bit (Short PacKet)

This bit indicates that the data size transferred from the host does not satisfy the maximum packet size (including 0-byte) set by PKS in the EP0 Control Register (EP0C) when the data has been received successfully. This bit is an interrupt factor, and writing "1" is ignored. Clear it by writing "0". A read-modify-write access reads the bit as "1".

Value	Description
0	Received data size satisfies the maximum packet size
1	Received data size does not satisfy the maximum packet size

[bit8:7] Reserved: Reserved bits

The written value has no effect onoperation. They are always read as "0".

[bit6:0] SIZE: Packet Size Indication bits (packet SIZE)

These bits indicate the number of data bytes written to the receive buffer after EP0's OUT packet transfer has finished. The SIZE bits are updated to a valid value when the DRQO interrupt factor of the EP0O Status Register (EP0OS) has been set.

Example: 8 bytes => "0x08", 64 bytes => "0x40" (maximum value)

5.9. EP1 to EP5 Status Registers (EP1S to EP5S)

The EP1 to EP5 Status Registers (EP1S to EP5S) indicate the status of the Endpoint 1 to Endpoint 5.

The following figure shows the bit configuration of the EP1 to EP5 Status Registers (EP1S to EP5S).

■ EP1 Status Register (EP1S)

bit	15	14	13	12	11	10	9	8
Field	BFINI	DRQIE	SPKIE	Reserved	BUSY	DRQ	SPK	SIZE1
Attribute	R/W	R/W	R/W	-	R	R/W	R/W	R
Initial value	1	0	0	X	0	0	0	X

bit	7	6	5	4	3	2	1	0
Field	SIZE1							
Attribute	R	R	R	R	R	R	R	R
Initial value	X	X	X	X	X	X	X	X

■ EP2 to EP5 Status Registers (EP2S to EP5S)

bit	15	14	13	12	11	10	9	8
Field	BFINI	DRQIE	SPKIE	Reserved	BUSY	DRQ	SPK	Reserved
Attribute	R/W	R/W	R/W	-	R	R/W	R/W	-
Initial value	1	0	0	X	0	0	0	X

bit	7	6	5	4	3	2	1	0
Field	Reserved	SIZE2 to SIZE5						
Attribute	-	R	R	R	R	R	R	R
Initial value	0	X	X	X	X	X	X	X

The following explains the function of each bit in the EP1 to EP5 Control Registers (EP1S to EP5S).

[bit15] BFINI: Send/Receive Buffer Initialization bit (BuFfer INItial)

This bit initializes the send/receive buffer of transfer data. The BFINI bit is also automatically set by setting the RST bit of the UDC Control Register (UDCC). If the RST bit was used for resetting, therefore, set the RST bit to "0" before clearing the BFINI bit.

Value	Description
0	Clears the initialization
1	Initializes the send/receive buffer

<Note>

The EP1 to EP5 send/receive buffers have a double-buffer configuration. The BFINI bit initialization initializes the double buffers concurrently and also initializes the DRQ and SPK bits. Before initializing the buffer, make sure that the DRQ bit is set, and check the BUSY bit to make sure that there is no access from the host, and then configure the STAL bit.

[bit14] DRQIE: Packet Transfer Interrupt Enable bit (Data ReQuest Interrupt Enable)

This bit enables interrupts generated by the "DRQ" interrupt factor of the EP1 to EP5 Status Registers.

Value	Description
0	Disables interrupts generated by the DRQ factor
1	Enables interrupts generated by the DRQ factor

<Note>

To use the automatic buffer transfer mode (DMAE = 1), set DMA and enable transfer before enabling the DRQIE bit.

[bit13] SPKIE: Short Packet Interrupt Enable bit (SPK Interrupt Enable)

This bit enables interrupts generated by the "SPK" interrupt factor of the EP1 to EP5 Status Registers.

Value	Description
0	Disables interrupts generated by the SPK factor
1	Enables interrupts generated by the SPK factor

[bit12] Reserved: Reserved bit

The written value has no effect on operation. The read value is undefined.

[bit11] BUSY: Busy Flag bit (BUSY flag)

This bit indicates that the host is currently gaining write or read access to the send/receive buffer. The BUSY bit is automatically set or reset.

Value	Description
0	No access from the host
1	Write or read access from the host is in process

<Note>

If the BUSY bit is set to "1" while the DRQ bit is set to "1", it indicates that the host is currently accessing either of the double buffers that is not accessed by the CPU or via DMA.

Usually, control using the BUSY bit is not required. To initialize the buffer by setting BFINI, however, take the following steps previously.

1. Make sure that the DRQ bit has been set, and check the BUSY bit to make sure that there is no access from the host.
2. Set the STAL bit.

[bit10] DRQ: Packet Transfer Interrupt Request bit (Data ReQuest)

This bit indicates that the EP1 to EP5 packet transfer has normally ended, and processing of the data is required. The DRQ bit is an interrupt factor, and writing "1" is ignored. Clear the DRQ bit by writing "0" while it is "1". A read-modify-write access reads the bit as "1".

Value	Description
0	Clears the interrupt factor
1	Packet transfer normally ended

<Note>

If automatic buffer transfer mode (DMAE = 1) is not used, "0" must be written to the DRQ bit after data has been written or read to/from the send/receive buffer. Switch the access buffers once the DRQ bit is cleared. That DRQ = 0 may not be read after the DRQ bit is cleared. If the transfer direction is set to IN, and the DRQ bit is cleared without writing buffer data while the DRQ bit is "1", it implies that 0-byte data is set. If DIR of the EP1 to EP5 Control Registers (EP1C to EP5C) is set to "1" at initial settings, the DRQ bit of corresponding Endpoint is set at the same time. Also while the DRQ bit is not set, "0" must not be written.

[bit9] SPK: Short Packet Interrupt Request bit (Short PacKet)

This bit indicates that the data size transferred from the host does not satisfy the maximum packet size (including 0-byte) set by PKS in the EP1 to EP5 Control Registers (EP1C to EP5C) when the data has been received successfully. This bit is an interrupt factor, and writing "1" is ignored. Clear it by writing "0". A read-modify-write access reads the bit as "1".

Value	Description
0	Received data size satisfies the maximum packet size
1	Received data size does not satisfy the maximum packet size

<Note>

The SPK bit is not set during data transfer in the IN direction.

[EP2 to EP5: bit8:7] Reserved: Reserved bits

In EP2 to EP5, these bits are reserved. The written value has no effect on operation. They are always read as "0".

[(EP1: bit8:7) bit6:0] SIZE: packet SIZE

These bits indicate the number of data bytes written to the receive buffer when OUT packet transfer of EP1 to EP5 has finished. The SIZE bit is updated to a valid value when the DRQ interrupt factor of the EP1 to EP5 Status Registers (EP1S to EP5S) has been set.

The maximum transfer data size of Endpoint 1 to Endpoint 5 is as follows:

EndPoint	Maximum transfer size	Indication range
1	256 bytes	0x000 to 0x100
2 to 5	64 bytes	0x00 to 0x40

<Note>

These bits are set to the data size transferred from the host in the OUT direction and written to the buffer. Therefore, a value read during transfer in the IN direction has no effect on operation.

5.10. EP0 to EP5 Data Registers (EP0DTH to EP5DTH/EP0DTL to EP5DTL)

The EP0 to EP5 Data Registers (EP0DTH to EP5DTH/EP0DTL to EP5DTL) control writing or reading transfer data to/from the send/receive buffer for Endpoint 0 to Endpoint 5.

The following figure shows the bit configuration of the EP0 to EP5 Data Registers (EP0DTH to EP5DTH/EP0DTL to EP5DTL).

■ EP0DTH to EP5DTH

bit	15	14	13	12	11	10	9	8
Field	BFDT							
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	X	X	X	X	X	X	X	X

■ EP0DTL to EP5DTL

bit	7	6	5	4	3	2	1	0
Field	BFDT							
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	X	X	X	X	X	X	X	X

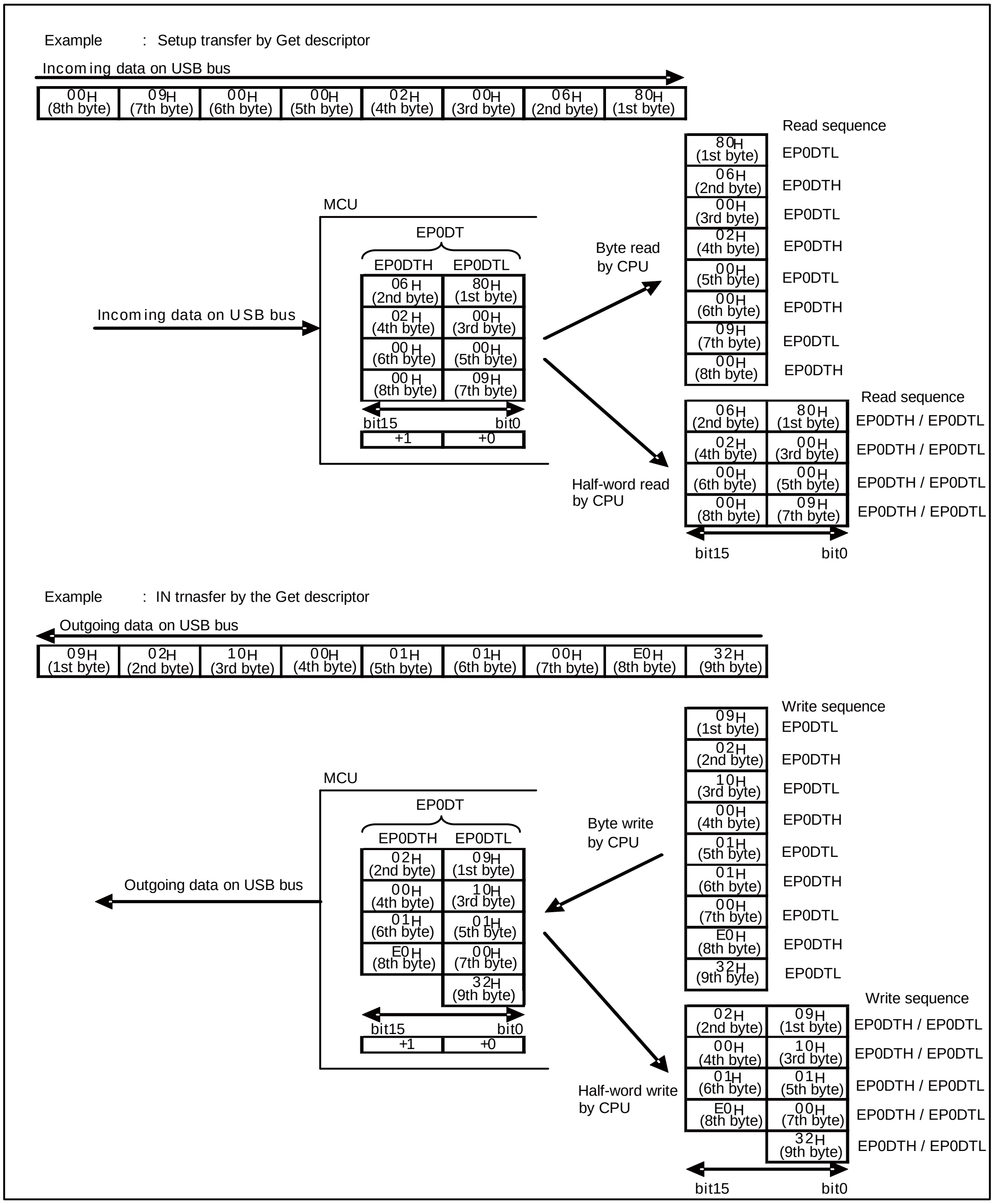
The following explains the function of each bit in the EP0 to EP5 Data Registers (EP0DTH to EP5DTH/EP0DTL to EP5DTL).

[bit15:0] BFDT: Endpoint Send/Receive Buffer Data bits (BuFfer DaTa)

A register used for data write/read to/from the send/received buffer for each end point.

<Notes>

- The CPU can access the EP0 to EP5 Data Registers (EP0DTH to EP5DTH/EP0DTL to EP5DTL) either by the byte or by the half-word.
  - Byte access
    First access low-order (EPxDTL) and then high-order (EPxDTH). Subsequently, access low-order (EPxDTL) and high-order (EPxDTH) alternately.
- This register must not be accessed by the bit operation instruction.



The DMA transfer can only access the EP0 to EP5 Data Registers (EP0DTH to EP5DTH/EP0DTL to EP5DTL) by the half-word. (See "Automatic data size transfer mode" of "3.6 DMA transfer function".)

CHAPTER 3-2: USB Host

This chapter explains the functions and operations of the USB host.

1. Overview of USB host
2. USB host configuration
3. USB host operations
4. USB host setting procedure examples
5. USB host registers

CODE: FW03H-E18.4

1. Overview of USB Host

This section explains the functions and operations of the USB host.

■ Features of USB host

The USB host has the following features:

- Automatic detection of full-speed or low-speed transfer
- Support of full-speed or low-speed transfer
- Automatic detection of device connection or disconnection
- Support of USB bus reset sending function
- Support of IN, OUT, SETUP, and SOF tokens
- Automatic sending of handshake packet for IN token (excluding STALL)
- Automatic detection of handshake packet for OUT token
- Support of maximum packet length of up to 256 bytes
- Support of actions against errors (CRC error, toggle error, and timeout)
- Support of Wake-up function
- Support of Cypress's original USB host functions which can also be operated as USB device by switching the operation mode. (For restrictions in the USB host specifications, see Table 1-1.)

<Note>

Set the base clock to 13 MHz or higher when using the USB host.

Table 1-1 Restrictions in USB host specifications

		Host
Hub support		○[*1]
Transfer functions	Bulk transfer	○
	Control transfer	○
	Interrupt transfer	○
	Isochronous transfer	○
Transfer speed modes	Low Speed	○
	Full Speed	○
PRE packet support		x
SOF packet support		○
Error types	CRC error	○
	Toggle error	○
	Timeout	○
	Max. packet < Received data	○
Detection of device connection or disconnection		○
Detection of transfer speed		○

○: Supported.
x: Not supported.

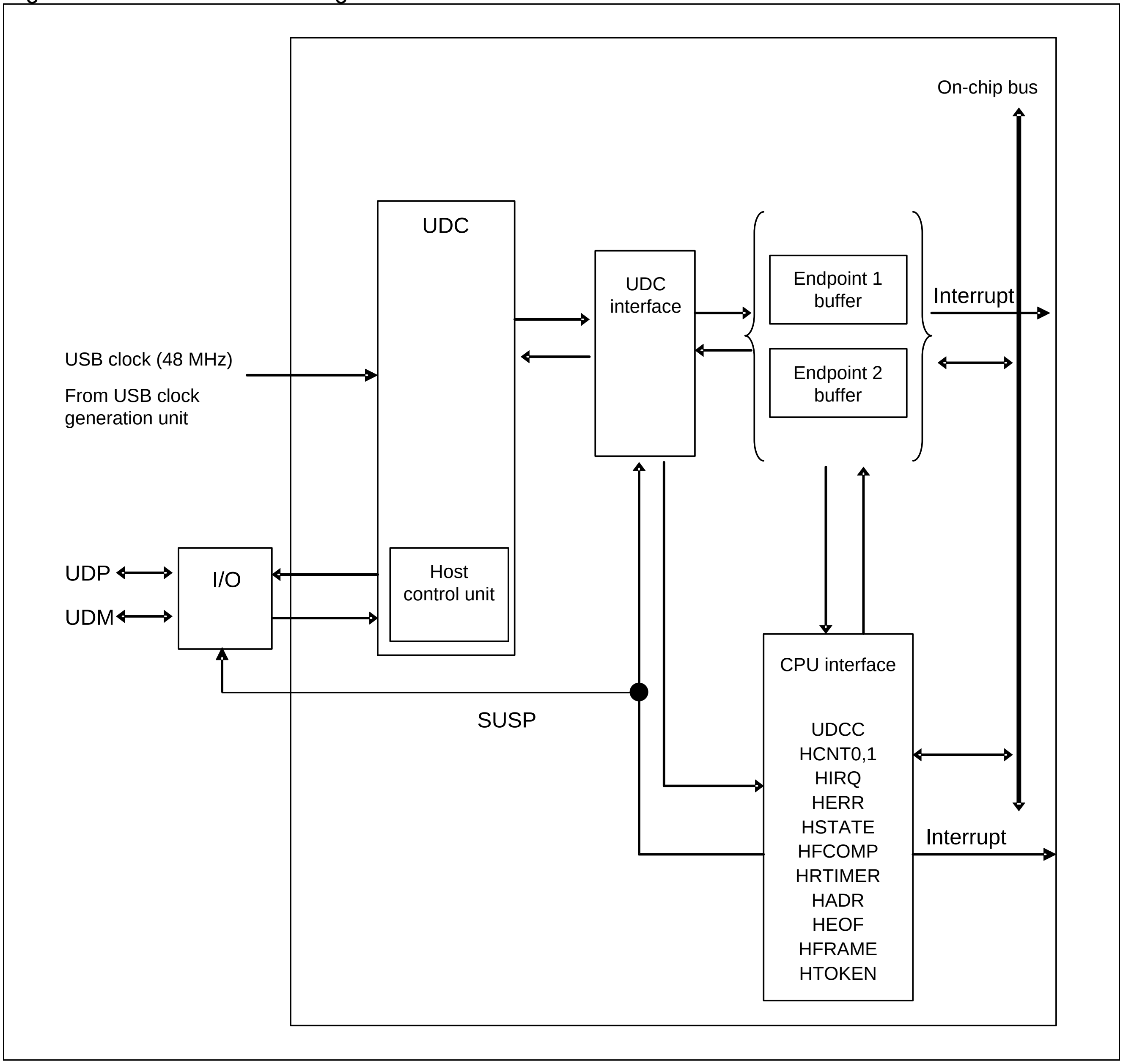
*1: Supports a hub of up to one stage in only the full-speed mode.

2. USB Host Configuration

Figure 2-1 shows the USB host block diagram.

■ USB host block diagram

Figure 2-1 USB host block diagram



3. USB Host Operations

This section explains the operations of the USB host.

3.1. Device connection

This section shows how to detect that an external USB device is connected using software.

■ Host function setting

To carry out USB operation, configure the setting of the USB clock generation unit and enable the USB clock output while the USBEN bit of the USB Enable Register (USBEN) is "0" (USB operation disabled). Next, set the USBEN bit to "1" (USB operation enabled). Then, to operate the USB as a host, set "1" to the HOST bit of Host Control Register 0 (HCNT0).

■ States whether or not an external USB device is connected

When an external USB device is not connected, both of host pins D+ and D- are set to "LOW" by the pull-down resistor. In this case, the CSTAT bit of the Host Status Register (HSTATE) is "0" and the TMODE bit is undefined. When an external USB device is connected, the CSTAT bit of the Host Status Register (HSTATE) is changed to "1".

■ Detection of external USB device connection

When a connection of an external USB device is detected, the CNNIRQ bit of the Host Interrupt Register (HIRQ) is set to "1". If "1" is set to the CNNIRE bit of Host Control Register 0 (HCNT0), a device connection interrupt occurs. To clear this interrupt, write "0" to the CNNIRQ bit of the Host Interrupt Register (HIRQ). When detecting a device connection by polling, instead of an interrupt, use the following steps to create a program.

1. Set the CNNIRE bit of Host Control Register 0 (HCNT0) to "0".
2. Check that the CNNIRQ bit of the Host Interrupt Register (HIRQ) changes to "1".

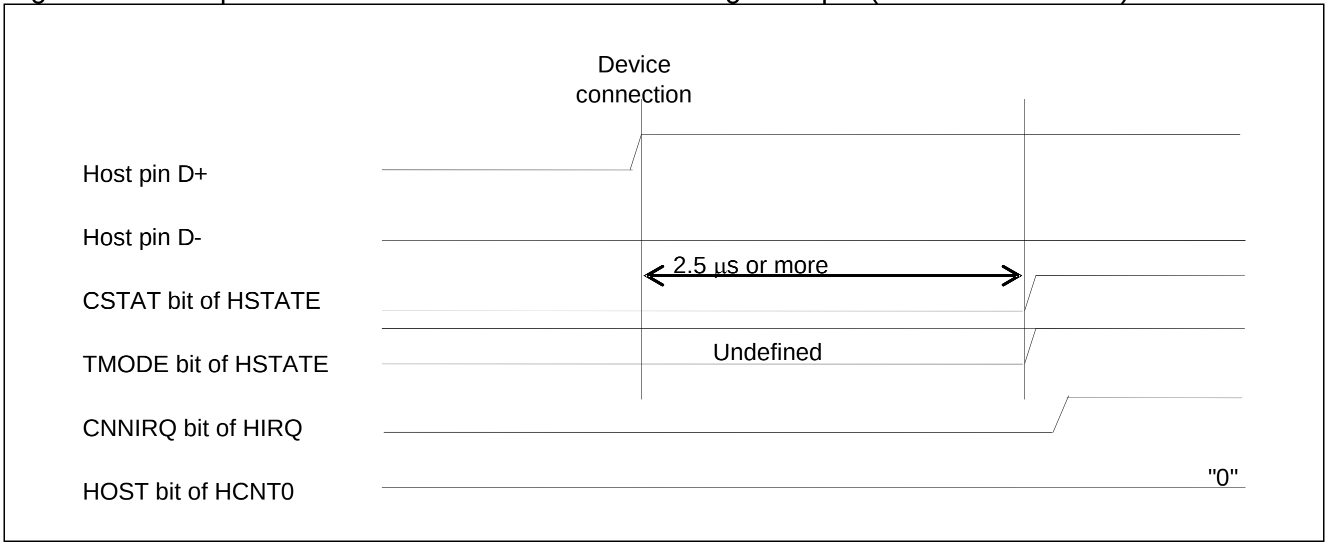
■ Obtaining the transfer speed of the remote USB device and selecting clocks

To obtain the possible transfer speed of the remote USB device after detecting a connection, check the value of the TMODE bit of the Host Status Register (HSTATE). The following shows the relationships between the transfer speed and the value of the TMODE bit of the Host Status Register (HSTATE).

- The destination is a device in the full-speed mode. -> TMODE="1"
- The destination is a device in the low-speed mode. -> TMODE="0"

If the RST bit of the UDC control register (UDCC) is "1" after obtaining the transfer speed of an external USB device, update the CLKSEL bit of the Host Status Register (HSTATE) according to the obtained transfer speed.

Figure 3-1 Full-speed device connection detection timing example (HCNT0:HOST="0")



<Notes>

- When 2.5µs or more lapsed after an external USB device was connected, the CSTAT bit of the Host Status Register (HSTATE) is changed to "1".
- The TMODE and CSTAT bits of the Host Status Register (HSTATE) are updated regardless of the setting of the HOST bit of Host Control Register 0 (HCNT0). The CNNIRQ and DIRQ bits of the Host Interrupt Register (HIRQ) are set to "1" if conditions are satisfied.

3.2. USB bus resetting

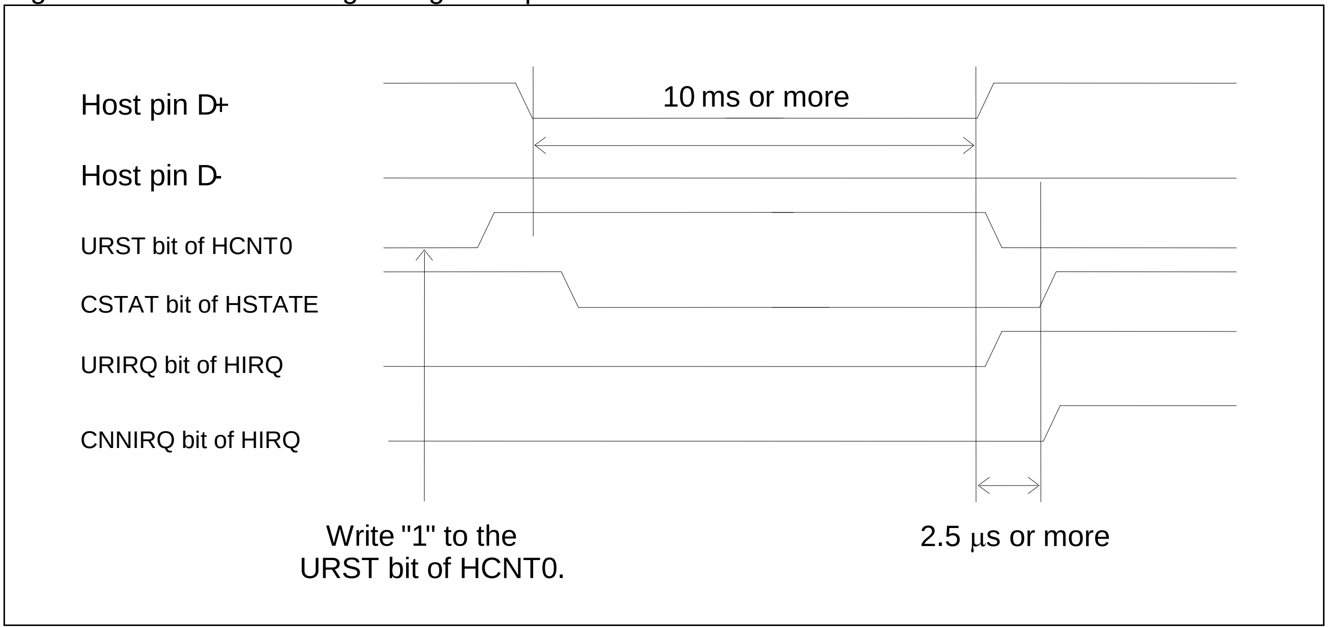
The USB bus is reset by sending SE0 for 10 ms or more if the URST bit of Host Control Register 0 (HCNT0) is set to "1" in the host mode. After USB bus resetting has been completed, the URST bit of Host Control Register 0 (HCNT0) is set to "0", and the URIRQ bit of the Host Interrupt Register (HIRQ) is set to "1". If the URIRE bit of Host Control Register 0 (HCNT0) is then set to "1", an interrupt occurs. To clear this interrupt, write "0" to the URIRQ bit of the Host Interrupt Register (HIRQ).

■ Notes on before and after resetting the USB bus

Note the following points when resetting the USB bus.

1. To check that the device is connected before resetting the USB bus, make sure that the CSTAT bit of the Host Status Register (HSTATE) is set to "1".
2. Resetting the USB bus changes the CSTAT bit of the Host Status Register (HSTATE) to "0", resulting in the USB device being disconnected. At this time, the DIRQ bit of the Host Interrupt Register (HIRQ) is not set to "1".
3. After USB bus resetting has been completed, compare the value of the CLKSEL bit with that of the TMODE bit in the Host Status Register (HSTATE). If they do not match, update the CLKSEL bit to make a match. Update the CLKSEL bit when the RST bit of the UDC Control Register (UDCC) is "1".
4. After USB bus resetting has been completed, check that the USB device is connected using one of the bits shown below, and execute token processing.
   - CNNIRQ bit of Host Interrupt Register (HIRQ)
   - CSTAT bit of Host Status Register (HSTATE)

Figure 3-2 Device resetting timing example



<Note>

No token is issued if a connection of the USB device is not detected after USB bus resetting has been completed.

3.3. Token packet

When issuing an IN, OUT, or SETUP token in the host mode, use the following setting steps to send a token packet.

1. Set the Host Address Register (HADR).
2. Set the DIR and PKS bits of the EP1 Control Register (EP1C) or EP2 Control Register (EP2C).
3. Write the required data to the Host Token Endpoint Register (HTOKEN).

When issuing an SOF token, set the Frame Setup Register (HFRAME) and EOF Setup Register (HEOF), and write the required data to the Host Token Endpoint Register (HTOKEN). The setting above is not required if no change is made in the HADR, EP1C, EP2C, HFRAME, and HEOF registers.

■ Token packet setting

In the host mode, use endpoint 1 and endpoint 2 buffers to send and receive data.
When issuing an IN, OUT, or SETUP token, specify the destination address in the Host Address Register (HADR). Then, specify the maximum number of bytes for each packet in the PKS bit and the transfer direction of each packet in the DIR bit of the EP1 Control Register (EP1C) or EP2 Control Register (EP2C) respectively.

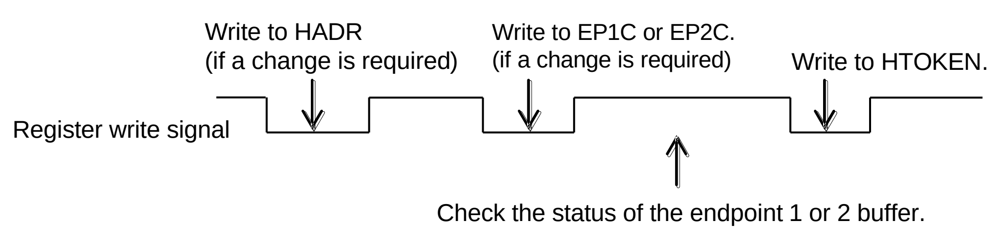
If the DIR bit of the EP1 Control Register (EP1C) is "1", the endpoint 1 buffer is used as an OUT buffer. The endpoint 2 buffer is used as an IN buffer. Then set "0" to the DIR bit of the EP2 Control Register (EP2C).
If the DIR bit of the EP1 Control Register (EP1C) is "0", the endpoint 1 buffer is used as an IN buffer. The endpoint 2 buffer is used as an OUT buffer. Then set "1" to the DIR bit of the EP2 Control Register (EP2C).

Take the following steps to execute token processing.

1. Specify the DIR and PKS bits of the EP1 Control Register (EP1C) and EP2 Control Register (EP2C).
2. If the target endpoint n (n: 1 or 2) is set to the OUT direction, write send data to the endpoint n (n: 1 or 2) buffer. Also set "0" to the DRQ bit of the EPn Status Register (EPnS: n = 1 or 2).
   If the IN direction is selected, read the DRQ bit of the EPn Status Register (EPnS: n = 1 or 2), and check that its value is "0".
3. Specify the target endpoint, token, and toggle data in the Host Token Endpoint Register (HTOKEN).

The USB circuit sends a token packet in the order of Sync, token, address, endpoint, CRC5, and EOP based on the specified token; however, Sync, CRC5, and EOP are sent automatically. After one packet has been sent, the CMPIRQ bit of the Host Interrupt Register (HIRQ) is set to "1". The TKNEN bit of the Host Token Endpoint Register (HTOKEN) is set to "0b000" (see "3.7 SOF interrupt"). At this time, if the CMPIRE bit of Host Control Register 0 (HCNT0) is "1", an interrupt occurs. To clear this interrupt, write "0" to the CMPIRQ bit of the Host Interrupt Register (HIRQ).

Figure 3-3 Example of register setting to issue an IN, OUT, or SETUP token



When issuing an SOF token, specify the EOF time in the EOF Setup Register (HEOF) and the frame number in the Frame Setup Register (HFRAME) respectively. Then specify an SOF token code in the TKNEN bit of the Host Token Endpoint Register (HTOKEN). After this, Sync, SOF token, frame number, CRC5, and EOP are sent, the SOFBUSY bit of the Host Status Resister (HSTATE) is set to "1", and the Frame Setup Register (HFRAME) is incremented by one. The CMPIRQ bit of the Host Interrupt Register (HIRQ) is also set to "1", causing the TKNEN bit of the Host Token Endpoint Register (HTOKEN) to be cleared to "(000)b". If the CMPIRE bit of Host Control Register 0 (HCNT0) is "1", an interrupt occurs. When SOF is sent automatically, an interrupt by CMPIRQ does not occur. To clear a token completion interrupt, write "0" to the CMPIRQ bit of the Host Interrupt Register (HIRQ).
SOF is automatically sent every 1 ms while the SOFBUSY bit of the Host Status Register (HSTATE) is "1". The following shows the conditions (SOF stop conditions) to set the SOFBUSY bit of the Host Status Register (HSTATE) to "0".

- Write "0" to the SOFBUSY bit of the Host Status Register (HSTATE).
- Reset the USB bus (write "1" to the URST bit of HCNT0).
- Write "1" to the SUSP bit of the Host Status Register (HSTATE).
- Disconnect the USB device (when the CSTAT bit of HSTATE is "0").

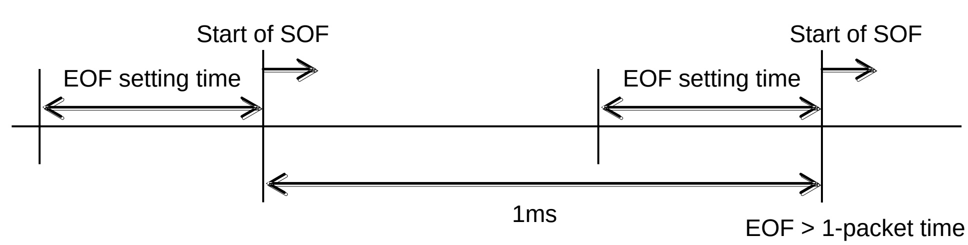
Take the following steps to change the USB from the host mode to the device mode.

1. Set "0" to the SOFBUSY bit of the Host Status Register (HSTATE).
2. Check the following conditions.
   - The SOFBUSY bit of the Host Status Register (HSTATE) is cleared to "0".
   - The TKNEN bit of the Host Token Endpoint Register (HTOKEN) is set to "000".
   - The SUSP bit of the Host Status Register (HSTATE) is set to "0".
3. Set "1" to the RST bit of the UDC Control Register (UDCC).
4. Change the operation mode from the host mode to the device mode.

To set the SOFBUSY bit of the Host Status Register (HSTATE) to "1" again, send an SOF token once more.

The EOF Setup Register is used to prevent SOF from being sent simultaneously with other tokens. If the TKNEN bit of the Host Token Endpoint Register (HTOKEN) is written in the period from the EOF setting time to the SOF starting time, the specified token is placed into the wait state. After SOF has been sent, the token in the wait state is issued. The EOF Setup Register specifies a 1-bit time as the time unit. For example, if "0x10" is specified in the EOF Setup Register, the time is set to 16×1/12MHz=1333.3ns in the full-speed mode and 16×1/1.5MHz=10666.6ns in the low-speed mode. When the EOF setting time is shorter than the 1-packet time, SOF may be sent doubly during execution of other token. In this case, the LSTSOF bit of the Host Error Status Register (HERR) is set to "1", and SOF is not sent. If "1" is set to the LSTSOF bit of the Host Error Status Register (HERR), the value of the EOF Setting Register must be increased (see the explanation of the EOF Setup Register).

Figure 3-4 SOF timing



3.4. Data packet

When sending a data packet after a token packet, transfer toggle data based on the value of the TGGL bit of the Host Token Endpoint Register (HTOKEN). Further, send endpoint 1 or 2 buffer data, CRC16 data, and EOP depending on the value of the DIR bit of the EP1 Control Register (EP1C).
When receiving a data packet, compare the value of the TGGL bit of the Host Token Endpoint Register (HTOKEN) with the received toggle data. If they match, the received data is distributed to the Endpoint 1 or Endpoint 2 buffer depending on the value of the DIR bit of the EP1 Control Register (EP1C) to check for a CRC16 error.

■ Data packet

Take the following steps to send or receive a data packet after sending a token packet.

1. For sending
   - Automatically send Sync.
   - If the TGGL bit of the Host Token Endpoint Register (HTOKEN) is "0", send DATA0. If the TGGL bit is "1", send DATA1.
   - If the DIR bit of the EP1 Control Register (EP1C) is "1", select the Endpoint 1 buffer. If the DIR bit of the EP1 Control Register (EPIC) is "0", select the Endpoint 2 buffer. Then, send all the target data.
   - Send a 16-bit CRC.
   - Send a 2-bit EOP.
   - Send a 1-bit J State.

2. For receiving
   - Receive Sync.
   - Receive toggle data, and compare it with the value of the TGGL bit of the Host Token Endpoint Register (HTOKEN).
   - If the toggle data matches the value of the TGGL bit, check the DIR bit of the EP1 Control Register (EP1C). If the DIR bit is "1", select the Endpoint 2 buffer. If the DIR bit of the EP1 Control Register (EP1C) is "0", select the Endpoint 1 buffer. Then, distribute the received data to the respective buffers.
   - Verify the 16-bit CRC when EOF is received.

When the HOST bit of Host Control Register 0 (HCNT0) is "1", set the inverted value to the respective DIR bits of the EP1 Control Register (EP1C) and EP2 Control Register (EP2C). For example, if "0" is set to the DIR bit of the EP1 Control Register (EP1C), set "1" to the DIR bit of the EP2 Control Register (EP2C).

3.5. Handshake packet

A handshake packet is used to notify the remote device of the status of the local device.

■ Handshake packet

A handshake packet sends either one of ACK, NAK, and STALL from the receiving side when it is judged that the receiving side is ready to receive data normally. If the USB circuit receives a handshake packet, the type of the received handshake packet is set to the HS bit of the Host Error Status Register (HERR). If the USB circuit sends a handshake packet, the type of the sent handshake packet is set to the HS bit of the Host Error Status Register (HERR).

3.6. Retry function

When a NAK or CRC error occurs at the end of a packet, if "1" is set to the RETRY bit of Host Control Register 1 (HCNT1), processing is retried repeatedly for the period specified in the Retry Timer Setting Register (HRTIMER).

■ Retry function

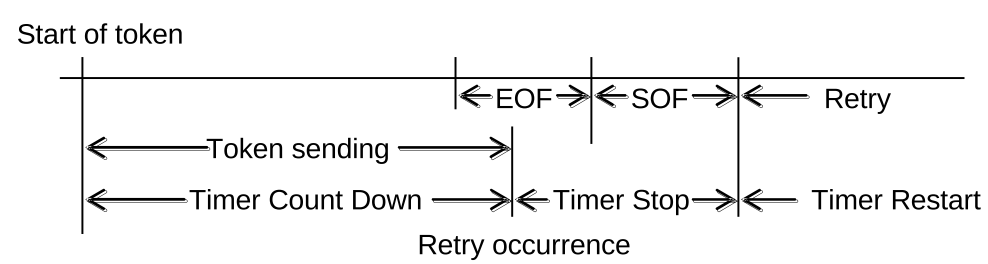
When an error* other than STALL or device disconnection occurs, the target token is retried if the RETRY bit of Host Control Register 1 (HCNT1) is "1". The following shows the conditions to end retry processing.

* : HERR:HS="01", HERR:RERR="1", HERR:TOUT="1", HERR:TGERR="1",HERR:CRC="1", HERR:STUFF="1"

- The RETRY bit of Host Control Register 1 (HCNT1) is set to "0".
- "0" is detected in the retry timer.
- The interrupt flag is generated by SOF (SOFIRQ of HIRQ = "1").
- ACK is detected.
- A device disconnection is detected.

The retry timer is activated at start of a token, and counted down by a 1-bit transfer clock. If retry occurs in the EOF area, counting stops. If a SOF token is ended while the SOFIRQ bit of HIRQ is "0", counting restarts from the timer value at the time when counting stopped. When the retry timer runs out to "0" and a packet ends, the CMPIRQ bit of the Host Interrupt Register (HIRQ) is set to "1".

Figure 3-5 Retry timer operation (SOFIRQ of HIRQ = "0")



When retry processing is ended, end information of the EOP is set to each register.

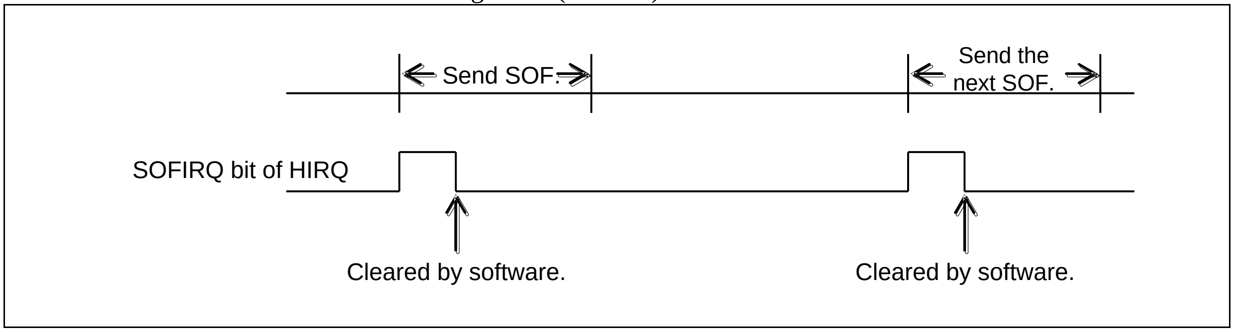
3.7. SOF interrupt

The SOFIRQ bit of the Host Interrupt Register (HIRQ) is set to "1" at start of SOF depending on the setting of the SOFSTEP bit of Host Control Register 1 (HCNT1) and SOF Interrupt Frame Compare Register (HFCOMP). If the SOFIRE bit of Host Control Register 0 (HCNT0) is "1", an interrupt occurs. When SOF processing is executed using the Host Token Endpoint Register (HTOKEN), the SOFIRQ bit of the Host Interrupt Register (HIRQ) is not set to "1".

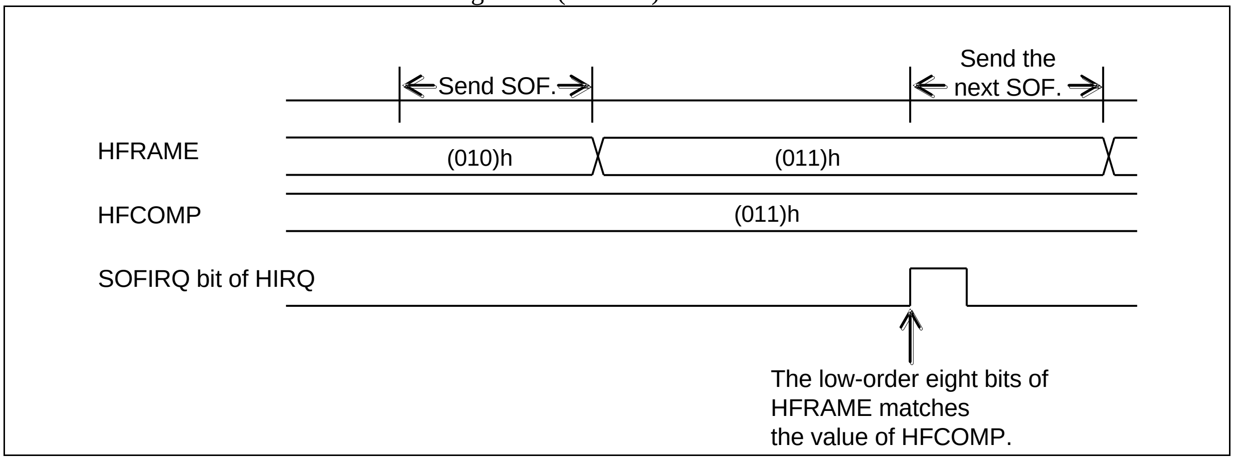
■ SOF interrupt

When the SOFSTEP bit of Host Control Register 1 (HCNT1) is "0", the value of the SOF Interrupt Frame Compare Register (HFCOMP) is compared with the low-order eight bits of the frame number for SOF token. If they match, "1" is set to the SOFIRQ bit of the Host Interrupt Register (HIRQ) when sending SOF. When the SOFSTEP bit of Host Control Register 1 (HCNT1) is "1", "1" is set to the SOFIRQ bit of the Host Interrupt Register (HIRQ) each time SOF is sent.

1. When the SOFSTEP bit of Host Control Register 1 (HCNT1) is "1":



2. When the SOFSTEP bit of Host Control Register 1 (HCNT1) is "0":



If "1" is set to the CANCEL bit of Host Control Register 1 (HCNT1), the target token is not sent when it is set at the following timing.

· A token other than SOF is set to the Host Token Endpoint Register (HTOKEN) in the EOF area.

If a token is set at this timing, the following operations are carried out.

· If the SOFIRQ bit of the Host Interrupt Register (HIRQ) is set to "1" when the next SOF is sent, the TKNEN bit of the Host Token Endpoint Register (HTOKEN) is immediately cleared to "0b000". In this case, that token is not sent.

The TKNEN bit of the Host Token Endpoint Register (HTOKEN) is cleared at the following timing.

At this timing, the CMPIRQ bit of the Host Interrupt Register (HIRQ) is not set to "1". When the SOFIRQ bit is set to "1", the TCAN bit of the Host Interrupt Register (HIRQ) indicates that a token is canceled. When retrying to send a token, write "0" to the TCAN bit of the Host Interrupt Register (HIRQ). Then write a token to be sent to the TKNEN bit of the Host Token Endpoint Register (HTOKEN).

If "0" is set to the CANCEL bit of Host Control Register 1 (HCNT1), the token specified in the Host Token Endpoint Register (HTOKEN) is sent following SOF.

Figure 3-6 Token cancellation example at CANCEL bit of HCNT1 = "1"

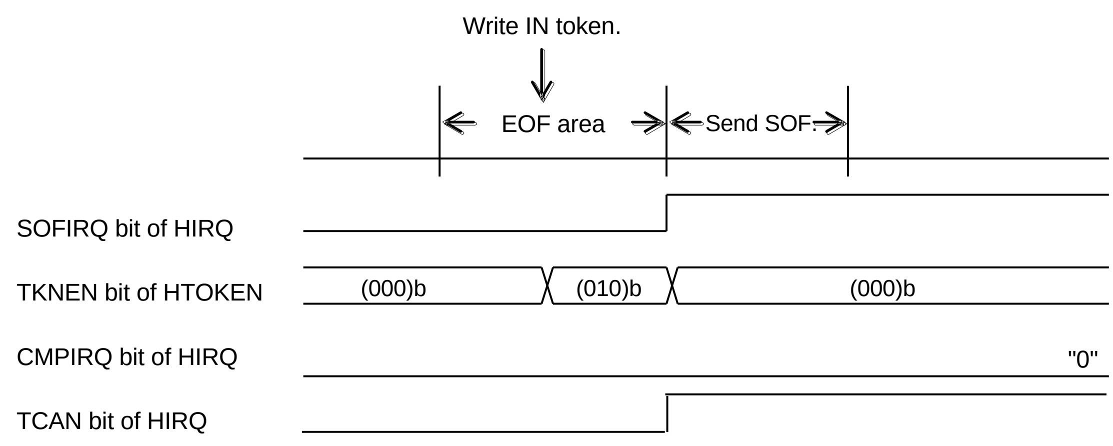
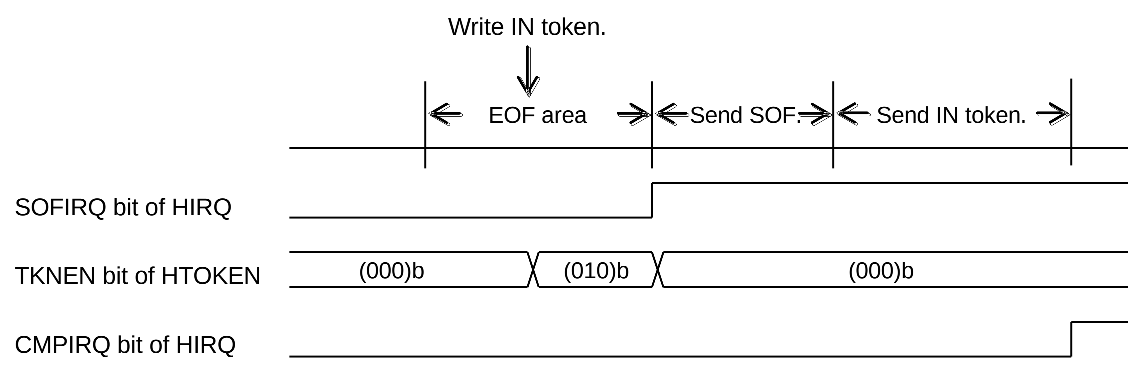


Figure 3-7 Token operation example at CANCEL bit of HCNT1 = "0"



3.8. Error status

The USB host supports error information.

■ Error status

1. Stuffing Error
   If "1" is successively set to six bits, "0" is inserted into one bit. If "1" is successively detected in seven bits, it is judged to be Stuffing Error, and the STUFF bit of the Host Error Status Register (HERR) is set to "1". To clear this status, write "0" to the STUFF bit. If the next token is sent without clearing the STUFF bit, a factor is reflected on the STUFF bit when the next token is ended.

2. Toggle Error
   When sending an IN token, the toggle data of a data packet is compared with the value of the TGGL bit of the Host Token Endpoint Register (HTOKEN). If they do not match, the TGERR bit of the Host Error Register (HERR) is set to "1". To clear the TGERR bit, write "0" to the TGERR bit of the Host Error Register (HERR). If the next token is sent without clearing the TGERR bit, a factor is reflected on the TGERR bit when the next token is ended.

3. CRC Error
   When receiving an IN token, data and CRC of the received data packet are obtained with the CRC polynomial "G(X) = X16 + X15 + X2 + 1". If the remainder is not "(800d)h", it means that CRC Error occurs, and the CRC bit of the Host Error Register (HERR) is set to "1". To clear the CRC bit, write "0" to the CRC bit of the Host Error Register (HERR). If the next token is sent without clearing the CRC bit, a factor is reflected on the CRC bit when the next token is ended.

4. Time Out Error
   "1" is set to the TOUT bit of the Host Error Status Register (HERR) when:

   - A data packet or handshake packet has not been input in the specified time;
   - SE0 has been detected during data receiving; or
   - Stuffing Error has been detected.

   To clear the TOUT bit, write "0" to the TOUT bit of the Host Error Register (HERR). If the next token is sent without clearing the TOUT bit, a factor is reflected on the TOUT bit when the next token is ended.

5. Receive Error
   If EP1 is used as a receive buffer, the value of the PKS bit of the EP1 Control Register (EP1C) is used as the receive packet size. If EP2 is used as a receive buffer, the value of the PKS bit of the EP2 Control Register (EP2C) is used as the receive packet size. When the received data exceeds the specified receive packet size, the RERR bit of the Host Error Status Register (HERR) is set to "1". To clear the RERR bit, write "0" to the RERR bit of the Host Error Register (HERR). If the next token is sent without clearing the RERR bit, a factor is reflected on the RERR bit when the next token is ended.

3.9. End of packet

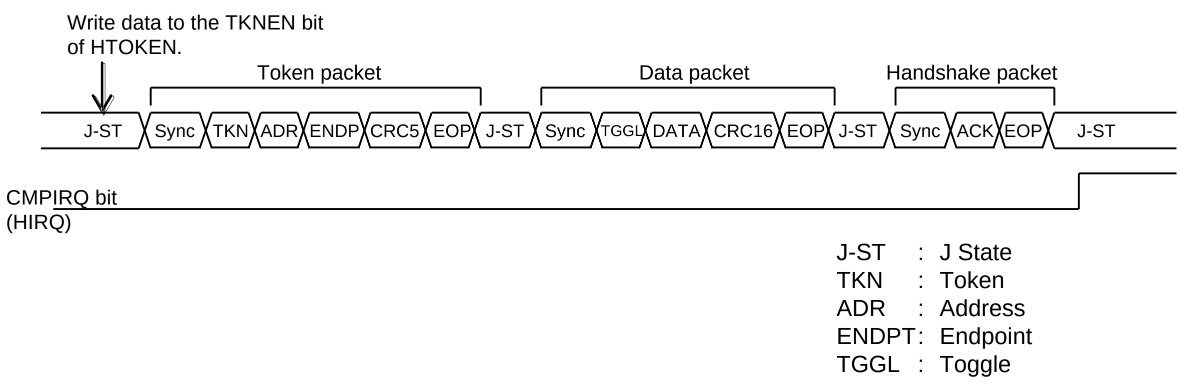
If one packet is ended in the USB host, the CMPIRQ bit of the Host Interrupt Register (HIRQ) is set to "1". At this time, if the CMPIRE bit of Host Control Register 0 (HCNT0) is "1", an interrupt occurs.

■ Packet end timing

When one packet ends, the interrupt flag is generated when:

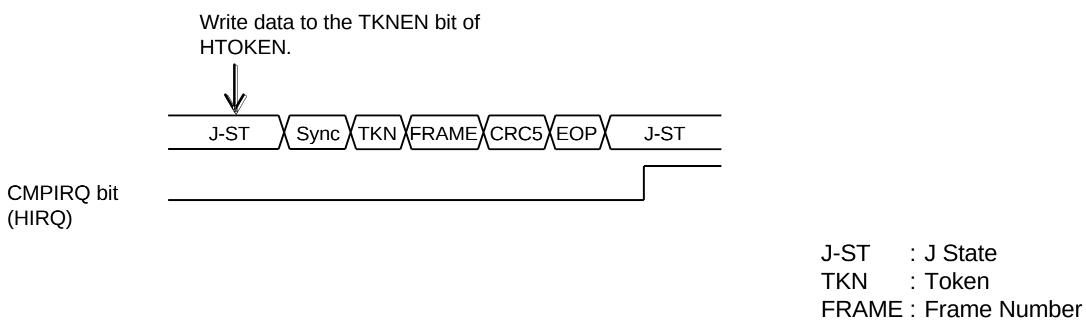
- The TKNEN bit of the Host Token Endpoint Register (HTOKEN) is "(001)b", "(010)b", or "(011)b" (SETUP token, IN token, or OUT token).

Figure 3-8 Timing example 1 when setting the CMPIRQ bit of the Host Interrupt Register (HIRQ)



- The TKNEN bit of the Host Token Endpoint Register (HTOKEN) is "(100)b" (SOF token).

Figure 3-9 Timing example 2 (SOF token) when setting the CMPIRQ bit of the Host Interrupt Register (HIRQ)



3.10. Suspend and resume operations

The USB host supports suspend and resume operations.

■ Suspend operation

If "1" is set to the SUSP bit of the Host Status Register (HSTATE), the procedure below is performed, and the USB circuit is placed into the suspend state.

- The USB bus is placed in the high-impedance state.
- A circuit block with no clock required is stopped.

If the USB circuit is placed in the suspend state, the SUSP bit of the Host Status Register (HSTATE) is set to "1".

However, the following operations are prohibited while resetting the USB bus.

- "1" is set to the SOFBUSY bit of the Host Status Register (HSTATE) or the USB circuit is placed into the suspend state during data transfer.
- Clocks supplied to the USB are stopped in the suspend state.

Take the following steps to stop clocks.

1. Change to the stop or timer mode.
2. Set the UCEN bit of the USB Clock Setup Register (UCCR) to "0".

■ Resume operation

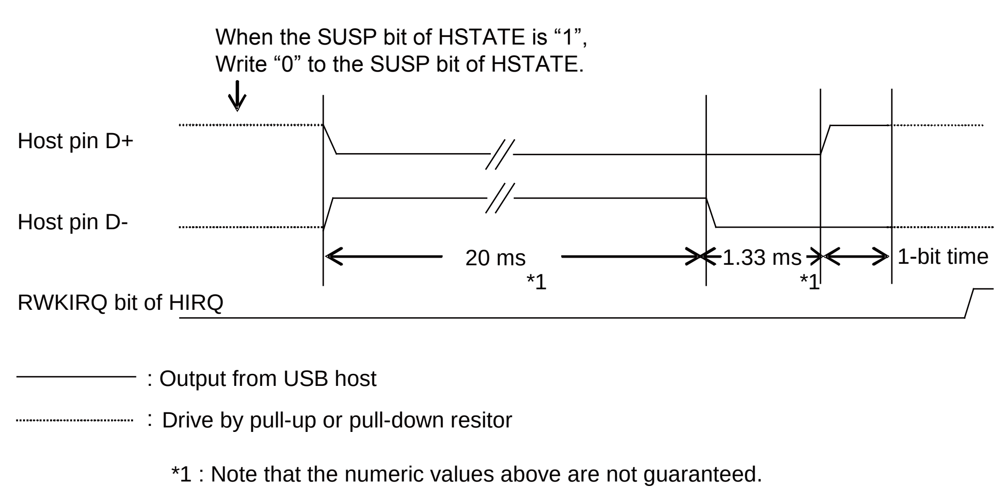
The USB bus changes from the suspend state to the resume state to resume processing when one of the following conditions is satisfied.

