触摸屏技术文档

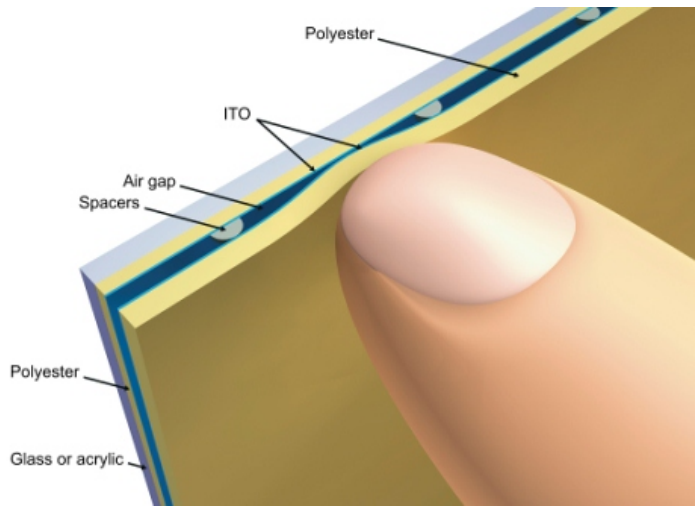
第一章 基础知识

1、认识触摸屏

今天的电气和电子设备采用了以下 5 种类型的触摸屏技术：电阻式、表面电容式、投射电容式、表面声波式和红外线式。其中前三种适合用于移动设备和消费电子产品，后两种技术做出的触摸屏不是太昂贵就是体积太大，因此不适合上述应用。

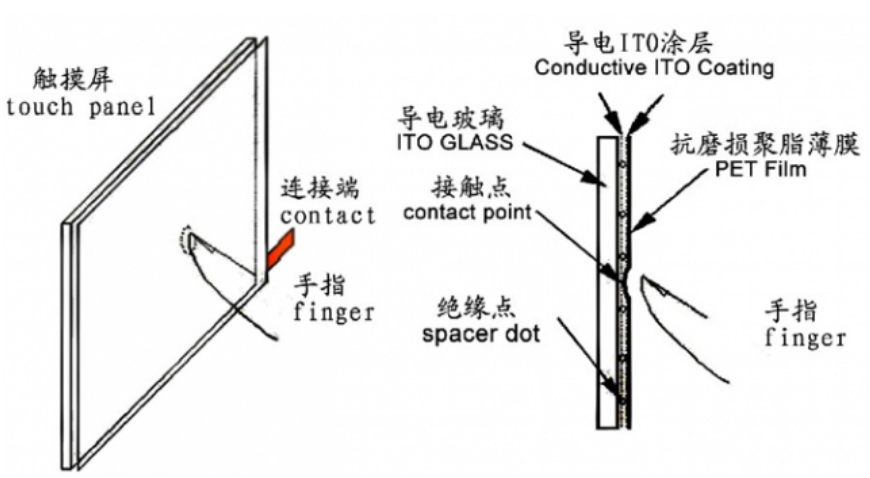
电阻式触摸屏（见图 1）

需要一定的压力才能激活，电阻式触摸屏采用了三明治架构实现，上下两层是印刷在塑料（PET）薄膜上的导电性铟锡氧化物(ITO)，中间隔以空气。



该空气隙由很多微小的间隔器来保持，当两个导电层被手指（或铁笔）压到一起时才算是完成了一次‘触摸’，而触摸的位置通过测量 X 轴和 Y 轴上的电压比就可检测出来。根据采用多少根线将数据传输到微控制器进行处理，电阻式触摸屏可分为四线、五线、六线和八线版本。

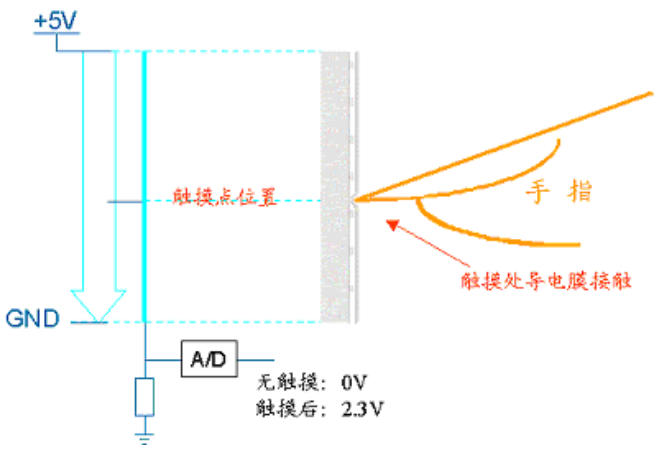
触摸屏(触摸屏的工作原理)由触摸检测部件和触摸屏控制器组成，触摸检测部件安装在显示器屏幕前面，用于检测用户触摸位置，接受后送触摸屏控制器；而触摸屏控制器的主要作用是从触摸点检测装置上接收触摸信息，并将它转换成触点坐标，再送给 CPU，它同时能接收 CPU 发来的命令并加以执行。

