32-bit RISC Microcontroller

TXZ/TXZ+ Family

Reference Manual
I²C Interface
(I2C-B)

Revision2.3

2023-06

TOSHIBA ELECTRONIC DEVICES & STORAGE CORPORATION

Contents

List of figures

List of Table

Preface

Related document

Document name
Product Information
Exception
Clock Control and Operation Mode

Conventions

- Numeric formats follow the rules as shown below:
  Hexadecimal: 0xABC
  Decimal: 123 or 0d123 – Only when it needs to be explicitly shown that they are decimal numbers.
  Binary: 0b111 – It is possible to omit the "0b" when the number of bit can be distinctly understood from a sentence.
- "_N" is added to the end of signal names to indicate low active signals.
- It is called "assert" that a signal moves to its active level, "deassert" to its inactive level.
- When two or more signal names are referred, they are described like as [m: n].
  Example: S[3: 0] shows four signal names S3, S2, S1 and S0 together.
- The characters surrounded by ***[ ]*** defines the register.
  Example: ***[ABCD]***
- "n" substitutes suffix number of two or more same kind of registers, fields, and bit names.
  Example: ***[XYZ1], [XYZ2], [XYZ3] → [XYZn]***
- "x" substitutes suffix number or character of units and channels in the Register List.

  In case of unit, "x" means A, B, and C . . .
  Example: ***[ADACR0], [ADBCR0], [ADCCR0] → [ADxCR0]***
  In case of channel, "x" means 0, 1, and 2 . . .
  Example: ***[T32A0RUNA], [T32A1RUNA], [T32A2RUNA] → [T32AxRUNA]***
- The bit range of a register is written like as [m: n].
  Example: Bit[3: 0] expresses the range of bit 3 to 0.
- The configuration value of a register is expressed by either the hexadecimal number or the binary number.
  Example: ***[ABCD]***<EFG> = 0x01 (hexadecimal), ***[XYZn]***<VW> = 1 (binary)
- Word and Byte represent the following bit length.
  Byte: 8 bits
  Half word: 16 bits
  Word: 32 bits
  Double word: 64 bits
- Properties of each bit in a register are expressed as follows:
  R: Read only
  W: Write only
  R/W: Read and Write are possible
- Unless otherwise specified, register access supports only word access.
- The register defined as reserved must not be rewritten. Moreover, do not use the read value.
- The value read from the bit having default value of "-" is unknown.
- When a register containing both of writable bits and read-only bits is written, read-only bits should be written with their default value, In the cases that default is "-", follow the definition of each register.
- Reserved bits of the Write-only register should be written with their default value. In the cases that default is "-", follow the definition of each register.
- Do not use read-modified-write processing to the register of a definition which is different by writing and read out.

Terms and Abbreviation

Some of abbreviations used in this document are as follows:

ANF	Analog Noise Filter
DNF	Digital Noise Filter
INT	Interrupt
I^2C	Inter-Integrated Circuit
I2CS	I^2C wakeup circuit from Stand-by mode
Fm	Fast-mode
Fm+	Fast-mode Plus
STD	Standard-mode

1. Outline

The I^2C can operate as a transceiver circuit of 1ch (SCL, SDA) in 1 unit circuit. The list of functions is shown below.

Function classification (Note1)	Function	A Functional Description or the range
Transmission speed Control	Prescaler dividing selection	It is dividing about a Prescaler clock to 1/1, 1/2, 1/3 to 1/30, 1/31 and 1/32.
	Clock source	A selection setup of the HIGH/LOW time of SCL is possible in master mode.
	The maximum transfer rate	1Mbps (it corresponds to Fm+) (fsys = 8 to 200 MHz)
Communication Format	I^2C bus format	Selection of Addressing/Data Free Format is possible. Selection of a master/slave is possible.
	Data length	1 to 8 bits
	acknowledge	The existence of acknowledging can be chosen.
	START/STOP condition	Generating of START/STOP condition is possible.
	Slave address	Only a 7-bit addressing format. 2 sets of slave addresses can be set up. (1st/2nd Slave Address)
	General call	Detection of a general call is possible in slave mode.
Transmission and reception Control	Arbitration	Multi-master Clock synchronization Existence selection of Arbitration lost detection is possible.
	Repetitive start detection, generating	Detection of a repetitive start of a bus line (at the time of slave mode) and generating (at the time of master mode) are possible.
	Noise cancellation	Digital
Ganged control	Interruption	4 kinds (The completion interruption of transmission, Arbitration lost detection interruption, Bus free detection interruption, NACK reception detection interruption
	DMA request	A setup according to transmission and reception is possible.
	Software reset	Reset by the software of an I^2C circuit is possible.
	Bus terminal state monitor function	The level monitor of SDA and SCL pin
	Address match Wakeup function	Slave address match detection can use the release factor for the Low power consumption mode release.

Note1: It does not support HS (High Speed) mode, 10-bit addressing, and a START byte.

Note2: There is a function in which it cannot support depending on products, such as slope control, I/O correspondence at the time of the power supply OFF, an Input voltage (VIH/VIL), and an Output voltage (VOL=0.4V, VDD>2V,3mA sink). Please refer to the "Electrical Characteristics" chapter of the Datasheet(DS) for details.

2. Configuration

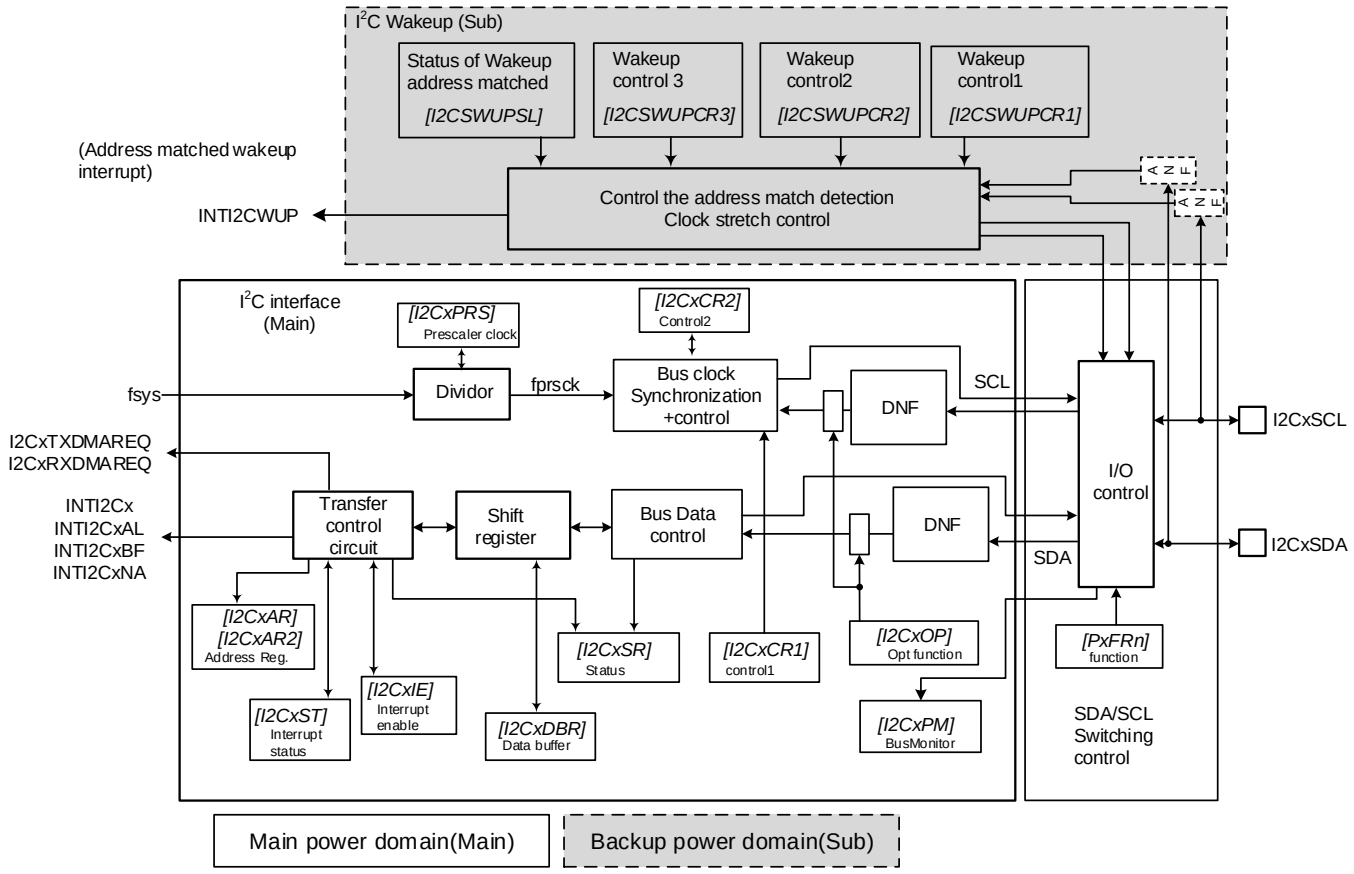


Figure 2.1 I²C interface block diagram

The circuit for the Wakeup is the expanded function. Analog NF (noise filter) and an expanded function,

Since it does not implement depend on the specification of a product, please see the datasheet of each product.

Table 2.1 List of Signals

No.	Symbol	Signal name	I/O	Related reference manual
1	fsys	System clock	Input	Clock Control and Operation Mode
2	I2CxSCL	SCL signal	Input/output	Product Information
3	I2CxSDA	SDA signal	Input/output	Product Information
4	INTI2Cx	I²C interruption	Output	Exception
5	INTI2CxAL	I²C arbitration lost detection interruption	Output	Exception
6	INTI2CxBF	I²C-bus free detection interruption	Output	Exception
7	INTI2CxNA	I²C NACK detection interruption	Output	Exception
8	INTI2CWUP	I²C Wakeup interruption	Output	Exception
9	I2CxTXDMAREQ	Transmitting DMA request	Output	Product Information
10	I2CxRXDMAREQ	Receiving DMA request	Output	Product Information

3. Details of a Function and operation

When using I²C, please set a clock enabling bit corresponding with the fsys supply on/off register A (***[CGFSYSENA], [CGFSYSMENA]***) or B (***[CGFSYSENB], [CGFSYSMENB]***) and fc supply on/off register (***[CGFCEN]***) as “1” (clock supply).

An applicable register and the bit position vary according to a product. Therefore, the register may not exist with the product. Please refer to "Clock Control and Operation Mode" of the reference manual for the details.

3.1. Configuration of the I²C bus

The I²C bus is connected to devices via the SDA and SCL pins and can communicate with multiple devices.

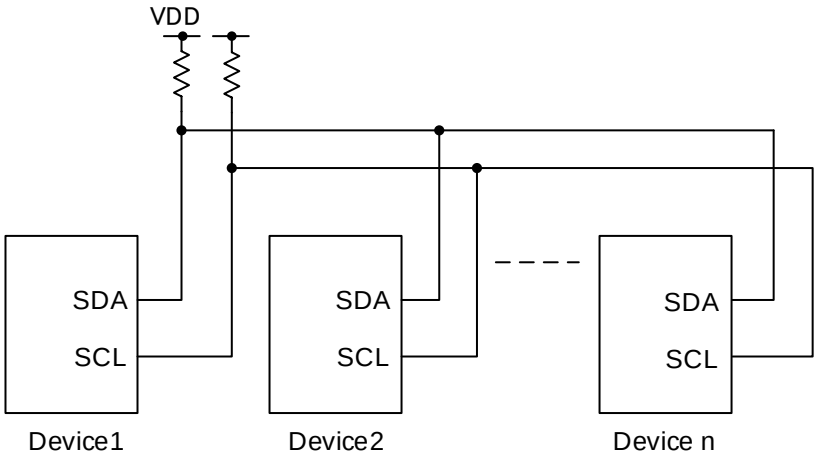


Figure 3.1 multiple devices in I²C

This module operates as a master or slave device on the I²C bus. The master device drives the serial clock line (SCL) of the bus, send 8-bit address, and sends or receives data of 1 to 8 bits.

The slave device receives 8-bit addresses and sends or receives serial data of 1 to 8 bits in synchronization with the serial clock on the bus.

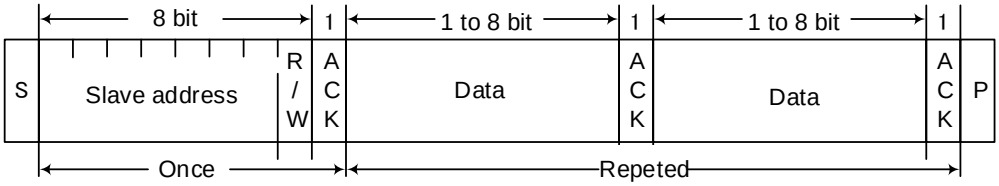
The device that operates as a receiver can output an acknowledge signal after reception of serial data and the device that operates as a transmitter can receive that acknowledge signal, regardless of whether the device is a master or slave. The master device can output a clock for the acknowledge signal.

In the multi master mode in which multiple masters exist on the same bus, serial clock synchronization and arbitration lost to maintain consistency of serial data are supported.

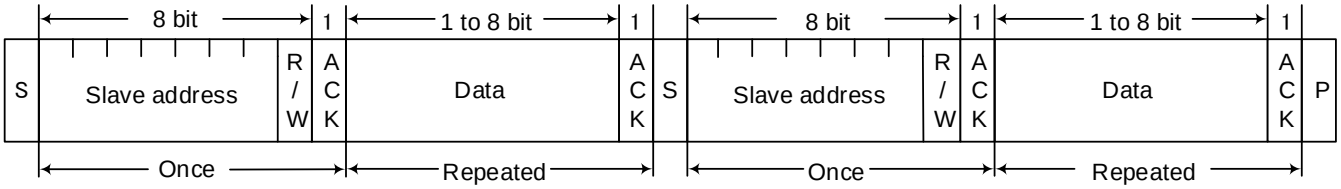
3.2. Data format

Below shows the data formats used in the I²C interface.

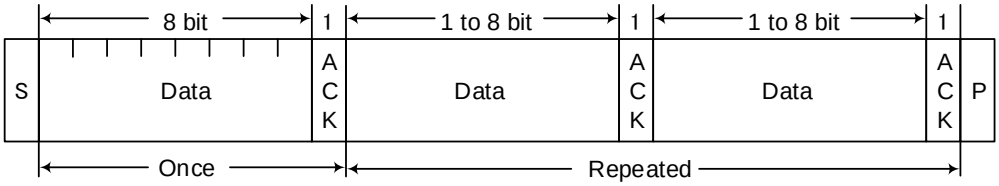
(a) Addressing format



(b) Addressing format(with repeated start condition)



(c) Free data format(master-transmitter to slave-receiver)



Note: S: START condition
R/W: Direction bit
ACK: Acknowledge bit
P: STOP condition

Figure 3.2 The data format of an I²C interface

3.3. Functional Description

3.3.1. Setting the clocks of Bits per Transfer and the Acknowledgement Mode

(1) The number of clocks of data transfer

The number of clocks of data transfer is set to ***[I2CxCR1]***<BC[2:0]> by ***[I2CxCR1]***<ACK>. For the relation of the number of clocks of data transfer, and ***[I2CxCR1]***<BC[2:0]> and ***[I2CxCR1]***<ACK>, refer to Table 3.1.

Setting ***[I2CxCR1]***<ACK> to "1" selects the acknowledge mode. In the acknowledgement mode, the Master device adds one clock for the acknowledgement after sending data's clocks and then require the INTI2Cx interrupt request.

The Slave device counts one clock for the acknowledgement after counting data's clocks and then require the INTI2Cx interrupt request.

By setting ***[I2CxCR1]***<ACK> to "0", the non-acknowledgement mode is activated.

In the non-acknowledgement mode, the Master device requires the INTI2Cx interrupt request after counting the Clocks for Data bits sending.

And the Slave devices require the INTI2Cx interrupt request after counting the data's clocks.

However, the second byte of the general call is necessary to be controlled by software to generate an ACK signal on the contents of the second byte. (For example, the second byte is received in non-acknowledge mode; in the receive interrupt routine, an ACK clock is output in a pseudo manner using 1-bit data output process.)

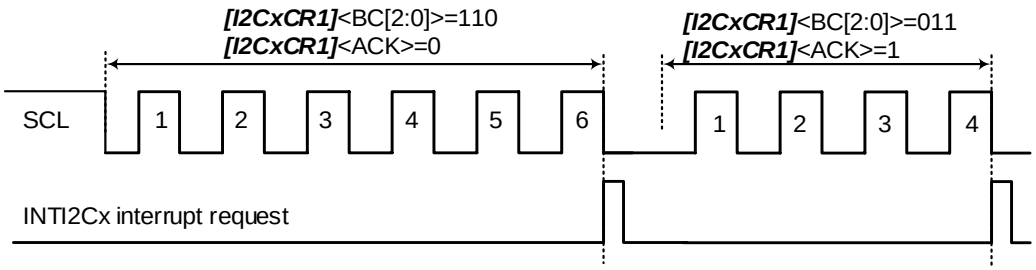


Figure 3.3 The number of data transfer clocks, *[I2CxCR1]*<BC[2:0]>, *[I2CxCR1]*<ACK>

"Table 3.1 The number of clocks of data transfer" shows the relationship between the number of clocks of data transfer, ***[I2CxCR1]***<BC[2:0]>, and ***[I2CxCR1]***<ACK>.