- "0" is set to the SUSP bit of the Host Status Register (HSTATE).
- The host pin D+ or D- is placed in the K-state mode.
- A device disconnection is detected.
- A device connection is detected.

After the RWKIRQ bit of the Host Interrupt Register (HRQ) has been set to "1", a token can be issued. The following shows the operation timing for each condition.

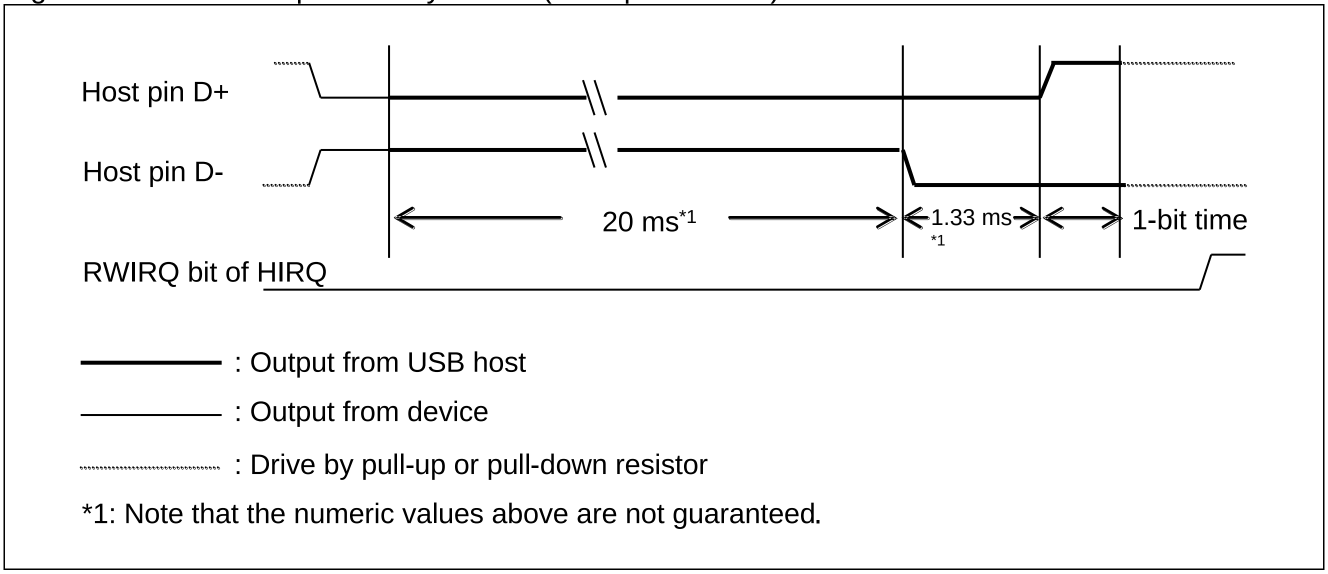
- "0" is set to the SUSP bit of the Host Status Register (HSTATE).

Figure 3-10 Resume operation with register (Full-speed mode)



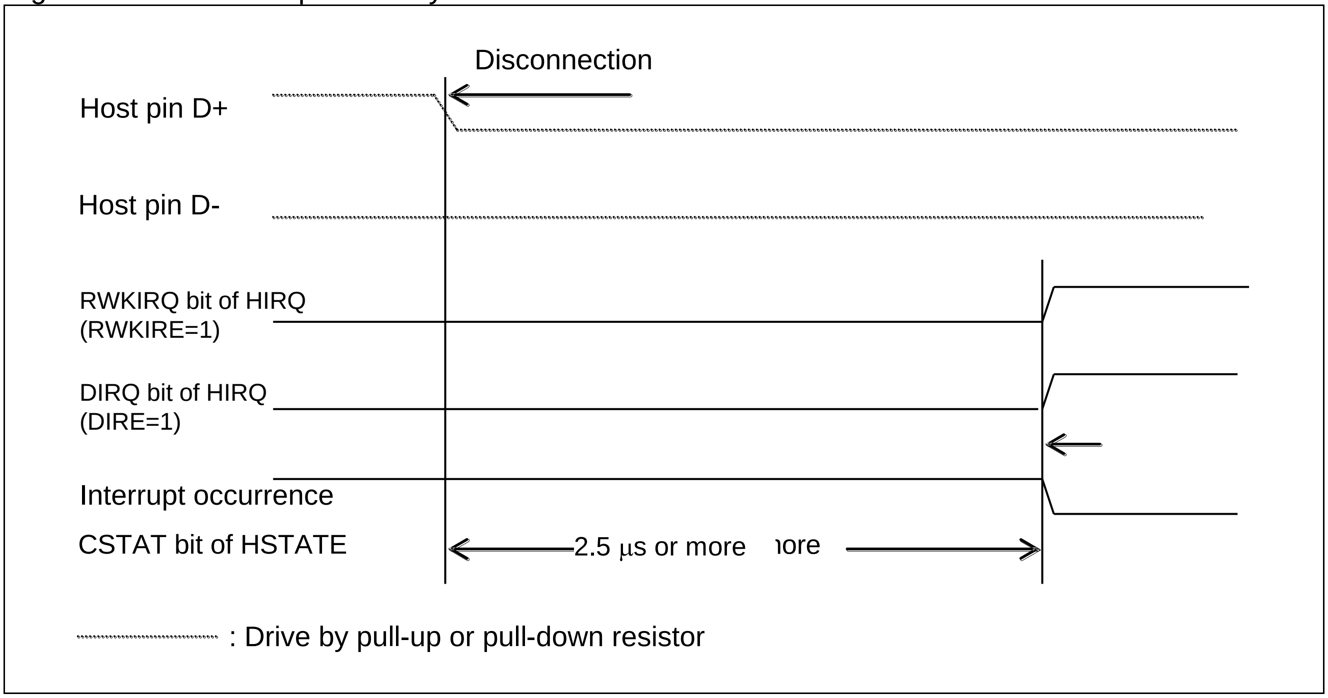
· The state that host pin D+ or D- is placed in the K-state mode has been detected.

Figure 3-11 Resume operation by device (Full-speed mode)



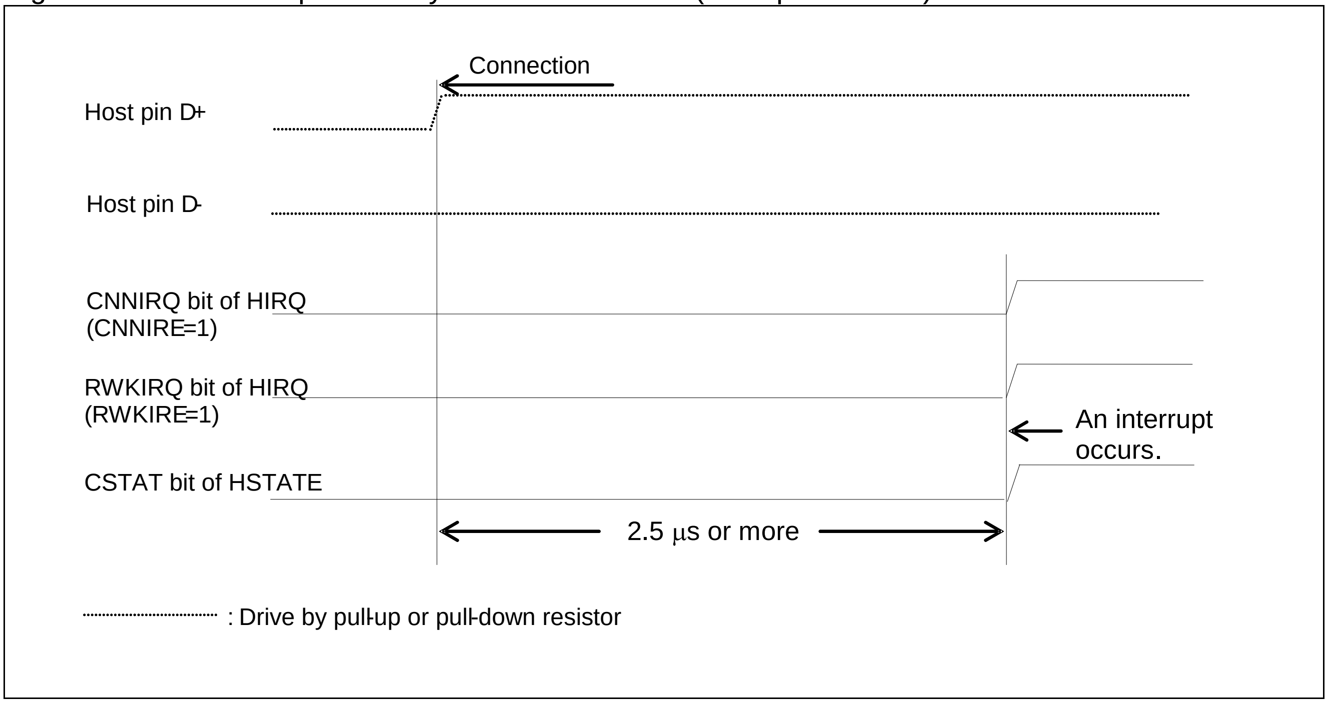
· A device disconnection is detected.

Figure 3-12 Resume operation by device disconnection



· A device connection is detected.

Figure 3-13 Resume operation by device connection (Full-speed mode)



3.11. Device disconnection

The device disconnection timer starts when both the host pins D+ and D- are set to "LOW". If "LOW" is detected for 2.5 μs or more, the CSTAT bit of the Host Status Register (HSTATE) is set to "0".

■ Device disconnection

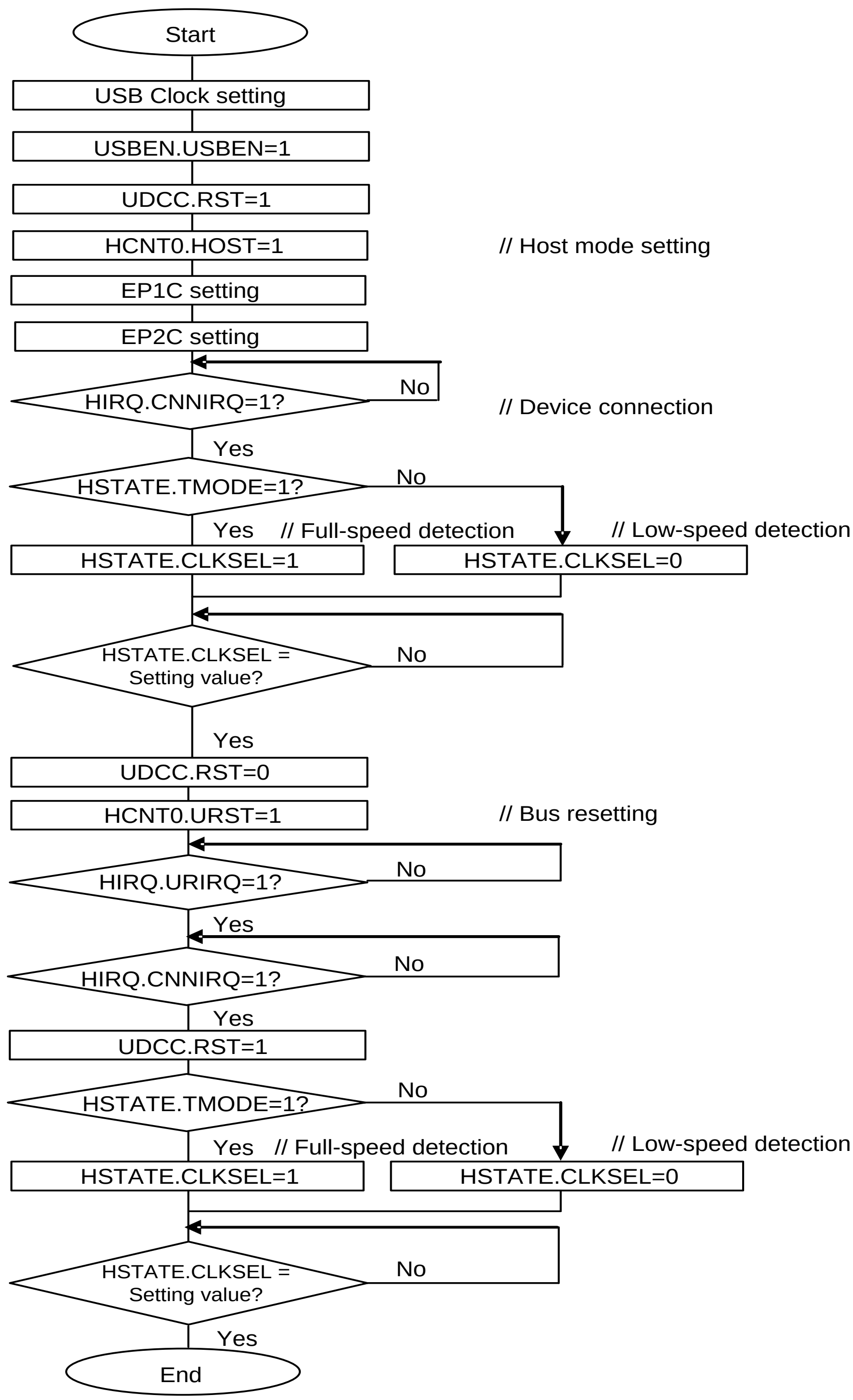
If both the host pins D+ and D- remain set to "LOW" for 2.5 μs or more regardless of the host or device mode, it is judged that the device has been disconnected. This then sets "0" to the CSTAT bit of the Host Status Register (HSTATE) and "1" to the DIRQ bit of the Host Interrupt Register (HIRQ). At this time, if the DIRE bit of Host Control Register 0 (HCNT0) is "1", an interrupt occurs. To clear this interrupt, write "0" to the DIRQ bit of the Host Interrupt Register (HIRQ).

If the USB bus is reset, it is judged that the device has been disconnected. In this case, the CSTAT bit of the Host Status Register (HSTATE) is set to "0", but the DIRQ bit of the Host Interrupt Register (HIRQ) is not set to "1".

4. USB Host Setting Procedure Examples

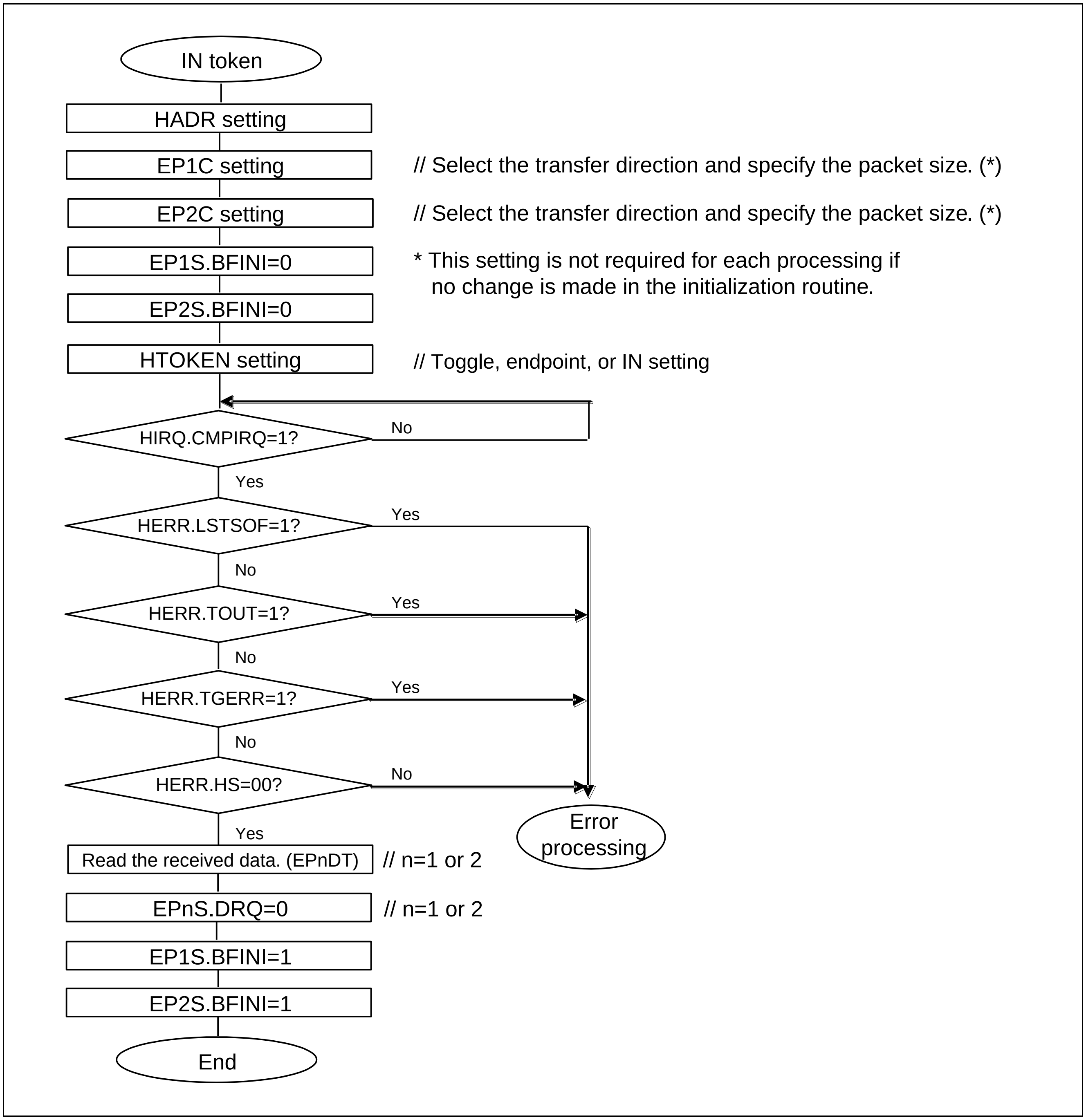
The following shows the flowchart for the USB host tokens.

■ Initialization and device detection

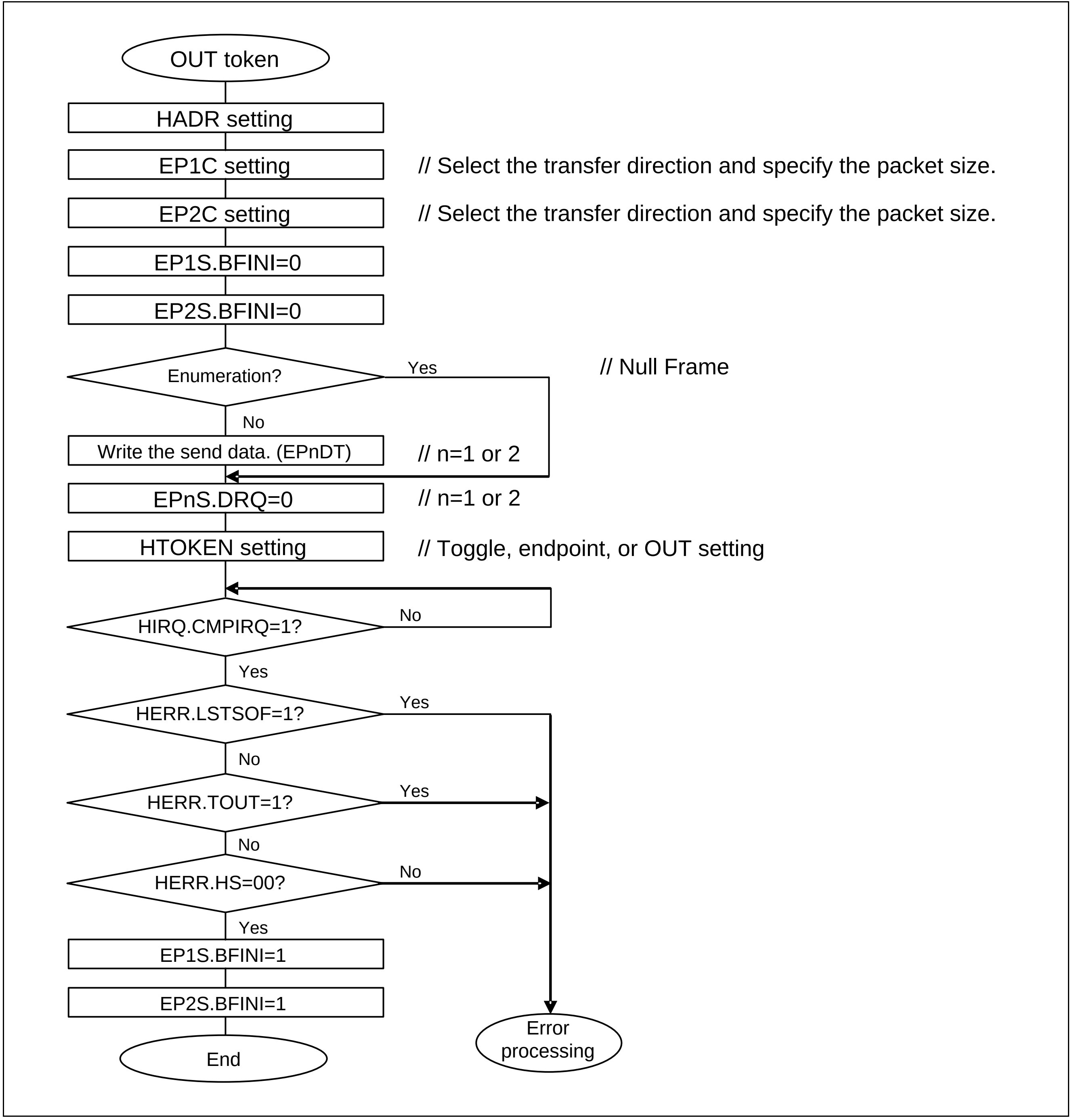


■ IN, OUT, or SETUP token

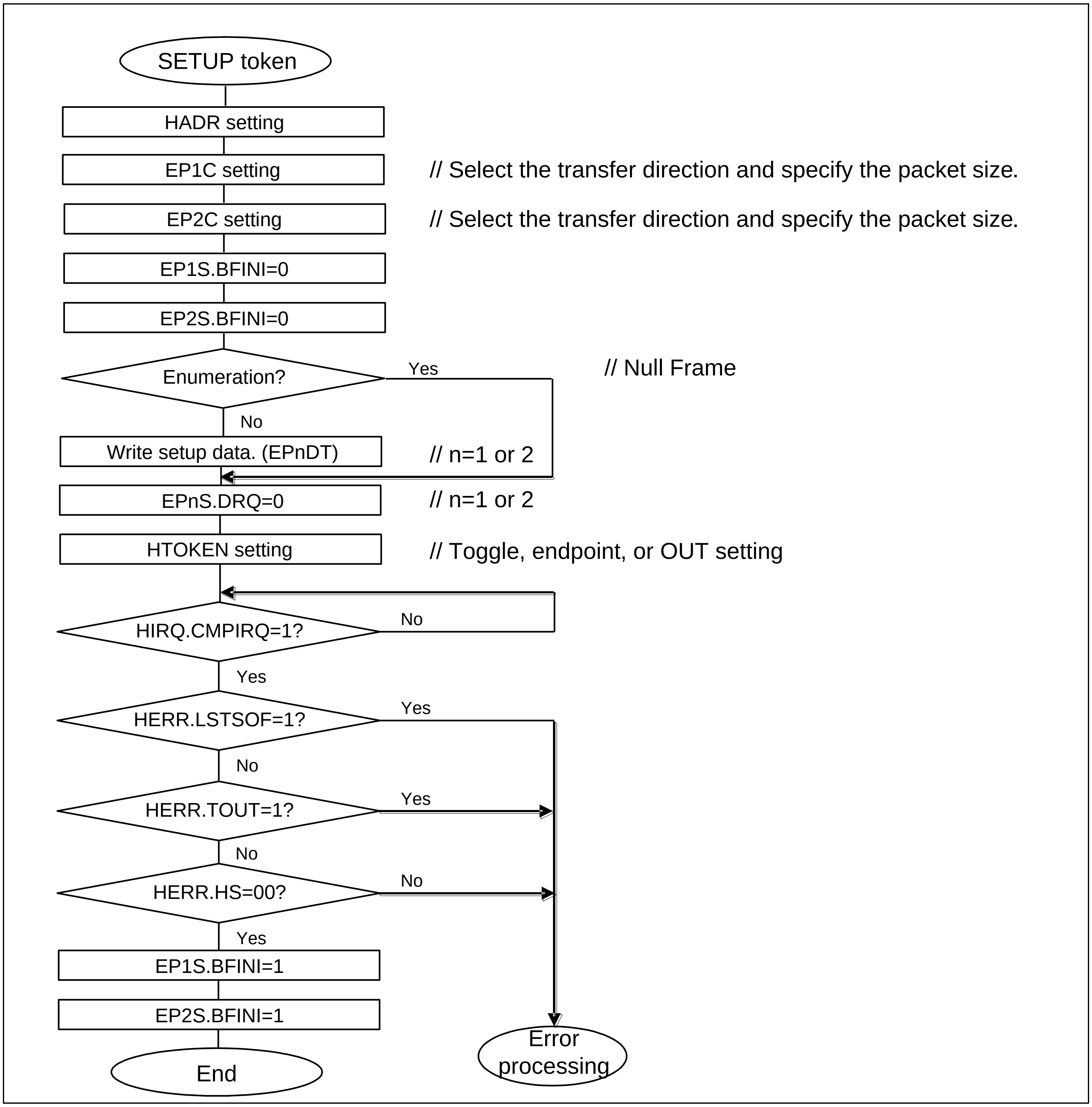
● IN token



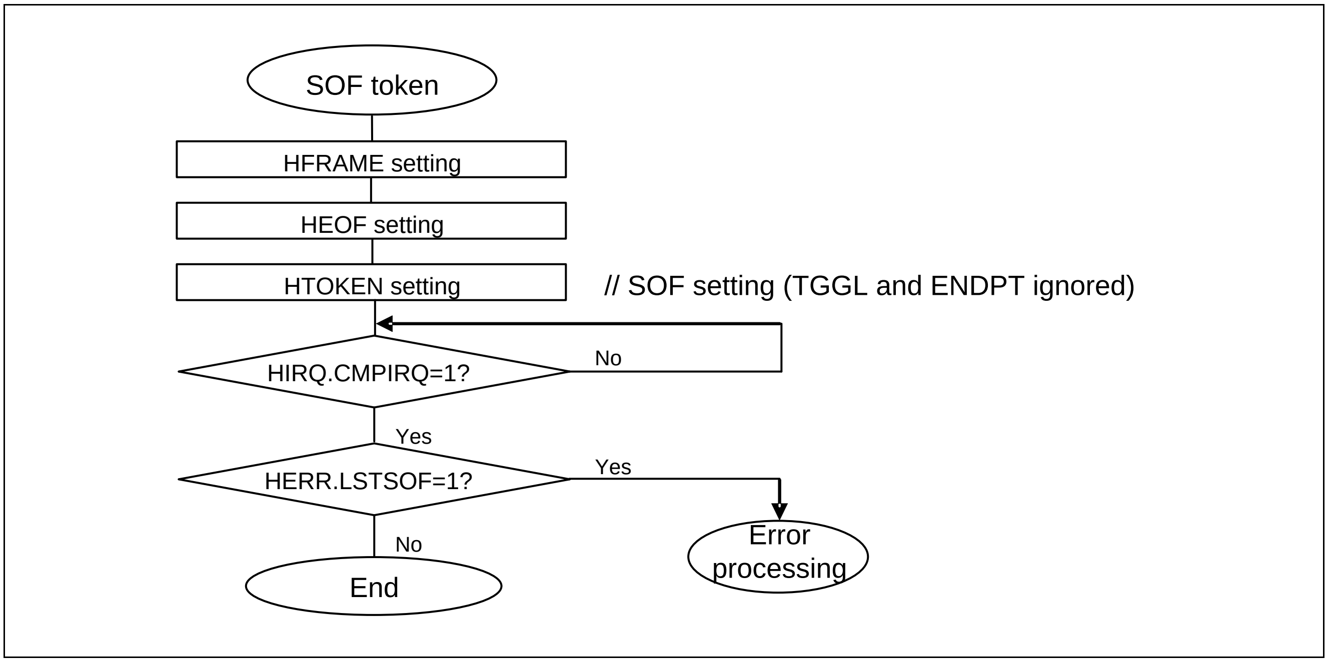
● OUT token


OUT token
HADR setting
EP1C setting
// Select the transfer direction and specify the packet size.
EP2C setting
// Select the transfer direction and specify the packet size.
EP1S.BFINI=0
EP2S.BFINI=0
Enumeration?
Yes
// Null Frame
No
Write the send data. (EPnDT)
// n=1 or 2
EPnS.DRQ=0
// n=1 or 2
HTOKEN setting
// Toggle, endpoint, or OUT setting
HIRQ.CMPIRQ=1?
No
Yes
HERR.LSTSOF=1?
Yes
No
HERR.TOUT=1?
Yes
No
HERR.HS=00?
No
Yes
EP1S.BFINI=1
EP2S.BFINI=1
End
Error processing

- **SETUP token**


SETUP token
HADR setting
EP1C setting
// Select the transfer direction and specify the packet size.
EP2C setting
// Select the transfer direction and specify the packet size.
EP1S.BFINI=0
EP2S.BFINI=0
Enumeration?
Yes
// Null Frame
No
Write setup data. (EPnDT)
// n=1 or 2
EPnS.DRQ=0
// n=1 or 2
HTOKEN setting
// Toggle, endpoint, or OUT setting
HIRQ.CMPIRQ=1?
No
Yes
HERR.LSTSOF=1?
Yes
No
HERR.TOUT=1?
Yes
No
HERR.HS=00?
No
Yes
EP1S.BFINI=1
EP2S.BFINI=1
End
Error processing

■ SOF token


SOF token
HFRAME setting
HEOF setting
HTOKEN setting
// SOF setting (TGGL and ENDPT ignored)
HIRQ.CMPIRQ=1?
No
Yes
HERR.LSTSOF=1?
Yes
No
Error processing
End

5. USB Host Registers

This section explains the configurations and functions of the registers used for the USB host.

■ List of USB host registers

Abbreviation	Register name	Reference
UDCC	UDC Control Register	*
EP1C	EP1 Control Register	*
EP2C	EP2 Control Register	*
EP1S	EP1 Status Register	*
EP2S	EP2 Status Register	*
EP1DTH	EP0 Data Register high-order	*
EP1DTL	EP0 Data Register low-order	*
EP2DTH	EP0 Data Register high-order	*
EP2DTL	EP0 Data Register low-order	*
HCNT0	Host Control Register 0	5.1
HCNT1	Host Control Register 1	
HIRQ	Host Interrupt Register	5.2
HERR	Host Error Status Register	5.3
HSTATE	Host Status Register	5.4
HFCOMP	SOF Interrupt Frame Compare Register	5.5
HRTIMER	Retry Timer Setup Register	5.6
HADR	Host Address Register	5.7
HEOF	EOF Setup Register	5.8
HFRAME	Frame Setup Register	5.9
HTOKEN	Host Token Endpoint Register	5.10

*: See chapter "USB Device".

■ UDCC:RST dependent register bit update timing list

	Register	bit
Register bits to be updated when UDCC:RST=1	HCNT0	HOST
	HSTATE	CLKSEL
	EP1C	EPEN, TYPE, DIR, PKS1
	EP2C	EPEN, TYPE, DIR, PKS2
Register bits initialized when UDCC:RST=1 (Update when UDCC:RST=0)	HCNT0	URST
	HIRQ	TCAN, RWKIRQ, URIRQ, CMPIRQ, CNNIRQ, DIRQ, SOFIRQ
	HERR (All bits)	LSTSOF, RERR, TOUT, CRC, TGERR, STUFF, HS
	HSTATE	SOFBUSY, SUSP
	HFRAME	FRAME0, FRAME1
	HTOKEN (All bits)	TGGL, TKNEN, ENDPT
	EP1S	BFINI, DRQ, SPK
	EP2S	BFINI, DRQ, SPK
Register bits unaffected by UDCC:RST	HCNT0	RWKIRE, URIRE, CMPIRE, CNNIRE, DIRE, SOFIRE
	HCNT1	SOFSTEP, CANCEL, RETRY
	HIRQ	CNNIRQ, DIRQ
	HFCOMP	HFRAMECOMP
	HSTATE	TMODE, CSTAT
	HRTIMER0, 1, 2	RTIMER0, 1, 2
	HADR	Address
	HEOF	EOF0, 1

5.1. Host Control Registers 0 and 1 (HCNT0 and HCNT1)

Host Control Registers 0 and 1 (HCNT0 and HCNT1) are used to specify the USB operation mode and interrupt.

■ Host Control Register 1 (HCNT1)

bit	15	14	13	12	11	10	9	8
Field	Reserved	Reserved	Reserved	Reserved	Reserved	SOFSTEP	CANCEL	RETRY
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	1
Reset enabled or not*	x	x	x	x	x	x	x	x

* : Enables or disables a reset with the RST bit of UDCC. x: Not to be reset. ○: To be reset.

■ Host Control Register 0 (HCNT0)

bit	7	6	5	4	3	2	1	0
Field	RWKIRE	URIRE	CMPIRE	CNNIRE	DIRE	SOFIRE	URST	HOST
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0
Reset enabled or not*	x	x	x	x	x	x	○	x

* : Enables or disables a reset with the RST bit of UDCC. x: Not to be reset. ○: To be reset.

[bit15:11] Reserved: Reserved bits

Always set it to "0".

[bit10] SOFSTEP (SOF STEP) SOF interrupt occurrence selection bit

This is a SOF interrupt occurrence selection bit.

If this bit is set to "1", the SOF interrupt flag (HIRQ:SOFIRQ) is set to "1" each time SOF is sent.

If this bit is set to "0", the set value of the SOF Interrupt Frame Compare Register (HFCOMP) is compared with the low-order eight bits of the SOF frame number. If they match, the SOF interrupt flag (HIRQ:SOFIRQ) is set to "1".

Value	Description
0	An interrupt occurred due to the HFCOMP setting.
1	An interrupt occurred.

<Notes>

- If a SOF token (TKNEN="001") is sent by the setting of the Host Token Endpoint Register (HTOKEN), the SOF interrupt flag (HIRQ:SOFIRQ) is not set to "1" regardless of the setting of this bit.
- This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit9] CANCEL (token CANCEL enable) token cancellation enable bit

This is a token cancellation enable bit.

When "1" is set to this bit, if the target token is written to the Host Token Endpoint Register (HTOKEN) in the EOF area (specified in the EOF Setting Register), its sending is canceled. When "0" is set to this bit, token sending is not canceled even if the target token is written to the register. The cancellation of token sending is detected by reading the TCAN bit of the Host Interrupt Register (HIRQ).

Value	Description
0	Continues a token.
1	Cancels a token.

<Note>

This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit8] RETRY (RETRY enable) retry enable bit

this is a retry enable bit.

If this bit is set to "1", the target token is retried if a NAK or error* occurs. Retry processing is performed during the time that is specified in the Retry Timer Setting Register (HRTIMER).

*: HERR:RERR="1", HERR:TOUT="1", HERR:CRC="1", HERR:TGERR="1", HERR:STUFF="1"

Value	Description
0	Does not retry token sending.
1	Retries token sending.

<Note>

This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit7] RWKIRE (Remove WaKe up Interrupt Request Enable) resume interrupt enable bit

This is a resume interrupt enable bit.

When "1" is set to this bit, an interrupt occurs if the RWKIRQ bit of the Host Interrupt Register (HIRQ) is set to "1". When "0" is set to this bit, an interrupt does not occur even if the RWIRQ bit of the Host Interrupt Register (HIRQ) is set to "1".

Value	Description
0	Disables an interrupt after restarting.
1	Enables an interrupt after restarting.

<Note>

This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit6] URIRE (Usb bus Rest Interrupt Request Enable) bus reset interrupt enable bit

This is a bus reset interrupt enable bit.

When "1" is set to this bit, an interrupt occurs if the URIRQ bit of the Host Interrupt Register (HIRQ) is set to "1". When "0" is set to this bit, an interrupt does not occur even if the URIRQ bit of the Host Interrupt Register (HIRQ) is set to "1".

Value	Description
0	Disables an interrupt after resetting the USB bus.
1	Enables an interrupt after resetting the USB bus.

<Note>

This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit5] CMPIRE (CoMPletion Interrupt Request Enable) token completion interrupt enable bit

This is a token completion interrupt enable bit.

When "1" is set to this bit, an interrupt occurs if the CMPIRQ bit of the Host Interrupt Register (HIRQ) is set to "1". When "0" is set to this bit, an interrupt does not occur even if the CMPIRQ bit of the Host Interrupt Register (HIRQ) is set to "1".

Value	Description
0	Disables an interrupt at completion.
1	Enables an interrupt at completion.

<Note>

This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit4] CNNIRE (CoNNection Interrupt Request Enable) device connection detection interrupt enable bit

This is a device connection detection interrupt enable bit.

When "1" is set to this bit, an interrupt occurs if the CNNIRQ bit of the Host Interrupt Register (HIRQ) is set to "1". When "0" is set to this bit, an interrupt does not occur even if the CNNIRQ bit of the Host Interrupt Register (HIRQ) is set to "1".

Value	Description
0	Disables an interrupt at device connection.
1	Enables an interrupt at device connection.

<Note>

This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit3] DIRE (Disconnection Interrupt Request Enable) device disconnection detection interrupt enable bit

This is a device disconnection detection interrupt enable bit.

When "1" is set to this bit, an interrupt occurs if the DIRQ bit of the Host Interrupt Register (HIRQ) is set to "1". When "0" is set to this bit, an interrupt does not occur even if the DIRQ bit of the Host Interrupt Register (HIRQ) is set to "1".

Value	Description
0	Disables an interrupt at device disconnection.
1	Enables an interrupt at device disconnection.

<Note>

This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit2] SOFIRE (Start Of Frame Interrupt Request Enable) SOF interrupt enable bit

This is a SOF interrupt enable bit.

When "1" is set to this bit, an interrupt occurs if the SOFIRQ bit of the Host Interrupt Register (HIRQ) is set to "1". When "0" is set to this bit, an interrupt does not occur even if the SOFIRQ bit of the Host Interrupt Register (HIRQ) is set to "1".

Value	Description
0	Disables an interrupt when sending SOF.
1	Enables an interrupt when sending SOF.

<Note>

This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit1] URST (Usb bus ReSeT) bus reset bit

This is a bus reset bit.

When "1" is set to this bit, the USB bus is reset. This bit continues to be "1" during USB bus resetting, and changes to "0" when USB bus resetting is ended. If "0" is set to this bit, no processing is performed.

Value	Description
0	Holds the status of the USB bus.
1	Resets the USB bus.

<Notes>

- No processing is performed even if this bit is set to "1" while the RST bit of the UDC Control Register (UDCC) is "1".
- This bit is not allowed to be set to "1" while the SUSP bit of the Host Status Register (HSTATE) is "1" or during token sending.
- The Host Control Register (HCNT0 or HCNT1) is not allowed to be written while this bit is "1".

[bit0] HOST (HOST mode) host mode bit

This is a host mode bit.

When "1" is set to this bit, the USB acts as a host. When "0" is set to this bit, the USB acts as a device.

Value	Description
0	Device mode
1	Host mode

<Notes>

- This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC).
- Change the value of this bit while the RST bit of the UDC Control Register (UDCC) is "1".
- The operation mode does not transition to the required one immediately after it was changed using this bit. Read this bit to check that the operation mode has changed.
- Before changing from the host mode to the device mode, check that the following conditions are satisfied and also set "1" to the RST bit of the UDC Control Register (UDCC).
  - The SOFBUSY bit of the Host Status Register (HSTATE) is set to "0".
  - The TKNEN bits of the Host Token Endpoint Register (HTOKEN) are set to "000".
  - The SUSP bit of the Host Status Register (HSTATE) is set to "0".
- Before changing from the device mode to the host mode, set "1" to the HCONX bit of the UDC Control Register (UDCC), and disconnect the host or HUB.

5.2. Host Interrupt Register (HIRQ)

The Host Interrupt Register (HIRQ) indicates the USB host interrupt request flags. A host interrupt can occur by setting the interrupt enable bit of the Host Control Register (HCNT0 or HCNT1), excluding the TCAN bit.
Host Interrupt Register (HIRQ) should be accessed with a byte access instruction.

bit	7	6	5	4	3	2	1	0
Field	TCAN	Reserved	RWKIRQ	URIRQ	CMPIRQ	CNNIRQ	DIRQ	SOFIRQ
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0
Reset enabled or not*	○	○	○	○	○	x	x	○

* : Enables or disables a reset with the RST bit of UDCC. x: Not to be reset. ○: To be reset.

[bit7] TCAN (Token CANcel flag) token cancellation flag

This is a token cancellation flag.

If this bit is set to "1", it means that token sending is canceled based on the setting of the CANCEL bit of Host Control Register 1 (HCNT1). When this bit is "0", it means that token sending is not canceled. If this bit is written with "0", it is set to "0". However, if this bit is written with "1", its value is ignored.

A read-modify-write access reads the bit as 1.

Value	Description
0	Token has not been canceled.
1	Token has been canceled.

<Notes>

- This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).
- No interrupt occurs even if this bit is set. To cancel this with interrupt processing, check that token sending is canceled during SOF interrupt processing.

[bit6] Reserved: Reserved bit

Always set it to "0".

[bit5] RWKIRQ (Remove WaKe up Interrupt ReQuest) remote Wake-up end flag

This is a remote Wake-up end flag.

If this bit is set to "1", it means that remote Wake-up is ended. When this bit is "0", it has no meaning. If this bit is written with "0", it is set to "0". However, if this bit is written with "1", its value is ignored.

When the RWKIRE bit of Host Control Register 0 (HCNT0) is "1", an interrupt occurs if this bit is set to "1".

A read-modify-write access reads the bit as 1.

Value	Description
0	Issues no interrupt request by restart.
1	Issues an interrupt request by restart.

<Note>

This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit4] URIRQ (Usb bus Reset Interrupt ReQuest) bus reset end flag

This is a bus reset end flag.

If this bit is set to "1", it means that USB bus resetting is ended. When this bit is "0", it has no meaning. If this bit is written with "0", it is set to "0". However, if this bit is written with "1", its value is ignored.

When the URIRE bit of Host Control Register 0 (HCNT0) is "1", an interrupt occurs if this bit is set to "1". A read-modify-write access reads the bit as 1.

Value	Description
0	Issues no interrupt request by USB bus resetting.
1	Issues an interrupt request by USB bus resetting.

<Note>

This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit3] CMPIRQ (CoMPletion Interrupt ReQuest) token completion flag

This is a token completion flag.

If this bit is set to "1", it means that a token is completed. When this bit is "0", it has no meaning. If this bit is written with "0", it is set to "0". However, if this bit is written with "1", its value is ignored.

When the CMPIRE bit of Host Control Register 0 (HCNT0) is "1", an interrupt occurs if this bit is set to "1".

Value	Description
0	Issues no interrupt request by token completion.
1	Issues an interrupt request by token completion.

<Notes>

- This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).
- This bit is not set to "1" even if the TCAN bit of the Host Interrupt Register (HIRQ) changes to "1".
- Take the following steps when this bit is set to 1 by finishing IN token or Isochronous IN taken.
  1) Read HS bit of Host Error Status Register (HERR), then set CMPIRQ bit to 0.
  2) Set DRQIE bit of EPn Status Register (EPnS) (n=1 or 2) to 1 if HS bit of Host Error Status Register (HERR) is equal to 00 and wait until DRQ bit changes to 1.
  Finish the IN token processing if HS bit is not equal to 00.
  3) Read the received data if DRQ bit of EPn Status Register (EPnS) changes to 1.

[bit2] CNNIRQ (CoNNection Interrupt ReQuest) device connection detection flag

This is a device connection detection flag.

If this bit is set to "1", it means that a device connection is detected. When this bit is "0", it has no meaning. If this bit is written with "0", it is set to "0". However, if this bit is written with "1", its value is ignored.

When the CNNIRE bit of Host Control Register 0 (HCNT0) is "1", an interrupt occurs if this bit is set to "1". A read-modify-write access reads the bit as 1.

Value	Description
0	Issues no interrupt request by detecting a device connection.
1	Issues an interrupt request by detecting a device connection.

<Notes>

- This bit is not initialized even if 1 is set to the RST bit of the UDC Control Register (UDCC).
- A device connection is also detected in the device mode.

[bit1] DIRQ (Disconnection Interrupt ReQuest) device disconnection detection flag

This is a device disconnection detection flag.

If this bit is set to "1", it means that a device disconnection is detected. When this bit is "0", it has no meaning. If this bit is written with "0", it is set to "0". However, if this bit is written with "1", its value is ignored.

When the DIRE bit of Host Control Register 0 (HCNT0) is "1", an interrupt occurs if this bit is set to "1".
A read-modify-write access reads the bit as 1.

Value	Description
0	Issues no interrupt request by detecting a device disconnection.
1	Issues an interrupt request by detecting a device disconnection.

<Notes>

- This bit is not initialized even if 1 is set to the RST bit of the UDC Control Register (UDCC).
- A device disconnection is also detected in the device mode.

[bit0] SOFIRQ (Start Of Frame Interrupt ReQuest) SOF starting flag

This is a SOF starting flag.

If this bit is set to "1", it means that SOF token sending is started. When this bit is "0", it has no meaning. If this bit is written with "0", it is set to "0". However, if this bit is written with "1", its value is ignored.

When the SOFIRE bit of Host Control Register 0 (HCNT0) is "1", an interrupt occurs if this bit is set to "1".
A read-modify-write access reads the bit as 1.

Value	Description
0	Issues no interrupt request by starting a SOF token.
1	Issues an interrupt request by starting a SOF token.

<Note>

This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).

5.3. Host Error Status Register (HERR)

The Host Error Status Register (HERR) indicates whether or not an error occurs while sending or receiving data in the host mode.
Host Error Status Register (HERR) should be accessed with a byte access instruction.

bit	15	14	13	12	11	10	9	8
Field	LSTSOF	RERR	TOUT	CRC	TGERR	STUFF	HS	
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	11	
Reset enabled or not*	○	○	○	○	○	○	○	

*: Enables or disables a reset with the RST bit of UDCC. x: Not to be reset. ○: To be reset.

[bit15] LSTSOF (LoST SOF) lost SOF flag

This is a lost SOF flag.

If this bit is set to "1", it means that the SOF token could not be sent in the host mode because other token is in process. When this bit is "0", it means that no lost SOF error is detected. If this bit is written with "0", it is set to "0". However, if this bit is written with "1", its value is ignored.

Value	Description
0	SOF has been sent.
1	SOF sending error

<Note>

This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit14] RERR (Receive Error) receive error flag

This is a receive error flag.

When this bit is set to "1", it means that the received data exceeds the specified maximum number of packets in the host mode. If a receive error is detected, bit13 (TOUT) of this register is also set to "1". When this bit is "0", it means that no error occurs. If this bit is written with "0", it is set to "0". However, if this bit is written with "1", its value is ignored.

Value	Description
0	No receive error has occurred.
1	Maximum packet receive error has occurred.

<Note>

This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit13] TOUT (Time OUT) timeout flag

This is a timeout flag.

If this bit is set to "1", it means that no response is returned to a token from the device within the specified time after the token has been sent in host mode. When this bit is "0", it means that no timeout is detected. When this bit is "0", it means that no error occurs. If this bit is written with "0", it is set to "0". However, if this bit is written with "1", its value is ignored.

Value	Description
0	No timeout has occurred.
1	Timeout has occurred.

<Note>

This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit12] CRC (CRC error) CRC error flag

This is a CRC error flag.

If this bit is set to "1", it means that a CRC error is detected in the host mode. When this bit is "0", it means that no CRC error is detected. If a CRC error is detected, bit13 (TOUT) of this register is also set to "1". When this bit is "0", it means that no CRC error is detected. If this bit is written with "0", it is set to "0". However, if this bit is written with "1", its value is ignored.

Value	Description
0	No CRC error has occurred.
1	CRC error has occurred.

<Note>

This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit11] TGERR (ToGgle ERRor) toggle error flag

This is a toggle error flag.

If this bit is set to "1", it means that the data of this bit does not match the value of the received toggle data. When this bit is "0", it means that no toggle error is detected. If this bit is written with "0", it is set to "0". However, if this bit is written with "1", its value is ignored.

Value	Description
0	No toggle error has occurred.
1	Toggle error has occurred.

<Note>

This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit10] STUFF (STUFFing error) stuffing error flag

This is a stuffing error flag.

If this bit is set to "1", it means that a bit stuffing error is detected. When this bit is "0", it means that no stuffing error is detected. If a stuffing error is detected, bit13 (TOUT) of this register is also set to "1". If this bit is written with "0", it is set to "0". However, if this bit is written with "1", its value is ignored.

Value	Description
0	No stuffing error has occurred.
1	Stuffing error has occurred.

<Note>

This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).

[bit9:8] HS (Hand Shake status) handshake status flags

These are handshake status flags.

These flags indicate the status of a handshake packet to be sent or received.

These flags are set to "NULL" when no handshake occurs due to an error or when a SOF token has been ended with the TKNEN bits of the Host Token Endpoint Register (HTOKEN).

These bits are updated when sending or receiving has been ended.

HS bits change values 11 under the following condition. However, if HS bits are written except the following conditions, the values are ignored.

- HS bits indicate values except 11 and write the value 11 to HS bits.

Table 5-1 Handshake

bit9	bit8	Handshake
0	0	ACK
0	1	NAK
1	0	STALL
1	1	NULL

<Note>

This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).

5.4. Host Status Register (HSTATE)

The Host Status Register (HSTATE) indicates the state of the USB circuit such as a device connection or transfer mode. Note that the setting of the CLKSEL bit is also effective in the device mode.

bit	7	6	5	4	3	2	1	0
Field	Reserved	Reserved	ALIVE	CLKSEL	SOFBUSY	SUSP	TMODE	CSTAT
Attribute	-		R/W	R/W	R/W	R/W	R	R
Initial value	X		0	1	0	0	1	0
Reset enabled or not*	-		x	x	○	○	x	x

*: Enables or disables a reset with the RST bit of UDCC. x: Not to be reset. ○: To be reset.

[bit7:6] Reserved: Reserved bits

The values of these bits are undefined in read mode. Even if "0" or "1" is written to these bits, it has no effect on LSI operations.

[bit5] ALIVE (keep-ALIVE)

This bit is used to specify the keep-alive function in the low-speed mode. If this bit it set to "1" while the CLKSEL bit of the Host Status Register (HSTATE) is "0", SE0 is output instead of SOF. This bit is effective when the CLKSEL bit of the Host Status Register (HSTATE) is "0". If the CLKSEL bit is "1", SOF is output regardless of the setting of the ALIVE bit.

Value	Description
0	SOF output
1	SE0 output (Keep-alive)

[bit4] CLKSEL (CLocK SELect) USB operation clock selection bit

This is a USB operation clock selection bit.

Value	Description
0	Low-speed clock
1	Full-speed clock

<Notes>

- This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC).
- Change the value of this bit while the RST bit of the UDC Control Register (UDCC) is "1".
- The setting of this bit is also effective in the device mode.
  In the device mode, this bit must not be set to 0.
- Use the on-chip bus (HCLK) clock with 13 MHz or more.

[bit3] SOFBUSY (SOF BUSY) SOF busy flag

This is a SOF busy flag.

When a SOF token is sent using the Host Token Endpoint Register (HTOKEN), this bit is set to "1", which means that the SOF timer is active. When this bit is "0", it means that the SOF timer is under suspension. To stop the active SOF timer, write "0" to this bit. However, if this bit is written with "1", its value is ignored.

Value	Description
0	The SOF timer is stopped.
1	The SOF timer is active.

<Notes>

- This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).
- The SOF timer does not stop immediately after "0" has been set to this bit to stop the SOF timer. To check whether or not the SOF timer is stopped, read this bit.

[bit2] SUSP (SUSPend) suspend setting bit

This is a suspend setting bit.

If this bit is set to "1", the USB circuit is placed into the suspend state. If this bit is set to "0" while it is "1" or the USB bus is placed into the k-state mode, the suspend state is released, and the RWIRQ bit of the Host Interrupt Register (HIRQ) is set to "1".

Table 5-2 Suspend setting

Process	Operation
Set to "1".	Suspend
Set "0" while this bit is "1".	Resume
Others	Holds the state.

<Notes>

- This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).
- Do not set this bit to "1" while the USB is active (during USB bus resetting, data transfer, or SOF timer running).
- USB clock must not be stopped in the suspend state.
- If the value of this bit is changed, it is not immediately reflected on the state of the USB bus. To check whether or not the state is updated, read this bit.

[bit1] TMODE (Transmission MODE) transmission mode flag

This is a transmission mode flag.

If this bit is "1", it means that the device is connected in the full-speed mode. When this bit is "0", it means that the device is connected in the low-speed mode. This bit is valid when the CSTAT bit of the Host Status Register (HSTATE) is "1".

Value	Description
0	Low Speed
1	Full Speed

<Notes>

- This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC).
- Use the base clock (HCLK) with 13 MHz or more.

[bit0] CSTAT (Connect STATus) connection status flag

This is a connection status flag.

When this bit is "1", it means that the device is connected. When this bit is "0", it means that the device is disconnected.

Value	Description
0	Device is disconnected.
1	Device is connected.

<Note>

This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC).

5.5. SOF Interrupt Frame Compare Register (HFCOMP)

The SOF Interrupt Frame Compare Register (HFCOMP) is used to specify the data to be compared with the low-order eight bits of a frame number when sending a SOF token. When the SOFSTEP bit of Host Control Register 0 (HCNT0) is "0", the value of this register is compared with that of the low-order eight bits of a frame number. If they match, the SOFIRQ bit of the Host interrupt Register (HIRQ) is set to "1" when starting SOF sending. When the SOFIRE bit of Host Control Register 0 (HCNT0) is "1", an interrupt occurs.

bit	15	14	13	12	11	10	9	8
Field	FRAMECOMP							
Attribute	R/W							
Initial value	00000000							
Reset enabled or not*	x							

*: Enables or disables a reset with the RST bit of UDCC. x: Not to be reset. ○: To be reset.

[bit15:8] FRAMECOMP : frame compare data

These are frame compare data.

These bits are used to specify the data to be compared with the low-order eight bits of a frame number when sending a SOF token.

If the SOFSTEP bit of Host Control Register 0 (HCNT0) is "0", the frame number of SOF is compared with the value of this register when sending a SOF token. If they match, "1" is set to the SOFIRQ bit of the Host Interrupt Register (HIRQ).

The setting of this register is invalid when the SOFSTEP bit of Host Control Register 0 (HCNT0) is "0".

<Note>

This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC).

5.6. Retry Timer Setup Register (HRTIMER)

The Retry Timer Setup Register (HRTIMER) is used to specify the token retry time.

bit	15	14	13	12	11	10	9	8
Field	RTIMER1							
Attribute	R/W							
Initial value	00000000							
Reset enabled or not*	x							

* : Enables or disables a reset with the RST bit of UDCC. x: Not to be reset. ○: To be reset.

bit	7	6	5	4	3	2	1	0
Field	RTIMER0							
Attribute	R/W							
Initial value	00000000							
Reset enabled or not*	x							

* : Enables or disables a reset with the RST bit of UDCC. x: Not to be reset. ○: To be reset.

bit	7(23)	6(22)	5(21)	4(20)	3(19)	2(18)	1(17)	0(16)
Field	Reserved						RTIMER2	
Attribute	-						R/W	
Initial value	X						00	
Reset enabled or not*	-						x	

*: Enables or disables a reset with the RST bit of UDCC. x: Not to be reset. ○: To be reset.