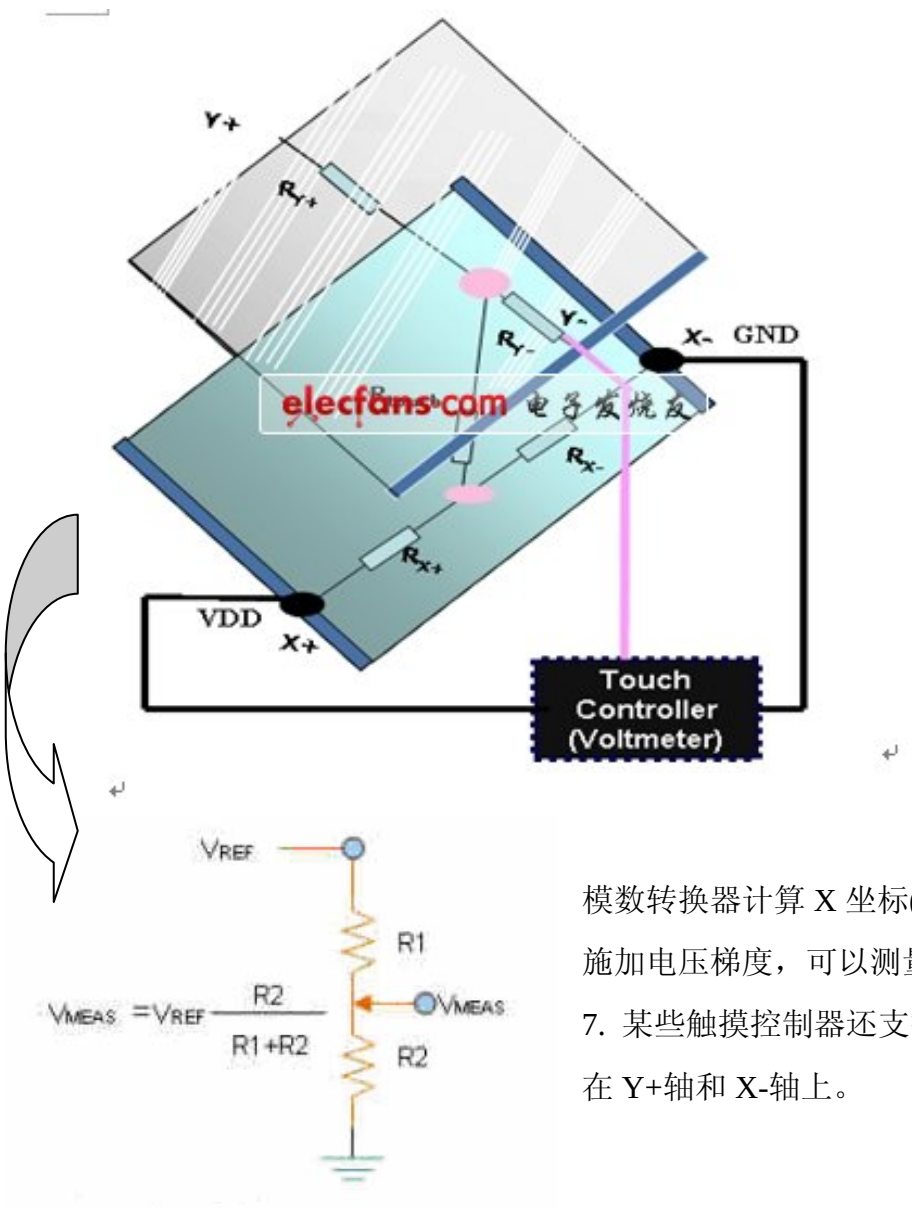


它的内表面也涂有一层涂层、在他们之间有许多细小的（小于 1/1000 英寸）的透明隔离点把两层导电层隔开绝缘。

2、工作原理

当手指触摸屏幕时，平常相互绝缘的两层导电层就在触摸点位置有了一个接触，因其中一面导电层接通 Y 轴方向的 5V 均匀电压场，使得侦测层的电压由零变为非零，控制器侦测到这个接通后，进行 A/D 转换，并将得到的电压值与 5V 相比即可得触摸点的 Y 轴坐标，同理得出 X 轴的坐标，这就是所有电阻技术触摸屏共同的最基本原理。





1. 电阻式触摸屏是表面覆盖触摸响应薄膜的透明玻璃板。

2. 电阻式触摸屏面板有两个电阻层(氧化铟锡)组成，中间是一层很薄的分隔层。

3. 电阻触摸屏的两个薄膜层组成一个电阻网络，充当触摸位置检测功能的分压电路。

4. 触摸屏会在电阻网络组成的分压器上引起电压变化，这个电压用于确定触摸屏幕的触点位置。

5. **触摸屏控制器(TSC)**把捕捉的模拟电压信号转换成数字触摸坐标信号。内置模数转换通道，充当测量模拟电压的电压计。

6. 在触摸屏幕后，起到电压计作用的触摸控制器首先在 X+点施加电压梯度 VDD，在 X-点施加接地电压 GND。然后，检测 Y 轴电阻上的模拟电压，并把模拟电压转换成数值，用模数转换器计算 X 坐标(图 2)。在这种情况下，Y-轴变成感应线。同样地，在 Y+和 Y-点分施加电压梯度，可以测量 Y 坐标。