Table 3.1 The number of clocks of data transfer

[I2CxCR1]<BC[2:0]>	Acknowledge mode(***[I2CxCR1]***<ACK>)			
	0: No-ACK		1: ACK mode	
	Data length	The number of clocks	Data length	The number of clocks
000	8	8	8	9
001	1	1	1	2
010	2	2	2	3
011	3	3	3	4
100	4	4	4	5
101	5	5	5	6
110	6	6	6	7
111	7	7	7	8

<BC[2:0]> is cleared by "000" by a start condition or software reset.

Therefore, a slave address and the direction bit are always transferred in 8-bit length, or otherwise <BC[2:0]> maintains the specified value.

Note: Set ***[I2CxCR1]***<ACK> before transmitting or receiving a slave address. If ***[I2CxCR1]***<ACK> has been cleared, matching of slave addresses, and detection of a direction bit are not performed properly.

(2) Acknowledgement signal output

In Acknowledgement mode, the SDA pin is changing as follows during the clock cycle and generates acknowledgement signals.

– In master mode

In transmitter mode, the SDA pin should be opened until the transmitter receives an ACK signal from the receiver as acknowledgement.

In receive mode, the SDA pin should be pulled "LOW" to generate an ACK signal until the receiver sends an ACK signal to the transmitter when ***[I2CxOP]***<MFACK>=0. When ***[I2CxOP]***<MFACK>=1, the SDA pin should be pulled "HIGH".

– In slave mode

When the received slave address matches the slave address specified in ***[I2CxAR]***<SA>, or the general call is received, during the clock period for acknowledging, the SDA pin should be pulled "LOW" level and an acknowledge signal is generated.

In transmitter mode, data transfer after the received slave address matches the slave address or the general call is received, the SDA should be opened to receive an acknowledge signal from the receiver.

In receiver mode, the SDA pin should be pulled "LOW" level to generate an acknowledge signal.

Table 3.2 Comparison between the status of SCL and SDA in acknowledge mode

Mode	Pin	Conditions	Transmitter	Receiver
Master	SCL	-	Clocks are added for an acknowledge signal.	Clocks are added for an acknowledge signal.
	SDA	-	The SDA pin is opened to receive an acknowledge signal.	The SDA pin is pulled “LOW” for an acknowledge signal or SDA pin is pulled “HIGH” for a Non-acknowledge signal.
Slave	SCL	-	Checking and counting the clocks for an acknowledge signal.	Checking and counting the clocks for an acknowledge signal.
	SDA	A slave address matches the preset slave address, or the receiver receives a General call	-	The SDA pin is pulled “LOW” for an acknowledge signal.
		A slave address matches the preset slave address, or at the time of transfer after receipt of a general call	The SDA pin is opened to receive an acknowledge signal.	The SDA pin is pulled “LOW” for an acknowledge signal.

3.3.2. Serial clock

(1) Clock source

In master mode, it is set the HIGH and LOW period of a serial clock with ***[I2CxOP]***<NFSEL>,***[I2CxCR1]***<SCK>.

Table 3.3 Serial Clock HIGH&LOW time, Clock Rate(example)

[I2CxOP] <NFSEL>	*[I2CxCR1]* <SCK>	t_{HIGH} = (i x Tprsck)	t_{LOW} = (j x Tprsck)	Clock Rate (kHz)(Note 1)			
				fsys=40MHz <PRSCK>=2, pCLK=50(ns) Fm+ supported		fsys=80MHz <PRSCK>=5, pCLK=62.5(ns) Fm+ supported	
		i	j	$t_{HIGH}+t_{LOW}$ (ns)	f_{SCL}	$t_{HIGH}+t_{LOW}$ (ns)	f_{SCL}
0 (Digital)	000	8	12	-	Not used	500+750	800.00
	001	10	14	500+700	833.33	625+875	666.67
	010	14	18	700+900	625.00	875+1125	500.00
	011	22	26	1100+1300	416.67	1375+1625	333.33
	100	38	42	1900+2100	250.00	2375+2625	200.00
	101	70	74	3500+3700	138.89	4375+4625	111.11
	110	134	138	6700+6900	73.53	8375+8625	58.82
	111	262	266	13100+13300	37.88	16375+16625	30.30

Tprsck: Prescaler clock width (1/fprsck)

Note1: The maximum transfer speed range (Fm / Fm+ correspondence) may differ depending on the product (pin), Please check with the "Product Information" of the reference manual.

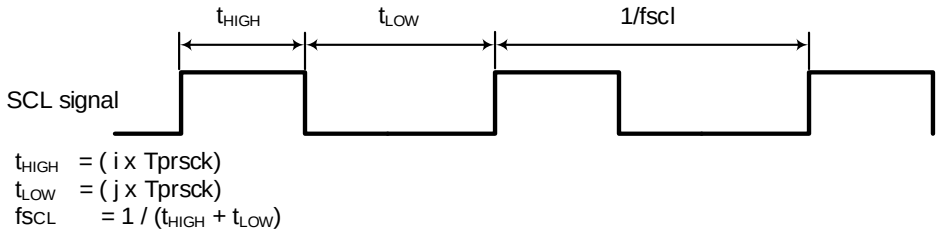


Figure 3.4 I²C SCL Output

Note: The t_{HIGH} period may differ from the specified value if the rising edge becomes blunt depending on the combination of bus load capacitance and pull-up register. If the clock synchronization function for synchronizing clocks from multiple clocks is used, the actual clock period may differ from the specified setting.

In master mode, the hold time when START conditions generated and the setup time when a STOP condition is generated are defined as the following.

Hold time($t_{HD:STA}$):

[I2CxOP]<SREN>=0: t_{HIGH} [s]

[I2CxOP]<SREN>=1: 8Tprsck [s]

Setup time($t_{SU;STO}$):

[I2CxPRS]<PRSCK>=1: t_{HIGH} [s]

[I2CxPRS]<PRSCK>≠1: t_{HIGH} - Tprsck [s]

Setup time of the Repeated START condition is as follows.

Setup time($t_{SU;STA}$)

[I2CxOP]<SREN>=0: Go and keep the time for I²C mode by software.

[I2CxOP]<SREN>=1: t_{LOW} [s]

When ***[I2CxCR2]***<PIN> is set to "1" in slave mode, the time to the release of SCL is defined as t_{LOW}[s].

In both master and slave modes, the high level period must be 4×Tprsck[s] or longer and the low level period must be 5×Tprsck[s] or longer for externally input serial clocks, regardless of the ***[I2CxCR1]***<SCK> setting.

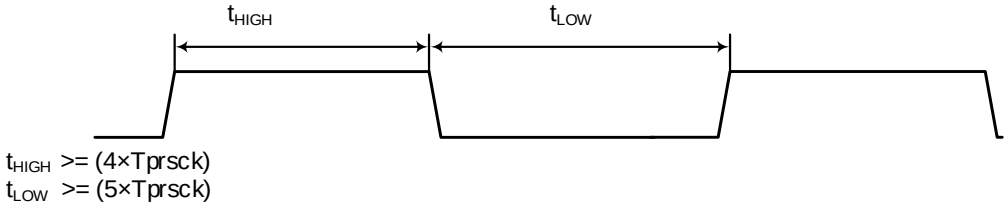


Figure 3.5 SCL input

The serial clock rate to be output from the master is set through ***[I2CxCR1]***<SCK[2:0]> and the ***[I2CxPRS]***<PRSCK[4:0]>.

The Prescaler clock which is divided according to ***[I2CxPRS]***<PRSK[4:0]> is used as the reference clock for generating the serial clock. The Prescaler clock is further divided according to ***[I2CxCR1]***<SCK[2:0]> and used as the serial clock.

<A serial transfer rate>

The serial clock rate (f_{SCL}) is determined by prescaler setting value "p" (***[I2CxPRS]***<PRSCK[4:0]>, p=1 to 32) and serial clock setting value "n" (***[I2CxR1]***<SCK[2:0]>, n=0 to 7) based on the operating frequency (fsys) as follows.

<NFSEL>=0:

$$\text{Serial clock rate: } f_{SCL}(\text{kHz}) = \frac{\text{fsys(MHz)}}{p \times (2^{n+2} + 16)} \times 1000$$

Notice

The allowed range is depended on the operating frequency (fsys).

The allowed range of Prescaler setting value "p" (***[I2CxPRS]***<PRSCK[4:0]>) varies depending on the operating frequency (fsys) and must satisfy the following condition.

Fast-mode Plus (Fm+), DNF using(<NFSEL>=0)

20ns < Prescaler clock width: Tprsck (ns) ≤ 65ns

Fast-mode (Fm), Standard-mode(STD), DNF using(<NFSEL>=0)

50ns < Prescaler clock width: Tprsck (ns) ≤ 150ns

Setting the Prescaler clock width out of this range is prohibited in both master and slave modes.

The serial clock rate may not be constant due to the clock synchronization function.

The usable fsys condition of Fm+/Fm, STD is as following.

Table 3.4 fsys condition with Fm+/Fm, STD

NF selection	Operation Mode	fsys(MHz)
DNF	Fm+	fsys ≥ 15.39
	Fm, STD	fsys ≥ 6.67

Typical serial clock setting table for each fsys frequency is shown in the table below for Fast-mode Plus (Fm +) and Fast-mode (Fm).Standard-mode (STD)(Table 3.5 to Table 3.12)

Table 3.5 Serial Clock setting table (DNF using, for Fm+)(1)

Unit: kHz

fsys=32.0MHz		SCK[2:0]							
PRSCK [4:0]	pCLK (ns)	000	001	010	011	100	101	110	111
00010	62.50	-	666.67	500.00	333.33	200.00	111.11	58.82	30.30

Table 3.6 Serial Clock setting table (DNF using, for Fm+)(2)

Unit: kHz

fsys=40.0MHz		SCK[2:0]							
PRSCK [4:0]	pCLK (ns)	000	001	010	011	100	101	110	111
00010	50.00	-	833.33	625.00	416.67	250.00	138.89	73.53	37.88

-: not used

Table 3.7 Serial Clock setting table (DNF using, for Fm+)(3)

Unit: kHz

fsys=80.0MHz		SCK[2:0]							
PRSCK [4:0]	pCLK (ns)	000	001	010	011	100	101	110	111
00011	37.50	-	-	833.33	555.56	333.33	185.19	98.04	50.51
00100	50.00	-	833.33	625.00	416.67	250.00	138.89	73.53	37.88
00101	62.50	800.00	666.67	500.00	333.33	200.00	111.11	58.82	30.30

-: not used

Table 3.8 Serial Clock setting table (DNF using, for Fm+)(4)

Unit: kHz

fsys=100.0MHz		SCK[2:0]							
PRSCK [4:0]	pCLK (ns)	000	001	010	011	100	101	110	111
00100	40.00	-	-	781.25	520.83	312.50	173.61	91.91	47.35
00101	50.00	-	833.33	625.00	416.67	250.00	138.89	73.53	37.88
00110	60.00	833.33	694.44	520.83	347.22	208.33	115.74	61.27	31.57

-: not used

Table 3.9 Serial Clock setting table (DNF using, for Fm/STD)(1)

Unit: kHz

fsys=20.0MHz		SCK[2:0]							
PRSCK [4:0]	pCLK (ns)	000	001	010	011	100	101	110	111
00010	100.00	-	-	312.50	208.33	125.00	69.44	36.76	18.94
00011	150.00	333.33	277.78	208.33	138.89	83.33	46.30	24.51	12.63

-: not used

Table 3.10 Serial Clock setting table (DNF using, for Fm/STD)(2)

Unit: kHz

fsys=40.0MHz		SCK[2:0]							
PRSCK [4:0]	pCLK (ns)	000	001	010	011	100	101	110	111
00100	100.00	-	-	312.50	208.33	125.00	69.44	36.76	18.94
00101	125.00	-	333.33	250.00	166.67	100.00	55.56	29.41	15.15
00110	150.00	333.33	277.78	208.33	138.89	83.33	46.30	24.51	12.63

-: not used

Table 3.11 Serial Clock setting table (DNF using, for Fm/STD)(3)

Unit: kHz

fsys=80.0MHz		SCK[2:0]							
PRSCK [4:0]	pCLK (ns)	000	001	010	011	100	101	110	111
00111	87.50	-	-	-	238.10	142.86	79.37	42.02	21.65
01000	100.00	-	-	312.50	208.33	125.00	69.44	36.76	18.94
01001	112.50	-	-	277.78	185.19	111.11	61.73	32.68	16.84
01010	125.00	-	333.33	250.00	166.67	100.00	55.56	29.41	15.15
01011	137.50	363.64	303.03	227.27	151.52	90.91	50.51	26.74	13.77
01100	150.00	333.33	277.78	208.33	138.89	83.33	46.30	24.51	12.63

-: not used

Table 3.12 Serial Clock setting table (DNF using, for Fm/STD)(4)

Unit: kHz

fsys=100.0MHz		SCK[2:0]							
PRSCK [4:0]	pCLK (ns)	000	001	010	011	100	101	110	111
01000	80.00	-	-	-	260.42	156.25	86.81	45.96	23.67
01001	90.00	-	-	347.22	231.48	138.89	77.16	40.85	21.04
01010	100.00	-	-	312.50	208.33	125.00	69.44	36.76	18.94
01011	110.00	-	-	284.09	189.39	113.64	63.13	33.42	17.22
01100	120.00	-	347.22	260.42	173.61	104.17	57.87	30.64	15.78
01101	130.00	-	320.51	240.38	160.26	96.15	53.42	28.28	14.57
01110	140.00	357.14	297.62	223.21	148.81	89.29	49.60	26.26	13.53
01111	150.00	333.33	277.78	208.33	138.89	83.33	46.30	24.51	12.63

-: not used

(2) Clock synchronization

The I²C bus is driven by using the wired-AND connection due to its pin structure. The first master that pulls its clock line to the "LOW" level overrides other masters producing the "HIGH" level on their clock lines. This must be detected and responded by the masters producing the "HIGH" level.

I²C has a clock synchronization function to ensure proper transfer operation even when multiple masters exist on a bus.

For example, the clock synchronization procedure for a bus with two masters is shown below.

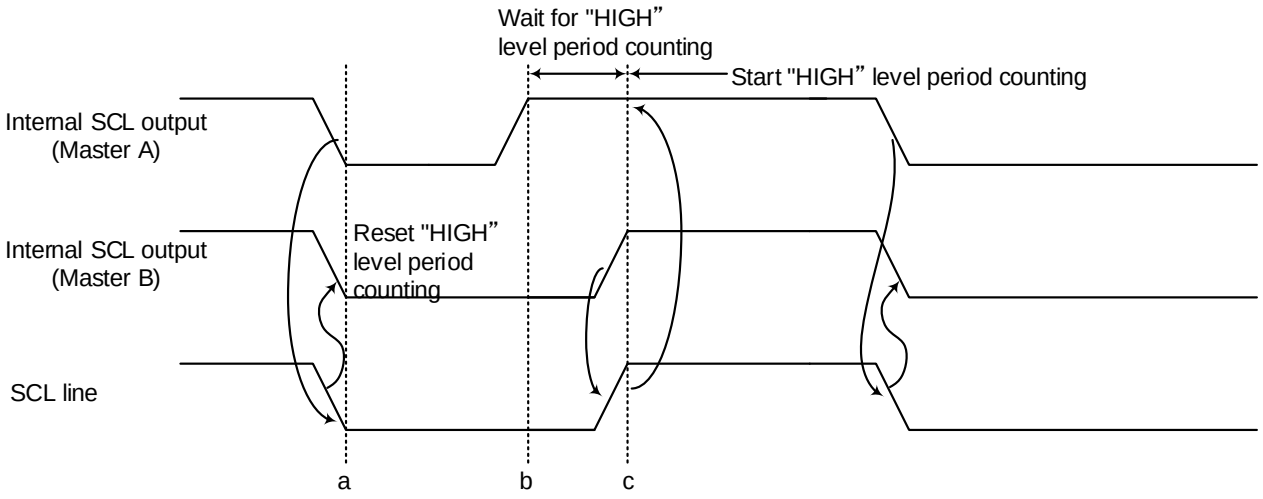


Figure 3.6 The example of clock synchronization

At the point a, Master A pulls its internal SCL output to the "LOW" level, bringing the SCL bus line to the "LOW" level. Master B detects this transition, resets its "HIGH" level period counter, and pulls its internal SCL output level to the "LOW" level.

Master A completes counting of its "LOW" level period at the point b, and brings its internal SCL output to the "HIGH" level. However, Master B still keeps the SCL bus line at the "LOW" level, and Master A stops counting of its "HIGH" level period counting. After Master A detects that Master B brings its internal SCL output to the "HIGH" level and brings the SCL bus line to the "HIGH" level at the point c, it starts counting of its "HIGH" level period.

After that Master finishes counting the "HIGH" level period, the Master pulls the SCL pin to "LOW" and the SCL bus line becomes "LOW".

This way, the clock on the bus is determined by the master with the shortest "HIGH" level period and the master with the longest "LOW" level period among those connected to the bus.

3.3.3. Master/Selection of a slave

Setting ***[I2CxCR2]***<MST> to "1" configures the I²C to operate as a master device.

Setting ***[I2CxCR2]***<MST> to "0" configures the I²C as a slave device.

[I2CxSR]<MST> is cleared to "0" by the hardware when it detects the stop condition on the bus or the arbitration lost.

3.3.3.1. Transmitter/ receiver selection

Setting ***[I2CxCR2]***<TRX> to "1" configures the I²C as a transmitter. Setting <TRX> to "0" configures the I²C as a receiver.

In the slave mode, when data is transmitted in the addressing format. If the value of the direction bit (R/W) is "1", ***[I2CxSR]***<TRX> is set to "1" by the hardware. If the bit is "0", ***[I2CxSR]***<TRX> is cleared to "0".

