Application Note for FT5426 - 5526 CTPM

Application Note for FT5426 - 5526CTPM	
Project name	Touch panel
Document ref	[Document ref]
Version	0.1
Release date	2015.9.06
Owner	B.F.Lu
Classification	
Distribution List	
Approval	

3/F,Kingdom Sci-Tech Building,
5th Gaoxinnan Avenue, Hi-Tech Park,
Nanshan District ,Shenzhen, Gungdong, P.R. China

ZIP :518057
T +86 755 26588222
F +86 755 26712499
E support@focaltech-systems.com

www.focaltech-systems.com

Revision History

Date	Version	List of changes	Author	Approved by
2015.09.06	0.1	Initial draft.	Lu bingfeng	

目录

Terminology

CTP	Capacitive touch panel
CTPM	Capacitive touch panel module
TX	Transmitter
RX	Receiver

1 CTPM interface to Host

Figure 1-1 shows how CTPM communicates with host device. I^{2}C interface supported by FT5426-5526 that is two-wire serial bus consisting of data line SDA and clock line SCL, used for serial data transferring between host and slave device.

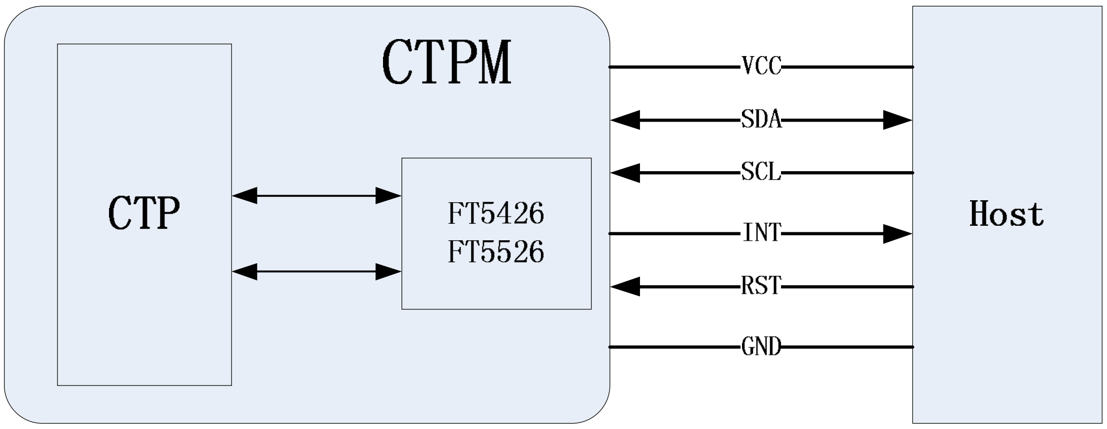


Figure 1-1 CTPM and Host connection

INT port and RST port form the control interface. The INT port is controlled by FT5426-5526, it will send out an interrupt request signal to the host when there is a valid touch on CTPM. Host can send the reset signal to CTPM via RST port to reset the FT5426-5526 if needed. The Power Supply voltage of CTPM ranges from 2.8V to 3.6V. For details, please refer to Table 1-1.

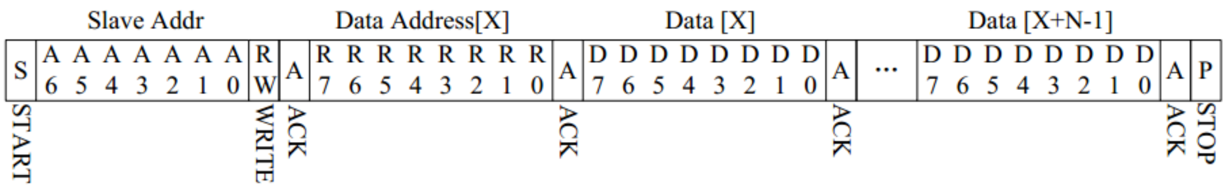
Table 1-1 Description for CTPM and Host interface

Port Name	Description
VCC	CTPM power supply, ranges from 2.8V to 3.6V.
SDA	I^{2}C data input and output.
SCL	I^{2}C clock input.
INT	The interrupt request signal from CTPM to Host.
RST	The reset signal from host to CTPM, active low, and the low pulse width should be more than 1ms.
GND	Power ground.