7. 某些触摸控制器还支持触摸压力测量，即 Z 轴测量。测量 Z 轴坐标时，电压梯度施加在 Y+轴和 X-轴上。

3、四线触摸屏

四线触摸屏包含两个阻性层。其中一层在屏幕的左右边缘各有一条垂直总线，另一层在屏幕的底部和顶部各有一条水平总线，见下图。为了在 X 轴方向进行测量，将左侧总线偏置为 0V，右侧总线偏置为 VREF。将顶部或者底部总线连接到 ADC，当顶层和底层相接触时既可作为一次测量。

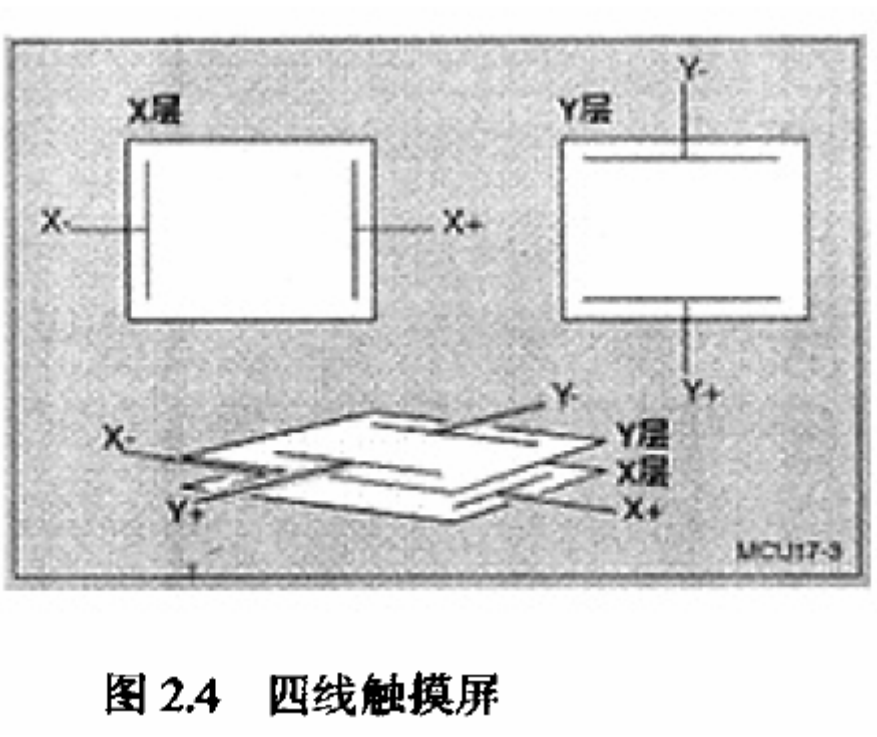
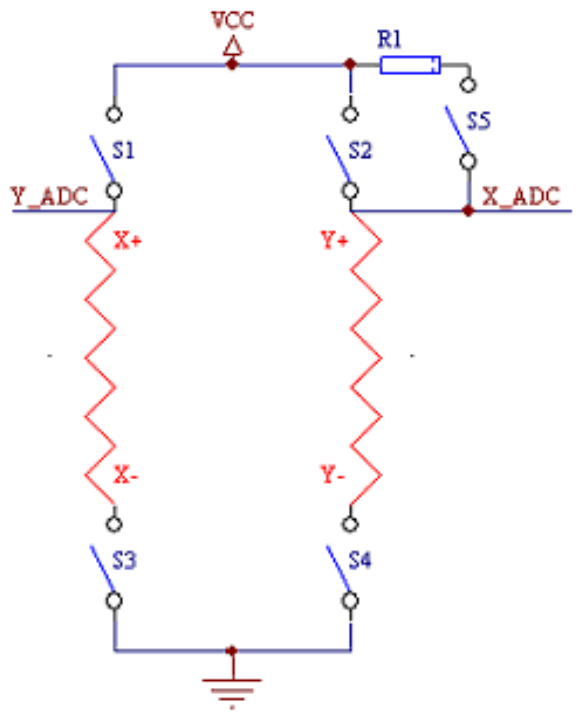


图 2.4 四线触摸屏

下图是四线电阻触摸屏测量时的等效电路（图 6-7）：



测量时，分为以下 3 个步骤：

（1） 起初，在触摸屏没有被按下的时候，触摸屏的 X 轴和 Y 轴不会接触在一起，此时这个电路处在“Pen Down Detect”状态。S1、S2、S4 断开，S3、S5 闭合。X+~X-的整个轴上的电压均为 0V（GND），Y-端悬空，Y+端由于有上拉电阻 R1 的存在而呈现高电平。当“Pen Down”后，X 轴和 Y 轴受挤压而接触导通后，Y 轴上的电压由于连通到 X 轴接地而变为低电平，**此低电平可做为中断触发信号来通知 CPU 发生“Pen Down”事件。**

（2） 当检测到“PenDown”事件后，CPU 立刻进入 X 轴坐标测量状态：S1、S3 闭合，S2、S4、S5 断开（Y+、Y-两断悬空）。由于 X 轴和 Y 轴在接触点按下而连通，因此 Y＋端的 X_ADC 可以认为是 X 轴的分压采样点（通过测量 X_ADC 的电压可以得到 X+到

