Chapter 33 I2C Interface

33.1 Overview

The Inter-Integrated Circuit (I2C) is a two wired (SCL and SDA), bi-directional serial bus that provides an efficient and simple method of information exchange between devices. This I2C bus controller supports master mode acting as a bridge between AMBA protocol and generic I2C bus system.

33.1.1 Features

I2C Controller supports the following features:

- Item Compatible with I2C-bus
- AMBA APB slave interface
- Supports master mode of I2C bus
- Software programmable clock frequency and transfer rate up to 400Kbit/sec
- Supports 7 bits and 10 bits addressing modes
- Interrupt or polling driven multiple bytes data transfer
- Clock stretching and wait state generation
- I2C0 / I2C1 / I2C2 /I2C3 are in peripheral sub-system

33.2 Block Diagram

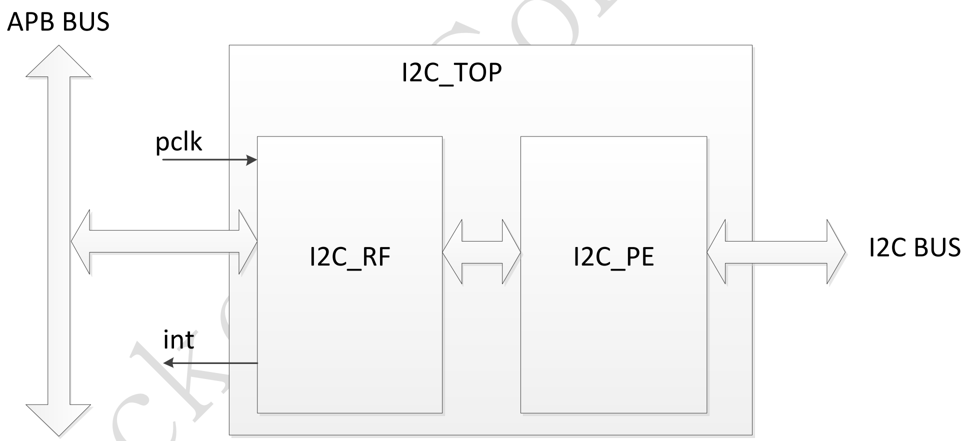

Fig.33-1I2C architecture

I2C_RF

I2C_RF module is used to control the I2C controller operation by the host with APB interface. It implements the register set and the interrupt functionality. The CSR component operates synchronously with the pclk clock.

I2C_PE

I2C_PE module implements the I2C master operation for transmit data to and receive data from other I2C devices. The I2C master controller operates synchronously with the pclk.

I2C_TOP

I2C_TOP module is the top module of the I2C controller.

33.3 Function description

This chapter provides a description about the functions and behavior under various conditions.

The I2C controller supports only Masterfunction. Itsupports the 7-bits/10-bits addressing mode and support general call address. The maximum clock frequency and transfer rate can be up to 400Kbit/sec.

The operations of I2C controller is divided to 2 parts and described separately: initialization and master mode programming.

33.3.1 Initialization

The I2C controller is based on AMBA APB bus architecture and usually is part of a SOC. So before I2C operates, some system setting & configuration must be conformed, which includes:

- I2C interrupt connection type: CPU interrupt scheme should be considered. If the I2C interrupt is connected to extra Interrupt Controller module, we need decide the INTC vector.
- I2C Clock Rate: The I2C controller uses the APB clock as the system clock so the APB clock will determine the I2C bus clock. The correct register setting is subject to the system requirement.

33.3.2 Master Mode Programming

1. SCL Clock: When the I2C controller is programmed in Master mode, the SCL frequency is determine by I2C_CLKDIV register. The SCL frequency is calculated by the following formula：

SCL Divisor = 8*((CLKDIVL+1) + (CLKDIVH+1))

SCL = PCLK/ SCLK Divisor

2. Data Receiver Register Access

When the i2c controller received MRXCNT bytes data， CPU can get the datas through register RXDATA0 ~ RXDATA7. The controller can receive up to 32 byte data in one transaction.

When MRXCNT register is written, the I2C controller will start to drive SCL to receive data.

