FocalTech

Application Note for CTPM

Application Note for CTPM	
Project name	Touch panel
Document ref	[Document ref]
Version	1.1
Release date	15 Apr,2011
Owner	Lihua
Classification	Confidential
Distribution List	[Distribution list]
Approval	

R3-B4-A, South Area, Shenzhen Hi-Tech Industrial Park,
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 + Signature
18 Jan, 2010	0.1	Initial draft	Xiaoxu Du
17 Mar,2010	0.2	Add raw data protocol	Xiaoxu Du
22 Mar,2010	0.3	Add system information protocol	Xiaoxu Du
26 Mar,2010	0.4	Add calibration related parameters	Xiaoxu Du
08 May,2010	0.5	Add information to operating mode	Xinming Wang
07 Jul, 2010	0.6	Change Protocol and add information	Yunfeng Yuan
18 Aug, 2010	0.7	Modified to release version	Xiaoxu Du
22 Dec, 2010	0.8	Modify explanation for register 0xA4	Xiaoxu Du
13 Feb, 2011	0.9	Add ten point support Modify mode description	Xiaoxu Du
17 Feb, 2011	1.0	Modify FT5306DE4 typical application schematic	Xiaoxu Du
15 Apr,2011	1.1	Modify FT5206GE1/FT5306DE4/FT5406EE8 application schematic	Lihua

Table of Contents

Terminology

CTP – Capacitive touch panel

CTPM – Capacitive touch panel module

1 I^{2}C Interface

1.1 CTPM interface to Host

Figure 1-1 shows how CTPM communicates with the Host，there are three kind of communication between CTPM and Host，we will introduce each communication in this section.

Transfer the data via I^{2}C

Send interrupt when there is a valid touch

Host send Wakeup signal to CTPM

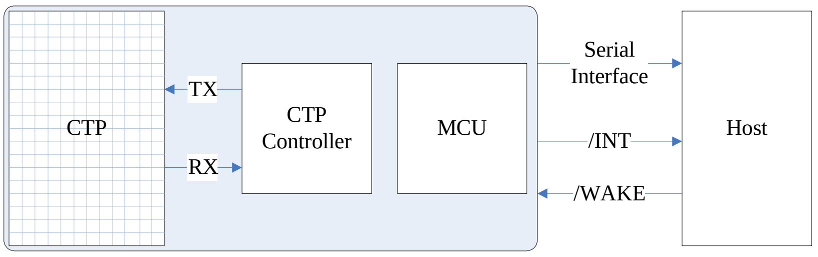


Figure 1-1 CTPM and Host connection

The Power Supply voltage of CTPM is 2.8V~3.3V, interface supply voltage is 2.8V~3.3V. There are Control Interface and Data Interface. As

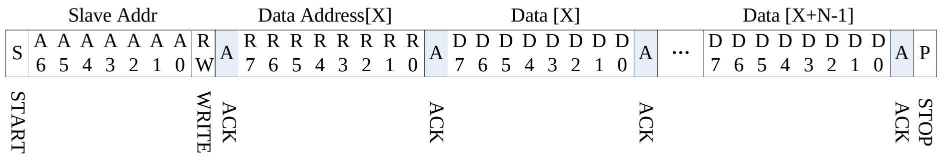
Figure 1-1 demonstrates, Serial interface is the data interface, /INT and /WAKE are the control interface. For the detail, please refer to Table 1-1.

Table 1-1 Description for TP module and Host interface

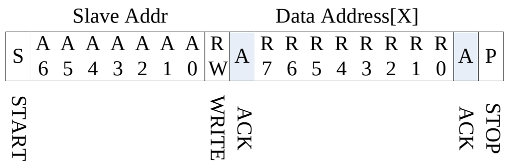
Port Name	Voltage	Polar	Description
Serial interface	2.8~3.3V		Serial interface is for data transfer between Host and CTPM. CTPM support both I2C and SPI interface
/INT	2.8~3.3V	LOW	The interrupt from the CTPM to the Host
/WAKE*	2.8~3.3V	LOW	Wakeup signal from host to the CTPM

1.2 I^{2}C Read/Write Interface description

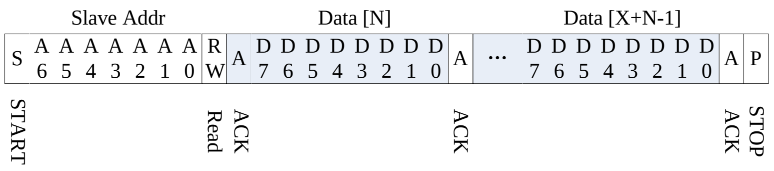
Write N bytes to I2C slave



Set Data Address



Read X bytes from I^{2}C Slave



1.3 Interrupt signal from CTPM to Host

As for standard CTPM, host need to use both interrupt control signal and serial data interface to get the touch data. There are two kind of method to use interrupt: interrupt trigger and interrupt query.