接触点，以及 X-到接触点的比例），从而计算出 X 轴的坐标
（3） 采样完 X 轴的坐标后，S1、S3、S5 断开，S2、S4 闭合，同样原理，我们可以进一步得到 Y 轴的坐标。

4、S3C2410 模数转换器（ADC）及触摸屏控制器

S3C2410 内置 1 个 8 信道的 10bit 模数转换器（ADC），该 ADC 能以 500KSPS 的采样资料将外部的模拟信号转换为 10bit 分辩率的数字量。同时 ADC 部分能与 CPU 的触摸屏控制器协同工作，完成对触摸屏绝对地址的测量。
特性：
－分辩率：10bit
－相信误差：＋/- 2LSB
－最大转换速率：500KSPS
－模拟量输入范围：0～3.3V
－分步 X/Y 坐标测量模式
－自动 X/Y 坐标测量模式
－中断等待模式
下图是 ADC 及触摸屏控制器部分的逻辑示意图（图 6-8）

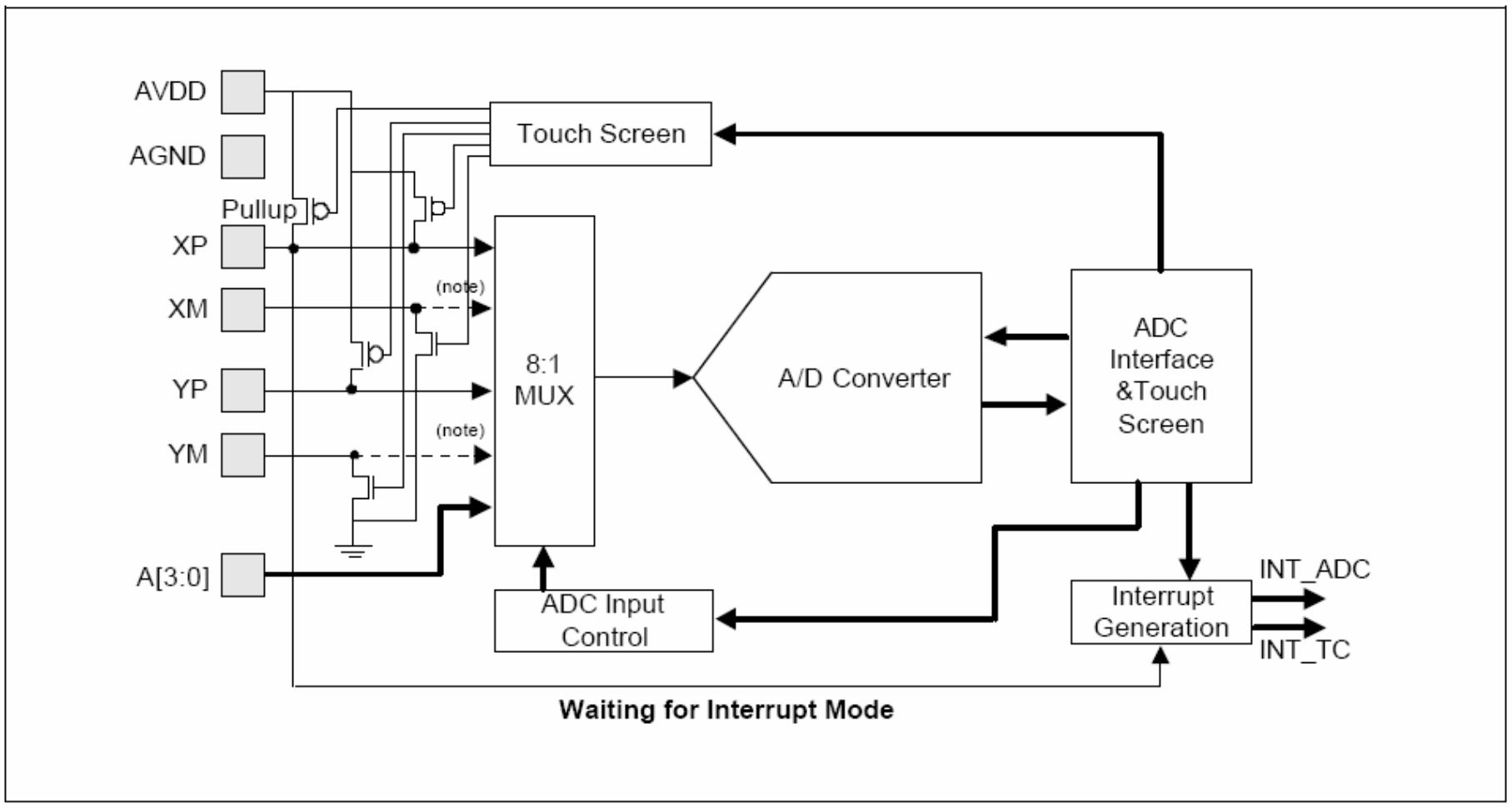


Figure 16-1. ADC and Touch Screen Interface Functional Block Diagram

随后的图是在 S3C2410 的 ADC 以及触摸屏控制器的基础上外接触摸屏的示意图，以及外部电路的实际原理图。需要补充说明的是，图中 Q1、Q2 为 P 沟道 MOS 管，开门电压为 1.8V；Q3、Q4 为 N 沟道 MOS 管，开门电压为 2.7V。运用学过的电子电路的知识，我们知道当 MOS 管导通后（栅极电压达到开门电压之后），MOS 管的源-漏极之间可以认为是直通的（导通电阻为毫欧级），即可以把 MOS 管认为是图 4-7 中真正的“开关”。 AVDD 是外部模拟参考源，一般接 3.3V 电源，XP、XM 和 YP、YM 分别是触摸屏的 4 条引线，各自对应 X 轴和 Y 轴电阻。