1.1 I2C Read/Write Interface description

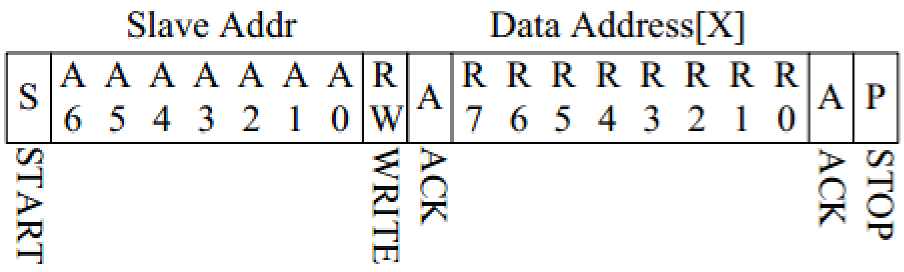
It is important to note that the SDA and SCL must connect with a pull-high resistor respectively before you read/write I^{2}C data.

1.1.1 Host write data to slave

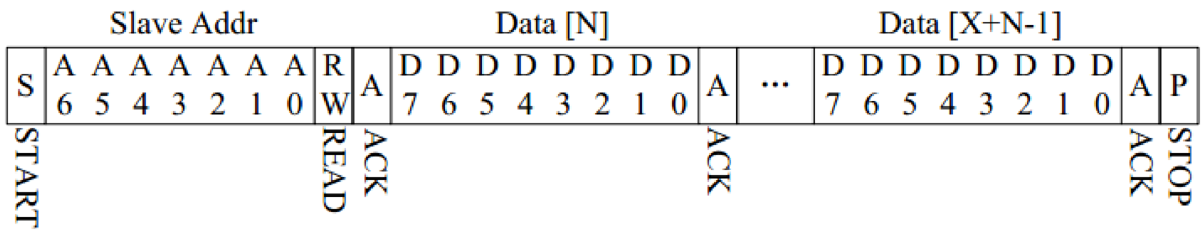


1.1.2 Host read data from slave

Step1: write data address



Step2: read data



1.2 Interrupt signal from CTPM to Host

As for standard CTPM, host needs to use both interrupt signal and I2C interface to get the touch data. CTPM will output an interrupt request signal to the host when there is a valid touch. Then host can get the touch data via I2C interface. If there is no valid touch detected, the INT will output high level, and the host does not need to read the touch data. There are two kinds of method to use interrupt: interrupt trigger and interrupt polling.

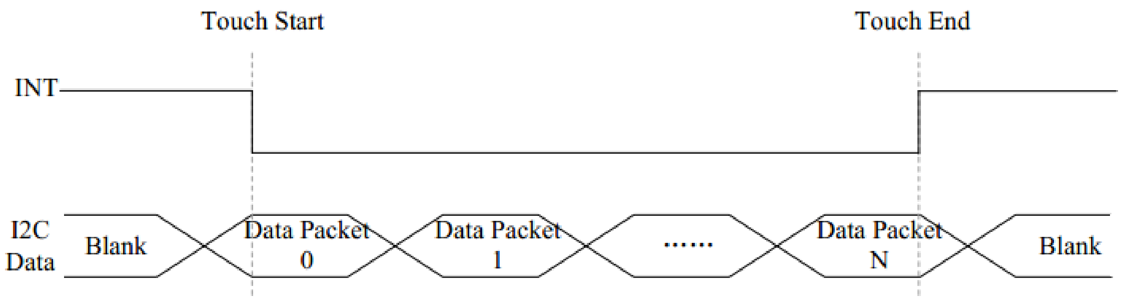


Figure 1-2 Interrupt polling mode

As for interrupt polling mode, INT will always be pulled to low level when there is a valid touch point, and be high level when a touch finished.