As a master device, the I²C receives acknowledgement from a slave device. If the direction bit of "1" is transmitted, ***[I2CxSR]***<TRX> is set to "0" by the hardware. If the direction bit is "0", ***[I2CxSR]***<TRX> changes to "1". If the I²C does not receive an acknowledgement, ***[I2CxSR]***<TRX> retains the previous value.

[I2CxSR]<TRX> is cleared to "0" by the hardware when it detects the stop condition on the bus or the arbitration lost.

"Table 3.13 Operation of *[I2CxSR]* <TRX> in each mode" shows each operation mode depends on the ***[I2CxSR]***<TRX> value.

Note: When ***[I2CxCR1]***<NOACK>=1, the slave address detection and general call detection are disabled, and thus ***[I2CxSR]***<TRX> remains unchanged.

Table 3.13 Operation of *[I2CxSR]* <TRX> in each mode

Mode	Direction bit	Condition for state change	TRX after change
Slave	0	The received slave address matches the value specified in <SA> (<SA2>).	0
	1		1
Master	0	ACK received	1
	1		0

When I²C is used in free data format, the slave address and direction bit are not recognized and bits immediately following a START condition are handled as data. Therefore, ***[I2CxSR]***<TRX> is not changed by the hardware.

3.3.4. Enabling of the I²C bus

When ***[I2CxCR2]***<I2CM> is set to "1". I²C bus mode is selected.

Ensure that the I²C bus interface pins are at "HIGH" level before setting ***[I2CxCR2]***<I2CM> to "1". Also, ensure that the bus is free before switching the operating mode to the port mode.

Note: When ***[I2CxCR2]***<I2CM> = 0, no value can be written to ***[I2CxCR2]***<SWRES> for Software Reset and also bits in the ***[I2CxCR2]*** register other than ***[I2CxCR2]***<I2CM> bit.
Before setting ***[I2CxCR2]***, write "1" to ***[I2CxCR2]***<I2CM> to select the I²C bus mode.

3.3.5. Generating START and STOP Conditions

When ***[I2CxSR]***<BB> is "0", setting "1" to ***[I2CxCR2]***<MST>, ***[I2CxCR2]*** <TRX>, ***[I2CxCR2]***<BB>, ***[I2CxCR2]***<PIN> causes the I²C to start a sequence for generating the START condition and to output the slave address and the direction bit prospectively written in the data buffer register. ***[I2CxCR1]***<ACK> must be set to "1" in advance.

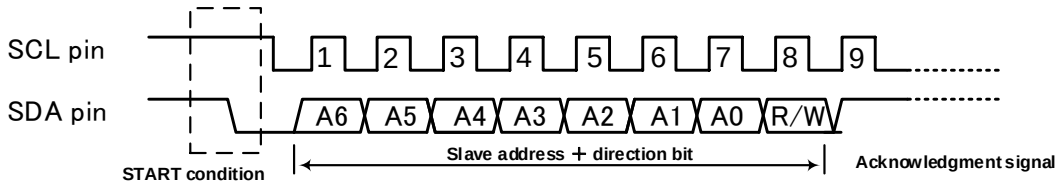


Figure 3.7 Generating of a START condition and a slave address

When ***[I2CxSR]***<BB> is "1", setting "1" to ***[I2CxCR2]***<MST>, ***[I2CxCR2]***<TRX, ***[I2CxCR2]***<PIN> and "0" to ***[I2CxCR2]***<BB> causes the I²C to start a sequence for generating the stop condition on the bus. The contents of <MST, TRX, BB, PIN> should not be altered until the STOP condition appears on the bus.

If the SCL bus line is pulled "LOW" by other devices when the stop condition is generated, the STOP condition is generated after the SCL line is released.

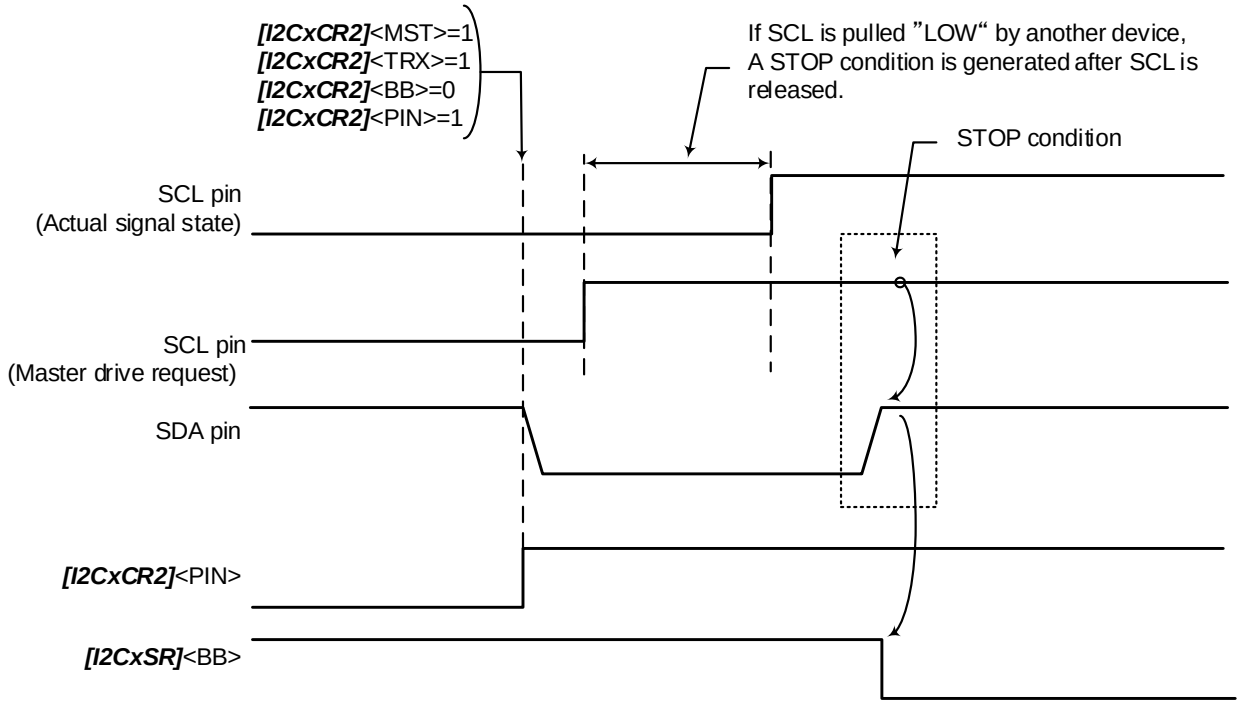


Figure 3.8 Generating of a STOP condition

[I2CxSR]<BB> can be read to check the bus state. ***[I2CxSR]***<BB> is set to "1" when the START condition is detected on the bus (the bus is busy), and cleared to "0" when the STOP condition is detected (the bus is free).

Table 3.14 shows typical setting examples according to the ***[I2CxSR]*** state.

Although the ***[I2CxCR2]***<MST>, <TRX>,<BB>,<PIN> bits are given independent functions, they are used in typical combinations, as shown below, according to the ***[I2CxSR]*** setting.

Table 3.14 The example of a setting

[I2CxSR]			*[I2CxCR2]*				Operation
[7] MST	[5] BB	[4] PIN	[7] MST	[6] TRX	[5] BB	[4] PIN	
0	0	1	0	0	0	0	Wait for a START condition as a slave
			1	1	1	1	Generate a START condition.
1	1	0	1	1	0	1	Generate a STOP condition.

Note: When writing to these bits, do not change ***[I2CxCR2]***<I2CM> by mistake.

3.3.6. Selection of slave address match detection and General call detection

For a slave device, when using the slave address match detection and General call detection in slave mode, the following setting is required.

The ***[I2CxCR1]***<NOACK> is set to enable or disable for a slave address match detection and General call detection in slave mode.

When ***[I2CxCR1]***<NOACK>=0 and ***[I2CxOP]***<GCDI>=0, it is enable for a slave address match detection and General call detection. If ***[I2CxOP]***<GCDI>=1, General call detection is disable.

When ***[I2CxCR1]***<NOACK>=1, it is disabled for a slave address match detection and General call detection.

The slave device ignores a slave address and general calls sent from the master and returns non-acknowledgement. The INTI2Cx interrupt request is not generated.

In master mode, the bit of ***[I2CxCR1]***<NOACK> is ignored and has no effect on operation.

3.3.7. General call detection monitor

In the I^2C bus mode (***[I2CxAR]***<ALS> = 0), when Slave mode is selected, a slave address and General call can be detected.

When ***[I2CxCR1]***<NOACK>=0 and ***[I2CxOP]***<GCDI>=0, if the device receives a general call (8 bits received immediately after a START condition are all zero), ***[I2CxSR]***<AD0> is set to "1". (At this time, ***[I2CxSR]***<AAS> is also set to "1".)

When ***[I2CxCR1]***<NOACK>=1, a General call cannot be detected; therefore, if a General call is received, ***[I2CxSR]***<AD0> remains "0". (At this time, ***[I2CxSR]***<AAS> also remains "0".)

[I2CxSR]<AD0> is cleared to "0" when the START or STOP condition is detected on the bus.

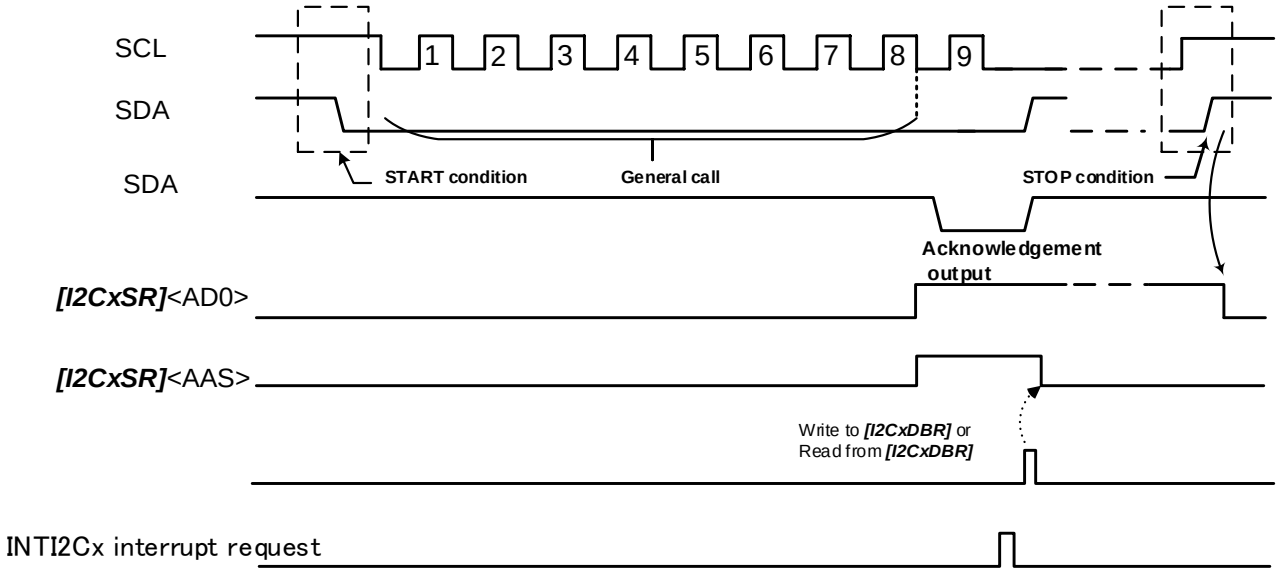


Figure 3.9 Change of a General call detection monitor

3.3.8. Setup of a slave address and address recognition mode

To use I²C bus mode, clear ***[I2CxAR]***<ALS> to "0" and set a slave address to ***[I2CxAR]***<SA>.

If free data format that does not use a slave address is used, set "1" to ***[I2CxAR]***<ALS>. Note that the I²C is used in free data format, a slave address and direction bit are not recognized. They are treated as data immediately after a START condition.

When the second slave address is used, set the value to ***[I2CxAR2]***<SA2EN>. When <SA2EN>=1, the second slave address is valid.

The received Slave addresses are compared with the first address (***[I2CxAR]***<SA>) and the second address (***[I2CxAR2]***<SA2>) sequentially. The result of a comparison is stored to ***[I2CxOP]***<SAST> and <SA2ST>.

When the same address is used for the first address and the second address, only the detection result of the first address is returned when the received slave address matches them.

3.3.9. Slave address Match detection monitor

Two slave addresses can be specified in this product. In I²C bus mode (***[I2CxAR]***<ALS> = 0), when Slave mode is selected, a match of slave addresses can be detected. Received slave addresses are compared with ***[I2CxAR]***<SA> and ***[I2CxAR2]***<SA2> sequentially. If the same address is used for ***[I2CxAR]***<SA> and ***[I2CxAR2]***<SA2>, only the comparison result with ***[I2CxAR]***<SA> is returned when the received slave address matches them. Note that if one slave address is used, use ***[I2CxAR]***<SA> only.

When ***[I2CxCR1]***<NOACK> is cleared to "0", the address match detection is enabled. If ***[I2CxSR]***<AAS> receives a General call or the slave address(1st Slave address) which is the same as the ***[I2CxAR]***<SA>, ***[I2CxSR]***<AAS> is set to "1".

To use the second slave address, set the value to ***[I2CxAR2]***<SA2> and ***[I2CxAR2***]<SA2EN>=1. Whether <SA> or <SA2>is used for match detection, this can be distinguish with ***[I2CxOP]***<SA2ST> or <SAST>.

When "1" is set to ***[I2CxCR1]***<NOACK>, address match detection is disabled. If this device receives a general call or the slave address which is the same as the ***[I2CxAR]***<SA> and the ***[I2CxAR]***<SA2>, ***[I2CxSR]***<AAS> is not set to "1".

When Free data format mode (***[I2CxAR]***<ALS> is set to "1"), ***[I2CxSR]***<AAS> is set to "1" when the first data word has been received.

[I2CxSR]<AAS> is cleared to "0" when data is written to or read from ***[I2CxDBR]***.

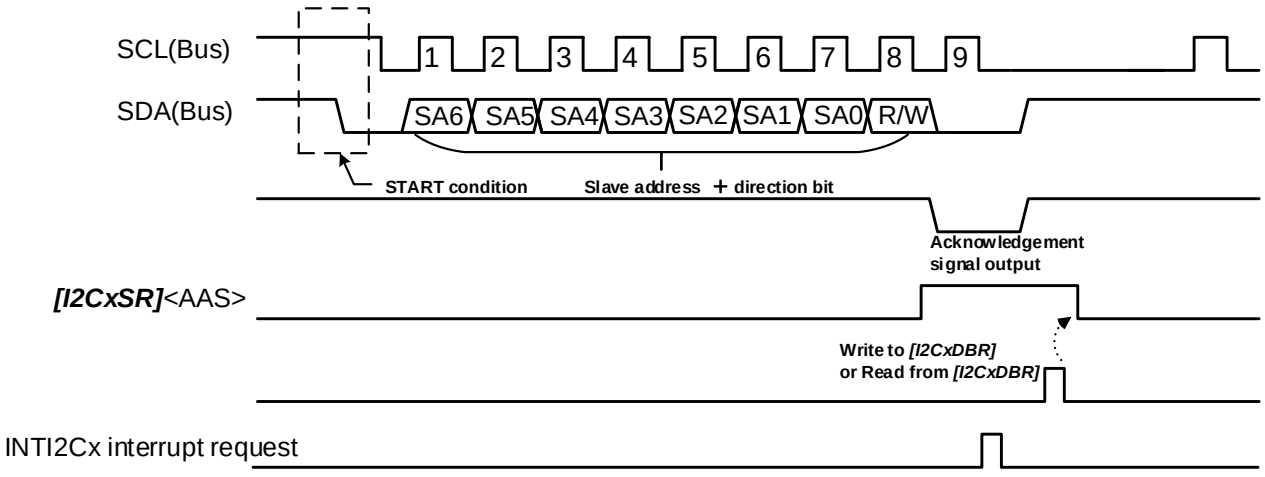


Figure 3.10 Change of a slave address Match detection monitor

3.3.10. Arbitration Lost detection monitor

The I^2C bus has the multi-master capability (there are two or more masters on a bus), and requires the bus arbitration procedure to ensure correct data transfer.

The I^2C-bus arbitration takes place on the SDA line.

The arbitration procedure for two masters on a bus is shown below.

Up until the point a, Master A and Master B output the same data. At the point a, Master A outputs the "LOW" level and Master B outputs the "HIGH" level.

Then Master A pulls the SDA bus line to the "LOW" level because the line has the wired-AND connection.

When the SCL line goes high at the point b, the slave device reads the SDA line data, i.e., data transmitted by Master A. At this time, data transmitted by Master B becomes invalid.

This condition of Master B is called "Arbitration Lost". Master B releases its SDA pin, so that it does not affect the data transfer initiated by another master. If two or more masters have transmitted exactly the same first data word, the arbitration procedure continues with the second data word

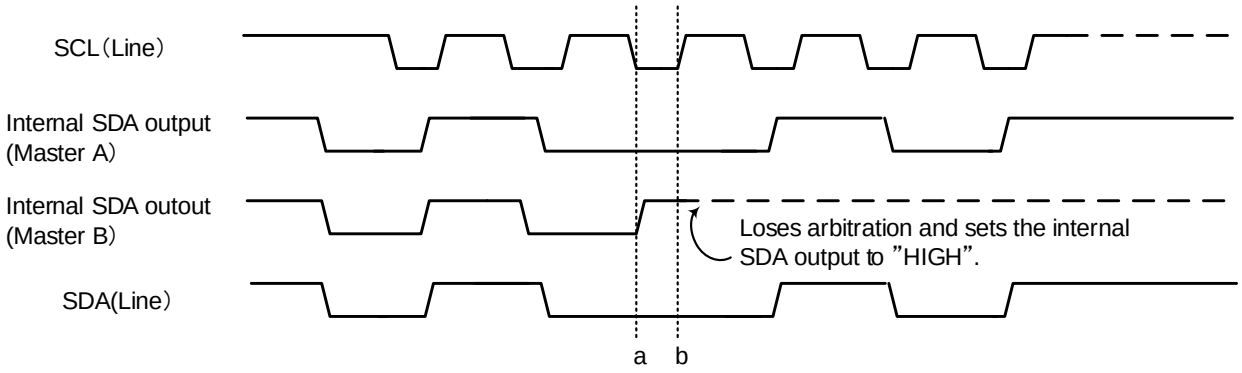


Figure 3.11 Arbitration Lost

If an Arbitration Lost interrupt is enabled, an ***[I2CxSR]***<AL> flag is set at the point “b” in Figure 3.11 . When subsequently data transfer is executed the number of preset clocks, an Arbitration Lost interrupt occurs.