3. Transmit Trasmitter Register

Data to transmit are written to TXDATA0~7 by CPU. The controller can transmit up to 32 byte data in one transaction. The lower byte will be transmitted first.

When MTXCNT register is written, the I2C controller will start to transmit data.

4. Start Command

Write 1 to I2C_CON[3], the controller will send I2C start command.

5. Stop Command

Write 1 to I2C_CON[4], the controller will send I2C stop command

6. I2C Operation mode

There are four i2c operation modes.

When I2C_CON[2:1] is 2’b00, the controller transmit all valid data in

TXDATA0~TXDATA7 byte by byte. The controller will transmit lower byte first.

When I2C_CON[2:1] is 2'b01,the controller will transmit device address in MRXADDR first (Write/Read bit = 0) and then transmit device register address in MRXRADDR. After that, the controller will assert restart signal and resend MRXADDR (Write/Read bit = 1). At last, the controller enter receive mode.

When I2C_CON[2:1] is 2'b10, the controller is in receive mode, it will triggered clock to read MRXCNT byte data.

When I2C_CON[2:1] is 2'b11,the controller will transmit device address in MRXADDR first (Write/Read bit = 1) and then transmit device register address in MRXRADDR . After that, the controller will assert restart signal and resend MRXADDR (Write/Read bit = 1). At last, the controller enter receive mode.

7. Read/Write Command

When I2C_OPMODE(I2C_CON[2:1]) is 2'b01 or 2'b11, the Read/Write command bit is decided by controller itself.

In RX only mode (I2C_CON[2:1] is 2'b10), the Read/Write command bit is decided by MRXADDR[0].

In TX only mode (I2C_CON[[2:1] is 2'b00), the Read/Write command bit is decided by TXDATA[0].

8. Master Interrupt Condition

There are 7 interrupt bits in I2C_ISR register related to master mode.

Byte transmitted finish interrupt (Bit 0): The bit is asserted when Master complete transmitting a byte.

Byte received finish interrupt (Bit 1): The bit is asserted when Master complete receiving a byte.

MTXCNT bytes data transmitted finish interrupt (Bit 2): The bit is asserted when Master complete transmitting MTXCNT bytes.

MRXCNT bytes data received finish interrupt (Bit 3): The bit is asserted when Master complete receiving MRXCNT bytes.

Start interrupt(Bit 4): The bit is asserted when Master finish asserting start command to I2C bus.

Stop interrupt (Bit 5): The bit is asserted when Master finish asserting stop command to I2C bus.

NAK received interrupt (Bit 6): The bit is asserted when Master received a NAK handshake.

9. Last byte acknowledge control

If I2C_CON[5] is 1, the I2C controller will transmit NAK handshake to slave when the last byte received in RX only mode.

If I2C_CON[5] is 0, the I2C controller will transmit ACK handshake to slave when the last byte received in RX only mode.

10. How to handle nak handshake received

If I2C_CON[6] is 1, the I2C controller will stop all transactions when NAK handshake received. And the software should take responsibility to handle the problem.

If I2C_CON[6] is 0, the I2C controller will ignore all NAK handshake received.

11. I2C controller data transfer waveform

- **Bit transferring**

(a) Data Validity

The SDA line must be stable during the high period of SCL, and the data on SDA line can only be changed when SCL is in low state.

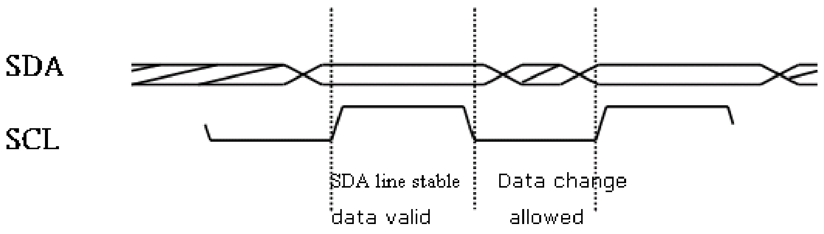

Fig.33-2I2C DATA Validity

(b) START and STOP conditions

START condition occurs when SDA goes low while SCL is in high period. STOP condition is generated when SDA line goes high while SCL is in high state.