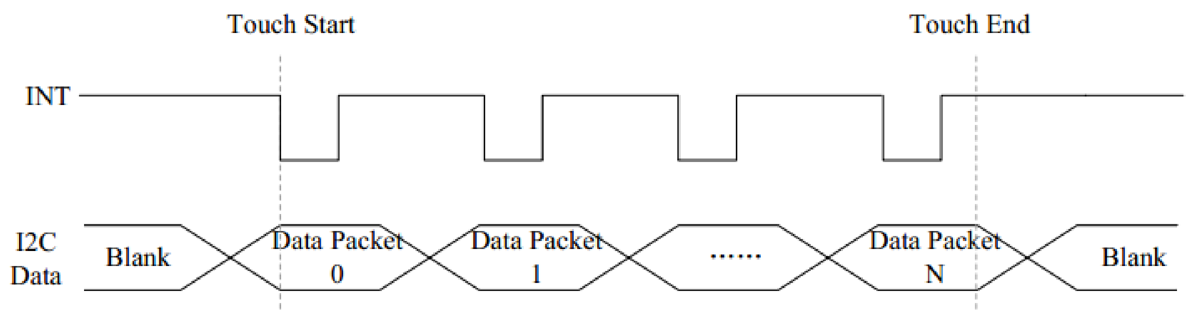


Figure 1-3 Interrupt trigger mode

While for interrupt trigger mode, INT signal will be set to low if there is a touch detected. But whenever an update of valid touch data, CTPM will produce a valid pulse on INT port for INT signal, and host can read the touch data periodically according to the frequency of this pulse. In this mode, the pulse frequency is the touch data updating rate

1.3 Reset signal from Host to CTPM.

Host can send the reset signal via RST port to reset FT5426-5526. The reset signal should not be set to low while in normal working mode. The RST port can also be used to active the CTPM in hibernate mode. Note that the reset pulse width should be more than 1ms.