[bit23:18] Reserved: Reserved bits

The values of these bits are undefined in read mode. Even if "0" or "1" is written to these bits, it has no effect on LSI operations.

[bit17:0] HRTIMER0, 1, 2 : Retry timer setting bits

These are retry timer setting bits.

These bits are used to specify the retry time in this register. The retry timer is activated when token sending starts while the RETRY bit of Host Control Register 1 (HCNT1) is "1". The retry time is then decremented by one when a 1-bit transfer clock (12 MHz in the full-speed mode) is output. When the retry timer reaches "0", the target token is sent, and processing is ended.

If a token retry occurs in the EOF area, the retry timer is stopped until SOF sending is ended. After SOF sending has been completed, the retry timer restarts with the value that is set when the timer stopped.

<Notes>

- This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC). If data is written while the RST bit of the UDC Control Register (UDCC) is "1", the written data is ignored.
- Write this register in the host mode. bit15 to bit0 of this register are set to "0" in the device mode. Even if data is written to bit15 to bit0 of this register, it is ignored.

5.7. Host Address Register (HADR)

The Host Address Register (HADR) is used as an address field to send a token.

bit	15	14	13	12	11	10	9	8
Field	Reserved	Address						
Attribute	-	R/W						
Initial value	X	0000000						
Reset enabled or not*	-	x						

* : Enables or disables a reset with the RST bit of UDCC. x: Not to be reset. ○: To be reset.

[bit15] Reserved: Reserved bit

The values of this bit is undefined in read mode. Even if "0" or "1" is written to this bit, it has no effect on LSI operations.

[bit14:8] Address : address bits

These are address bits.

These bits are used to specify a token address.

<Note>

This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC).

5.8. EOF Setup Register (HEOF)

The EOF Setup Register (HEOF) is used to specify the token disable time before sending a SOF token. If both the following conditions are satisfied, a request token is sent after a SOF token has been transferred.

- When the value of the SOF timer is compared with that of this register, it is less than the value of this register.
- An IN, OUT, or SETUP token sending request has been issued.

This is a function to prevent a SOF token generated by hardware from being sent together with other tokens. The time unit of this register is the 1-bit transfer time.

bit	15	14	13	12	11	10	9	8
Field	Reserved		EOF1					
Attribute	-		R/W					
Initial value	X		000000					
Reset enabled or not*	-		x					

* : Enables or disables a reset with the RST bit of UDCC. x: Not to be reset. ○: To be reset.

bit	7	6	5	4	3	2	1	0
Field	EOF0							
Attribute	R/W							
Initial value	00000000							
Reset enabled or not*	x							

*: Enables or disables a reset with the RST bit of UDCC. x: Not to be reset. ○: To be reset.

[bit15:14] Reserved: Reserved bits

The values of these bits are undefined in read mode. Even if "0" or "1" is written to these bits, it has no effect on LSI operations.

[bit13:0] EOF1, EOF0 (End Of Frame) : EOF bits

These are EOF bits.

These bits are used to specify the time to disable token sending before transferring SOF. Specify the time with a margin, which is longer than the one-packet length. The time unit is the 1-bit transfer time.

Setting example: MAXPKT = 64 bytes, full-speed mode

(Token_length + packet_length + header + CRC)×7/6 + Turn_around_time
=(34 bit + 546 bit)×7/6 + 36 bit = 712.7 bit

Therefore, set "0x2C9".

<Note>

This bit is not initialized even if "1" is set to the RST bit of the UDC Control Register (UDCC).

5.9. Frame Setup Register (HFRAME)

The Frame Setup Register (HFRAME) is used to specify a frame number when sending a SOF token. If SOF sending is set to the TKNEN bit of the Host Token Endpoint Register (HTOKEN), the SOF timer is activated. After this, SOF is sent automatically every 1 ms. The Frame Setup Register is automatically incremented by one each time SOF is ended.

bit	15	14	13	12	11	10	9	8
Field			Reserved				FRAME1	
Attribute			-				R/W	
Initial value			X				000	
Reset enabled or not*			-				○	

* : Enables or disables a reset with the RST bit of UDCC. x: Not to be reset. ○: To be reset.

bit	7	6	5	4	3	2	1	0
Field				FRAME0				
Attribute				R/W				
Initial value				00000000				
Reset enabled or not*				○				

*: Enables or disables a reset with the RST bit of UDCC. x: Not to be reset. ○: To be reset.

[bit15:11] Reserved: Reserved bits

The values of these bits are undefined in read mode. Even if "0" or "1" is written to these bits, it has no effect on LSI operations.

[bit10:0] FRAME1, FRAME0 : frame setting bits

These are frame setting bits.
These bits are used to specify a frame number of SOF.

<Notes>

- This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).
- Specify a frame number in this register before setting SOF in the TKNEN bit of the Host Token Endpoint Register (HTOKEN).
- This register is not allowed to be written while the SOFBUSY bit of the Host Status Register (HSTATE) is "1" and a SOF token is in process.

5.10. Host Token Endpoint Register (HTOKEN)

The Host Token Endpoint Register (HTOKEN) is used to specify toggle, endpoint, and token.

bit	7	6	5	4	3	2	1	0
Field	TGGL	TKNEN			ENDPT			
Attribute	R/W	R/W			R/W			
Initial value	0	000			0000			
Reset enabled or not*	○	○			○			

*: Enables or disables a reset with the RST bit of UDCC. x: Not to be reset. ○: To be reset.

[bit7] TGGL (ToGGLe) toggle bit

This is a toggle bit.

This bit is used to set toggle data. Toggle data is sent depending on the setting of this bit. When receiving toggle data, received toggle data is compared with the toggle data indicated by this bit to verify whether or not an error occurs.

Value	Description
0	DATA0
1	DATA1

<Notes>

- This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).
- Set this bit when the TKNEN bits of the Host Token Endpoint Register (HTOKEN) are "000".

[bit6:4] TKNEN (ToKeN ENable) token enable bits

These are token enable bits.

These bits send a token according to the settings. After operation has been ended, the TKNEN bits are set to "000", and the CMPIRQ bit of the Host Interrupt Register (HIRQ) is set to "1". If the CMPIRE bit of Host Control Register 0 (HCNT0) is "1", an interrupt occurs.

The settings of the TGGL and ENDPT bits are ignored when sending a SOF token.

Table 5-3 Token setting

bit6	bit5	bit4	Operation
0	0	0	Sends no data.
0	0	1	Sends SETUP token.
0	1	0	Sends IN token.
0	1	1	Sends OUT token.
1	0	0	Sends SOF token.
1	0	1	Sends Isochronous IN.
1	1	0	Sends Isochronous OUT.
1	1	1	Reserved (Setting disabled)

<Notes>

- This bit is set to the initial value when "1" is set to the RST bit of the UDC Control Register (UDCC).
- The PRE packet is not supported.
- Do not set "100" to the TKNEN bit when the SOFBUSY bit of the Host Status Register (HSTATE) is "1".
- Change the USB to the host mode before writing data to this bit.
- When issuing a token again after the token interrupt flag (CMPIRQ) has been set to "1", wait for 3 cycles or more after a USB transfer clock (12 MHz in the full-speed mode, 1.5 MHz in the low-speed mode) was output, then write data to this bit.
- When the device is disconnected (CSTAT of HSTATE = "0"), token sending is not performed even if data is written to this bit.
- Read the value of TKNEN bit if a new value is written in it. Continue writing in this bit until a retrieved value equals a new value written in. During this checking process, it is needed to prevent any interrupt.
- Take the following steps when CMPIRQ bit of Host Interrupt Register (HIRQ) is set to 1 by finishing IN token or Isochronous IN token.
  1) Read HS bit of Host Error Status Register (HERR), then set CMPIRQ bit to 0.
  2) Set DRQIE bit of EPn Status Register (EPnS) (n=1 or 2) to 1 if HS bit of Host Error Status Register (HERR) is equal to 00 and wait until DRQ bit changes to 1.
  Finish the IN token processing if HS bit is not equal to 00.
  3) Read the received data if DRQ bit of EPn Status Register (EPnS) changes to 1.

[bit3:0] ENDPT (ENDPoinT) endpoint bits

These are endpoint bits.
These bits are used to specify an endpoint to send or receive data to or from the device.

<Note>

This bit is initialized when "1" is set to the RST bit of the UDC Control Register (UDCC).

CHAPTER 4: Ethernet

For the Ethernet, refer to the “Ethernet Part”.

CODE: 9xETHERTOP-E01.2

CHAPTER 5-1: CAN Prescaler

This chapter explains the CAN prescaler.

CODE: 9BFCANPRE-E01.5

1. Overview and Configuration

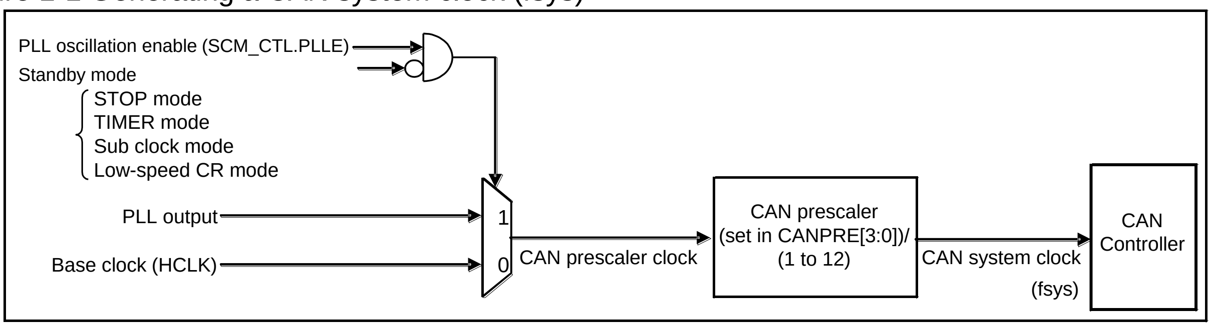
The CAN prescaler generates a CAN system clock (fsys) and supplies it to the CAN.

The CAN prescaler divides a CAN prescaler clock by 1 to 12, and supplies it to the CAN as a CAN system clock (fsys).

Figure 1-1 shows the block diagram of the CAN prescaler.

■ CAN block diagram

Figure 1-1 Generating a CAN system clock (fsys)



■ Explanation of Operations

The CAN prescaler selects the following as a CAN prescaler clock, and supplies it to the CAN after frequency dividing.

- For PLL: PLL output
- For others (including Standby mode in Figure 1-1): Base clock (HCLK)

■ Frequency

Make sure that the CAN system clock output by the CAN prescaler is 16 MHz or less.

2. CAN Prescaler Register

This chapter describes the CAN Prescaler Register.

Abbreviation	Register name	Reference
CANPRE	CAN Prescaler Register	2.1

2.1. CAN Prescaler Register (CANPRE)

The CAN Prescaler Register is used to configure the CAN system clock (fsys) generation prescaler.

■ Register configuration

bit	7	6	5	4	3	2	1	0
Field	Reserved	Reserved			CANPRE			
Attribute	-	-			R/W			
Initial value	0	000			1011			

■ Register functions

[bit7] Reserved: Reserved bit

Be sure to write "0".

[bit6:4] Reserved: Reserved bits

Logical 0 is always read. In the write mode, set "0".

[bit3:0] CANPRE: CAN prescaler setting bits

These bits are used to specify a divided CAN prescaler. The divided clock is supplied as CAN system clock to CAN macro.

Value	Description
0000	CAN prescaler clock is not divided.
0001	CAN prescaler clock is divided to 1/2.
001x	CAN prescaler clock is divided to 1/4.
01xx	CAN prescaler clock is divided to 1/8.
1000	CAN prescaler clock is divided to 2/3. The clock duty is 67%.
1001	CAN prescaler clock is divided to 1/3.
1010	CAN prescaler clock is divided to 1/6.
1011	CAN prescaler clock is divided to 1/12. [Initial value]
110x	CAN prescaler clock is divided to 1/5.
111x	CAN prescaler clock is divided to 1/10.

<Notes>

- Before changing the value of the CAN prescaler setting bit, set the initialization bit (Init) of the CAN Control Register (CTRLR) to "1", and stop all bus operations.
- To use the PLL output as a CAN prescaler clock, set the initialization bit (Init) of the CAN Control Register (CTRLR) to "0" after PLL oscillation has been stabilized.
- Make sure that the CAN system clock output by the CAN prescaler is 16 MHz or less.

CHAPTER 5-2: CAN Controller

This chapter explains CAN.

CODE: FC42L-E02.6

1. Overview

The CAN controller complies with CAN protocol version 2.0A/B, a standard protocol for serial communication. CAN is widely used in various industrial fields such as automobile and factory automation.

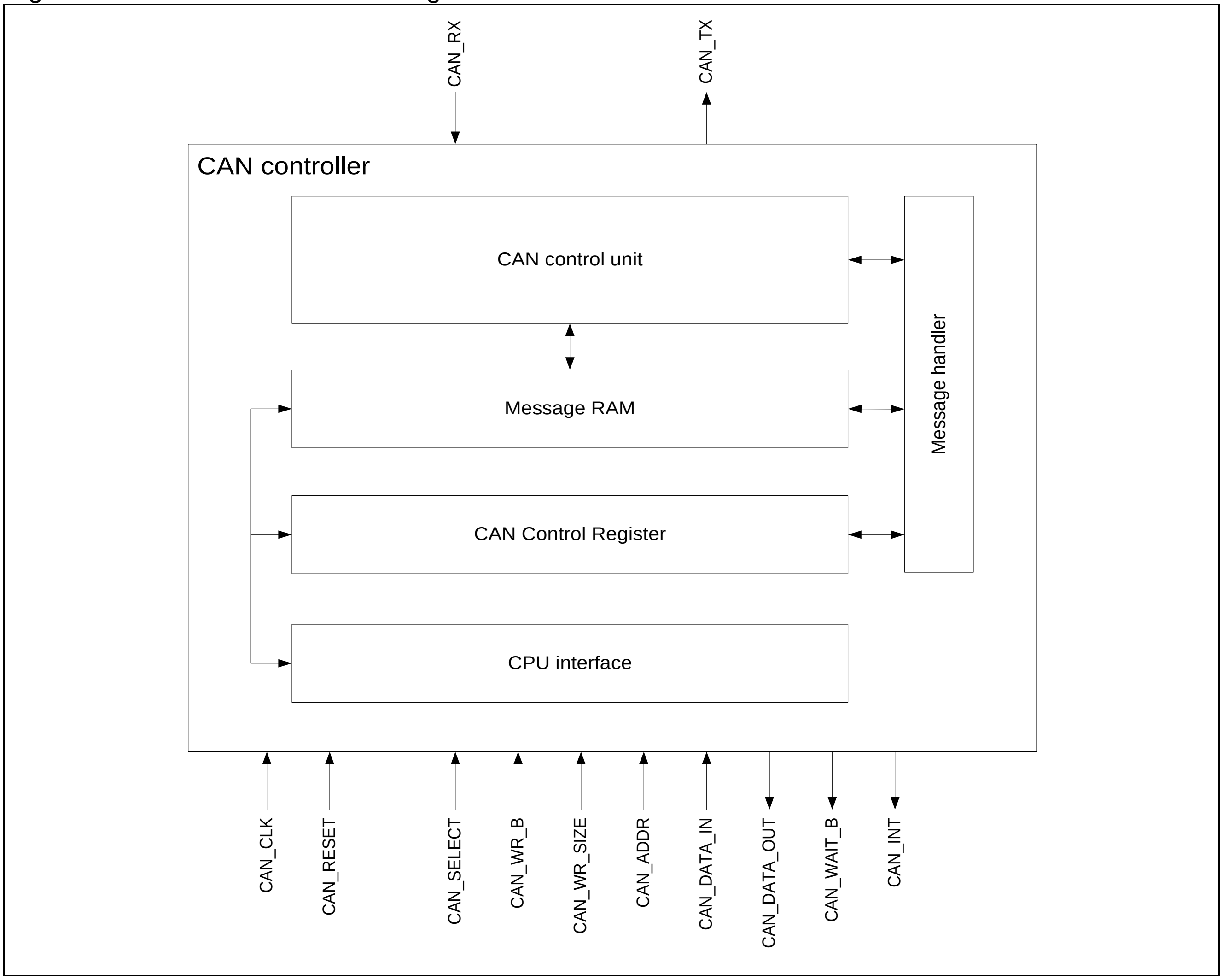
The CAN controller has the following features:

- Supports CAN protocol version 2.0A/B
- Supports a bit rate up to 1 Mbit/s
- Identifier mask for each message object
- Supports programmable FIFO mode
- Maskable interrupt
- Supports 32 message buffers
- Supports programmable loop-back mode for self-test operation
- Read and write from/to the message buffer using interface registers

2. Configuration

Figure 2-1 shows the block diagram of the CAN controller.

Figure 2-1 CAN controller block diagram



- CAN control unit
  Controls the CAN protocol and the serial registers for serial/parallel conversion to transfer send/receive messages.
- Message RAM
  Stores message objects
- Registers
  All registers used by CAN.
- Message handler
  Controls the message RAM and CAN control unit.
- CPU interface
  Controls the internal bus interface.

3. CAN Controller Operations

This section explains the operations and functions of the CAN controller.

Following functions are included:

3.1 Message objects
3.2 Message transmission
3.3 Message reception
3.4 FIFO buffer function
3.5 Interrupt function
3.6 Bit timing
3.7 Test mode
3.8 Software initialization

3.1. Message objects

The following explains message objects and the interface of the message RAM.

■ Message objects

The configuration of message objects in the message RAM (excluding the MsgVal, NewDat, IntPnd, and TxRqst bits) is not initialized by a hardware reset. Initialize the message objects by the CPU, or set the MsgVal bit to disable (MsgVal = "0"). Configure the CAN Bit Timing Register (BTR) while the Init bit in the CAN Control Register (CTRLR) is "1".

To configure message objects, set the message interface registers (the IFx Mask Register, IFx Arbitration Register, IFx Message Control Register, and IFx Data Register), and then write a message number to the corresponding IFx Command Request Register. By writing the message number, the interface register data will be transferred to the addressed message object.

When the Init bit in the CAN Control Register is cleared to "0", the CAN controller starts operation. The received data that have passed acceptance filtering are stored into the message RAM. Messages with pending transmission requests are transferred from the message RAM to the shift register in the CAN controller, and then sent to the CAN bus.

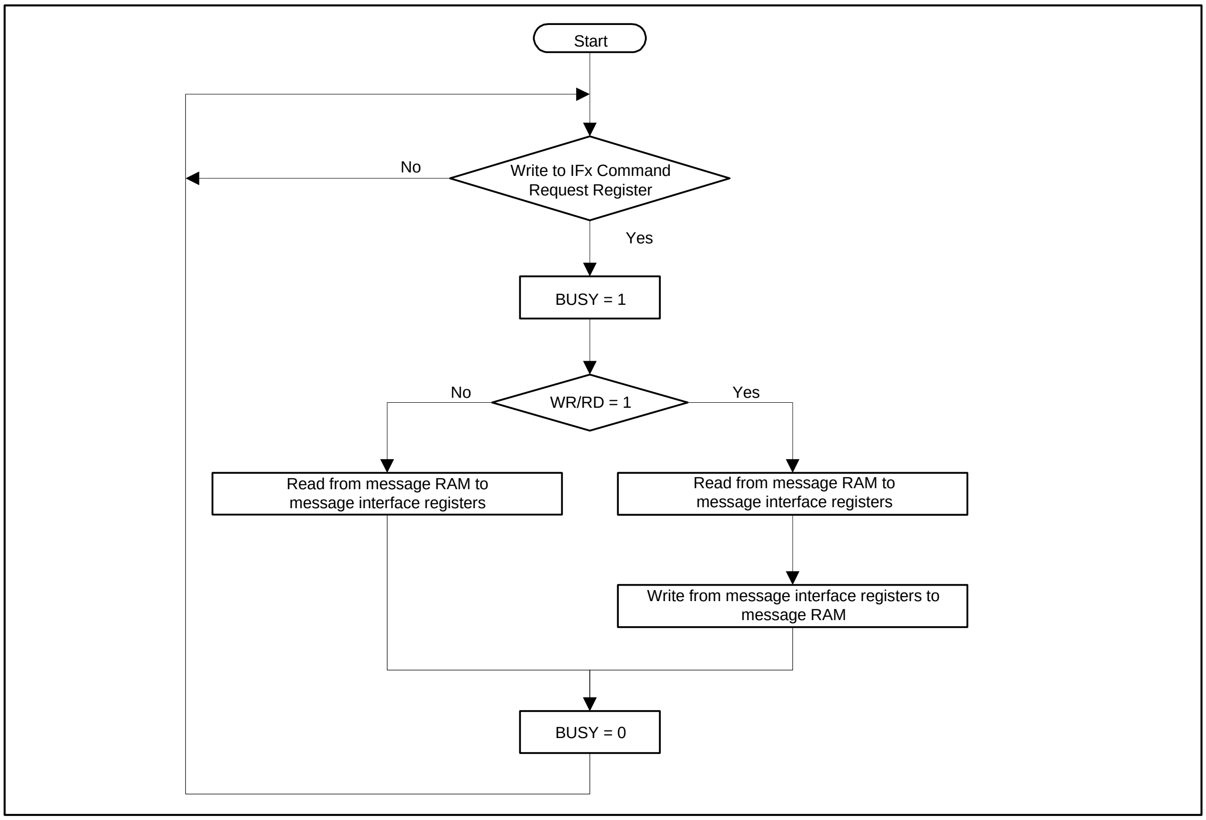
The CPU reads the received messages and updates outgoing messages via message interface registers. The CPU is interrupted according to the configuration of the CAN Control Register and IFx Message Control Register (message object).

■ Data transfer from/to message RAM

When data transfer starts between the message interface registers and message RAM, the BUSY bit in the IFx Command Request Register is set to "1". After the transfer has finished, the BUSY bit is cleared to "0". (See Figure 3-1)

The IFx Command Register selects whether to transfer complete data or only partial data of one message object. The structure of the message RAM does not allow the writing of single bits/bytes of one message object. The complete data of one message object is always written to the message RAM. Therefore, the data from the message interface registers to the message RAM is transferred in a read-modify-write cycle.

Figure 3-1 Data transfer between the message interface registers and message RAM



3.2. Message transmission

The following explains how to configure the send message objects, and about the transmission.

■ Sending messages

If there is no data transfer between the message interface registers and message RAM, the MsgVal bit in the CAN Message Valid Register and the TxRqst bit in the CAN Transmit Request Register are evaluated. A valid message object with the highest priority of pending transmission requests is transferred to the shift register for transmission. Then the NewDat bit of the message object is reset to "0".

When the transmission has finished successfully, and if there is no new data in the message object (NewDat = "0"), the TxRqst bit is reset to "0". If TxIE is set to "1", then the IntPnd bit is set to "1" after a successful transmission. If the CAN controller lost the arbitration on the CAN bus, or if an error occurred during transmission, the message is resent immediately when the CAN bus becomes idle.

■ Transmission priority

The transmission priority of the message objects is determined by the message number. Message object 1 has the highest priority, while message object 32 (the largest number of the installed message objects) has the lowest priority. If two or more transmission requests are pending, they are transferred in the order of corresponding message number from smallest to largest.

<Notes>

- In one of the following conditions, the messages may not be sent until any of the events described below occurs.

  Conditions : (1) A message buffer with the lowest priority is used for transmission.
  (2) The TxRqst bit was previously set to "1", but is set to "0" to abort transmission.
  (3) The TxRqst bit is set to "1" again at the timing of (2).

  Events : - A valid message flows on the CAN bus.
  - A transmission request is issued to another message buffer.
  - CAN is initialized by the Init bit.

  If canceling the transmission is required to suit system operations, execute the following steps.

  1. Execute one of the following steps.
     - Do not use a message buffer with the lowest priority as a send message buffer.
     - After aborting the transmission, generate any of the above events.

  2. Set the TxRqst bit to "1" again.

- If the message objects of ID28 to ID0, DLC3 to DLC0, Xtd, and Data7 to Data0 are changed while the TxRqst bit is "1", message objects before and after the change may be mixed for transmission, or the message objects after the change may not be transmitted. Therefore, be sure to change them while the TxRqst bit is "0".

■ Configuring a send message object

Table 3-1 shows how a send object should be initialized.

Table 3-1 Initialization of a send message object

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	1	0	0	0	appl.	0	appl.	0

The IFx Arbitration Register (ID28 to ID0 and Xtd bit), given by the application, defines the ID and the type of the outgoing message.

If the standard frame (11-bit ID) is set, then ID28 to ID18 are used, and ID17 to ID0 are ignored. If the extended frame (29-bit ID) is set, then ID28 to ID0 are used.

If TxIE bit is set to "1", then the IntPnd bit is set to "1" after a successful transmission of the message object.

If the RmtEn bit is set to "1", the TxRqst bit is set to "1" after receiving the corresponding remote frame, and a data frame is sent automatically.

The data register (DLC3 to DLC0, Data0 to Data7) settings are given by the application.

When UMask is set to "1", the IFx Mask Register (Msk28 to Msk0, UMask, MXtd, and MDir bits) is used to receive remote frames with the IDs grouped by the mask setting, and then enable the transmission (by setting the TxRqst bit to "1"). For details, see Remote Frame in "3.3 Message reception".

<Note>

The Dir bit in the IFx Mask Register must not be mask-enabled.

■ Updating a send message object

The CPU can update the data of a send message object via the message interface registers.

The send message object data is written by four bytes of the corresponding IFx data register (in the unit of IFx data register A or IFx data register B). Therefore, the send message object cannot be changed by a single byte.

To update 8-byte data, write 0x0087 to the IFx Command Mask Register, and the message number to the IFx Command Request Register. This concurrently updates the send message object data (of 8-byte) and write "1" to the TxRqst bit.

If both the NewDat and TxRqst bits are set to "1", the NewDat bit is reset to "0" once the transmission is started.

<Notes>

- To update data, update it by four bytes of the IFx Data Register A or IFx Data Register B.
- If the message objects of ID28 to ID0, DLC3 to DLC0, Xtd, and Data7 to Data0 are changed while the TxRqst bit is "1", message objects before and after the change may be mixed for transmission, or the message objects after the change may not be transmitted. Therefore, be sure to change them while the TxRqst bit is "0".

3.3. Message reception

The following explains how to configure the receive message object and about the reception.

■ Acceptance filtering for received messages

When the arbitration and control field (ID + IDE + RTR + DLC) of a message is completely shifted into the shift register of the CAN controller, scanning of the message RAM is started to compare matching with a valid message object.

Then the arbitration field and mask data (including MsgVal, UMask, NewDat, and EoB) are loaded from a message object in the message RAM, and the message object is compared with the arbitration field of the shift register including mask data.

This operation is repeated "until a matching is detected between a message object and the arbitration field of the shift register", or "until the last word of the message RAM is reached."When a matching is detected, scanning of the message RAM is stopped, and the CAN controller processes data depending of the type of the received frame (data frame or remote frame).

■ Reception priority

The reception priority of the message objects is determined by the message number. Message object 1 has the highest priority, while message object 32 (the largest number of the installed message objects) has the lowest priority. If two or more objects are matched in the acceptance filtering, therefore, the object with the smallest message number becomes the receive message object.

■ Data frame reception

The CAN controller transfers the received message from the shift register into the message RAM of the message object matched in the acceptance filtering. The stored data includes all arbitration fields and the data length code as well as data bytes. This is implemented (to keep the ID and the data bytes) even if the IFx Mask Register is set to be masked.

The NewDat bit is set to "1" upon the reception of new data. When the CPU reads the message object, reset the NewDat bit to "0". If the NewDat bit has already been set to "1" upon the reception of a message, the MsgLst is set to "1" indicating that the previous data was lost.

If the RxIE bit has been set to "1", reception of a message buffer causes the IntPnd bit in the CAN Interrupt Pending Register to be set to "1". Then the TxRqst bit of the message object is reset to "0". This is implemented to prevent transmission of a remote frame when the requested data frame is received during the transmission.

■ Remote frame

One of the following three operations is selected when a remote frame is received. The selection depends on how the matching message object is configured.

1. Dir = "1" (Direction = Send), RmtEn = "1", UMask = "1" or "0"
   Receives the matched remote frame, sets only the TxRqst of this message object to "1", and automatically replies (sends) data frame to the remote frame. (Message objects other than TxRqst bit remain unchanged.)

2. Dir = "1" (Direction = Send), RmtEn = "0", UMask = "0"
   Does not receive an incoming remote frame, even if it matches the message object, and disables the remote frame. (The TxRqst bit of the message object remains unchanged.)

3. Dir = "1" (Direction = Send), RmtEn = "0", UMask = "1"
   If an incoming remote frame matches the message object, the TxRqst bit of the message object is reset to "0", and the remote frame is handled as if it were a received data frame. The received arbitration field and control field (ID + IDE + RTR + DLC) are stored into the message object in the message RAM, and the NewDat bit of this message object is set to "1", The data field of the message object remains unchanged.

■ Configuring a receive message object

Table 3-2 shows how a receive message object should be initialized.

Table 3-2 Initialization of a receive message object

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	0	0	0	appl.	0	0	0	0

The IFx Arbitration Register (ID28 to ID0 and Xtd bit) is given by the application. The register defines the ID and the type of a received message, used for the acceptance filtering.

If the standard frame (11-bit ID) is set, then ID28 to ID18 are used, and ID17 to ID0 are ignored. When a standard frame is received, ID17 to ID0 are reset to "0". If the extended frame (29-bit ID) is set, then ID28 to ID0 are used.

When the RxIE has been set to "1", and when a received data frame is stored into the message object, then the IntPnd bit is set to "1".

The data length code (DLC3 to DLC0) is given by the application. When the CAN controller stores the received data frame into the message object, it stores the received data length code and eight bytes data. If the data length code is less than eight, undefined data is written to the remaining bytes of the message object.

When UMask is set to "1", the IFx Mask Register (Msk28 to Msk0, UMask, MXtd, and MDir bits) is used to allow the reception of data frames with the IDs grouped by the mask setting. For details, see Data Frame Reception in "3.3 Message reception".

<Note>

The Dir bit in the IFx Mask Register must not be mask-enabled.

■ Handling a received message

The CPU can read a received message any time via the message interface registers.

The following shows an example of handling a received message. Write "0x007F" to the IFx Command Register, and a message number of the message object to the IFx Command Request Register. This procedure transfers a received message of the specified message number from the message RAM to the message interface registers. Then the NewDat bit and IntPnd bit of the message object can be cleared to "0" according to the configuration of the IFx Command Mask Register.

An incoming message is received if it is matched in the acceptance filtering. If the message object uses a mask for acceptance filtering, the masked data is excluded from the acceptance filtering to determine whether or not the message should be received.

The NewDat bit indicates whether a new message has been received since the last time the message object was read.

The MsgLst bit indicates that the previous received data was lost because the next data is received before the previous data is read from the message object. The MsgLst bit is not automatically reset.

During transmission of a remote frame, if a data frame matched in the acceptance filtering is received, the TxRqst bit is automatically reset to "0".

3.4. FIFO buffer function

The following explains the configuration of a FIFO buffer of the message object and its operations in handling received messages.

■ Configuration of FIFO buffer

The configuration of the receive message object belonging to a FIFO buffer is the same as that of a receive message object except the EoB bit. (See Configuring a Receive Message Object in "3.3 Message reception".)

A FIFO buffer is used by concatenating two or more receive message objects. To store received messages into this FIFO buffer, the ID and the mask settings of the receive message objects must be matched when they are used.

The first receive message object of the FIFO buffer has the lowest message number, i.e., the highest priority. In the last receive message object of the FIFO buffer, set "1" to the EoB bit to indicate that the object is the end of the FIFO buffer block. (Except in the last message object, the EoB bit in each message object that uses the FIFO buffer configuration must be set to "0".)

<Notes>

- Be sure to configure the same settings for the ID and the masks of message objects used in the FIFO buffer.
- When the FIFO buffer is not used, be sure to set the EoB bit to "1".

■ Receiving messages using FIFO buffers

A received message, when it matches the FIFO buffer ID, is stored into the receive message object in the FIFO buffer with the lowest message number.

When a message is stored into the receive message object in the FIFO buffer, the NewDat bit of this receive message object is set to "1". When the NewDat bit is set in receive message object while the EoB bit is set to "0", the receive message object is protected until the last receive message object (with EoB bit = "1") is reached. Meanwhile, the CAN controller does not write to the FIFO buffer.

When both of the following conditions are met, the next incoming message is written to the last message object and therefore overwrites the previous message.

- Valid data is stored into the last FIFO buffer
- The NewDat bit of the receive message object is not written by "0" (to release the write protect)

If "0" is not written to the NewDat bit (to release the write protect) of the receive message object while valid data is stored into the last FIFO buffer, the next incoming message is written to the last message object and overwrites the previous message.

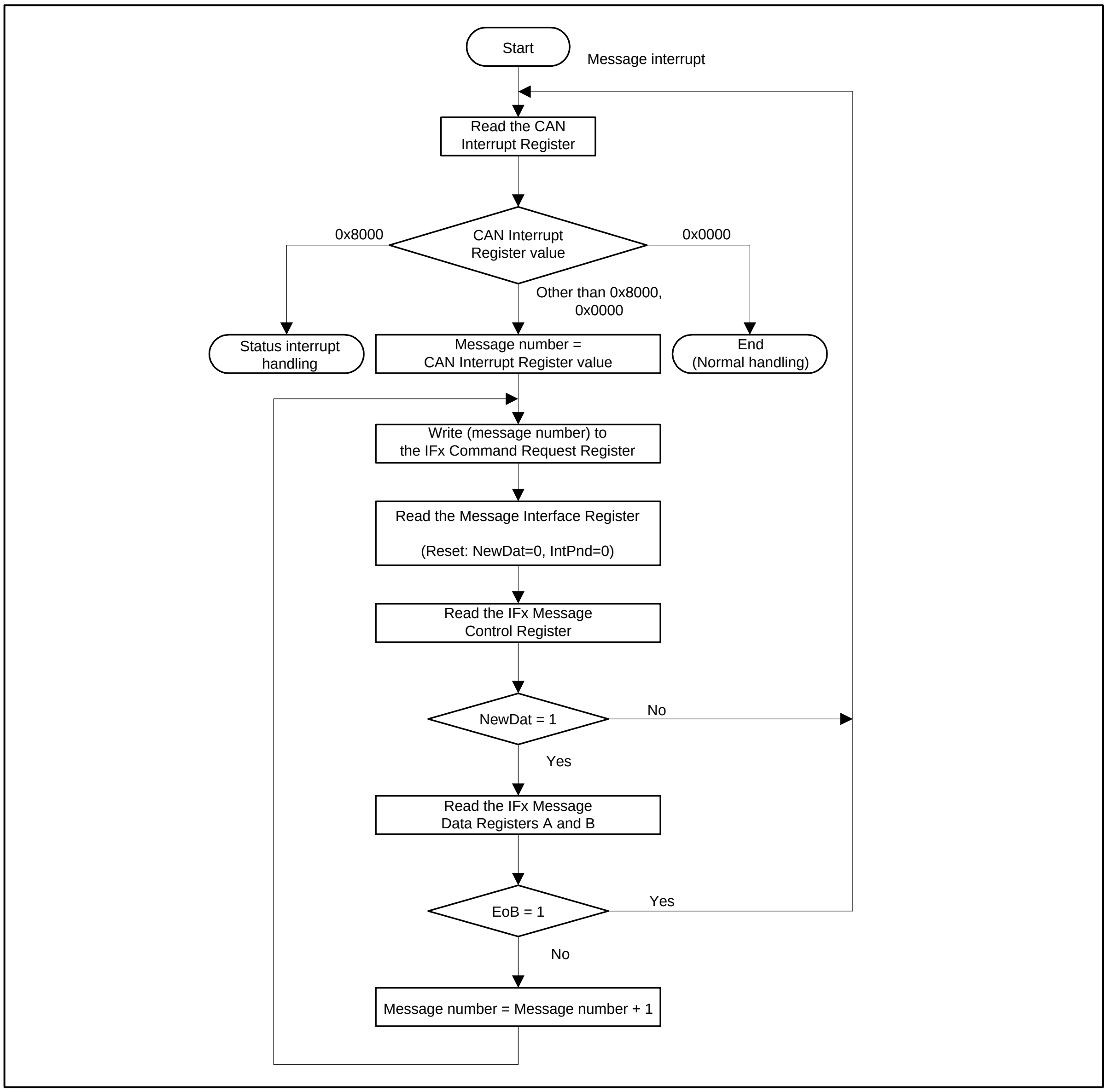
■ Reading from FIFO buffer

To read the contents of a receive message object, the CPU transfers the object to the Message Interface Register by writing the received message number to the IFx Command Request Register. Then, set WR/RD in the IFx Command Mask Register to "0" (read), set TxRqst/NewDat = 1, IntPnd = 1, and set the NewDat bit and IntPnd bit to "0".

To assure the correct FIFO buffer function, be sure to first read a receive message object in the FIFO buffer with the lowest message number, and then other objects in ascending order.

Figure 3-2 shows how the CPU handles the message objects the FIFO buffer concatenates.

Figure 3-2 CPU handling of FIFO buffer



3.5. Interrupt function

The following explains the interrupt handing using the status interrupt (IntId = 0x8000) and message interrupt (IntId = Message number).

If two or more interrupts are pending, the CAN Interrupt Register points to a pending interrupt code with the highest priority. The chronological order of the interrupt codes are neglected, and the interrupt code with the highest priority is always shown. The interrupt code is retained until the CPU clears it.

The status interrupt (0x8000 of the IntId bit) has the highest priority.

Priority of message interrupts is determined by the message number. A smaller number has a higher priority while the larger the lower.

A message interrupt is cleared by clearing the IntPnd bit of the message object. A status interrupt is cleared by reading the CAN Status Register.

The IntPnd bit in the CAN interrupt Pending Register indicates whether an interrupt has been caused. When no interrupts are pending, the IntPnd bit retains "0".

While the IE bit in the CAN Control Register, and the TxIE bit and RxIE bit in the IFx Message Control Register are set to "1", if the IntPnd bit turns to "1", then the interrupt line to the CPU becomes active. The interrupt line remains active until the CAN Interrupt Pending Register is cleared to "0" (the interrupt factor is reset) or the IE bit in the CAN Control Register is reset to "0".

The 0x8000 value of the CAN Interrupt Register indicates that the CAN Status Register has been updated by the CAN controller. This interrupt has the highest priority. The interrupt by updating the CAN Status Register can enable or disable the setting of the CAN Interrupt Register using the EIE bit and SIE bit in the CAN Control Register. The interrupt line to the CPU can be controlled by the IE bit in the CAN Control Register.

A write access from the CPU can update (reset) the RxOk bit, TxOk bit, and LEC bit in the CAN Status Register. However, the write access cannot generate or reset an interrupt.

Except the 0x8000 and 0x0000 values, the CAN Interrupt Register indicates that a message interrupt is pending, and that the interrupt has the highest priority.

The CAN Interrupt Register is updated even when IE is reset.

The factor of a message interrupt to the CPU can be checked from the CAN Interrupt Register or CAN Interrupt Pending Register. (See "4.5 Message handler registers") When clearing a message interrupt, the message data can be read concurrently. If a message interrupt indicated by the CAN Interrupt Register is cleared, the CAN Interrupt Register sets another interrupt with the next higher priority. This waits for the next interrupt handling. If no interrupts are pending, the CAN Interrupt Register shows the 0x0000 value.

<Notes>

- A status interrupt (IntId = 0x8000) is cleared by a read access to the CAN Status Register.
- A write access to the CAN Status Register will not generate a status interrupt (IntId = 0x8000).

3.6. Bit timing

The following provides the overview of the bit timing and explains about the bit timing in the CAN controller.

Each CAN node in the CAN network has its own clock generator (usually a quartz oscillator). The time parameter of the bit time can be configured individually for each CAN node. Even if each CAN node's oscillator has a different cycle (fosc), a common bit rate can be generated.

The oscillator frequencies vary slightly because of changes in temperature or voltage, or deterioration of components. As long as the frequencies vary only within the tolerance range (df) of the oscillators, the CAN nodes can compensate for the different bit rates by resynchronizing to the bit stream.

The bit time can be divided into four segments according to the CAN specifications (see Figure 3-3), into the synchronization segment (Sync_Seg), the propagation time segment (Prop_Seg), the phase buffer segment 1 (Phase_Seg1), and the phase buffer segment 2 (Phase_Seg2). Each segment consists of the programmable number of time quanta (See Table 3-3). The basic unit of the time quantum (tq) is defined by CAN controller's system clock "fsys" and the baud rate prescaler (BRP).

tq = BRP / fsys

CAN's system clock "fsys" is the frequency of its clock input (See Figure 2-1). Synchronization segment Sync_Seg is a timing in the bit time where edges of the CAN bus level are expected to occur. Propagation time segment Prop_Seg compensates for the physical delay times within the CAN network. Phase buffer segments Phase_Seg1 and Phase_Seg2 must specify the sampling points. Resynchronization jump width (SJW) must define the width within which resynchronization can move the sampling point to compensate for edge phase errors.

Figure 3-3 Bit timing

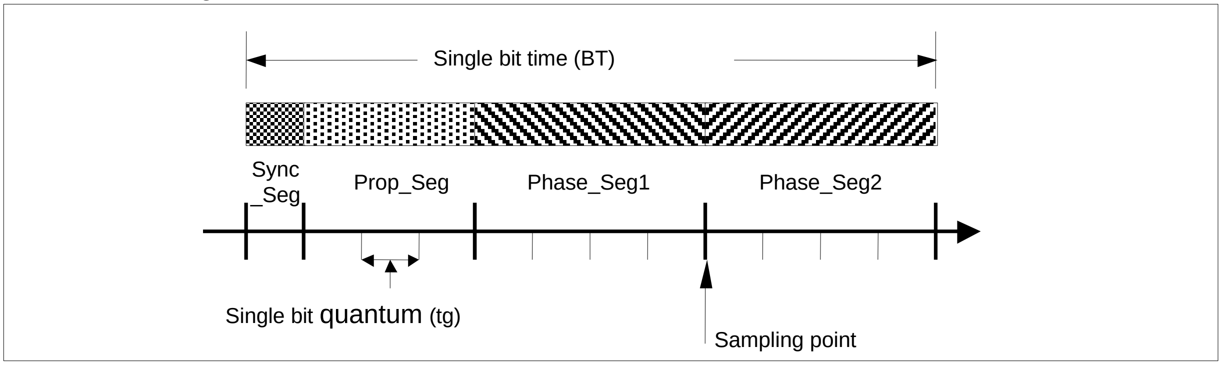


Table 3-3 CAN bit time parameters

Parameter	Range	Function
BRP	[1 to 32]	Defines the length of time quantum tq.
Sync_Seg	1 tq	Fixed length. Synchronization to system clock.
Prop_Seg	[1 to 8] tq	Compensates for the physical delay times.
Phase_Seg1	[1 to 8] tq	Assures edge phase errors before the sampling point. May be prolonged temporarily by synchronization.
Phase_Seg2	[1 to 8] tq	Assures edge phase errors after the sampling point. May be shortened temporarily by synchronization.
SJW	[1 to 4] tq	Resynchronization jump width. Will not be longer than either of the phase buffer segments.

The following shows the bit timing in the CAN controller.

Figure 3-4 The bit timing in the CAN controller

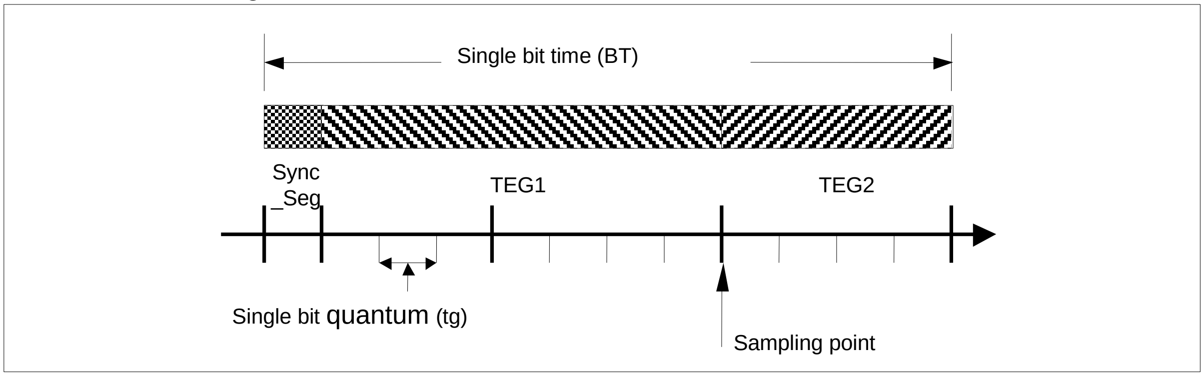


Table 3-4 CAN controller parameters

Parameter	Range	Function
BRPE, BRP	[0 to 1023]	Defines the length of time quantum tq. Can extend the prescaler by up to 1024 by the Bit Timing Register and the Prescaler Extension Register.
Sync_Seg	1 tq	Synchronization to system clock. Fixed length.
TSeg1	[1 to 15] tq	A time segment before the sampling point. Equivalent to Prop_Seg and Phase_Seg1. Can be controlled by the Bit Timing Register.
TSeg2	[0 to 7] tq	A time segment after the sampling point. Equivalent to Phase_Seg2. Can be controlled by the Bit Timing Register.
SJW	[0 to 3] tq	Resynchronization jump width. Can be controlled by the Bit Timing Register.

The following shows the relations among the parameters:

tq = ([BRPE, BRP]+1) / fsys

BT = SYNC_SEG + TEG1 + TEG2

= (1 + (TSeg1 + 1) + (TSeg2 + 1)) × tq

= (3 + TSeg1 + TSeg2) × tq

3.7. Test mode

The following explains how to configure test mode, and about its operations.

■ Test mode setting

Test mode is entered by setting the Test bit in the CAN Control Register to "1". In test mode, the Tx1, Tx0, LBack, Silent, and Basic bits in the CAN Test Register are enabled.

When the Test bit in the CAN Control Register is set to "0", all test register functions are disabled.

■ Silent mode

The CAN controller can be set in silent mode by programming the Silent bit in the CAN Test Register to "1".

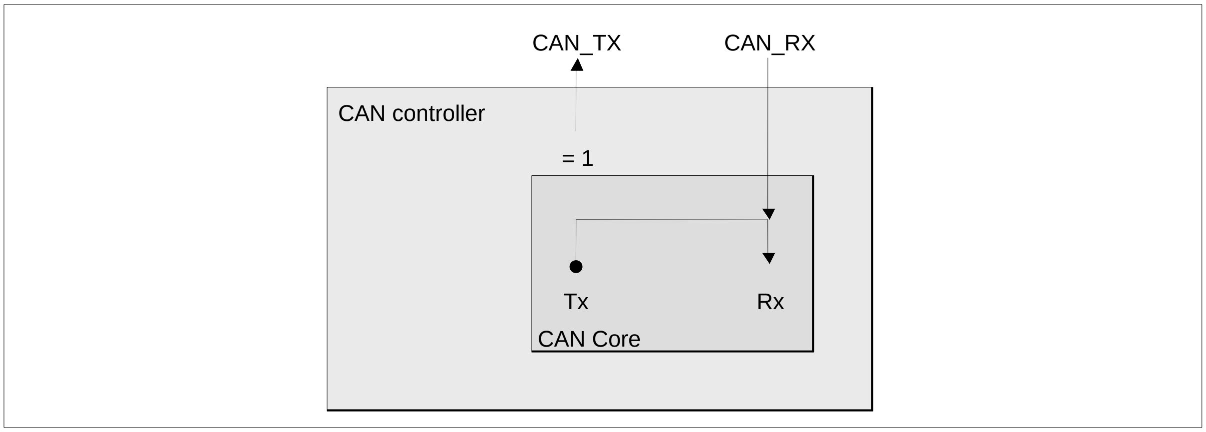
In silent mode, the CAN controller can receive data frames and remote frames, but only outputs recessive bits onto the CAN bus and does not send messages and ACK.

When the CAN controller is required to send dominant bits (ACK bits, overload flags, active error flags), the CAN controller uses the internal rerouting circuit to send them to the RX side. In this operation, the RX side can receive dominant bits rerouted inside the CAN controller even when the CAN bus remains in a recessive state.

In silent mode, the analysis of CAN bus traffic is possible without being affected by transmission of the dominant bits (ACK bits, error flags).

Figure 3-5 shows the connection of the CAN_TX and CAN_RX signals to the CAN controller in silent mode.

Figure 3-5 CAN controller in silent mode



■ Loop back mode

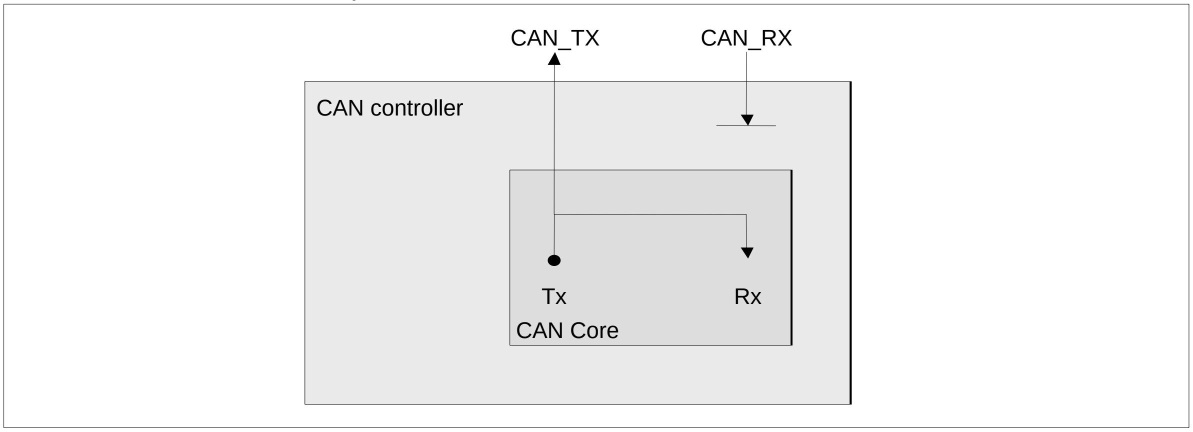
The CAN controller can be set in loop back mode by programming the LBack bit in the CAN Test Register to "1".

Loop back mode can be used for self-diagnostic functions.

In loop back mode, TX is connected with RX inside the CAN controller. The CAN controller treats the transmitted messages as messages received by RX, and stores the messages passed acceptance filtering into the receive buffer.

Figure 3-6 shows the connection of the CAN_TX and CAN_RX signals to the CAN controller in loop back mode.

Figure 3-6 CAN controller in loop back mode



<Note>

Being independent of external signals, the CAN controller does not sample dominant bits in the acknowledgement slot of a data/remote frame. This usually causes the CAN controller to generate acknowledgement errors. In this test mode, however, the errors are not caused.

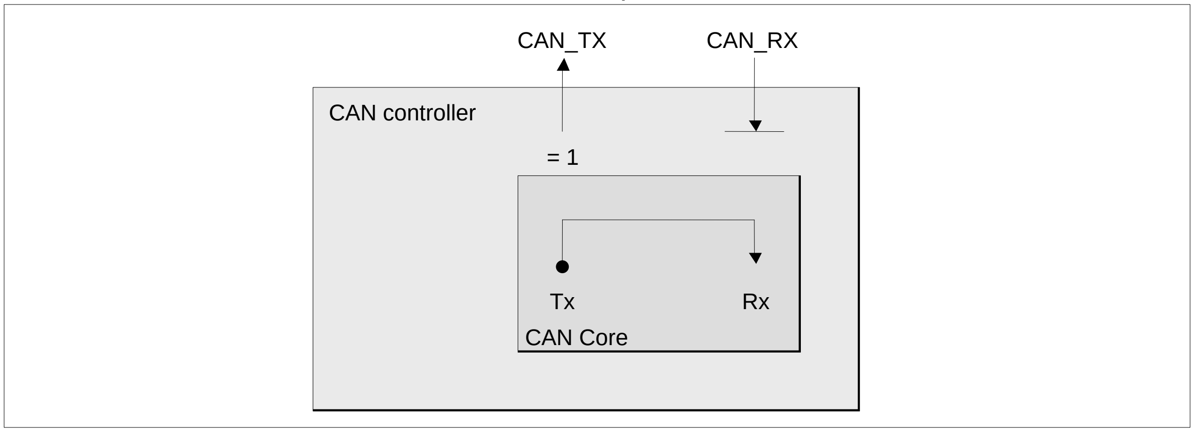
■ Combination of silent mode and loop back mode

Loop back mode and silent mode can be combined by setting the LBack bit and Silent bit in the CAN Test Register to "1" at the same time.

This mode can be used for "Hot self-test". The "Hot self-test" means that the CAN controller can be tested in loop back mode without affecting operation of the CAN system, because a constant recessive value is output from the CAN_TX pin and the input to the CAN_RX pin is ignored.

Figure 3-7 shows the connection of the CAN_TX and CAN_RX signals to the CAN controller when silent mode and loop back mode are combined.

Figure 3-7 CAN controller in combined silent and loop back modes



■ Basic mode

The CAN controller can be set in basic mode by programming the Basic bit in the CAN Test Register to "1".

In basic mode the CAN controller runs without using the message RAM.

The IF1 Message Interface Register is used to control transmission.

First when sending a message, the contents of transmission are configured in the IF1 Message Register. Then the BUSY bit in the IF1 Command Request Register is set to "1" to request transmission. While the BUSY bit is set to "1", the IF1 Message Interface Register is locked or the transmission is pending.

When the BUSY bit is set to "1", the CAN controller performs the following operation:

Immediately when the CAN bus becomes idle, the CAN controller loads the contents of the IF1 Message Interface Register to the send shift register to start transmission. When the transmission has finished successfully, the BUSY bit is reset to "0", and the locked IF1 Message Interface Register is released.

While pending, the transmission can be aborted by resetting the BUSY bit in the IF1 Command Request Register to "0". If the BUSY bit is reset to "0" during the transmission, a possible retransmission in case of lost arbitration or error detection is disabled.

The IF2 Message Interface Register is used to control reception.

All contents of the message are received without using acceptance filtering. The contents of the received message can be read by setting the BUSY bit in the IF2 Command Request Register to "1".

When the BUSY bit is set to "1", the CAN controller performs the following operation:

· Stores the received message (the contents of the receive shift register) into the IF2 Message Interface Register without any acceptance filtering.

If a new message is stored into the IF2 Message Interface Register, the CAN controller sets the NewDat bit to "1". When an additional message is received while the NewDat bit is "1", then CAN controller sets MsgLst to "1".

<Notes>

· In basic mode, all the message objects related to control and status bits are ignored as well as the control mode setting of the IFx Command Mask Register.
· The message number of the command request register is ignored.
· The NewDat bit and MsgLst bit in the IF2 Message Control Register retain their usual function, DLC3 to DLC0 indicates the received DLC, and other control bits are read as "0".

■ Software control of the CAN_TX pin

CAN_TX is a CAN send pin and has four output functions:

· Outputs serial data (Usual output)
· Outputs CAN sampling point signals to monitor the bit timing of the CAN controller
· Outputs a constant dominant value
· Outputs a constant recessive value

The output of constant dominant and recessive values, combined with CAN_RX monitoring function of the CAN receive pin, can be used to check the CAN bus physical layer.