Here is the timing to get touch data.

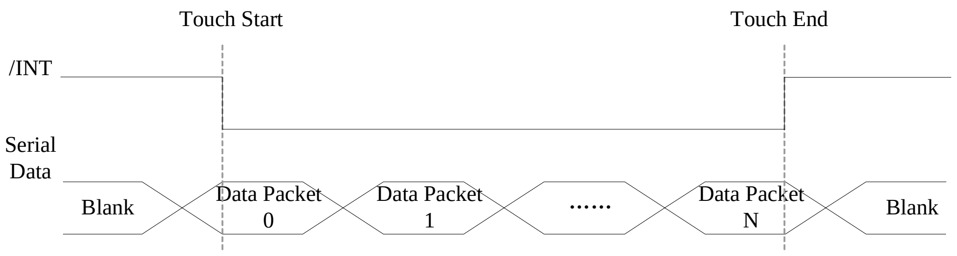


Figure 1-2 Interrupt query mode

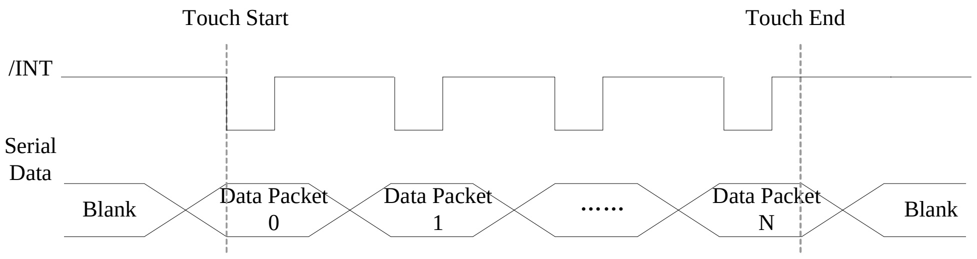


Figure 1-3 Interrupt trigger mode

Host use general I2C protocol to read the touch data or the information from CTPM . CTPM will send host a interrupt signal when there is a valid touch. Then host can use the serial data interface to get the touch data. If there is no valid touch detected, the /INT will not be pulled up, the host do not need to read the touch data.

NOTE: “valid touch” may have different definition in various systems. For example, in some systems, the valid touch is defined as there is one more valid touch point. But in some other systems, the valid touch is defined as one more valid touch with valid gestures. In usual, /INT will be pulled up when there is a valid touch point, and to be low when a touch finishes.

As for interrupt trigger mode, /INT signal will be low if there is a touch detected. But for per update of valid touch data, CTPM will produce a valid pulse for /INT signal, host can read the touch data periodically according to the frequency of this pulse. In this mode, the pulse frequency is the touch data update frequency.

.

1.4 Wakeup signal from Host to CTPM

Host can use the Wakeup Signal to wakeup the I^{2}C slave device.

This pin should be connected to GND when flash programming while in normal running mode it should not be connected to GND.

2 CTP Register Mapping

This chapter describes the standard FTS Capacitive Touch Panel products communication registers in address order for each device mode. The most detailed descriptions of the Standard Products communication registers are in the Register Definitions section of each chapter. The device modes are listed in the table below, along with each mode’s register prefix.

Device Mode	Val	Description
Work	000b	Read touch point and gesture
Factory	100b	Read raw data

2.1 Work Mode

In this mode the CTP is fully functional as a touch screen controller. Read and write access address is just logical address which is not enforced by hardware or firmware. Here is the operating mode register map.

Work Mode Register Map

Address	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Host Access
00h	DEVIDE_MODE		Device Mode[2:0]							RW
01h	GEST_ID	Gesture ID[7:0]								R
02h	TD_STATUS					Number of touch points[3:0]				R
03h	TOUCH1_XH	1stEvent Flag				1st Touch X Position[11:8]				R
04h	TOUCH1_XL	1st Touch X Position[7:0]								R
05h	TOUCH1_YH	1st Touch ID[3:0]				1st Touch Y Position[11:8]				R
06h	TOUCH1_YL	1st Touch Y Position[7:0]								R
07h										
08h										
09h	TOUCH2_XH	2ndEvent Flag				2ndTouch X Position[11:8]				R
0Ah	TOUCH2_XL	2nd touch X Position[7:0]								R
0Bh	TOUCH2_YH	2nd Touch ID[3:0]				2nd Touch Y Position[11:8]				R
0Ch	TOUCH2_YL	2nd Touch Y Position[7:0]								R
0Dh										R
0Eh										R
0Fh	TOUCH3_XH	3rdEvent Flag				3rd Touch X Position[11:8]				R
10h	TOUCH3_XL	3rd Touch X Position[7:0]								R
11h	TOUCH3_YH	3rd Touch ID[3:0]				3rd Touch Y Position[11:8]				R
12h	TOUCH3_YL	3rd Touch Y Position[7:0]								R
13h										R
14h										R