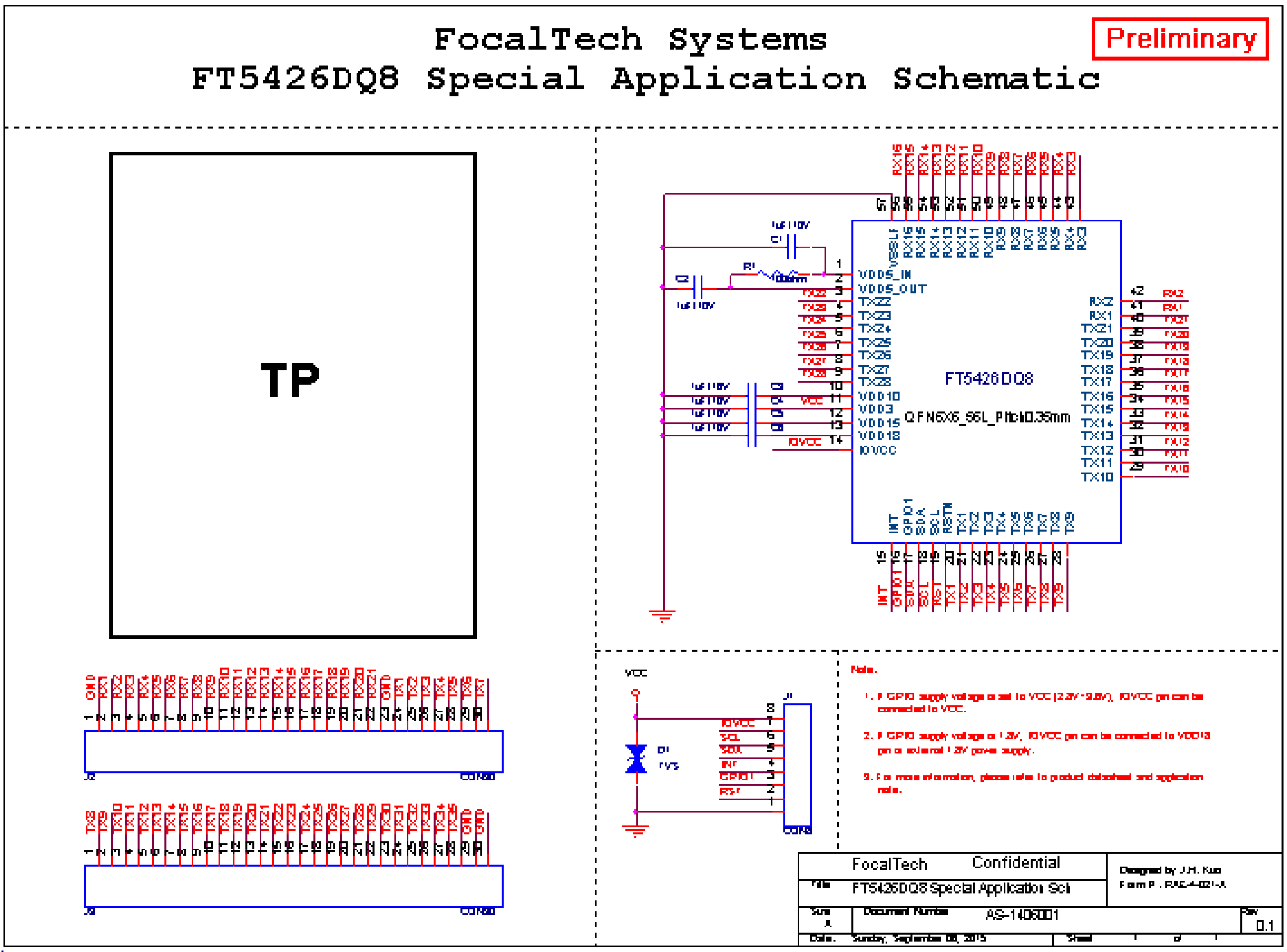
2 Standard Application Circuit

Table 2-1 is a brief summary of the FT5426-5526 application features.

Table 2-1 Brief features of FT5426-5526

IC Type	FT5426DQ8	FT5526EEZ
Operating Voltage(V)	2.8 ~ 3.6	2.8 ~ 3.6
Channel	28 TX + 16 RX	35 TX + 21 RX
Panel Size	≤8”	≤10.1”
Touch points	10	10
Interface	I^2C	I^2C
Report rate	>100Hz	>100Hz
Package (mm)	QFN 56L 6x6x0.6mm Pitch =0.35mm	QFN 68L 8x8x0.8mm Pitch =0.4mm