The output mode of the CAN_TX pin can be controlled by the Tx1 and Tx0 bits in the CAN Test Register.

<Note>

When using CAN message transmission or any of the loop back, silent, or basic modes, the CAN_TX must be set to the serial data output.

3.8. Software initialization

The following explains about initialization using software.

The sources of software initialization are as follows:

· Hardware reset
· Setting the Init bit in the CAN Control Register
· Shift to a busoff state

A hardware reset initializes all other than the message RAM (excluding the MsgVal, NewDat, IntPnd, and TxRqst bits). The message RAM must be initialized, after the hardware reset, by the CPU or by setting the MsgVal in the message RAM to "0". The Bit Timing Register must be configured before clearing the Init bit in the CAN Control Register to "0".

The Init bit in the CAN Control Register is set to "1" in the following conditions:

· Writing "1" from the CPU
· Hardware reset
· In a busoff state

When the Init bit is set to "1", all message transfer from/to the CAN bus is stopped, and the CAN_TX pin in the CAN bus output is in a recessive state (excluding CAN_TX test mode).

Setting the Init bit to "1" does not change the error counter and any register.

When the Init bit and CCE bit in the CAN Control Register are set to "1", the Bit Timing Register for baud rate control and Prescaler Extension Register can be configured.

The software initialization is completed by resetting the Init bit to "0".

By waiting for the occurrence of a consecutive 11 recessive bits (i.e., bus idle) after the Init bit is reset to "0", the message is transferred after synchronization with data transfer on the CAN bus.

Before changing message object masks ID, Xtd, EoB, and RmtEn during normal operation, the MsgVal must be disabled.

4. CAN Registers

The following registers are provided for CAN.
- CAN Control Register (CTRLR)
- CAN Status Register (STATR)
- CAN Error Counter (ERRCNT)
- CAN Bit Timing Register (BTR)
- CAN Interrupt Register (INTR)
- CAN Test Register (TESTR)
- CAN Prescaler Extension Register (BRPER)
- IFx Command Request Register (IFxCREQ)
- IFx Command Mask Register (IFxCMSK)
- IFx Mask Registers 1, 2 (IFxMSK1, IFxMSK2)
- IFx Arbitration 1, 2 (IFxARB1, IFxARB2)
- IFx Message Control Register (IFxMCTR)
- IFx Data Register A1, A2, B1, B2 (IFxDTA1, IFxDTA2, IFxDTB1, IFxDTB2)
- CAN Transmit Request Registers 1, 2 (TREQR1, TREQR2)
- CAN New Data Registers 1, 2 (NEWDT1, NEWDT2)
- CAN Interrupt Pending Registers 1, 2 (INTPND1, INTPND2)
- CAN Message Valid Registers 1, 2 (MSGVAL1, MSGVAL2)

■ Total control register list

Table 4-1 Total control register list

Abbreviation	Register name	Reference
CTRLR	CAN Control Register	4.2.1
STATR	CAN Status Register	4.2.2
ERRCNT	CAN Error Counter	4.2.3
BTR	CAN Bit Timing Register	4.2.4
INTR	CAN Interrupt Register	4.2.5
TESTR	CAN Test Register	4.2.6
BRPER	CAN Prescaler Extension Register	4.2.7

■ Message interface register list

Table 4-2 Message interface register list

Abbreviation	Register name	Reference
IF1CREQ	IF1 Command Request Register	4.3.1
IF1CMSK	IF1 Command Mask Register	4.3.2
IF1MSK1	IF1 Mask Register 1	4.3.3
IF1MSK2	IF1 Mask Register 2	4.3.3
IF1ARB1	IF1 Arbitration Register 1	4.3.4
IF1ARB2	IF1 Arbitration Register 2	4.3.4
IF1MCTR	IF1 Message Control Register	4.3.5
IF1DTA1	IF1 Data A Register 1 (Little endian)	4.3.6
IF1DTA2	IF1 Data A Register 2 (Little endian)	4.3.6
IF1DTB1	IF1 Data B Register 1 (Little endian)	4.3.6
IF1DTB2	IF1 Data B Register 2 (Little endian)	4.3.6
IF1DTA2	IF1 Data A Register 2 (Big endian)	4.3.6
IF1DTA1	IF1 Data A Register 1 (Big endian)	4.3.6
IF1DTB2	IF1 Data B Register 2 (Big endian)	4.3.6
IF1DTB1	IF1 Data B Register 1 (Big endian)	4.3.6
IF2CREQ	IF2 Command Request Register	4.3.1
IF2CMSK	IF2 Command Mask Register	4.3.2
IF2MSK1	IF2 Mask Register 1	4.3.3
IF2MSK2	IF2 Mask Register 2	4.3.3
IF2ARB1	IF2 Arbitration Register 1	4.3.4
IF2ARB2	IF2 Arbitration Register 2	4.3.4
IF2MCTR	IF2 Message Control Register	4.3.5
IF2DTA1	IF2 Data A Register 1 (Little endian)	4.3.6
IF2DTA2	IF2 Data A Register 2 (Little endian)	4.3.6
IF2DTB1	IF2 Data B Register 1 (Little endian)	4.3.6
IF2DTB2	IF2 Data B Register 2 (Little endian)	4.3.6
IF2DTA2	IF2 Data A Register 2 (Big endian)	4.3.6
IF2DTA1	IF2 Data A Register 1 (Big endian)	4.3.6
IF2DTB2	IF2 Data B Register 2 (Big endian)	4.3.6
IF2DTB1	IF2 Data B Register 1 (Big endian)	4.3.6

■ Message handler register list

Table 4-3 Message handler register list

Abbreviation	Register name	Reference
TREQ1	CAN Transmit Request Register 1	4.5.1
TREQ2	CAN Transmit Request Register 2	4.5.1
NEWDT1	CAN New Data Register 1	4.5.2
NEWDT2	CAN New Data Register 2	4.5.2
INTPND1	CAN Interrupt Pending Register 1	4.5.3
INTPND2	CAN Interrupt Pending Register 2	4.5.3
MSGVAL1	CAN Message Valid Register 1	4.5.4
MSGVAL2	CAN Message Valid Register 2	4.5.4

4.1. CAN register functions

An address space of 256 bytes is allocated to the CAN registers. The CPU gains access to the message RAM via the message interface registers.

This section lists CAN registers, and describes the detailed function of each register.

■ Total control registers

- CAN Control Register (CTRLR)
- CAN Status Register (STATR)
- CAN Error Counter (ERRCNT)
- CAN Bit Timing Register (BTR)
- CAN Interrupt Register (INTR)
- CAN Test Register (TESTR)
- CAN Prescaler Extension Register (BRPER)

■ Message interface registers

- IFx Command Request Register (IFxCREQ)
- IFx Command Mask Register (IFxCMSK)
- IFx Mask Registers 1, 2 (IFxMSK1, IFxMSK2)
- IFx Arbitration Registers 1, 2 (IFxARB1, IFxARB2)
- IFx Message Control Register (IFxMCTR)
- IFx Data Registers A1, A2, B1, B2 (IFxDTA1, IFxDTA2, IFxDTB1, IFxDTB2)

■ Message handler registers

- CAN Transmit Request Registers 1, 2 (TREQR1, TREQR2)
- CAN New Data Registers 1, 2 (NEWDT1, NEWDT2)
- CAN Interrupt Pending Registers 1, 2 (INTPND1, INTPND2)
- CAN Message Valid Registers 1, 2 (MSGVAL1, MSGVAL2)

4.2. Total control registers

Total control registers control the CAN protocol and operating modes, and provide status information.

■ Total control registers

- CAN Control Register (CTRLR)
- CAN Status Register (STATR)
- CAN Error Counter (ERRCNT)
- CAN Bit Timing Register (BTR)
- CAN Interrupt Register (INTR)
- CAN Test Register (TESTR)
- CAN Prescaler Extension Register (BRPER)

4.2.1. CAN Control Register (CTRLR)

The CAN Control Register controls the operating modes of the CAN controller.

■ Register configuration

- CAN Control Register (high-order byte)

bit	15	14	13	12	11	10	9	8
Field	Reserved							
Attribute	-							
Initial value	0x00							

- CAN Control Register (low-order byte)

bit	7	6	5	4	3	2	1	0
Field	Test	CCE	DAR	Reserved	EIE	SIE	IE	Init
Attribute	R/W	R/W	R/W	-	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	1

■ Register functions

[bit15:8] Reserved: Reserved bits

These bits are read as "0", and must be set to "0" when writing.

[bit7] Test: Test mode enable bit

Value	Function
0	Normal operation [Initial value]
1	Test mode

<Note>

The Test bit can be set to "1" only while the Init bit is "1".

[bit6] CCE: Bit Timing Register write enable bit

Value	Function
0	Disables write access to the CAN Bit Timing Register and CAN Prescaler Extension Register. [Initial value]
1	Enables write access to the CAN Bit Timing Register and CAN Prescaler Extension Register. This setting is valid while the Init bit is "1".

[bit5] DAR: Automatic retransmission disable bit

Value	Function
0	Enables automatic retransmission when arbitration is lost or an error is detected. [Initial value]
1	Disables automatic retransmission.

Based on the CAN specification (ISO11898. See 6.3.3 Recovery Sequence), the CAN controller automatically resends frames when arbitration is lost or an error is detected during transfer. To allow the automatic retransmission, set the DAR bit to "0". To operate CAN in Time Triggered CAN (TTCAN, See ISO11898-1) environments, set the DAR bit to "1".

<Notes>

- In the mode where the DAR bit is set to "1", the TxRqst bit and the NewDat bit of a message object behave differently. (For message objects, see "4.4 Message objects")
  - When frame transmission has started, the TxRqst bit of the message object is reset to "0" while NewDat remains set.
  - When frame transmission has finished successfully, the NewDat bit is reset to "0".

  If arbitration is lost or an error is detected during transmission, the NewDat bit remains set.
  To restart the transmission, the CPU must set the TxRqst to "1".

- If the DAR bit in the CAN Control Register (CTRLR) is changed from "0" to "1" during frame transmission (TxRqst = "1"), a frame being transmitted will be transmitted again. Therefore, change the DAR bit only while the Init bit is "1".
- A transmission using two or more message buffers while the DAR bit is set to "1" assumes the following operations:
  - If the TxRqst in other message buffer is set to "1" before or during frame transmission (TxRqst bits in multiple message buffers are set to "1"), all the set TxRqst bits are reset to "0" upon the start of frame transmission, and data in the message buffer with the highest priority will be sent.

    When frame transmission has finished successfully, the NewDat bit of the sent message buffer is reset to "0" and, if TxIE of the message buffer is "1" then, IntPnd of the message object is set to "1".

    Data in other message buffers will not be sent because their TxRqst bits have been reset to "0" upon the start of frame transmission.
    Check the message buffer sent by NewDat and IntPnd, and then set TxRqst and NewDat to "1" again for another message buffer to be sent.

[bit4] Reserved: Reserved bit

This bit is read as "0", and must be set to "0" when writing.

[bit3] EIE: Error interrupt code enable bit

Value	Function
0	A change of the BOff or EWarn bit in the CAN Status Register disables the setting of interrupt code in the CAN Interrupt Register. [Initial value]
1	A change of the BOff or EWarn bit in the CAN Status Register enables the setting of status interrupt code in the CAN Interrupt Register.

[bit2] SIE: Status interrupt code enable bit

Value	Function
0	A change of the TxOk, RxOk, or LEC bit in the CAN Status Register disables the setting of interrupt code in the CAN Interrupt Register. [Initial value]
1	A change of the TxOk, RxOk, or LEC bit in the CAN Status Register enables the setting of status interrupt code in the CAN Interrupt Register. A change of TxOk, RxOk, or LEC bit caused by write access from the CPU is not set in the CAN Interrupt Register.

[bit1] IE: Interrupt enable bit

Value	Function
0	Disables interrupt generation. [Initial value]
1	Enables interrupt generation.

[bit0] Init: Initialization bit

Value	Function
0	CAN controller operations enabled.
1	Initialization [Initial value]

<Notes>

- The busoff recovery sequence (see CAN Specification Rev. 2.0) cannot be shortened by setting or resetting the Init bit. If the device enters busoff state, the CAN controller itself sets the Init bit to "1", stopping all bus operations. If the Init bit is cleared to "0" from the busoff state, the bus operation remains stopped until 129 bus idle sequences (one bus idle sequence consists of 11 recessive bits) occur consecutively. When the bus recovery sequence has completed, the error counter is reset.
- If the Init bit is set to "1" and then reset to "0" during the busoff recovery sequence, the busoff recovery sequence restarts from the beginning (sends a set of 11 recessive bits 129 times).
- To write to the CAN Bit Timing Register, set the Init and CCE bits to "1".
- Setting the Init bit to "1" during transfer stops data reception immediately.
- To set the Init bit to "1" during transmission, set the Init bit to "1" after the transmission has finished. If you set the Init bit to "1" during transmission, set the Init bit to "0" and then wait for a two-bit time to perform the transmission setting (TxRqst="1").
- Before making transition to low consumption mode (stop mode or clock mode), and before changing clock supply, the Init bit must be set to "1" to initialize the CAN controller.
- To change the division ratio of clock supplied to the CAN interface by using the following registers, set the Init bit to "1" to stop the CAN controller previously.
  - CAN Bit Timing Register (BTR)
  - CAN Prescaler Extension Register (BRPER)
  - CAN Prescaler (CANPRE)

4.2.2. CAN Status Register (STATR)

The CAN Status Register indicates the CAN status and a CAN bus state.

■ Register configuration

- CAN Status Register (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	Reserved							
Attribute	-							
Initial value	0x00							

- CAN Status Register (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	BOff	EWarn	EPass	RxOk	TxOk	LEC		
Attribute	R,WX	R,WX	R,WX	R,W	R,W	R,W		
Initial value	0	0	0	0	0	000		

■ Register functions

[bit15:8] Reserved: Reserved bits

These bits are read as "0", and must be set to "0" when writing.

[bit7] BOff: Busoff bit

Value	Function
0	CAN bus is not in busoff state. [Initial value]
1	CAN bus is in busoff state.

[bit6] EWarn: Warning bit

Value	Function
0	Both the send and receive counters are below 96. [Initial value]
1	Send or receive counter has reached or exceeded 96.

[bit5] EPass: Error passive bit

Value	Function
0	Both the send and receive counters are below 128 (error active state). [Initial value]
1	The RP bit of the receive counter is "1", or the send counter is between 128 and 255 (error passive state).

[bit4] RxOk: Successful message reception bit

Value	Function
0	No message has been transferred successfully on the CAN bus, or the bus is in idle state. [Initial value]
1	A messages has been transferred successfully on the CAN bus.

[bit3] TxOk: Successful message transmission bit

Value	Function
0	The bus is in idle state, or no message has been sent successfully. [Initial value]
1	A messages has been sent successfully.

<Note>

The RxOk and TxOk bits can be reset only by the CPU.

[bit2:0] LEC: Last error code bits

bit2:0	State	Function
0	Normal	Successful transmission or reception. [Initial value]
1	Stuff error	Six or more dominant or recessive bits have been detected consecutively in a message.
2	Form error	A wrong fixed format part of a received frame has been detected.
3	Ack error	A sent message was not acknowledged by another node.
4	Bit 1 error	In the sent message data excluding the arbitration field, bits that have been sent as recessive data is detected as dominant data.
5	Bit 0 error	In the sent message data excluding the arbitration field, bits that have been sent as dominant data is detected as recessive data. This bit is set each time 11 recessive bits are detected during bus recovery. The bus recovery sequence can be monitored by reading this bit.
6	CRC error	The CRC data in a received message did not match the calculated CRC value.
7	Undetected	If the CPU wrote "7" to the LEC bit, and the LEC value is read as "7" afterward, it indicates that no bus event has been detected since the CPU wrote the value. (The bus is in idle state)

The LEC bit holds a code that indicates the last error occurred on the CAN bus. When a message has been transferred (sent or received) without error, this bit is cleared to "0". The undetected code "7" is written by the CPU to check for code updates.

<Notes>

- If the BOff and EWarn bits change while the EIE bit is "1", or if the RxOk, TxOk, and LEC bits change while the SIE bit is "1", the status interrupt code (0x8000) is written to the CAN Interrupt Register.
- Writing from the CPU updates the RxOk and TxOk bits, and this erases the RxOk and TxOk bits set by the CAN controller. If the RxOk and TxOk bits are used, clear the RxOk and TxOk bits within the time (45 × BT) after they are set to "1". BT indicates one bit time.
- If a change of the LEC bit causes an interrupt while the SIE bit is "1", do not write to the CAN Status Register.
- No interrupt is caused by a change of the EPass bit, or writing to the RxOk, TxOk, and LEC bits from the CPU.
- When the BOff bit has turned to "1", the EPass bit and EWarn bit are "1". When the EPass bit has turned to "1", the EWarn bit is "1".
- The status interrupt (0x8000) of the CAN Interrupt Register is cleared by reading this register.

4.2.3. CAN Error Counter (ERRCNT)

The CAN Error Counter indicates the receive error passive, the receive error counter, and the send error counter.

■ Register configuration

- CAN Error Counter (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	RP	REC[6:0]						
Attribute	R,WX	R,WX						
Initial value	0	0000000						

- CAN Error Counter (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	TEC[7:0]							
Attribute	R,WX							
Initial value	0x00							

■ Register functions

[bit15] RP: Receive error passive indication

Value	Function
0	The receive error counter is below the error passive level. [Initial value]
1	The receive error counter has reached the error passive level defined in the CAN specification.

[bit14:8] REC[6:0]: Receive error counter

A receive error counter value. The range of the receive error counter value is between 0 and 127.
If the receive error counter reaches or exceeds 128, the RP bit is set to "1", and the counter is not refreshed.

Example: If a receive error adds 8 to REC[6:0] = 127 with RP = 0,
then REC[6:0] = 127 with RP = 1.
If a receive error adds 8 to REC[6:0] = 126 with RP = 0,
then REC[6:0] = 126 with RP = 1.
If a receive error adds 8 to REC[6:0] = 119 with RP = 0,
then REC[6:0] = 127 with RP = 0.
If reception is successful when REC[6:0] = 126 and RP = 1,
then REC[6:0] = 125 and RP = 0.

[bit7:0] TEC[7:0]: Send error counter

A send error counter value. The range of the send error counter value is between 0 and 255.
If the send error counter reaches or exceeds 256, the Init bit of the CAN Control Register is set to "1", and the counter is not refreshed.

Example: If a send error adds 8 to TEC[7:0] = 255 with Init = 0,
then TEC[7:0] = 255 with Init = 1.
If a send error adds 8 to TEC[7:0] = 254 with Init = 0,
then TEC[7:0] = 254 with Init = 1.
If a receive error adds 8 to TEC[7:0] = 247 with Init = 0,
then TEC[7:0] = 255 with Init = 0.

4.2.4. CAN Bit Timing Register (BTR)

The CAN Bit Timing Register configures the prescaler and the bit timing.

■ Register configuration

- CAN Bit Timing Register (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	Reserved	TSeg2			TSeg1			
Attribute	-	R/W			R/W			
Initial value	0	010			0011			

- CAN Bit Timing Register (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	SJW		BRP					
Attribute	R/W		R/W					
Initial value	00		000001					

■ Register functions

[bit15] Reserved: Reserved bit

This bit is read as "0", and must be set to "0" when writing.

[bit14:12] TSeg2: Time segment 2 setting bits

Valid programmed values are 0 to 7. The TSeg2 + 1 value is the time segment 2.
The time segment 2 is equivalent to the Phase Buffer Segment (PHASE_SEG2) in the CAN specification.

[bit11:8] TSeg1: Time segment 1 setting bits

Valid programmed values are 1 to 15. The 0 value must not be used. The TSeg1 + 1 value is the time segment 1.
The time segment 1 is equivalent to the Propagation Segment (PROP_SEG) + Phase Buffer Segment 1 (PHASE_SEG1) in the CAN specification.

[bit7:6] SJW: Resynchronization jump width setting bits

Valid programmed values are 0 to 3. The SJW + 1 value is the resynchronization jump width.

[bit5:0] BRP: Baud rate prescaler setting bits

Valid programmed values are 0 to 63. The BRP + 1 value is the baud rate prescaler.
It determines the basic unit of time quantum (tq) for the CAN controller by dividing the system clock (fsys).

<Note>

The CAN Bit Timing Register and CAN Prescaler Extension Register must be configured while the Init bit and CCE bit in the CAN Control Register are set to "1".

4.2.5. CAN Interrupt Register (INTR)

The CAN Interrupt Register indicates message interrupt code and status interrupt code.

■ Register configuration

- CAN Interrupt Register (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	IntId15 to IntId8							
Attribute	R,WX							
Initial value	0x00							

- CAN Interrupt Register (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	IntId7 to IntId0							
Attribute	R,WX							
Initial value	0x00							

■ Register functions

Value	Function
0x0000	No interrupt
0x0001 to 0x0020	An interrupt factor indicates a message object number. (Message interrupt code)
0x0021 to 0x7FFF	Unused.
0x8000	Indicates an interrupt by a change in the CAN Status Register. (Status interrupt code)
0x8001 to 0xFFFF	Unused.

If two or more interrupts are pending, the CAN Interrupt Register indicates a high-priority interrupt code. If a high-priority interrupt code is generated while an interrupt code is set to the CAN Interrupt Register, the CAN Interrupt Register is updated to the high-priority interrupt code.

High-priority interrupt codes are arranged in the order of status interrupt code (0x8000), message interrupt codes (0x0001, 0x0002, 0x0003, ..., 0x0020).

When the IE bit of the CAN Control Register is set to "1" while the IntId bit is not 0x0000, a CPU interrupt signal becomes active. When the IntId bit is set to 0x0000 (an interrupt factor is reset) or the IE bit of the CAN Control Register is reset to "0", an interrupt signal becomes inactive.

To clear a message interrupt code, reset the IntPnd bit of the target message object (see "4.4 Message objects" for the message object) to "0".

A status interrupt code is cleared by reading the CAN Status Register.

<Note>

To read the CAN Interrupt Register, access it in halfword or word mode.

4.2.6. CAN Test Register (TESTR)

The CAN Test Register is used to set the test mode and monitor the RX pin. For operations, see "3.7 Test mode".

■ Register configuration

- CAN Test Register (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	Reserved							
Attribute	-							
Initial value	0x00							

- CAN Test Register (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	Rx	Tx1	Tx0	LBack	Silent	Basic	Reserved	Reserved
Attribute	R,WX	R/W	R/W	R/W	R/W	R/W	-	-
Initial value	r	0	0	0	0	0	0	0

The initial value "r" of Rx in bit 7 indicates the level on the CAN bus.

■ Register functions

[bit15:8] Reserved: Reserved bits

These bits are read as "0", and must be set to "0" when writing.

[bit7] Rx: Rx pin monitor bit

Value	Function
0	Indicates that the CAN bus is in the dominant state.
1	Indicates that the CAN bus is in the recessive state.

[bit6:5] Tx1, Tx0: TX pin control bits

bit6	bit5	Function
0	0	Normal operation. [Initial value]
0	1	Outputs a sampling point to the Tx pin.
1	0	Outputs a dominant to the TX pin.
1	1	Outputs a recessive to the TX pin.

[bit4] LBack: Loop back mode

Value	Function
0	Disables loop back mode. [Initial value]
1	Enables loop back mode.

[bit3] Silent: Silent mode

Value	Function
0	Disables silent mode. [Initial value]
1	Enables silent mode.

[bit2] Basic: Basic mode

Value	Function
0	Disables basic mode. [Initial value]
1	Enables basic mode. The IF1 register is used for a sent message, and the IF2 register for a received message.

[bit1:0] Reserved: Reserved bits

These bits are read as "0", and must be set to "0" when writing.

<Notes>

- After setting "1" to the Test bit of the CAN Control Register, write data to this register. When the Test bit of the CAN Control Register is set to "1", test mode becomes valid. If the Test bit of the CAN Control Register is set to "0" during processing, test mode changes to normal mode.
- If the Tx bits are set to a value other than "00", no message can be sent.

4.2.7. CAN Prescaler Extension Register (BRPER)

The CAN Prescaler Extension Register is used to extend the prescaler used in the CAN controller by combining it with the prescaler specified at a CAN bit timing.

■ Register configuration

- CAN Prescaler Extension Register (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	Reserved							
Attribute	-							
Initial value	0x00							

- CAN Prescaler Extension Register (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	Reserved				BRPE			
Attribute	-				R/W			
Initial value	0000				0000			

■ Register functions

[bit15:4] Reserved: Reserved bits

These bits are read as "0", and must be set to "0" when writing.

[bit3:0] BRPE: Baud rate prescaler extension bits

These bits are used to extend the baud rate prescaler up to 1023 by combining BRP and BRPE in the CAN Bit Timing Register.

The value "{BRPE (MSB: 4 bits), BRP (LSB: 6 bits)} + 1" is set as the prescaler value of the CAN controller.

4.3. Message interface registers

The CAN controller provides two message interface registers to control an access from the CPU to the message RAM.

The CAN controller provides two message interface registers to control an access from the CPU to the message RAM. These two registers are used to avoid a confliction between an access from the CPU to the message RAM and an access from the CAN controller to the message RAM by buffering the data (message object) transferred or to be transferred. A message object (see "4.4 Message objects" for message object) is used to collectively transfer data between the message interface registers and message RAM.

Two message interface registers have the same functions, excluding basic test mode, and can be operated independently. For example, the IF2 Message Interface Register can be used to read data from the message RAM while the IF1 Message Interface Register is being used to write data to the message RAM. Table 4-2 shows two message interface registers.

Each Message Interface Register consists of two components: (1) Command Register (Command Request and Command Mask Registers) and (2) Message Buffer Register (Mask, Arbitration, Message Control, and Data Registers) controlled with the Command Register. The Command Mask Register indicates the data transfer direction and also which part in a message object is to be transferred. The Command Request Register is used to select a message number and perform the operation specified in the Command Mask Register.

4.3.1. IFx Command Request Register (IFxCREQ)

The IFx Command Request Register is used to select a message number of the message RAM and transfer data between the message RAM and Message Buffer Register. In basic test mode, IF1 is used to control sending and IF2 to control receiving.

■ Register configuration

- IFx Command Request Register (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	BUSY	Reserved						
Attribute	R/W	-						
Initial value	0	0000000						

- IFx Command Request Register (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	Message Number							
Attribute	R/W							
Initial value	0x01							

■ Register functions

A message transfer starts between the message RAM and Message Buffer Register (Mask, Arbitration, Message Control, and Data Registers) immediately after a message number has been written to the IFx Command Request Register. This write operation sets the BUSY bit to "1" and continues transfer processing while the BUSY bit is "1". When transfer processing is ended, the BUSY bit is reset to "0".

If the CPU accesses the Message Interface Register while the BUSY bit is "1", the CPU waits until the BUSY bit is set to "0" (for 3 to 6 clock cycles after data has been written to the Command Request Register).

The method for using the BUSY bit is different in basic test mode. The IF1 Command Request Register, which is used as a send message, starts message sending when the BUSY bit is set to "1". When message transfer has finished successfully, the BUSY bit is reset to "0". Resetting the BUSY bit to "0" enables canceling message transfer at any time.

The IF2 Command Request Register, which is used for receiving message, stores the received message in the IF2 Message Interface Register when the BUSY bit is set to "1".

[bit15] BUSY: Busy flag bit

· Other than basic test mode

Value	Function
0	Indicates that data transfer is not performed between the Message Interface Register and message RAM. [Initial value]
1	Indicates that data transfer is being performed between the Message Interface Register and message RAM.

· Basic test mode

· IF1 Command Request Register

Value	Function
0	Disables message sending.
1	Enables message sending.

· IF2 Command Request Register

Value	Function
0	Disables message receiving.
1	Enables message receiving.

[bit14:8] Reserved: Reserved bits

These bits are read as "0", and must be set to "0" when writing.

[bit7:0] Message Number: Message number (32 message buffers)

Value	Function
0x00, 0x40, 0x60, 0x80, 0xA0, 0xC0, 0xE0	Setting is prohibited. If specified, it is interpreted as 0x20, causing 0x20 to be read.
0x01 to 0x20	Specifies a message number to perform processing.
0x21 to 0x3F, 0x41 to 0x5F, 0x61 to 0x7F, 0x81 to 0x9F, 0xA1 to 0xBF, 0xC1 to 0xDF, 0xE1 to 0xFF	Setting is prohibited. If specified, it is interpreted as one of 0x01 to 0x1F, causing the interpreted value to be read.

<Note>

The BUSY bit can be read and written. Therefore, writing any data to this bit does not affect operations, excluding in basic test mode (see "3.7 Test mode" for basic test mode).

4.3.2. IFx Command Mask Register (IFxCMSK)

The IFx Command Mask Register is used to control the transfer direction between the Message Interface Register and message RAM and specify which data is to be updated. This register is invalid in basic test mode.

■ Register configuration

- IFx Command Mask Register (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	Reserved							
Attribute	-							
Initial value	0x00							

- IFx Command Mask Register (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	WR/RD	Mask	Arb	Control	CIP	TxRqst/ NewDat	Data A	Data B
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

■ Register functions

[bit15:8] Reserved: Reserved bits

These bits are read as "0", and must be set to "0" when writing.

[bit7] WR/RD: Writing or reading control bit

Value	Function
0	Indicates that data is read from the message RAM. Reading from the message RAM is performed by writing data to the IFx Command Request Register. What data is to be read from the message RAM depends on the setting of the Mask, Arb, Control, CIP, TxRqst/NewDat, Data A, or Data B bit. [Initial value]
1	Indicates that data is written to the message RAM. Writing to the message RAM is performed by writing data to the IFx Command Request Register. What data is to be written to the message RAM depends on the setting of the Mask, Arb, Control, CIP, TxRqst/NewDat, Data A, or Data B bit.

<Note>

After resetting, data of the message RAM is unfixed. The message RAM cannot be read while its data is unfixed.

The meaning of the bit6 to bit0 in the IFx Command Mask Register depends on the transfer direction specified with the WR or RD bit.

● When the transfer direction is "writing" (WR/RD="1")

[bit6] Mask: Mask data update bit

Value	Function
0	Indicates that mask data (ID mask + MDir + MXtd) of a message object*1 is not updated. [Initial value]
1	Indicates that mask data (ID mask + MDir + MXtd) of a message object*1 is updated.

*1: See "4.4 Message objects".

[bit5] Arb: Arbitration data update bit

Value	Function
0	Indicates that arbitration data (ID + Dir + Xtd + MsgVal) of a message object*1 is not updated. [Initial value]
1	Indicates that arbitration data (ID + Dir + Xtd + MsgVal) of a message object*1 is updated.

*1: See "4.4 Message objects".

[bit4] Control: Control data update bit

Value	Function
0	Indicates that control data (IFx Message Control Register) of a message object*1 is not updated. [Initial value]
1	Indicates that control data (IFx Message Control Register) of a message object*1 is updated.

*1: See "4.4 Message objects".

[bit3] CIP: Interrupt clear bit

If this bit is set to "0" or "1", it does not affect CAN controller operations.

[bit2] TxRqst/NewDat: Message transmission request bit

Value	Function
0	Indicates that the TxRqst bits of the message object*1 and CAN Transmit Request Register are not changed. [Initial value]
1	Indicates that the TxRqst bits of the message object*1 and CAN Transmit Request Register are set to "1" (transmission requested).

*1: See "4.4 Message objects".

[bit1] Data A: Data0 to Data3 update bit

Value	Function
0	Indicates that Data0 to Data3 of a message object*1 is not updated. [Initial value]
1	Indicates that Data0 to Data3 of a message object*1 is updated.

*1: See "4.4 Message objects".

[bit0] Data B: Data4 to Data7 update bit

Value	Function
0	Indicates that Data4 to Data7 of a message object*1 is not updated. [Initial value]
1	Indicates that Data4 to Data7 of a message object*1 is updated.

*1: See "4.4 Message objects".

<Notes>

- When the TxRqst or NewDat bit of the IFx Command Mask Register is set to "1", the setting of the TxRqst bit in the IFx Message Control Register becomes invalid.
- This register is invalid in basic test mode.

● When the transfer direction is "reading" (WR/RD="0")

[bit6] Mask: Mask data update bit

Value	Function
0	Indicates that data (ID mask + MDir + MXtd) is not transferred from a message object*1 to IFx Master Register 1 or 2. [Initial value]
1	Indicates that data (ID mask + MDir + MXtd) is transferred from a message object*1 to IFx Master Register 1 or 2.

*1: See "4.4 Message objects".

[bit5] Arb: Arbitration data update bit

Value	Function
0	Indicates that data (ID + Dir + Xtd + MsgVal) is not transferred from a message object*1 to IFx Arbitration Register 1 or 2. [Initial value]
1	Indicates that data (ID + Dir + Xtd + MsgVal) is transferred from a message object*1 to IFx Arbitration Register 1 or 2.

*1: See "4.4 Message objects".

[bit4] Control: Control data update bit

Value	Function
0	Indicates that data is not transferred from a message object*1 to the IFx Message Control Register. [Initial value]
1	Indicates that data is transferred from a message object*1 to the IFx Message Control Register.

*1: See "4.4 Message objects".

[bit3] CIP: Interrupt clear bit

Value	Function
0	Indicates that the IntPnd bits of the message object*1 and CAN Interrupt Pending Register are held. [Initial value]
1	Indicates that the IntPnd bits of the message object*1 and CAN Interrupt Pending Register are cleared to "0".

*1: See "4.4 Message objects".

[bit2] TxRqst/NewDat: Data update bit

Value	Function
0	Indicates that the NewDat bits of the message object*1 and CAN New Data Register are held. [Initial value]
1	Indicates that the NewDat bits of the message object*1 and CAN New Data Register are cleared to "0".

*1: See "4.4 Message objects".

[bit1] Data A: Data0 to Data3 update bit

Value	Function
0	Indicates that data of the message object*1 and CAN Data Register A1 or A2 are held. [Initial value]
1	Indicates that data of the message object*1 and CAN Data Register A1 or A2 are updated.

*1: See "4.4 Message objects".

[bit0] Data B: Data4 to Data7 update bit

Value	Function
0	Indicates that data of the message object*1 and CAN Data Register B1 or B2 are held. [Initial value]
1	Indicates that data of the message object*1 and CAN Data Register B1 or B2 are updated.

*1: See "4.4 Message objects".

<Notes>

- The IntPnd and NewDat bits can be reset to "0" by reading a message object. However, the value before reset by reading is set to the IntPnd and NewDat bits of the IFx Message Control Register.
- This register is invalid in basic test mode.

4.3.3. IFx Mask Registers 1, 2 (IFxMSK1, IFxMSK2)

The IFx Mask Registers 1 and 2 are used to write or read message object mask data of the message RAM. The specified mask data is invalid in basic test mode.

For the function of each bit, see "4.4 Message objects".

■ Register configuration

- IFx Mask Register 2 (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	MXtd	MDir	Reserved	Msk28 to Msk24				
Attribute	R/W	R/W	R1,W1	R/W				
Initial value	1	1	1	11111				

- IFx Mask Register 2 (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	Msk23 to Msk16							
Attribute	R/W							
Initial value	0xFF							

- IFx Mask Register 1 (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	Msk15 to Msk8							
Attribute	R/W							
Initial value	0xFF							

- IFx Mask Register 1 (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	Msk7 to Msk0							
Attribute	R/W							
Initial value	0xFF							

For the explanation of each bit in this register, see "4.4 Message objects".

Read "1" in the reserved bit (bit 13 of IFx Mask Register 2). Set "1" in write mode.

4.3.4. IFx Arbitration Registers 1, 2 (IFxARB1, IFxARB2)

The IFx Arbitration Registers 1 and 2 are used to write or read message object arbitration data of the message RAM. This register is invalid in basic test mode.

For the function of each bit, see "4.4 Message objects".

■ Register configuration

- IFx Arbitration Register 2 (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	MsgVal	Xtd	Dir	ID28 to ID24				
Attribute	R/W	R/W	R/W	R/W				
Initial value	0	0	0	00000				

- IFx Arbitration Register 2 (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	ID23 to ID16							
Attribute	R/W							
Initial value	0x00							

- IFx Arbitration Register 1 (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	ID15 to ID8							
Attribute	R/W							
Initial value	0x00							

- IFx Arbitration Register 1 (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	ID7 to ID0							
Attribute	R/W							
Initial value	0x00							

For the explanation of each bit in this register, see "4.4 Message objects".

<Note>

If the MsgVal bit of a message object is cleared to "0" during transmission, the TxOk bit of the CAN Status Register is set to "1" when transmission has been completed. However, the TxRqst bits of the message object and CAN Transmit Request Register are not cleared to "0". Use the Message Interface Register to clear the TxRqst bit to "0".

4.3.5. IFx Message Control Register (IFxMCTR)

The IFx Message Control Register is used to write or read message object control data of the message RAM. IF1 Message Control Register is invalid in basic test mode. The NewDat and MsgLst bits of the IF2 Message Control Register are used to perform normal operations. The DLC bits indicate the DLC of the received message. The other control bits are invalid ("0").

For the function of each bit, see "4.4 Message objects".

■ Register configuration

- IFx Message Control Register (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

- IFx Message Control Register (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	EoB	Reserved			DLC3-0			
Attribute	R/W	-			R/W			
Initial value	0	000			0			

For the explanation of each bit in this register, see "4.4 Message objects".

<Note>

The values of the TxRqst, NewDat, and IntPnd bits are set as shown below depending on the setting of the WR or RD bit in the IFx Command Mask Register.

- When the transfer direction is "writing" (IFx Command Mask Register: WR/RD="1")
  - The TxRqst bit of this register is valid only when the TxRqst or NewDat bit of the IFx Command Mask Register is set to "0".

- When the transfer direction is "reading" (IFx Command Mask Register: WR/RD="0")
  - If the IntPnd bits of the message object and CAN Interrupt Pending Register are reset by setting the CIP bit of the IFx Command Mask Register to "1" and writing data to the IFx Command Request Register, the value of the IntPnd bit that is specified before reset is stored in this register.
  - If the NewDat bits of the message object and CAN New Data Register are reset by setting the TxRqst or NewDat bit of the IFx Command Mask Register to "1" and writing data to the IFx Command Request Register, the value of the NewDat bit that is specified before reset is stored in this register.

4.3.6. IFx Data Registers A1, A2, B1, and B2 (IFxDTA1, IFxDTA2, IFxDTB1, and IFxDTB2)

The IFx Data Registers A1, A2, B1, and B2 are used to write or read message object sending or receiving data to or from the message RAM. Those registers are used only to send or receive a data frame, and not to send or receive a remote frame.

■ Register configuration

	addr+3	addr+2	addr+1	addr+0
IFx Data A Register 1 (Little endian)			Data(1)	Data(0)
IFx Data A Register 2 (Little endian)	Data(3)	Data(2)		
IFx Data B Register 1 (Little endian)			Data(5)	Data(4)
IFx Data B Register 2 (Little endian)	Data(7)	Data(6)		
IFx Data A Register 2 (Big endian)			Data(2)	Data(3)
IFx Data A Register 1 (Big endian)	Data(0)	Data(1)		
IFx Data B Register 2 (Big endian)			Data(6)	Data(7)
IFx Data B Register 1 (Big endian)	Data(4)	Data(5)		

- IFx Data Register

bit	15 7	14 6	13 5	12 4	11 3	10 2	9 1	8 0
Field	Data							
Attribute	R/W							
Initial value	0x00							

■ Register functions

- Send message data setting
  The set data is sent in the order of Data(0), Data(1), ..., Data(7), beginning with the MSB (bit 7 or bit 15).

- Received message data
  The received message data is stored in the order of Data(0), Data(1), ..., Data(7), beginning with the MSB (bit 7 or bit 15).

<Notes>

- If the received message data is less than eight bytes in length, undefined data is written to the remaining bytes of the Data Register.
- To transfer data to a message object, it is processed every four bytes in the Data A or Data B Register; therefore, it is impossible to update only a part of 4-byte data.

4.4. Message objects

The message RAM provides 32 message objects. To avoid a confliction when simultaneously accessing the message RAM from the CPU and the CAN controller, the CPU cannot directly access message objects. The message RAM is accessed via the IFx Message Interface Register.

This section explains the configuration and functions of a message object.

■ Configuration of message object

Message object

UMask	Msk28 to Msk0	MXtd	MDir	EoB	NewDat		MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
MsgVal	ID28 to ID0	Xtd	Dir	DLC3 to DLC0	Data0	Data1	Data2	Data3	Data4	Data5	Data6	Data7

<Note>

A message object is not initialized using the Init bit of the CAN Control Register or the hardware reset function. For the hardware reset function, release the hardware reset function, and initialize the message RAM using the CPU or set MsgVal of the message RAM to "0".

■ Functions of message object

The ID28 to ID0, Xtd, and Dir bits are used to indicate the ID and message type when sending a message. They are used in the acceptance filter together with the Msk28 to Msk0, MXtd, and MDir bits when receiving a message.

ID, IDE, RTR, DLC, and DATA in a data or remote frame that passed through the acceptance filter are respectively stored in ID28 to ID0, Xtd, Dir, DLC3 to DLC0, and Data7 to Data0 of a message object. Xtd indicates whether the received frame is an extension or standard frame. If Xtd is "1", a 29-bit ID (extension frame) is received. If Xtd is "0", a 11-bit ID (standard frame) is received.

When the received data or remote frame matches one or more message objects, it is stored in the message object with the lowest message number. For details, see Acceptance Filter for Received Messages in "3.3 Message reception".

MsgVal : Valid message bit

Value	Function
0	Message objects are invalid. Disables message sending/receiving.
1	Message objects are valid. Enables message sending/receiving.

<Notes>

- Reset the MsgVal bit of an unused message object to "0" before clearing the Init bit of the CAN Control Register to "0".
- Be sure to reset the MsgVal bit of a message object to "0" before changing the value of ID28 to ID0, Xtd, Dir, or DLC3 to DLC0.
- If the MsgVal bit of a message object is cleared to "0" during transmission, the TxOk bit of the CAN Status Register is set to "1" when transmission has been completed. However, the TxRqst bits of the message object and CAN Transmit Request Register are not cleared to "0". Use the Message Interface Register to clear the TxRqst bits to "0".

UMask : Acceptance mask enable bit

Value	Function
0	Does not use Msk28 to Msk0, MXtd, or MDir.
1	Uses Msk28 to Msk0, MXtd, or MDir.

<Notes>

- Change the value of the UMask bit when the Init bit of the CAN Control Register is "1" or the MsgVal bit is "0".
- When the Dir bit is "1" and the RmtEn bit is "0", operations vary depending on the setting of the UMask bit.
  - If the UMask bit is "1", reset the TxRqst bit to "0" when a remote frame has been received through the acceptance filter. The received ID, IDE, RTR, and DLC are stored in a message object, and the NewDat bit is set to "1" while data remains unchanged (data is handled as a data frame).
  - If the UMask bit is "0", the TxRqst bit is held and a remote frame is ignored even if it has been received.

ID28 to ID0 : Message ID

	Function
ID28 to ID0	Specifies a 29-bit ID (extension frame).
ID28 to ID18	Specifies a 11-bit ID (standard frame).

Msk28 to Msk0: ID mask

Value	Function
0	Masks the bit that corresponds to the ID of a message object.
1	Does not mask the bit that corresponds to the ID of a message object.

Xtd: Extension ID enable bit

Value	Function
0	Uses the 11-bit ID (standard frame) for message object.
1	Uses the 29-bit ID (extension frame) for message object.

MXtd : Extension ID mask bit

Value	Function
0	Does not compare the set value of the Xtd bit in a message object with that of the IDE bit of a received frame. Determines whether to perform the comparison as the ID of a standard frame or extension frame based on the IDE bit of a received frame.
1	Compares the set value of the Xtd bit in a message object with that of the IDE bit of a received frame.

<Note>

When a 11-bit ID (standard frame) is set to a message object, the ID of a received data frame is written to ID28 to ID18. Msk28 to Msk18 are used to mask the ID.

Dir: Message direction bit

Value	Function
0	Indicates the receiving direction. When the TxRqst bit is set to "1", a remote frame is sent. When the TxRqst bit is set to "0", a data frame that passed through the acceptance filter is received.
1	Indicates the transmission direction. When the TxRqst bit is set to "1", a data frame is sent. When the TxRqst is "0" and the RmtEn bit is "1", the CAN controller sets the TxRqst bit to "1" if a data frame that passed through the acceptance filter is received.

MDir : Message direction mask bit

Value	Function
0	Masks the message direction bit (Dir) through the acceptance filter.
1	Does not mask the message direction bit (Dir) through the acceptance filter.

<Note>

Always set the Mdir bit to "1".

EoB: End of buffer bit (For details, see "3.4 FIFO buffer function".)

Value	Function
0	Indicates that a message object is used as a FIFO buffer, not the last message.
1	Indicates a single message object or the last message object in the FIFO buffer.

<Notes>

- The EoB bit is used to configure a FIFO buffer for message objects 2 to 32.
- When processing a single message object without using a FIFO buffer, be sure to set the EoB bit to "1".

NewDat: Data update bit

Value	Function
0	Indicates that no valid data resides.
1	Indicates that valid data resides.

MsgLst : Message lost

Value	Function
0	Message lost does not occur.
1	Message lost occurs.

<Note>

The MsgLst bit is valid only when the Dir bit is "0" (receiving direction).

RxIE: Receiving interrupt flag enable bit

Value	Function
0	Does not change the value of the IntPnd bit after frame receiving has succeeded.
1	Changes the IntPnd bit to "1" after frame receiving has succeeded.

TxIE: Transmission interrupt flag enable bit

Value	Function
0	Does not change the value of the IntPnd bit after frame transmission has succeeded.
1	Changes the IntPnd bit to "1" after frame transmission has succeeded.

IntPnd: Interrupt pending bit

Value	Function
0	No interrupt factor is detected.
1	An interrupt factor is detected. If other high-priority interrupt is not found, the IntId bit of the CAN Interrupt Register indicates this message object.

RmtEn: Remote enable

Value	Function
0	Does not change the value of the TxRqst bit when a remote frame has been received.
1	Sets the TxRqst bit to "1" when a remote frame is received while the Dir bit is "1".

<Note>

When the Dir bit is "1" and the RmtEn bit is "0", operations vary depending on the setting of the UMask bit.

- If the UMask bit is "1", reset the TxRqst bit to "0" when a remote frame has been received through the acceptance filter. The received ID, IDE, RTR, and DLC are stored in a message object. The NewDat bit is set to "1" while data remains unchanged (data is handled as a data frame).
- If the UMask bit is "0", the TxRqst bit is held and a remote frame is ignored even if it has been received.

TxRqst : Transmission request bit

Value	Function
0	Indicates the sending idle state (neither the sending state nor the sending wait state).
1	Indicates the sending or sending wait state.

DLC3 to DLC0: Data length code

Value	Function
0 to 8	The data frame length is 0 to 8 bytes.
9 to 15	Setting is prohibited. 8-byte length if specified.

<Note>

The received DLC is stored in the DLC bit if a data frame is received.

Data 0 to Data7: Data 0 to Data 7

	Function
Data 0	First data byte in CAN data frame
Data 1	2nd data byte in CAN data frame
Data 2	3rd data byte in CAN data frame
Data 3	4th data byte in CAN data frame
Data 4	5th data byte in CAN data frame
Data 5	6th data byte in CAN data frame
Data 6	7th data byte in CAN data frame
Data 7	8th data byte in CAN data frame

<Notes>

- Serial data is output from the MSB (bit 7 or bit 15) to the CAN bus.
- If the received message data is less than eight bytes in length, unfixed data is written to the remaining bytes of the Data Register.
- To transfer data to a message object, it is processed every four bytes in the Data A or Data B Register; therefore, it is impossible to update only a part of 4-byte data.

4.5. Message handler registers

Message handler registers are all in read only mode. The TxRqst, NewDat, IntPnd, and MsgVal bits of a message object and the IntId bit indicate the status.

■ Message handler registers

- CAN Transmit Request Registers 1, 2 (TREQR1, TREQR2)
- CAN New Data Registers 1, 2 (NEWDT1, NEWDT2)
- CAN Interrupt Pending Registers 1, 2 (INTPND1, INTPND2)
- CAN Message Valid Registers 1, 2 (MSGVAL1, MSGVAL2)

4.5.1. CAN Transmit Request Registers 1, 2 (TREQR1, TREQR2)

The CAN Transmit Request Registers indicate the TxRqst bits of all message objects. These registers check which message object transmission request is pending by reading the TxRqst bit.

■ Register configuration

- CAN Transmit Request Register 2 (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	TxRqst32 to TxRqst25							
Attribute	R,WX							
Initial value	0x00							

- CAN Transmit Request Register 2 (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	TxRqst24 to TxRqst17							
Attribute	R,WX							
Initial value	0x00							

- CAN Transmit Request Register 1 (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	TxRqst16 to TxRqst9							
Attribute	R,WX							
Initial value	0x00							

- CAN Transmit Request Register 1 (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	TxRqst8 to TxRqst1							
Attribute	R,WX							
Initial value	0x00							

■ Register functions

TxRqst32 to TxRqst1: Transmission request bits

Value	Function
0	Indicates the sending idle state (neither the sending state nor the sending wait state).
1	Indicates the sending or sending wait state.

The following shows conditions to set or reset the TxRqst bit.

- Setting conditions
  - Set "1" to the WR/RD bit of the IFx Command Mask Register and "1" to the TxRqst bit, and write data to the IFx Command Request Register to set the TxRqst bit to a specific message object.
  - Set "1" to the WR/RD bit of the IFx Command Mask Register, "0" to the TxRqst bit, and "1" to the Control bit, and "1" to the TxRqst bit of the IFx Message Control Register. Then write data to the IFx Command Request Register to set the TxRqst bit to a specific message object.
  - If the Dir bit is "1" and the RmtEn bit is "1", the TxRqst bit is set by receiving a remote frame that passed through the acceptance filter.

- Resetting conditions
  - Set "1" to the WR/RD bit of the IFx Command Mask Register, "0" to the TxRqst bit, and "1" to the Control, and "0" to the TxRqst bit of the IFx Message Control Register. Then write data to the IFx Command Request Register to reset the TxRqst bit of a specific message object.
  - The TxRqst bit is reset when frame transmission has finished successfully.
  - If the Dir bit is "1", the RmtEn bit is "0", and the UMask bit is "1", the TxRqst bit is reset by receiving a remote frame that passed through the acceptance filter.

<Notes>

- In one of the following conditions, the messages may not be sent until any of the events described below occurs.

  Conditions : (1) A message buffer with the lowest priority is used for transmission.
  (2) The TxRqst bit was previously set to "1", but is set to "0" to abort transmission.
  (3) The TxRqst bit is set to "1" again at the timing of (2).

  Events : - A valid message flows on the CAN bus.
  - A transmission request is issued to another message buffer.
  - CAN is initialized by the Init bit.

  If canceling the transmission is required to suit system operations, execute the following steps.

  1. Execute one of the following steps.
     - Do not use a message buffer with the lowest priority as a send message buffer.
     - After aborting the transmission, generate any of the above events.

  2. Set the TxRqst bit to "1" again.

- If the message objects of ID28 to ID0, DLC3 to DLC0, Xtd, and Data7 to Data0 are changed while the TxRqst bit is "1", message objects before and after the change may be mixed for transmission, or the message objects after the change may not be transmitted. Therefore, be sure to change them while the TxRqst bit is "0".

4.5.2. CAN New Data Registers 1, 2 (NEWDT1, NEWDT2)

The CAN New Data Registers indicate the NewDat bits of all message objects. These registers check which message object data is updated by reading the NewDat bit.

■ Register configuration

- CAN New Data Register 2 (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	NewDat32 to NewDat25							
Attribute	R,WX							
Initial value	0x00							

- CAN New Data Register 2 (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	NewDat24 to NewDat17							
Attribute	R,WX							
Initial value	0x00							

- CAN New Data Register 1 (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	NewDat16 to NewDat9							
Attribute	R,WX							
Initial value	0x00							

- CAN New Data Register 1 (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	NerwDat8 to NewDat1							
Attribute	R,WX							
Initial value	0x00							

■ Register functions

NerwDat32 to NewDat1: Data update bit

Value	Function
0	Indicates that no valid data resides.
1	Indicates that valid data resides.

The following shows conditions to set or reset the NewDat bit.

- Setting conditions
  - Set "1" to the WR/RD bit of the IFx Command Mask Register, and "1" to the Control bit, and "1" to the NewDat bit of the IFx Message Control Register. Then write data to the IFx Command Request Register to set the NewDat bit to a specific message object.
  - The NewDat bit is set by receiving a data frame that passed through the acceptance filter.
  - If the Dir bit is "1", the RmtEn bit is "0", and the UMask bit is "1", the NewDat bit is set by receiving a remote frame that passed through the acceptance filter.

- Resetting conditions
  - Set "0" to the WR/RD bit of the IFx Command Mask Register and "1" to the NewDat bit, and write data to the IFx Command Request Register to reset the NewDat bit of a specific message object.
  - Set "1" to the WR/RD bit of the IFx Command Mask Register, and "1" to the Control bit, and "0" to the NewDat bit of the IFx Message Control Register. Then write data to the IFx Command Request Register to reset the NewDat bit of a specific message object.
  - The NewDat bit is reset after data has been transferred to the transmission shift register (internal register).

4.5.3. CAN Interrupt Pending Registers 1, 2 (INTPND1, INTPND2)

The CAN Interrupt Pending Registers indicate the IntPnd bits of all message objects. These registers check which message object is pending for interrupt by reading the IntPnd bits.

■ Register configuration

- CAN Interrupt Pending Register 2 (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	IntPnd32 to IntPnd25							
Attribute	R,WX							
Initial value	0x00							

- CAN Interrupt Pending Register 2 (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	IntPnd24 to IntPnd17							
Attribute	R,WX							
Initial value	0x00							

- CAN Interrupt Pending Register 1 (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	IntPnd16 to IntPnd9							
Attribute	R,WX							
Initial value	0x00							

- CAN Interrupt Pending Register 1 (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	IntPnd8 to IntPnd1							
Attribute	R,WX							
Initial value	0x00							

■ Register functions

IntPnd32 to IntPnd1: Interrupt pending bit

Value	Function
0	No interrupt factor is detected.
1	An interrupt factor is detected.

The following shows conditions to set or reset the IntPnd bit.

- Setting conditions
  - If the TxIE bit is set to "1", the IntPnd bit is set when frame transmission has been completed normally.
  - If the RxIE bit is set to "1", the IntPnd bit is set when a frame that passed through the acceptance filter was received normally.
  - Set "1" to the WR/RD bit of the IFx Command Mask Register, and "1" to the Control bit, and "1" to the IntPnd bit of the IFx Message Control Register. Then write data to the IFx Command Request Register to set the IntPnd bit of a specific message object.

- Resetting conditions
  - Set "0" to the WR/RD bit of the IFx Command Mask Register and "1" to the CIP bit, and write data to the IFx Command Request Register to reset the IntPnd bit of a specific message object.
  - Set "1" to the WR/RD bit of the IFx Command Mask Register, and "1" to the Control bit, and "0" to the IntPnd bit of the IFx Message Control Register. Then write data to the IFx Command Request Register to reset the IntPnd bit of a specific message object.

4.5.4. CAN Message Valid Registers 1, 2 (MSGVAL1, MSGVAL2)

The CAN Message Valid Registers indicate the MsgVal bits of all message objects. These registers check which message object is valid by reading the MsgVal bits.

