触摸屏控制实验

1953729 吴浩泽

## 实验目的

通过实验掌握触摸屏（TSP）的设计与控制方法。

掌握S3C2410X处理器的A/D转换功能。

## 实验设备

硬件：Embest EduKit-IV平台，ULINK2仿真器套件，PC机。

软件：μVision IDE for ARM集成开发环境，Windows 98/2000/NT/XP。

## 实验内容

点击触摸屏任意位置，将触摸屏坐标转换为液晶对应坐标后显示坐标位置。

## 实验原理

**1. 触摸屏(TSP)**

触摸屏（TSP: Touch Screen Panel）按其技术原理可分为五类：矢量压力传感式、电阻式、电容式、红外线式和表面声波式，其中电阻式触摸屏在嵌入式系统中用的较多。

表面声波触摸屏

表面声波触摸屏的边角有X、Y轴声波发射器和接收器，表面有X、Y轴横竖交叉的超声波传输。当触摸屏幕时，从触摸点开始的部分被吸收，控制器根据到达X、Y轴的声波变化情况和声波传输速度计算出声波变化的起点，即触摸点。

电容感应触摸屏

人相当于地，给屏幕表面通上一个很低的电压，当用户触摸屏幕时，手指头吸收走一个很小的电流，这个电流分别从触摸屏四个角或四条边上的电极中流出，并且理论上流经这四个电极的电流与手指到四角的距离成比例，控制器通过对这四个电流比例的计算，得出触摸点的位置。

红外线触摸屏

红外线触摸屏，是在显示器屏幕的前面安装一个外框，外框里有电路板，在X、Y方向排布红外发射管和红外接收管，一一对应形成横竖交叉的红外线矩阵。当有触摸时，手指或其它物体就会挡住经过该处的横竖红外线，由控制器判断出触摸点在屏幕的位置。

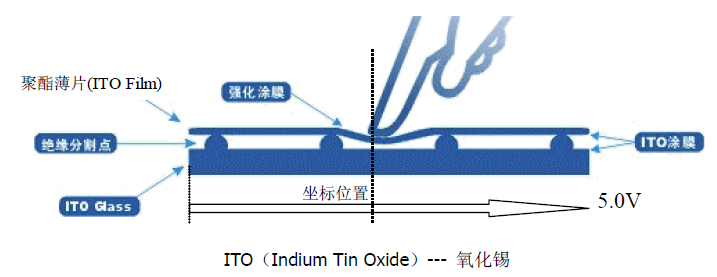
电阻触摸屏

电阻触摸屏是一个多层的复合膜，由一层玻璃或有机玻璃作为基层，表面涂有一层透明的导电层，上面再盖有一层塑料层，它的内表面也涂有一层透明的导电层，在两层导电层之间有许多细小的透明隔离点把它们隔开绝缘。工业中常用ITO（Indium Tin Oxide 氧化锡）导电层。当手指触摸屏幕时，平常绝缘的两层导电层在触摸点位置就有了一个接触，控制器检测到这个接通后，其中一面导电层接通Y 轴方向的5V 均匀电压场，另一导电层将接触点的电压引至控制电路进行A/D 转换， 得到电压值后与5V 相比即可得触摸点的Y 轴坐标，同理得出X 轴的坐标。这是所有电阻技术触摸屏共同的基本原理。电阻式触摸屏根据信号线数又分为四线、五线、六线...电阻触摸屏等类型。信号线数越多，技术越复杂，坐标定位也越精确。

四线电阻触摸屏，采用国际上评价很高的电阻专利技术：包括压模成型的玻璃屏和一层透明的防刮塑料，或经过硬化、清晰或抗眩光处理的尼龙，内层是透明的导体层，表层与底层之间夹着拥有专利技术的分离点（Separator Dots）。这类触摸屏适合于需要相对固定人员触摸的高精度触摸屏的应用场合，精度超过4096x4096，有良好的清晰度和极微小的视差。主要优点还表现在：不漂移， 精度高，响应快，可以用手指或其它物体触摸，防尘防油污等，主要用于专业工程师或工业现场。

Embest EduKit-IV 采用四线式电阻式触摸屏，点数为320x240，实验系统由触摸屏、触摸屏控制电路和数据采集处理三部分组成。

被按下的触摸屏状态：



触摸屏按下



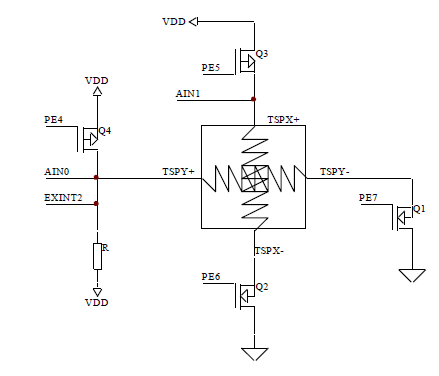
本实验台所使用的触摸屏外观图

**2. 等效电路结构**

电阻触摸屏采用一块带统一电阻外表面的玻璃板。聚酯表层紧贴在玻璃面上，通过小的透明的绝缘颗粒与玻璃面分开。聚酯层外表面坚硬耐用，内表面有一个传导层。当屏幕被触摸时，传导层与玻璃面表层进行电子接触。产生的电压就是所触摸位置的模拟表示。