2.1 FT5426DQ8 typical application schematic

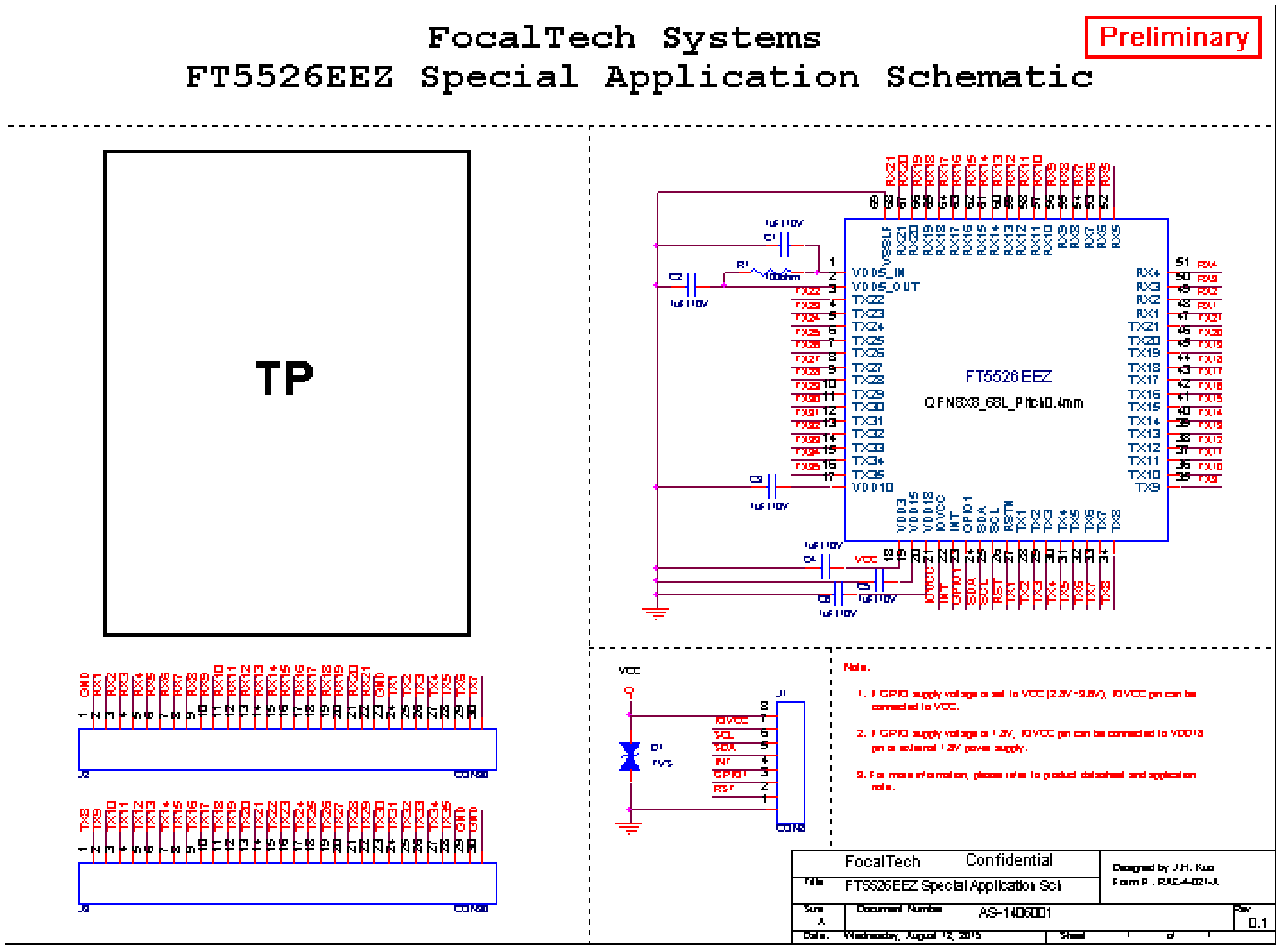


Note:

1. If GPIO supply voltage is set to VCC (2.8V~3.6V), IOVCC pin can be connected to VCC.
2. If GPIO supply voltage is 1.8V, IOVCC pin can be connected to VDD18 pin or external 1.8V power supply.
3. For more information, please refer to product datasheet and application note.

FocalTech Confidential		Designed by J.H. Kuo	
Title	FT5426DQ8 Special Application Sch	Form P. RAE-4-021-A	
Size A	Document Number AS-1406001		Rev 0.1
Date:	Sunday, September 06, 2015	Sheet 1 of 1	

2.2 FT5526EEZ typical application schematic



Note:

1. If GPIO supply voltage is set to VCC (2.8V~3.6V), IOVCC pin can be connected to VCC.
2. If GPIO supply voltage is 1.8V, IOVCC pin can be connected to VDD18 pin or external 1.8V.

3 CTPM Register Mapping

This chapter describes the standard CTPM communication registers in address order for working mode.