■ Register configuration

- CAN Message Valid Register 2 (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	MsgVal32 to MsgVal25							
Attribute	R,WX							
Initial value	0x00							

- CAN Message Valid Register 2 (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	MsgVal24 to MsgVal17							
Attribute	R,WX							
Initial value	0x00							

- CAN Message Valid Register 1 (High-order byte)

bit	15	14	13	12	11	10	9	8
Field	MsgVal16 to MsgVal9							
Attribute	R,WX							
Initial value	0x00							

- CAN Message Valid Register 1 (Low-order byte)

bit	7	6	5	4	3	2	1	0
Field	MsgVal8 to MsgVal1							
Attribute	R,WX							
Initial value	0x00							

■ Register functions

MsgVal32 to MsgVal1: Message valid bit

Value	Function
0	Message objects are invalid. Disables message sending/receiving.
1	Message objects are valid. Enables message sending/receiving.

The following shows conditions to set or reset the MsgVal bit.

· Setting conditions
  Set "1" to the WR/RD bit of the IFx Command Mask Register, and "1" to the Arb bit, and "1" to the MsgVal bit of the IFx Arbitration Register 2. Then write data to the IFx Command Request Register to set the MsgVal bit of a specific message object.

· Resetting conditions
  Set "1" to the WR/RD bit of the IFx Command Mask Register, and "1" to the Arb bit, and "0" to the MsgVal bit of the IFx Arbitration Register 2. Then write data to the IFx Command Request Register to reset the MsgVal bit of a specific message object.

5. Notes

Table 5-1 and Table 5-2 show input and output signals.

Table 5-1 Table of input and output signals (Input signal)

NO	Signal name	I/O	Polarity	EDGE*1	Functions
1	CAN_CLK	I	-	-	Operation clock
2	CAN_RESET	I	H	ASYNC	Reset. When this signal is "H", initialization is performed.
3	CAN_SELECT	I	H	CAN_CLK↑	Register select signal. When this signal is "H", the register indicated by CAN_ADDR is selected.
4	CAN_WR_B	I	L	CAN_CLK↑	Access direction signal. This indicates read direction when this signal is "H" and CAN_SELECT="H", and indicates write direction when this signal is "L" and CAN_SELECT="H".
5	CAN_WR_SIZE [1:0]	I	-	CAN_CLK↑	Access size. During read, this signal is ignored and 32 bit access is performed. However, CAN_WR_SIZE="11" is disabled. · "00" : 8 bit access · "01" : 16 bit access · "10" : 32 bit access · "11" : Setting is prohibited. (32 bit access) When CAN_SELECT="H", this signal is enabled.
6	CAN_ADDR [7:0]	I	-	CAN_CLK↑	Address signal. When CAN_SELECT="H", the register for access is selected by CAN_WR_SIZE and this signal.
7	CAN_DATA_IN [31:0]	I	-	CAN_CLK↑	Writing data input to register.
8	CAN_RX	I	-	ASYNC	CAN receiving data input.

Table 5-2 Table of input and output signals (Output signal)

NO	Signal name	I/O	Polarity	EDGE*1	Initial value	Functions
9	CAN_DATA_OUT [31:0]	O	-	CAN_CLK↑	-	Register data output. When there is no read to register, "L" is returned.
10	CAN_WAIT_B	O	L	CAN_CLK↑	H	Transfer signal. This signal indicates data transferring state between message RAM and interface register. When this signal is "L", access to interface register (IF1/IF2) is disabled.
11	CAN_INT	O	H	CAN_CLK↑	L	Interrupt signal. When this signal is "H", interrupt is requested.
12	CAN_TX	O	-	CAN_CLK↑	H	CAN transmit data output.

*1: Timing of change is indicated.

CHAPTER 6-1: HDMI-CEC/Remote Control Reception

HDMI-CEC/remote control reception is explained as follows.

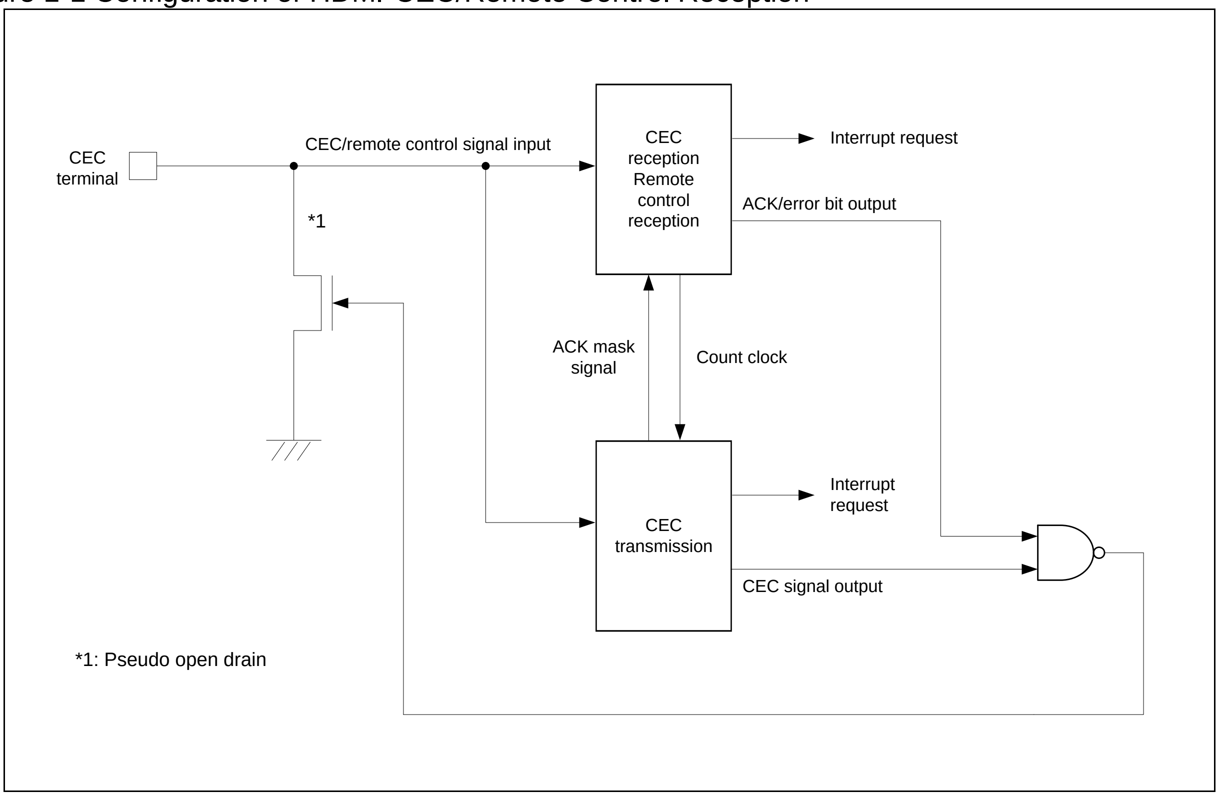
CODE: 9BFRCECTOP-E1.0

1. Configuration

Configuration of HDMI-CEC/remote control reception is as follows.

■ Configuration

Figure 1-1 Configuration of HDMI-CEC/Remote Control Reception



● CEC Reception/Remote Control Reception

See a separate chapter "CEC Reception/Remote Control Reception".

● CEC Transmission

See a separate chapter "CEC Transmission".

2. Revision

Revision of HDMI-CEC/remote control reception in each product is as follows.

Table 1 List of revision in each product

Revision	Product number		
RCCEC_rev1	MB9AFB44L MB9AFB44M MB9AFB44N	MB9AFB42L MB9AFB42M MB9AFB42N	MB9AFB41L MB9AFB41M MB9AFB41N
	MB9AFA44L MB9AFA44M MB9AFA44N	MB9AFA42L MB9AFA42M MB9AFA42N	MB9AFA41L MB9AFA41M MB9AFA41N
	MB9AF344L MB9AF344M MB9AF344N	MB9AF342L MB9AF342M MB9AF342N	MB9AF341L MB9AF341M MB9AF341N
	MB9AF144L MB9AF144M MB9AF144N	MB9AF142L MB9AF142M MB9AF142N	MB9AF141L MB9AF141M MB9AF141N
	MB9AF156M MB9AF156N MB9AF156R	MB9AF155M MB9AF155N MB9AF155R	MB9AF154M MB9AF154N MB9AF154R
	MB9AFA32L, MB9AFA32M, MB9AFA32N		MB9AFA31L, MB9AFA31M, MB9AFA31N
	MB9AF132M, MB9AF132N		MB9AF131M, MB9AF131N
RCCEC_rev2	MB9AFB44LA MB9AFB44MA MB9AFB44NA MB9AFB44LB MB9AFB44MB MB9AFB44NB	MB9AFB42LA MB9AFB42MA MB9AFB42NA MB9AFB42LB MB9AFB42MB MB9AFB42NB	MB9AFB41LA MB9AFB41MA MB9AFB41NA MB9AFB41LB MB9AFB41MB MB9AFB41NB
	MB9AFA44LA MB9AFA44MA MB9AFA44NA MB9AFA44LB MB9AFA44MB MB9AFA44NB	MB9AFA42LA MB9AFA42MA MB9AFA42NA MB9AFA42LB MB9AFA42MB MB9AFA42NB	MB9AFA41LA MB9AFA41MA MB9AFA41NA MB9AFA41LB MB9AFA41MB MB9AFA41NB
	MB9AF344LA MB9AF344MA MB9AF344NA MB9AF344LB MB9AF344MB MB9AF344NB	MB9AF342LA MB9AF342MA MB9AF342NA MB9AF342LB MB9AF342MB MB9AF342NB	MB9AF341LA MB9AF341MA MB9AF341NA MB9AF341LB MB9AF341MB MB9AF341NB
	MB9AF144LA MB9AF144MA MB9AF144NA MB9AF144LB MB9AF144MB MB9AF144NB	MB9AF142LA MB9AF142MA MB9AF142NA MB9AF142LB MB9AF142MB MB9AF142NB	MB9AF141LA MB9AF141MA MB9AF141NA MB9AF141LB MB9AF141MB MB9AF141NB
	MB9AF156MA MB9AF156NA MB9AF156RA	MB9AF155MA MB9AF155NA MB9AF155RA	MB9AF154MA MB9AF154NA MB9AF154RA

Revision	Product number		
RCCEC_rev2	MB9AFAA2L MB9AFAA2M MB9AFAA2N	MB9AFAA1L MB9AFAA1M MB9AFAA1N	
	MB9AF1A2M MB9AF1A2N	MB9AF1A1M MB9AF1A1N	
	MB9BF529S MB9BF529T	MB9BF528S MB9BF528T	
	MB9BF429S MB9BF429T	MB9BF428S MB9BF428T	
	MB9BF329S MB9BF329T	MB9BF328S MB9BF328T	
	MB9BF129S MB9BF129T	MB9BF128S MB9BF128T	
RCCEC_rev3	MB9AF156MB MB9AF156NB MB9AF156RB	MB9AF155MB MB9AF155NB MB9AF155RB	MB9AF154MB MB9AF154NB MB9AF154RB
	MB9BF529SA MB9BF529TA	MB9BF528SA MB9BF528TA	
	MB9BF429SA MB9BF429TA	MB9BF428SA MB9BF428TA	
	MB9BF329SA MB9BF329TA	MB9BF328SA MB9BF328TA	
	MB9BF129SA MB9BF129TA	MB9BF128SA MB9BF128TA	

3. Usage Notes of HDMI-CEC

RCCEC_rev1

- If external load is large, arbitration lost is occurred. Countermeasure is necessary. (ex. reduce pull-up resistor)

Other than RCCEC_rev3

- If "Polling message" is sent, set "0x0F" to RCADR1 or RCADR2 register for NACK response.
- If ACK response is done to other device transmission when NACK response is set for "Polling message, set below.

1. Store 0x0 to SFREE register.
2. Monitor CEC line with GPIO and wait until 1 lasts for the signal free time.
3. Store frame data to TXDATA register and store 0x0F to RCADR1 or RCADR2 register.
   It sends a massage after 3~4 clocks of 32.768 kHz clock when TXDATA is stored 0x0F.

If the device receives a frame from another node within 2~3 clocks after storing TXDATA, the bus error occurs and if the device receives a frame from another node within 3~4 clocks after storing TXDATA, the arbitration lost occurs. In these cases:

4-A-1. Set RCADR1 or RCADR2 to former value from 0x0F to reply ACK

4-A-2. Return back to step 2 above

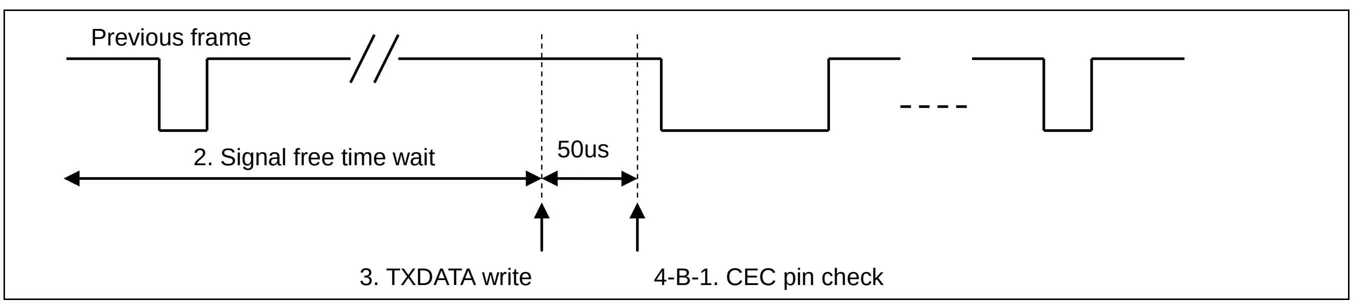
If the device receives a frame from another node within 1~2 clocks after storing TXDATA, take these steps.

4-A-3. Monitor CEC line with GPIO after 50µs from storing TXDATA

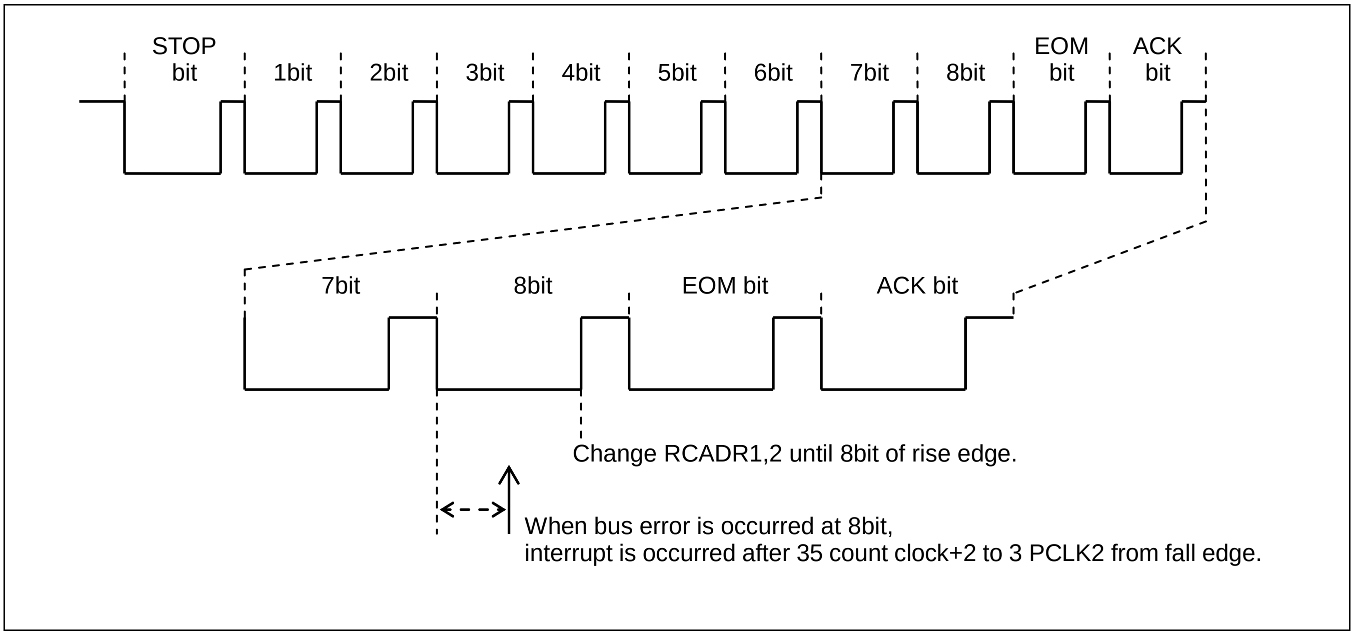
4-A-4. Set TXEN to 1-> 0 -> 1 immediately when GPIO finds state low on the CEC line

4-A-5. Set RCADR1 or RCADR2 to former value from 0x0F to reply ACK

4-A-6. Return back to step 2 above



- If RCADR1 or RCADR2 is changed in middle of communication and bus error occurred, change until rise edge of 8bit.



CHAPTER 6-2: CEC Reception/Remote Reception

Functions and operations of CEC reception/remote reception are explained as follows.

1. Overview
2. Configuration
3. Operations
4. Example of Setting
5. Registers

CODE: 9BFRCEC-E1.0

1. Overview

CEC reception/remote reception is used for receiving HDMI-CEC signals and infrared remote control signals. The features are as follows.

■ Features

- Capable of adjusting detection timings for start bit and data bit
- Equipped with noise filter
- Operating modes supporting the following standards can be selected
  - SIRCS
  - NEC/Association for Electric Home Appliances
  - HDMI-CEC

■ Features of operating modes

● SIRCS mode

- Start bit detection and interrupt output
- Minimum pulse width violation detection
- Device address comparison
- Counter overflow detection and interrupt output

● NEC/Association for Electric Home Appliances mode

- Start bit detection and interrupt output
- Repeat code detection and interrupt output
- Minimum pulse width violation detection
- Counter overflow detection and interrupt output

● HDMI-CEC mode

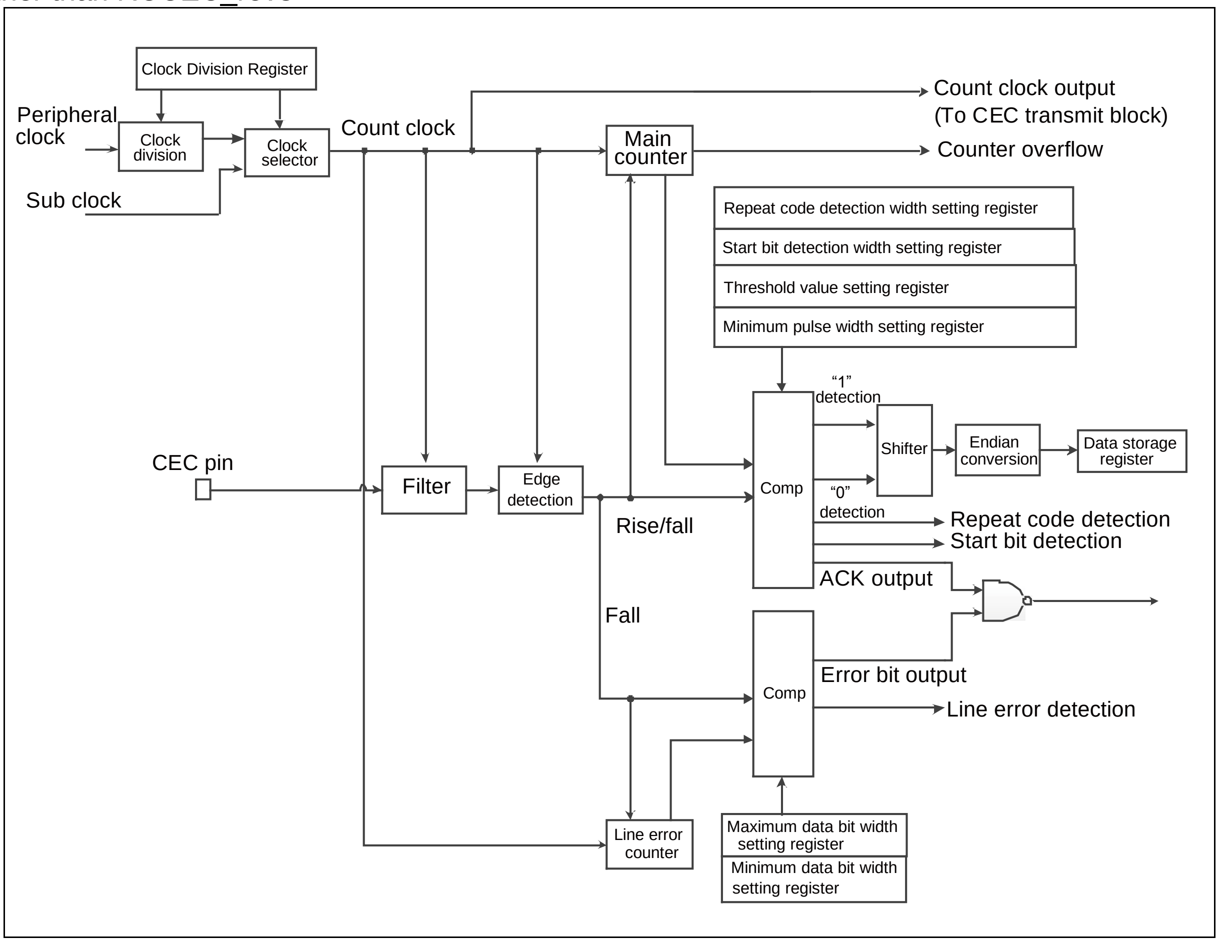
- Start bit detection and interrupt output
- Minimum pulse width violation detection
- Counter overflow detection and interrupt output
- Device address comparison
- Minimum data bit width violation detection and interrupt output (supporting HDMI-CEC line error handling standard)
- Automatic error pulse output (supporting HDMI-CEC line error handling standard)
- Maximum data bit width violation detection and interrupt output
- EOM detection
- ACK detection and interrupt output
- Automatic ACK output

2. Configuration

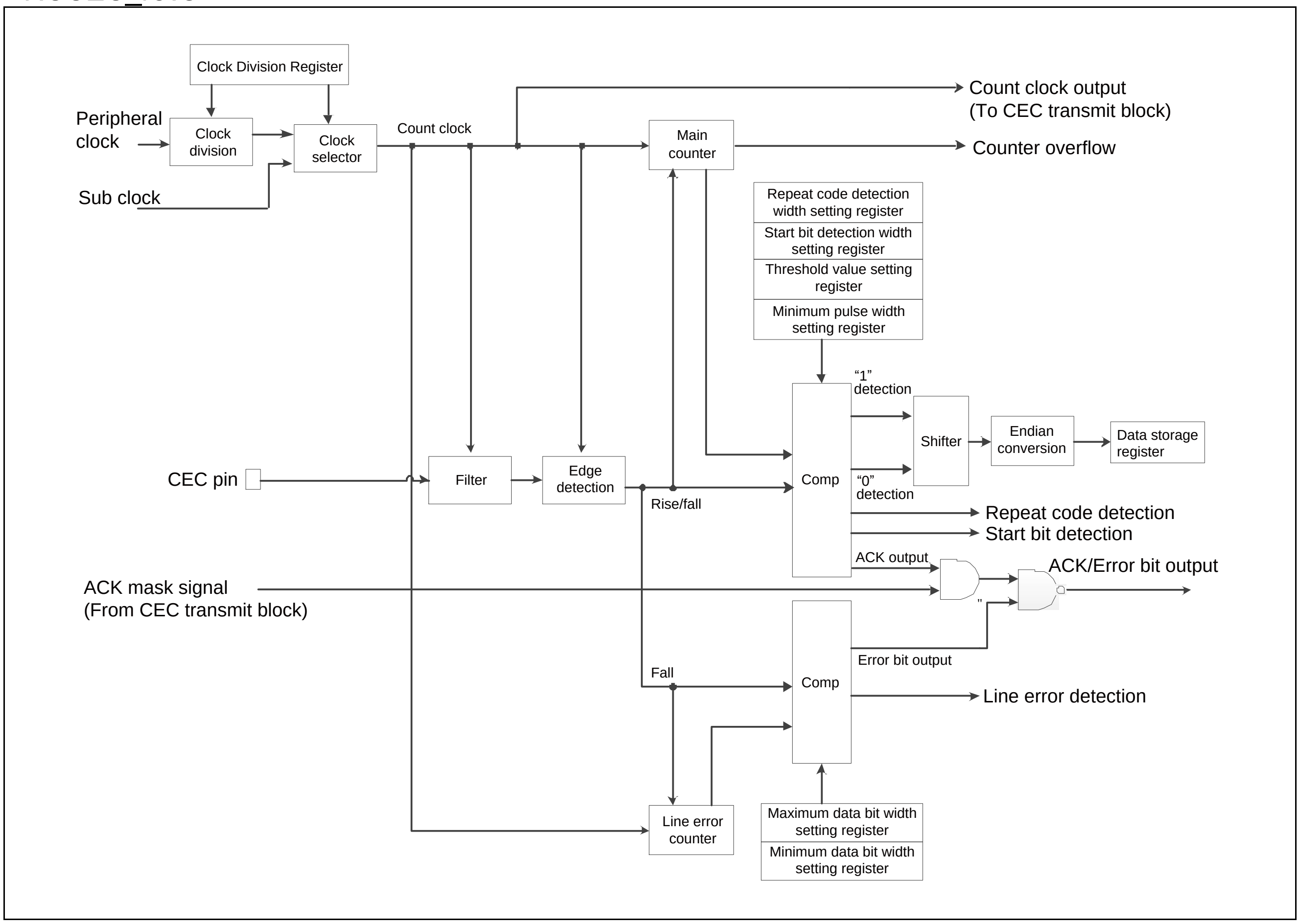
Block diagram of CEC reception/remote reception is as follows.

■ Block Diagram

Figure 2-1 CEC Reception/Remote Reception Block Diagram
- Other than RCCEC_rev3



- RCCEC_rev3


Clock Division Register
Peripheral clock
Clock division
Clock selector
Sub clock
Count clock
Main counter
Count clock output
(To CEC transmit block)
Counter overflow
Repeat code detection width setting register
Start bit detection width setting register
Threshold value setting register
Minimum pulse width setting register
"1" detection
"0" detection
Shifter
Endian conversion
Data storage register
CEC pin
Filter
Edge detection
Rise/fall
Comp
Repeat code detection
Start bit detection
ACK output
ACK/Error bit output
ACK mask signal
(From CEC transmit block)
Error bit output
Fall
Comp
Line error detection
Line error counter
Maximum data bit width setting register
Minimum data bit width setting register

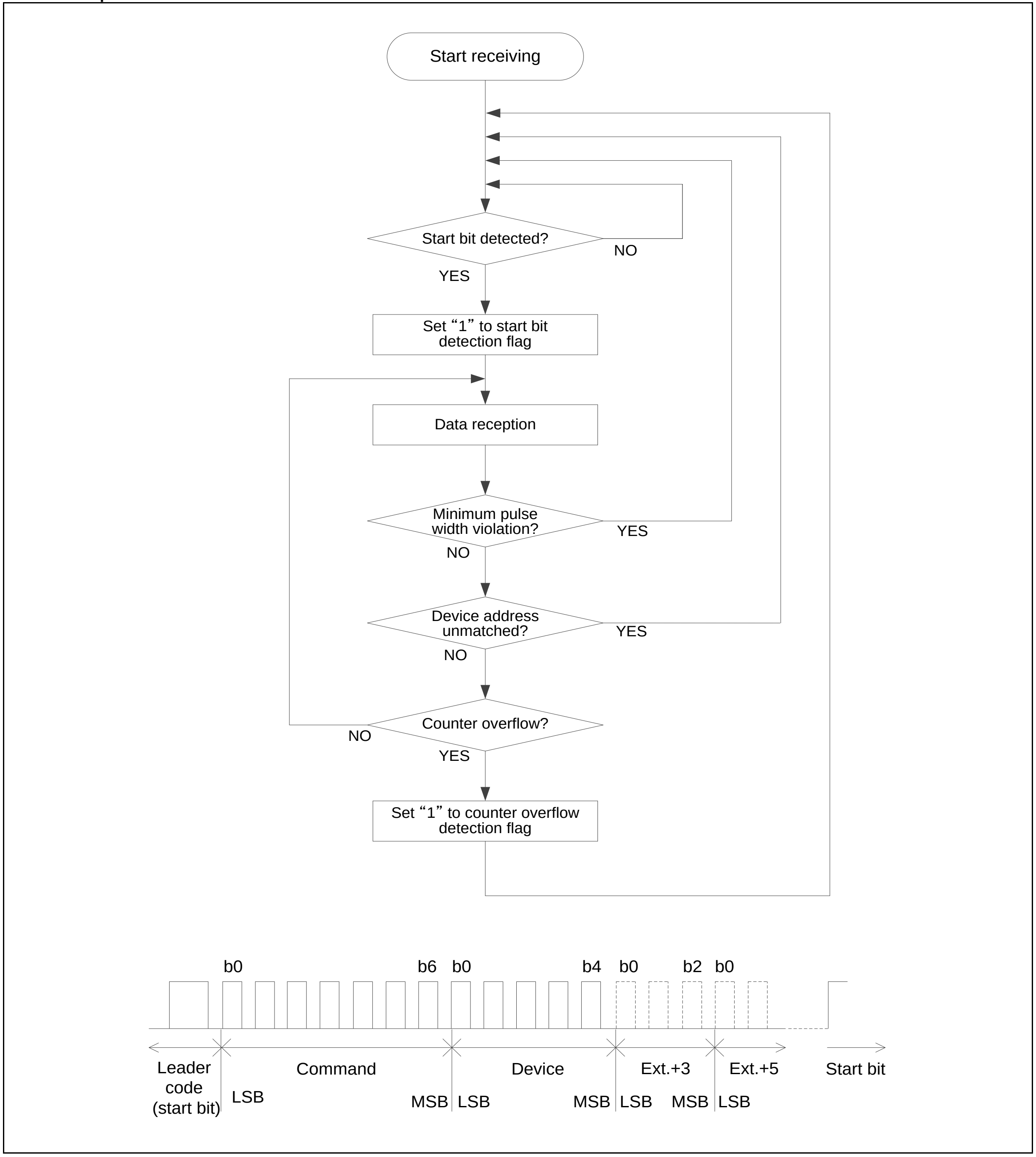
3. Operations

This chapter explains the operations of CEC reception/remote control reception.

3.1. SIRCS mode

3.1.1. Operational flow chart and waves of SIRCS mode

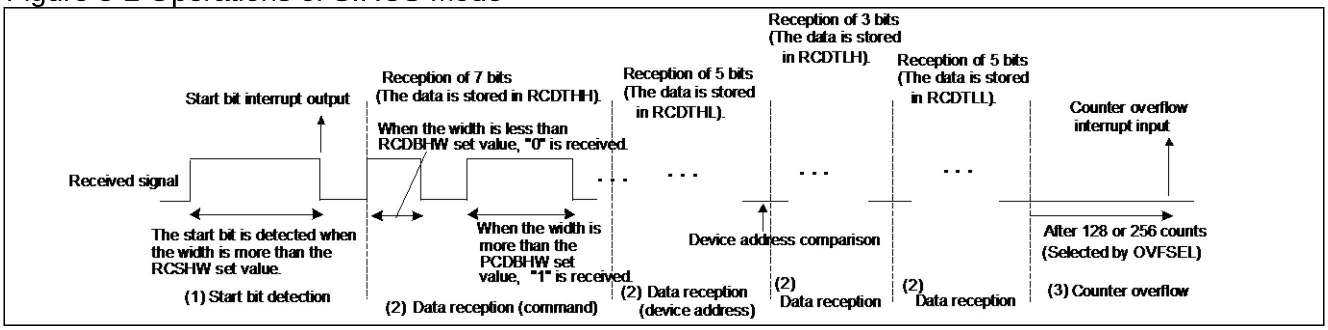
Figure 3-1 Operational Flow Chart and Waves



3.1.2. Basic operations of SIRCS mode

The SIRCS mode counts the width of "High" duration in the received signal with the count clock, and receives the data.

Figure 3-2 Operations of SIRCS Mode



■ Basic operations

The basic operations are as follows:

(1) If the width of "High" duration more than the set value of RCSHW is input, the start bit is detected and the data receiving state is entered.

(2) Figure 3-2 shows the operation at THSEL=0 (RCCR register). In the operation, "0" is received for the signal less than the RCDBHW set value and "1" is received for the signal more than the RCDBHW set value.
After receiving the 7-bit command, the device address is received for the data reception. 5-bit device address becomes an address match if its address is the same as either of RCADR1 or RCADR2 value. When the address is not matched with the both values, the state returns to the start bit detection waiting state.

(3) For overflowing after data is received, the start bit detection waiting state is resumed.

3.1.3. Start bit detection and interrupt output

Figure 3-3 Start Bit Detection of SIRCS Mode

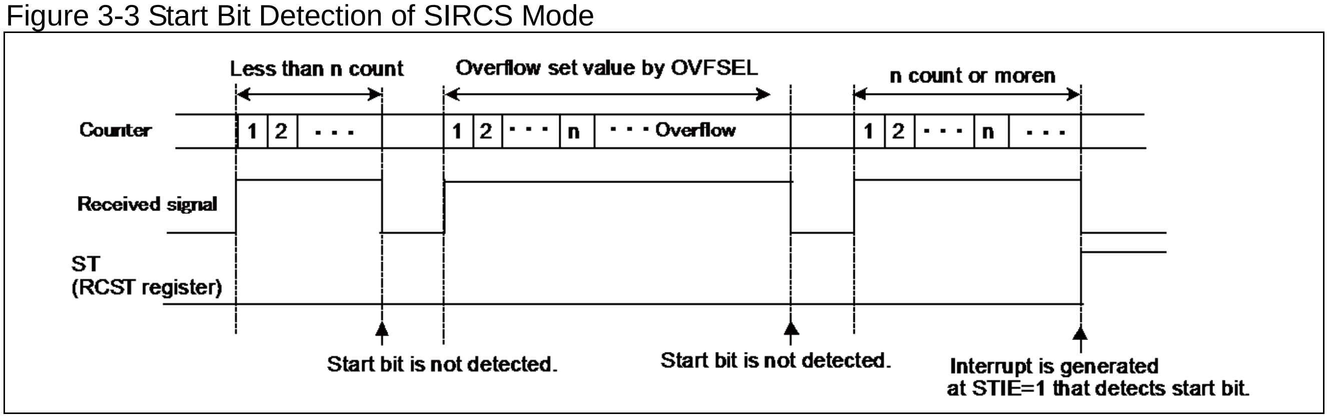


Figure 3-3 explains the start bit detection when RCSHW=n is set.
If the width of "High" duration of "n" or more is input with the start bit detection waited, ST=1 (RCST register) is set by detecting the start bit. Moreover, when STIE=1 (RCST register) is set beforehand, the interrupt is output by detecting the start bit.
Moreover, when the width of "High" duration more than the number of counts specified by OVFSEL (RCST register) setting is input, the overflow occurs and the start bit is not detected.

3.1.4. Minimum pulse width violation

Figure 3-4 Minimum Pulse Width Violation

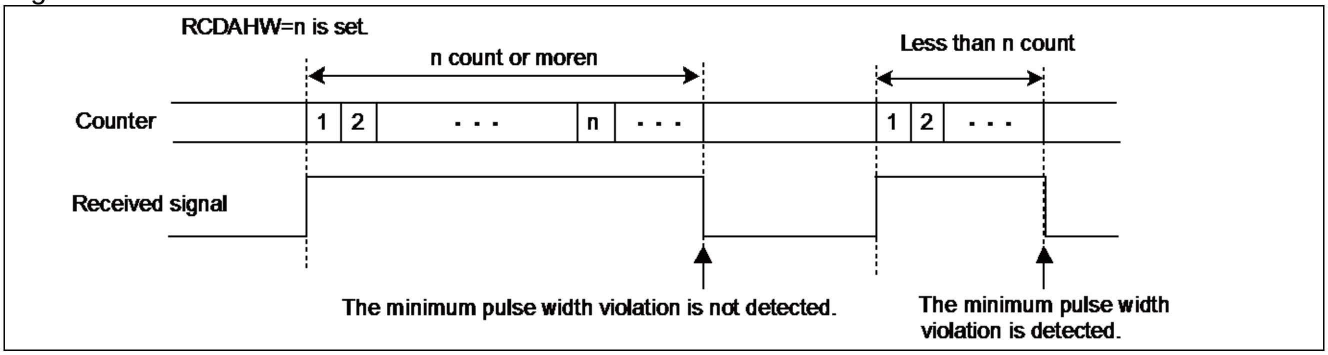


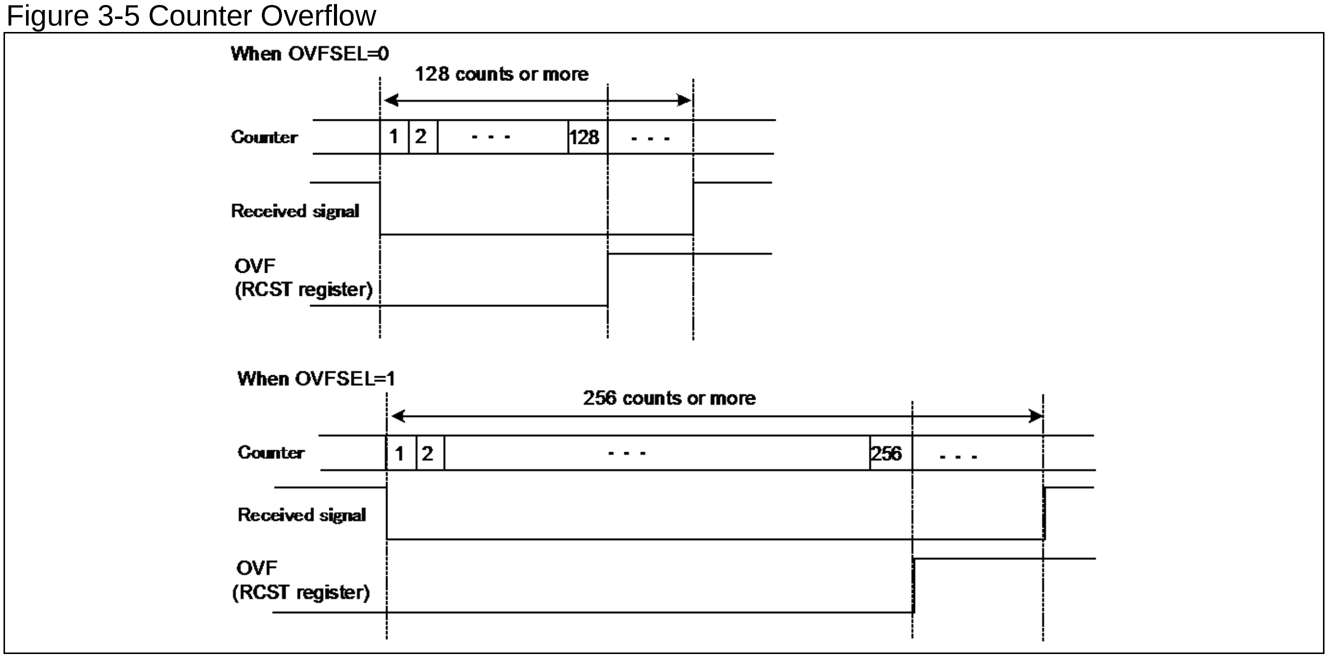
Figure 3-4 explains the minimum pulse width violation when RCDAHW=n is set.
When the signal of less than n is input during the reception operation, the state of the start bit detection waiting is resumed by detecting the minimum pulse width violation.

3.1.5. Device address comparison

In the SIRCS mode, the 5-bit device address is received. For ADRCE=1 (RCCR register), the device address comparison is executed.
The device address becomes an address match if its address is the same as either of RCADR1 or RCADR2 value. When the address is not matched with the both values, the start bit detection waiting state is resumed.

3.1.6. Counter overflow detection and interrupt output

Figure 3-5 Counter Overflow

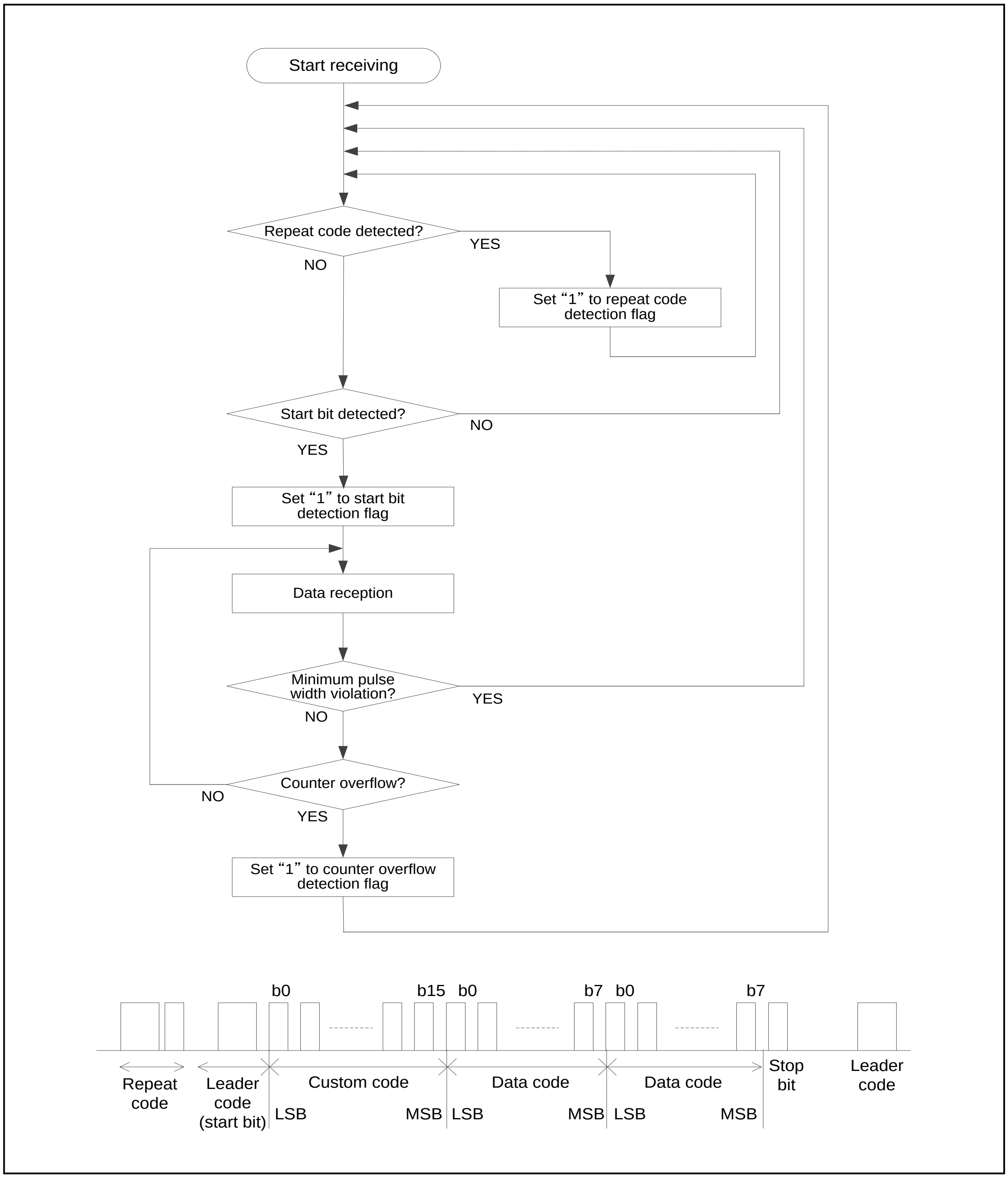


For OVFSEL=0 (RCST register), an overflow occurs and the start bit detection waiting state is resumed when High or Low input continues more than 128 counts. Moreover, for OVFSEL=1, an overflow occurs at 256 counts.
When OVFIE=1 (RCST register) is set beforehand, the interrupt is output after an overflow.

3.2. Operations of NEC/Association for Electric Home Appliances mode

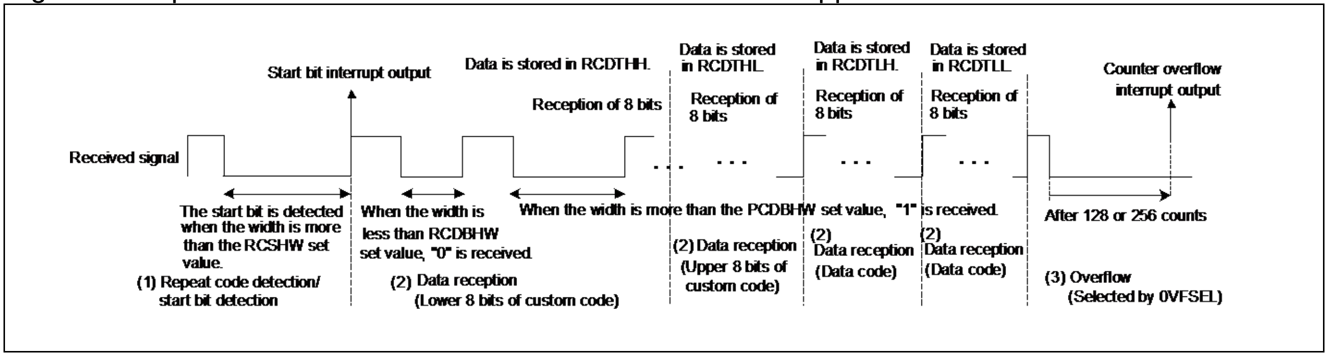
3.2.1. Operational flow chart and waves of NEC/Association for Electric Home Appliances mode

Figure 3-6 Operational flow chart and waves of NEC/ Association for Electric Home Appliances Mode



In NEC/Association for Electric Home Appliances mode, the count clock counts the width of “Low” duration of the received signal and the data is received.

Figure 3-7 Operations of NEC/ Association for Electric Home Appliances mode



■ Basic Operations

The basic operations are as follows:

(1) When the width of "Low" duration of the RCSHW set value or less and the RCRHW set value or more is input, the repeat code is detected. Moreover, if the width of "Low" duration of the RCSHW set value or more is input, the data reception state is entered by detecting the start bit.

(2) Figure 3-7 shows the operations for THSEL=0 (RCCR register). In the operations, "0" is received for the signal of less than the RCDBHW set value and "1" is received for the signal of the RCDBHW set value or more.
In the data reception, the custom code of two bytes and data code of two bytes are received

(3) When an overflow occurs after the data reception, the start bit/repeat bit detection waiting state is resumed.

3.2.2. Start bit detection

Figure 3-8 Start bit detection

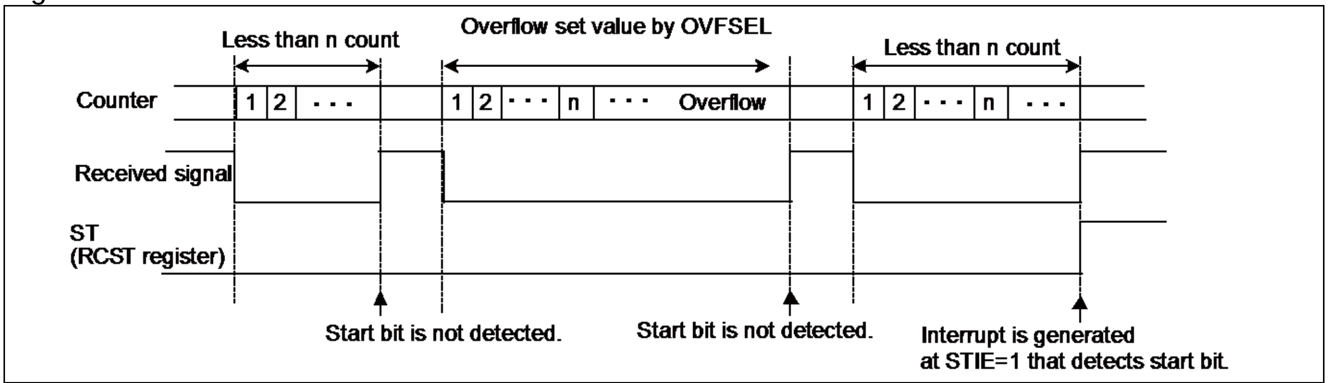


Figure 3-8 explains the start bit detection when "RCSHW=n" is set.
When the width of "Low" duration of n or more is input during the start bit detection waiting, ST=1 (RCST register) is set by detecting the start bit. Moreover, when STIE=1 (RCST register) is set beforehand, the interrupt is output by detecting the start bit.
Moreover, when the width of "Low" duration of the number of counts specified by OVFSEL (RCST register) setting or more is input, an overflow occurs and the start bit is not detected.

3.2.3. Repeat code detection

Figure 3-9 Repeat code detection

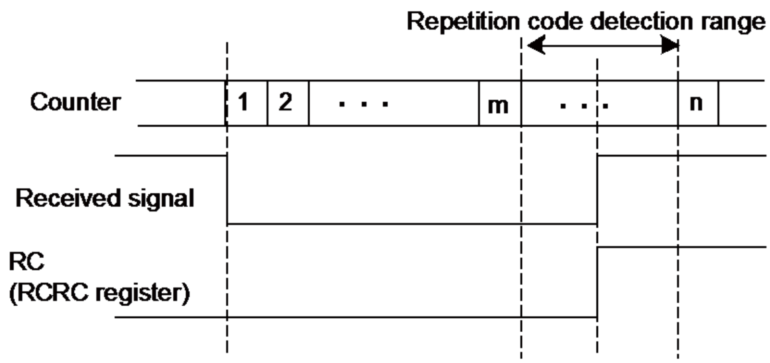


Figure 3-9 explains the start bit detection when RCRHW=m and RCSHW=n are set.
When the "Low" signal of the width of less than n and m or more is input at the reception beginning, RC=1 (RCRC register) is set by detecting the repeat code.
The repeat code is detected only in NEC/Association for Electric Home Appliances mode.

3.2.4. Minimum pulse width violation

Figure 3-10 Minimum pulse width violation

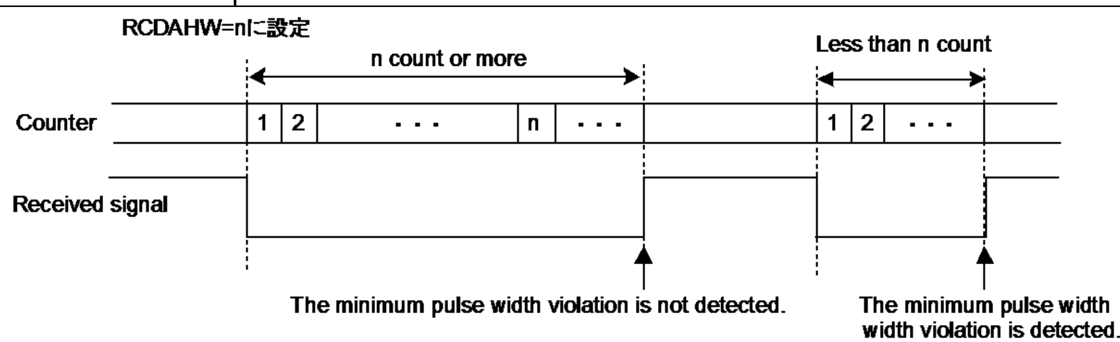
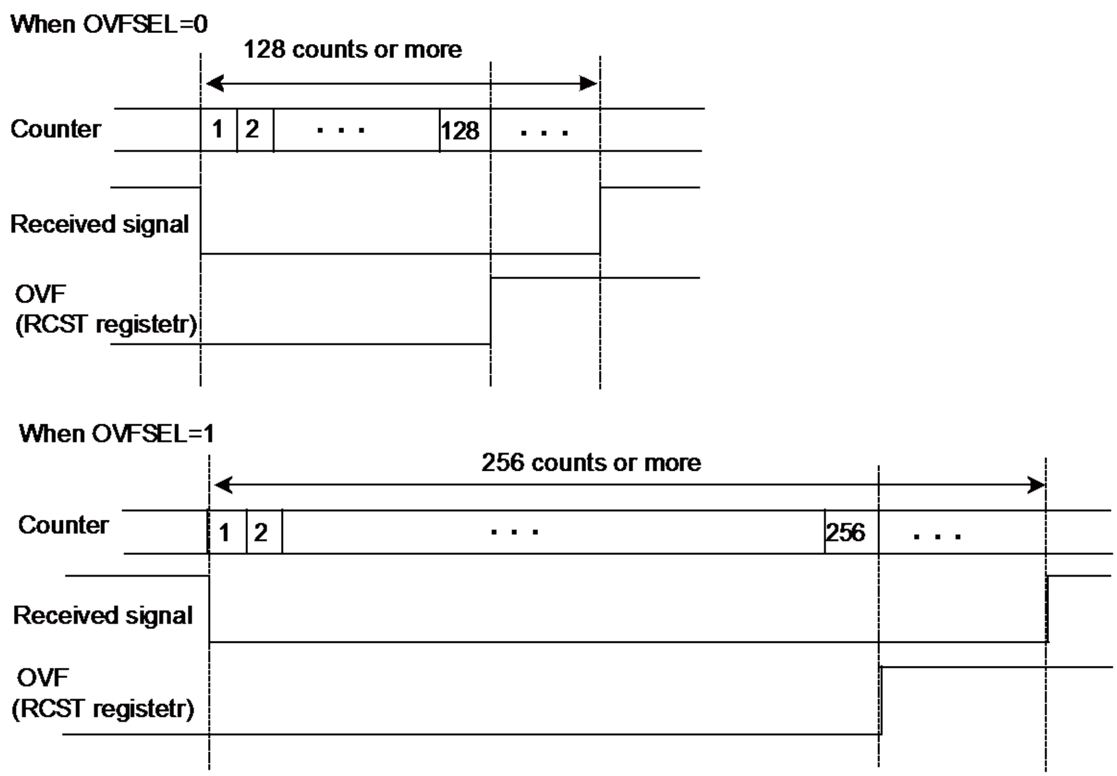


Figure 3-10 explains the minimum pulse width violation when RCDAHW=n is set.
When the width of "Low" duration of less than n is input during the reception operation, the start bit detection waiting state is resumed by detecting the minimum pulse width violation.