15h	TOUCH4_XH	4thEvent Flag		4th Touch X Position[11:8]	R
16h	TOUCH4_XL	4th Touch X Position[7:0]			R
17h	TOUCH4_YH	4th Touch ID[3:0]		4th Touch Y Position[11:8]	R
18h	TOUCH4_YL	4th Touch Y Position[7:0]			R
19h					R
1Ah					R
1Bh	TOUCH5_XH	5thEvent Flag		5th Touch X Position[11:8]	R
1Ch	TOUCH5_XL	5thTouch X Position[7:0]			R
1Dh	TOUCH5_YH	5th Touch ID[3:0]		5th Touch Y Position[11:8]	R
1Eh	TOUCH5_YL	5th Touch Y Position[7:0]			R
1Fh					R
20h					R
21h	TOUCH6_XH	6thEvent Flag		6th Touch X Position[11:8]	
22h	TOUCH6_XL	6thTouch X Position[7:0]			
23h	TOUCH6_YH	6th Touch ID[3:0]		6th Touch Y Position[11:8]	
24h	TOUCH6_YL	6th Touch Y Position[7:0]			
25h					
26h					
27h	TOUCH7_XH	7thEvent Flag		7th Touch X Position[11:8]	
28h	TOUCH7_XL	7thTouch X Position[7:0]			
29h	TOUCH7_YH	7th Touch ID[3:0]		7th Touch Y Position[11:8]	
2Ah	TOUCH7_YL	7th Touch Y Position[7:0]			
2Bh					
2Ch					
2Dh	TOUCH8_XH	8thEvent Flag		8th Touch X Position[11:8]	
2Eh	TOUCH8_XL	8thTouch X Position[7:0]			
2Fh	TOUCH8_YH	8th Touch ID[3:0]		8th Touch Y Position[11:8]	
30h	TOUCH8_YL	8th Touch Y Position[7:0]			
31h					
32h					

33h	TOUCH9_XH	9th Event Flag		9th Touch X Position[11:8]		
34h	TOUCH9_XL	9th Touch X Position[7:0]				
35h	TOUCH9_YH	9th Touch ID[3:0]		9th Touch Y Position[11:8]		
36h	TOUCH9_YL	9th Touch Y Position[7:0]				
37h						
38h						
39h	TOUCH10_XH	10th Event Flag		10th Touch X Position[11:8]		
3Ah	TOUCH10_XL	10th Touch X Position[7:0]				
3Bh	TOUCH10_YH	10th Touch ID[3:0]		10th Touch Y Position[11:8]		
3Ch	TOUCH10_YL	10th Touch Y Position[7:0]				
3Dh						
3Eh						
3Fh	Reserved					
…	…					
7Fh	Reserved					
80h	ID_G_THGROUP	valid touching detect threshold.				R/W
81h	ID_G_THPEAK	valid touching peak detect threshold.				R/W
82h	ID_G_THCAL	the threshold when calculating the focus of touching.				R/W
83h	ID_G_THWATER	the threshold when there is surface water.				R/W
84h	ID_G_THTEMP	the threshold of temperature compensation.				R/W
85h						R/W
86h	ID_G_CTRL				Power control mode[1:0]	R/W
87h	ID_G_TIME_ENTER _MONITOR	The timer of entering monitor status				R/W
88h	ID_G_PERIODACTIVE			Period Active[3:0]		R/W
89h	ID_G_PERIOD MONITOR	The timer of entering idle while in monitor status				R/W
8Ah						R/W
8Bh						R/W
8Ch						R/W
8Dh						R/W
8Eh						R/W
8Fh						R/W
90h						R/W

91h			R/W
92h			R/W
93h			R/W
94h			R/W
95h			R/W
96h			R/W
97h			R/W
98h			R/W
99h			R/W
9Ah			R/W
9Bh			R/W
9Ch			R/W
9Dh			R/W
9Eh			R/W
9Fh			R/W
A0h	ID_G_AUTO_CLB _MODE	auto calibration mode	R/W
A1h	ID_G_LIB_ VERSION_H	Firmware Library Version H byte	R
A2h	ID_G_LIB _VERSION_L	Firmware Library Version L byte	R
A3h	ID_G_CIPHER	Chip vendor ID	R
A4h	ID_G_MODE	the interrupt status to host	R
A5h	ID_G_PMODE	Power Consume Mode	
A6h	ID_G_FIRMID	Firmware ID	R
A7h	ID_G_STATE	Running State	
A8h	ID_G_FT5201ID	CTPM Vendor ID	R
A9h	ID_G_ERR	Error Code	R
AAh	ID_G_CLB	Configure TP module during calibration in Test Mode	R/W
ABh			R/W
ACh			R/W
ADh			R/W
AEh	ID_G_B_AREA_TH	The threshold of big area	R/W
AFh			R/W
…	…		
FDh	Reserved		
FEh	LOG_MSG_CNT	The log MSG count	R
FFh	LOG_CUR_CHA	Current character of log message, will point to the next character when one character is read.	R