3.1 Working Mode

The CTP is fully functional as a touch screen controller in working mode. The access address to read and write is just logical address which is not enforced by hardware. Here is the working mode register map.

Register Map [Working Mode]

ADDR	RW	Name	b7	b6	b5	b4	b3	b2	b1	b0
0x00	RW	Mode_Switch		Device Mode[2:0]						
0x01	RO	Guesture	Gesture ID [7:0]							
0x02	RO	Cur Point	Number of touch points[7:0]							
0x03	RO	TOUCH1_XH	1st Event Flag				1st Touch X Position[11:8]			
0x04	RO	TOUCH1_XL	1st Touch X Position[7:0]							
0x05	RO	TOUCH1_YH	1st Touch ID[3:0]				1st Touch Y Position[11:8]			
0x06	RO	TOUCH1_YL	1st Touch Y Position[7:0]							
0x07	RO	TOUCH1_WEIGHT	1st Touch Weight[7:0]							
0x08	RO	TOUCH1_MISC	1st Touch Area[3:0]							
0x09	RO	TOUCH2_XH	2nd Event Flag				2nd Touch X Position[11:8]			
0x0A	RO	TOUCH2_XL	2nd Touch X Position[7:0]							
0x0B	RO	TOUCH2_YH	2nd Touch ID[3:0]				2nd Touch Y Position[11:8]			
0x0C	RO	TOUCH2_YL	2nd Touch Y Position[7:0]							
0x0D	RO	TOUCH2_WEIGHT	2nd Touch Weight[7:0]							
0x0E	RO	TOUCH2_MISC	2nd Touch Area[3:0]							
0x0F	RO	TOUCH3_XH	3rd Event Flag				3rd Touch X Position[11:8]			
0x10	RO	TOUCH3_XL	3rd Touch X Position[7:0]							
0x11	RO	TOUCH3_YH	3rd Touch ID[3:0]				3rd Touch Y Position[11:8]			
0x12	RO	TOUCH3_YL	3rd Touch Y Position[7:0]							
0x13	RO	TOUCH3_WEIGHT	3rd Touch Weight[7:0]							
0x14	RO	TOUCH3_MISC	3rd Touch Area[3:0]							
0x15	RO	TOUCH4_XH	4th Event Flag				4th Touch X Position[11:8]			
0x16	RO	TOUCH4_XL	4th Touch X Position[7:0]							
0x17	RO	TOUCH4_YH	4th Touch ID[3:0]				4th Touch Y Position[11:8]			
0x18	RO	TOUCH4_YL	4th Touch Y Position[7:0]							
0x19	RO	TOUCH4_WEIGHT	4th Touch Weight[7:0]							
0x1A	RO	TOUCH4_MISC	4th Touch Area[3:0]							