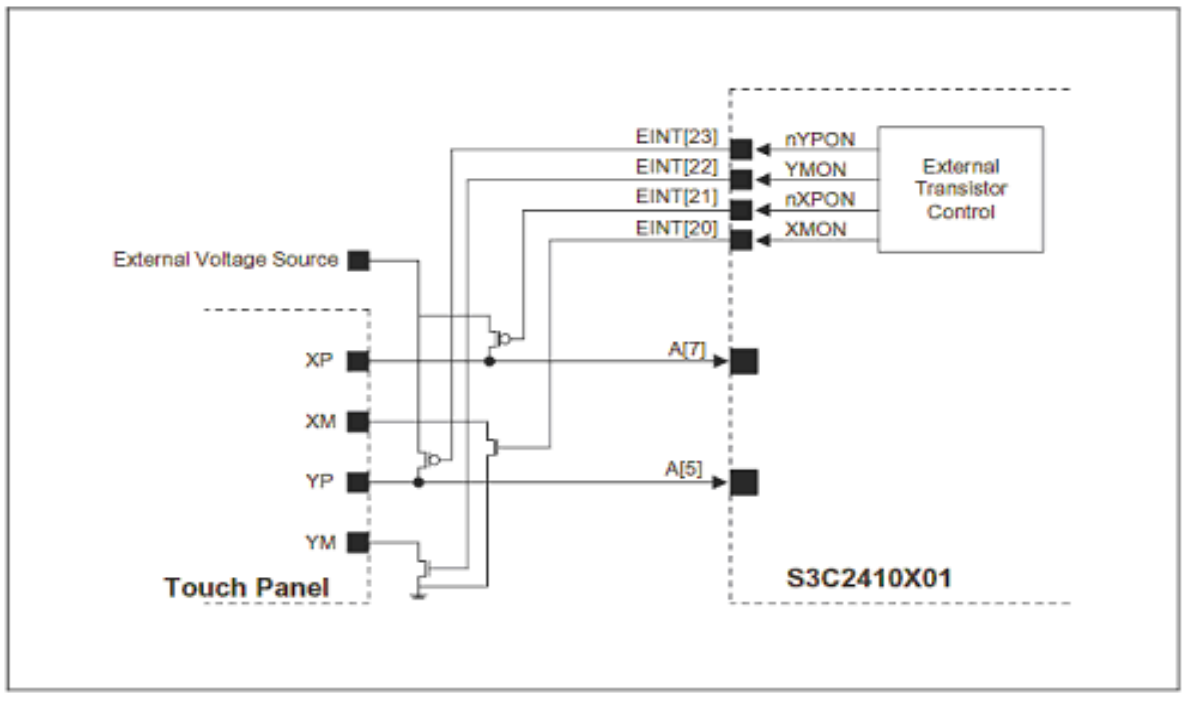


图 6-8 中 XP,XM,YP,YM 对应触摸屏接口的 TSXP,TSXM,TSYP,TSYM.

ADC 及触摸屏控制器的工作模式：

1、 ADC 普通转换模式（Normal Converson Mode）

普通转换模式（AUTO_PST=0,XY_PST=0）是用来进行一般的 ADC 转换之用的，例如通过 ADC 测量电池电压等等。

2、 独立 X/Y 轴坐标转换模式（Separate X/Y Position Conversion Mode）

独立 X/Y 轴坐标转换模式其实包含了 X 轴模式和 Y 轴模式 2 种模式。

首先进行 X 轴的坐标转换（AUTO_PST=0，XY_PST=1），X 轴的转换资料会写到 ADCDAT0 寄存器的 XPDAT 中，**等待转换完成后，触摸屏控制器会产生相应的中断。**

然后进行 Y 轴的坐标转换（AUTO_PST=0，XY_PST=2），Y 轴的转换资料会写到 ADCDAT1 寄存器的 YPDAT 中，**等待转换完成后，触摸屏控制器会产生相应的中断。**

3、 自动 X/Y 轴坐标转换模式（Auto X/Y Position Conversion Mode）

自动 X/Y 轴坐标转换模式（AUTO_PST=1，XY_PST=0）将会自动地进行 X 轴和 Y 轴的转换操作，随后产生相应的中断。

4、 中断等待模式（Wait for InterruptMode）

在系统等待“Pen Down”即触摸屏按下的时候，其实是处于中断等待模式。一旦被按下，实时产生“INT_TC”中断信号。每次发生此中断都，X 轴和 Y 轴坐标转换资料都可以从相应的资料寄存器中读出。