等效电路示意图



触摸屏的等效电路

**3. 触摸屏原点**

电阻式触摸屏是通过电压的变化范围来判定按下触摸屏的位置，所以其原点就是触摸屏X电阻面和Y电阻面接通产生最小电压处。随着电阻的增大，A/D转换所产生数值不断增加，形成坐标范围。

触摸原点的确定有很多种方法，比如常用的对角定位法、四点定位法、实验室法等。

对角定位法

系统先对触摸屏的对角坐标进行采样，根据数值确定坐标范围，可采样一条对角线或两条对角线的顶点坐标。这种方法简单易用，但是需要多次采样操作并进行比较，以取得定位的准确性。本实验板采用这种定位方法。

四点定位法

同对角定位法一样，需要进行数据采样，只是需要采样四个顶点坐标以确定有效坐标范围，程序根据四个采样值的大小关系进行坐标定位。这种方法的定位比对角定位法可靠，所以被现在许多带触摸屏的设备终端使用。

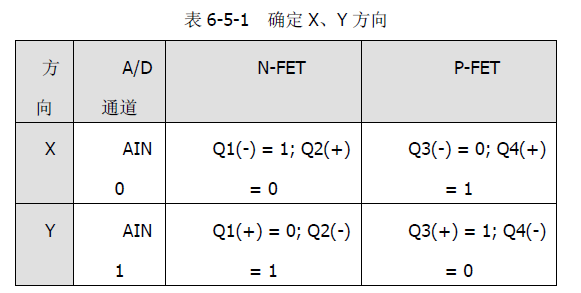
实验室法

触摸屏的坐标原点、坐标范围由生产厂家在出厂前根据硬件定义好。定位方法是按照触摸屏和硬件电路的系统参数，对批量硬件进行最优处理定义取得。这种方法适用于触摸屏构成的电路系统有较好的电气特性，且不同产品有较大相似性的场合。

**4. 触摸屏的坐标**

触摸屏坐标值可以采用多种不同的计算方式。常用的有多次采样取平均值法、二次平方处理法等。Embest EduKit-IV 教学系统的触摸屏坐标值计算采用取平均值法，首先从触摸屏的四个顶角得到两个最大值和两个最小值，分别标识为Xmax、Ymax和Xmin、Ymin。

参照图6-5-4 组成的坐标识别控制电路，X、Y方向的确定见表6-5-1。



注意：在Embest EduKit-IV 实验台中AIN0对应AIN7，AIN1对应AIN5

当触摸屏被按下时，首先导通FET管组Q2和Q4，X+与X- 回路加上+5V电源，同时将FET管组Q1和Q3关闭，断开Y+和Y-；再启动处理器的A/D转换通道0，电路电阻与触摸屏按下产生的电阻输出分量电压，并由A/D转换器将电压值数字化，计算出X轴的坐标。

接着先导通FET管组Q1和Q3，Y+与Y- 回路加上+5V电源，同时将MOS管组Q2和Q4关闭，断开X+和X-；再启动处理器的A/D转换通道1，电路电阻与触摸屏按下产生的电阻输出分量电压，并由A/D转换器将电压值数字化，计算出Y轴的坐标。