A master compares the SDA bus line level and the internal SDA output level at the rising of the SCL line.

If there is a difference between these two values, Arbitration Lost occurs and ***[I2CxSR]***<AL> is set to "1".

When ***[I2CxSR]***<AL> is set to "1", ***[I2CxSR]***<MST, TRX> are cleared to "0", causing the I^2C to operate as a slave receiver.

Therefore, the serial bus interface circuit stops the clock output during data transfer after ***[I2CxSR]***<AL> is set to "1".

[I2CxSR]<PIN> is cleared to “0” when data transfer is completed. The I^2C pulls the SCL Line to the LOW level.

An Arbitration Lost occurs during the transfer of the slave addresses and direction bit in Master B. In this case, Master B receives a slave address sent from other master devices, like other slave devices. Regardless of the match between the received slave address and ***[I2CxAR]***<SA>, <PIN> is cleared to "0" and then INTI2Cx is generated.

[I2CxSR]<AL> is cleared to "0" when data is written to or read from ***[I2CxDBR]*** or data is written to ***[I2CxCR2]***.

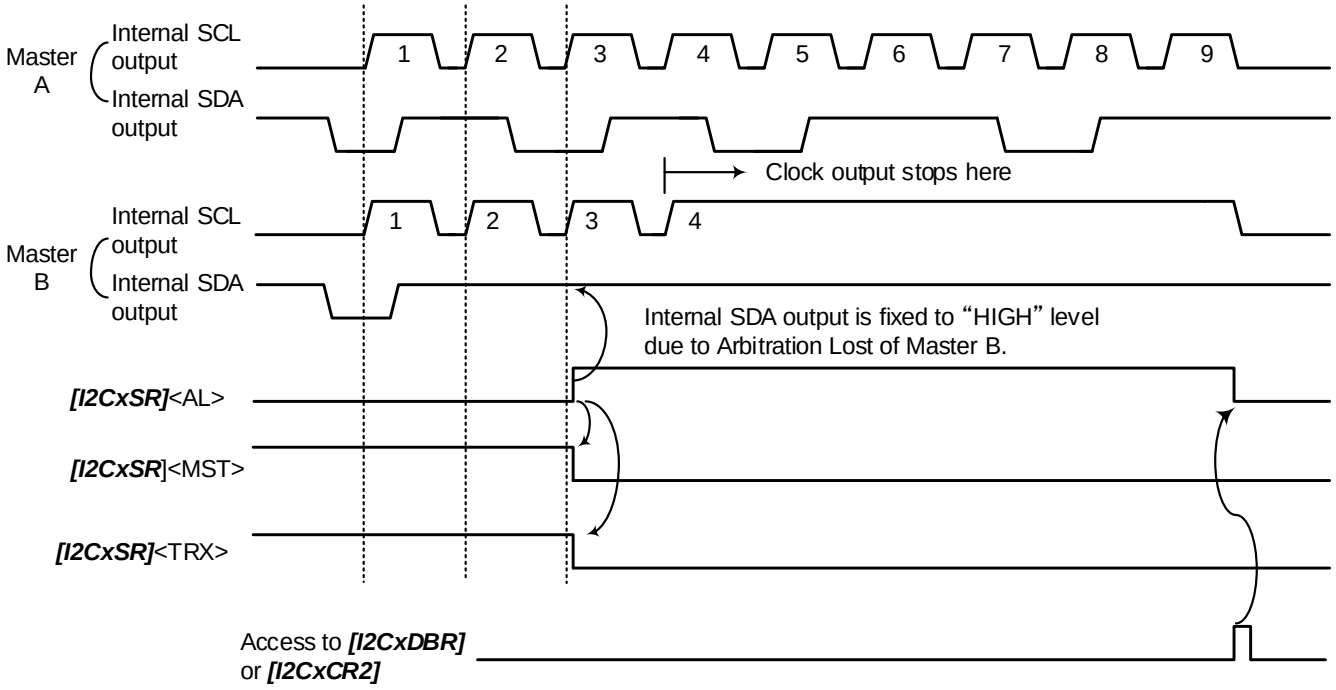


Figure 3.12 Arbitration Lost operation (the above-mentioned internal flag shows the master B)

This MCU only supports normal arbitration lost function that conforms to I^2C-bus specification. It does not support the Arbitration Lost function when a NACK is transferred. In this case, ***[I2CxSR]***<LRB> should be checked in the interrupt service routine of the completion of the transmission.

Note that multiple transmissions start almost simultaneously in a multitasking fashion, the following cases are assumed:

(1) Arbitration is detected before the transmissions start, and they are cancelled.
(2) Arbitration is detected after the transmissions start.

In addition, the state of a device can be known by <AL>, <MST>, and <TRX>.

Note: When Arbitration Lost does not occur in Single Master mode, set ***[I2CxOP]***<DISAL>=1. And also does not set the interrupt request of Arbitration Lost and monitor flag.

3.3.11. Last received Bit Monitor

[I2CxSR]<LRB> is always updated to the value of the SDA on the rising edge of the SCL Line.

In the acknowledgement mode, reading ***[I2CxSR]***<LRB> immediately after generation of the INTI2Cx interrupt request causes ACK signal to be read.

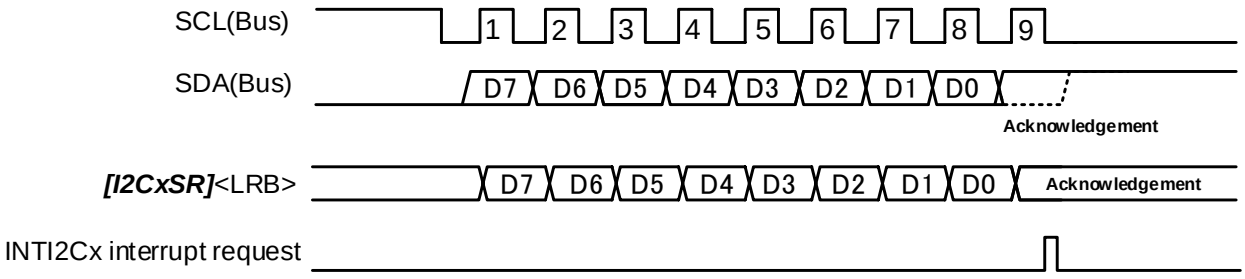


Figure 3.13 Change of the Last Received Bit Monitor

3.3.12. Repeated START detection

In slave mode, when a Repeated START is detected on the bus line, ***[I2CxOP]*** <RSTA> is set to "1". A Repeated START is used for a master device to change the direction of transmission without terminating data transfer to a slave device.

Since <RSTA> is only initialized by a reset, clear the flag(<RSTA>=0) when STOP condition occurs, bus free,.etc. (refer to Figure 3.14)

In case ***[I2CxPRS]***<PRSCK> ≠ 1 in Master mode, <RSTA> is set to "1" after the first START condition after reset. Perform clear processing at the first INTI2Cx interrupt service routine, etc. (refer to Figure 3.15)

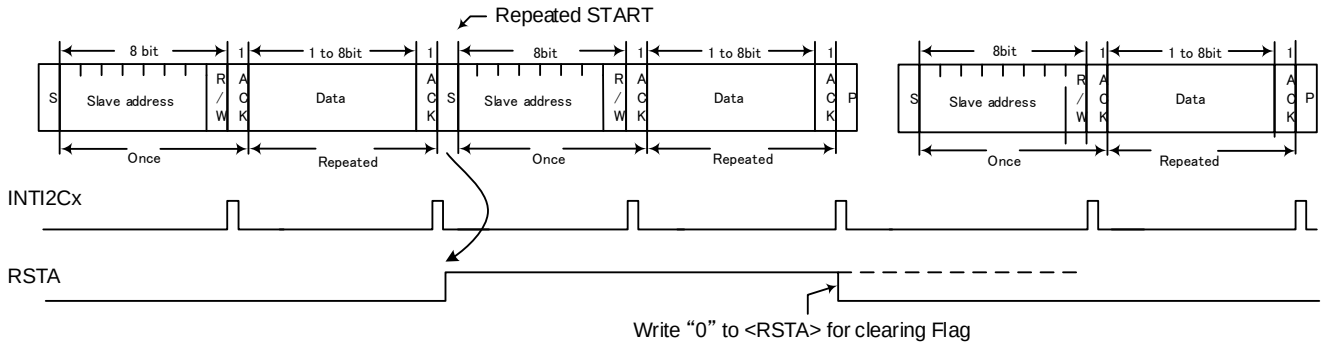


Figure 3.14 Repeated START detection Flag (Slave mode, Mater mode: <PRSCK>=1)

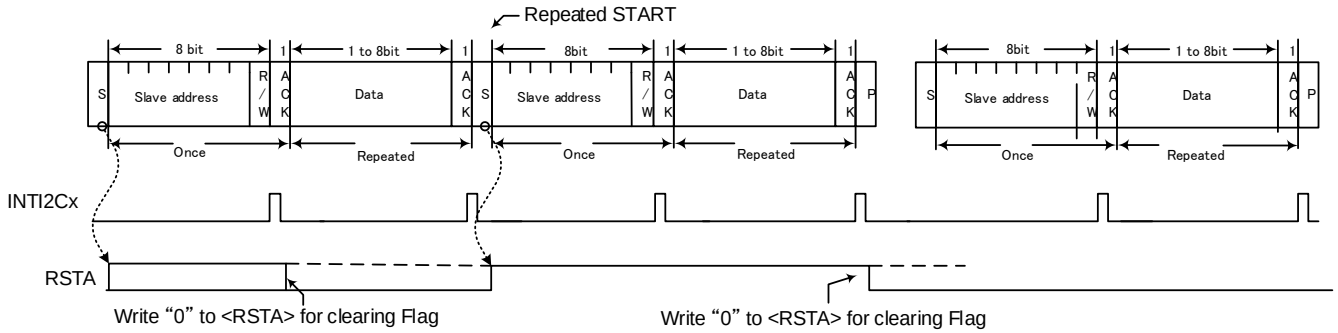


Figure 3.15 Repeated START detection Flag (Master mode: <PRSCK>≠1)

3.3.13. Software reset

I²C has a software reset function which initializes I²C. When I²C locks by a noise etc., I²C can be initialized by using this function.

Writing sequentially "10" and "01" to ***[I2CxCR2]***<SWRES[1:0]> causes software reset.

After a software reset generating, although I²C is initialized, ***[I2CxCR2]*** <I2CM> and ***[I2CxDBR]*** register are not initialized.

At this time, ***[I2CxSR]***<LRB> becomes undefined. This flag state is the same as the pin state at which software reset has occurred.

3.3.14. Noise cancellation

The SCL pin and SDA pin have the noise cancelling function. Digital noise cancellation will enable setting with ***[I2CxOP]***<NFSEL>=0.

In digital noise cancellation, signals less than Tprsck (Prescaler clock width) is eliminated as noise.

When using I²C Wakeup Function, Analog noise Filter becomes used.

3.3.15. Interrupt service request and release

The I²C interface has four interrupts, INTI2Cx interrupt, bus free detection interrupt, NACK detection interrupt, and Arbitration Lost interrupt.

(1) INTI2Cx interruption

In the case of master mode, after transmission of the number of clocks of data transfer set to ***[I2CxCR1]***<BC[2:0]> and ***[I2CxCR1]***<ACK> is completed, The INTI2Cx interrupt request is generated.

In the case of a slave mode, if the following conditions are established based on the above-mentioned conditions, an INTI2Cx interrupt occurs.

- When ***[I2CxCR1]***<NOACK> is "0", after the output of the acknowledge signal which is generated when the received slave address matches the slave address set to ***[I2CxAR]***<SA[6:0]>.(It is the same when setting ***[I2CxAR2]***<SA2>)
- When ***[I2CxCR1]***<NOACK> is "0", after the acknowledge signal is generated when a General-call address is received, matched Slave address, or at the end of data transfer after receiving a General call.

When an interrupt request (INTI2Cx) is generated, ***[I2CxSR]***<PIN> is cleared to "0". While ***[I2CxSR]***<PIN> is cleared to "0", the I²C pulls the SCL line to the "LOW" level.

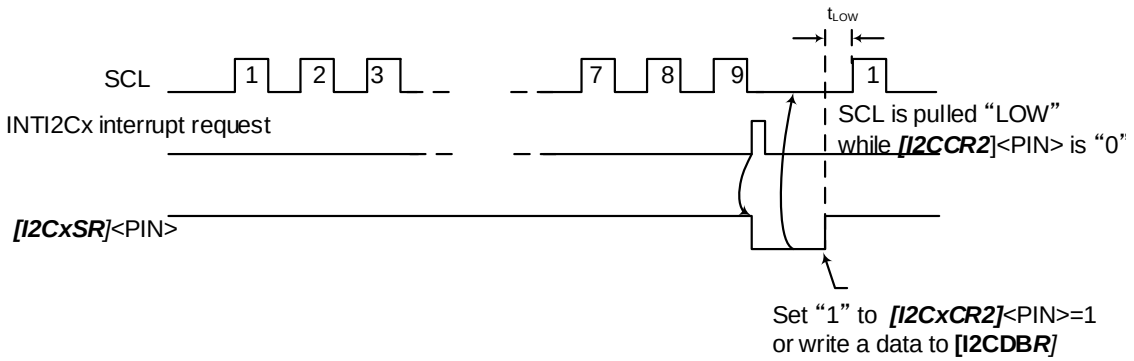


Figure 3.16 *[I2CSR]*<PIN> and SCL(<SELPINCD>=0)

In case ***[I2CxIE]***<SELPINCD>=0, ***[I2CxSR]***<PIN> is set to "1" when data is written to ***[I2CxDBR]***.

It takes a period of t_{LOW} for the SCL line to be released after ***[I2xCSR]***<PIN> is set to "1". When the program writes "1" to ***[I2CxSR]***<PIN>, it is set to "1". However, writing "0" does not clear this bit to "0".

Note: When Arbitration Lost occurs while a slave address and direction bit are transferred in the master mode, ***[I2CxSR]***<PIN> is cleared to "0" and INTI2Cx occurs. This does not relate to whether a slave address matches ***[I2CxAR]***<SA>.

(2) Bus free detection interrupt

A bus free detection interrupt occurs when the ***[I2CxSR]***<BB> flag of the device are changed (changed from <BB>=1 to <BB>=0).

When a bus free detection interrupt is used, set “1” to ***[I2CxIE]***<INTI2CBF>.

Note that <BB> is used to detect a bus free status regardless of the value of the slave address. When a STOP condition is detected, the bus status changes to the bus free status from the bus busy status. Thus, the <BB> flag changes to the bus free status and the bus free interrupt occur.

(3) NACK detection interrupt

When ***[I2CxCR1]***<ACK>=1 and ***[I2CxIE]***<INTNACK>=1, if the master receives a NACK, a NACK reception detection interrupt occurs.

This operation is NACK detection in which this device acts as the transmitter in master mode or slave mode. An ACK detection interrupt occurs at the same timing as INTI2Cx; therefore, check the interrupt with an ***[I2CxST]*** interrupt status flag.

(4) Arbitration Lost Detection interrupt

When Arbitration Lost is detected where Arbitration Lost interrupt is enabled. After the ***[I2CxSR]***<AL> flag is set, when transmission of the set-up number of clocks of data transfer is completed, an Arbitration Lost interrupt occurs.

3.3.16. DMA request output control

Data can be transferred using DMA in I^2C mode. But one master device and one slave device are connected to a bus line and the number of transfer data is decided beforehand.

A request is output one clock after an INTI2Cx transmission/ receive interrupt occurs; therefore, DMA sequential transfer between ***[I2CxDBR]*** and memory can be performed.

A request for DMA can be controlled with ***[I2CxIE]***<DMARI2CTX> in transmission or with <DMARI2CRX> in reception respectively Enable or Disable each register.

To transfer data using DMA, the number of bytes to be transferred should be preliminarily specified on both transmission and reception.

If Arbitration Lost occurs during I^2C transfer, the INTI2CxAL interrupt and the INTI2Cx interrupt occurs, A DMA request does not occur.

3.4. I²C Address Match Wakeup Function

This function is effective when the wakeup block (sub) is implemented.

When the I²C block is used in slave mode, if the frame address on the I²C bus matches a self-address (slave address) in Low power consumption mode, an interrupt (INTI2CWUP) occurs to clear Low power consumption mode.

When using this function, before shifting to the Low power consumption mode, stop the I²C interface (main) under bus-free.

The Main block (main) of I²C Interface keep the no System clock status except the IDLE mode of Low power consumption.

3.4.1. Clock stretch function

After the MCU confirms the match of slave addresses in slave mode, it returns an ACK and then performs the clock stretch operation that pulls the SCL to "LOW".