5、 闲置模式（Standby Mode）

在该模式下转换资料寄存器中的值都被保留为上次转换时的资料。

ADC 及触摸屏控制器的寄存器详解

ADCCON ： ADC 控制寄存器（见图 6-9）

ENABLE_START ：

置 1：启动 ADC 转换

置 0：无操作

RESR_START ：

置 1：允许读操作启动 ADC 转换

置 0：禁止读操作启动 ADC 转换

STDBM：

置 1：将 ADC 置为闲置状态（模式）

置 0：将 ADC 置为正常操作状态

SEL_MUX：选择需要进行转换的 ADC 信道

PRSCVL：ADC 转换时钟预分频参数

PRSCEN：ADC 转换时钟使能

ECFLG：ADC 转换完成标志位（只读）

为 1：ADC 转换结束

为 0：ADC 转换进行中

ADCTSC ： 触摸屏控制寄存器（见图 6-10）

XY_PST ： 对 X/Y 轴手动测量模式进行选择

AUTO_PST：X/Y 轴的自动转换模式使能位

PULL_UP ： XP 端的上拉电阻使能位

XP_SEN ： 设置 nXPON 输出状态

XM_SEN ： 设置 XMON 输出状态

YP_SEN ： 设置 nYPON 输出状态

YM_SEN ： 设置 YMON 输出状态

ADCDLY ： ADC 转换周期等待定时器（见图 6-11）

ADCDAT0 ： ADC 资料寄存器 0（见图 6-12）

XPDATA ： X 轴转换资料寄存器

XY_PST ： 选择 X/Y 轴自动转换模式

AUTO_PST：X/Y 轴自动转换使能位

UPDOWN ： 选择中断等待模式的类型

为 0：按下产生中断

为 1：释放产生中断

ADCDAT1 ： ADC 资料寄存器 1（见图 6-13）

定义类同于 ADCDAT0。

A/D Conversion Time

大家可以看到 经过预分频 MUX 得到 GCLK，如果 GCLK frequency is 50MHz，the prescaler value is 49, total 10-bit conversion time is as follo ws

A/D converter freq. = 50MHz/(49+1) = 1MHz

Conversion time = 1/(1MHz / 5cycles) = 1/200KHz = 5 us

第二章 代码分析

Touchpanel.c

说明：

NOTE: Before ADC conversion, Touch screen uses X-tal clock (3.68MHz). During ADC conversion GCLK (Max. 50MHz) is used.

在 ADC 转换之前会用到外部时钟，X-tal clock (3.68MHz). 之后才有到 glck (Max. 50MHz)