确定X、Y方向后坐标值的计算可通过以下方式求得（请参照程序设计）：

X = (Xmax - Xa)×480/(Xmax-Xmin) Xa = [X1+ X2+ ...+ Xn]/n

Y = (Ymax-Ya)×640/(Ymax-Ymin) Ya = [Y1+ Y2+ …+ Yn]/n

**5. A/D转换器（ADC）**

A／D转换器的类型

A／D转换器种类繁多，分类方法也很多。其中常见的包括以下分类：

按照工作原理可分为：计数式A／D转换器、逐次逼近型、双积分型和并行A／D转换几类。

按转换方法可分为：直接A／D转换器和间接A／D转换器。所谓直接转换是指将模拟量转换成数字量；而间接转换则是指将模拟量转换成中间量，再将中间量转换成数字量。

按分辨率可分为：二进制的4位、6位、8位、10位、12位、14位、16位和BCD码的3位半、4位半、5位半等。

按转换速度可分为：低速（转换时间≥1s）、中速（转换时间≤lms）、高速（转换时间≥1μs）和超高速（转换时间≤1ns）。

按输出方式可分为：并行、串行、串并行等。

A／D转换器的工作原理

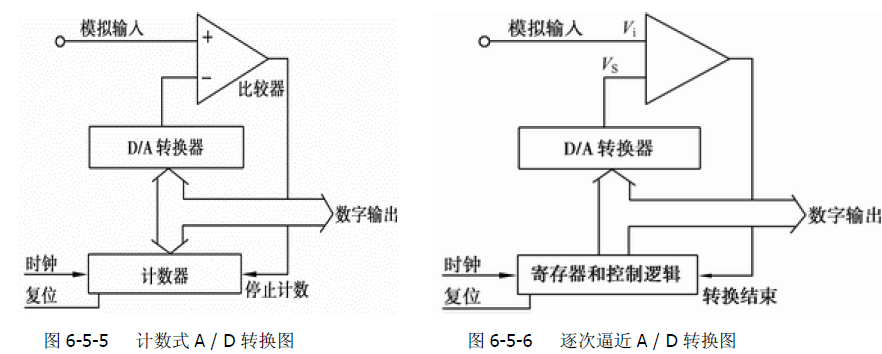
A／D转换的方法很多，下面介绍常用的A／D转换原理。

(1) 计数式

这种A／D转换原理最简单直观，它由D／A转换器、计数器和比较器组成，如图6-5-5所示。计数器由零开始计数，将其计数值送往D／A转换器进行转换，将生成的模拟信号与输入模拟信号在比较器内进行比较，若前者小于后者，则计数值加1，重复D／A转换及比较过程。因为计数值是递增的，所以D／A输出的模拟信号是一个逐步增加的量，当这个信号值与输出模拟量比较相等时（在允许的误差范围内），比较器产生停止计数信号，计数器立即停止计数。此时D／A转换器输出的模拟量就为模拟输入值，计数器的值就是转换成的相应的数字量值。 这种A／D转换器结构简单、原理清楚，但是转换速度与精度之间存在严重矛盾即若要提高转换速度，则转换器输出与输入的误差就越大，反之亦然。所以在实际中很少使用。

(2) 逐次逼近式

逐次逼近A／D转换器是由一个比较器、D／A转换器、寄存器及控制逻辑电路组成，如图6-5-6所示。和计数式相同，逐次逼近式也要进行比较，以得到转换数字值。但在逐次逼近式中，是用一个寄存器控制D／A转换器。逐次逼近式是从高位到低位依次开始逐位试探比较。S3C2410X处理器集成了这种A/D转换器。



逐次逼近式转换过程如下：初始时寄存器各位清为0，转换时，先将最高位置1，送入D／A 转换器，经D／A 转换后生成的模拟量送入比较器中与输入模拟量进行比较，若Vs＜Vi，该位的1 被保留，否则被清除。然后次高位置为1，将寄存器中新的数字量送入D／A 转换器，输出的Vs 再与Vi 比较，若Vs＜Vi，保留该位的1，否则清除。重复上述过程，直至最低位。最后寄存器中的内容即为输入模拟值转换成的数字量。

对于n 位逐次逼近式A／D 转换器，要比较n 次才能完成一次转换。因此，逐次逼近式A／D 转换器的转换时间取决于位数和时钟周期。转换精度取决于D／A 转换器和比较器的精度，一般可达0.01％，转换结果也可串行输出。逐次逼近式A／D 转换器可应用于许多场合，是应用最为广泛的一种A／D 转换器。

A／D 转换器主要性能指标

(1) 分辨率

分辨率是指A／D 转换器能分辨的最小模拟输入量。通常用能转换成的数字量的位数来表示， 如8 位、10 位、12 位、16 位等。位数越高，分辨率越高。如分辨率为10 位，表示A／D 转换器能分辨满量程的1／1024 的模拟增量，此增量亦可称为1LSB 或最低有效位的电压当量。

(2) 转换时间

转换时间是A／D 转换完成一次转换所需的时间。即从启动信号开始到转换结束并得到稳定数字输出量为止的时间。一般来说，转换时间越短则转换速度就越快。不同的A／D 转换器转换时间差别较大，通常为微秒数量级。

(3) 量程

量程是指所能转换的输入电压范围。

(4) 绝对精度

A／D 转换器的绝对精度是指在输出端产生给定的数字代码的情况下，实际需要的模拟输入值与理论上要求的模拟输入值之差。

(5) 相对精度

相对精度是指A／D 转换器的满刻度值校准以后，任意数字输出所对应的实际模拟输入值（中间值）与理论值（中间值）之差。线性A／D转换器的相对精度就是它的线性度。精度代表电气或工艺精度，其绝对值应小于分辨率，因此常用1 LSB的分数形式来表示。