3.2.5. Counter overflow detection and interrupt output

Figure 3-11 Counter overflow



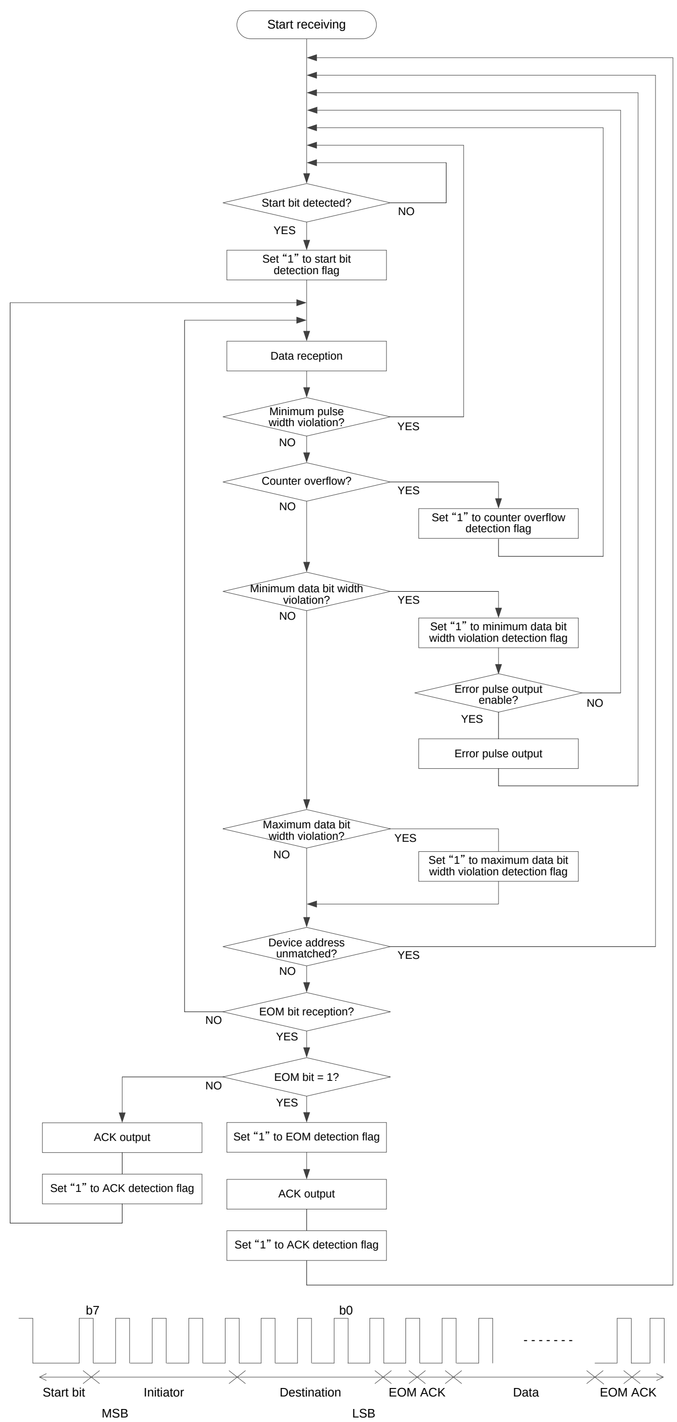
If the "High" or "Low" input of 128 counts or more continues for OVFSEL=0(RCST register), an overflow occurs and the start bit detection waiting state is resumed. Moreover, an overflow occurs with 256 counts of the continuous "High" or "Low" input for OVFSEL=1.

When OVFIE=1 (RCST register) is set beforehand, an overflow occurs and an interrupt is output.

3.3. HDMI-CEC mode

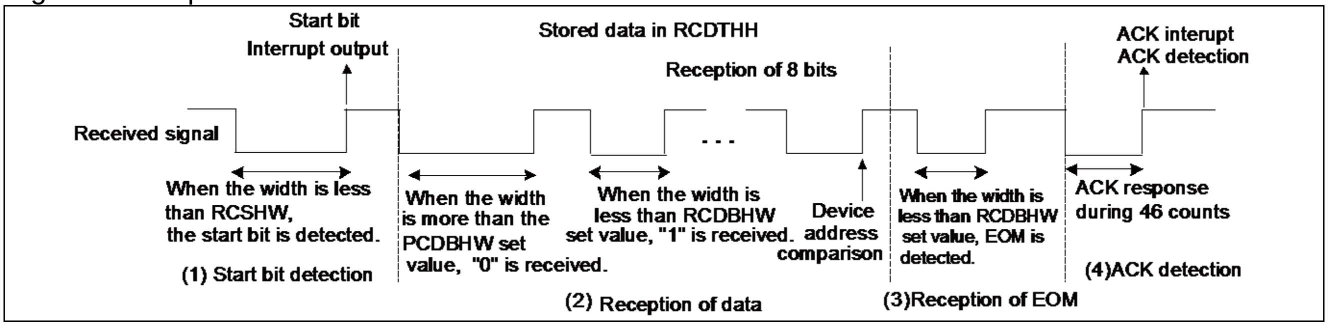
3.3.1. Operational flow chart and waves in HDMI-CEC mode

Figure 3-12 Operational flow chart and waves in HDMI-CEC mode



In the HDMI-CEC mode, the count clock counts the width of "Low" duration of the received signal and the data is received.

Figure 3-13 Operations in HDMI-CEC mode



■ Basic Operations

The basic operations are as follows:

(1) When the width of "Low" duration of less than the RCSHW set value is input, the start bit is detected and the data receiving state is resumed.

(2) Figure 3-13 shows the operations at THSEL=1 (RCCR register). For a signal of the RCDBHW set value or more, "0" is received, and for a signal of less than the RCDBHW set value, "1" is received.
Received data of 8 bits is stored in RCDTHH and the lower 4 bits are compared with the device address. If the destination of 4 bits is the same as either of RCADR1 or RCADR2 value, the address becomes the address match. When the address is not matched with the both values, the start bit detection waiting state is resumed.

(3) When EOM is detected after the data reception, EOM=1 (RCST register) is set and the data reception is completed. When EOM is not detected, EOM=0 (RCST register) is held and the data receiving state is resumed to store the received data in RCDTHH again.

(4) When "Low" signal is input after the reception of the EOM bit, the ACK signal is output and the start bit detection waiting state is resumed.

3.3.2. Start bit detection and interrupt output

Figure 3-14 Detection of start bit in HDMI-CEC mode

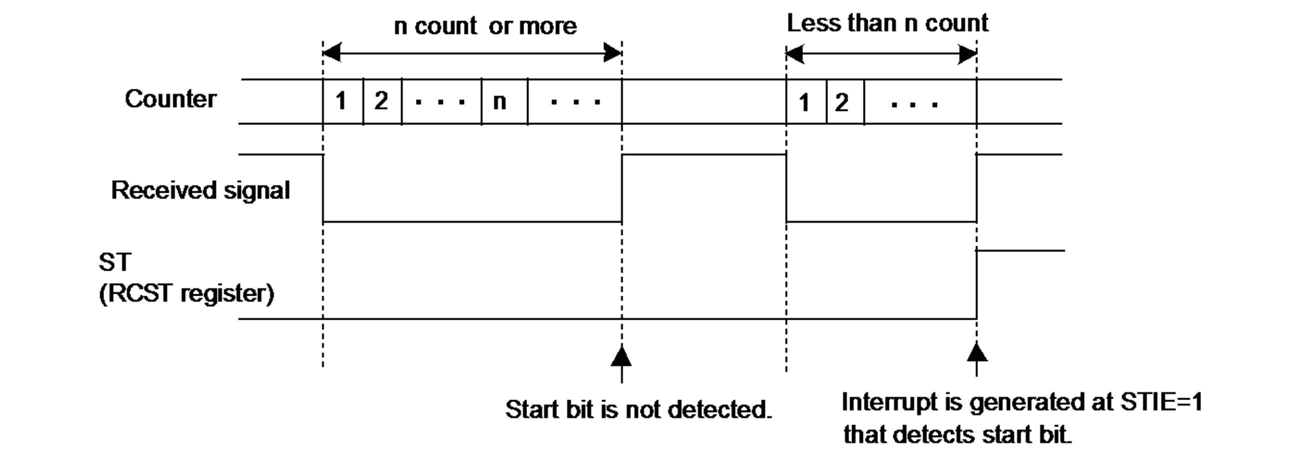


Figure 3-14 shows the start bit detection when "RCSHW=n" is set (the operations for THSEL=1). When the width of "Low" duration of less than n is input with the start bit detection waiting, the start bit is detected and ST=1 (RCST register) is set. Moreover, when STIE=1 (RCST register) is set beforehand, the interrupt is output by detecting the start bit.

3.3.3. Minimum pulse width violation

Figure 3-15 Minimum pulse width violation

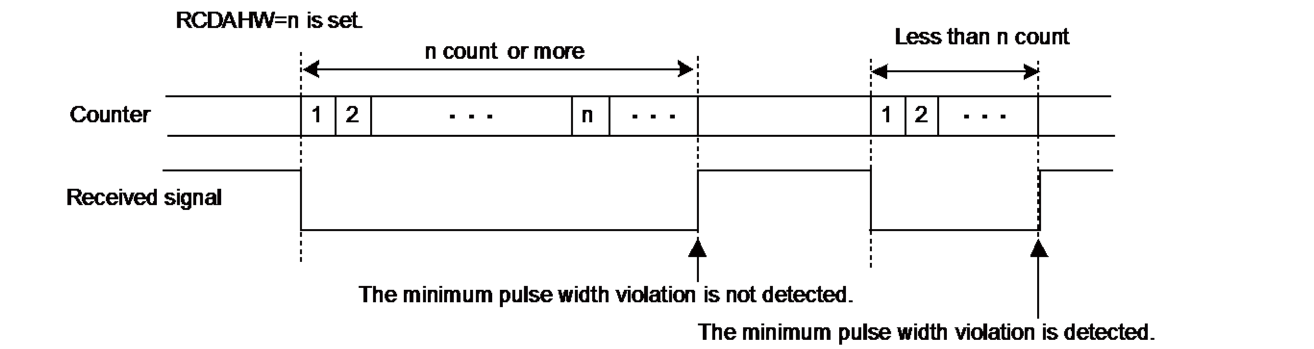
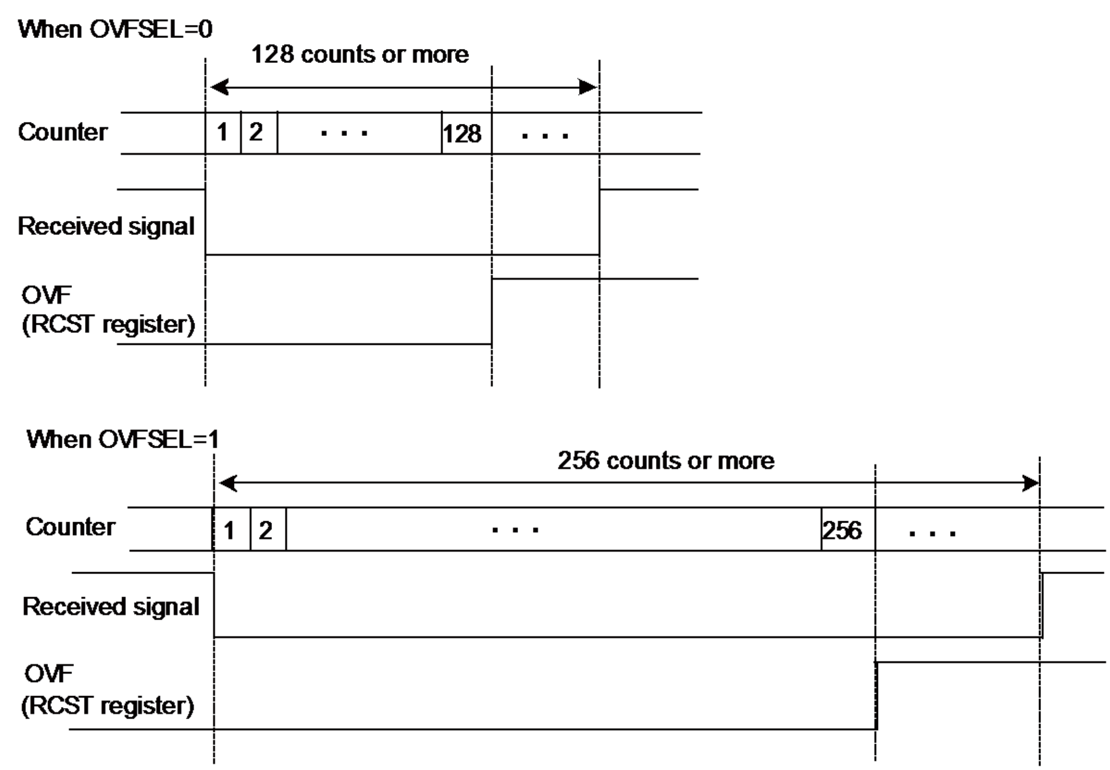


Figure 3-15 shows the minimum pulse width violation when RCDAHW=n is set.
When the signal of less than n is input during the reception operation, the minimum pulse width violation is detected and the start bit detection waiting state is resumed.

3.3.4. Counter overflow detection and interrupt output

Figure 3-16 Counter overflow



If the "High" or "Low" input of 128 counts or more continues for OVFSEL=0(RCST register), an overflow occurs and the start bit detection waiting state is resumed. Moreover, an overflow occurs with 256 counts of the continuous "High" or "Low" input for OVFSEL=1.
When "OVFIE=1 (RCST register) " is set beforehand, an overflow occurs and an interrupt is output.

3.3.5. Device address comparison

In the HDMI-CEC mode, the destination of 4 bits is received. For ADRCE=1 (RCCR register), the device address comparison is executed.
If the destination is the same as either of RCADR1 or RCADR2 value, the address becomes the address match. Moreover, for the broadcast address, an address match is achieved.
When the address is not matched with the both values, the start bit detection waiting state is resumed.

3.3.6. Data bit width violation and error pulse automatic output

Figure 3-17 Minimum data bit width violation

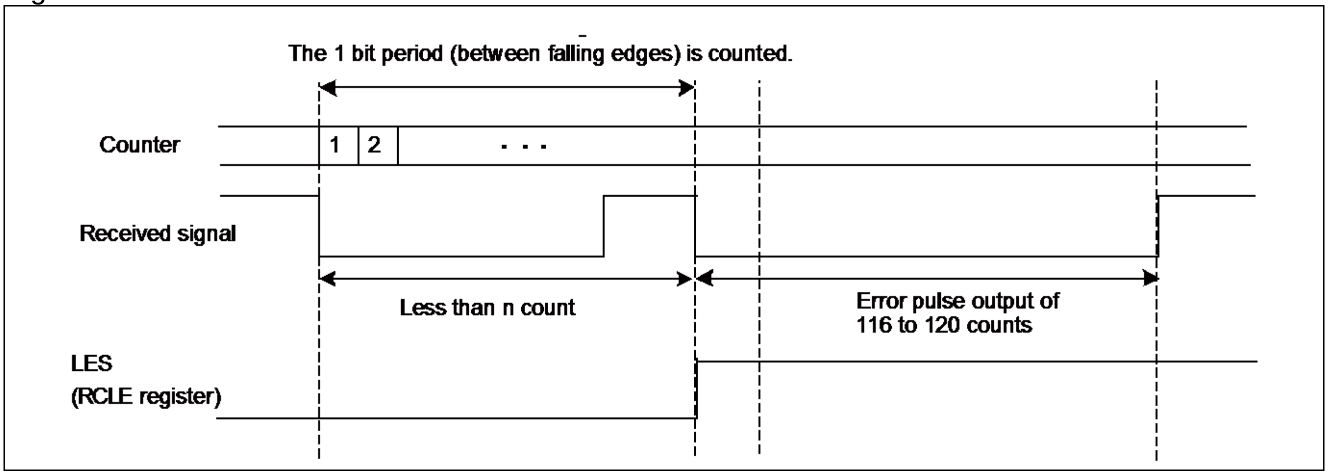


Figure 3-17 explains the minimum data bit width violation when RCLESW=n is set.
At LES=1 (RCLE register), when the 1 bit period (the period between the falling edges) is smaller than the set value of minimum data bit width setting register (RCLESW), the minimum data bit width violation is detected and LES=1 (RCLE register) is set.
When LESIE=1 (RCLE register) is set beforehand, the interrupt is output by detecting the violation of minimum data bit width. Moreover, when EPE=1 (RCLE register) is set, by detecting the violation, the error pulse is output as shown in Figure 3-17.

Figure 3-18 Maximum data bit width violation

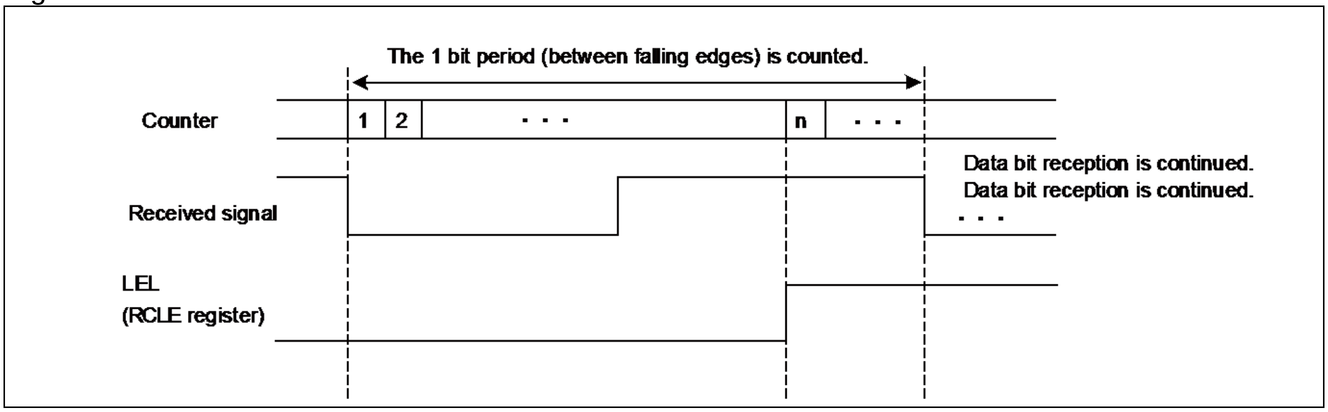


Figure 3-17 explains the minimum data bit width violation when RCLELW=n is set.
For LELE=1 (RCLE register), when the 1 bit period (the period between the falling edges) is more than the set value of maximum data bit width setting register (RCLELW), LEL=1 (RCLE register) is set by detecting the maximum data bit width violation. When LELIE=1 (RCLE register) is set beforehand, the interrupt is output by detecting the maximum data bit width violation.

3.3.7. EOM detection

Figure 3-19 EOM detection

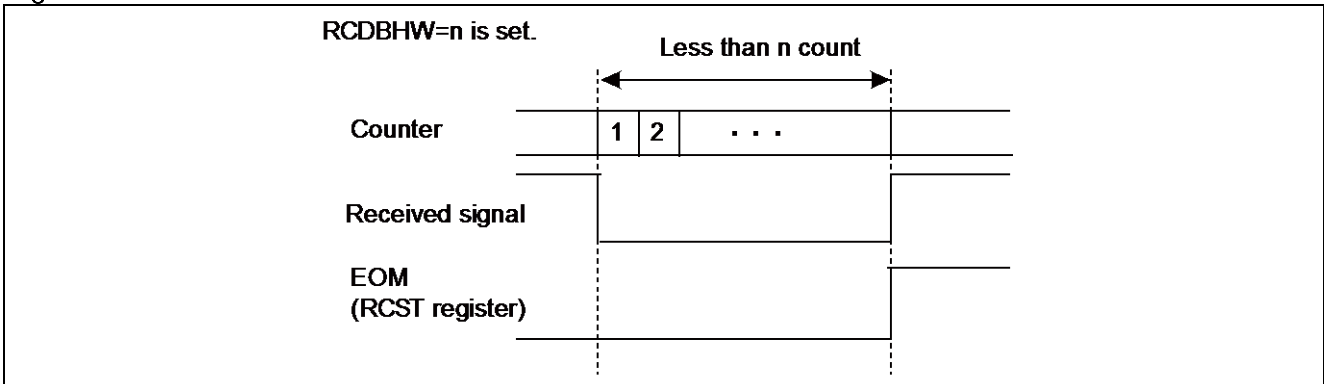
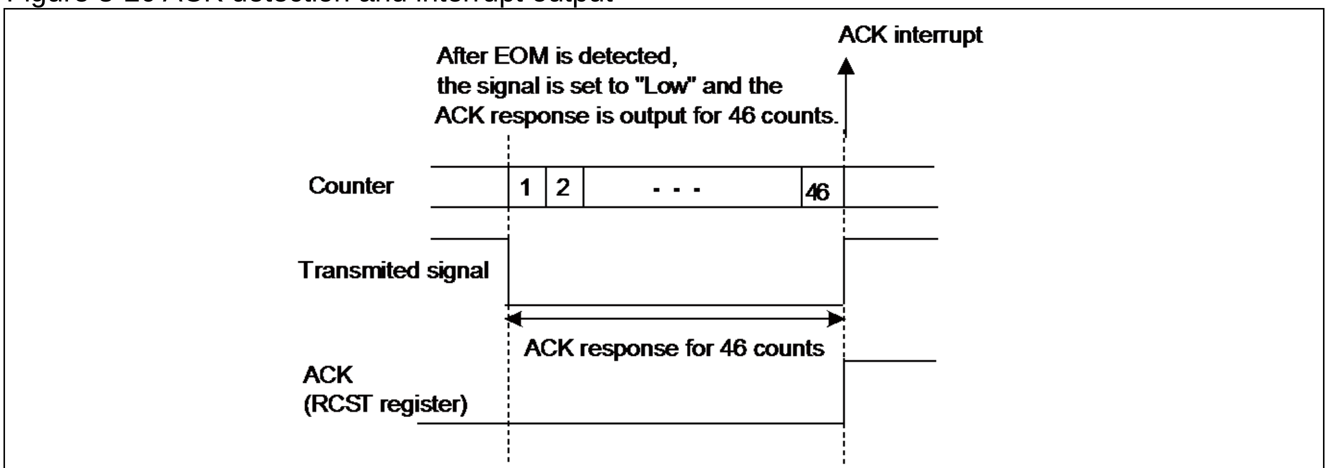


Figure 3-19 shows the operation for THSEL=1 (RCCR register). If the "Low" signal of less than RCDBHW set value is input in EOM bit receiving state, EOM=1 (RCST register) is set by detecting EOM.

3.3.8. ACK detection and interrupt output

Figure 3-20 ACK detection and interrupt output



When "Low" signal is input after EOM detection, "Low" signal is output for 46 counts as ACK response. If the "High" signal is input after "Low" signal is output, ACK=1 (RCST register) is set by detecting the ACK signal. When ACKIE=1 (RCST register) is set beforehand, the interrupt is output by detecting ACK signal.
When address enable bit (ADRCE) of the RCCR register is "1", ACK signal is output only if the address match is detected. For the broadcast address, though it is considered to be the address match, ACK response is not executed.

Table 4-1 ACK output and ACK interrupt

Received destination address	ADRCE	RCADR1, RCADR2		ACK output*	ACK interrupt
0x0 to 0xE	0	-		ACK	occur
	1	0x00 to 0x0E	match	ACK	occur
			not match	NACK	not occur
		0x0F		NACK	not occur
0xF	-	-		NACK	occur

*: When ACKMEN bit of CEC transmission unit is 1 and during transmission, it will always be NACK.

3.4. Noise filter

When the input of CEC signal changes in the width of less than two clocks of the count clock, the input signal is judged to be a noise and removed.

4. Example of Setting

Example of setting is explained as follows (in case of operating clock at 32.768 kHz).

Table 4-1 Example of setting in remote mode (SIRCS)

Registers	Setting value	Remarks
Reception Control Register	MOD=00, THSEL=0, ADRCE=1	
Reception Interrupt Control Register	ACKIE=0, OVFIE=1	
	OVFSEL=0	3.9 ms
Start Bit Detection Width Setting Register	76	2.3 ms
Minimum Pulse Width Setting Register	17	0.52 ms
Threshold Value Setting Register	37	1.1 ms

Table 4-2 Example of setting in remote mode (NEC)

Registers	Setting value	Remarks
Reception Control Register	MOD=10, THSEL=0	
Reception Interrupt Control Register	ACKIE=0, OVFIE=1	
	OVFSEL=1	7.8 ms
Start Bit Detection Width Setting Register	144	4.4 ms
Minimum Pulse Width Setting Register	15	0.46 ms
Threshold Value Setting Register	52	1.6 ms
Repeat Code Interrupt Control Register	RCIE=1	
Repeat Code Detection Width Setting Register	65	2.0 ms

Table 4-3 Example of setting in HDMI-CEC remote mode

Registers	Setting value	Remarks
Reception Control Register	MOD=11, THSEL=1, ADRCE=1	
Reception Interrupt Control Register	ACKIE=1, OVFIE=1	
	OVFSEL=1	7.8 ms
Start Bit Detection Width Setting Register	114	3.5 ms
Minimum Pulse Width Setting Register	13	0.4 ms
Threshold Value Setting Register	42	1.3 ms
Maximum/Minimum Data Bit Width Violation Control Register	LELIE=1, LESIE=1, LELE=1, LESE=1, EPE=1	
Maximum Data Bit Width Setting Register	91	2.8 ms
Minimum Data Bit Width Setting Register	65	2.0 ms

5. Registers

The list of registers is as follows.

Table 5-1 Registers List

Abbreviated Register Name	Register Name	Reference
RCCR	Reception Control Register	5.1
RCST	Reception Interrupt Control Register	5.2
RCADR1	Device Address Setting Register 1	5.3
RCADR2	Device Address Setting Register 2	5.3
RCSHW	Start Bit Detection Width Setting Register	5.4
RCDAHW	Minimum Pulse Width Setting Register	5.5
RCDBHW	Threshold Value Setting Register	5.6
RCDTHH	Data Save Register HH	5.7
RCDTHL	Data Save Register HL	
RCDTLH	Data Save Register LH	
RCDTLL	Data Save Register LL	
RCCKD	Clock Division Register	5.8
RCRC	Repeat Code Interrupt Control Register	5.9
RCRHW	Repeat Code Detection Width Setting Register	5.10
RCLE	Data Bit Width Violation Interrupt Control Register	5.11
RCLESW	Minimum Data Bit Width Setting Register	5.12
RCLELW	Maximum Data Bit Width Setting Register	5.13

5.1. Reception Control Register (RCCR)

Configuration of Reception Control Register (RCCR) bits is as follows.

bit	7	6	5	4	3	2	1	0
Field	THSEL	Reserved			ADRCE	MOD1	MOD0	EN
Attribute	R/W				R/W	R/W	R/W	R/W
Initial Value	0				0	0	0	0

[bit7] THSEL: Threshold value selection bit

Use RCDAHW and RCDBHW to set a reference for determining "0" or "1".

States	THSEL	
	0	1
W > RCDAHW W < RCDBHW	"0" data	"1" data
W > RCDAHW W ≥ RCDBHW	"1" data	"0" data

[bit6:4] Reserved: Reserved bits

"0" is always read.
Set "0" for write.

[bit3] ADRCE: Address comparison enable bit

Initial value of this bit is "0" (comparison disabled) and setting this bit to “1” enables comparison between reception address and device address.
An ACK/OVF interrupt will be generated only if the address is matched when comparison is enabled.
In CEC mode, an ACK response will be returned when address match is detected. If the address is an broadcast address, it will be handled as a match but no ACK response will be returned.
In modes other than SIRCS mode or HDMI-CEC mode, set this bit to “0”.

[bit2:1] MOD1, MOD0: Operation mode setting bits

bit2	bit1	Function
0	0	SIRCS mode [Initial value]
0	1	Setting prohibited
1	0	NEC/Association for Electric Home Appliances mode
1	1	HDMI-CEC mode

In modes other than SIRCS mode (MOD1=1), input signals will be inverted internally.
“H” width comparison is applied to "L" width.

[bit0] EN: Operation enable bit

Setting this bit to "1" will start reception operation.
The initial value is "0" (stop).

<Note>

Do not change the following setting registers and bits while this bit is "1" (operating).

THSEL bit, ADRCE bit and MOD bit of RCCR register
OVFSEL bit of RCST register
RCSHW, RCDAHW, RCDBHW, and RCCKD registers
RCRC, RCRHW, RCLE, RCLELW, and RCLESW registers

If RCADR1, RCADR2 is changed while this bit is "1", see CHAPTER 6-1: HDMI-CEC/Remote Control Reception 3. Usage notes of HDMI-CEC.

5.2. Reception Interrupt Control Register (RCST)

Configuration of Reception Interrupt Control Register (RCST) bits is as follows.

bit	7	6	5	4	3	2	1	0
Field	STIE	ACKIE	OVFIE	OVFSEL	ST	ACK	EOM	OVF
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

[bit7] STIE: Start bit interrupt enable bit

Value	Description
0	Interrupt disabled
1	Interrupt enabled

[bit6] ACKIE: ACK interrupt enable bit

Value	Description
0	Interrupt disabled
1	Interrupt enabled

This bit is valid only in HDMI-CEC mode.

[bit5] OVFIE: Counter overflow interrupt enable bit

Value	Description
0	Interrupt disabled
1	Interrupt enabled

This interrupt will be generated only if an overflow is detected after a start bit is detected.
No interrupt will be generated without detecting a start bit.

[bit4] OVFSEL: Counter overflow detection condition setting bit

Value	Description
0	An overflow will occur after the counter counted 128 clocks.
1	An overflow will occur after the counter counted 256 clocks.

[bit3] ST: Start bit detection bit

Value	Description
0	Start bit has not been detected
1	Start bit has been detected

Writing "0" will clear this bit.
An interrupt will be generated if a start bit is detected while STIE bit is "1".

[bit2] ACK: ACK detection bit

Value	Description
0	ACK not detected
1	ACK detected

Writing "0" will clear this bit.
An interrupt will be generated if an ACK is detected while ACKIE bit is "1".
An interrupt will be generated only if the address is matched when address comparison is enabled.
This bit is valid only in HDMI-CEC mode.

[bit1] EOM: EOM detection bit

Value	Description
0	EOM not detected
1	EOM detected

Writing "0" will clear this bit.
This bit is valid only in HDMI-CEC mode.

[bit0] OVF: Counter overflow detection bit

Value	Description
0	Counter overflow not detected
1	Counter overflow detected

An interrupt will be generated only if the address is matched when address comparison is enabled.
Writing "0" will clear this bit.
In SIRCS mode, OVF flag will not be set until the second byte is received.

5.3. Device Address Setting Register 1, 2 (RCADR1, RCADR2)

Configuration of Device Address Setting Register 1, 2 (RCADR1, RCADR2) bits is as follows.

bit	7	6	5	4	3	2	1	0
Field	Reserved			RCADR1, 2				
Attribute				R/W				
Initial Value				00000				

[bit7:5] Reserved: Reserved bits

"0" is always read.
Set "0" for write.

[bit4:0] RCADR1, 2: Device address setting bits

Address set in this register will be compared to the received device address or HDMI-CEC destination.
In HDMI-CEC mode, if “0x0F”(broadcast address) is set to this register, ACK response is not given by the an address reception including broadcast address

5.4. Start Bit Detection Width Setting Register (RCSHW)

Configuration of Start Bit Detection Width Setting Register (RCSHW) bits is as follows.

bit	7	6	5	4	3	2	1	0
Field	RCSHW							
Attribute	R/W							
Initial Value	0x00							

This register is used to set a duration of the start bit.
If "H" with a width over the set value is received, it is identified as a start bit.
If the width of received signals is less than the set value, the start bit will not be detected and it once again becomes a state to wait for detecting a start bit.
When OVFSEL=0, the set value must be RCSHW $\leq$ 127 (equal to or less than a value not to be detected as overflow).

5.5. Minimum Pulse Width Setting Register (RCDAHW)

Configuration of the Minimum Pulse Width Setting Register (RCDAHW) bits is as follows.

bit	7	6	5	4	3	2	1	0
Field	RCDAHW							
Attribute	R/W							
Initial Value	0x00							

[bit7:0] RCDAHW

This is register used to set the minimum pulse width duration.
Values to be set in this register must be: 2 ≤ RCDAHW < RCDBHW.
In CEC mode, it must be RCDAHW < 46 (less than the ACK response pulse width).
If a signal with a width < RCDAHW is received, it will be detected as minimum pulse width violation.

5.6. Threshold Value Setting Register (RCDBHW)

Configuration of the threshold Value Setting Register (RCDBHW) bits is as follows.

bit	7	6	5	4	3	2	1	0
Field	RCDBHW							
Attribute	R/W							
Initial Value	0x00							

[bit7:0] RCDBHW

This is register used to set the threshold value of data reception signal width.
Do not set a value less than RCCDAHW.
Be sure to set a value: RCCDAHW < RCCDBHW < RCSHW.

5.7. Data Save Register (RCDTHH, RCDTHL, RCDTLH, RCDTLL)

Configuration of the Data Save Register (RCDTHH, RCDTHL, RCDTLH, RCDTLL) bits is as follows.

bit	31	30	29	28	27	26	25	24
Field	RCDTHH							
Attribute	R							
Initial Value	0x00							

bit	23	22	21	20	19	18	17	16
Field	RCDTHL							
Attribute	R							
Initial Value	0x00							

bit	15	14	13	12	11	10	9	8
Field	RCDTLH							
Attribute	R							
Initial Value	0x00							

bit	7	6	5	4	3	2	1	0
Field	RCDTLL							
Attribute	R							
Initial Value	0x00							

This register is used to store received data.
In HDMI-CEC mode, the received data will be stored in the RCDTHH.
In remote control mode, every 8 bits reception will be stored from RCDTHH.
If a counter overflow interrupt is generated, the bits already received by then will be stored from the MSB.
If EN bit of the RCCR register is "0", unknown values will be read from this register.
If signals over 4 bytes are received, the excess will be ignored and not be reflected to the register.

5.8. Clock Division Setting Register (RCCKD)

Configuration of the Clock Division Setting Register (RCCKD) bits is as follows.

bit	15	14	13	12	11	10	9	8
Field	Reserved			CKSEL	CKDIV			
Attribute				R/W	R/W			
Initial Value				0	0000			

bit	7	6	5	4	3	2	1	0
Field	CKDIV							
Attribute	R/W							
Initial Value	0x00							

[bit15:13] Reserved: Reserved bits

"0" is always read.

Set "0" for write.

[bit12] CKSEL: Operating clock selection bit

Value	Description
0	Clock divided from peripheral clock (PCLK) is selected.
1	Sub-clock is selected.

[bit11:0] CKDIV: Operating clock division setting bits

Division ratio becomes CKDIV + 1.

1 division (no division) through 4096 division can be set (no division if CKSEL=1).

5.9. Repeat Code Interrupt Control Register (RCRC)

This register controls repeat code interrupts.

bit	7	6	5	4	3	2	1	0
Field	Reserved			RCIE	Reserved			RC
Attribute				R/W				R/W
Initial Value				0				0

[bit7:5] Reserved: Reserved bits

"0" is always read.
Set "0" for write.

[bit4] RCIE: Repeat Code Interrupt enable bit

Value	Description
0	Interrupt disabled
1	Interrupt enabled

[bit3:1] Reserved: Reserved bits

"0" is always read.
Set "0" for write.

[bit0] RC: Repeat code detection flag bit

Process	Description
"0" is read	Repeat code not detected
"1" is read	Repeat code detected
"0" is written	This flag will be cleared
"1" is written	No effect

<Note>

Repeat code is detected only in NEC/Association for Electric Home Appliances mode.

5.10. Repeat Code Detection Width Setting Register (RCRHW)

This register is used to set the detection width used for determining a repeat code.

bit	7	0
Field	RCRHW	
Attribute	R/W	
Initial Value	0x00	

[bit7:0] RCRHW: Repeat code detection width setting bits

These bits are used to set the detection width for a repeat code.

If a signal width with RCRHW < "H" width < RCSHW is received while waiting for a start bit or a repeat code, it will be detected as a repeat code.

A value to be set to this register must be RCRHW < RCSHW.

<Note>

Repeat code is detected only in NEC/Association for Electric Home Appliances mode.

5.11. Data Bit Width Violation Interrupt Control Register (RCLE)

This register controls maximum/minimum data bit width violation.

bit	7	6	5	4	3	2	1	0
Field	LELIE	LESIE	LELE	LESE	EPE	Reserved	LEL	LES
Attribute	R/W	R/W	R/W	R/W	R/W		R/W	R/W
Initial Value	0	0	0	0	0		0	0

[bit7] LELIE: Maximum data bit width violation interrupt enable bit

Value	Description
0	Interrupt disabled
1	Interrupt enabled

[bit6] LESIE: Minimum data bit width violation interrupt enable bit

Value	Description
0	Interrupt disabled
1	Interrupt enabled

[bit5] LELE: Maximum data bit width violation detection enable bit

Value	Description
0	Maximum data bit width violation detection disabled
1	Maximum data bit width violation detection enabled

[bit4] LESE: Minimum data bit width violation detection enable bit

Value	Description
0	Minimum data bit width violation detection disabled
1	Minimum data bit width violation detection enabled

[bit3] EPE: Error pulse output enable bit

Value	Description
0	Output disabled
1	Output enabled

If a minimum data bit width violation is detected when EPE="1", "L" pulses at 116 through 120 cycles will be output.

[bit2] Reserved: Reserved bit

"0" is always read.
Set "0" for write.

[bit1] LEL: Maximum data bit width violation detection flag bit

Process	Description
"0" is read	Maximum data bit width violation has not been detected
"1" is read	Maximum data bit width violation has been detected
"0" is written	This flag will be cleared
"1" is written	No effect on operation

[bit0] LES: Minimum data bit width violation detection flag bit

Process	Description
"0" is read	Minimum data bit width violation has not been detected
"1" is read	Minimum data bit width violation has been detected
"0" is written	This flag will be cleared
"1" is written	No effect on operation

<Note>

Maximum/minimum data bit width violation is detected only in HDMI-CEC mode.

5.12. Maximum Data Bit Width Setting Register (RCLELW)

This register is used to set a maximum data bit width.

bit	7	0
Field	RCLELW	
Attribute	R/W	
Initial Value	0x00	

[bit7:0] RCLELW: Maximum data bit width setting bits

These bits are used to set a maximum data bit width.

If a data bit with a width more than RCLELW is received, it will be detected as a maximum data bit width violation.

<Note>

Maximum data bit width violation is detected only in HDMI-CEC mode.

5.13. Minimum Data Bit Width Setting Register (RCLESW)

This register is used to set a minimum data bit width.

bit	7 … 0
Field	RCLESW
Attribute	R/W
Initial Value	0x00

[bit7:0] RCLESW: Minimum data bit width setting bits

These bits are used to set a minimum data bit width.

If a data bit with a width less than RCLESW is received, it will be detected as a minimum data bit width violation.

<Note>

Minimum data bit width violation is detected only in HDMI-CEC mode.

CHAPTER 6-3: CEC Transmission

Functions and operations of CEC (Consumer Electronics Control) transmission are as follows.

1. Overview of CEC Transmission
2. Block Diagram of CEC Transmitting Circuit
3. CEC Transmission Interrupts
4. CEC Transmission Registers
5. CEC Transmission Operations
6. CEC Transmission Register Set

CODE: FIP007-E01-01

1. Overview of CEC Transmission

CEC signals standardized by HDMI (High Definition Multimedia Interface) are transmitted. The outline of transmission specification is as follows.

■ Automatic header transmission

Signal free is recognized to automatically transmit a header block.

■ Bus error detection

Arbitration lost is recognized to generate a status interrupt.

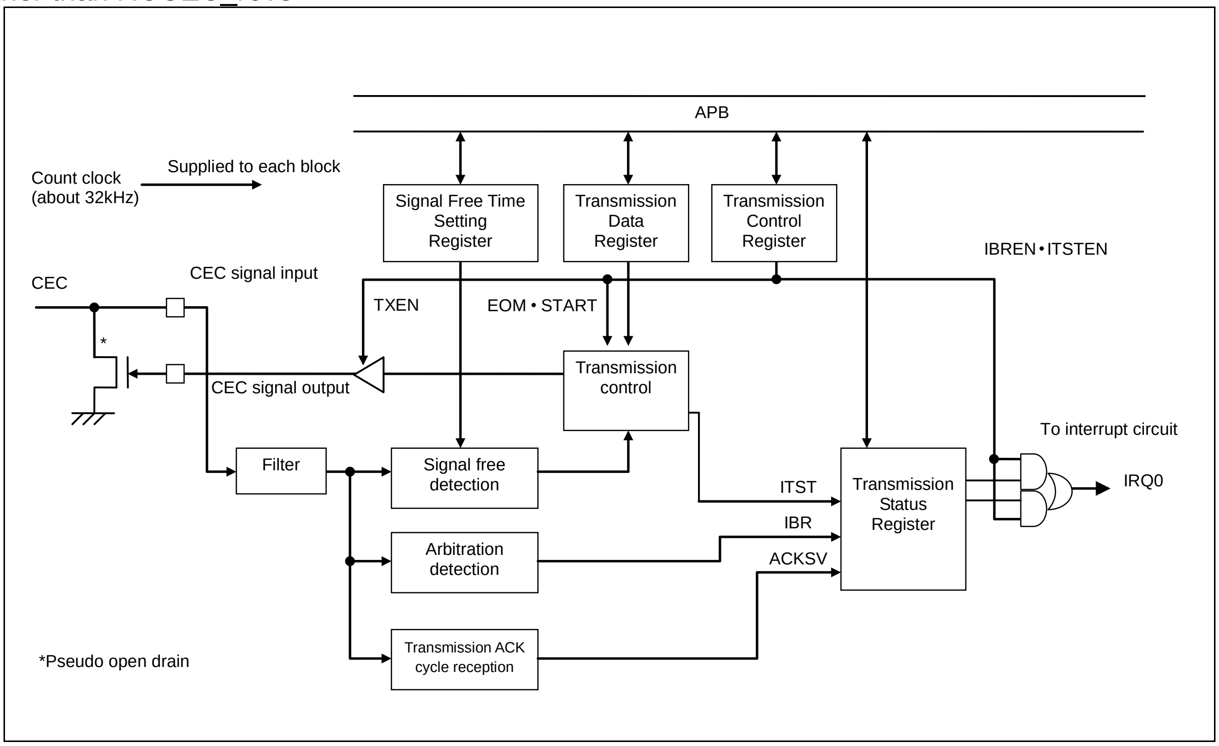
■ Data transmission

· Setting 1 byte data automatically generate START, EOM and ACK to output CEC transmission.
· After 1 block (1 byte data, EMO and ACK) is transmitted, a transmission status interrupt is generated.

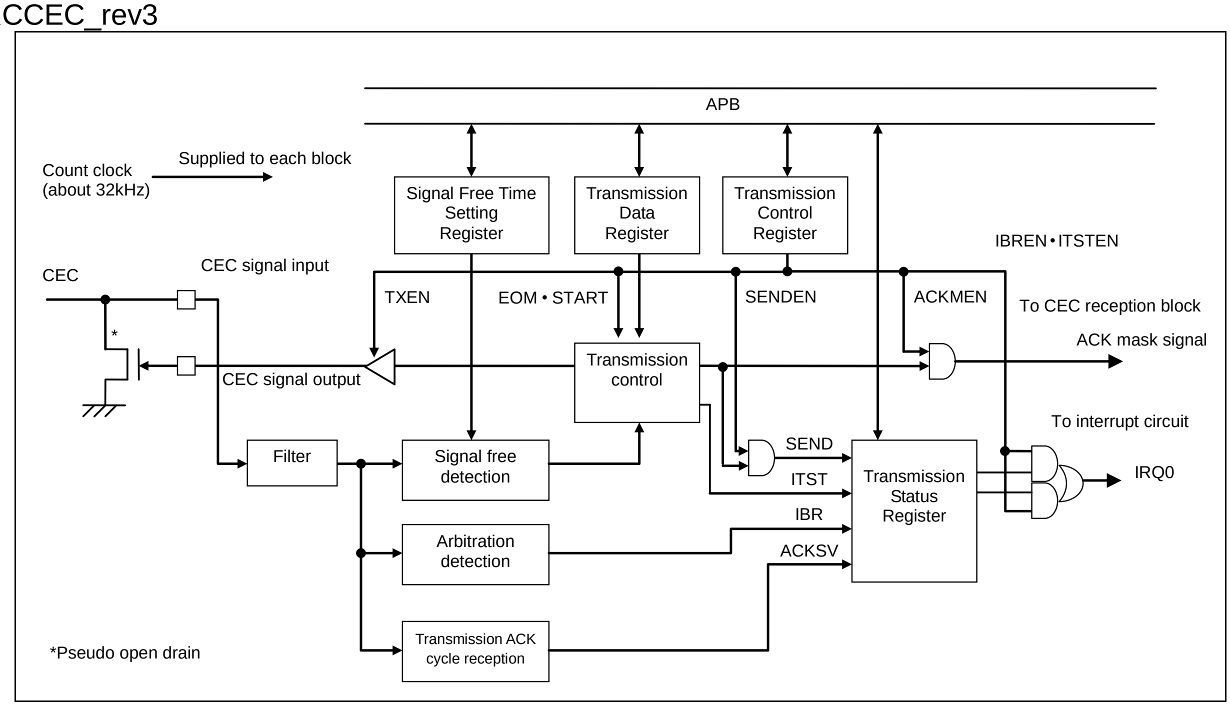
2. Block Diagram of CEC Transmitting Circuit

Figure 2-1 shows the block diagram of CEC transmitting circuit.

Figure 2-1 Block Diagram of CEC Transmitting Circuit
- Other than RCCEC_rev3



- RCCEC_rev3



3. CEC Transmission Interrupts

A table summarizing interrupt request flags, interrupt enable bits and interrupt factors for CEC transmission is shown as follows.

■ Interrupt control bits and interrupt factors

Interrupt control bits and interrupt factors are shown in Table 3-1.

Table 3-1 Interrupt Control Bits and Interrupt Factors in each Mode

Transmission status (TXSTS)	Transmission control (TXCTRL)	Interrupt factor	Interrupt factor output signal
Interrupt request flag bit	Interrupt request enable bit		
ITST: bit4	ITSTEN: bit4	Transmission status detected	IRQ0
IBR: bit5	IBREN: bit5	Bus error detected	

4. CEC Transmission Registers

CEC transmission registers are as follows.

■ CEC Transmission registers

Table 4-1 CEC Transmission Registers

Abbreviated Register Name	Register Name	Reference
TXCTRL	Transmission Control Register	6.1
TXDATA	Transmission Data Register	6.2
TXSTS	Transmission Status Register	6.3
SFREE	Signal Free Time Setting Register	6.4

5. CEC Transmission Operations

Operations of CEC transmission are explained as follows.

5.1. CEC Transmission Operations

Basic operations for transmission are explained as follows.

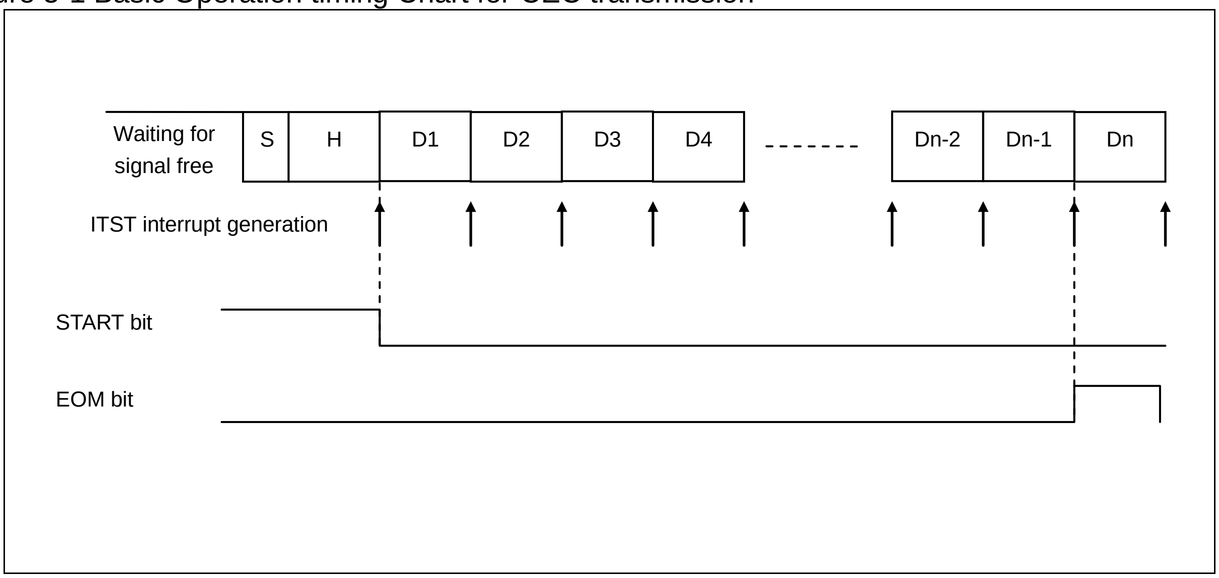
■ Basic operations

Basic operations are as follows:

- First set count clock for CEC from reception side.
- Next make various transmission setups and write transmitting data to TXDATA register to wait until signal free is detected. When signal free is detected, a start bit will automatically be transmitted.
- After the start bit is transmitted, 1 byte data set in the TXDATA register, data set in the EOM setting bits and ACK bit are automatically transmitted.
- As ITST bit interrupt of TXSTS register will be generated after the ACK bit is automatically transmitted. If the ACK cycle value is correct, make various transmission setups and write transmitting data for next transmission.
- Continue the transmission with the EOM at "1" until the complete transmissions end.

The basic operation timing for CEC transmission is shown in Figure 5-1.

Figure 5-1 Basic Operation timing Chart for CEC transmission



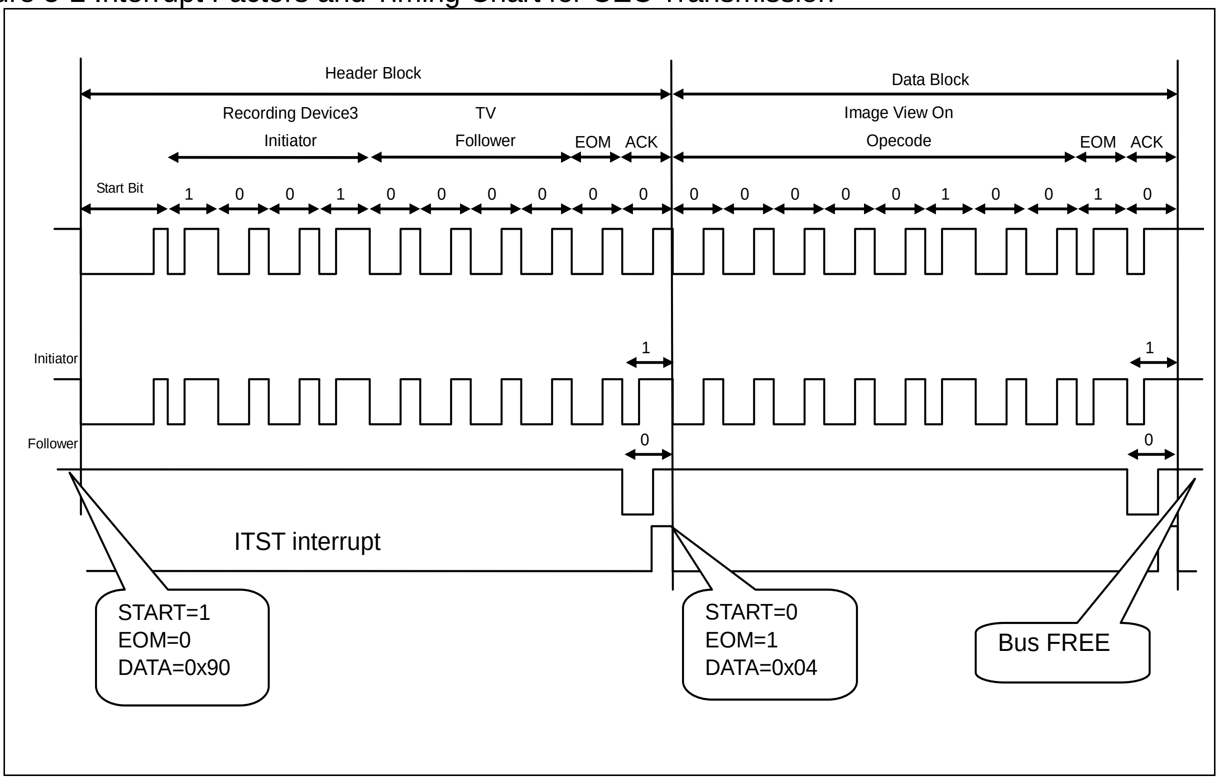
5.2. Interrupt Factors and Timing Chart

Interrupt factors and timing chart are as follows.

■ Interrupt Factors and Timing Chart

Figure 5-2 shows a transmission for a header block and a single data block in the ITST interrupt factors and timing chart.

Figure 5-2 Interrupt Factors and Timing Chart for CEC Transmission



5.3. Arbitration Lost Detection

Arbitration lost detection is as follows.

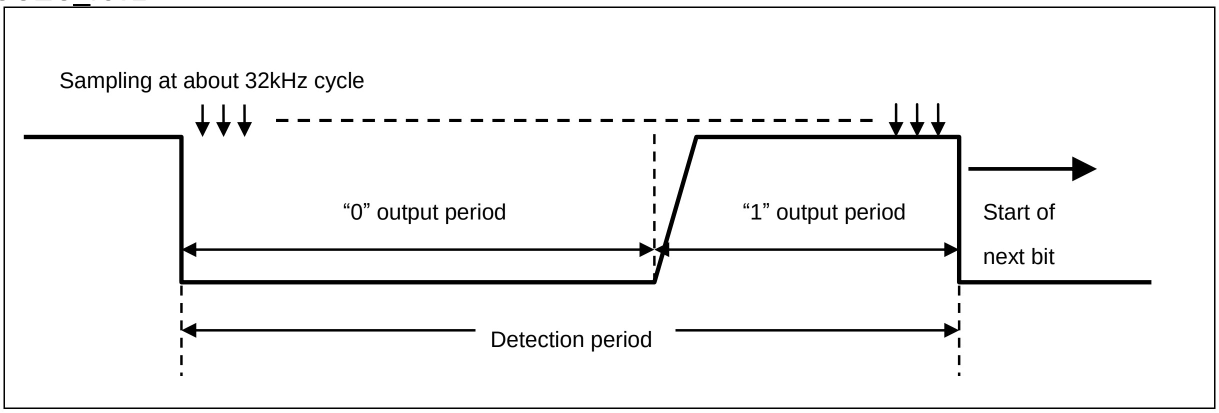
■ How to detect arbitration lost

Figure 5-3 shows how to detect arbitration lost.

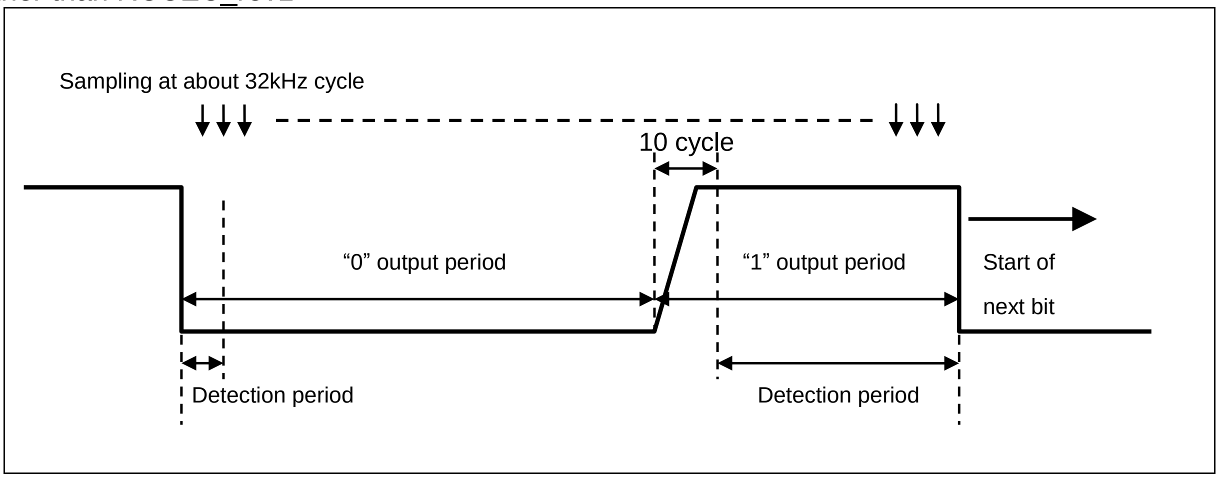
Data on the bus is sampled with about 32 kHz cycle per bit during the following detection period and compared to the transmission output. If any difference is continuously detected, an arbitration lost will be detected. If the arbitration lost is detected, IBR of the TXSTS register becomes "1".