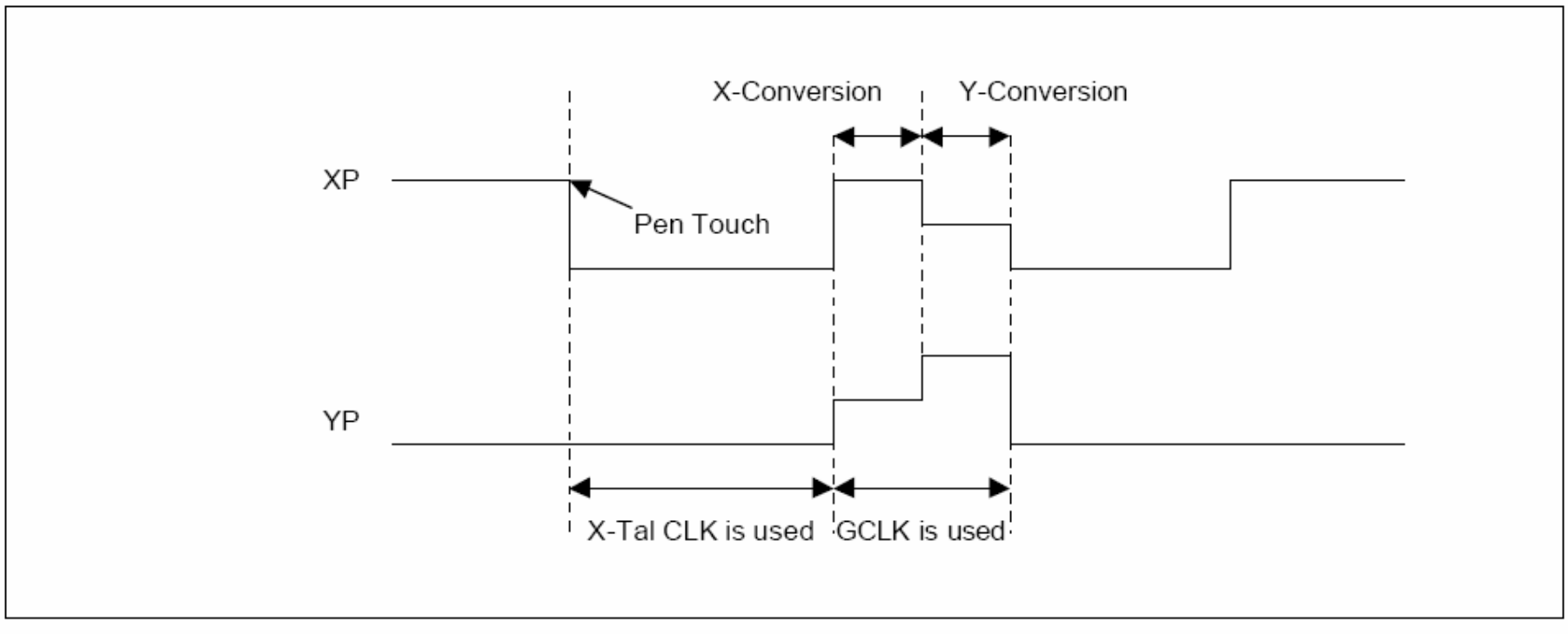


Figure 16-2. ADC and Touch Screen Operation signal

rADCDLY=50000; //Normal conversion mode delay about (1/3.6864M)*50000=13.56ms

rADCCON=(1<<14)+(ADCPRS<<6); //ADCPRS En, ADCPRS Value

PRSCEN：ADC 转换时钟使能 14 位

(ADCPRS<<6) 说明如下： 6-13 位

PRSCVL：ADC 转换时钟预分频参数

NOTE: ADC Freqeuncy should be set less than PCLK by

5times. (Ex. PCLK=10MHZ, ADC Freq.< 2MHz)

rADCTSC=0xd3; //Wfait,XP_PU,XP_Dis,XM_Dis,YP_Dis,YM_En Y＋端的 X_ADC 可以认为是 X 轴的分压采样点

1101 0011

XY_PST ： measurement of X-Position or Y-Position 11 = Waiting for Interrupt Mode 每次发生此中断都，X 轴和 Y 轴坐标转换资料都可以从相应的资料寄存器中读出。

AUTO_PST：X/Y轴的自动转换模式使能位 AUTO_PST 0 = Normal ADC conversion

PULL_UP ： XP端的上拉电阻使能位 Pull-up Switch Enable 0 = XP Pull-up Enable. XP,XM,YP,YM对应触摸屏接口的TSXP,TSXM,TSYP,TSYM.

XP_SEN ： 设置nXPON输出状态 1 = XP Output Driver Disable.

XM_SEN ： 设置XMON输出状态

YP_SEN ： 设置nYPON输出状态

YM_SEN ： 设置YMON输出状态

```
pISR_ADC = (int)AdcTsAuto; //把真正的中断函数写进去。
rINTMSK=~BIT_ADC;          //ADC Touch Screen Mask bit clear
rINTSUBMSK=~(BIT_SUB_TC);
    Uart_Printf("\nPress any key to quit!\n");
    Uart_Printf("\nStylus Down, please..... \n");
    Uart_Getch();

    rINTSUBMSK|=BIT_SUB_TC;
    rINTMSK|=BIT_ADC;
    Uart_Printf("Touch Screen Test is Finished!!!\n");
```

==========中断开始了==========

```
void __irq AdcTsAuto(void)
{
U32 saveAdcdly;
```

```
if(rADCDAT0&0x8000) //中断刚开始没设置是按下还是抬起中断，所以加个 if 语句一定设置成按下触发中断
{
    //Uart_Printf("\nStylus Up!!\n");
    rADCTSC&=0xff;    // Set stylus down interrupt bit
}


//中断一触发说明光标按下，按下来进行 ADC 转换
//等待中断时【4】和【3】位一定要设置为 10，触发后从新设置
//(1<<3)     1 = XP Pull-up Disable.
//(1<<2)     1 = Auto Sequential measurement of X-position, Y-position      设置成自动检测 x，y 坐标
rADCTSC=(1<<3)|(1<<2);              //Pull-up disable, Seq. X,Y postion measure.

saveAdcdly=rADCDLY; //ADC Start or interval delay register
```