**6. S3C2410X处理器的A/D转换**

处理器内部集成了采用近似比较算法（计数式）的8路10位ADC，集成零比较器，内部产生比较时钟信号；支持软件使能休眠模式，以减少电源损耗。其主要特性：

a) 精度（Resolution）：10-bit

b) 微分线性误差（Differential Linearity Error）：± 1.5 LSB

c) 积分线性误差（Integral Linearity Error）：± 2.0 LSB

d) 最大转换速率（Maximum Conversion Rate）：500 KSPS

e) 输入电压（Input voltage range）：0-3.3V

f) 片上采样保持电路

g) 正常模式

h) 单独X,Y坐标转换模式

i) 自动X,Y坐标顺序转换模式

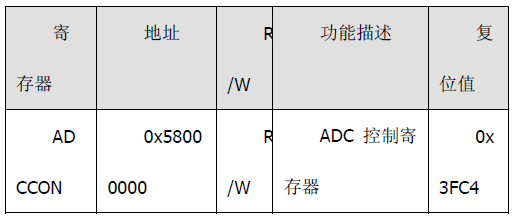
**j)** 等待中断模式

**7. S3C2410X处理器TSP控制器**

寄存器组

处理器集成的TSP只使用到3个个寄存器，即ADC控制寄存器（ADCCON）、触摸屏控制寄存器（ADCTSC）、ADC数据寄存器（ADCDAT）。

ADC控制寄存器（ADCCON）



ADCCON[15]: A/D转换结束标志

0：A/D转换正在进行； 1：A/D转换结束

ADCCON[14]: AD转换预分频允许

0：不允许预分频 1：允许预分频

ADCCON[13:6]:预分频值PRSCVL

PRSCVL在0到255之间，实际的分频值为PRSCVL+1

ADCCON[5:3]: 模拟信道输入选择

000 = AIN0 001 = AIN1 010 = AIN2 011 = AIN3

100 = AIN4 101 = AIN5 110 = AIN6 111 = AIN7

ADCCON[2]: 待机模式选择位

0:正常模式 1:待机模式

ADCCON[1]: A/D转换读－启动选择位

0:禁止Start-by-read 1:允许Start-by-read

ADCCON[0]: A/D转换器启动

0:A/D转换器不工作 1: A/D转换器开始工作

触摸屏控制寄存器（ADCTSC）



ADCTSC[8]: 保留，必须为0

ADCTSC[7]: 选择YMON输出值

0：输出为0 1：输出为1

ADCTSC[6]: 选择YPON输出值

0：输出为0 1：输出为1

ADCTSC[5]: 选择XMON输出值

0：输出为0 1：输出为1

ADCTSC[4]: 选择XPON输出值

0：输出为0 1：输出为1

ADCTSC[3]: 上拉开关使能

0：上拉使能 1：上拉禁止

ADCTSC[2]: 自动按顺序转换X,Y坐标选择位

0：正常模式 1：自动按顺序转换X,Y坐标使能

ADCTSC[1:0]: 手工设置X,Y坐标转换

00：无操作 01：X坐标转换 10：Y坐标转换 11：等待中断模式

ADC数据寄存器（ADCDAT0,ADCDAT1）



DCDAT0[15]: 等待中断模式，Stylus电平选择

0:低电平 1:高电平

ADCDAT0[14]: 自动按照先后顺序转换X,Y坐标

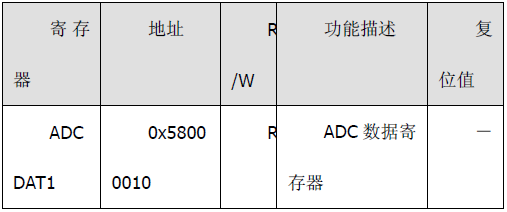
0:正常ADC顺序 1:按照先后顺序转换

ADCDAT0[13:12]: 自定义X,Y位置

00:无操作模式 01:测量X位置 10:测量Y位置 11:等待中断模式

ADCDAT0[11:10]: 保留

ADCDAT0[9:0]: X坐标转换数据值



ADCDAT1[15:10]与ADCDAT0[15:10]功能相同

ADCDAT0[9:0]: Y坐标转换数据值

A/D转换的转换时间计算

例如PCLK为50MHz，PRESCALER=49；所有10位转换时间为：

50 MHz / (49+1) =1MHz

转换时间为1/(1M/5 cycles)＝5us，

注意，A/D转换器的最大工作时钟为2.5MHz，所以最大的采样率可以达到500ksps。

**8. 电路设计**



当手指触摸屏幕时，平常绝缘的两层导电层在触摸点位置就有了一个接触，控制器检测到这个接通后，产生中断通知CPU进行A/D转换；具体原理如下：当触摸屏被按下时，首先导通FET管组Q602和Q604，X轴回路加上+5V电源，同时将FET管组Q1和Q3关闭；再启动处理器的A/D转换通AIN7，电路电阻与触摸屏按下产生的电阻输出分量电压，并由A/D转换器将电压值数字化，计算出X轴的坐标。中断处理程序通过导通不同MOS管组，使接触部分与控制器电路构成电阻电路，并产生一个电压降作为坐标值输出。

**9. 软件程序设计**

实验主要是对S3C2410X中的TSP模块进行操作，所以软件程序也主要是对A/D和TSP模块中的寄存器进行操作，包括对ADC控制寄存器（ADCCON）、TSP控制寄存器（ADCTSC）、ADC数据寄存器（ADCDAT）的读写操作。同时为了观察点触的触摸屏的位置信息，可以通过串口在超级终端里面观察。

