晶采光電科技股份有限公司
AMPIRE CO., LTD.

SPECIFICATIONS FOR
Capacitive Touch Panel

CUSTOMER	
CUSTOMER PART NO.	
AMPIRE PART NO.	AP-CTP050B0NEI3000000
APPROVED BY	
DATE	

☐Approved For Specifications
☑**Approved For Specifications & Sample**

AMPIRE CO., LTD.
4F., No.116, Sec. 1, Sintai 5th Rd., Sijhih City, Taipei County 221, Taiwan (R.O.C.)

APPROVED BY	CHECKED BY	ORGANIZED BY

RECORD OF REVISION

Revision Date	Page	Contents	Editor
2015/1/5	--	New Release	Mark

1 SCOPE

The specification is applied to Projected Capacitive Multi touch panel mad by AMPIRE CO.,LTD.

2 FUNCTION

The touch panel module is an input device for the industrial and consumer application. It is usually placed on the front of the flat display panel (such as TFT). The gap between the touch panel module and flat display panel is **0.5mm** at least. When user touch the touch panel module by finger, the built-in touch controller will do the signal sensing, calculation and indicated the coordinate point. The Host controller can get the coordinate data via the touch panel **I2C** interface.

3 FEATURE

3.1 Construction: Touch Sensor, Touch Controller and FPC.
3.2 Touch Controller : EXC7200
3.3 Resolution : 2048 x 2048
3.4 Interface: **I2C**
3.5 Power Supply Voltage : 3.3V
3.6 Touch Finger Number : 2

4 PHYSICAL SPECIFICATIONS

Item	Specifications	unit
Panel Outline	115.2(H) X 74(V) X 1.25	mm
Touch Panel Active Area	109.4(H) X 66.2(V)	mm
Touch Sensor Thickness	1.1	mm

5 OPTICAL CHARACTERISTIC

Item	Specifications	unit
Transparency	≧ 85%	--

6 ABSOLUTE MAX RATING

Do not apply voltage , temperature over the specify value all the time. Otherwise, the touch panel maybe module lost it's function.

6.1 Electrical Absolute Max Rating

Item	Symbol	Values		UNIT	Note
		Min.	Max.		
Power Supply voltage	VIN	-0.3	5.0	V	DGND =0V

6.2 Environmental Rating

Item	Symbol	Values		UNIT	Note
		Min.	Max.		
Operating temperature	Top	-20	+70	℃	1,2
Storage temperature	Tstg	-30	+80	℃	1,2

Note 1: Humidity : 20% ~90% at max 50.

Note 2: All terms under 1 atmisohere.

7 ELECTRICAL CHARACTERISTICS

Specify the normal operating condition

(DGND=0V)

Item	Symbol	Min.	Typ.	Max.	Unit	Note
Power Supply Voltage	VIN	3.0	3.3	3.6	V	
Low Level Input Voltage	VIL	0	--	0.8	V	1
High Level Input Voltage	VIH	0.8*VIN	--	VIN	V	1
Power Consumption	I_{VIN}		33		mA	