2.1.1 DEVICE_MODE

This register is the device mode register, configure it to determine the current mode of the chip**.**

Address	Bit Address	Register Name	Description
00h	6:4	Device Mode [2:0]	000b Work Mode 100b Factory Mode – read raw data

2.1.2 GEST_ID

This register describes the gesture of a valid touch.

Address	Bit Address	Register Name	Description
01h	7:0	Gesture ID [7:0]	Gesture ID 0x10 Move UP 0x14 Move Left 0x18 Move Down 0x1C Move Right 0x48 Zoom In 0x49 Zoom Out 0x00 No Gesture

2.1.3 TD_STATUS

This register is the Touch Data status register.

Address	Bit Address	Register Name	Description
02h	3:0	Number of touch points[3:0]	How many points detected. 1-5 is valid.
	7:4		

2.1.4 TOUCHn_XH (n:1-10)

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

Address	Bit Address	Register Name	Description
03h ~ 39h	7:6	Event Flag	00b: Put Down 01b: Put Up 10b: Contact 11b: Reserved
	5:4		Reserved
	3:0	Touch X Position [11:8]	MSB of Touch X Position in pixels

2.1.5 TOUCHn_XL (n:1-10)

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

Address	Bit Address	Register Name	Description
04h ~ 3Ah	7:0	Touch X Position [7:0]	LSB of the Touch X Position in pixels

2.1.6 TOUCHn_YH (n:1-10)

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

Address	Bit Address	Register Name	Description
05h ~ 3Bh	7:4	Touch ID[3:0]	Touch ID of Touch Point
	3:0	Touch X Position [11:8]	MSB of Touch Y Position in pixels

2.1.7 TOUCHn_YL (n:1-10)

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

Address	Bit Address	Register Name	Description
06h ~ 3Ch	7:0	Touch X Position [7:0]	LSB of The Touch Y Position in pixels

2.1.8 ID_G_THGROUP

This register describes valid touching detect threshold.

Address	Bit Address	Register Name	Description
80h	7:0	ID_G_THGROUP	The actual value will be 4 times of the register's value. Default:280/4

2.1.9 ID_G_THPEAK

This register describes valid touching peak detect threshold.

Address	Bit Address	Register Name	Description
81h	7:0	ID_G_ THPEAK	Default:60

2.1.10 ID_G_ THCAL

This register describes threshold when calculating the focus of touching.

Address	Bit Address	Register Name	Description
82h	7:0	ID_G_ THCAL	Default:16

2.1.11 ID_G_ THWATER

This register describes threshold when there is surface water.

Address	Bit Address	Register Name	Description
83h	7:0	ID_G_ THWATER	Default:60

2.1.12 ID_G_ THTEMP

This register describes threshold of temperature compensation.

Address	Bit Address	Register Name	Description
84h	7:0	ID_G_ THTEMP	Default:10

2.1.13 ID_G_ THDIFF

This register describes threshold whether the coordinate is different from the original.

Address	Bit Address	Register Name	Description
85h	7:0	ID_G_ THDIFF	The actual value must be 32timers of the register's value. Default :20

2.1.14 ID_G_ CTRL

This register describes the run mode of microcontroller controlled by host

Address	Bit Address	Register Name	Description
86h	0	ID_G_ CTRL	0: not auto jump 1:auto jump

2.1.15 ID_G_ TIMEENTERMONITOR

This register describes the time delay value when entering monitor status.

Address	Bit Address	Register Name	Description
87h	7:0	ID_G_TIME ENTERMONITOR	Default :2

2.1.16 ID_G_ PERIODACTIVE

This register describes the period of active status, it should not less than 12

Address	Bit Address	Register Name	Description
88h	4:0	ID_G_ PERIOD ACTIVE	Range form 3 to 14,default 12
	7:4		

2.1.17 ID_G_ PERIODMONITOR

This register describes period of monitor status, it should not less than 30.

Address	Bit Address	Register Name	Description
89h	7:0	ID_G_ PERIOD MONITOR	Default:40

2.1.18 ID_G_ AUTO_CLB_MODE

This register describes auto calibration mode.

Address	Bit Address	Register Name	Description

A0h	7:0	ID_G_ AUTO_ CLB_MODE	8’h 00: enable auto calibration 8’h ff: disable auto calibration

2.1.19 ID_G_ LIB_VERSION_H

This register describes library version high byte.