## 实验步骤

**1. 准备实验环境**

使用ULINK2仿真器连接Embest EduKit-IV实验平台的主板JTAG接口；使用Embest EduKit-IV实验平台附带的交叉串口线，连接实验平台主板上的COM2和PC机的串口（一般PC只有一个串口，如果有多个请自行选择，笔记本没有串口设备的可购买USB转串口适配器扩充）；使用Embest EduKit-IV实验平台附带的电源适配器，连接实验平台主板上的电源接口。

**2. 串口接收设置**

在PC机上运行windows自带的超级终端串口通信程序，或者使用实验平台附带光盘内设置好了的超级终端，设置超级终端：波特率115200、1位停止位、无校验位、无硬件流控制，或者使用其它串口通信程序。（注：超级终端串口的选择根据用户的PC串口硬件不同，请自行选择，如果PC机只有一个串口，一般是COM1）

**3. 打开实验例程**

1）拷贝实验平台附带光盘DISK3\_S3C2410\03-Codes\01-MDK\Mini2410-IV文件夹到MDK的安装路径：Keil\ARM\Boards\Embest\（如果本实验之前已经拷贝，可以跳过这一步）。（注：用户也可拷贝工程到任意目录，本实验为了便于教学，故统一实验路径）；

2）运行μVision IDE for ARM软件，点击菜单栏“Project”，选择“Open Project…”，在弹出的对话框选择实验例程目录6.5\_Tsp\_Test子目录下的Tsp\_Test.Uv2工程。

3）默认打开的工程在源码编辑窗口会显示实验例程的说明文件readme.txt，详细阅读并理解实验内容。

4）工程提供了两种运行方式：一是下载到SDRAM中调试运行，二是固化到Nor Flash中运行。用户可以在工具栏Select Target下拉框中选择在RAM中调试运行还是固化Flash中运行。如下图所示：



选择运行方式

下面实验将介绍下载到SDRAM中调试运行，所以我们在Select Target下拉框中选择Uart\_Test IN RAM。

5）接下来开始编译链接工程，在菜单栏“Projiet”选择“Build target”或者“Rebuild all target files”编译整个工程，用户也可以在工具栏单击“”或者“”进行编译。

6） 编译完成后，在输出窗口可以看到编译提示信息，比如“".\SDRAM\TSP\_Test.axf" - 0 Error(s), 1 Warning(s).”，如果显示“0 Error(s)”即表示编译成功。

7）拨动实验平台电源开关，给实验平台上电，单击菜单栏Debug->Start/Stop Debug Session项将编译出来的映像文件下载到SDRAM中，或者单击工具栏“”按钮来下载。

8）下载完成后，单击菜单栏Debug->Run项运行程序，或者单击工具栏“”按钮来全速运行程序。用户也可以使用进行单步调试程序。

9）全速运行后，用户可以在超级终端看到程序运行的信息，出现“Please input words, then press Enter”提示后输入想要发送的数据，并已回车作为发送字符串的结尾标志。

10）用户可以Stop程序运行，使用μVision IDE for ARM的一些调试窗口跟踪查看程序运行的信息。

注：如果在第4）步用户选择在Flash中运行，则编译链接成功后，单击菜单栏Flash->Download项将程序固化到NorFlash中，或者单击工具栏按钮“”固化程序，从实验平台的主板拔出JTAG线，给实验平台重新上电，程序将自动运行。

**4. 观察实验结果**

在执行到第8）步时，可以看到超级终端上输出如下字符。

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | | |  | | --- | | boot success...  ===============================================  | WELCOME TO EMBEST EDUKIT IV |  ===============================================  Touch Screen Test Example.  Stylus Down!!  X-Posion[AIN5] is 12  Y-Posion[AIN7] is 6  Stylus Down!!  X-Posion[AIN5] is 0010  Y-Posion[AIN7] is 49 | | |