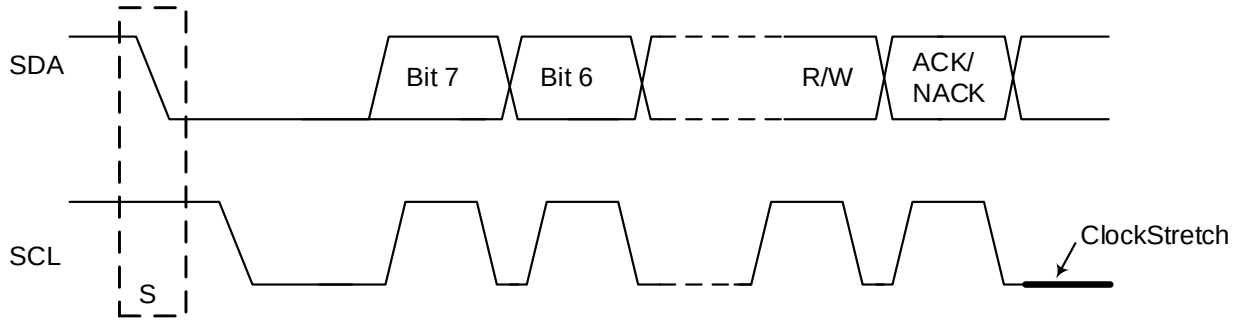


Figure 3.17 Clock stretch function

Note: If the MCU enters Low power consumption mode during the clock stretch operation, the I²C bus maintains locked state, so that the address match wakeup function cannot be used.

3.4.2. The operation flow of the Address Match Wakeup Function

The flow from address Match detection to interrupt generating, the release of a Low power consumption mode, and clock stretch release are shown in a figure.

(1) When the MCU detects an address match, the MCU returns an ACK to generate INTI2CWUP.

(2) Low power consumption mode is released, and the MCU starts the clock stretch function.

(3) In interrupt service routine after Low power consumption mode is released, the interrupt request is cleared with ***[I2CSWUPCR1]*** <INTEND>.

(4) The clock stretch function is released with ***[I2CSWUPCR1]***<I2RES>.

When ***[I2CSWUPCR1]***<INTEND> is changed to “0” from “1” in interrupt service routine in the interrupt routine(3), the I²C information is transmitted to the main block from the wakeup block.

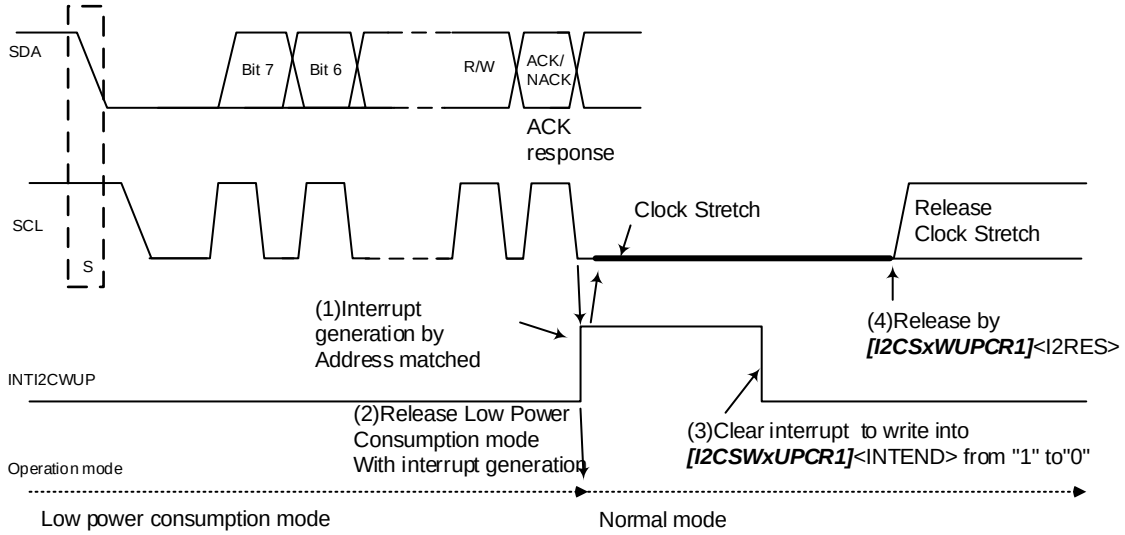


Figure 3.18 Address match Wakeup function

Note 1: When the MCU performs the address match wakeup function, the I^2C bus is locked until the MCU returns to the Normal operation(Clock Stretch).

Note 2: The Master returns an ACK signal only, the slave device selects an ACK or NACK signal and then send it.

Note that the detection conditions for slave addresses and the General call must be set the same when the address match wakeup function is used.

- ***[I2CxAR]***<SA> = ***[I2CSWUPCR2]***<WUPSA1>; same address, detecting condition
- ***[I2CxAR2]***<SA2>= ***[I2CSWUPCR3]***<WUPSA2>; same address, detecting condition
- Setting contents of ***[I2CxCR1]***<ACK>, <NOACK> & ***[I2CxOP]***<MFACK> is as same as ***[I2CSWUPCR1]***<ACK>, <SGCDI>

4. Register

4.1. Register list

The register and address of I²C are shown below.

Function name		channel/unit	Base Address	
			Type1	Type2
I²C Wakeup (Sub)	I2CS	-	0x4003E800	-
I²C interface (Main)	I2C	ch0	0x400A0000	0x400D1000
		ch1	0x400A1000	0x400D2000
		ch2	0x400A2000	0x400D3000
		ch3	0x400A3000	0x400D4000
		ch4	0x400A4000	0x400D5000

Note: The channel/unit and base address type are different by products. Please refer to "Product Information" of the reference manual for the details.

I2C (I²C interface)

Register name		Address(Base+)
I²C control register 1	***[I2CxCR1]***	0x0000
I²C data buffer register	***[I2CxDBR]***	0x0004
I²C address register	***[I2CxAR]***	0x0008
I²C control register 2	***[I2CxCR2]***	0x000C
I²C status register	***[I2CxSR]***	
I²C Prescaler clock setting register	***[I2CxPRS]***	0x0010
I²C interruption enable register	***[I2CxIE]***	0x0014
I²C interruption status register	***[I2CxST]***	0x0018
I²C Expanded function setting register	***[I2CxOP]***	0x001C
I²C Bus pin monitor register	***[I2CxPM]***	0x0020
I²C 2nd address register	***[I2CxAR2]***	0x0024

x" is Channel number.
The register of each channel is the same composition.

I2CS (I²C Wakeup)

Register name		Address(Base+)
I²C Wakeup Control register 1	***[I2CSWUPCR1]***	0x0000
I²C Wakeup Control register 2	***[I2CSWUPCR2]***	0x0001
I²C Wakeup Control register 3	***[I2CSWUPCR3]***	0x0002
I²C Wakeup Status register	***[I2CSWUPSL]***	0x0003

Note: The Above-mentioned register can serve as only byte access.

4.2. Register descriptions

4.2.1. *[I2CxCR1]* (I^2C control register 1)

Bit	Bit Symbol	After reset	Type	Function
31:8	-	0	R	Read as "0".
7:5	BC[2:0]	000	R/W	Selection of the number of transmission bits 000: 8 Bits 001: 1 Bit 010: 2 Bits 011: 3 Bits 100: 4 Bits 101: 5 Bits 110: 6 Bits 111: 7 Bits
4	ACK	0	R/W	Clock Generation / detection for acknowledgment. 0: None 1: Generated or Counted.
3	NOACK	0	R/W	Slave address match detection and General call detection. 0: the slave address match or General call detection is enabled. 1: the slave address match or General call detection is disabled. This bit has no meaning when ***[I2CxAR]***<ALS> is “1”. When ***[I2CxCR1]***<NOACK>=0, it is enabled for a slave address match detection and General call detection. If the received address matches its slave address specified at ***[I2CxAR]*** or is equal to the General call address, The I^2C pulls the SDA line to the "LOW" level during the 9th clock and outputs an acknowledgement signal. When ***[I2CxCR1]***<NOACK>=1, it is disabled for a slave address match detection and General call detection. If the received address matches its slave address specified at ***[I2CxAR]*** or is equal to the General call address, The I^2C release the SDA line to the "HIGH" level and outputs a non-acknowledgment signal.
2:0	SCK[2:0]	000	R/W	Select internal SCL output clock frequency (Note1) 000: n = 0 001: n = 1 010: n = 2 011: n = 3 100: n = 4 101: n = 5 110: n = 6 111: n = 7 ***[I2CxCR1]***<SCK[2:0]> is used to set the high and low periods of the serial clock to be output in master mode. The Prescaler clock which is divided according to ***[I2CxPRS]***<PRSK[4:0]> is used as the reference clock for generating the serial clock. The Prescaler clock is further divided according to ***[I2CxCR1]***<SCK [2:0]> and used as the serial clock. The default setting of the Prescaler clock is divided by 1(=fsys).

Note1: Do not change the setting of <SCK> while I^2C is operating.

Note: Don't change the contents of the registers when the START condition is generated, the STOP condition is generated or the data transfer is in progress.
Write data to the registers before the START condition is generated or during the period from when an interrupt request is generated for stopping the data transfer until it is released.

4.2.2. *[I2CxDBR]* (I²C data buffer register)

Bit	Bit Symbol	After reset	Type	Function
31:8	-	0	R	Read as "0".
7:0	DB[7:0]	0x00	R	When RD, reading the Received data
			W	When WR, writing the transfer data.

(Individual explanation)

a. <DB[7:0]>

Storing the serial transmission data.

When this register is used as the transmission device, write the data in left aligned to <DB[7:0]>.

When this register is used as the reception device, the reception data using serial transfer is stored to ***[I2CxDBR]***<DB[7:0]> in right aligned.

When the master needs to transmit a slave address, the transfer target address is written to ***[I2CxDBR]***<DB[7:1]> and the transfer direction is specified in ***[I2CxDBR]***]<DB[0]> as follows:

0: Master transmission → Slave reception

1: Master reception ← Slave transmission

When all the bits in the ***[I2CxDBR]*** register are written as "0", a general call can be sent out on the bus.

In both transmission and reception modes, a write to the ***[I2CxDBR]***register or read from the ***[I2CxDBR]*** register release the internal interrupt after the current transfer and initiates the next transfer.

[I2CxDBR] is provided as a transmission/receive buffer, it should be used as a dedicated transmit buffer in transmit mode and as a dedicated receive buffer in receive mode. This register in transmit mode and as a dedicated receive buffer in receive mode.

This register should be accessed on a transfer -by - transfer basis.

4.2.3. *[I2CxAR]* (I²C address register)

Bit	Bit Symbol	After reset	Type	Function
31:8	-	0	R	Read as "0".
7:1	SA[6:0]	0x00	R/W	A setup of the 1st slave address
0	ALS	0	R/W	Specify address recognition mode 0: Recognize its slave address (addressing format). 1: Do not recognize its slave address (free data format).

(Individual explanation)

a. <SA[6:0]>

Setting the 1st Slave address (7bits) in the Slave device.

When a slave address can be recognized with ***[I2CxAR]***<ALS> as described later, the data transfer operation is determined by 7 bits address (+1 direction bit) that is transferred from the master immediately after a start condition is set.

b. <ALS>

Specify Address recognition mode.

0: Permit address recognition mode (addressing format).

1: Forbid address recognition mode (free data format).

4.2.4. *[I2CxCR2]* (I^2C control register 2)

Bit	Bit Symbol	After reset	Type	Function
31:8	-	0	R	Read as "0".
7	MST	0	W	A master / slave selection 0: Slave 1: Master Notes: refer to “3.3.2. Serial clock”.
6	TRX	0	W	Transmission/receiving selection 0: Receiver 1: Transmitter Notes: refer to “3.3.3.1. Transmitter/ receiver selection”.
5	BB	0	W	Generating of START/STOP condition 0: STOP condition generating 1: START condition generating Notes: refer to “3.3.5. Generating START and STOP Conditions”.
4	PIN	1	W	Release the process request 0: Invalid 1: release the Process request Notes: refer to “3.3.15. Interrupt service request and release”
3	I2CM	0	W	I^2C operation control 0: Disable 1: Enable This bit cannot be cleared to 0 to disable I^2C operation while transfer operation is being performed. Read the status register to confirm the completion of data transfer before disabling the I^2C operation.
2	-	0	R	Read as "0".
1:0	SWRES[1:0]	00	W	Software reset generation Write "10" followed by "01" to generate a reset. Writing “10” followed by “01” into this 2bits generates a software reset. (reset width is 1 clock as fsys) A software reset release the SCL and SDA lines (HIGH state) if the device is data transfer and break the communication. The I^2C bus interface is initialized except ***[I2CxCR2]***<I2CM> and ***[I2CxDBR]*** register When software reset is performed, make sure to write “0” to ***[I2CxCR2]***[7:4]

Note: the ***[I2CxCR2]***<MST>,<TRX>,<BB>,<PIN> bits are given independent functions, they are used in typical combinations, as shown below, according to the ***[I2CxSR]*** setting.
When writing to these bits, make sure that notice.

Note: Don't change the contents of the registers, except ***[I2CxCR2]***<SWRST[1:0]>, when the START condition is generated, the STOP condition is generated or the data transfer is in progress.
Write data to the registers before the START condition is generated or during the period from when an interrupt request is generated for stopping the data transfer until it is released.

[I2CxSR]			*[I2CxCR2]*				Operation
[7]MST	[5]BB	[4]PIN	[7]MST	[6]TRX	[5]BB	[4]PIN	
0	0	1	0	0	0	0	Waiting for START condition as a slave.
			1	1	1	1	START condition generating
1	1	0	1	1	0	1	STOP condition generating

4.2.5. *[I2CxSR]* (I²C status register)

Bit	Bit Symbol	After reset	Type	Function
31:8	-	0	R	Read as "0".
7	MST	0	R	A master / slave selection status monitor 0: Slave 1: Master
6	TRX	0	R	Transmission / receiving selection status monitor 0: Receiver 1: Transmitter
5	BB	0	R	Bus state monitor 0: Bus free 1: Bus Busy
4	PIN	1	R	A processing demand state and SCL line state monitor 0: Under a processing demand, the SCL line "LOW" 1: Processing demand nothing, SCL line free
3	AL	0	R	Arbitration Lost detection monitor 0: Invalid 1: Detection
2	AAS	0	R	Slave address match detection monitor 0: Invalid 1: Detection
1	AD0	0	R	General call detection monitor 0: Invalid 1: Detection
0	LRB	0	R	The last receiving bit monitor 0: The last receiving bit 0 1: The last receiving bit 1

4.2.6. *[I2CxPRS]* (I²C Prescaler clock setting register)

Bit	Bit Symbol	After reset	Type	Function
31:5	-	0	R	Read as "0".
4:0	PRSCK[4:0]	00001	R/W	Prescaler clock frequency for generating the Serial clock 00000: P = divided by 32 00001: P = divided by 1 00010: P = divided by 2 - - - 11111: P = divided by 31 Please be sure to refer to (***[I2CxCR1]*** (I2C control register 1)).

4.2.7. *[I2CxIE]* (I^2C interrupt enable register)

Bit	Bit Symbol	After reset	Type	Function
31:7	-	0	R	Read as "0".
6	SELPINCD	0	R/W	Extended PIN Release condition setting. 0: The pin is not set (PIN=0) by reading DBR. (When the DMA is not used, use this setting) 1: The pin is set (PIN=1) by reading DBR. (When the DMA is used, use this setting.)
5	DMARI2CTX	0	R/W	Enables/disables a DMAC transmission request output 0: Disable 1: Enable
4	DMARI2CRX	0	R/W	Enables/disables a DMAC reception request output. 0: Disable 1: Enable
3	INTNACK	0	R/W	I^2C NACK detection interrupt enable or disable (Note 1) 0: Disable 1: Enable
2	INTI2CBF	0	R/W	I^2C Bus free detection interrupt enable or disable (Note 1) 0: Disable 1: Enable
1	INTI2CAL	0	R/W	Arbitration Lost detection interrupt enable or disable setting.(Note 1) 0: Disable 1: Enable
0	INTI2C	0	R/W	I^2C interrupt enable or disable setting.(Note 1) 0: Disable 1: Enable

Note1: These settings enable or disable an interrupt output from the I^2C functions. The interrupt setting of the CPU is required as well.

4.2.8. *[I2CxST]* (I^2C interrupt status register)

Bit	Bit Symbol	After reset	Type	Function
31:4	-	0	R	Read as "0".
3	NACK	0	R	Indicates NACK detection interrupt status.(Note 1) 0: No interrupts 1: Interrupts occurs
			W	Clears NACK detection interrupt status. 0: Invalid 1: Clear
2	I2CBF	0	R	Indicates INTI2CxBF interrupt status.(Note 1) 0: No interrupts 1: Interrupt occurs
			W	Clear INTI2CxBF interrupt status. 0: Invalid 1: Clear
1	I2CAL	0	R	Indicate INTI2xCAL interrupt status.(Note 1) 0: No interrupts 1: Interrupt occurs
			W	Clear INTI2xCAL interrupt status. 0: Invalid 1: Clear
0	I2C	0	R	Indicate INTI2Cx interrupt status.(Note 1) 0: No interrupts 1: Interrupt occurs
			W	Clear the I^2C interrupt status. 0: Invalid 1: Clear

Note1: These bits indicate the status of I^2C interrupt generation regardless of the setting of ***[I2CxIE]***<IE>.