Address	Bit Address	Register Name	Description
A1h	7:0	ID_G_LIB_VERSION_H	R: xx

2.1.20 ID_G_ LIB_VERSION_L

This register describes library version low byte.

Address	Bit Address	Register Name	Description
A2h	7:0	ID_G_ LIB_VERSION_L	R: xx

2.1.21 ID_G_ CIPHER

This register describes vendor’s chip id.

Address	Bit Address	Register Name	Description
A3h	7:0	ID_G_ CIPHER	R: xx

2.1.22 ID_G_ MODE

This register describes the interrupt status to host.

Address	Bit Address	Register Name	Description
A4h	7:0	ID_G_ MODE	0: Polling mode 1: Trigger mode

2.1.23 ID_G_ PMODE

This register describes the power consumption mode of the TPM when in running status.

Address	Bit Address	Register Name	Description
A5h	7:0	ID_G_ PMODE	0: active 1: monitor 3: hibernate(deep sleep)

2.1.24 ID_G_ FIRMWARE_ID

This register describes the firmware id of the application.

Address	Bit Address	Register Name	Description
A6h	7:0	ID_G_ FIRMWARE_ID	R: xx

2.1.25 ID_G_ STATE

This register is used to configure the run mode of TPM.

Address	Bit Address	Register Name	Description
A7h	7:0	ID_G_ STATE	0: configure 1: work 2: calibration

			3: factory 4: auto calibration

2.1.26 ID_G_ FT5201ID

This register describes vendor's chip id

Address	Bit Address	Register Name	Description
A8h	7:0	ID_G_ FT5201ID	R: xx

2.1.27 ID_G_ ERR

This register describes the error code when the TPM is running.

Address	Bit Address	Register Name	Description
A9h	7:0	ID_G_ ERR	ERR Code 8'h00:OK 8'h03:chip register writing inconsistent with reading 8'h05:chip start fail 8'h1A:no match among the basic input(such as TX_ORDER) while calibration

2.1.28 ID_G_ CLB

This register is used to configure the TPM when Calibration

Address	Bit Address	Register Name	Description
AAh	7:0	ID_G_ CLB	Mapping the Array of G_Bank1, total length is NUM_TX+NUM_RX+1. the array address increases 1 after every write.

2.2 Factory Mode

In this mode, CTP will provide some panel related information. Host can get the following information in this mode

Raw data of touch panel

Panel configure related information

Factory Mode Register Map

Address	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Host Access
00h	DEVIDE_MODE	ST_SCAN	Device Mode[2:0]							RW
01h	ROW_ADDR	The address of the row to be read								RW
02h	CLB	Calibration command &status								RW
03h	ROW_NUM	Panel row number								RW
04h	COL_NUM	Panel column number								RW
05h	DRIVER_VOL	Driver voltage of chip								RW
06h	START_RX	Setting the RX start number								RW
07h	GAIN	Control the difference value for touching								RW
08h	ORIGIN_XH	High byte of origin X coordinate								RW

09h	ORIGIN_XL	Low byte of origin X coordinate		RW
0Ah	ORIGIN_YH	High byte of origin Y coordinate		RW
0Bh	ORIGIN_YL	Low byte of origin Y coordinate		RW
0Ch	RES_WH	High byte of width of resolution		RW
0Dh	RES_WL	Low byte of width of resolution		RW
0Eh	RES_HH	High byte of height of resolution		RW
0Fh	RES_HL	Low byte of height of resolution		RW
10h	RAWDATA0_H	High byte of raw data 0		R
11h	RAWDATA0_L	Low byte of raw data 0		R
12h	RAWDATA1_H	High byte of raw data 1		R
13h	RAWDATA1_L	Low byte of raw data 1		R
…	…	…		
4Ah	RAWDATA29_H	High byte of raw data 29		R
4Bh	RAWDATA29_L	Low byte of raw data 29		R
4Ch	TH_POINT_NUM	Touch point number support		RW
4Dh	Reserved			
4Eh	Reserved			
4Fh	Reserved			
50h	TX_ORDER_0	TX Order, start from zero		RW
51h	TX_ORDER_1			RW
…	…	…		RW
77h	TX_ORDER_39			RW
78h	ROW0_CAC	Charge Amplifier feedback Capacitance of ROW0		RW
79h	ROW1_CAC	Charge Amplifier feedback Capacitance of ROW1		RW
…	…	…		
9Fh	ROW39_CAC	Charge Amplifier feedback Capacitance of ROW39		RW
A0h	COL0_CAC	Charge Amplifier feedback Capacitance of COL0		RW
…	…	…		
BDh	COL29_CAC	Charge Amplifier feedback Capacitance of COL29		RW
BEh	Reserved			
BFh	ROW0_1_OFFSET	Offset of ROW1	Offset of ROW0	RW
…	…	…	…	
D2h	ROW38_39_OFFSET	Offset of ROW39	Offset of ROW38	RW
D3h	COL0_1_OFFSET	Offset of COL1	Offset of COL0	RW
…	…	…	…	
E1h	COL28_29_OFFSET	Offset of COL29	Offset of COL28	RW
…	…			
FEh	LOG_MSG_CNT	The log MSG count		R