0x1B	RO	TOUCH5_XH	5th Event Flag		5th Touch X Position[11:8]	
0x1C	RO	TOUCH5_XL	5th Touch X Position[7:0]			
0x1D	RO	TOUCH5_YH	5th Touch ID[3:0]		5th Touch Y Position[11:8]	
0x1E	RO	TOUCH5_YL	5th Touch Y Position[7:0]			
0x1F	RO	TOUCH5_WEIGHT	5th Touch Weight[7:0]			
0x20	RO	TOUCH5_MISC	5th Touch Area[3:0]			
0x21	RO	TOUCH6_XH	6th Event Flag		6th Touch X Position[11:8]	
0x22	RO	TOUCH6_XL	6st Touch X Position[7:0]			
0x23	RO	TOUCH6_YH	6st Touch ID[3:0]		6st Touch Y Position[11:8]	
0x24	RO	TOUCH6_YL	6st Touch Y Position[7:0]			
0x25	RO	TOUCH6_WEIGHT	6st Touch Weight[7:0]			
0x26	RO	TOUCH6_MISC	6st Touch Area[3:0]			
0x27	RO	TOUCH7_XH	7th Event Flag		7th Touch X Position[11:8]	
0x28	RO	TOUCH7_XL	7st Touch X Position[7:0]			
0x29	RO	TOUCH7_YH	7st Touch ID[3:0]		7st Touch Y Position[11:8]	
0x2A	RO	TOUCH7_YL	7st Touch Y Position[7:0]			
0x2B	RO	TOUCH7_WEIGHT	7st Touch Weight[7:0]			
0x2C	RO	TOUCH7_MISC	7st Touch Area[3:0]			
0x2D	RO	TOUCH8_XH	8th Event Flag		8st Touch X Position[11:8]	
0x2E	RO	TOUCH8_XL	8st Touch X Position[7:0]			
0x2F	RO	TOUCH8_YH	8st Touch ID[3:0]		8st Touch Y Position[11:8]	
0x30	RO	TOUCH8_YL	8st Touch Y Position[7:0]			
0x31	RO	TOUCH8_WEIGHT	8st Touch Weight[7:0]			
0x32	RO	TOUCH8_MISC	8st Touch Area[3:0]			
0x33	RO	TOUCH9_XH	9th Event Flag		9st Touch X Position[11:8]	
0x34	RO	TOUCH9_XL	9st Touch X Position[7:0]			
0x35	RO	TOUCH9_YH	9st Touch ID[3:0]		9st Touch Y Position[11:8]	
0x36	RO	TOUCH9_YL	9st Touch Y Position[7:0]			
0x37	RO	TOUCH9_WEIGHT	9st Touch Weight[7:0]			
0x38	RO	TOUCH9_MISC	9st Touch Area[3:0]			
0x39	RO	TOUCH10_XH	10th Event Flag		10st Touch X Position[11:8]	
0x3A	RO	TOUCH10_XL	10st Touch X Position[7:0]			
0x3B	RO	TOUCH10_YH	10st Touch ID[3:0]		10st Touch Y Position[11:8]	

0x3C	RO	TOUCH10_YL	10st Touch Y Position[7:0]		
0x3D	RO	TOUCH10_WEIGHT	10st Touch Weight[7:0]		
0x3E	RO	TOUCH10_MISC	10st Touch Area[3:0]		

3.2 DEVICE_MODE

This is the device mode register, which is configured to determine the current mode of the chip.

Address	Bit Address	Register Name	Description
0x00	6:4	[2:0]Device Mode	000b WORKING Mode 100b TEST Mode

3.3 GEST_ID

This register describes the gesture of a valid touch.

Address	Bit Address	Register Name	Description
0x01	7:0	Gesture ID[7:0]	Gesture ID 0x10 Move Up 0x14 Move Right 0x18 Move Down 0x1C Move Left 0x48 Zoom In 0x49 Zoom Out 0x00 No Gesture

3.4 TD_STATUS

This register is the Touch Data status register.

Address	Bit Address	Register Name	Description
0x02	7:0	Number of touch points [7:0]	The detected point number, max. 10

3.5 Pn_XH (n:1-5)

This register describes MSB of the X coordinate of the nth touch point and the corresponding event flag.

Address	Bit Address	Register Name	Description
0x03 0x09 0x0F 0x15 0x1B 0x21 0x27 0x2D 0x33 0x39	7:6	Event Flag	00b: Press Down 01b: Lift Up 10b: Contact 11b: No event
	5:4	Reserved	
	3:0	Touch X Position [11:8]	MSB of Touch X Position in pixels

3.6 Pn_XL (n:1-5)

This register describes LSB of the X coordinate of the nth touch point.

Address	Bit Address	Register Name	Description
0x04 0x0A 0x10 0x16 0x1C 0x22 0x28 0x2E 0x34 0x3A	7:0	Touch X Position [7:0]	LSB of the Touch X Position in pixels

3.7 Pn_YH (n:1-5)

This register describes MSB of the Y coordinate of the nth touch point and corresponding touch ID.