4.2.9. *[I2CxOP]* (Expanded function setting register)

Bit	Bit Symbol	After reset	Type	Function
31:8	-	0	R	Read as "0".
7	DISAL	0	R/W	Control the Arbitration Lost detected function 0: Enable Arbitration Lost detect function 1: Disable Arbitration Lost detect function
6	SA2ST	0	R	Discriminates the received slave address. 0: Not match <SA2> 1: Matches <SA2>
5	SAST	0	R	Discriminates the received slave address. 0: Not match <SA> 1: Matches <SA>
4	NFSEL	0	R/W	Select Noise cancellation 0: Digital 1: Can not be set, Only Write as "0".
3	RSTA	0	R	Detects a Repeated START flag 0: Not detected 1: Detected
			W	0: Clear (Note1) 1: Invalid
2	GCDI	0	R/W	Sets General call detection. (Valid only when <NOACK>=0) 0: Detection is ON. 1: Detection is OFF.
1	SREN	0	R	Sets the Repeated START output. (Valid only when the MCU in master mode.) 0: Repeated START has completed. 1: Repeated START is ongoing.
			W	0: Invalid 1: Repeated START request
0	MFACK	0	R/W	Selects an ACK Output. 0: ACK Output 1: NACK Output Be careful, It cannot be used in a free data format.

Note1: <RSTA> is initialized by RESET only.

4.2.10. *[I2CxPM]* (I²C Bus pin monitor register)

Bit	Bit Symbol	After reset	Type	Function
31:2	-	0	R	Read as "0".
1	SDA	0	R	Monitors the SDA pin. 0: LOW level 1: HIGH level
0	SCL	0	R	Monitors the SCL pin. 0: LOW level 1: HIGH level

4.2.11. *[I2CxAR2]* (I²C 2nd address register)

Bit	Bit Symbol	After reset	Type	Function
31:8	-	0	R	Read as "0".
7:1	SA2[6:0]	0x00	R/W	A setup of the 2nd slave address
0	SA2EN	0	R/W	A use setup of the 2nd slave address 0: No use 1: Use

Note: set “0” to ***[I2CxCR1]***<NOACK> before using this register.

4.2.12. *[I2CSWUPCR1]* (I²C Wakeup control register 1)

Bit	Bit Symbol	After reset	Type	Function
7	BUSY	0	R	0: STOP condition detection 1: START condition detection
6	SGCDI	0	R/W	0: General call detection On 1: General call detection Off
5	ACK	0	R/W	0: ACK Output ("0" Outputs) 1: No ACK Output ("1" Output) (NACK setup)
4	I2RES	1	R	I² C bus reset 0: Reset release 1: Under reset
			W	0: Reset release 1: Reset done
3	RW	0	R	The transceiver demand monitor from a master 0: Slave reception 1: Slave transmission
2	-	0	R	Read as "0".
1	GC	0	R	General call detection status 0: No detection 1: Detection
0	INTEND	0	R/W	Interrupt release 0: - 1: Interrupt release

Note 1: When <I2RES>=1, I²C bus is reset; however, the setting is not automatically returned to "0". When the reset state is released, set <I2RES>=0 prior to the reset.

Note 2: When the reset is performed with <I2RES>, all read registers of the I²C wakeup block (sub) are initialized; however write data is not initialized.

Note 3: After the reset is released with <I2RES>, the circuit operations are performed along to the preset values. Therefore, set slave addresses or ACK signals when <I2RES>=1.

Note 4: <RW> is "0" regardless of transmission/receive request from the master when the addresses do not match.

4.2.13. *[I2CSWUPCR2]* (I²C Wakeup Control Register 2)

Bit	Bit Symbol	After reset	Type	Function
7:1	WUPSA1[6:0]	0x00	R/W	The 1st slave address setting
0	-	0	R	Read as "0".

4.2.14. *[I2CSWUPCR3]* (I²C Wakeup control register 3)

Bit	Bit Symbol	After reset	Type	Function
7:1	WUPSA2[6:0]	0x00	R/W	The 2nd slave address setting
0	WUPSA2EN	0	R/W	Selection for the 2nd slave address 0: No use 1: used

4.2.15. *[I2CSWUPSL]* (I²C status register)

Bit	Bit Symbol	After reset	Type	Function
7:3	-	-	-	Read as "0".
2	WUPSA2	0	R	Receiving status of the 2nd address 0: Not matched 2nd Slave address. 1: Matched 2nd Slave address.
1	WUPSA	0	R	Receiving status of the 1st address 0: Not matched 1st Slave address. 1: Matched 1st Slave address.
0	-	0	R	Read as "0".

5. Usage example

5.1. Data transfer procedure

5.1.1. Device Initialization

After ensuring that the SDA and SCL pins are high (Bus free), set ***[I2CxCR2]***<I2CM> to "1" for enable the I^2C.

Next, write "1" to ***[I2CxCR1]***<ACK]> and "0" to ***[I2CxCR1]***<NOACK>. Writing "000" to ***[I2CxCR1]***<BC[2:0]> at the time.

These setting enable acknowledge operation, slave address match detection, and general call detection and set the data length to 8 bits. Set "t_{HIGH}" and "t_{LOW}" in ***[I2CxCR1]***<SCK>.

Then, program ***[I2CxAR]*** by specifying a slave address at ***[I2CxAR]***<SA[6:0]> and an address recognition mode at ***[I2CxAR]***<ALS>. (<ALS> must be cleared to "0" when using the addressing format).

Finally, write "0" to ***[I2CxCR2]***<MST>,***[I2CxCR2]***<TRX>,***[I2CxCR2]***<BB>, write "1" to ***[I2CxCR2]***<PIN>, write "00" to ***[I2CxCR2]***<SWRES[1:0]> to configure the device as a slave receiver.

In case not-using the DMA, Set ***[I2CxIE]***<SELPINCD> to "0".

5.1.2. Generating of START condition and slave address

Before generating the start condition or slave address, execute the following operations: check the bus free condition (***[I2CxSR]***<BB> = 0); set "1" to ***[I2CxCR1]***<ACK>, and write data that contains a slave address of ***[I2CxDBR]*** and the direction bit. When "1" is written to ***[I2CxCR2]***<MST>,***[I2CxCR2]***<TRX>,***[I2CxCR2]***<BB>, and ***[I2CxCR2]***<PIN>, the slave address and direction bit are output to the I^2C bus. Note that the time of t_{HIGH} is required until the SCL pin falls after the START condition.

Subsequently, an INTI2Cx interrupt occurs on the falling edge of the 9th clock of SCL to clear ***[I2CxSR]***<PIN> to "0". At this time, SCL should be pulled "LOW" level while ***[I2CxSR]***<PIN> is "0". Only when the slave device returns an acknowledge signal, the direction of ***[I2CxSR]***<TRX> is changed by hardware according to the direction bit at the timing when an INTI2Cx interrupt request occurs. If an acknowledge signal is not returned, ***[I2CxSR]***<TRX> is not changed.

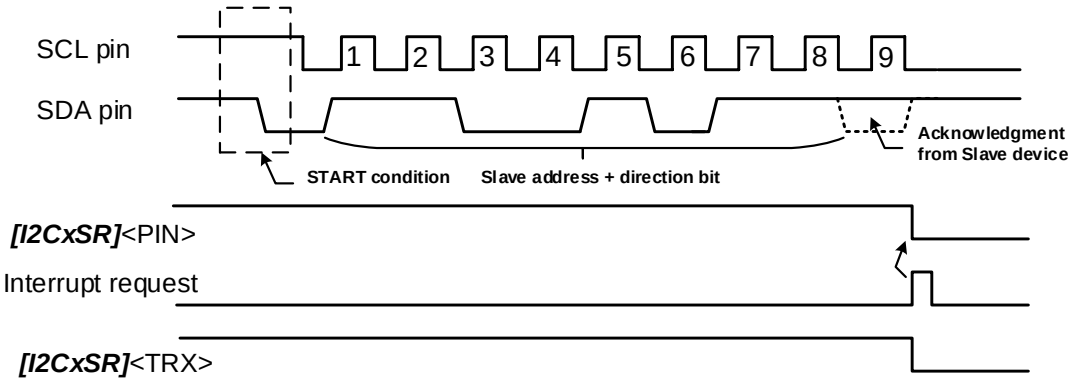


Figure 5.1 Generating of a START condition and a slave address

5.1.3. 1 word of data transfer

At the end of a data word transfer, the INTI2Cx interrupt is generated to test ***[I2CxSR]***<MST> to determine whether the I^2C is in the master or slave mode.

5.1.3.1 When *[I2CxSR]*<MST> is “1” (master mode)

Test ***[I2CxSR]***<TRX> to determine whether the I^2C is configured as a transmitter or a receiver.

a. ***[I2CxSR]***<TRX>=1(Transmitter mode)

Check acknowledge from the receiver using the ***[I2CxSR]***<LRB> flag.

If ***[I2CxSR]***<LRB> is "0", that means the receiver requires further data. If the next data to be transmitted has eight bits, the data is written into ***[I2CxDBR]***. If the data has different length, ***[I2CxCR1]***<BC[2:0]> and ***[I2CxCR1]***<ACK> are programmed and the transmission data is written into ***[I2CxDBR]***.

Writing the data makes ***[I2CxSR]***<PIN> to "1", causing the SCL pin to generate a serial clock for transferring a next data word, and the SDA pin to transfer the data word.

Note that even if the receiver requests the next data, a STOP condition can be issued.

After the transfer is completed, the INTI2Cx interrupt request is generated, ***[I2CxSR]***<PIN> is cleared to "0", and the SCL pin is pulled to the "LOW" level.

To transmit more data words, test ***[I2CxSR]***<LRB> again and repeat the above procedure.

[I2CxSR]<LRB> =1 means that the receiver does not request the data; therefore, the transmitter generates a STOP condition (described later) to stop data transfer.

Note that even if the receiver requests the next data, data can be transmitted.

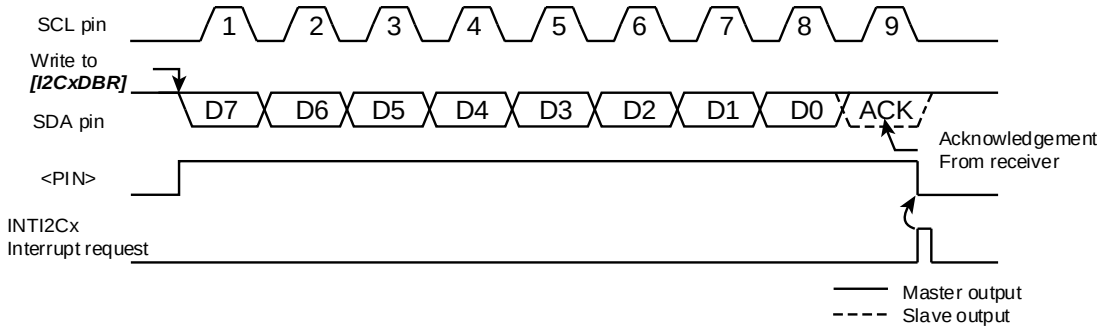


Figure 5.2 In case of *[I2CxCR1]* <BC[2:0]> = 000, *[I2CxCR1]* <ACK> = 1

b. [I2CxSR]<TRX>=0 (Receiver mode)

If dummy data (0x00) is written to ***[I2CxDBR]***, or "1" is set to ***[I2CxCR2]***<PIN>, a One-word transfer clock and acknowledge signal are output. After an INTI2Cx interrupt indicating the completion of reception occurs, read receive data from ***[I2CxDBR].***

If the data has different length, set ***[I2CxCR1]***<BC[2:0]> again and set ***[I2CxCR1]***<ACK> to "1", and then Write the ***[I2CxDBR]*** to "0x00" or set the ***[I2CxCR2]***<PIN > to "1".

(Data that immediately after the slave address is transmitted is undefined.)

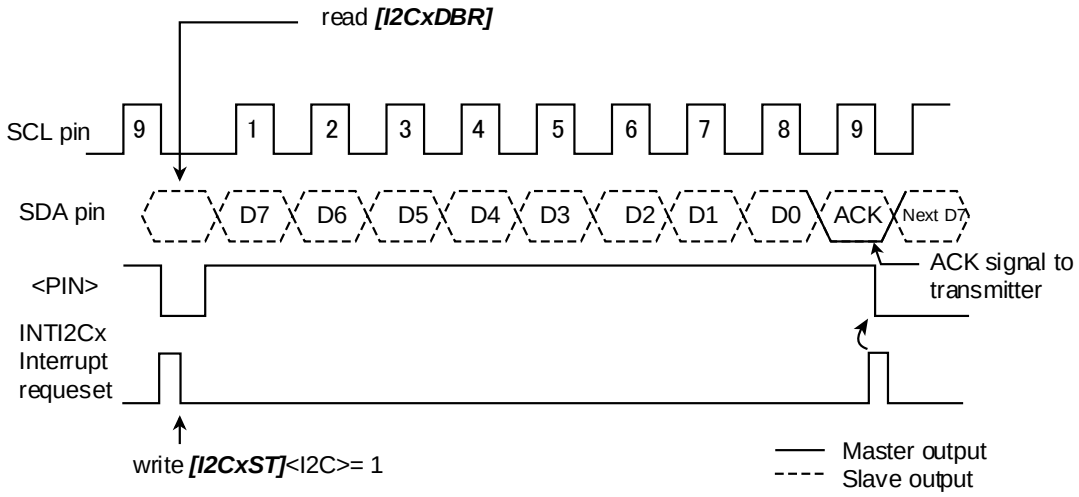


Figure 5.3 *[I2CxCR1]*<BC[2:0]> = 000, *[I2CxCR1]*<ACK> = 1

C. ***[I2CxSR]***<TRX>=0 (when receiving the last word)

To determine the last word, perform the pseudo-communication without an acknowledge signal.

The process flows are described below:

Before Received the last data, the following procedure is required to terminate the data transmission from the transmitter.

1. Read the received data from ***[I2CxDBR]***.
2. Set ***[I2CxOP]***<MFACK> to “1” for the NACK transmission.
3. Write the Dummy data (0x00) into ***[I2CxDBR]*** to set the ***[I2CxCR2]***<PIN> as "1".

When "1" is set to ***[I2CxCR2]***<PIN>, One- word transfer is started in NACK transmission. After One-word transfer is complete, proceed with the following process:

1. Read the received data from ***[I2CxDBR]***.
2. Generate the STOP condition, Terminates the data transmission.

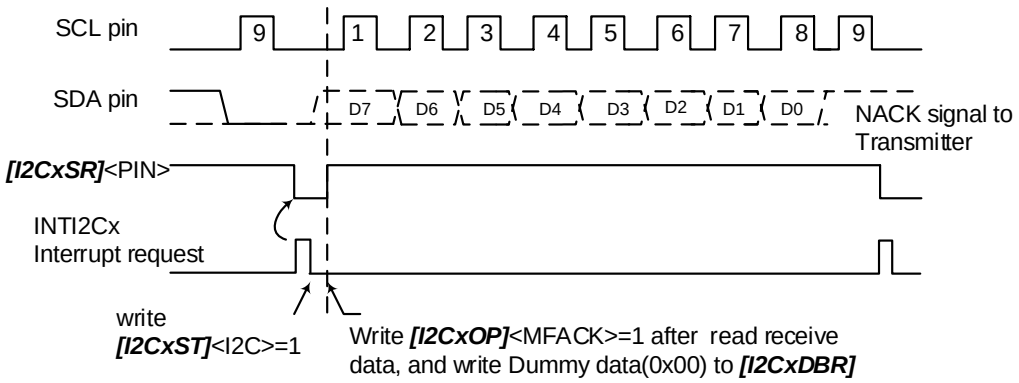


Figure 5.4 Terminating data transmission in the Master receiver mode

5.1.3.2 When *[I2CxSR]*<MST> is "0" (slave mode)

In the slave mode, the normal slave mode processing or the processing as a result of Arbitration Lost is carried out.

In the slave mode, the I^2C generates the INTI2Cx interrupt request on the following status

(1) When ***[I2CxCR1]***<NOACK> is "0", after outputting the acknowledge signal when the received slave address matches the slave address set in [I2CxAR]<SA>.
(2) When ***[I2CxCR1]***<NOACK> is "0", after outputting the acknowledge signal when the General call is received
(3) When a data transfer has been completed in response to a received slave address matching or a General call, if the I^2C detects Arbitration Lost in the master mode, it switches to the slave mode. Upon the completion of data word transfer in which Arbitration Lost is detected, the INTI2Cx interrupt request is generated

Table 5.1 shows the operation of interrupt requests and ***[I2CxSR]***<PIN> after Arbitration Lost.

Table 5.1 The operation of INTI2Cx interrupt request and *[I2CxSR]*<PIN> after Arbitration Lost

	In master mode, detecting the Arbitration Lost when sending the Slave address.	In Master transmitter mode, detecting the Arbitration Lost when sending the Data.
INTI2Cx interrupt request	When a word transmission is completed, INTI2Cx interrupt is generated.	
[I2CxSR]<PIN>	Clear ***[I2CxSR]***<PIN> to "0".	

The INTI2Cx interrupt request is generated, ***[I2CxSR]***<PIN> is cleared to "0", and the SCL pin is pulled to the "LOW" level.

When data is written to or read from ***[I2CxDBR]*** or when ***[I2CxSR]***<PIN> is set to "1", the SCL pin is released after a period of t_{LOW}.

[I2CxSR]<AL>,***[I2CxSR]***<TRX>, ***[I2CxSR]***<AAS> and ***[I2CxSR]***<AD0> are tested to determine the processing required.

Table 5.2 shows the slave mode states and required processing.