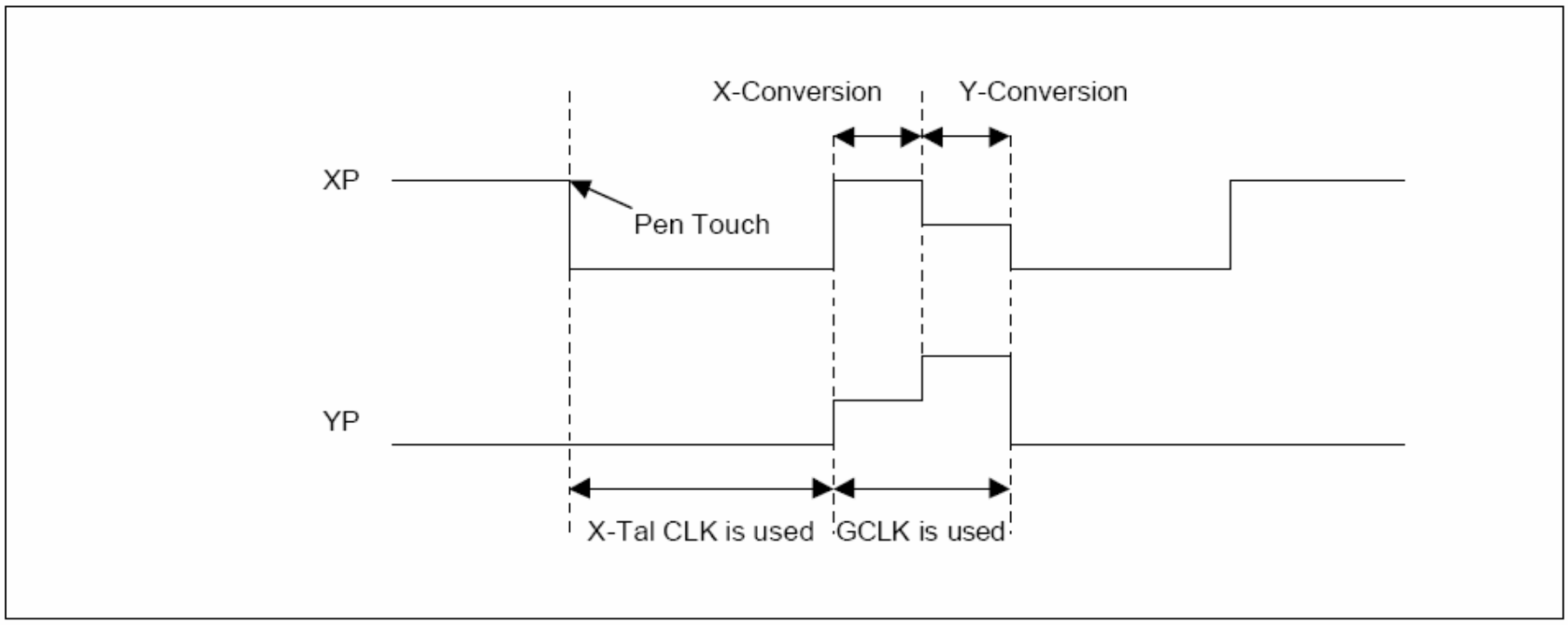


Figure 16-2. ADC and Touch Screen Operation signal

```
rADCDLY=40000;                  //Normal conversion mode delay about (1/50M)*40000=0.8ms
rADCCON|=0x1;                   //ADC 转换开始，开始后该位清零

while(rADCCON & 0x1);           //开始后第一位清零，检测是否开始
//检测 ADC 转换是否结束
while(!(rADCCON & 0x8000));     //check if EC(End of Conversion) flag is high, This line is necessary~!!
//检测中断 INT_ADC 是否已请求
while(!(rSRCPND & (BIT_ADC)));  //check if ADC is finished with interrupt bit
xdata=(rADCDAT0&0x3ff);   //得到 X  坐标
ydata=(rADCDAT1&0x3ff); //得到 Y 坐标
//check Stylus Up Interrupt.


//再度启用中断看光标是否已抬起，抬起则跳出 while 循环

/*Indicate the interrupt request status.
```

0 = The interrupt has not been requested.
1 = The interrupt source has asserted the
interrupt request.*/
如果你想收到另一个来此同一个中断源的有效请
求，你应该清除相应的位，然后使能中断。

rSUBSRCPND|=BIT_SUB_TC; //清除相应的位

ClearPending(BIT_ADC);//----------
rINTSUBMSK=~(BIT_SUB_TC);
rINTMSK=~(BIT_ADC);

```
register i;
    rSRCPND = bit;//晴空中断请求 一切都没有Not requested,
    rINTPND = bit;//中断未决
    i = rINTPND;
```

rADCTSC =0xd3; // //等待中断 Wfait,XP_PU,XP_Dis,XM_Dis,YP_Dis,YM_En 启用中断
rADCTSC=rADCTSC|(1<<8); // Detect stylus up interrupt signal. 设成抬起中断

while(1) //to check Pen-up state
{
 if(rSUBSRCPND & (BIT_SUB_TC)) //check if ADC is finished with interrupt bit 若 INT_TC 已请求 判断这个中断源是否已经请求了。
 {
  //Uart_Printf("Stylus Up Interrupt~!\n");
  break; //if Stylus is up(1) state
 }
}
Uart_Printf("count=%03d XP=%04d, YP=%04d\n", count++, xdata, ydata); //X-position Conversion data

//为下一次中断作准备
rADCDLY=saveAdcdly; // 本函数之前已经设置过了 //Normal conversion mode delay about (1/3.6864M)*50000=13.56ms
rADCTSC=rADCTSC&~(1<<8); // Detect stylus Down interrupt signal. 0 = Detect Stylus Down Interrupt Signal. 是按下中断而不是 UP
rSUBSRCPND|=BIT_SUB_TC; //作用是用于标示出哪个中断请求被触发 因此这个寄存器指出那个中断源在等待请求服务
注意SRCPND 的每个位都由中断源自动置位，
对于一个特定中断源的中断服务程序中，SRCPND 寄存器的相应位必须被清除目的是下次能正确得到同一个中断源的中断请求。如果你从中断服务程序返回却没有清除该位，中断控制器将操作好像又有同一个中断源的中断请求到来。换言之，如果SRCPND的一个特殊位置1，其总是认为一个有效的中断请求等待服务。
rINTSUBMSK=~(BIT_SUB_TC); // U nmask sub interrupt (TC)
ClearPending(BIT_ADC);
}