## 实验参考程序

|  |  |
| --- | --- |
| |  | | --- | | /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* name: tsp\_test  \* func:  \* para: none  \* ret: none  \* modify:  \* comment:  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  void tsp\_test(void)  {  uart\_printf(" Touch Screen Test Example.\n");  rADCDLY = 50000; // ADC Start or Interval Delay  rGPGCON |= 0xFF000000;  rGPGUP = 0xFFFF;  rADCTSC = (0<<8) | (1<<7) | (1<<6) | (0<<5) | (1<<4) | (0<<3) | (1<<2) | (0);  //auto sequential x/y position conversion,no operation, XP pull-up  rADCCON = (1<<14) | (ADCPRS<<6) | (5<<3) | (0<<2) | (0<<1) | (0);  // Enable Prescaler,Prescaler,AIN7/5 fix,Normal,Disable read start,No operation  rADCTSC = (0<<8) | (1<<7) | (1<<6) | (0<<5) | (1<<4) | (0<<3) | (0<<2) | (3);  //YM:GND,YP:AIN5,XM:Hi-z,XP:external voltage,XP pullup En,AUTO sequential,Waiting for interrupt mode  delay(100);  pISR\_ADC = (UINT32T)tsp\_int; // pISR\_ADC  rINTMSK &= ~(BIT\_ADC);  rINTSUBMSK = ~(BIT\_SUB\_TC);  g\_nKeyPress = 1;  while(g\_nKeyPress==1) // only for board test to exit  {  if(uart\_getkey()) return; // or press any key to exit  }  rINTSUBMSK |= BIT\_SUB\_TC;  rINTMSK |= BIT\_ADC;  uart\_printf(" end.\n");  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* name: tsp\_int  \* func: Touch screen interrupt handler (ADC)  \* para: none  \* ret: none  \* modify:  \* comment:  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  void \_\_irq tsp\_int(void)  {  UINT32T szPos[40];  rINTSUBMSK |= (BIT\_SUB\_ADC | BIT\_SUB\_TC);// Mask sub interrupt (ADC and TC)  // TC(Touch screen Control) Interrupt  if( rADCTSC & 0x100)  {  //uart\_printf(" Stylus Up!!\n");  rADCTSC&=0xff; // Set stylus down interrupt  }  else  {  uart\_printf(" Stylus Down!!\n");  szPos[30] = g\_nPosX;  szPos[34] = g\_nPosY;  // <X-Position Read>  //Hi-Z,AIN5,GND,Ext vlt,Pullup Dis,Normal,X-position  rADCTSC = (0<<8)|(0<<7)|(1<<6)|(1<<5)|(0<<4)|(0<<3)|(0<<2)|(1);  //adc input ain5  rADCCON = (1<<14)|(39<<6)|(5<<3)|(0<<2)|(1<<1)|(0);  rADCDAT0;  delay(10);  for(i = 0,g\_nPosX=0; i<nSampleNo; i++)  {  while(!(0x8000 & rADCCON)); // Check ECFLG  szPos[i] = (0x3ff & rADCDAT0);  g\_nPosX += szPos[i];  }  g\_nPosX = g\_nPosX/nSampleNo;  g\_nPosX = (g\_nPosX-45)\*640/685;  uart\_printf(" X-Posion[AIN5] is %04d\n", g\_nPosX);  // <Y-Position Read>  //GND,Ext vlt,Hi-Z,AIN7,Pullup Dis,Normal,Y-position  rADCTSC = (0<<8)|(1<<7)|(0<<6)|(0<<5)|(1<<4)|(0<<3)|(0<<2)|(2);  //adc input ain7  rADCCON = (1<<14)|(39<<6)|(7<<3)|(0<<2)|(1<<1)|(0);  rADCDAT1;  delay(10);  for(i = 0,g\_nPosY=0; i<nSampleNo; i++)  {  while(!(0x8000 & rADCCON)); // Check ECFLG  szPos[i] = (0x3ff & rADCDAT1);  g\_nPosY += szPos[i];  }  g\_nPosY = g\_nPosY/nSampleNo;  g\_nPosY = (g\_nPosY-70)\*480/340;  uart\_printf(" Y-Posion[AIN7] is %04d\n", g\_nPosY);  //GND,AIN,Hi-z,AIN,Pullup En,Normal,Waiting mode  rADCTSC=(1<<8)|(1<<7)|(1<<6)|(0<<5)|(1<<4)|(0<<3)|(0<<2)|(3);  }  #ifdef BOARDTEST  sprintf(&szPos, "(X1,Y1):(%d,%d)",szPos[30],szPos[34]);  print\_lcd(195,170,0x20,&szPos);  sprintf(&szPos, "(X2,Y2):(%d,%d)",g\_nPosX,g\_nPosY);  print\_lcd(195,178,0x1c,&szPos);  #endif  rSUBSRCPND |= BIT\_SUB\_TC;  rINTSUBMSK = ~(BIT\_SUB\_TC); // Unmask sub interrupt (TC)  ClearPending(BIT\_ADC);  } | |