Note 1: SDA , SCL ,RESET

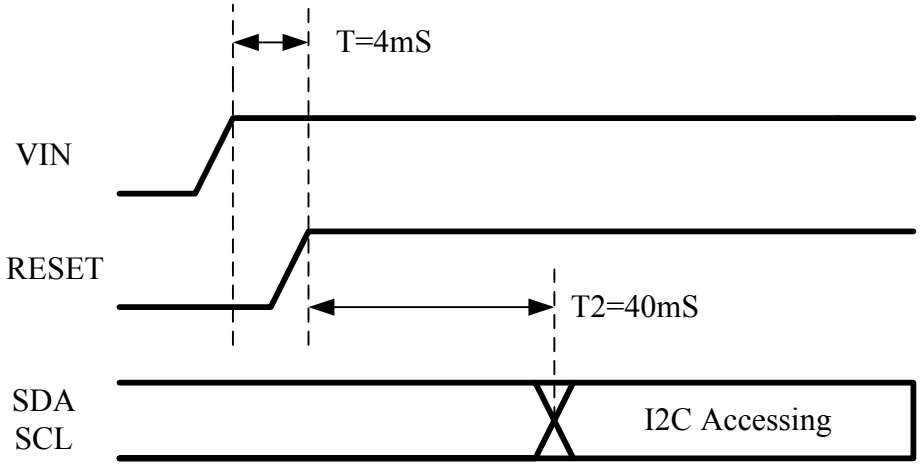
8 INTERFACE PIN ASSIGNMENT

Pin	Name	Description
1	DGND	Power GND
2	SDA	I^2C Data
3	SCL	I^2C Clock
4	VDD	Power supply 3.3V
5	INT	Active “Low”
6	XRES	Active “Low”

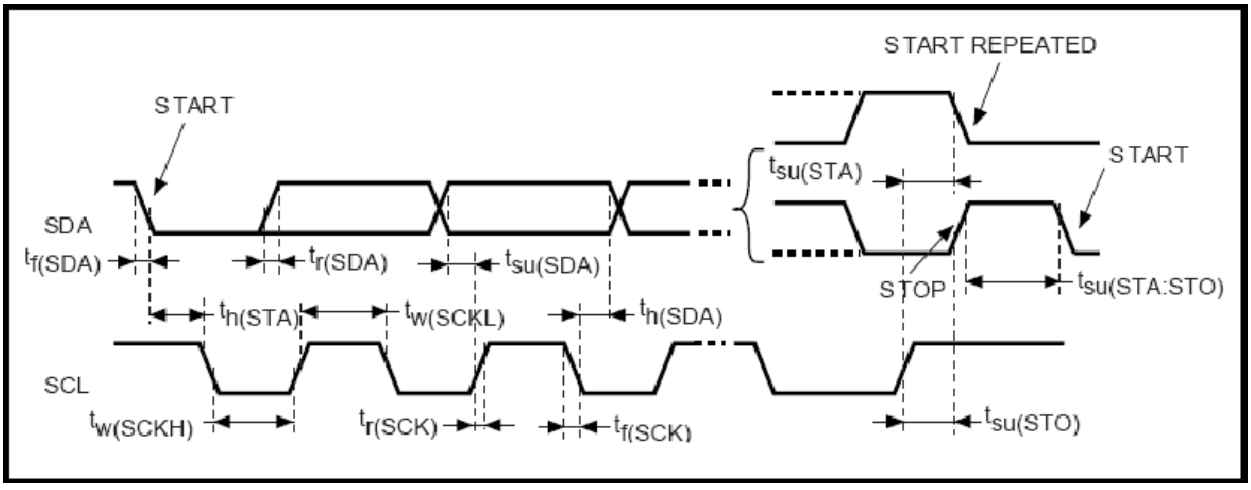
9 TIMING

9.1 POWER ON TIMING

RESET Keep Low 4mS at least. After RESET goes to high, do not accessing the I2C bus 40mS at least. Because the controller need do the internal initiation.



9.2 I2C AC TIMING



Symbol	Parameter	SCL = 100KHz		SCL = 400KHz		Unit
		Min	Max	Min	Max	
$t_{w(SCLL)}$	SCL clock low time	4.7		1.3		µs
$t_{w(SCLH)}$	SCL clock high time	4.0		0.6		
$t_{su(SDA)}$	SDA setup time	250		100		ns
$t_{h(SDA)}$	SDA data hold time	0		0	900	
$t_{r(SDA)}$ $t_{r(SCL)}$	SDA and SCL rise time		1000		300	
$t_{f(SDA)}$ $t_{f(SCL)}$	SDA and SCL fall time		300		300	
$t_{h(STA)}$	Start condition hold time	4.0		0.6		µs
$t_{su(STA)}$	Repeated Start condition setup time	4.7		0.6		
$t_{su(STO)}$	Stop condition setup time	4.0		0.6		µs
$t_{w(STO:STA)}$	Stop to Start condition time (bus free)	4.7		1.3		µs