Table 5.2 Processing in slave mode

I2CxSR <TRX>	I2CxSR <AL>	I2CxSR <AAS>	I2CxSR <AD0>	State	Processing
1	1	1	0	Arbitration Lost is detected while the slave address was being transmitted and the I²C received a slave address with the direction bit "1" transmitted by another master.	Set the number of bits in a data word to ***[I2CxCR1]***<BC[2:0]> and write the transmit data into ***[I2CDBR]***.
	0	1	0	In the slave receiver mode, the I²C received a slave address with the direction bit "1" transmitted by the master.	
		0	0	In the slave transmitter mode, the I²C has completed a transmission of one data word.	Test ***[I2CxSR]***<LRB>. If it has been set to "1", that means the receiver does not require further data. Set ***[I2CxCR2]***<PIN> to 1 and reset <TRX> to 0 to release the bus. If ***[I2CxSR]***<LRB>has been reset to "0", that means the receiver requires further data. Set the number of bits in the data word to ***[I2CxCR1***]<BC[2:0]> and write the transmit data to the ***[I2CxDBR].***
0	1	1	1/0	Arbitration Lost is detected while a slave address is being transmitted, and the I²C receives either a slave address with the direction bit "0" or a General call address transmitted by another master	Write the ***[I2CxDBR]*** (a dummy data “0x00”) to set ***[I2CxCR2]*** <PIN> to 1, or write "1" to <PIN>.
		0	0	Arbitration Lost is detected while a slave address or a data word is being transmitted, and the transfer is terminated.	The serial bus interface circuit is in slave mode. Therefore, clear ***[I2CxSR]***<AL> to “0”. To set “1”to ***[I2CxSR]***<PIN>, write dummy data (0x00) to ***[I2CxDBR]***.
	0	1	1/0	In the slave receiver mode, the I²C received either a slave address with the direction bit "0" or a General call address transmitted by the master.	Write the ***[I2CxDBR]*** (a dummy data “0x00”) to set ***[I2CxCR2]*** <PIN> to 1, or write "1" to ***[I2CxCR]***<PIN>.
		0	1/0	In the slave receiver mode, the I²C has completed a reception of a data word.	Set the number of bits in the data word to ***[I2CxCR1]***<BC[2:0]> and read the received data from ***[I2CxDBR]***, write dummy data(0x00) to ***[I2CxDBR]***

Note: In salve mode, if ***[I2CxAR]***<SA> is set to "0x00" a START byte (0x01) of the I²C bus standard is received, a slave Address match is detected and ***[I2CxSR]***<TRX> is set to "1". Do not set ***[I2CxAR]***<SA> to "0x00".

5.1.4. Generating of STOP condition

When ***[I2CxSR]***<BB> is "1", writing "1" to ***[I2CxCR2]***<MST>,***[I2CxCR2]***<TRX>,***[I2CxCR2]***<PIN> and "0" to ***[I2CxCR2]***<BB> causes the I²C to start a sequence for generating the STOP condition on the bus.

Do not alter the contents of ***[I2CxCR2]*** <MST>,***[I2CxCR2]*** <TRX>, ***[I2CxCR2]***<BB>,***[I2CxCR2]*** <PIN> until the STOP condition appears on the bus.

If another device is holding down the SCL bus line, the I²C waits until the SCL line is released.

After that, the SDA pin goes "HIGH", and then causing the STOP condition to be generated. Refer to "3.3.2 Serial clock" for the setup time from when the SCL line is released until the STOP condition occurs.

The STOP condition can be checked using the monitor pin ***[I2CxSR]***<BB> for the bus.

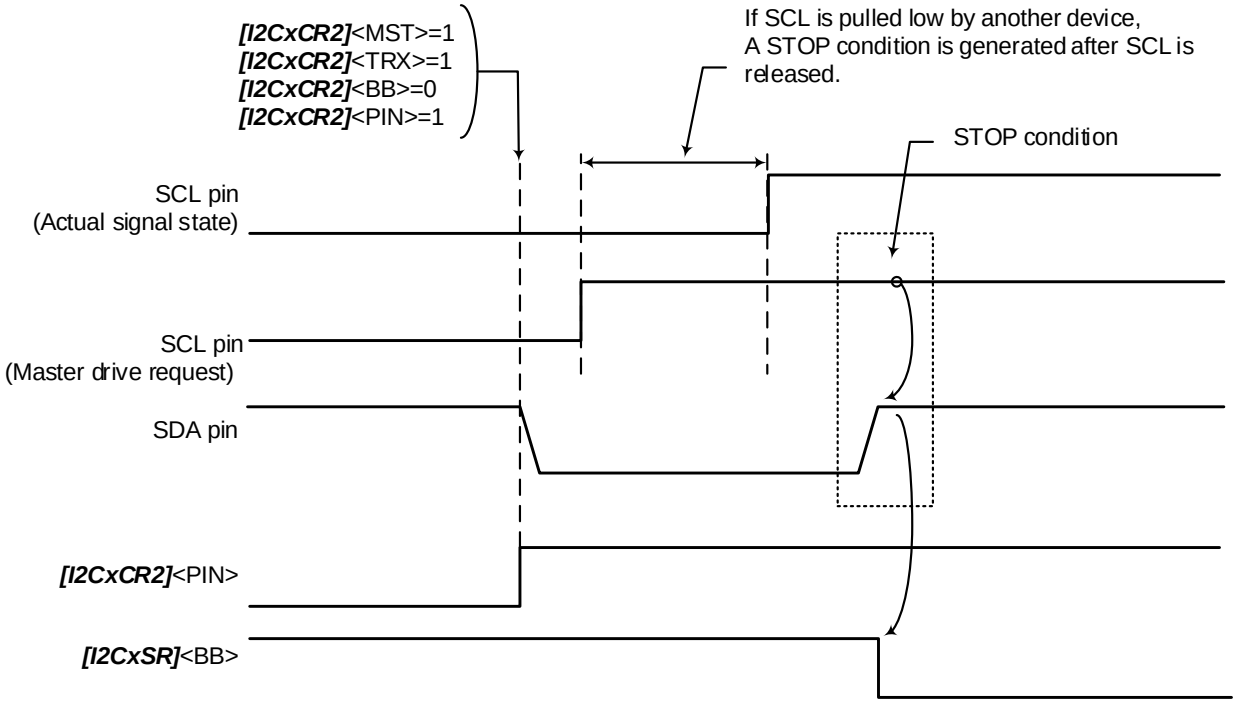


Figure 5.5 Generating of STOP condition

5.1.5. Procedure of Repeated START

Repeated START is used when a master device changes the data transfer direction without terminating the transfer to a slave device.

Repeated START is an operation that occurs after an ACK/NACK output caused by an INTI2Cx interrupt.

[I2CxOP]<SREN> should be set until the clock stretch function is cleared after the occurrence of interrupts.

The procedure of generating a Repeated START in the master mode is described below.

(1) In case of ***[I2CxOP]***<SREN>=1

- Set ***[I2CxOP]***<SREN>=1 when ***[I2CxSR]***<BB>=1 (a)
- Write a slave address and direction bit to the data buffer. (b)
- After that, write "1" to ***[I2CxCR2]***<MST>, <TRX>, <PIN>, and <BB> (c)
- Output a Repeated START condition on the I^2C bus. (d)

Note: Do not set "0" to ***[I2CxCR2]***<PIN> when ***[I2CxOP]***<SREN>=1.

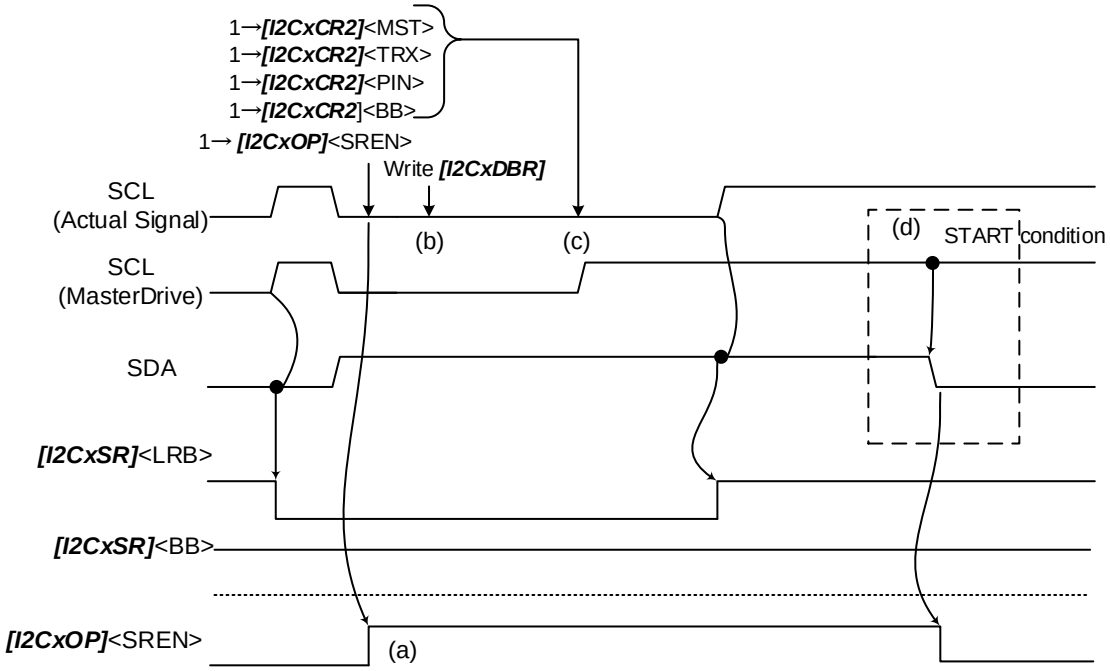


Figure 5.6 Repeated START (<SREN>=1)

(2) In case of ***[I2CxOP]***<SREN>=0

- Write "0" to ***[I2CxCR2]***<MST, TRX, BB> , write "0" to ***[I2CxCR2]***<PIN> and then open the I^2C bus. (a)
- At this time, SDA pin keeps "High" level, SCL pin keeps "Open", but the bus keeps "busy" status because of nothing the STOP condition on the bus.
  and then waiting for "0" by testing the ***[I2CxSR]***<BB>, confirm the open of the SCLx pin. (b)

  On the other hand,. possible to use the Bus free detection interrupt (refer to "3.3.15. Interrupt service request and release (2) Bus free detection interrupt) to confirm the open of the SCLx pin. (b')
- Waiting for "1" by testing the <LRB>, confirm the nothing pull to "Low" of SCL line by other devices (c)
- For keeping the set-up time for a Repeated START condition ($t_{su:STA}$), go and keep the time by software from confirming the Bus free to generating the START condition. (d)
  (Standard-mode: 4.7μs(Min.), Fast-mode: 0.6μs(Min.), Fast-mode Plus: 0.26μs(Min.))
- After confirming that the bus is free according to the above procedure, START condition is generated according to the procedure of "5.1.2. Generating of START condition and slave address " (e)

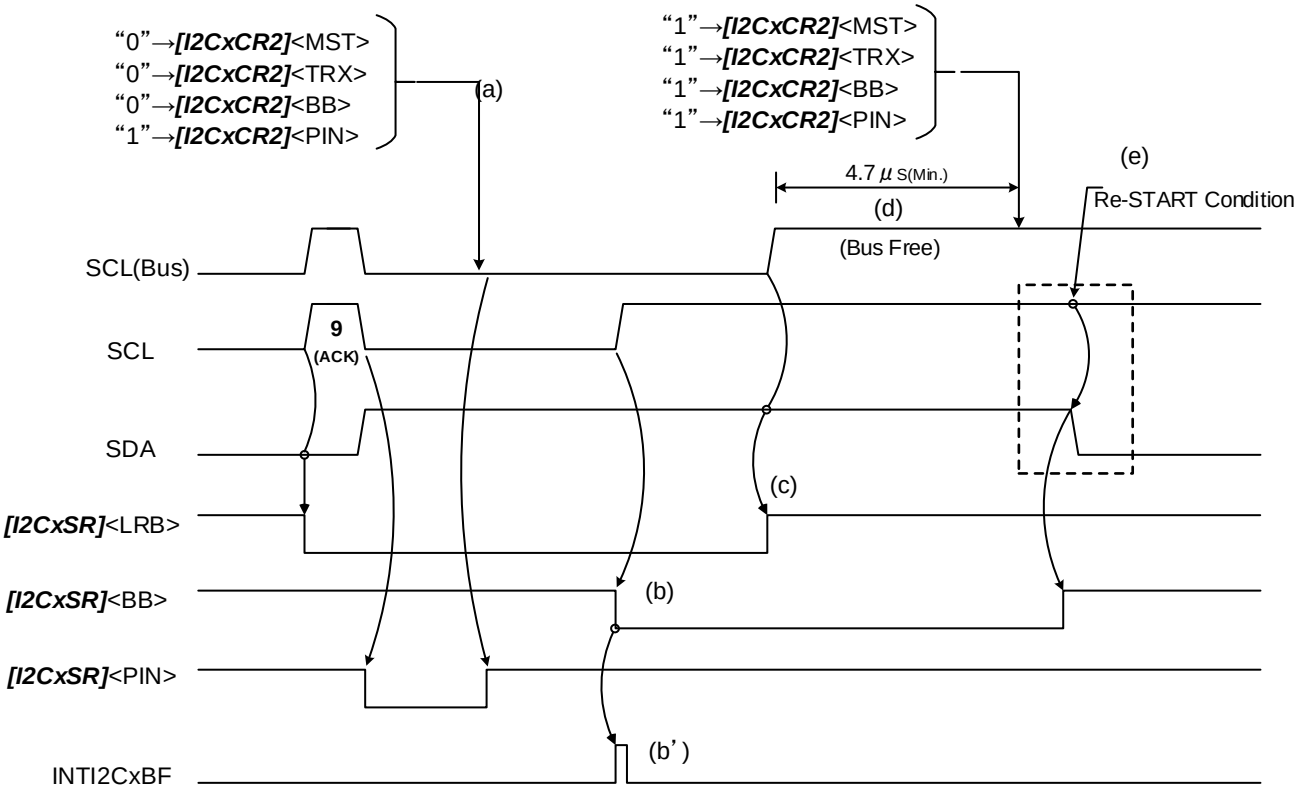


Figure 5.7 Repeated START (<SREN>=0)(Standard mode)

5.1.6. Data transfer by DMA

Data can be transferred using DMA in I^2C mode. But one master device and one slave device are connected to a bus line and the number of transfer data is decided beforehand.

In I^2C bus mode, when an INTI2Cx interrupt occurs, a DMA request occurs almost simultaneously (after 1 clock); therefore consecutive transfer can be executed between ***[I2CxDBR]*** (data buffer) and the memory using DMA transfer.

A DMA request for transmission and reception are separately executed.

Set enable or disable to the DMAC control register for transmission and reception before using the DMA.

To transfer data using DMA, the number of bytes to be transferred should be preliminarily specified on both transmission and reception.

If Arbitration Lost occurs during I^2C transfer, the INTI2CxAL interrupt and the INTI2Cx interrupt occurs, then the communication stops. A DMA request does not occur.

To set the PIN (***[I2CxSR]***<PIN>) by reading ***[I2CxDBR]*** in receiver mode then release the I^2C Bus, set “1” to ***[I2CxIE]***<SELPINCD>.

5.1.7. Transfer procedures in master mode

1. Generate START condition and slave address.
2. Check ***[I2CxSR]***<LRB> and <TRX> in interrupt service routine of INTI2Cx after output slave address.
3. When ***[I2CxSR]***<LRB> is "1", output a STOP condition and complete the transferring because no slave Device which has the sent slave address is on the bus.
4. When ***[I2CxSR]***<LRB> is "0", check <TRX> because slave device which has the sent slave address is on the bus. The proceeding is depended on <TRX>.

(1) When <TRX> is "1" (transmitter mode)

a. Set DMA such as transfer bit width, burst size, and the total transferring number, etc. Subtract 1 from the number of data to be transmitted. And enable to receive DMA request of INTI2Cx with ***[I2CxIE]***<DMARI2CTX>

b. Write the first data into ***[I2CxDBR]***. Start transmits the first data by this writing.

c. Start DMA transfer by the DMA request of INTI2Cx which is after the completed transfer. And transmit the 2nd data and following data.

d. When the specified number of DMA transfer (N-1 times) ends, a DMA transfer end interrupt occurs. Change the settings to ignore a DMA request reception in this interrupt service routine. (Set the ***[I2CxIE]***<DMARI2CTX> to “0”)

e. Generate STOP condition to stop transmission at the interrupt service routine of INTI2Cx.

(2) When <TRX> is "0" (receiver mode)

a. Set DMA such as transfer bit width, burst size, and the total transferring number, etc. Subtract 2 from the number of data to be transmitted. And enable to receive DMA request of INTI2Cx by ***[I2CxIE]***<DMARI2CRX>.

b. Read dummy data from ***[I2CxDBR]***. Start receiving the first data by this reading

c. Start DMA transfer by the DMA request of INTI2Cx which is after the completed receiving. And receive the 2nd data and the following data.

d. When the specified number of DMA transfer (N-2 times) ends, a DMA transfer end interrupt occurs. Change the settings to ignore a DMA request reception in this interrupt service routine. (Set the ***[I2CxIE]***<DMARI2CRX> to “0”)

e. Set "1" to <MFACK> to generate a NACK signal by an INTI2Cx interrupt that occurs at (n-1) data reception, and read then receive data from ***[I2CxDBR]***.

f. Generate STOP condition to stop transmission at the interrupt service routine of INTI2Cx.

5.1.8. Transfer procedures in slave mode

1. Receive START condition and slave address.
2. Check <TRX> in interrupt service routine of INTI2Cx after receiving slave address
3. Check <TRX>. The proceeding is depended on <TRX>.