FFh	LOG_CUR_CHA	Current character of log message, will point to the next character when one character is read.	R

2.2.1 DEVICE_MODE

This register is the device mode register, configure it to determine the current mode of the chip**.**

Address	Bit Address	Register Name	Description
00h	7	ST_SCAN	By default, it is 0; If set to 1, the frame scan begins, and ST_SCAN gets back to 0 once this frame scan finishes. Normally, a full frame scan tales no more than 10ms.
	6:4	Device Mode[2:0]	000b Work Mode 100b Factory Mode – read raw data

2.2.2 ROW_ADDR

This register is the Touch Data status register.

Address	Bit Address	Register Name	Description
01h	7:0	Row address	The address of the row to be read Please delay for more than 100us, then read the raw data

2.2.3 CLB

This register is for the calibration command, calibration status check and calibration result writing.

Address	**Bit Address**	**Register Name**	**Description**
0x02	7:0	CLB	Default value after reset is 0xFF. 0x04: set to start calibration; 0x05: set to start to store calibration result to flash memory.

2.2.4 ROWDATAN_H

This register is the Touch Data status register.

Address	Bit Address	Register Name	Description
(10+2n)h	7:0	High byte of raw data N	High byte of raw data N If N exceeds the column number will return 0xff

2.2.5 ROWDATAN_L

This register is the Touch Data status register.

Address	Bit Address	Register Name	Description
(10+2n+1)h	7:0	Low byte of raw data N	Low byte of raw data N

			If N exceeds the column number will return 0xff

3 CTPM Application Introduction

3.1 Standard Application information of FT5X06

Figure3-1,Figure3-2,Figure3-3 demonstrate the typical FT5x06 application schematic. It consists of FT's Capacitive Touch Panel(CTP), FT5X06 chip, and some peripheral components. According to the size of CTPM, you can choose the numbers of TX and RX needed.