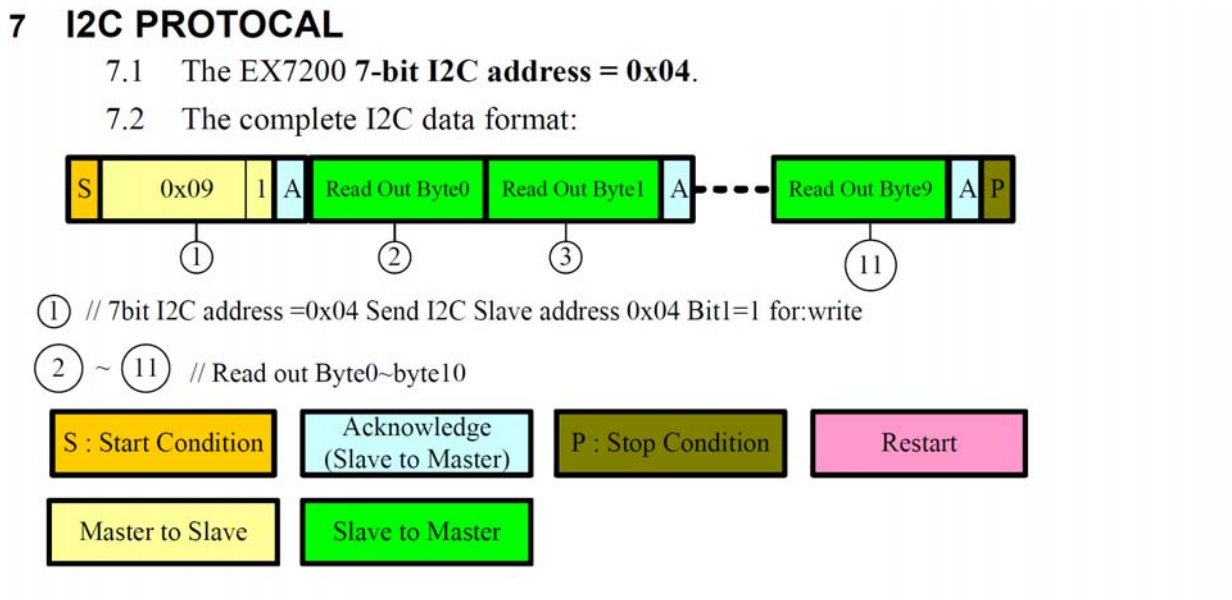
10 TOUCH CONTROLLER SOFTWARE PROTOCAL

10.1 Device Slave Address :

The EXC7200 7-bit slave Address=0x04

10.2 The complete Read back data format:

(From HOST to Device)



10.3 Report Byte Information

Read Out Byte Number	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
BYTE0	0	0	0	0	0	1	0	0
	Report ID = 0x04							
BYTE1								
	BIT[7] : When this bit =1 is Valid touch. BIT[6:2] : Contact ID. BIT[1] : In range bit , this bit is always 1. BIT[0] : This bit 1 for touch down , 0 for touch lift Example: 0x83: 1st Touch (Contact ID=0) Down. 0x82: 1st Touch (Contact ID=0)Lift. 0x87: 2nd Touch (Contact ID=1)Down. 0x86: 2nd Touch (Contact ID=1)Lift.							
BYTE2	Touch X [3:0]				Don't care			
	X Coordination Bit [3:0] in BYTE2 Bit [7: 4]							
BYTE3	Touch X [10:4]							
	X Coordination Bit [10:4] in BYTE3							
BYTE4	Touch Y [3:0]				Don't care			
	Y Coordination Bit [3:0] in BYTE2 Bit [7: 4]							