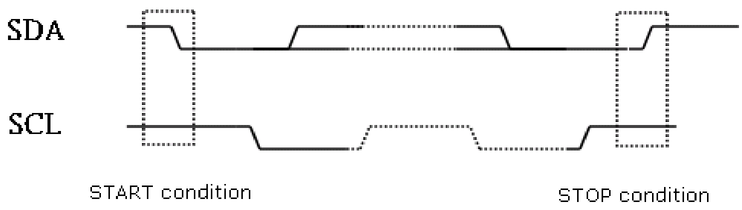

Fig.33-3I2C Start and stop conditions

- **Data transfer**

(a) Acknowledge

After a byte of data transferring (clocks labeled as 1~8), in 9^{th} clock the receiver must assert an ACK signal on SDA line, if the receiver pulls SDA line to low, it means "ACK", on the contrary, it's "NOT ACK".

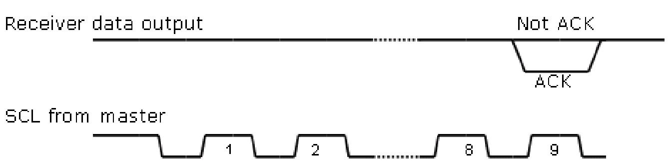

Fig.33-4I2C Acknowledge

(b) Byte transfer

The master own I2C bus might initiate multi byte to transfer to a slave. The transfer starts from a "START" command and ends in a "STOP"command. After every byte transfer, the receiver must reply an ACK to transmitter.

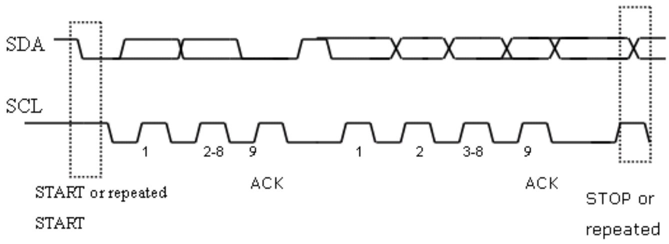

Fig.33-5I2C byte transfer

33.4 Register Description

This section describes the control/status registers of the design.

33.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
I2C_CON	0x0000	W	0x00000000	control register
I2C_CLKDIV	0x0004	W	0x00060006	Clock divisor register
I2C_MRXADDR	0x0008	W	0x00000000	the slave address accessed for master receive mode
I2C_MRXRADDR	0x000c	W	0x00000000	the slave register address accessed for master recevie mode
I2C_MTXCNT	0x0010	W	0x00000000	master transmit count
I2C_MRXCNT	0x0014	W	0x00000000	master receive count
I2C_IEN	0x0018	W	0x00000000	interrupt enable register
I2C_IPD	0x001c	W	0x00000000	interrupt pending register
I2C_FCNT	0x0020	W	0x00000000	finished count
I2C_TXDATA0	0x0100	W	0x00000000	I2C transmit data register 0
I2C_TXDATA1	0x0104	W	0x00000000	I2C transmit data register 1
I2C_TXDATA2	0x0108	W	0x00000000	I2C transmit data register 2
I2C_TXDATA3	0x010c	W	0x00000000	I2C transmit data register 3
I2C_TXDATA4	0x0110	W	0x00000000	I2C transmit data register 4
I2C_TXDATA5	0x0114	W	0x00000000	I2C transmit data register 5
I2C_TXDATA6	0x0118	W	0x00000000	I2C transmit data register 6
I2C_TXDATA7	0x011c	W	0x00000000	I2C transmit data register 7
I2C_RXDATA0	0x0200	W	0x00000000	I2C receive data register 0
I2C_RXDATA1	0x0204	W	0x00000000	I2C receive data register 1
I2C_RXDATA2	0x0208	W	0x00000000	I2C receive data register 2
I2C_RXDATA3	0x020c	W	0x00000000	I2C receive data register 3
I2C_RXDATA4	0x0210	W	0x00000000	I2C receive data register 4
I2C_RXDATA5	0x0214	W	0x00000000	I2C receive data register 5
I2C_RXDATA6	0x0218	W	0x00000000	I2C receive data register 6
I2C_RXDATA7	0x021c	W	0x00000000	I2C receive data register 7