Figure 5-3 Arbitration Lost Detection Period

- RCCEC_rev1



- Other than RCCEC_rev1

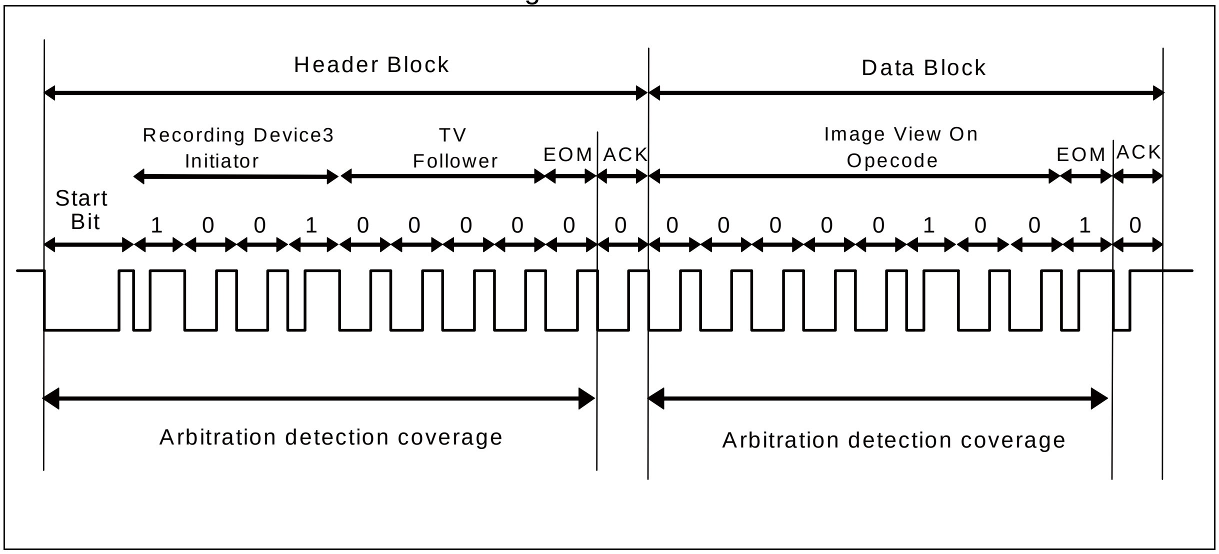


■ Detection coverage of arbitration lost

Figure 5-4 shows the detection coverage of arbitration lost.

The detection coverage becomes to the EOM during each block transfer excluding ACK cycle.

Figure 5-4 Arbitration Lost Detection Coverage



5.4. Signal Free Detection

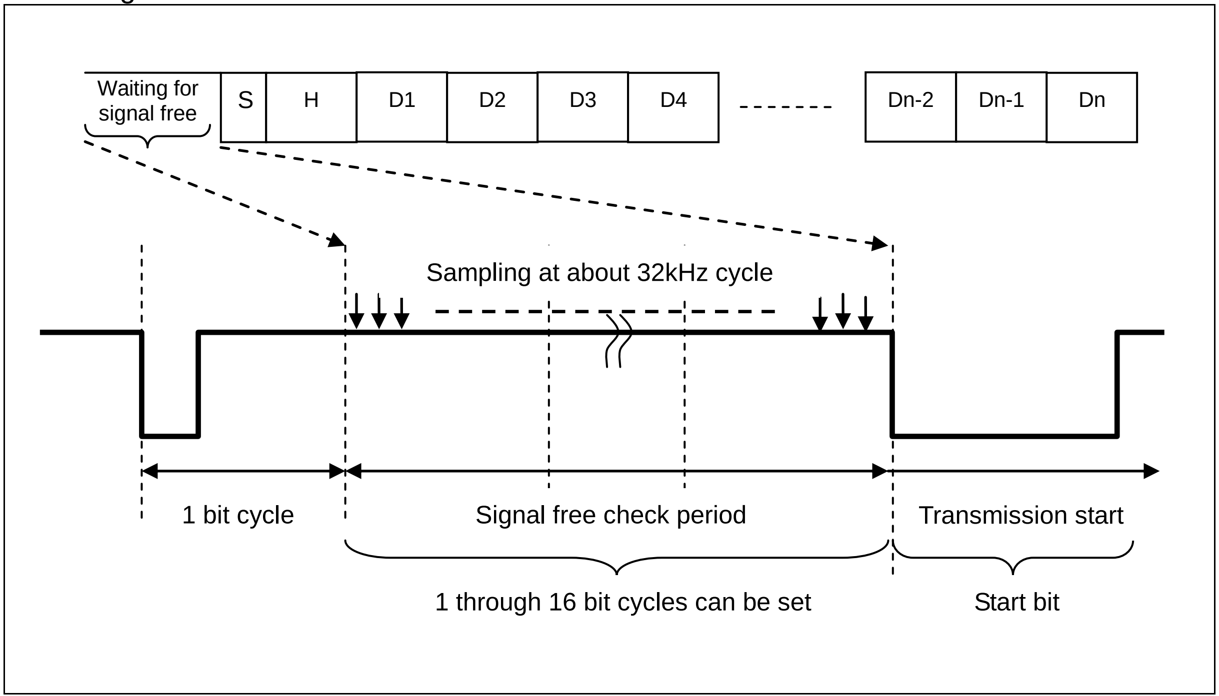
Signal free detection is as follows.

■ How to detect signal free

Figure 5-5 shows signal free detection.

If no change is found on the CEC bus during the cycles set in the SFREE register after the previous frame end, it becomes signal free detection state.

Figure 5-5 Signal Free Detection



If 5bit signal free time from last fall edge of previous frame to fall edge of start bit should be secured, set "3" to signal free setting register.

RCCEC_rev3

If signal free time should be secured 5bit after other device transmission except this device address, it is available. If SEND bit at the start bit detection interrupt reception is "0", it is possible to determine the transmission from the other device.

5.5. Filtering

Filtering CEC signal input of transmission side is described as follows.

■ Filtering CEC signals

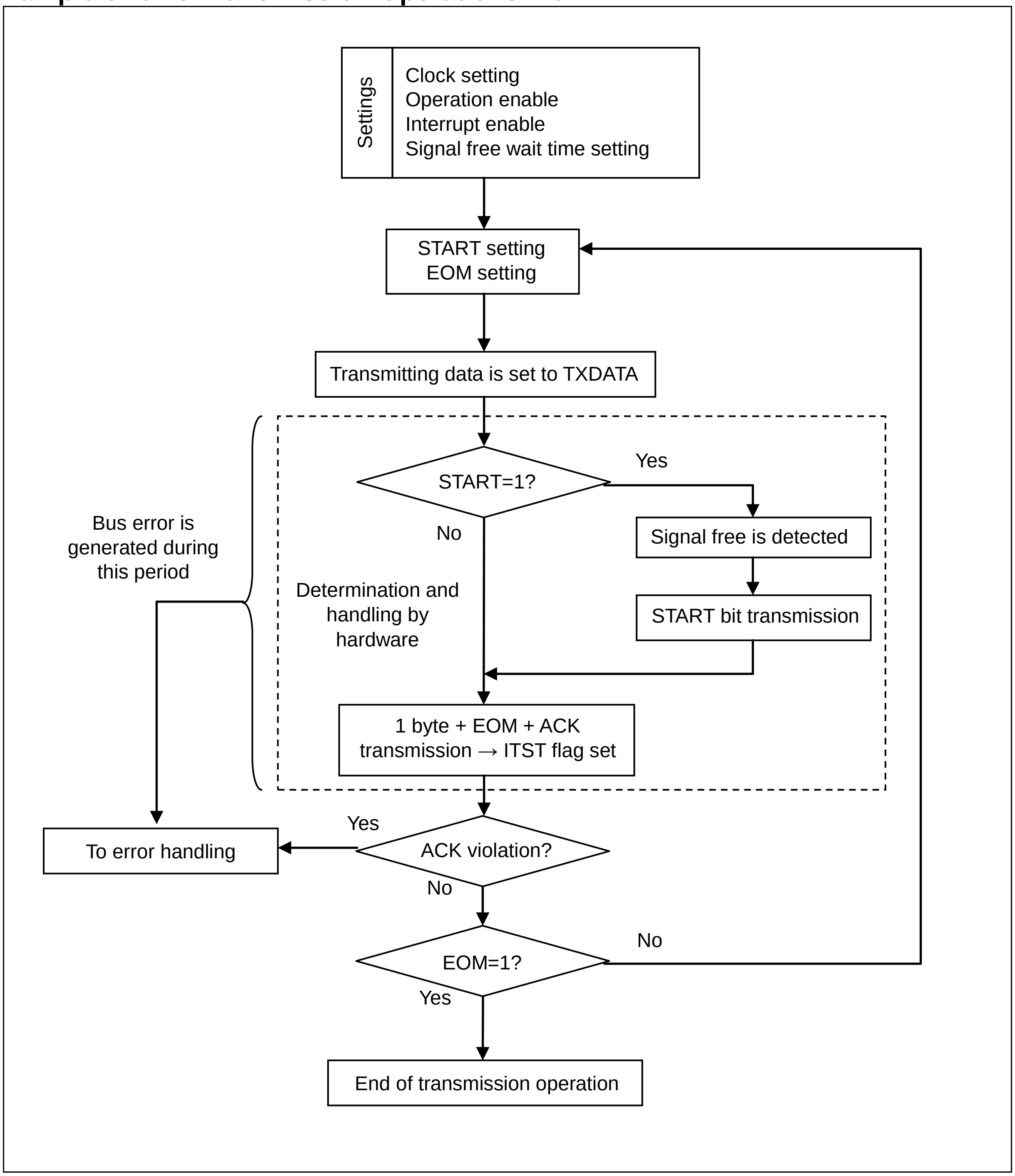
If a CEC signal input is changed within a width less than 2 count clocks, it is determined as noise and the signal will be removed.

An input changed within a width more than 2 count clocks is determined as CEC signal and passes through the filter.

5.6. CEC Transmission Operations Flow

CEC transmission operations flow is described as follows.

■ Example of CEC Transmission Operations Flow



6. CEC Transmission Register Set

All of CEC transmission registers is explained as follows.

6.1. Transmission Control Register (TXCTRL)

Transmission Control Register (TXCTRL) controls CEC transmission.

- Other than RCCEC_rev3

bit	7	6	5	4	3	2	1	0
Field	Reserved		IBREN	ITSTEN	EOM	START	Reserved	TXEN
Attribute	R/W		R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	00		0	0	0	0	0	0

- RCCEC_rev3

bit	7	6	5	4	3	2	1	0
Field	SENDEN	ACKMEN	IBREN	ITSTEN	EOM	START	Reserved	TXEN
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

- Other than RCCEC_rev3

[bit7:6] Reserved: Reserved bits

"0" is read.

Set "0" to these bits for write.

- RCCEC_rev3

[bit7] SENDEN: Sending flag enable bit

This bit controls operation of SEND bit in Transmission status register (TXSTS).

Value	Description
0	Disable SEND bit operation
1	Enable SEND bit operation

[bit6] ACKMEN: ACK mask enable bit

This bit controls ACK mask.

When the ACKMEN bit is "1" and sending, ACK output is masked.

Value	Description
0	Disable ACK mask
1	Enable ACK mask

[bit5] IBREN: Bus error detection interrupt enable bit

- This bit controls the interrupt request from bit5, the IBR bit in the TXSTS register.
- When the IBREN bit is enabled and bit5, the IBR bit in the TXSTS register is set, an interrupt request will be generated to the CPU.

Value	Description
0	Disables interrupt request
1	Enables interrupt request

[bit4] ITSTEN: transmission status interrupt enable bit

- This bit controls the interrupt request from bit4, the ITST bit in the TXSTS register.
- When the ITSTEN bit is enabled and bit4, the ITST bit in the TXSTS register is set, an interrupt request will be generated to the CPU.

Value	Description
0	Disables interrupt request
1	Enables interrupt request

[bit3] EOM: EOM setting bit

- This controls EOM transmission bit.
- Combination with the START bit will select block transmission.

Value	Description
0	Outputs EOM0
1	Outputs EOM1

[bit2] START: START setting bit

- This bit sets a header block transmission which adds the START bit to transmitting data.
- Combination with the EOM bit will select block transmission.

Value	Description
0	START bit transmission invalid
1	START bit transmission valid

EOM and START setups make CEC transmission to the following block transmission.

EOM bit	START=1	START=0
0	Header block transmission (beginning of frame)	Data block (with subsequent block)
1	Header block transmission (Polling Message)	Final data block (end of frame)

[bit1] Reserved: Reserved bit

"0" is read.

Set "0" to this bit for write.

[bit0] TXEN: Transmission operation enable bit

- This bit controls CEC transmission operations.
- When the TXEN bit it is changed to disable, automatic clearing for each bit of the status register will occur.

Value	Description
0	CEC transmission operation disabled
1	CEC transmission operation enabled

<Note>

When "0" is set to the TXEN bit, outputs will immediately be stopped. Incorrect wave form may be output for the CEC signal at that time.

6.2. Transmission Data Register (TXDATA)

Transmission Data Register (TXDATA) is used to set up transmission data.

bit	7	0
Field	TXDATA[7:0]	
Attribute	R/W	
Initial Value	0x00	

When a value is set to the TXDATA register, one of the following CEC transmissions will be started depending on the condition.

If the following conditions are met, a header block transmission will automatically be started.

- TXEN=1.
- START=1.
- IDLE is detected on the CEC bus during a period set in the SFREE register.

<Note>

When you set a value to the TXDATA register, if IDLE for a period set in the SFREE register has been detected, a header block transmission will be started immediately after setup to the TXDATA register.

If the following conditions are met, a data block transmission will immediately be started.

- TXEN=1.
- START=0.

6.3. Transmission Status Register (TXSTS)

Transmission Status Register (TXSTS) is used to indicate transmission statuses.

- Other than RCCEC_rev3

bit	7	6	5	4	3	2	1	0
Field	Reserved		IBR	ITST	Reserved			ACKSV
Attribute	R/W		R/W	R/W	R/W			R
Initial Value	00		0	0	000			0

- RCCEC_rev3

bit	7	6	5	4	3	2	1	0
Field	SEND	Reserved	IBR	ITST	Reserved			ACKSV
Attribute	R	R/W	R/W	R/W	R/W			R
Initial Value	0	0	0	0	000			0

- Other than RCCEC_rev3

[bit7:6] Reserved: Reserved bits

"0" is read.
Set "0" to these bits for write.

- RCCEC_rev3

[bit7] SEND: Sending flag bit

This bit indicates that CEC transmission is sending.
If SENDEN bit is "1" and CEC transmission is sending from start of start bit to end of ACK bit, this bit is "1".
When SENDEN bit is "0", this bit is "0".
It is invalid to set this bit.

Value	Description
0	Not sending or SEND bit is "0"
1	Sending (When SENDEN is "1")

[bit6] Reserved: Reserved bits

"0" is read.
Set "0" to these bits for write.

[bit5] IBR: Bus error detection interrupt request bit

- When arbitration lost is detected, the IBR bit is set to "1".
- The IBR bit is cleared by writing "0".
- Writing "1" to the IBR bit does not effect to the bit value.
- Read value by read-modify-write operation becomes "1" independent of the bit value.

Value	Description
0	Clears interrupt factor
1	Detects interrupt factor

<Notes>

- When "1" is automatically set to the IBR bit, if it is cleared at the same time by writing "0", the clearing will be ignored and "1" will be set.
- Be sure to write "0" while the IBR bit is "1". It may be cleared not knowing it will be automatically set to "1".
- If a line error signal is detected, the IBR bit will also be set to "1" as a bus error is detected.

[bit4] ITST: Transmission status interrupt request bit

- When communication of a status bit at 10 bit in each block transfer is completed, the ITST bit will be set to "1".
- The ITST bit is cleared by writing "0".
- Writing "1" to the ITST bit does not effect the bit value.
- Read value by read-modify-write operation becomes "1" independent of the bit value.

Value	Description
0	Clears interrupt factor
1	Detects interrupt factor

<Notes>

- When "1" is attempted to automatically set to the ITST bit, if it is cleared at the same time by writing "0", the clearing will be ignored and "1" will be set.
- Be sure to write "0" while the ITST bit is "1". It may be cleared not knowing it will be automatically set to "1".

[bit3:1] Reserved: Reserved bits

"0" is read.
Set "0" to these bits for write.

[bit0] ACKSV: ACK cycle value bit

- This bit indicates received data values in ACK cycle at 10 bit in each block transfer.
- This bit is updated when the ITST bit is changed from "0" to "1".
- Writing "1" to the ACKSV bit does not effect to the bit value.

Value	Description
0	"0" is received in ACK cycle
1	"1" is received in ACK cycle

6.4. Signal Free Time Setting Register (SFREE)

Signal Free Time Setting Register (SFREE) is used to set a signal free time checked before starting transmission.

bit	7	6	5	4	3	2	1	0
Field	Reserved				SFREE[3:0]			
Attribute	R/W				R/W			
Initial Value	0000				0000			

[bit7:4] Reserved: Reserved bits

"0" is read.

Set "0" to these bits for write.

[bit3:0] SFREE[3:0]: Signal free time setting bits

- These bits are used to set a time to check free state on the CEC bus before starting transmission.
- After no communication for bit cycle set on the CEC bus is found, transmission operation will be started.

Value	Description
0000	(Set value + 1) cycle Ex1) 0000: 1bit cycle　　Ex2) 0111:8bit cycle Ex3) 1000: 9bit cycle　　Ex3) 1111:16bit cycle
0001	
...	
1110	
1111	

Appendixes

This chapter shows the register map, list of notes, limitations and product type list.

CODE: 9BFAPPENDIXES-E03.0

A. Register Map

This chapter shows the register map.

CODE: 9BFREGMAP-E06.0

1. Register Map

Register map is shown on the table every module/function.

[How to read the each table]

Module/function name and its base address

Clock/Reset Base_Address : 0x4001_0000

Base_Address	Register			
+ Address	+3	+2	+1	+0
0x000	-	-	-	SCM_CTL[W] 00000-0-
0x004	-	-	-	SCM_STR[W] 00000-0-
0x008	STB_CTL[W] 00000000 00000000 -------- ---0--00			
0x00C	-	-	RST_STR[W] -------0 00000-01	

- : Reserved area
* : Test register area

Initial value after reset

"1"	:	Initial value is "1"
"0"	:	Initial value is "0"
"X"	:	Initial value is undefined
" - "	:	Reserved bit

Register name

Access unit
(B : byte, H : half word, W : word)

Rightmost register address (For word-length access, the "+0" column of the register is the LSB of the data.)

<Notes>

- The register table is represented in the little-endian.
- When performing a data access, the addresses should be as below according to the access size.
  - Word access: Address should be multiples of 4 (least significant 2 bits should be "0x00")
  - Half word access: Address should be multiples of 2 (least significant bit should be "0x0")
  - Byte access: -
- Do not access the test register area.
- Do not access the area that is not written in the register table.
- When the register is accessed by larger unit than register size, for the reserved area to access at the same time, the read value is undefined, and writing is invalid.
- The respective meanings of *1 to *8 in the register map are as follows:

- *1: Initial value for TYPE0.
- *2: Initial value for TYPE1 to TYPE7.
- *3: Initial value for TYPE0, TYPE3, and TYPE7.
- *4: Initial value for TYPE1, TYPE2, TYPE4, and TYPE5.
- *5: Initial value for TYPE6, TYPE8, and TYPE9.
- *6: Initial value for TYPE3 and TYPE7.
- *7: Initial value for TYPE6 and TYPE8.
- *8: Initial value for TYPE9 to TYPE12.

1.1. Flash I/F

Base_Address : 0x4000_0000

■ **Products other than TYPE6, and TYPE8 to TYPE12**

Base_Address + Address	Register +3	Register +2	Register +1	Register +0
0x000	FASZR[B,H,W]			
0x004	FRWTR[B,H,W]			
0x008	FSTR[B,H,W]			
0x00C	*			
0x010	FSYNDN[B,H,W]			
0x014	FBFCR[B,H,W]			
0x018 - 0x0FC	-	-	-	-
0x100	CRTRMM[B,H,W]			
0x104 - 0xFFC	-	-	-	-

■ **TYPE6, and TYPE8 to TYPE11 products**

Base_Address + Address	Register +3	Register +2	Register +1	Register +0
0x000	-	-	-	-
0x004	FRWTR[B,H,W]			
0x008	FSTR[B,H,W]			
0x00C - 0x01C	-	-	-	-
0x020	FICR[B,H,W]			
0x024	FISR[B,H,W]			
0x028	FICLR[B,H,W]			
0x02C - 0x0FC	-	-	-	-
0x100	CRTRMM[B,H,W]			
0x104 - 0xFFC	-	-	-	-

■ TYPE12 products

Base_Address + Address	Register +3	Register +2	Register +1	Register +0
0x000	-	-	-	-
0x004	FRWTR[B,H,W]			
0x008	FSTR[B,H,W]			
0x00C - 0x01C	-	-	-	-
0x020	FICR[B,H,W]			
0x024	FISR[B,H,W]			
0x028	FICLR[B,H,W]			
0x02C - 0x084	-	-	-	-
0x088	FSTR1[B,H,W]			
0x08C - 0x0FC	-	-	-	-
0x100	CRTRMM[B,H,W]			
0x104 - 0xFFC	-	-	-	-

Note:

For details of Flash I/F registers, see "FLASH PROGRAMMING MANUAL" of the product used.

1.2. Unique ID

Base_Address : 0x4000_0200

Base_Address + Address	Register +3	Register +2	Register +1	Register +0
0x000	UIDR0 [W] XXXXXXXX XXXXXXXX XXXXXXXX XXXX----			
0x004	UIDR1 [W] -------- -------- ---XXXXX XXXXXXXX			
0x008 - 0xDFC	-	-	-	-

1.3. Clock/Reset

Base_Address : 0x4001_0000

Base_Address + Address	Register +3	+2	+1	+0
0x000	-	-	-	SCM_CTL[W] 00000-0-
0x004	-	-	-	SCM_STR[W] 00000-0-
0x008	STB_CTL[W] 00000000 00000000 -------- ---0-000			
0x00C	-	-	RST_STR[W] -------0 00000-01	
0x010	-	-	-	BSC_PSR[W] -----000
0x014	-	-	-	APBC0_PSR[W] ------00
0x018	-	-	-	APBC1_PSR[W] 1--0--00
0x01C	-	-	-	APBC2_PSR[W] 1--0--00
0x020	-	-	-	SWC_PSR[W] X-----00
0x024 - 0x027	-	-	-	-
0x028	-	-	-	TTC_PSR[W] ------00
0x02C - 0x02F	-	-	-	-
0x030	-	-	-	CSW_TMR[W] -0000000
0x034	-	-	-	PSW_TMR[W] ---0-000
0x038	-	-	-	PLL_CTL1[W] 00000000
0x03C	-	-	-	PLL_CTL2[W] ---00000
0x040	-	-	CSV_CTL[W] -111--00 ------11	

A. Register Map

Base_Address + Address	Register			
	+3	+2	+1	+0
0x044	-	-	-	CSV_STR[W] ------00
0x048	-	-	FCSWH_CTL[W] 11111111 11111111	
0x04C	-	-	FCSWL_CTL[W] 00000000 00000000	
0x050	-	-	FCSWD_CTL[W] 00000000 00000000	
0x054	-	-	-	DBWDT_CTL[W] 0-0-----
0x058	-	-	-	*
0x05C - 0x05F	-	-	-	-
0x060	-	-	-	INT_ENR[W] --0--000
0x064	-	-	-	INT_STR[W] --0--000
0x068	-	-	-	INT_CLR[W] --0--000
0x06C - 0xFFC	-	-	-	-

1.4. HW WDT

Base_Address : 0x4001_1000

Base_Address + Address	Register			
	+3	+2	+1	+0
0x000	WDG_LDR[W] 00000000 00000000 11111111 11111111			
0x004	WDG_VLR[W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX			
0x008	WDG_CTL[W]			
	-	-	-	------11
0x00C	WDG_ICL[W]			
	-	-	-	XXXXXXXX
0x010	WDG_RIS[W]			
	-	-	-	-------0
0x014 - 0xBFC	-	-	-	-
0xC00	WDG_LCK[W] 00000000 00000000 00000000 00000001			
0xC04 - 0xFFC	-	-	-	-

1.5. SW WDT

Base_Address : 0x4001_2000

Base_Address + Address	Register			
	+3	+2	+1	+0
0x000	WdogLoad[W] 11111111 11111111 11111111 11111111			
0x004	WdogValue[W] 11111111 11111111 11111111 11111111			
0x008	WdogControl[W]			
	-	-	-	------00
0x00C	WdogIntClr[W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX			
0x010	WdogRIS[W]			
	-	-	-	-------0
0x014 - 0xBFC	-	-	-	-
0xC00	WdogLock[W] 00000000 00000000 00000000 00000000			
0xC04 - 0xFFC	-	-	-	-

1.6. Dual_Timer

Base_Address : 0x4001_5000

Base_Address + Address	Register +3	+2	+1	+0
0x000	Timer1Load[W] 00000000 00000000 00000000 00000000			
0x004	Timer1Value[W] 11111111 11111111 11111111 11111111			
0x008	Timer1Control[W] -------- -------- -------- 00100000			
0x00C	Timer1IntClr[W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX			
0x010	Timer1RIS[W] -------- -------- -------- -------0			
0x014	Timer1MIS[W] -------- -------- -------- -------0			
0x018	Timer1BGLoad[W] 00000000 00000000 00000000 00000000			
0x020	Timer2Load[W] 00000000 00000000 00000000 00000000			
0x024	Timer2Value[W] 11111111 11111111 11111111 11111111			
0x028	Timer2Control[W] -------- -------- -------- 00100000			
0x02C	Timer2IntClr[W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX			
0x030	Timer2RIS[W] -------- -------- -------- -------0			
0x034	Timer2MIS[W] -------- -------- -------- -------0			
0x038	Timer2BGLoad[W] 00000000 00000000 00000000 00000000			
0x040 - 0xFFC	-	-	-	-

1.7. MFT

unit0 Base_Address : 0x4002_0000
unit1 Base_Address : 0x4002_1000
unit2 Base_Address : 0x4002_2000

Base_Address	Register			
+ Address	+3	+2	+1	+0
0x000	-	-	OCCP0[H,W] 00000000 00000000	
0x004	-	-	OCCP1[H,W] 00000000 00000000	
0x008	-	-	OCCP2[H,W] 00000000 00000000	
0x00C	-	-	OCCP3[H,W] 00000000 00000000	
0x010	-	-	OCCP4[H,W] 00000000 00000000	
0x014	-	-	OCCP5[H,W] 00000000 00000000	
0x018	-	-	OCSB10[B,H,W] -110--00	OCSA10[B,H,W] 00001100
0x01C	-	-	OCSB32[B,H,W] -110--00	OCSA32[B,H,W] 00001100
0x020	-	-	OCSB54[B,H,W] -110--00	OCSA54[B,H,W] 00001100
0x024	-	-	OCSC[B,H,W] --000000	-
0x028	-	-	TCCP0[H,W] 11111111 11111111	
0x02C	-	-	TCDT0[H,W] 00000000 00000000	
0x030	-	-	TCSA0[B,H,W] 000---00 01000000	
0x034	-	-	TCSB0[B,H,W] -------- -----000	
0x038	-	-	TCCP1[H,W] 11111111 11111111	
0x03C	-	-	TCDT1[H,W] 00000000 00000000	

Base_Address + Address	Register +3	+2	+1	+0
0x040	-	-	TCSA1[B,H,W] 000---00 01000000	
0x044	-	-	TCSB1[B,H,W] -------- -----000	
0x048	-	-	TCCP2[H,W] 11111111 11111111	
0x04C	-	-	TCDT2[H,W] 00000000 00000000	
0x050	-	-	TCSA2[B,H,W] 000---00 01000000	
0x054	-	-	TCSB2[B,H,W] -------- -----000	
0x058	-	-	OCFS32[B,H,W] 00000000	OCFS10[B,H,W] 00000000
0x05C	-	-	-	OCFS54[B,H,W] 00000000
0x060	-	-	ICFS32[B,H,W] 00000000	ICFS10[B,H,W] 00000000
0x064	-	-	-	-
0x068	-	-	ICCP0[H,W] XXXXXXXX XXXXXXXX	
0x06C	-	-	ICCP1[H,W] XXXXXXXX XXXXXXXX	
0x070	-	-	ICCP2[H,W] XXXXXXXX XXXXXXXX	
0x074	-	-	ICCP3[H,W] XXXXXXXX XXXXXXXX	
0x078	-	-	ICSB10[B,H,W] ------00	ICSA10[B,H,W] 00000000
0x07C	-	-	ICSB32[B,H,W] ------00	ICSA32[B,H,W] 00000000
0x080	-	-	WFTM10[H,W] 00000000 00000000	
0x084	-	-	WFTM32[H,W] 00000000 00000000	

Base_Address + Address	Register			
	+3	+2	+1	+0
0x088	-	-	WFTM54[H,W] 00000000 00000000	
0x08C	-	-	WFSA10[H,W] ---00000 000000	
0x090	-	-	WFSA32[H,W] ---00000 000000	
0x094	-	-	WFSA54[H,W] ---00000 000000	
0x098	-	-	WFIR[H,W] 00000000 0000--00	
0x09C	-	-	NZCL[H,W] -------- ---00000	
0x0A0	-	-	ACCP0[H,W] 00000000 00000000	
0x0A4	-	-	ACCPDN0[H,W] 00000000 00000000	
0x0A8	-	-	ACCP1[H,W] 00000000 00000000	
0x0AC	-	-	ACCPDN1[H,W] 00000000 00000000	
0x0B0	-	-	ACCP2[H,W] 00000000 00000000	
0x0B4	-	-	ACCPDN2[H,W] 00000000 00000000	
0x0B8	-	-	-	ACSB[B,H,W] -000-111
0x0BC	-	-	ACSA[B,H,W] --000000 --000000	
0x0C0	-	-	ATSA[H,W] --000000 --000000	
0x0C4 - 0x0FC	-	-	-	-

1.8. PPG

Base_Address : 0x4002_4000

Base_Address	Register			
+ Address	+3	+2	+1	+0
0x000	-	-	TTCR0 [B,H,W] 11110000	-
0x004	-	-	-	*
0x008	-	-	COMP0 [B,H,W] 00000000	-
0x00C	-	-	-	COMP2 [B,H,W] 00000000
0x010	-	-	COMP4 [B,H,W] 00000000	-
0x014	-	-	-	COMP6 [B,H,W] 00000000
0x018 - 0x01C	-	-	-	-
0x020	-	-	TTCR1 [B,H,W] 11110000	-
0x024	-	-	-	*
0x028	-	-	COMP1 [B,H,W] 00000000	-
0x02C	-	-	-	COMP3 [B,H,W] 00000000
0x030	-	-	COMP5 [B,H,W] 00000000	-
0x034	-	-	-	COMP7 [B,H,W] 00000000
0x038 - 0x03C	-	-	-	-
0x040	-	-	TTCR2 [B,H,W] 11110000	-
0x044	-	-	-	*
0x048	-	-	COMP8 [B,H,W] 00000000	-
0x04C	-	-	-	COMP10 [B,H,W] 00000000

Base_Address + Address	Register			
	+3	+2	+1	+0
0x050	-	-	COMP12 [B,H,W] 00000000	-
0x054	-	-	-	COMP14 [B,H,W] 00000000
0x58 - 0x0FC	-	-	-	-
0x100	-	-	TRG0 [B,H,W] 00000000 00000000	
0x104	-	-	REVC0 [B,H,W] 00000000 00000000	
0x108 - 0x13C	-	-	-	-
0x140	-	-	TRG1 [B,H,W] -------- 00000000	
0x144	-	-	REVC1 [B,H,W] -------- 00000000	
0x148 - 0x1FC	-	-	-	-
0x200	-	-	PPGC0 [B,H,W] 00000000	PPGC1 [B,H,W] 00000000
0x204	-	-	PPGC2 [B,H,W] 00000000	PPGC3 [B,H,W] 00000000
0x208	-	-	PRLH0 [B,H,W] XXXXXXXX	PRLL0 [B,H,W] XXXXXXXX
0x20C	-	-	PRLH1 [B,H,W] XXXXXXXX	PRLL1 [B,H,W] XXXXXXXX
0x210	-	-	PRLH2 [B,H,W] XXXXXXXX	PRLL2 [B,H,W] XXXXXXXX
0x214	-	-	PRLH3 [B,H,W] XXXXXXXX	PRLL3 [B,H,W] XXXXXXXX
0x218	-	-	-	GATEC0 [B,H,W] --00---00
0x21C - 0x23C	-	-	-	-
0x240	-	-	PPGC4 [B,H,W] 00000000	PPGC5 [B,H,W] 00000000

Base_Address + Address	Register			
	+3	+2	+1	+0
0x244	-	-	PPGC6 [B,H,W] 00000000	PPGC7 [B,H,W] 00000000
0x248	-	-	PRLH4 [B,H,W] XXXXXXXX	PRLL4 [B.H.W] XXXXXXXX
0x24C	-	-	PRLH5 [B,H,W] XXXXXXXX	PRLL5 [B,H,W] XXXXXXXX
0x250	-	-	PRLH6 [B,H,W] XXXXXXXX	PRLL6 [B,H,W] XXXXXXXX
0x254	-	-	PRLH7 [B,H,W] XXXXXXXX	PRLL7 [B,H,W] XXXXXXXX
0x258	-	-	-	GATEC4 [B,H,W] --00--00
0x25C - 0x27C	-	-	-	-
0x280	-	-	PPGC8 [B,H,W] 00000000	PPGC9 [B,H,W] 00000000
0x284	-	-	PPGC10 [B,H,W] 00000000	PPGC11 [B,H,W] 00000000
0x288	-	-	PRLH8 [B,H,W] XXXXXXXX	PRLL8 [B,H,W] XXXXXXXX
0x28C	-	-	PRLH9 [B,H,W] XXXXXXXX	PRLL9 [B,H,W] XXXXXXXX
0x290	-	-	PRLH10 [B,H,W] XXXXXXXX	PRLL10 [B,H,W] XXXXXXXX
0x294	-	-	PRLH11 [B,H,W] XXXXXXXX	PRLL11 [B,H,W] XXXXXXXX
0x298	-	-	-	GATEC8 [B,H,W] --00--00
0x29C - 0x2BC	-	-	-	-
0x2C0	-	-	PPGC12 [B,H,W] 00000000	PPGC13 [B,H,W] 00000000
0x2C4	-	-	PPGC14 [B,H,W] 00000000	PPGC15 [B,H,W] 00000000

Base_Address + Address	Register			
	+3	+2	+1	+0
0x2C8	-	-	PRLH12 [B,H,W] XXXXXXXX	PRLL12 [B,H,W] XXXXXXXX
0x2CC	-	-	PRLH13 [B,H,W] XXXXXXXX	PRLL13 [B,H,W] XXXXXXXX
0x2D0	-	-	PRLH14 [B,H,W] XXXXXXXX	PRLL14 [B,H,W] XXXXXXXX
0x2D4	-	-	PRLH15 [B,H,W] XXXXXXXX	PRLL15 [B,H,W] XXXXXXXX
0x2D8	-	-	-	GATEC12 [B,H,W] --00--00
0x2DC - 0x2FC	-	-	-	-
0x300	-	-	PPGC16 [B,H,W] 00000000	PPGC17 [B,H,W] 00000000
0x304	-	-	PPGC18 [B,H,W] 00000000	PPGC19 [B,H,W] 00000000
0x308	-	-	PRLH16 [B,H,W] XXXXXXXX	PRLL16 [B,H,W] XXXXXXXX
0x30C	-	-	PRLH17 [B,H,W] XXXXXXXX	PRLL17 [B,H,W] XXXXXXXX
0x310	-	-	PRLH18 [B,H,W] XXXXXXXX	PRLL18 [B,H,W] XXXXXXXX
0x314	-	-	PRLH19 [B,H,W] XXXXXXXX	PRLL19 [B,H,W] XXXXXXXX
0x318	-	-	-	GATEC16[B,H,W] --00--00
0x31C - 0x33C	-	-	-	-
0x340	-	-	PPGC20 [B,H,W] 00000000	PPGC21 [B,H,W] 00000000
0x344	-	-	PPGC22 [B,H,W] 00000000	PPGC23 [B,H,W] 00000000
0x348	-	-	PRLH20 [B,H,W] XXXXXXXX	PRLL20 [B,H,W] XXXXXXXX

Base_Address + Address	Register			
	+3	+2	+1	+0
0x34C	-	-	PRLH21 [B,H,W] XXXXXXXX	PRLL21 [B,H,W] XXXXXXXX
0x350	-	-	PRLH22 [B,H,W] XXXXXXXX	PRLL22 [B,H,W] XXXXXXXX
0x354	-	-	PRLH23 [B,H,W] XXXXXXXX	PRLL23 [B,H,W] XXXXXXXX
0x358	-	-	-	GATEC20 [B,H,W] --00--00
0x35C - 0x37C	-	-	-	-
0x380	-	-	-	IGBTC [B,H,W] 00000000
0x384 - 0xFFC	-	-	-	-

1.9. Base Timer

ch.0 Base Address : 0x4002_5000
ch.1 Base Address : 0x4002_5040
ch.2 Base Address : 0x4002_5080
ch.3 Base Address : 0x4002_50C0
ch.4 Base Address : 0x4002_5200
ch.5 Base Address : 0x4002_5240
ch.6 Base Address : 0x4002_5280
ch.7 Base Address : 0x4002_52C0
ch.8 Base Address : 0x4002_5400
ch.9 Base Address : 0x4002_5440
ch.10 Base Address : 0x4002_5480
ch.11 Base Address : 0x4002_54C0
ch.12 Base Address : 0x4002_5600
ch.13 Base Address : 0x4002_5640
ch.14 Base Address : 0x4002_5680
ch.15 Base Address : 0x4002_56C0

Base_Address	Register			
+ Address	+3	+2	+1	+0
0x000	-	-	PCSR/PRLL [H,W] XXXXXXXX XXXXXXXX	
0x004	-	-	PDUT/PRLH/DTBF [H,W] XXXXXXXX XXXXXXXX	
0x008	-	-	TMR [H,W] 00000000 00000000	
0x00C	-	-	TMCR [B,H,W] -0000000 00000000	
0x010	-	-	TMCR2 [B,H,W] -------0	STC [B,H,W] 0000-000
0x014 - 0x03C	-	-	-	-

1.10. IO Selector for ch.0-ch.3 (Base Timer)

Base Address : 0x4002_5100

Base_Address + Address	Register +3	+2	+1	+0
0x000	-	-	BTSEL0123 [B,H,W] 00000000	-
0x004 - 0x0FC	-	-	-	-

1.11. IO Selector for ch.4-ch.7(Base Timer)

Base Address : 0x4002_5300

Base_Address + Address	Register +3	+2	+1	+0
0x000	-	-	BTSEL4567 [B,H,W] 00000000	-
0x004 - 0x0FC	-	-	-	-

1.12. IO Selector for ch.8-ch.11(Base Timer)

Base Address : 0x4002_5500

Base_Address + Address	Register +3	+2	+1	+0
0x000	-	-	BTSEL89AB [B,H,W] 00000000	-
0x004 - 0x0FC	-	-	-	-

1.13. IO Selector for ch.12-ch.15 (Base Timer)

Base Address : 0x4002_5700

Base_Address + Address	Register			
	+3	+2	+1	+0
0x000	-	-	BTSELCDEF [B,H,W] 00000000	-
0x004 - 0x0FC	-	-	-	-

1.14. Software-based Simultaneous Startup (Base Timer)

Base Address : 0x4002_5F00

Base_Address + Address	Register			
	+3	+2	+1	+0
0x000 - 0x0FB	-	-	-	-
0x0FC	-	-	BTSSSR [B,H,W] XXXXXXXX XXXXXXXX	

1.15. QPRC

ch.0 Base Address : 0x4002_6000

ch.1 Base Address : 0x4002_6040

ch.2 Base Address : 0x4002_6080

Base_Address	Register			
+ Address	+3	+2	+1	+0
0x000	-	-	QPCR [H,W] 00000000 00000000	
0x004	-	-	QRCR [H,W] 00000000 00000000	
0x008	-	-	QPCCR [H,W] 00000000 00000000	
0x00C	-	-	QPRCR [H,W] 00000000 00000000	
0x010	-	-	QMPR [H,W] 11111111 11111111	
0x014	-	-	QICRH [B,H,W] --000000	QICRL [B,H,W] 00000000
0x018	-	-	QCRH [B,H,W] 00000000	QCRL [B,H,W] 00000000
0x01C	-	-	QECR [B,H,W] -------- -----000	
0x020 - 0x038	-	-	-	-
0x03C	QPCRR [B,H,W] 00000000 00000000		QRCRR [B,H,W] 00000000 00000000	

1.16. 12-bit A/DC

unit0 Base_Address : 0x4002_7000
unit1 Base_Address : 0x4002_7100
unit2 Base_Address : 0x4002_7200

■ TYPE0 to TYPE2, TYPE4, and TYPE5 products

Base_Address	Register			
+ Address	+3	+2	+1	+0
0x000	-	-	ADCR[B,H,W] 000-0000	ADSR[B,H,W] 00---000
0x004	-	-	-	*
0x008	-	-	SCCR[B,H,W] 1000-000	SFNS[B,H,W] ----0000
0x00C	SCFD[B,H,W] XXXXXXXX XXXX---- ---1--XX ---XXXXX			
0x010	-	-	SCIS3[B,H,W] 00000000	SCIS2[B,H,W] 00000000
0x014	-	-	SCIS1[B,H,W] 00000000	SCIS0[B,H,W] 00000000
0x018	-	-	PCCR[B,H,W] 1000-000	PFNS[B,H,W] --XX--00
0x01C	PCFD[B,H,W] XXXXXXXX XXXX---- ---1-XXX ---XXXXX			
0x020	-	-	-	PCIS[B,H,W] 00000000
0x024	CMPD[B,H,W] 00000000 00------		-	CMPCR[B,H,W] 00000000
0x028	-	-	ADSS3[B,H,W] 00000000	ADSS2[B,H,W] 00000000
0x02C	-	-	ADSS1[B,H,W] 00000000	ADSS0[B,H,W] 00000000
0x030	-	-	ADST0[B,H,W] 00010000	ADST1[B,H,W] 00010000
0x034	-	-	-	ADCT[B,H,W] 00000111
0x038	-	-	SCTSL[B,H,W] ----0000	PRTSL[B,H,W] ----0000
0x03C	-	-	-	ADCEN[B,H,W] --00--00
0x040 - 0x0FC	-	-	-	-

■ TYPE3, and TYPE6 to TYPE12 products

Base_Address + Address	Register +3	Register +2	Register +1	Register +0
0x000	-	-	ADCR[B,H,W] 000-0000	ADSR[B,H,W] 00---000
0x004	-	-	-	*
0x008	-	-	SCCR[B,H,W] 1000-000	SFNS[B,H,W] ----0000
0x00C	SCFD[B,H,W] XXXXXXXX XXXX---- ---1--XX ---XXXXX			
0x010	-	-	SCIS3[B,H,W] 00000000	SCIS2[B,H,W] 00000000
0x014	-	-	SCIS1[B,H,W] 00000000	SCIS0[B,H,W] 00000000
0x018	-	-	PCCR[B,H,W] 10000000	PFNS[B,H,W] --XX--00
0x01C	PCFD[B,H,W] XXXXXXXX XXXX---- ---1-XXX ---XXXXX			
0x020	-	-	-	PCIS[B,H,W] 00000000
0x024	CMPD[B,H,W] 00000000 00------		-	CMPCR[B,H,W] 00000000
0x028	-	-	ADSS3[B,H,W] 00000000	ADSS2[B,H,W] 00000000
0x02C	-	-	ADSS1[B,H,W] 00000000	ADSS0[B,H,W] 00000000
0x030	-	-	ADST0[B,H,W] 00010000	ADST1[B,H,W] 00010000
0x034	-	-	-	ADCT[B,H,W] 00000111
0x038	-	-	SCTSL[B,H,W] ----0000	PRTSL[B,H,W] ----0000
0x03C	-	-	ADCEN[B,H,W] 11111111 ------00	
0x040 - 0x0FC	-	-	-	-

1.17. 10-bit D/AC

Base_Address : 0x4002_8000

Base_Address + Address	Register			
	+3	+2	+1	+0
0x000	-	DACR0[B,H,W] -------0	DADR0[B,H,W] ------XX XXXXXXXX	
0x004	-	DACR1[B,H,W] -------0	DADR1[B,H,W] ------XX XXXXXXXX	
0x008 - 0x0FC	-	-	-	-

1.18. CR Trim

Base_Address : 0x4002_E000

Base_Address + Address	Register			
	+3	+2	+1	+0
0x000	-	-	-	MCR_PSR [B,H,W] ------01
0x004	-	-	MCR_FTRM[B,H,W] ------01 10000000 *1 ------01 10001110 *6 -------- 01111111 *4 ------10 00000000 *5	
0x008	-	-	-	MCR_TTRM [B,H,W] --011111
0x00C	MCR_RLR[W] 00000000 00000000 00000000 00000001			
0x010 - 0x0FC	-	-	-	-

1.19. EXTI

Base_Address : 0x4003_0000

Base_Address + Address	Register +3	+2	+1	+0
0x000	ENIR[B,H,W] 00000000 00000000 00000000 00000000			
0x004	EIRR[B,H,W] XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX			
0x008	EICL[B,H,W] 11111111 11111111 11111111 11111111			
0x00C	ELVR[B,H,W] 00000000 00000000 00000000 00000000			
0x010	ELVR1[B,H,W] 00000000 00000000 00000000 00000000			
0x014	-	-	NMIRR[B,H,W] -------- -------0	
0x018	-	-	NMICL[B,H,W] -------- -------1	
0x01C	-	-	-	-
0x020 - 0x0FC	-	-	-	-

1.20. INT-Req. READ

Base_Address : 0x4003_1000

■ Products other than TYPE3/TYPE7

Base_Address + Address	Register +3	Register +2	Register +1	Register +0
0x000	DRQSEL[B,H,W] 00000000 00000000 00000000 00000000			
0x004	*			
0x008	ODDPKS[B] ---00000	-	-	*
0x00C	-	-	-	IRQCMODE[B,H,W] -------0
0x010	EXC02MON[B,H,W] -------- -------- -------- ------00			
0x014	IRQ00MON[B,H,W] -------- -------- -------- -------0			
0x018	IRQ01MON[B,H,W] -------- -------- -------- -------0			
0x01C	IRQ02MON[B,H,W] -------- -------- -------- -------0			
0x020	IRQ03MON[B,H,W] -------- -------- ----0000 00000000			
0x024	IRQ04MON[B,H,W] -------- -------- -------- 00000000			
0x028	IRQ05MON[B,H,W] -------- 00000000 00000000 00000000			
0x02C	IRQ06MON[B,H,W] -------- ----0000 00000000 00000000			
0x030	IRQ07MON[B,H,W] -------- -------- -------- ------00			
0x034	IRQ08MON[B,H,W] -------- -------- -------- ----0000			
0x038	IRQ09MON[B,H,W] -------- -------- -------- ------00			
0x03C	IRQ10MON[B,H,W] -------- -------- -------- ----0000			

Base_Address + Address	Register			
	+3	+2	+1	+0
0x040	IRQ11MON[B,H,W] -------- -------- -------- ------00			
0x044	IRQ12MON[B,H,W] -------- -------- -------- ----0000			
0x048	IRQ13MON[B,H,W] -------- -------- -------- ------00			
0x04C	IRQ14MON[B,H,W] -------- -------- -------- ----0000			
0x050	IRQ15MON[B,H,W] -------- -------- -------- ------00			
0x054	IRQ16MON[B,H,W] -------- -------- -------- ----0000			
0x058	IRQ17MON[B,H,W] -------- -------- -------- ------00			
0x05C	IRQ18MON[B,H,W] -------- -------- -------- ----0000			
0x060	IRQ19MON[B,H,W] -------- -------- -------- ------00			
0x064	IRQ20MON[B,H,W] -------- -------- -------- ----0000			
0x068	IRQ21MON[B,H,W] -------- -------- -------- ------00			
0x06C	IRQ22MON[B,H,W] -------- -------- -------- ----0000			
0x070	IRQ23MON[B,H,W] -------- -------- -------0 00000000			
0x074	IRQ24MON[B,H,W] -------- -------- -------- --000000			
0x078	IRQ25MON[B,H,W] -------- -------- -------- ----0000			
0x07C	IRQ26MON[B,H,W] -------- -------- -------- ----0000			

A. Register Map

Base_Address + Address	Register			
	+3	+2	+1	+0
0x080	IRQ27MON[B,H,W] -------- -------- -------- ---00000			
0x084	IRQ28MON[B,H,W] -------- ------00 00000000 00000000			
0x088	IRQ29MON[B,H,W] -------- -------- ----0000 00000000			
0x08C	IRQ30MON[B,H,W] -------- ------00 00000000 00000000			
0x090	IRQ31MON[B,H,W] -------- -------- 00000000 00000000			
0x094	IRQ32MON[B,H,W] -------- -------- -------- ----0000			
0x098	IRQ33MON[B,H,W] -------- -------- -------- -----000			
0x09C	IRQ34MON[B,H,W] -------- -------- -------- ---00000			
0x0A0	IRQ35MON[B,H,W] -------- -------- -------- --000000			
0x0A4	IRQ36MON[B,H,W] -------- -------- -------- --000000			
0x0A8	IRQ37MON[B,H,W] -------- -------- -------- -0000000			
0x0AC	IRQ38MON[B,H,W] -------- -------- -------- -------0			
0x0B0	IRQ39MON[B,H,W] -------- -------- -------- -------0			
0x0B4	IRQ40MON[B,H,W] -------- -------- -------- -------0			
0x0B8	IRQ41MON[B,H,W] -------- -------- -------- -------0			
0x0BC	IRQ42MON[B,H,W] -------- -------- -------- -------0			

<table>
<tr><td rowspan="2">Base_Address
+ Address</td><td colspan="4">Register</td></tr>
<tr><td>+3</td><td>+2</td><td>+1</td><td>+0</td></tr>
<tr><td>0x0C0</td><td colspan="4">IRQ43MON[B,H,W]
-------- -------- -------- -------0</td></tr>
<tr><td>0x0C4</td><td colspan="4">IRQ44MON[B,H,W]
-------- -------- -------- -------0</td></tr>
<tr><td>0x0C8</td><td colspan="4">IRQ45MON[B,H,W]
-------- -------- -------- -------0</td></tr>
<tr><td rowspan="2">0x0CC</td><td colspan="4">IRQ46MON[B,H,W]</td></tr>
<tr><td colspan="4">-------- -------- 00000000 00000000</td></tr>
<tr><td>0x0D0</td><td colspan="4">IRQ47MON[B,H,W]
-------- -------- ----0--- --------</td></tr>
<tr><td>0x0D4 - 0x1FC</td><td>-</td><td>-</td><td>-</td><td>-</td></tr>
<tr><td>0x200</td><td colspan="4">DRQSEL1[B,H,W]
-------- -------- -------- ---00000</td></tr>
<tr><td>0x204</td><td colspan="4">DQESEL[B,H,W]
00000000 00000000 00000000 00000000</td></tr>
<tr><td>0x208</td><td colspan="4">*</td></tr>
<tr><td>0x20C</td><td>ODDPKS[B]
---00000</td><td>-</td><td>-</td><td>*</td></tr>
<tr><td>0x210</td><td>RCINTSEL3[B,H,W]
---00000</td><td>RCINTSEL2[B,H,W]
---00000</td><td>RCINTSEL1[B,H,W]
---00000</td><td>RCINTSEL0[B,H,W]
---00000</td></tr>
<tr><td>0x214</td><td>RCINTSEL7[B,H,W]
---00000</td><td>RCINTSEL6[B,H,W]
---00000</td><td>RCINTSEL5[B,H,W]
---00000</td><td>RCINTSEL4[B,H,W]
---00000</td></tr>
<tr><td>0x218 - 0xFFC</td><td>-</td><td>-</td><td>-</td><td>-</td></tr>
</table>

■ TYPE3/TYPE7 products

Base_Address + Address	Register +3	Register +2	Register +1	Register +0
0x000	*			
0x004	*			
0x008	*			
0x00C	-	-	-	-
0x010	EXC02MON[B,H,W] -------- -------- -------- ------00			
0x014	IRQ00MON[B,H,W] -------- -------- -------- -------0			
0x018	IRQ01MON[B,H,W] -------- -------- -------- -------0			
0x01C	IRQ02MON[B,H,W] -------- -------- -------- -------0			
0x020	IRQ03MON[B,H,W] -------- -------- -------- ----0000			
0x024	IRQ04MON[B,H,W] -------- -------- -------- -0000000			
0x028	IRQ05MON[B,H,W] -------- -------- -------- 0-------			
0x02C	IRQ06MON[B,H,W] -------- -------- -------- -------0			
0x030	IRQ07MON[B,H,W] -------- -------- -------- ------00			
0x034	IRQ08MON[B,H,W] -------- -------- -------- -------0			
0x038	IRQ09MON[B,H,W] -------- -------- -------- ------00			
0x03C	IRQ10MON[B,H,W] -------- -------- -------- -------0			
0x040	IRQ11MON[B,H,W] -------- -------- -------- ------00			
0x044	IRQ12MON[B,H,W] -------- -------- -------- -------0			

Base_Address + Address	Register +3	+2	+1	+0
0x048	IRQ13MON[B,H,W] -------- -------- -------- ------00			
0x04C	IRQ14MON[B,H,W] -------- -------- -------- -------0			
0x050	IRQ15MON[B,H,W] -------- -------- -------- ------00			
0x054	IRQ16MON[B,H,W] -------- -------- -------- -------0			
0x058	IRQ17MON[B,H,W] -------- -------- -------- ------00			
0x05C	IRQ18MON[B,H,W] -------- -------- -------- -------0			
0x060	IRQ19MON[B,H,W] -------- -------- -------- ------00			
0x064	IRQ20MON[B,H,W] -------- -------- -------- ------0			
0x068	IRQ21MON[B,H,W] -------- -------- -------- ------00			
0x06C	IRQ22MON[B,H,W] -------- -------- -------- -----000			
0x070	IRQ23MON[B,H,W] -------- -------- -------- --0--000			
0x074	IRQ24MON[B,H,W] -------- -------- -------- ----0000			
0x078	IRQ25MON[B,H,W] -------- -------- -------- --000000			
0x07C	IRQ26MON[B,H,W] -------- -------- -------- ----0000			
0x080	IRQ27MON[B,H,W] -------- -------- -------- --000000			
0x084	IRQ28MON[B,H,W] -------- -------- 00000000 00000000			

A. Register Map

Base_Address + Address	Register			
	+3	+2	+1	+0
0x088	IRQ29MON[B,H,W] -------- -------- -------- ---0----			
0x08C	IRQ30MON[B,H,W] -------- -------- -------- --0-----			
0x090	IRQ31MON[B,H,W] -------- -------- -------- -0------			