3.1.1 Standard application circuit of FT5206GE1

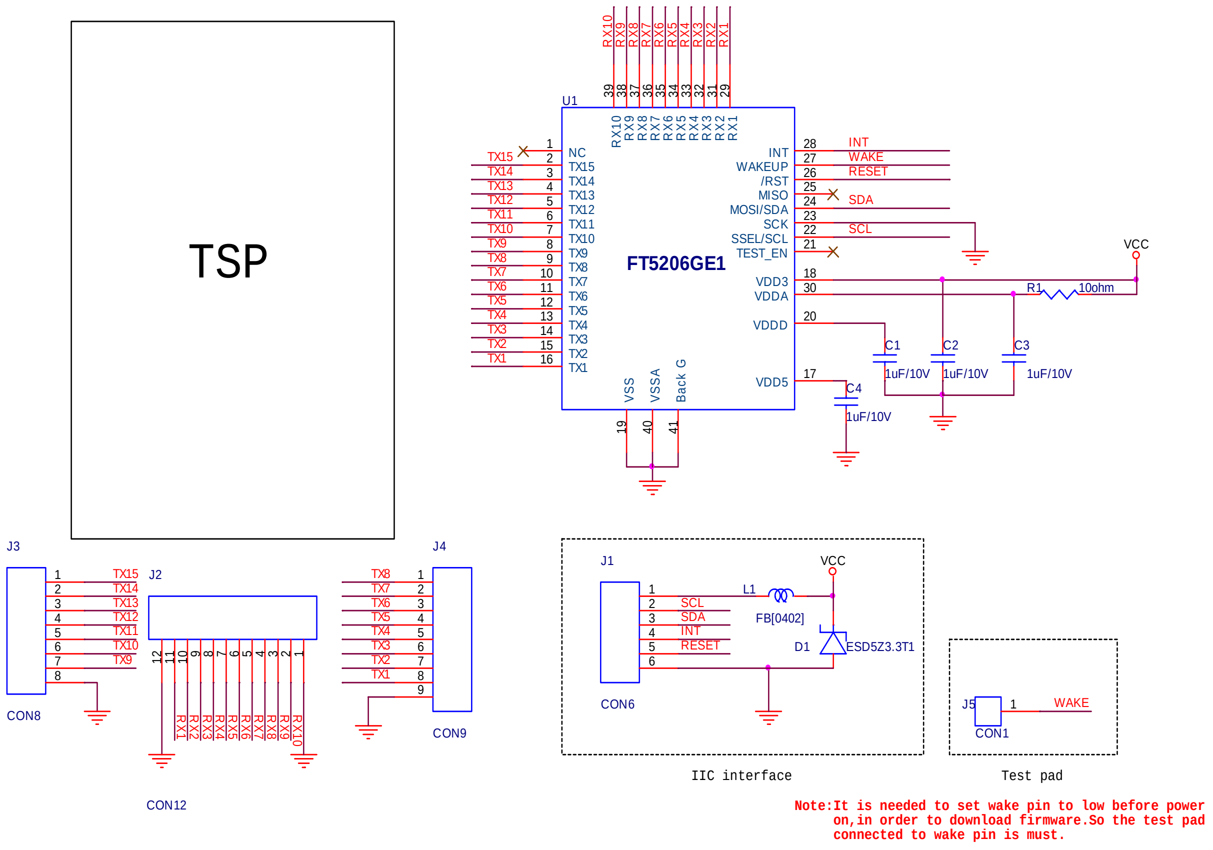


Figure 3-1 FT5206GE1 typical application schematic

3.1.2 Standard application circuit of FT5306DE4

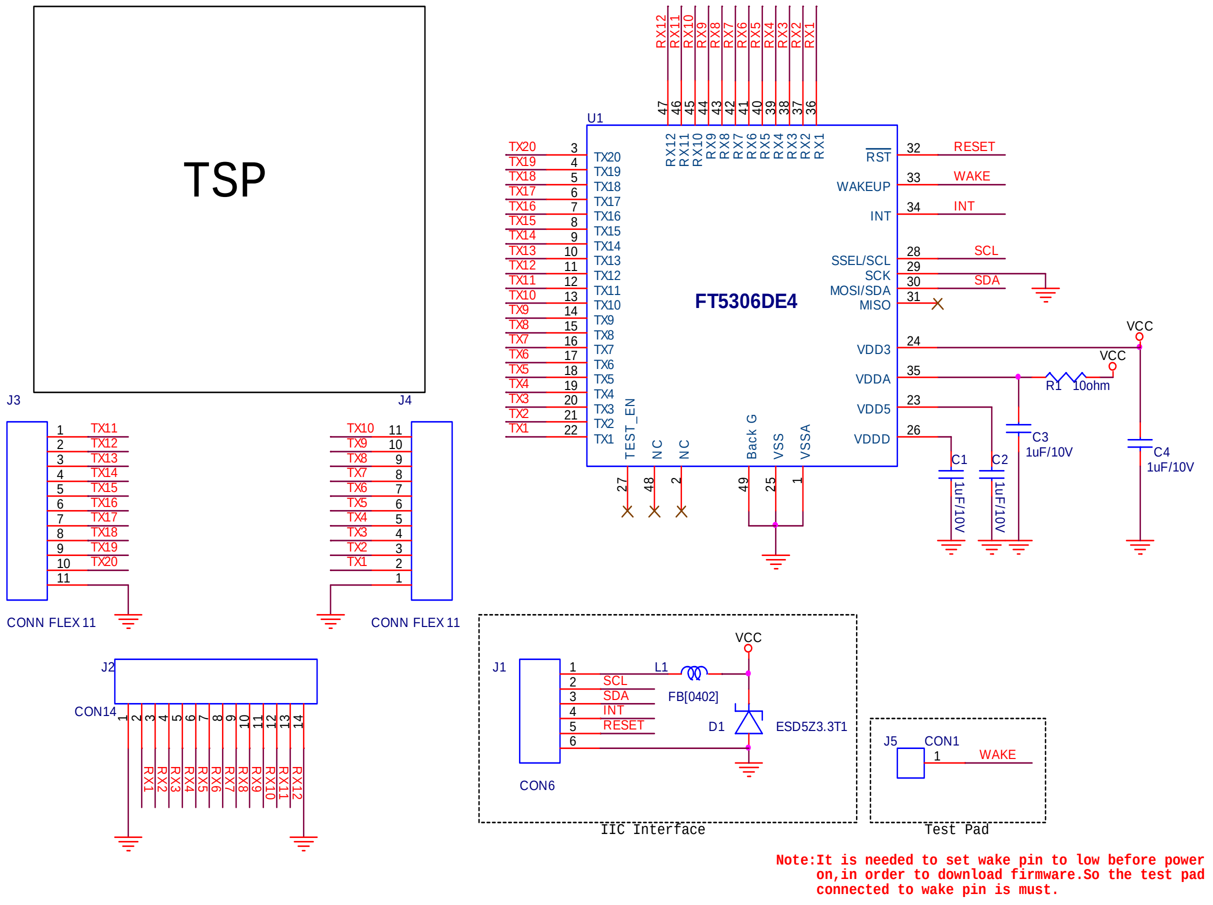


Figure 3-2 FT5306DE4 typical application schematic

3.1.3 Standard application circuit of FT5406EE8

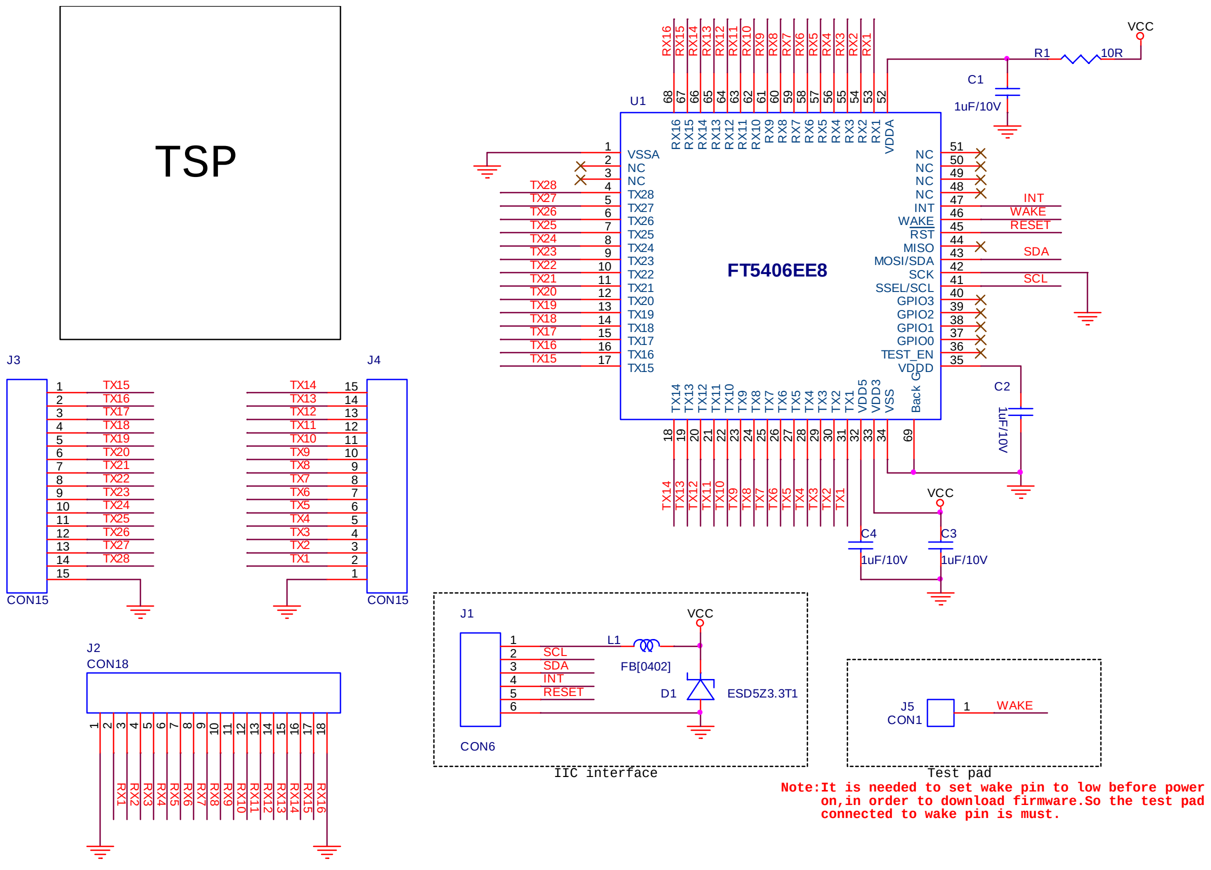

TSP
FT5406EE8
U1
J1
J2
J3
J4
J5
CON15
CON18
CON6
CON1
VCC
R1
10R
C1
C2
C3
C4
1uF/10V
L1
FB[0402]
D1
ESD5Z3.3T1
SCL
SDA
INT
RESET
WAKE
IIC interface
Test pad
Note:It is needed to set wake pin to low before power on,in order to download firmware.So the test pad connected to wake pin is must.

Figure 3-3 FT5406EE8 typical application schematic

4 Communication between host and CTPM

4.1 Communication Contents

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

4.2 I2C Example Code

```
/////////////////////////////////////////////////////////////////
// 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
    i2c_Stop();
}

/////////////////////////////////////////////////////////////////
```

```
// I2C read current bytes from device.
//
// Arguments: ucSlaveAdr - slave address
//                  pBuf - pointer of buffer
//                  ucBufLen - length of buffer
/////////////////////////////////////////////////////////////////
void i2cBurstCurrentBytes(BYTE ucSlaveAdr, BYTE *pBuf, BYTE ucBufLen)
{
    BYTE ucDummy; // loop dummy

    ucDummy = I2C_ACCESS_DUMMY_TIME;
    while(ucDummy--)
    {
        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
    i2c_Stop();
}
```