## 读书笔记

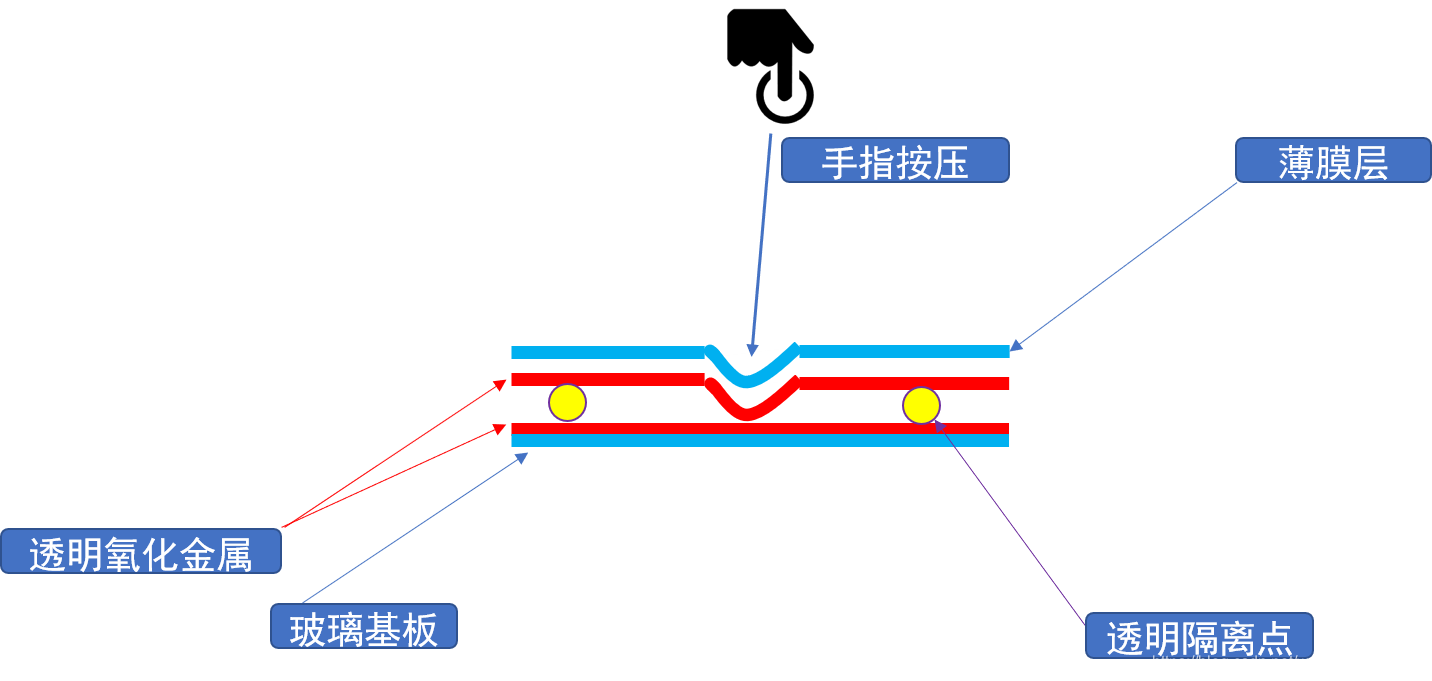
1. 简介

1.1 电阻式触摸屏

电阻式触摸屏利用压力感应进行触点检测控制，需要直接应力接触， 通过检测电阻来定位触摸位置 。

1.1.1 电阻式触摸屏的原理

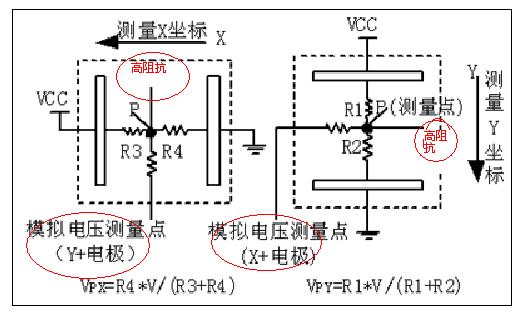
电阻触摸屏的主要部分是一块与显示器表面非常配合的电阻薄膜屏，这是一种多层的复合薄膜，它以一层玻璃或硬塑料平板作为基层，表面涂有一层透明氧化金属（透明的导电电阻）导电层，上面再盖有一层外表面硬化处理、光滑防擦的塑料层、它的内表面也涂有一层涂层、在他们之间有许多细小的（小于 1/1000 英寸）的透明隔离点把两层导电层隔开绝缘。



当手指触摸屏幕时，两层导电层在触摸点位置就有了接触，电阻发生变化，在 X 和 Y 两个方向上产生信号，然后送触摸屏控制器。控制器侦测到这一接触并计算出（ X Y ）的位置，再根据获得的位置模拟鼠标的方式运作。

电阻式触摸屏都需要一个AD 转换器， 所以一般来说驱动屏幕需要一个控制器芯片。这种屏幕可以用四线、五线、七线或八线来产生屏幕偏置电压，同时读回触摸点的电压。

1.1.2 电阻式触摸屏得到触点坐标



如上图所示，但在 X 轴方向的点极施加一定的电压，而 Y 轴方向不加电压时，在 X 轴的平行电场中，触点处的电压值可以在 Y 轴的测量点得到，知道了测量点处 Y 轴的电压（X对地电阻的电压），也就确定了 X 轴上的坐标；同理，当 Y 轴方向施加固定的电压时，可在 X 轴的测量点上得到对应的电压。经过两次的测量，就可以得出触点（X,Y）的坐标了。

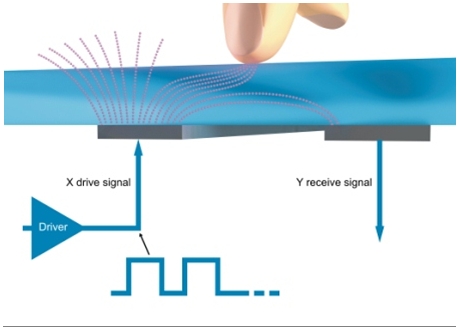
1.1.3 电阻式触摸屏的优缺点

优点： 精度高、价格便宜、抗干扰能力强、稳定性好 。

缺点： 容易被划伤、透光性不太好、不支持多点触摸。直接的感觉就是体验不如电容屏幕。

1.2 电容式触摸屏

电容屏利用人体感应进行触点检测控制，不需要直接接触或只需要轻微接触，通过检测感应电流来定位触摸坐标 。



当手指点击屏幕，会从接触点吸收小量电流，造成角落电极的压降，利用感应人体微弱电流的方式来达到触控的目的。

1.2.1 表面电容式电容触摸屏

表面电容式触摸屏技术是利用 ITO 铟锡氧化物，是 一种透明的导电材料导电膜，通过电场感应方式感测屏幕表面的触摸行为进行。但是表面电容式触摸屏有一些局限性，它只能识别一个手指或者一次触摸。