BYTE5	Touch Y [10:4]								
	Y Coordination Bit [10:4] in BYTE3								
BYTE6									
	Reserved								
BYTE7									
	Reserved								
BYTE8									
	Reserved								
BYTE9									
	Reserved								

10.4 Sample Code

```
/*********************************************************************
* Function Name    : u8 EXC7200_I2C_CoordRead(u8 Slave_Addr,u8 read_Nbyte , u16    *ByteReturn)
* Description      : Use GPIO Read N byte data from Slave device (Addr) to Host
* Input              : u8 Slave_Addr , Ex:0x04
*                    : u8 read_Nbyte
*                    : Point for touch information
*                       ByteReturn [0] = Point X1
*                       ByteReturn [1] = Point Y1
*                       ByteReturn [2] = T.B.D
*                       ByteReturn [3] = Point Y2
*                       ByteReturn [4] = Point Y2
*                       ByteReturn [5] = T.B.D
* Return
*
* #define EXC7200_I2C_CoordRead_1stTouch_Down    0x01
* #define EXC7200_I2C_CoordRead_1stTouch_Lift    0x81
* #define EXC7200_I2C_CoordRead_2ndTouch_Down    0x02
* #define EXC7200_I2C_CoordRead_2ndtTouch_Lift    0x82
*********************************************************************/

#define EXC7200_I2C_CoordRead_1stTouch_Down    0x01
#define EXC7200_I2C_CoordRead_1stTouch_Lift    0x81
#define EXC7200_I2C_CoordRead_2ndTouch_Down    0x02
#define EXC7200_I2C_CoordRead_2ndtTouch_Lift    0x82
#define EXC7200_I2C_CoordRead_error    0x00

u8 EXC7200_I2C_CoordRead(u8 Slave_Addr,u8 read_Nbyte , u16    *ByteReturn )
{
   u8 *pBuffer ,i ;
   u8 Byte[10] ;


   IO_I2C_start();    // Start Condiction
   IO_I2C_reg_cmd_para((Slave_Addr<<1)+1); // Send I2C Slave address+1 Bit0=1 for:read
                pBuffer=IO_I2C_read_Nbyte(read_Nbyte);      // read 10 byte

                for(i=0;i<10;i++)
                    {
                            Byte[i]=*pBuffer;
                            pBuffer++;
                    }

                if( (Byte[1]==0x83) | (Byte[1]==0x82))    //
                {
                    ByteReturn[0]=((u16)((Byte[3]&0x00ff)<<4))+((u16)((Byte[2]&0x00f0)>>4)); //Point X1
```

```
                    ByteReturn[1]=((u16)((Byte[5]&0x00ff)<<4))+((u16)((Byte[4]&0x00f0)>>4)); //Point Y1
                    ByteReturn[2]= 0xFFFF;
                    ByteReturn[3]= 0xFFFF;
                    ByteReturn[4]= 0xFFFF;
                    ByteReturn[5]= 0xFFFF;
                    Previous_X1=ByteReturn[0];
                    Previous_Y1=ByteReturn[1];

                    if ( (Byte[1]==0x83))
                    {
                            return EXC7200_I2C_CoordRead_1stTouch_Down;
                    }
                    if ( (Byte[1]==0x82))
                    {
                            return EXC7200_I2C_CoordRead_1stTouch_Lift;
                    }

   }
                if( (Byte[1]==0x87) | (Byte[1]==0x86))    //
                {
                    ByteReturn[3]=((u16)((Byte[3]&0x00ff)<<4))+((u16)((Byte[2]&0x00f0)>>4)); //Point X1
                    ByteReturn[4]=((u16)((Byte[5]&0x00ff)<<4))+((u16)((Byte[4]&0x00f0)>>4)); //Point Y1
                    ByteReturn[5]= 0xFFFF;
                    ByteReturn[0]= Previous_X1;
                    ByteReturn[1]= Previous_Y1;
                    ByteReturn[2]= Previous_Z1;

                    if ( (Byte[1]==0x87))