*Notes:Size:**B**- Byte (8 bits) access, **HW**- Half WORD (16 bits) access, **W**-WORD (32 bits) access*

33.4.2 Detail Register Description

I2C_CON
Address: Operational Base + offset (0x0000)
control register

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:7	RO	0x0	reserved
6	RW	0x0	act2nak operation when NAK handshake is received 1'b0: ignored 1'b1: stop transaction
5	RW	0x0	ack last byte acknowledge control last byte acknowledge control in master receive mode . 1'b0: ACK 1'b1: NAK
4	W1C	0x0	stop stop enable when this bit is written to 1，I2C will generate stop signal. It cleared itself when stop operation ends.
3	W1C	0x0	start start enable when this bit is written to 1，I2C will generate start signal. It cleared itself when start operation ends.
2:1	RW	0x0	i2c_mode 2'b00: transmit only 2'b01: transmit address (device + register address) --> restart -->transmit address –>receive only 2'b10:receive only 2'b11: transmit address (device + register address, write/read bit is 1) --> restart -->transmit address (device address) -->receive data
0	RW	0x0	i2c_en i2c module enable 1: i2c is enabled. 0: i2c is disabled.

I2C_CLKDIV
Address: Operational Base + offset (0x0004)
Clock divisor register

Bit	Attr	Reset Value	Description
31:16	RW	0x0006	CLKDIVH SCL high level clock count T（SCL_HIGH）= T(PCLK) *(CLKDIVH+1) * 8
15:0	RW	0x0006	CLKDIVL SCL low level clock count T（SCL_LOW）= T(PCLK) *(CLKDIVL+1) * 8

I2C_MRXADDR
Address: Operational Base + offset (0x0008)
the slave address accessed for master receive mode

Bit	Attr	Reset Value	Description
31:27	RO	0x0	reserved
26	RW	0x0	addhvld address high byte valid
25	RW	0x0	addmvld address middle byte valid
24	RW	0x0	addlvld address low byte valid
23:0	RW	0x000000	saddr master address register the lowest bit indicate write or read

I2C_MRXRADDR

Address: Operational Base + offset (0x000c)

the slave register address accessed for master receive mode

Bit	Attr	Reset Value	Description
31:27	RO	0x0	reserved
26	RW	0x0	sraddhvld address high byte valid
25	RW	0x0	sraddmvld address middle byte valid
24	RW	0x0	sraddlvld address low byte valid
23:0	RW	0x000000	sraddr slave register address accessed

I2C_MTXCNT

Address: Operational Base + offset (0x0010)

master transmit count

Bit	Attr	Reset Value	Description
31:6	RO	0x0	reserved
5:0	RW	0x00	mtxcnt master transmit count

I2C_MRXCNT

Address: Operational Base + offset (0x0014)

masterreceive count

Bit	Attr	Reset Value	Description
31:6	RO	0x0	reserved
5:0	RW	0x00	mrxcnt master receive count

I2C_IEN

Address: Operational Base + offset (0x0018)

interrupt enable register

Bit	Attr	Reset Value	Description
31:7	RO	0x0	reserved
6	RW	0x0	nakrcvien NAK handshake received interrupt enable

Bit	Attr	Reset Value	Description
5	RW	0x0	stopien stop operation finished interrupt enable
4	RW	0x0	startien start operation finished interrupt enable
3	RW	0x0	mbrfien MRXCNT data received finished interrupt enable
2	RW	0x0	mbtfien MTXCNT data transmit finished interrupt enable
1	RW	0x0	brfien byte receive finished interrupt enable
0	RW	0x0	btfien byte transmit finished interrupt enable

I2C_IPD
Address: Operational Base + offset (0x001c)
interrupt pending register

Bit	Attr	Reset Value	Description
31:7	RO	0x0	reserved
6	RW	0x0	nakrcvipd NAK handshake received interrupt pending bit
5	RW	0x0	stopipd stop operation finished interrupt pending bit
4	RW	0x0	startipd start operation finished interrupt pending bit
3	RW	0x0	mbrfipd MRXCNT data received finished interrupt pending bit
2	RW	0x0	mbtfipd MTXCNT data transmitfinished interrupt pending bit
1	RW	0x0	brfipd byte receivefinished interrupt pending bit
0	RW	0x0	btfipd byte transmitfinished interrupt pending bit