Address	Bit Address	Register Name	Description
0x05 0x0B 0x11 0x17 0x1D 0x23 0x29 0x2F 0x35 0x3B	7:4	Touch ID[3:0]	Touch ID of Touch Point, this value is 0x0F when the ID is invalid
	3:0	Touch Y Position [11:8]	MSB of Touch Y Position in pixels

3.8 Pn_YL (n:1-2)

This register describes LSB of the Y coordinate of the nth touch point.

Address	Bit Address	Register Name	Description
0x06 0x0C 0x12 0x18 0x1E 0x24 0x2A 0x30 0x36 0x3C	7:0	Touch Y Position [7:0]	LSB of the Touch Y Position in pixels

3.9 Pn_WEIGHT (n:1-5)

This register describes weight of the nth touch point.

Address	Bit Address	Register Name	Description
0x07 0x0D 0x13	7:0	Touch Weight[7:0]	Touch pressure value

0x19 0x1F 0x25 0x2B 0x31 0x37 0x3D			

3.10 Pn_MISC (n:1-5)

This register describes the miscellaneous information of the nth touch point.

Address	Bit Address	Register Name	Description
0x08 0x0E 0x14 0x1A 0x20 0x26 0x2C 0x32 0x38 0x3E	7:4	Touch Area[3:0]	Touch area value
	3:0	Reserved	

4 Communication between host and CTPM

4.1 Communication Contents

The data Host received from the CTPM through I2C interface are different depend on the configuration in Device Mode Register of the CTPM. Please refer to Section 2---CTPM Register Mapping.

4.2 I2C Example Code

The code is only for reference, if you want to learn more, please contact our FAE staff.

```
//////////////////////////////////////////////////////////////////////
// I2C write bytes to device.
// Arguments: ucSlaveAdr - slave address
//                 ucSubAdr - sub address
//                 pBuf - pointer of buffer
//                 ucBufLen - length of buffer
//////////////////////////////////////////////////////////////////////
void i2cBurstWriteBytes(BYTE ucSlaveAdr, BYTE ucSubAdr, BYTE *pBuf, BYTE ucBufLen)
{
   BYTE ucDummy; // loop dummy
   ucDummy = I2C_ACCESS_DUMMY_TIME;
   while(ucDummy--)
   {
      if (i2c_AccessStart(ucSlaveAdr, I2C_WRITE) == FALSE)
         continue;
      if (i2c_SendByte(ucSubAdr) == I2C_NON_ACKNOWLEDGE) // check non-acknowledge
```

```
            continue;
         while(ucBufLen--) // loop of writting data
         {
            i2c_SendByte(*pBuf); // send byte
            pBuf++; // next byte pointer
         } // while
         break;
      } // while
          i2c_Stop();
   }

   ////////////////////////////////////////////////////////////////////
   // I2C read bytes from device.
   //
   // Arguments: ucSlaveAdr - slave address
   //                    ucSubAdr - sub address
   //                    pBuf - pointer of buffer
   //                    ucBufLen - length of buffer
   ////////////////////////////////////////////////////////////////////
   void i2cBurstReadBytes(BYTE ucSlaveAdr, BYTE ucSubAdr, BYTE *pBuf, BYTE ucBufLen)
   {
      BYTE ucDummy; // loop dummy

      ucDummy = I2C_ACCESS_DUMMY_TIME;
      while(ucDummy--)
      {
         if (i2c_AccessStart(ucSlaveAdr, I2C_WRITE) == FALSE)
            continue;
         if (i2c_SendByte(ucSubAdr) == I2C_NON_ACKNOWLEDGE) // check non-acknowledge
            continue;
         if (i2c_AccessStart(ucSlaveAdr, I2C_READ) == FALSE)
            continue;
         while(ucBufLen--) // loop to burst read
         {
            *pBuf = i2c_ReceiveByte(ucBufLen); // receive byte
            pBuf++; // next byte pointer
         } // while
         break;
      } // while
```