(1)When <TRX> is "1"(Transmitter mode)

a. Set DMA such as transfer bit width, burst size, and the total transferring number, etc. Subtract 1 from the number of data to be transmitted. And enable to receive DMA request of INTI2Cx by ***[I2CxIE]***<DMARI2CTX>.

b. Write the first data into ***[I2CxDBR]***. This device can be received a clock from a master device and start transmitting the first data by this writing.

c. Start DMA transfer by the DMA request of INTI2Cx which is after the completed transfer. And transmit the 2nd data and following data.

d. When the specified number of DMA transfer (N-1 times) ends, a DMA transfer end interrupt occurs. Change the settings to ignore a DMA request reception in this interrupt service routine. (Set ***[I2CxIE]***<DMARI2CTX> to "0")

e. Wait for STOP condition from a master device without writing a data at the service routine of INTI2Cx.

(2)When <TRX> is "0"(Receiver mode)

a. Set DMA such as transfer bit width, burst size, and the total transferring number, etc. Subtract 2 from the number of data to be transmitted. And enable to receive DMA request of INTI2Cx by ***[I2CxIE]***<DMARI2CRX>.

b. Read dummy data from ***[I2CxDBR]***. Start receives the first data by this reading.

c. Start DMA transfer by the DMA request of INTI2Cx which is after the completed receiving. And receive the 2nd data and the following data.

d. When the specified number of DMA transfer (N-2 times) ends, a DMA transfer end interrupt occurs. Change the settings to ignore a DMA request reception in this interrupt service routine.

(Set ***[I2CxIE]***<DMARI2CRX to "0")

e. Set "1" to ***[I2CxOP***]<MFACK> to generate a NACK signal by an INTI2Cx interrupt that occurs at (n-1) data reception, and read then receive data from ***[I2CxDBR].***

f. Wait for STOP condition from a master device without writing a data at the service routine of INTI2Cx.

5.2. Wakeup operation and setting flow (Example)

<Initial setting> Wakeup function setting to enter STOP1 mode.

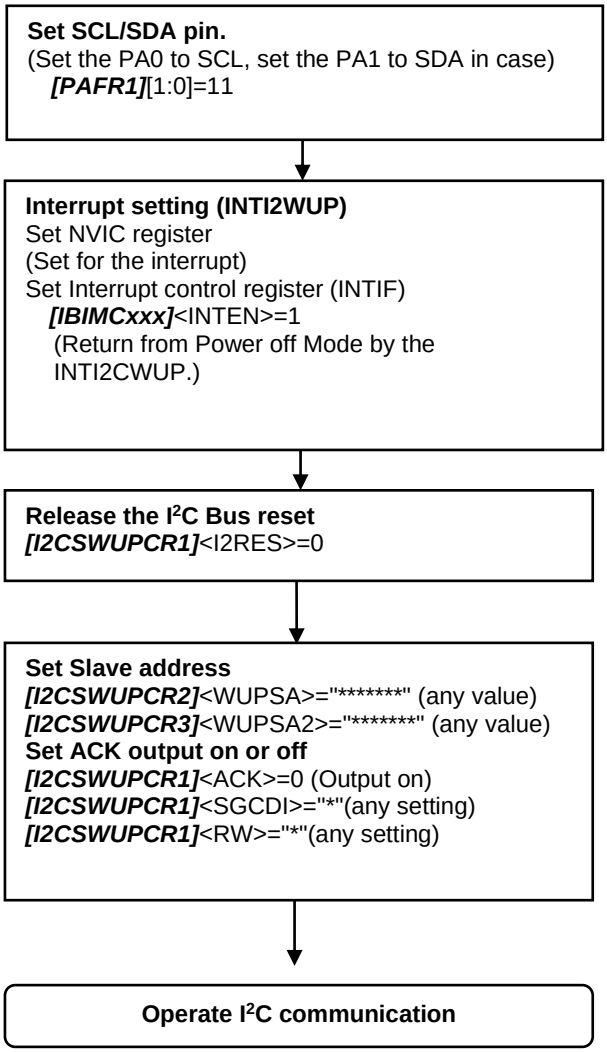


Figure 5.8 Wakeup Initialize setting

Note: The register can be written during the reset operation caused by ***[I2CSWUPCR1]***<I2RES>=1 even when reset operation continues.

The setting for the I^{2}C interface (main) should be completed in the release processing of the Low power consumption mode (STOP1 or STOP2). Communications after the address match wakeup function do not work properly unless the following registers are set.

[I2CxCR1]<BC[2:0]>,<ACK>,<NOACK>,<SCK>

[I2CxPRS]<PRSCK>

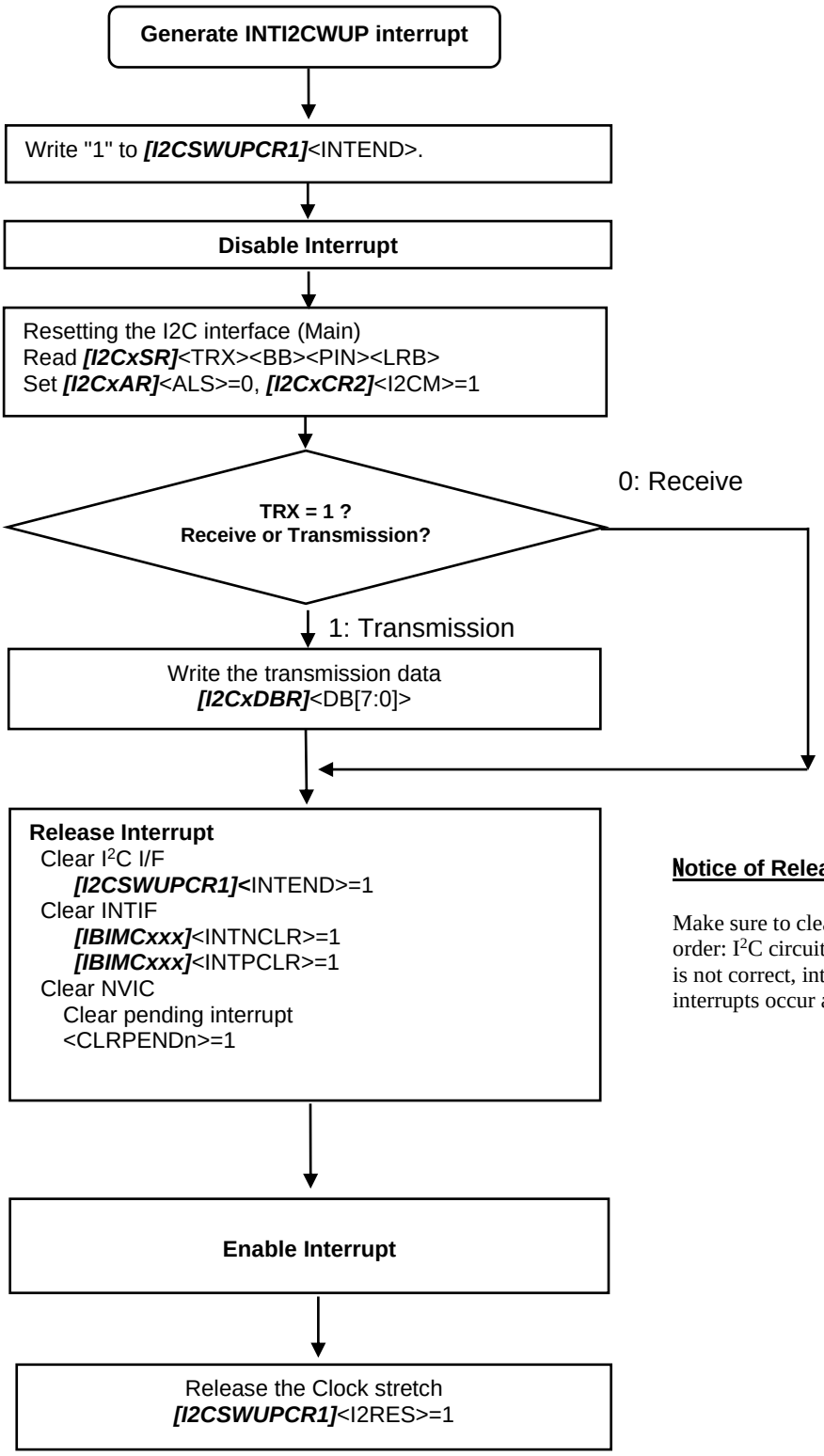
[I2CxOP]<NFSEL>,<MFACK>

[I2CxAR2]<SA2>,<SA2EN> ; any value

[I2CxIE] ; set usage function

Note: When not using the 2nd Slave address, ***[I2CxAR2]***<SA2> should be un-setting, <SA2EN> is disabled.

<When generating Interrupt>



Notice of Release interrupt

Make sure to clear the circuit along the following order: I2C circuit, INTIF, and then NVIC. If the order is not correct, interrupt factors remain; therefore, the interrupts occur again.

Figure 5.9 flow after wakeup

6. Precaution for Usage

This product does not meet the specifications among the AC electrical characteristics defined by the I^2C standard, depending on the functions of the implemented hardware.

Some items that do not meet the specifications may need to be handled by software. Please handled by the software when using the corresponding function.

The following items need to be supported by the software.

- Set-up time for a Repeated START condition($t_{SU;STA}$)

  Go and keep the set-up time by software. The corresponding item changes depending on the ***[I2CxOP]*** <SREN> setting.

  <SREN>=0: Required using the Standard mode and Fast-mode, Fast-mode Plus.

  <SREN>=1: Required using the Standard mode (In the Fast-mode & Fast-mode Plus, keeping the time by hardware)

- Bus free time between a STOP and START condition (t_{BUF})

  In all mode, go and keep the Bus free time by software.

7. Revision History

Table 7.1 Revision history

Revision	Date	Description
1.0	2017-09-11	First release
2.0	2018-03-23	Revised the Flash sentence Terms and Abbreviation: Added STD, Fm, Fm+ Revised the representation of signal level (High, H --> HIGH, Low, L -->LOW) 1.Outline: revised the Prescaler dividing selection, revised repeated start, Noise cancellation. Revised Table contents. 3.3.1: Revised the clearing condition of <BC[2:0]>. Revised Fig3.3 3.3.2: Revised the serial clock condition & tables add Table 3.5 to Table 3.12 of the Serial Clock setting table 3.3.8: Revised the register name 3.3.9: Add information of 1st slave address 3.3.10: Delete the (3) condition 3.3.12: Revised the repeated start for only using in the slave mode. 3.3.14: Revised the noise cancellation only for digital type 3.3.15: Revised the Analog noise filter when using wakeup function 3.4: Add the shift condition to the low power consumption mode. 4.2.1: Add "Note1" and revised the description of <ACK> and <NOACK>. Add "Note" for CR1. 4.2.4: Add "Note" for CR2. 4.2.6: Add "division by 2" of PRSCK[4:0] 4.2.7: Add " When the DMA is not used, use this setting" of bit6 4.2.8: Revised the description of each bit in Type W. Revised the contents of Bit3. 4.2.9: Revised the description of <SA2ST>,<SAST> and <NFSEL>, add "Note1" 4.12.12: revised Type of bit6(SGCDI) 5.1 -1: Add description about not-using DMA 5.1 -4: Revised the description of Generating of stop condition. 5.2: Revised description and Figure5.7,5.8 revised setting procedure
2.1	2018-12-07	Delete channel suffix number of I2CS. 1.: Modified the Table, Note1 & Note2 (7-bit slave addressing →7-bit addressing, 10-bit slave → 10-bit addressing, Output current → Output voltage, 3mA sink, refer to the "Product Information" of the reference manual → refer to "Electrical Characteristics" chapter of the Datasheet(DS)) 3.3.5: Revised from "writing" to "setting". 3.3.12: added the Figure 3.15, revised the Figure 3.14, modified the paragraph of Master mode condition. 4.2.7: Modified the Function of bit 6. 5.1: 6 Modified the contents in receiver mode. RESTRICTIONS revised and added the URL
2.2	2019-09-05	- Footer Layout is revised.(Copyright, Date, Rev) -2.: Revised Fig 2.1 -3.3.2: Added the symbols ($t_{HD;STA}$, $t_{SU;STO}$) and the information of parameter($t_{SU;STA}$) -3.3.15 (3): modified from ***[I2CxCR]*** to [***I2CxCR1]*** -3.4: Added the description for Wakeup Function. -4.1 Deleted Base Address of ch 5 to ch 7. -4.2.12 Note2 is modified. -5.1: Added paragraph number (5.1.1 to 5.1.8) -5.1.5: modified from ***[I2CxCR]*** to ***[I2CxCR2]*** Added the description of repeated START(added the Software method due to the <SREN>bit. Fig 5.6 revised. Fig 5.7 added) -6.: Added new paragraph as "Precaution for Usage".
2.3	2023-06-22	- 5.1.5. Procedure of Repeated START Deleted note2.

RESTRICTIONS ON PRODUCT USE

Toshiba Corporation and its subsidiaries and affiliates are collectively referred to as “TOSHIBA”.
Hardware, software and systems described in this document are collectively referred to as “Product”.

- TOSHIBA reserves the right to make changes to the information in this document and related Product without notice.
- This document and any information herein may not be reproduced without prior written permission from TOSHIBA. Even with TOSHIBA's written permission, reproduction is permissible only if reproduction is without alteration/omission.
- Though TOSHIBA works continually to improve Product's quality and reliability, Product can malfunction or fail. Customers are responsible for complying with safety standards and for providing adequate designs and safeguards for their hardware, software and systems which minimize risk and avoid situations in which a malfunction or failure of Product could cause loss of human life, bodily injury or damage to property, including data loss or corruption. Before customers use the Product, create designs including the Product, or incorporate the Product into their own applications, customers must also refer to and comply with (a) the latest versions of all relevant TOSHIBA information, including without limitation, this document, the specifications, the data sheets and application notes for Product and the precautions and conditions set forth in the "TOSHIBA Semiconductor Reliability Handbook" and (b) the instructions for the application with which the Product will be used with or for. Customers are solely responsible for all aspects of their own product design or applications, including but not limited to (a) determining the appropriateness of the use of this Product in such design or applications; (b) evaluating and determining the applicability of any information contained in this document, or in charts, diagrams, programs, algorithms, sample application circuits, or any other referenced documents; and (c) validating all operating parameters for such designs and applications. **TOSHIBA ASSUMES NO LIABILITY FOR CUSTOMERS' PRODUCT DESIGN OR APPLICATIONS.**
- **PRODUCT IS NEITHER INTENDED NOR WARRANTED FOR USE IN EQUIPMENTS OR SYSTEMS THAT REQUIRE EXTRAORDINARILY HIGH LEVELS OF QUALITY AND/OR RELIABILITY, AND/OR A MALFUNCTION OR FAILURE OF WHICH MAY CAUSE LOSS OF HUMAN LIFE, BODILY INJURY, SERIOUS PROPERTY DAMAGE AND/OR SERIOUS PUBLIC IMPACT ("UNINTENDED USE").** Except for specific applications as expressly stated in this document, Unintended Use includes, without limitation, equipment used in nuclear facilities, equipment used in the aerospace industry, lifesaving and/or life supporting medical equipment, equipment used for automobiles, trains, ships and other transportation, traffic signaling equipment, equipment used to control combustions or explosions, safety devices, elevators and escalators, and devices related to power plant. **IF YOU USE PRODUCT FOR UNINTENDED USE, TOSHIBA ASSUMES NO LIABILITY FOR PRODUCT.** For details, please contact your TOSHIBA sales representative or contact us via our website.
- Do not disassemble, analyze, reverse-engineer, alter, modify, translate or copy Product, whether in whole or in part.
- Product shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable laws or regulations.
- The information contained herein is presented only as guidance for Product use. No responsibility is assumed by TOSHIBA for any infringement of patents or any other intellectual property rights of third parties that may result from the use of Product. No license to any intellectual property right is granted by this document, whether express or implied, by estoppel or otherwise.
- **ABSENT A WRITTEN SIGNED AGREEMENT, EXCEPT AS PROVIDED IN THE RELEVANT TERMS AND CONDITIONS OF SALE FOR PRODUCT, AND TO THE MAXIMUM EXTENT ALLOWABLE BY LAW, TOSHIBA (1) ASSUMES NO LIABILITY WHATSOEVER, INCLUDING WITHOUT LIMITATION, INDIRECT, CONSEQUENTIAL, SPECIAL, OR INCIDENTAL DAMAGES OR LOSS, INCLUDING WITHOUT LIMITATION, LOSS OF PROFITS, LOSS OF OPPORTUNITIES, BUSINESS INTERRUPTION AND LOSS OF DATA, AND (2) DISCLAIMS ANY AND ALL EXPRESS OR IMPLIED WARRANTIES AND CONDITIONS RELATED TO SALE, USE OF PRODUCT, OR INFORMATION, INCLUDING WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, ACCURACY OF INFORMATION, OR NONINFRINGEMENT.**
- Do not use or otherwise make available Product or related software or technology for any military purposes, including without limitation, for the design, development, use, stockpiling or manufacturing of nuclear, chemical, or biological weapons or missile technology products (mass destruction weapons). Product and related software and technology may be controlled under the applicable export laws and regulations including, without limitation, the Japanese Foreign Exchange and Foreign Trade Law and the U.S. Export Administration Regulations. Export and re-export of Product or related software or technology are strictly prohibited except in compliance with all applicable export laws and regulations.
- Please contact your TOSHIBA sales representative for details as to environmental matters such as the RoHS compatibility of Product. Please use Product in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. **TOSHIBA ASSUMES NO LIABILITY FOR DAMAGES OR LOSSES OCCURRING AS A RESULT OF NONCOMPLIANCE WITH APPLICABLE LAWS AND REGULATIONS.**