1.2.2 投射式电容触摸屏

投射电容式触摸屏是传感器利用触摸屏电极发射出静电场线。一般用于投射电容传感技术的电容类型有两种： 自我电容和交互电容 。

自我电容： 又称绝对电容，自我电容通常是指扫描电极与地构成的电容。在玻璃表面有用 ITO 制成的横向与纵向的扫描电极，这些电极和地之间就构成一个电容的两极。当用手或触摸笔触摸的时候就会并联一个电容到电路中去，从而使在该条扫描线上的总体的电容量有所改变。在扫描的时候，控制 IC 依次扫描纵向和横向电极，并根据扫描前后的电容变化来确定触摸点坐标位置。笔记本电脑触摸输入板就是采用的这种方式。笔记本电脑的输入板采用 X\*Y 的传感电极阵列形成一个传感格子， 当手指靠近触摸输入板时，在手指和传感电极之间产生一个小量电荷。采用特定的运算法则处理来自行、列传感器的信号来确定手指的位置。

交互电容： 又叫做跨越电容，它是在玻璃表面的横向和纵向的 ITO 电极的交叉处形成电容。交互电容的扫描方式就是扫描每个交叉处的电容变化，来判定触摸点的位置。当触摸的时候就会影响到相邻电极的耦合，从而改变交叉处的电容量，交互电容的扫面方法可以侦测到每个交叉点的电容值和触摸后电容变化，因而它需要的扫描时间与自我电容的扫描方式相比要长一些，需要扫描检测 X\*Y 根电极。 目前智能手机 平板电脑等的触摸屏 ，都是采用 交互电容技术。

1.2.3 投射式电容触摸屏——交互电容详解

投射式电容触摸屏采用纵横两列电极组成感应矩阵，来感应触摸。以两个交叉的电极矩阵，即： X 轴电极和 Y 轴电极，来检测每一 格感应单元的电容变化 。

.2.4 电容式触摸屏的优缺点

优点： 手感好、无需校准、支持多点触摸、透光性好；

缺点： 成本高、精度不高、抗干扰能力差。

2. 电容触摸驱动 IC——OTT2001A介绍

OTT2001A ，最多支持 208 个通道。支持 SPI/IIC 接口。 IIC 接口模式下，该驱动 IC 与 STM32 的连接仅需要 4 根线： SDA 、 SCL 、 RST 和 INT，SDA 和 SCL 是 IIC 通信用的， RST 是复位脚（低电平有效）， INT 是中断输出信号。

2.1 寄存器介绍

（1）手势 ID 寄存器

手势 ID 寄存器（ 00H ）用于告诉 MCU ，哪些点有效，哪些点无效，从而读取对应的数据。

2）传感器控制寄存器（ODH）

传感器控制寄存器（ODH ），该寄存器也是 8 位，仅最高位有效，其他位都是保留，当最高位为 1 的时候，打开传感器（开始检测），当最高位设置为 0 的时候，关闭传感器（停止检测）。

（3）坐标数据寄存器（共 20 个）

坐标数据寄存器总共有 20 个，每个坐标占用 4 个寄存器

3. 软件分析

这里简要的分析一下电阻屏的驱动。

3.1 通信方式实现

/\*软件模拟SPI写数据\*/

void TP\_Write\_Byte(u8 num)

{

u8 count = 0;

for(count = 0;count < 8;count++)

{

if(num&0x80)

TDIN = 1;

else

TDIN = 0;

num <<= 1;

TCLK = 0;

TCLK = 1; //上升沿有效

}

} /\*从触摸屏读取 ADC 的数值\*/

u16 TP\_Read\_AD(u8 CMD)

{

u8 count = 0;

u16 Num = 0;

TCLK = 0; //先拉低时钟

TDIN = 0; //拉低数据线

TCS = 0; //选中触摸屏IC

TP\_Write\_Byte(CMD);//发送命令字

delay\_us(6);//ADS7846的转换时间最长为6us

TCLK = 0;

delay\_us(1);

TCLK = 1; //给1个时钟，清除BUSY

TCLK = 0;

for(count = 0;count < 16;count++)//读出16位数据,只有高12位有效

{

Num <<= 1;

TCLK = 0; //下降沿有效

TCLK = 1;

if(DOUT)

Num++;

}

Num >>= 4; //只有高12位有效，移除低四位

TCS = 1; //释放片选

return(Num);

}