I2C_FCNT
Address: Operational Base + offset (0x0020)
finished count

Bit	Attr	Reset Value	Description
31:6	RO	0x0	reserved
5:0	RW	0x00	fcnt finished count the count of data which has been transmitted or received for debug purpose

I2C_TXDATA0

Address: Operational Base + offset (0x0100)
I2C transmit data register 0

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	txdata0

I2C_TXDATA1
Address: Operational Base + offset (0x0104)
I2C transmit data register 1

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	txdata1

I2C_TXDATA2
Address: Operational Base + offset (0x0108)
I2C transmit data register 2

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	txdata2

I2C_TXDATA3
Address: Operational Base + offset (0x010c)
I2C transmit data register 3

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	txdata3

I2C_TXDATA4
Address: Operational Base + offset (0x0110)
I2C transmit data register 4

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	txdata4

I2C_TXDATA5
Address: Operational Base + offset (0x0114)
I2C transmit data register 5

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	txdata5

I2C_TXDATA6
Address: Operational Base + offset (0x0118)
I2C transmit data register 6

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	txdata6

I2C_TXDATA7
Address: Operational Base + offset (0x011c)
I2C transmit data register 7

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	txdata7

I2C_RXDATA0
Address: Operational Base + offset (0x0200)
I2C receive data register 0

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	rxdata0

I2C_RXDATA1
Address: Operational Base + offset (0x0204)
I2C receive data register 1

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	rxdata1

I2C_RXDATA2
Address: Operational Base + offset (0x0208)
I2C receive data register 2

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	rxdata2

I2C_RXDATA3
Address: Operational Base + offset (0x020c)
I2C receive data register 3

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	rxdata3

I2C_RXDATA4
Address: Operational Base + offset (0x0210)
I2C receive data register 4

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	rxdata4

I2C_RXDATA5
Address: Operational Base + offset (0x0214)
I2C receive data register 5

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	rxdata5

I2C_RXDATA6
Address: Operational Base + offset (0x0218)
I2C receive data register 6

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	rxdata6

I2C_RXDATA7
Address: Operational Base + offset (0x021c)

I2C receive data register 7

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	rxdata7

33.5 Interface description

Table 33-1I2C Interface Description

Module pin	Direction	Pad name	IOMUX
I2C0 Interface			
i2c0_sda	I/O	IO_I2C0pmusda_GPIO0a1	GRF_GPIO0A_IOMUX[3:2]=2'b01
i2c0_scl	I/O	IO_I2C0pmuscl_GPIO0a0	GRF_GPIO0A_IOMUX[1:0]=2'b01
I2C1 Interface			
i2c1_sda	I/O	IO_I2C1tpsda_GPIO0a3	GRF_GPIO0A_IOMUX[7:6]=2'b01
i2c1_scl	I/O	IO_I2C1tpscl_GPIO0a2	GRF_GPIO0A_IOMUX[5:4]=2'b01
I2C2 Interface			
i2c2_sda	I/O	IO_LCD0d18_EBCgdrl_I2C2sda_GPIO2c4	GRF_GPIO2C_IOMUX[9:8]=2'b11
i2c2_scl	I/O	IO_LCD0d19_EBCsdshr_I2C2scl_GPIO2c5	GRF_GPIO2C_IOMUX[11:10]=2'b11
I2C3 Interface			
i2c3_sda	I/O	IO_I2C3cifsda_HDMIddcsda_GPIO0a7	GRF_GPIO0A_IOMUX[15:14]=2'b01
i2c3_scl	I/O	IO_I2C3cifscl_HDMIddcscl_GPIO0a6	GRF_GPIO0A_IOMUX[13:12]=2'b01

33.6 Application Notes

The I2C controller core operation flow chart below is to describe how the software configures and performs an I2C transaction through this I2C controller core. Descriptions are divided into 3 sections, transmit only mode, receive only mode, and mix mode. Users are strongly advised to following.