                    {
                            return EXC7200_I2C_CoordRead_2ndTouch_Down;
                    }
                    if ( (Byte[1]==0x86))
                    {
                            return EXC7200_I2C_CoordRead_2ndtTouch_Lift;
                    }

   }


   return EXC7200_I2C_CoordRead_error;

}


// Example Interrupt function
void EXC7200_I2C_EXT_INT (void)
{

   u16 DataBuffer[10];
   u32 TPX1,TPY1,TPX2,TPY2;
   u16 Temp_X=0xFFFF,Temp_Y=0xFFFF;
   u16 temp;
   u8 Touch_size=4;
   u8    RStatus;

      while((ReadINT1())==0)
      {
                              RStatus = EXC7200_I2C_CoordRead(0x04,10,DataBuffer);


                              TPX1=(u16) DataBuffer[0]; //first X position
                              TPY1=(u16) DataBuffer[1]; //first Y position
```

```
                TPX2=(u16) DataBuffer[3]; //second point X position
                TPY2=(u16) DataBuffer[4]; //second point Y position


                // Remapping Touch X,Y to LCD X,Y
                TPX1*=Current_LCM_ID.LCD_X_Max;
                TPX1/=2048;
                TPY1*=Current_LCM_ID.LCD_Y_Max;
                TPY1/=2048;
                TPX2*=Current_LCM_ID.LCD_X_Max;
                TPX2/=2048;
                TPY2*=Current_LCM_ID.LCD_Y_Max;
                TPY2/=2048;


                switch (RStatus)
                {
                        case (EXC7200_I2C_CoordRead_1stTouch_Down):
                GUI_CircleFill(TPX1, TPY1, 2, rand()%0xFFFF);
                                // Do 1st touch down Function
                                break;
                        case (EXC7200_I2C_CoordRead_1stTouch_Lift):
                GUI_RectangleFill(TPX1-4, TPY1-4,TPX1+4, TPY1+4 ,rand()%0xFFFF);
                                // Do 1st touch Lift Function

                                break;
                        case (EXC7200_I2C_CoordRead_2ndTouch_Down):
                GUI_CircleFill(TPX1, TPY1, 2, RGB(128,128,128));
                GUI_CircleFill(TPX2, TPY2, 2, RGB(128,128,0));
                                // Do 2nd touch Down Function
                                break;
                        case (EXC7200_I2C_CoordRead_2ndtTouch_Lift):
                GUI_RectangleFill(TPX1-4, TPY1-4,TPX1+4, TPY1+4 ,RGB(128,128,128));
                GUI_RectangleFill(TPX2-4, TPY2-4,TPX2+4, TPY2+4 ,RGB(128,128,0));
                                // Do 2nd touch Lift Function
                                break;
                        default:
                                break;
                }

        }

}
```

11 RELIABILITY TEST CONDITIONS

Test Item	Test Conditions	Note
High Temperature Storage	80±3°C , Dry t=240 hrs	1,2
Low Temperature Storage	-30±3°C ,Dry t=240 hrs	1,2
High Temperature Operating	70±3°C , Dry t=240 hrs	1,2
Low Temperature Operating	-20±3°C ,Dry t=240 hrs	1,2
Thermal Shock Test	-30°C ~ 70°C 60 m in. (1 cycle) Total 10 cycle(Dry)	1,2

Note 1 : Condensation of water is not permitted on the module.

Note 2 : The module should be inspected after 1 hour storage in normal conditions

(15-35°C , 45-65%RH).

Definitions of life end point :

- Current drain should be smaller than the specific value.
- Function of the module should be maintained.
- Appearance and display quality should not have degraded noticeably.

8. OUTLINE DIMENSION