1.21. LCDC

Base_Address : 0x4003_2000

Base_Address	Register			
+ Address	+3	+2	+1	+0
0x000	-	LCDCC3[B,H,W] 0011111-	LCDCC2[B,H,W] --010100	LCDCC1[B,H,W] -00000--
0x004	LCDC_PSR[B,H,W] -------- -0000000 00000000 00000000			
0x008	LCDC_COMEN[B,H,W] -------- -------- -------- 00000000			
0x00C	LCDC_SEGEN1[B,H,W] 00000000 00000000 00000000 00000000			
0x010	LCDC_SEGEN2[B,H,W] -------- -------- -------- 00000000			
0x014	-	-	LCDC_BLINK[B,H,W] 00000000 00000000	
0x018	-	-	-	-
0x01C	LCDRAM03[B,H,W] 00000000	LCDRAM02[B,H,W] 00000000	LCDRAM01[B,H,W] 00000000	LCDRAM00[B,H,W] 00000000
0x020	LCDRAM07[B,H,W] 00000000	LCDRAM06[B,H,W] 00000000	LCDRAM05[B,H,W] 00000000	LCDRAM04[B,H,W] 00000000
0x024	LCDRAM11[B,H,W] 00000000	LCDRAM10[B,H,W] 00000000	LCDRAM09[B,H,W] 00000000	LCDRAM08[B,H,W] 00000000
0x028	LCDRAM15[B,H,W] 00000000	LCDRAM14[B,H,W] 00000000	LCDRAM13[B,H,W] 00000000	LCDRAM12[B,H,W] 00000000
0x02C	LCDRAM19[B,H,W] 00000000	LCDRAM18[B,H,W] 00000000	LCDRAM17[B,H,W] 00000000	LCDRAM16[B,H,W] 00000000
0x030	LCDRAM23[B,H,W] 00000000	LCDRAM22[B,H,W] 00000000	LCDRAM21[B,H,W] 00000000	LCDRAM20[B,H,W] 00000000
0x034	LCDRAM27[B,H,W] 00000000	LCDRAM26[B,H,W] 00000000	LCDRAM25[B,H,W] 00000000	LCDRAM24[B,H,W] 00000000
0x038	LCDRAM31[B,H,W] 00000000	LCDRAM30[B,H,W] 00000000	LCDRAM29[B,H,W] 00000000	LCDRAM28[B,H,W] 00000000
0x03C	LCDRAM35[B,H,W] 00000000	LCDRAM34[B,H,W] 00000000	LCDRAM33[B,H,W] 00000000	LCDRAM32[B,H,W] 00000000
0x040	LCDRAM39[B,H,W] 00000000	LCDRAM38[B,H,W] 00000000	LCDRAM37[B,H,W] 00000000	LCDRAM36[B,H,W] 00000000
0x044 - 0x0FC	-	-	-	-

1.22. GPIO

Base_Address : 0x4003_3000

Base_Address + Address	Register +3	Register +2	Register +1	Register +0
0x000	PFR0[B,H,W] ---- ---- ---- ---- 0000 0000 0001 1111			
0x004	PFR1[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x008	PFR2[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x00C	PFR3[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x010	PFR4[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x014	PFR5[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x018	PFR6[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x01C	PFR7[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x020	PFR8[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x024	PFR9[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x028	PFRA[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x02C	PFRB[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x030	PFRC[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x034	PFRD[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x038	PFRE[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x03C	PFRF[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x040 - 0x0FC	-	-	-	-

Base_Address	Register			
+ Address	+3	+2	+1	+0
0x100	PCR0[B,H,W] ---- ---- ---- ---- 0000 0000 0001 1111			
0x104	PCR1[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x108	PCR2[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x10C	PCR3[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x110	PCR4[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x114	PCR5[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x118	PCR6[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x11C	PCR7[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x120	PCRB[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x124	PCR9[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x128	PCRA[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x12C	PCRB[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x130	PCRC[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x134	PCRD[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x138	PCRE[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x13C	PCRF[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x140 - 0x1FC	-			
0x200	DDR0[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			

Base_Address + Address	Register			
	+3	+2	+1	+0
0x204	DDR1[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x208	DDR2[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x20C	DDR3[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x210	DDR4[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x214	DDR5[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x218	DDR6[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x21C	DDR7[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x220	DDR8[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x224	DDR9[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x228	DDRA[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x22C	DDRB[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x230	DDRC[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x234	DDRD[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x238	DDRE[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x23C	DDRF[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x240 - 0x2FC	-	-	-	-
0x300	PDIR0[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x304	PDIR1[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			

Base_Address	Register			
+ Address	+3	+2	+1	+0
0x308	PDIR2[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x30C	PDIR3[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x310	PDIR4[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x314	PDIR5[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x318	PDIR6[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x31C	PDIR7[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x320	PDIR8[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x324	PDIR9[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x328	PDIRA[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x32C	PDIRB[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x330	PDIRC[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x334	PDIRD[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x338	PDIRE[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x33C	PDIRF[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x340 - 0x3FC	-	-	-	-
0x400	PDOR0[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x404	PDOR1[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x408	PDOR2[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			

Base_Address + Address	Register			
	+3	+2	+1	+0
0x40C	PDOR3[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x410	PDOR4[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x414	PDOR5[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x418	PDOR6[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x41C	PDOR7[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x420	PDOR8[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x424	PDOR9[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x428	PDORA[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x42C	PDORB[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x430	PDORC[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x434	PDORD[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x438	PDORE[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x43C	PDORF[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x440 - 0x4FC	-	-	-	-
0x500	ADE[B,H,W] 1111 1111 1111 1111 1111 1111 1111 1111			
0x504 - 0x57C	-	-	-	-
0x580	SPSR[B,H,W] ---- ---- ---- ---- ---- ---- ---0 ---1 *1 ---- ---- ---- ---- ---- ---- ---0 0101 *2			
0x584 - 0x5FC	-	-	-	-

Base_Address + Address	Register +3	+2	+1	+0
0x600	EPFR00[B,H,W] ---- --00 ---- --11 --0- --0- 0000 --00			
0x604	EPFR01[B,H,W] 0000 0000 0000 0000 ---0 0000 0000 0000			
0x608	EPFR02[B,H,W] 0000 0000 0000 0000 ---0 0000 0000 0000			
0x60C	EPFR03[B,H,W] 0000 0000 0000 0000 ---0 0000 0000 0000			
0x610	EPFR04[B,H,W] --00 0000 --00 00-- --00 0000 -000 00--			
0x614	EPFR05[B,H,W] --00 0000 --00 00-- --00 0000 --00 00--			
0x618	EPFR06[B,H,W] 0000 0000 0000 0000 0000 0000 0000 0000			
0x61C	EPFR07[B,H,W] ---- 0000 0000 0000 0000 0000 0000 ----			
0x620	EPFR08[B,H,W] ---- 0000 0000 0000 0000 0000 0000 0000			
0x624	EPFR09[B,H,W] 0000 0000 0000 0000 0000 0000 0000 0000			
0x628	EPFR10[B,H,W] 0000 0000 0000 0000 0000 0000 0000 0000			
0x62C	EPFR11[B,H,W] ---- --00 0000 0000 0000 0000 0000 0000			
0x630	EPFR12[B,H,W] --00 0000 --00 00-- --00 0000 --00 00--			
0x634	EPFR13[B,H,W] --00 0000 --00 00-- --00 0000 --00 00--			
0x638	EPFR14[B,H,W] 0000 0000 0000 0000 0000 0000 0000 0000			
0x63C	EPFR15[B,H,W] 0000 0000 0000 0000 0000 0000 0000 0000			
0x640	EPFR16[B,H,W] ---- 0000 0000 0000 0000 0000 0000 ----			

A. Register Map

Base_Address + Address	Register +3	+2	+1	+0
0x644	EPFR17[B,H,W] ---- 0000 0000 0000 0000 0000 0000 ----			
0x648	EPFR18[B,H,W] ---- ---- ---- ---- ---- ---- ---- 0000			
0x64C - 0x6FC	-	-	-	-
0x700	PZR0[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x704	PZR1[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x708	PZR2[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x70C	PZR3[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x710	PZR4[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x714	PZR5[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x718	PZR6[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x71C	PZR7[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x720	PZR8[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x724	PZR9[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x728	PZRA[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x72C	PZRB[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x730	PZRC[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x734	PZRD[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			

Base_Address + Address	Register			
	+3	+2	+1	+0
0x738	PZRE[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x73C	PZRF[B,H,W] ---- ---- ---- ---- 0000 0000 0000 0000			
0x740 - 0x7FC	-	-	-	-
0x800	*			
0x804	*			
0x808 - 0xFFC	-	-	-	-

1.23. HDMI-CEC/Remote Control Receiver

ch.0 Base_Address : 0x4003_4000

ch.1 Base_Address : 0x4003_4100

Base_Address + Address	Register			
	+3	+2	+1	+0
0x000	-	-	-	TXCTRL[B,H,W] --0000-0
0x004	-	-	-	TXDATA[B,H,W] 00000000
0x008	-	-	-	TXSTS[B,H,W] --00---0
0x00C	-	-	-	SFREE[B,H,W] ----0000
0x010 - 0x03F	-	-	-	-
0x040	-	-	RCCR[B,H,W] 0---0000	RCST[B,H,W] 00000000
0x044	-	-	RCSHW[B,H,W] 00000000	RCDAHW[B,H,W] 00000000
0x048	-	-	RCDBHW[B,H,W] 00000000	-
0x04C	-	-	RCADR1[B,H,W] ---00000	RCADR2[B,H,W] ---00000
0x050	-	-	RCDTHH[B,H,W] 00000000	RCDTHL[B,H,W] 00000000
0x054	-	-	RCDTLH[B,H,W] 00000000	RCDTLL[B,H,W] 00000000
0x058	-	-	RCCKD[H,W] ---00000 00000000	
0x05C	-	-	RCRC[B,H,W] ---0---0	RCRHW[B,H,W] 00000000
0x060	-	-	RCLE[B,H,W] 00000-00	-
0x064	-	-	RCLELW[B,H,W] 00000000	RCLESW[B,H,W] 00000000
0x068 - 0x0FC	-	-	-	-

1.24. LVD

Base_Address : 0x4003_5000

■ TYPE0/TYPE1/TYPE2/TYPE4/TYPE5 products

Base_Address + Address	Register +3	Register +2	Register +1	Register +0
0x000	-	-	-	LVD_CTL [B,H,W] 010000--
0x004	-	-	-	LVD_STR [B,H,W] 0-------
0x008	-	-	-	LVD_CLR [B,H,W] 1-------
0x00C	LVD_RLR[W] 00000000 00000000 00000000 00000001			
0x010	-	-	-	LVD_STR2 0-------
0x014 - 0xFFC	-	-	-	-

■ TYPE3, and TYPE6 to TYPE12 products

Base_Address + Address	Register +3	Register +2	Register +1	Register +0
0x000	-	-	LVD_CTL[B, H, W] 1-0001-- 0-00000- *6 100001-- 000100-- *7 100001-- 000011-- *8	
0x004	-	-	-	LVD_STR[B,H,W] 0-------
0x008	-	-	-	LVD_CLR[B,H,W] 1-------
0x00C	LVD_RLR[W] 00000000 00000000 00000000 00000001			
0x010	-	-	-	LVD_STR2 01------
0x014 - 0x7FC	-	-	-	-

1.25. DS_Mode

Base_Address : 0x4003_5100

Base_Address + Address	Register			
	+3	+2	+1	+0
0x000	-	-	-	REG_CTL[B,H,W] -------0
0x004	-	-	-	RCK_CTL[B,H,W] ------01
0x008 - 0x6FC	-	-	-	-
0x700	-	-	-	PMD_CTL[B,H,W] -------0
0x704	-	-	-	WRFSR[B,H,W] ------00
0x708	-	-	WIFSR[B,H,W] ------00 00000000	
0x70C	-	-	WIER[B,H,W] ------00 00000-00	
0x710	-	-	-	WILVR[B,H,W] -----000
0x714	-	-	-	DSRAMR[B,H,W] ------00
0x718 - 0x7FC	-	-	-	-
0x800	BUR04[B,H,W] 00000000	BUR03[B,H,W] 00000000	BUR02[B,H,W] 00000000	BUR01[B,H,W] 00000000
0x804	BUR08[B,H,W] 00000000	BUR07[B,H,W] 00000000	BUR06[B,H,W] 00000000	BUR05[B,H,W] 00000000
0x808	BUR012[B,H,W] 00000000	BUR11[B,H,W] 00000000	BUR10[B,H,W] 00000000	BUR09[B,H,W] 00000000
0x80C	BUR16[B,H,W] 00000000	BUR15[B,H,W] 00000000	BUR14[B,H,W] 00000000	BUR13[B,H,W] 00000000
0x810 - 0xEFC	-	-	-	-

1.26. USB Clock

Base_Address : 0x4003_6000

■ Products other than TYPE2

Base_Address + Address	Register			
	+3	+2	+1	+0
0x000	-	-	-	UCCR[B,H,W] ------00
0x004	-	-	-	UPCR1[B,H,W] ------00
0x008	-	-	-	UPCR2[B,H,W] -----000
0x00C	-	-	-	UPCR3[B,H,W] ---00000
0x010	-	-	-	UPCR4[B,H,W] ---10111 *1 -0111011 *2
0x014	-	-	-	UP_STR[B,H,W] -------0
0x018	-	-	-	UPINT_ENR[B,H,W] -------0
0x01C	-	-	-	UPINT_CLR[B,H,W] -------0
0x020	-	-	-	UPINT_STR[B,H,W] -------0
0x024	-	-	-	UPCR5[B,H,W] ----0100
0x028 - 0x02C	-	-	-	-
0x030	-	-	-	USBEN[B,H,W] -------0
0x034 - 0x0FC	-	-	-	-

■ TYPE2 products

Base_Address	Register			
+ Address	+3	+2	+1	+0
0x000	-	-	-	UCCR[B,H,W] -0000000
0x004	-	-	-	UPCR1[B,H,W] ------00
0x008	-	-	-	UPCR2[B,H,W] -----000
0x00C	-	-	-	UPCR3[B,H,W] ---00000
0x010	-	-	-	UPCR4[B,H,W] -0111011
0x014	-	-	-	UP_STR[B,H,W] -------0
0x018	-	-	-	UPINT_ENR[B,H,W] -------0
0x01C	-	-	-	UPINT_CLR[B,H,W] -------0
0x020	-	-	-	UPINT_STR[B,H,W] -------0
0x024	-	-	-	UPCR5[B,H,W] ----0100
0x028	-	-	-	UPCR6[B,H,W] ----0010
0x02C	-	-	-	UPCR7[B,H,W] -------0
0x030	-	-	-	USBEN[B,H,W] -------0
0x034	-	-	-	USBEN1[B,H,W] -------0
0x038 - 0x0FC	-	-	-	-

1.27. CAN_Prescaler

Base_Address : 0x4003_7000

Base_Address + Address	Register			
	+3	+2	+1	+0
0x000	-	-	-	CANPRE[B,H,W] ----1011
0x004 - 0xFFC	-	-	-	-

1.28. MFS

■ Products other than TYPE8/TYPE12

ch.0 Base_Address : 0x4003_8000
ch.1 Base_Address : 0x4003_8100
ch.2 Base_Address : 0x4003_8200
ch.3 Base_Address : 0x4003_8300
ch.4 Base_Address : 0x4003_8400
ch.5 Base_Address : 0x4003_8500
ch.6 Base_Address : 0x4003_8600
ch.7 Base_Address : 0x4003_8700

Base_Address	Register			
+ Address	+3	+2	+1	+0
0x000	-	-	SCR/ IBCR[B,H,W] 0--00000	SMR[B,H,W] 000-00-0
0x004	-	-	SSR[B,H,W] 0-000011	ESCR/ IBSR[B,H,W] 00000000
0x008	-	-	RDR/TDR[H,W] -------0 00000000	
0x00C	-	-	BGR1[B,H,W] 00000000	BGR0[B,H,W] 00000000
0x010	-	-	ISMK[B,H,W] --------	ISBA[B,H,W] --------
0x014	-	-	FCR1[B,H,W] ---00100	FCR0[B,H,W] -0000000
0x018	-	-	FBYTE2[B,H,W] 00000000	FBYTE1[B,H,W] 00000000
0x1C			EIBCR[B, H, W] --001100	-
0x020 - 0x0FC	-	-	-	-

MFS Noise Filter Control Base_Address : 0x4003_8800

Base_Address	Register			
+ Address	+3	+2	+1	+0
0x000	-	-	I2CDNF[B,H,W] 00000000	
0x004 - 0x0FC	-	-	-	-

■ TYPE8/TYPE12 products

ch.0 Base_Address : 0x4003_8000
ch.1 Base_Address : 0x4003_8100
ch.2 Base_Address : 0x4003_8200
ch.3 Base_Address : 0x4003_8300
ch.4 Base_Address : 0x4003_8400
ch.5 Base_Address : 0x4003_8500
ch.6 Base_Address : 0x4003_8600
ch.7 Base_Address : 0x4003_8700
ch.8 Base_Address : 0x4003_8800
ch.9 Base_Address : 0x4003_8900
ch.10 Base_Address : 0x4003_8A00
ch.11 Base_Address : 0x4003_8B00
ch.12 Base_Address : 0x4003_8C00
ch.13 Base_Address : 0x4003_8D00
ch.14 Base_Address : 0x4003_8E00
ch.15 Base_Address : 0x4003_8F00

Base_Address + Address	Register			
	+3	+2	+1	+0
0x000	-	-	SCR/ IBCR[B,H,W] 0--00000	SMR[B,H,W] 00-000-0
0x004	-	-	SSR[B,H,W] 0-000011	ESCR/ IBSR[B,H,W] 00000000
0x008	-	-	RDR/TDR[H,W] -------0 00000000	
0x00C	-	-	BGR1[B,H,W] 00000000	BGR0[B,H,W] 00000000
0x010	-	-	ISMK[B,H,W] --------	ISBA[B,H,W] --------
0x014	-	-	FCR1[B,H,W] ---00100	FCR0[B,H,W] -0000000
0x018	-	-	FBYTE2[B,H,W] 00000000	FBYTE1[B,H,W] 00000000
0x1C			EIBCR[B, H, W --001000	-
0x020 - 0x0FC	-	-	-	-

1.29. CRC

Base_Address : 0x4003_9000

Base_Address + Address	Register			
	+3	+2	+1	+0
0x000	-	-	-	CRCCR[B,H,W] -0000000
0x004	CRCINIT[B,H,W] 11111111 11111111 11111111 11111111			
0x008	CRCIN[B,H,W] 00000000 00000000 00000000 00000000			
0x00C	CRCR[B,H,W] 11111111 11111111 11111111 11111111			

1.30. Watch Counter

Base_Address : 0x4003_A000

Base_Address + Address	Register			
	+3	+2	+1	+0
0x000	-	WCCR[B,H,W] 00--0000	WCRL[B,H,W] --000000	WCRD[B,H,W] --000000
0x004 - 0x00C	-	-	-	-
0x010	-	-	CLK_SEL[B,H,W] -----000 -------0	
0x014	-	-	-	CLK_EN[B,H,W] ------00
0x018 - 0xFFC	-	-	-	-

1.31. RTC

Base_Address : 0x4003_B000

■ TYPE3/TYPE4/TYPE5 products

Base_Address + Address	Register			
	+3	+2	+1	+0
0x000	WTCR1[B,H,W] 00000000 00000000 ---00000 -00000-0			
0x004	WTCR2[B,H,W] -------- -------- -----000 -------0			
0x008	WTBR[B,H,W] -------- 00000000 00000000 00000000			
0x00C	WTDR[B,H,W] --000000	WTHR[B,H,W] --000000	WTMIR[B,H,W] -0000000	WTSR[B,H,W] -0000000
0x010	-	WTYR[B,H,W] 00000000	WTMOR[B,H,W] ---00000	WTDW[B,H,W] -----000
0x014	ALDR[B,H,W] --000000	ALHR[B,H,W] --000000	ALMIR[B,H,W] -0000000	-
0x018	-	ALYR[B,H,W] 00000000	ALMOR[B,H,W] ---00000	-
0x01C	WTTR[B,H,W] -------- ------00 00000000 00000000			
0x020	-	-	WTCLKM[B,H,W] ------00	WTCLKS [B,H,W] -------0
0x024	-	-	WTCALEN[B,H,W] -------0	WTCAL [B,H,W] -0000000
0x028	-	-	WTDIVEN[B,H,W] ------00	WTDIV [B,H,W] ----0000
0x02C - 0xFFC	-	-	-	-

■ TYPE6 to TYPE12 products

Base_Address + Address	Register +3	Register +2	Register +1	Register +0
0x000	WTCR1[B,H,W] 00000000 00000000 ---00000 -00000-0			
0x004	WTCR2[B,H,W] -------- -------- -----000 -------0			
0x008	WTBR[B,H,W] -------- 00000000 00000000 00000000			
0x00C	WTDR[B,H,W] --000000	WTHR[B,H,W] --000000	WTMIR[B,H,W] -0000000	WTSR[B,H,W] -0000000
0x010	-	WTYR[B,H,W] 00000000	WTMOR[B,H,W] ---00000	WTDW[B,H,W] -----000
0x014	ALDR[B,H,W] --000000	ALHR[B,H,W] --000000	ALMIR[B,H,W] -0000000	-
0x018	-	ALYR[B,H,W] 00000000	ALMOR[B,H,W] ---00000	-
0x01C	WTTR[B,H,W] -------- ------00 00000000 00000000			
0x020	-	-	WTCLKM[B,H,W] ------00	WTCLKS [B,H,W] -------0
0x024	-	WTCALEN[B,H,W] -------0	WTCAL [B,H,W] ------00 00000000	
0x028	-	-	WTDIVEN[B,H,W] ------00	WTDIV [B,H,W] ----0000
0x02C	-	-	-	WTCALPRD [B,H,W] --010011
0x030	-	-	-	WTCOSEL [B,H,W] -------0
0x034 - 0xFFC	-	-	-	-

1.32. Low-speed CR Prescaler

Base_Address : 0x4003_B000

Base_Address + Address	Register +3	Register +2	Register +1	Register +0
0x000	-	-	-	LCR_PRSLD[B,H,W] --010011
0x004 - 0xFFC	-	-	-	-

1.33. EXT-Bus I/F

Base_Address : 0x4003_F000

Base_Address + Address	Register +3	+2	+1	+0
0x000	MODE0[W] -------- -------- --000-00 00000000			
0x004	MODE1[W] -------- -------- --000-00 00000000			
0x008	MODE2[W] -------- -------- --000-00 00000000			
0x00C	MODE3[W] -------- -------- --000-00 00000000			
0x010	MODE4[W] -------- -------- --000-00 00000001			
0x014	MODE5[W] -------- -------- --000-00 00000000			
0x018	MODE6[W] -------- -------- --000-00 00000000			
0x01C	MODE7[W] -------- -------- --000-00 00000000			
0x020	TIM0[W] 00000101 01011111 11110000 00001111			
0x024	TIM1[W] 00000101 01011111 11110000 00001111			
0x028	TIM2[W] 00000101 01011111 11110000 00001111			
0x02C	TIM3[W] 00000101 01011111 11110000 00001111			
0x030	TIM4[W] 00000101 01011111 11110000 00001111			
0x034	TIM5[W] 00000101 01011111 11110000 00001111			
0x038	TIM6[W] 00000101 01011111 11110000 00001111			
0x03C	TIM7[W] 00000101 01011111 11110000 00001111			

Base_Address + Address	Register +3	+2	+1	+0
0x040	AREA0[W] -------- -0001111 -------- 00000000			
0x044	AREA1[W] -------- -0001111 -------- 00010000			
0x048	AREA2[W] -------- -0001111 -------- 00100000			
0x04C	AREA3[W] -------- -0001111 -------- 00110000			
0x050	AREA4[W] -------- -0001111 -------- 01000000			
0x054	AREA5[W] -------- -0001111 -------- 01010000			
0x058	AREA6[W] -------- -0001111 -------- 01100000			
0x05C	AREA7[W] -------- -0001111 -------- 01110000			
0x060	ATIM0[W] -------- -------- ----0100 01011111			
0x064	ATIM1[W] -------- -------- ----0100 01011111			
0x068	ATIM2[W] -------- -------- ----0100 01011111			
0x06C	ATIM3[W] -------- -------- ----0100 01011111			
0x070	ATIM4[W] -------- -------- ----0100 01011111			
0x074	ATIM5[W] -------- -------- ----0100 01011111			
0x078	ATIM6[W] -------- -------- ----0100 01011111			
0x07C	ATIM7[W] -------- -------- ----0100 01011111			
0x080 - 0x2FC	-	-	-	-
0x300	DCLKR[W] -------- -------- -------- ---00001			
0x304 - 0x3FC	-	-	-	-

1.34. USB

ch.0 Base_Address : 0x4004_2100
ch.1 Base_Address : 0x4005_2100

Base_Address + Address	Register +3	Register +2	Register +1	Register +0
0x000	-	-	HCNT1[B,H,W] -----001	HCNT0[B,H,W] 00000000
0x004	-	-	HERR[B,H,W] 00000011	HIRQ[B,H,W] 0-000000
0x008	-	-	HFCOMP[B,H,W] 00000000	HSTATE[B,H,W] --010010
0x00C	-	-	HRTIMER(1/0)[B,H,W] 00000000 00000000	
0x010	-	-	HADR[B,H,W] -0000000	HRTIMER(2)[B,H,W] -----00
0x014	-	-	HEOF(1/0)[B,H,W] --000000 00000000	
0x018	-	-	HFRAME(1/0)[B,H,W] -----000 00000000	
0x01C	-	-	-	HTOKEN [B,H,W] 00000000
0x020	-	-	UDCC[B,H,W] -------- 10100-00	
0x024	-	-	EP0C[H,W] ------0- -1000000	
0x028	-	-	EP1C[H,W] 01100001 00000000	
0x02C	-	-	EP2C[H,W] 0110000- -1000000	
0x030	-	-	EP3C[H,W] 0110000- -1000000	
0x034	-	-	EP4C[H,W] 0110000- -1000000	
0x038	-	-	EP5C[H,W] 0110000- -1000000	
0x03C	-	-	TMSP[H,W] -----000 00000000	

A. Register Map

Base_Address	Register			
+ Address	+3	+2	+1	+0
0x040	-	-	UDCIE[B,H,W] --000000	UDCS[B,H,W] --000000
0x044	-	-	EP0IS[H,W] 10---1-- --------	
0x048	-	-	EP0OS[H,W] 100--00- -XXXXXXX	
0x04C	-	-	EP1S[H,W] 100-000X XXXXXXXX	
0x050	-	-	EP2S[H,W] 100-000- -XXXXXXX	
0x054	-	-	EP3S[H,W] 100-000- -XXXXXXX	
0x058	-	-	EP4S[H,W] 100-000- -XXXXXXX	
0x05C	-	-	EP5S[H,W] 100-000- -XXXXXXX	
0x060	-	-	EP0DTH [B,H,W] XXXXXXXX	EP0DTL [B,H,W] XXXXXXXX
0x064	-	-	EP1DTH [B,H,W] XXXXXXXX	EP1DTL [B,H,W] XXXXXXXX
0x068	-	-	EP2DTH [B,H,W] XXXXXXXX	EP2DTL [B,H,W] XXXXXXXX
0x06C	-	-	EP3DTH [B,H,W] XXXXXXXX	EP3DTL [B,H,W] XXXXXXXX
0x070	-	-	EP4DTH [B,H,W] XXXXXXXX	EP4DTL [B,H,W] XXXXXXXX
0x074	-	-	EP5DTH [B,H,W] XXXXXXXX	EP5DTL [B,H,W] XXXXXXXX
0x078 - 0x07C	-	-	-	-

1.35. DMAC

Base_Address : 0x4006_0000

Base_Address + Address	Register			
	+3	+2	+1	+0
0x000	DMACR[B,H,W] 00-00000 -------- -------- --------			
0x010	DMACA0[B,H,W] 00000000 0---0000 00000000 00000000			
0x014	DMACB0[B,H,W] --000000 00000000 00000000 -------0			
0x018	DMACSA0[B,H,W] 00000000 00000000 00000000 00000000			
0x01C	DMACDA0[B,H,W] 00000000 00000000 00000000 00000000			
0x020	DMACA1[B,H,W] 00000000 0---0000 00000000 00000000			
0x024	DMACB1[B,H,W] --000000 00000000 00000000 -------0			
0x028	DMACSA1[B,H,W] 00000000 00000000 00000000 00000000			
0x02C	DMACDA1[B,H,W] 00000000 00000000 00000000 00000000			
0x030	DMACA2[B,H,W] 00000000 0---0000 00000000 00000000			
0x034	DMACB2[B,H,W] --000000 00000000 00000000 -------0			
0x038	DMACSA2[B,H,W] 00000000 00000000 00000000 00000000			
0x03C	DMACDA2[B,H,W] 00000000 00000000 00000000 00000000			
0x040	DMACA3[B,H,W] 00000000 0---0000 00000000 00000000			
0x044	DMACB3[B,H,W] --000000 00000000 00000000 -------0			
0x048	DMACSA3[B,H,W] 00000000 00000000 00000000 00000000			
0x04C	DMACDA3[B,H,W] 00000000 00000000 00000000 00000000			

A. Register Map

Base_Address	Register			
+ Address	+3	+2	+1	+0
0x050	DMACA4[B,H,W] 00000000 0---0000 00000000 00000000			
0x054	DMACB4[B,H,W] --000000 00000000 00000000 -------0			
0x058	DMACSA4[B,H,W] 00000000 00000000 00000000 00000000			
0x05C	DMACDA4[B,H,W] 00000000 00000000 00000000 00000000			
0x060	DMACA5[B,H,W] 00000000 0---0000 00000000 00000000			
0x064	DMACB5[B,H,W] --000000 00000000 00000000 -------0			
0x068	DMACSA5[B,H,W] 00000000 00000000 00000000 00000000			
0x06C	DMACDA5[B,H,W] 00000000 00000000 00000000 00000000			
0x070	DMACA6[B,H,W] 00000000 0---0000 00000000 00000000			
0x074	DMACB6[B,H,W] --000000 00000000 00000000 -------0			
0x078	DMACSA6[B,H,W] 00000000 00000000 00000000 00000000			
0x07C	DMACDA6[B,H,W] 00000000 00000000 00000000 00000000			
0x080	DMACA7[B,H,W] 00000000 0---0000 00000000 00000000			
0x084	DMACB7[B,H,W] --000000 00000000 00000000 -------0			
0x088	DMACSA7[B,H,W] 00000000 00000000 00000000 00000000			
0x08C	DMACDA7[B,H,W] 00000000 00000000 00000000 00000000			
0x090 - 0x0FC	-	-	-	-

1.36. CAN

ch.0 Base_Address : 0x4006_2000
ch.1 Base_Address : 0x4006_3000

Base_Address + Address	Register +3	Register +2	Register +1	Register +0
0x000	STATR[B,H,W] -------- 00000000		CTRLR[B,H,W] -------- 000-0001	
0x004	BTR[B,H,W] -0100011 00000001		ERRCNT[B,H,W] 00000000 00000000	
0x008	TESTR[B,H,W] -------- X00000--		INTR[B,H,W] 00000000 00000000	
0x00C	-	-	BRPER[B,H,W] -------- ----0000	
0x010	IF1CMSK[B,H,W] -------- 00000000		IF1CREQ[B,H,W] 0------- 00000001	
0x014	IF1MSK2[B,H,W] 11-11111 11111111		IF1MSK1[B,H,W] 11111111 11111111	
0x018	IF1ARB2[B,H,W] 00000000 00000000		IF1ARB1[B,H,W] 00000000 00000000	
0x01C	-	-	IF1MCTR[B,H,W] 00000000 0---0000	
0x020	IF1DTA2[B,H,W] 00000000 00000000		IF1DTA1[B,H,W] 00000000 00000000	
0x024	IF1DTB2[B,H,W] 00000000 00000000		IF1DTB1[B,H,W] 00000000 00000000	
0x028 - 0x02F	-	-	-	-
0x030	IF1DTA1[B,H,W] 00000000 00000000		IF1DTA2[B,H,W] 00000000 00000000	
0x034	IF1DTB1[B,H,W] 00000000 00000000		IF1DTB2[B,H,W] 00000000 00000000	
0x038 - 0x03C	-	-	-	-
0x040	IF2CMSK[B,H,W] -------- 00000000		IF2CREQ[B,H,W] 0------- 00000001	
0x044	IF2MSK2[B,H,W] 11-11111 11111111		IF2MSK1[B,H,W] 11111111 11111111	

Base_Address + Address	Register +3	Register +2	Register +1	Register +0
0x048	IF2ARB2[B,H,W] 00000000 00000000		IF2ARB1[B,H,W] 00000000 00000000	
0x04C	-	-	IF2MCTR[B,H,W] 00000000 0---0000	
0x050	IF2DTA2[B,H,W] 00000000 00000000		IF2DTA1[B,H,W] 00000000 00000000	
0x054	IF2DTB2[B,H,W] 00000000 00000000		IF2DTB1[B,H,W] 00000000 00000000	
0x058 - 0x05C	-	-	-	-
0x060	IF2DTA1[B,H,W] 00000000 00000000		IF2DTA2[B,H,W] 00000000 00000000	
0x064	IF2DTB1[B,H,W] 00000000 00000000		IF2DTB2[B,H,W] 00000000 00000000	
0x068 - 0x07C	-	-	-	-
0x080	TREQR2[B,H,W] 00000000 00000000		TREQR1[B,H,W] 00000000 00000000	
0x084 – 0x08F	-	-	-	-
0x090	NEWDT2[B,H,W] 00000000 00000000		NEWDT1[B,H,W] 00000000 00000000	
0x094 - 0x09F	-	-	-	-
0x0A0	INTPND2[B,H,W] 00000000 00000000		INTPND1[B,H,W] 00000000 00000000	
0x0A4 - 0x0AF	-	-	-	-
0x0B0	MSGVAL2[B,H,W] 00000000 00000000		MSGVAL1[B,H,W] 00000000 00000000	
0x0B4 - 0xFFC	-	-	-	-

1.37. Ether-MAC

ch.0 Base_Address : 0x4006_4000

ch.1 Base_Address : 0x4006_7000

<Note>

For the register details of Ether-MAC block, refer to the "Ethernet Part".

1.38. Ether-Control

Base_Address : 0x4006_6000

<Note>

For the register details of Ether-Control block, refer to the "Ethernet Part".

1.39. WorkFlash_IF

Base_Address : 0x200E_0000

Base_Address	Register			
+ Address	+3	+2	+1	+0
0x000	WFASZR[B,H,W]			
0x004	WFRWTR[B,H,W]			
0x008	WFSTR[B,H,W]			
0x00C - 0xFFF	-	-	-	-

<Note>

For the register details of Workflash IF block, refer to the "Flash Programming Manual" of the product used.

B. List of Notes

This section explains notes for each function.

1. Notes when high-speed CR is used for the master clock

CODE: 9BPRECAUTION-E01.3

1. Notes when high-speed CR is used for the master clock

This section explains notes when the high-speed CR is used for the master clock.

The frequency of the high-speed CR varies depending on the temperature and/or the power supply voltage.
The following table shows notes on each function macro when the high-speed CR is used for the master clock.
Furthermore, pay attention to notes when the high-speed CR is used as an input clock of the PLL and the master clock is selected for PLL.

● Notes on Each Macro

Macro	Function/mode	Notes
Base Clock	HCLK/FCLK	The maximum frequency of the high-speed CR shall not exceed the upper limit of the internal operation clock frequency specified in the "Data Sheet" of the product used.
Timer	Multi-function Timer Base Timer Watch Timer Dual Timer Watch Dog Timer Quadrature	The frequency variation of the high-speed CR should be considered for the timer count value of each macro.
A/D Converter	Sampling Time Compare Tim	Considering the frequency variation of the high-speed CR, the sampling time and the compare time of the A/D converter shall satisfy the specification specified in the "Data Sheet" of the product used.
USB Ethernet-MAC CAN	-	As the frequency accuracy does not meet the required specification, these macros cannot be used when the high-speed CR is used for the master clock.
Multi Function Serial Interface	UART	Even if the frequency of the high-speed CR is the minimum or the maximum value, the baud rate error should be considered. The baud rate error shall not exceed the limit.
	CSIO I2C	The frequency variation of the high-speed CR should be considered for the communication of each macro.
	LIN	As the required frequency accuracy cannot be met, this function cannot be used as master. As slave, this function can be used. As a slave, the specified baud rate has more error at the maximum/minimum frequency of high-speed clock. So, if the error limit of the baud rate is exceeded, this function cannot be used.
Debug Interface	Serial Wire	As the frequency variation of the high-speed CR, the SWV(Serial Wire View) may not be used.
Flash Memory	Serial Write	The serial write cannot be supported for TYPE0, TYPE1, TYPE2, and TYPE4 products When the serial write is required, the clock should be supplied to the X0/X1pins.
External Bus Interface	Clock Output	When the external bus clock output is used, the frequency variation of the high-speed CR should be considered for devices to be connected.

C. List of Limitations

This section shows the differences between series.

CODE: 9BLIMITIONS-E02.0

1. List of Limitations for TYPE0 Products

This section shows the differences in the MB9A100A Series, MB9B500A/400A/300A/100A Series, MB9A100 Series and MB9B500/400/300/100 Series in a table.

The "Items" in the table are as written in this manual.

Item	Details
Timer Part 1.6.7 Hardware Watchdog Timer Load Register (WDG_LDR)	Following restrictions should be added to the <Notes> of "6.7. Hardware Watchdog Timer Load Register". • If a value is written to WDG_LDR again during the reloading period of the Hardware watchdog timer * (low-speed CR 4 cycle period after reloading the counter), the writing operation is ignored. Read the software of the appropriate register to check whether the writing value have been reflected to WDG_LDR properly. * The condition of counter reloading 1. Clearing watchdog timer (Writing a value to WDG_ICL register) 2. Writing a value to WDG_LDR register
Timer Part 1.6.9 Hardware Watchdog Timer Control Register (WDG_CTL)	Following restrictions should be added to the <Notes> of "6.9. Hardware Watchdog Timer Control Register". After writing "0" to the INTEN (watchdog counter enable) bit of the WDG_CTL register, if "1" is written again within 2 cycles of the low-speed CR (50KHz to 150KHz), operation may resume without reloading the count value from WDG_LDR. When setting the INTEN bit to "1" again after setting it to "0", always ensure a period of 2 clock cycles of the low-speed CR before setting. Alternatively, clear the timer using the WDG_ICL register immediately after writing "1" to INTEN to execute a reload.
Timer Part 3-2 Watch Counter	Following restrictions should be added to "CHAPTER 3-2: Watch Counter". *These restrictions are only for MB9A100 Series and MB9B500/400/300/100 Series. In Sub timer mode or Low speed CR timer mode, when the watch counter with sub crystal oscillator is used, the count value would be delayed from the actual time at the returning from an interrupt, by lengthening the interval of the low speed CR×35 cycles (Typ 350μs) watch counter. In Sub sleep mode or Low speed CR sleep mode, the counter value is not delayed.

<table>
<tr><th>Item</th><th>Details</th></tr>
<tr><td>Analog Macro Part
1-3.5.13
Sampling Time Selection Register (ADSS)</td><td>Following restrictions should be added to "5.13. Sampling Time Selection Register".

In this series, the sampling time set in the Sampling Time Setup Register (ADST1) cannot be used.
Enable the sampling time set in the Sampling Time Setup Register (ADST0) only.
Always write "0" to each bit of the Sampling Time Selection Register (ADSS0 to ADSS3).</td></tr>
<tr><td>Communication Macro Part
1-2.7.9
1-3.5.9
1-4.6.9
1-5.5.12
FIFO Byte Register (FBYTE)</td><td>Following notes should be added to
"7.9. FIFO Byte Register (FBYTE)" in chapter 1-2,
"5.9. FIFO Byte Register (FBYTE)" in chapter 1-3,
"6.9. FIFO Byte Register (FBYTE)" in chapter 1-4,
"5.12. FIFO Byte Register (FBYTE)" in chapter 1-5.

· If all the following conditions are met, the receive data full flag (SSR:RDRF) is not set to "1" despite the valid data of the number of FBYTE settings in the receive FIFO. If the setting value of FBYTE is "2" or more, this operation is not applied.
· The setting value of FBYTE is "1".
· Both the number of valid data of receive FIFO and the number of FBYTE settings are "1".
· The data in receive FIFO is read at the same time when the multi-function serial interface macro receives the data and the received data is written to receive FIFO.
However, in case that one of the followings occurs later, the receive data full flag (SSR:RDRF) is set to "1".
· Next data is received.
· The receive time idle of 8-bit time or more is detected when the receive FIFO idle is enabled (FCR:FRIIE=1).</td></tr>
<tr><td>Communication Macro Part 3-1.2
■ End-point configuration of the USB device</td><td>Following notes should be added to "■End-point configuration of USB device".
USB device does not support ISO (isochronous transfer).
Only Comb1 of setting combinations is valid.</td></tr>
<tr><td>Communication Macro Part 3-1.3.6
DMA transfer function</td><td>Following restrictions should be added to "■ Automatic data size transfer mode".

In this series, if the IN direction Automatic data size transfer mode is used in the Short packet transfer, packet transfer may not start even after DMA transfer is finished.
In addition, it is prohibited to set USB as both the transfer source and transfer destination.

[Workaround]
Transfer data using CPU.</td></tr>
</table>

<table>
<tr><th>Item</th><th>Details</th></tr>
<tr><td>Communication Macro
Part 3-1.3.7
NULL transfer function

Communication Macro
Part 3-1.5.3
EP1 to 5 Status Registers
(EP1C to EP5C)</td><td>The following description should be added as the NULL transfer mode restriction.

In this series, NULL transfer may not start after DMA transfer, even in the NULL transfer mode. Use this mode under the setting of EP1C to EP5C:NULE = "0".

[Workaround]
To perform the NULL transfer, firstly set DMAE = "0" and clear the DRQ bit without writing the buffer data.
See Notes of [bit10] DRQ bit in "23-1.5.9 EP1 to 5 Status Registers (EP1S to EP5S)".</td></tr>
<tr><td>Communication Macro
Part 3-1.5.3
EP1 to EP5 Control Register
(EP1C to EP5C)</td><td>[bit 14:13] TYPE: The following end-point transfer types are supported.
<table>
<tr><th>TYPE</th><th>Operation mode</th></tr>
<tr><td>00</td><td>Setting is prohibited</td></tr>
<tr><td>01</td><td>Setting is prohibited</td></tr>
<tr><td>10</td><td>Bulk transfer</td></tr>
<tr><td>11</td><td>Interrupt transfer</td></tr>
</table></td></tr>
<tr><td>Communication Macro
Part 3-1.5.10
EP0 to EP5 Data Registers
(EP0DTH to EP5DTH/
EP0DTL to EP5DTL)</td><td>Following restrictions should be added to "5.10. EP0 to EP5 Data Registers".

In this series, an indefinite data is read if serial read access to the above register is performed on the AHB bus.

[Workaround]
Please make the software to prevent the serial read. In the programming using C language, unintended serial read access on AHB bus may occur because of the optimization by the compiler option etc. Please refer to "■ Reference 1" for the workaround.</td></tr>
</table>

2. List of Limitations for TYPE1 Products

This section shows the differences in the MB9A002 Series, MB9A310 Series, MB9A110 Series, in a table.

The "Items" in the table are as written in this manual.

Item	Details
Communication Macro Part 1-2.7.9 1-3.5.9 1-4.6.9 1-5.5.12 FIFO Byte Register (FBYTE)	Following notes should be added to "7.9. FIFO Byte Register (FBYTE)" in chapter 1-2, "5.9. FIFO Byte Register (FBYTE)" in chapter 1-3, "6.9. FIFO Byte Register (FBYTE)" in chapter 1-4, "5.12. FIFO Byte Register (FBYTE)" in chapter 1-5. · If all the following conditions are met, the receive data full flag (SSR:RDRF) is not set to "1" despite the valid data of number of FBYTE settings in the receive FIFO. If the setting value of FBYTE is "2" or more, this operation is not applied. · The setting value of FBYTE is "1". · Both the number of valid data of receive FIFO and the number of FBYTE settings are "1" · The data in receive FIFO is read at the same time when the multi-function serial interface macro receives the data and the received data is written to receive FIFO. However, in case that one of the followings occurs later, the receive data full flag (SSR:RDRF) is set to "1". · Next data is received. · The receive time idle of 8-bit time or more is detected when the receive FIFO idle is enabled (FCR:FRIIE=1).

■ Reference 1

Example: If the following C source codes are compiled, serial read access may occur because of the optimization by the compiler option etc.

```
void do_ep0o(void)
{
        int i;
        int length;
        unsigned int b0,b1,b2,b3;

        b0 = (unsigned int)IO_EP0DT;
        b1 = (unsigned int)IO_EP0DT;
        b2 = (unsigned int)IO_EP0DT;
        b3 = (unsigned int)IO_EP0DT;
        buffer[0] = (unsigned short)b0;
        buffer[1] = (unsigned short)b1;
        buffer[2] = (unsigned short)b2;
        buffer[3] = (unsigned short)b3;
}
```

The following is a workaround. (Execute processing in the following order)

```
void do_ep0o(void)
{
        int i;
        int length;
        volatile int b0;

        b0 = (unsigned int)IO_EP0DT;
        buffer[0] = (unsigned short)b0;
        b0 = (unsigned int)IO_EP0DT;
        buffer[1] = (unsigned short)b0;
        b0 = (unsigned int)IO_EP0DT;
        buffer[2] = (unsigned short)b0;
        b0 = (unsigned int)IO_EP0DT;
        buffer[3] = (unsigned short)b0;
}
```

D. Product TYPE List

This section describes the product TYPE.

1. Product TYPE List

CODE: 9xTYPE_LIST-E04.0

1. Product TYPE List

In this manual, the products are classified into the following groups and are described as follows.
For the descriptions such as "TYPE0", see the relevant items of the target product in the list below.

Table 1 TYPE0 product list

Description in this manual	Flash memory size			
	512 Kbytes	384 Kbytes	256 Kbytes	128 Kbytes
TYPE0	MB9BF506N MB9BF506R MB9BF506NA MB9BF506RA MB9BF506NB MB9BF506RB	MB9BF505N MB9BF505R MB9BF505NA MB9BF505RA MB9BF505NB MB9BF505RB	MB9BF504N MB9BF504R MB9BF504NA MB9BF504RA MB9BF504NB MB9BF504RB	-
	MB9BF406N MB9BF406R MB9BF406NA MB9BF406RA	MB9BF405N MB9BF405R MB9BF405NA MB9BF405RA	MB9BF404N MB9BF404R MB9BF404NA MB9BF404RA	-
	MB9BF306N MB9BF306R MB9BF306NA MB9BF306RA MB9BF306NB MB9BF306RB	MB9BF305N MB9BF305R MB9BF305NA MB9BF305RA MB9BF305NB MB9BF305RB	MB9BF304N MB9BF304R MB9BF304NA MB9BF304RA MB9BF304NB MB9BF304RB	-
	MB9BF106N MB9BF106R MB9BF106NA MB9BF106RA	MB9BF105N MB9BF105R MB9BF105NA MB9BF105RA	MB9BF104N MB9BF104R MB9BF104NA MB9BF104RA	MB9BF102N MB9BF102R MB9BF102NA MB9BF102RA
	-	MB9AF105N MB9AF105R MB9AF105NA MB9AF105RA	MB9AF104N MB9AF104R MB9AF104NA MB9AF104RA	MB9AF102N MB9AF102R MB9AF102NA MB9AF102RA

Table 2 TYPE1 product list

Description in this manual	Flash memory size				
	512 Kbytes	384 Kbytes	256 Kbytes	128 Kbytes	64 Kbytes
TYPE1	MB9AF316M MB9AF316N MB9AF316MA MB9AF316NA	MB9AF315M MB9AF315N MB9AF315MA MB9AF315NA	MB9AF314L MB9AF314M MB9AF314N MB9AF314LA MB9AF314MA MB9AF314NA	MB9AF312L MB9AF312M MB9AF312N MB9AF312LA MB9AF312MA MB9AF312NA	MB9AF311L MB9AF311M MB9AF311N MB9AF311LA MB9AF311MA MB9AF311NA
	MB9AF116M MB9AF116N MB9AF116MA MB9AF116NA	MB9AF115M MB9AF115N MB9AF115MA MB9AF115NA	MB9AF114L MB9AF114M MB9AF114N MB9AF114LA MB9AF114MA MB9AF114NA	MB9AF112L MB9AF112M MB9AF112N MB9AF112LA MB9AF112MA MB9AF112NA	MB9AF111L MB9AF111M MB9AF111N MB9AF111LA MB9AF111MA MB9AF111NA

Table 3 TYPE2 product list

Description in this manual	Flash memory size		
	1 Mbyte	768 Kbytes	512 Kbytes
TYPE2	MB9BFD18S MB9BFD18T	MB9BFD17S MB9BFD17T	MB9BFD16S MB9BFD16T
	MB9BF618S MB9BF618T	MB9BF617S MB9BF617T	MB9BF616S MB9BF616T
	MB9BF518S MB9BF518T	MB9BF517S MB9BF517T	MB9BF516S MB9BF516T
	MB9BF418S MB9BF418T	MB9BF417S MB9BF417T	MB9BF416S MB9BF416T
	MB9BF318S MB9BF318T	MB9BF317S MB9BF317T	MB9BF316S MB9BF316T
	MB9BF218S MB9BF218T	MB9BF217S MB9BF217T	MB9BF216S MB9BF216T
	MB9BF118S MB9BF118T	MB9BF117S MB9BF117T	MB9BF116S MB9BF116T

Table 4 TYPE3 product list

Description in this manual	Flash memory size	
	128 Kbytes	64 Kbytes
TYPE3	MB9AF132K MB9AF132L	MB9AF131K MB9AF131L
	MB9AF132KA MB9AF132LA	MB9AF131KA MB9AF131LA
	MB9AF132KB MB9AF132LB	MB9AF132KB MB9AF132LB

Table 5 TYPE4 product list

Description in this manual	Flash memory size			
	512 Kbytes	384 Kbytes	256 Kbytes	128 Kbytes
TYPE4	MB9BF516N MB9BF516R	MB9BF515N MB9BF515R	MB9BF514N MB9BF514R	MB9BF512N MB9BF512R
	MB9BF416N MB9BF416R	MB9BF415N MB9BF415R	MB9BF414N MB9BF414R	MB9BF412N MB9BF412R
	MB9BF316N MB9BF316R	MB9BF315N MB9BF315R	MB9BF314N MB9BF314R	MB9BF312N MB9BF312R
	MB9BF116N MB9BF116R	MB9BF115N MB9BF115R	MB9BF114N MB9BF114R	MB9BF112N MB9BF112R

Table 6 TYPE5 product list

Description in this manual	Flash memory size	
	128 Kbytes	64 Kbytes
TYPE5	MB9AF312K	MB9AF311K
	MB9AF112K	MB9AF111K

Table 7 TYPE6 product list

Description in this manual	Flash memory size		
	256 Kbytes	128 Kbytes	64 Kbytes
TYPE6	MB9AFB44L MB9AFB44M MB9AFB44N MB9AFB44LA MB9AFB44MA MB9AFB44NA MB9AFB44LB MB9AFB44MB MB9AFB44NB	MB9AFB42L MB9AFB42M MB9AFB42N MB9AFB42LA MB9AFB42MA MB9AFB42NA MB9AFB42LB MB9AFB42MB MB9AFB42NB	MB9AFB41L MB9AFB41M MB9AFB41N MB9AFB41LA MB9AFB41MA MB9AFB41NA MB9AFB41LB MB9AFB41MB MB9AFB41NB
	MB9AFA44L MB9AFA44M MB9AFA44N MB9AFA44LA MB9AFA44MA MB9AFA44NA MB9AFA44LB MB9AFA44MB MB9AFA44NB	MB9AFA42L MB9AFA42M MB9AFA42N MB9AFA42LA MB9AFA42MA MB9AFA42NA MB9AFA42LB MB9AFA42MB MB9AFA42NB	MB9AFA41L MB9AFA41M MB9AFA41N MB9AFA41LA MB9AFA41MA MB9AFA41NA MB9AFA41LB MB9AFA41MB MB9AFA41NB
	MB9AF344L MB9AF344M MB9AF344N MB9AF344LA MB9AF344MA MB9AF344NA MB9AF344LB MB9AF344MB MB9AF344NB	MB9AF342L MB9AF342M MB9AF342N MB9AF342LA MB9AF342MA MB9AF342NA MB9AF342LB MB9AF342MB MB9AF342NB	MB9AF341L MB9AF341M MB9AF341N MB9AF341LA MB9AF341MA MB9AF341NA MB9AF341LB MB9AF341MB MB9AF341NB
	MB9AF144L MB9AF144M MB9AF144N MB9AF144LA MB9AF144MA MB9AF144NA MB9AF144LB MB9AF144MB MB9AF144NB	MB9AF142L MB9AF142M MB9AF142N MB9AF142LA MB9AF142MA MB9AF142NA MB9AF142LB MB9AF142MB MB9AF142NB	MB9AF141L MB9AF141M MB9AF141N MB9AF141LA MB9AF141MA MB9AF141NA MB9AF141LB MB9AF141MB MB9AF141NB

Table 8 TYPE7 product list

Description in this manual	Flash memory size	
	128 Kbytes	64 Kbytes
TYPE7	MB9AFA32L MB9AFA32M MB9AFA32N	MB9AFA31L MB9AFA31M MB9AFA31N
	MB9AF132M MB9AF132N	MB9AF131M MB9AF131N
	MB9AFAA2L MB9AFAA2M MB9AFAA2N	MB9AFAA1L MB9AFAA1M MB9AFAA1N
	MB9AF1A2L MB9AF1A2M MB9AF1A2N	MB9AF1A1L MB9AF1A1M MB9AF1A1N

Table 9 TYPE8 product list

Description in this manual	Flash memory size		
	512 Kbytes	384 Kbytes	256 Kbytes
TYPE8	MB9AF156M MB9AF156N MB9AF156R MB9AF156MA MB9AF156NA MB9AF156RA MB9AF156MB MB9AF156NB MB9AF156RB	MB9AF155M MB9AF155N MB9AF155R MB9AF155MA MB9AF155NA MB9AF155RA MB9AF155MB MB9AF155NB MB9AF155RB	MB9AF154M MB9AF154N MB9AF154R MB9AF154MA MB9AF154NA MB9AF154RA MB9AF154MB MB9AF154NB MB9AF154RB

Table 10 TYPE9 product list

Description in this manual	Flash memory size		
	256 Kbytes	128 Kbytes	64 Kbytes
TYPE9	MB9BF524K MB9BF524L MB9BF524M	MB9BF522K MB9BF522L MB9BF522M	MB9BF521K MB9BF521L MB9BF521M
	MB9BF324K MB9BF324L MB9BF324M	MB9BF322K MB9BF322L MB9BF322M	MB9BF321K MB9BF321L MB9BF321M
	MB9BF124K MB9BF124L MB9BF124M	MB9BF122K MB9BF122L MB9BF122M	MB9BF121K MB9BF121L MB9BF121M

Table 11 TYPE10 product list

Description in this manual	Flash memory size
	64 Kbytes
TYPE10	MB9BF121J

Table 12 TYPE11 product list

Description in this manual	Flash memory size
	64 Kbytes
TYPE11	MB9AF421K MB9AF421L
	MB9AF121K MB9AF121L

Table 13 TYPE12 product list

Description in this manual	Flash memory size	
	1.5 Mbytes	1 Mbytes
TYPE12	MB9BF529S MB9BF529T MB9BF529SA MB9BF529TA	MB9BF528S MB9BF528T MB9BF528SA MB9BF528TA
	MB9BF429S MB9BF429T MB9BF429SA MB9BF429TA	MB9BF428S MB9BF428T MB9BF428SA MB9BF428TA
	MB9BF329S MB9BF329T MB9BF329SA MB9BF329TA	MB9BF328S MB9BF328T MB9BF328SA MB9BF328TA
	MB9BF129S MB9BF129T MB9BF129SA MB9BF129TA	MB9BF128S MB9BF128T MB9BF128SA MB9BF128TA