- Transmit only mode (I2C_CON[1:0]=2'b00)

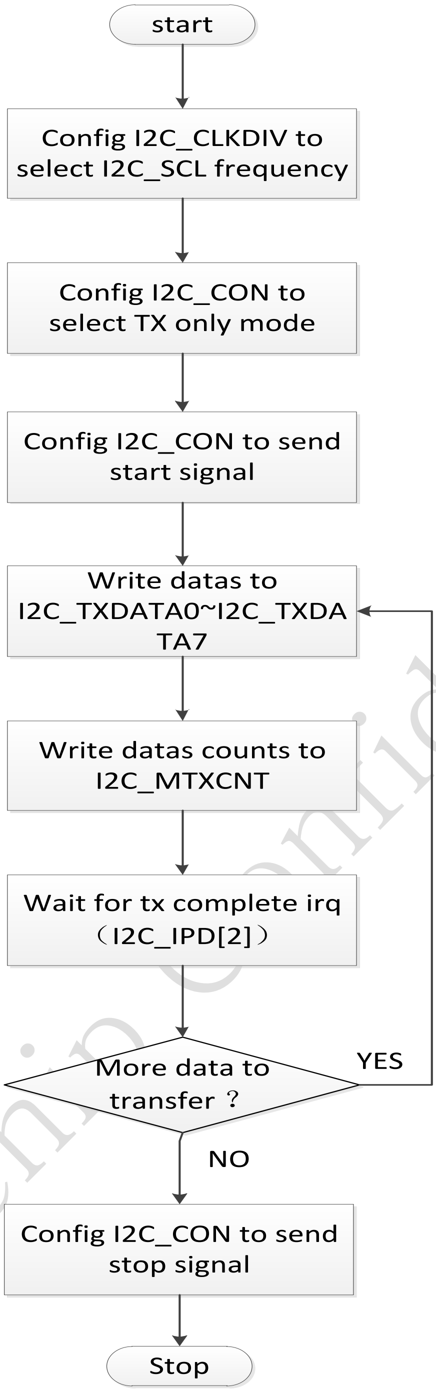

Fig.33-6I2C Flow chat for transmit only mode

- Receive only mode (I2C_CON[1:0]=2'b10)

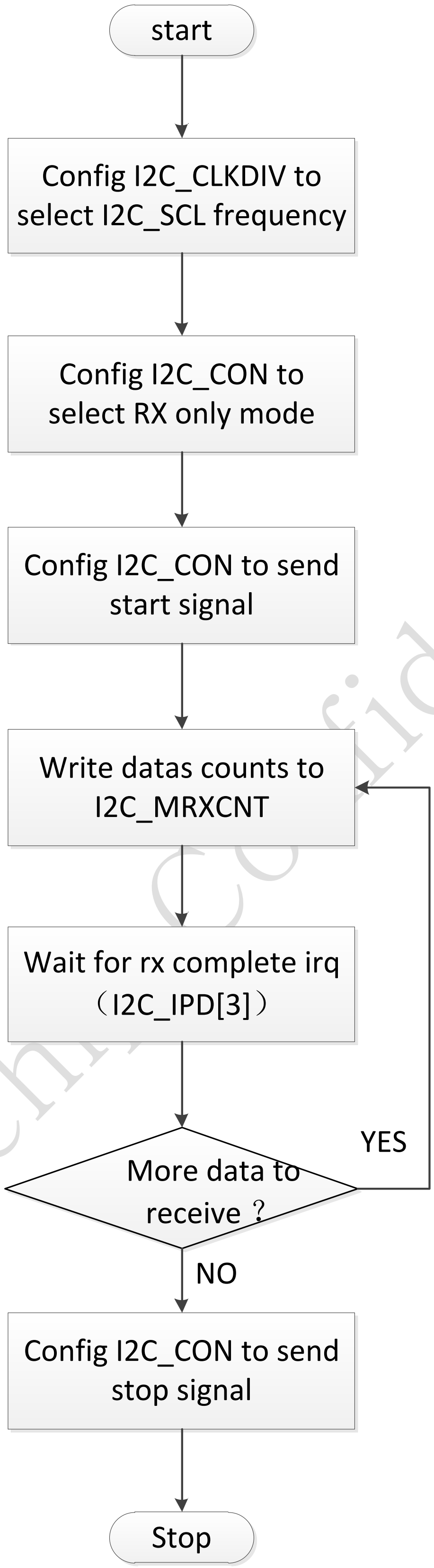

Fig.33-7I2C Flow chat for receive only mode

- Mix mode (I2C_CON[1:0]=2’b01 or I2C_CON[1:0]=2’b11)

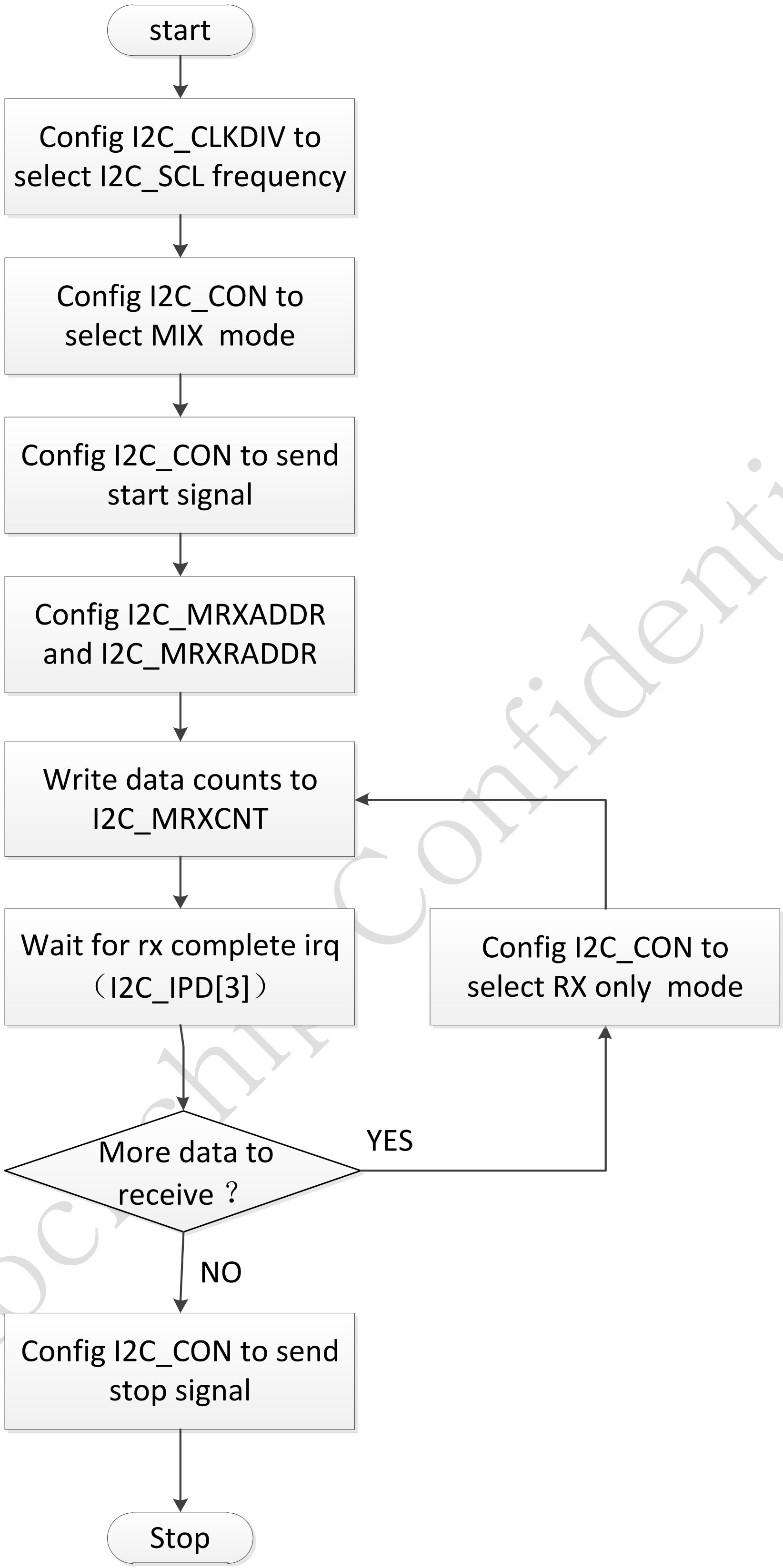

Fig.33-8I2C Flow chat for mix mode