青风手把手教你学 stm32f030 系列教程

---------- 库函数操作版本

出品论坛： www.qfv8.com 青风电子社区

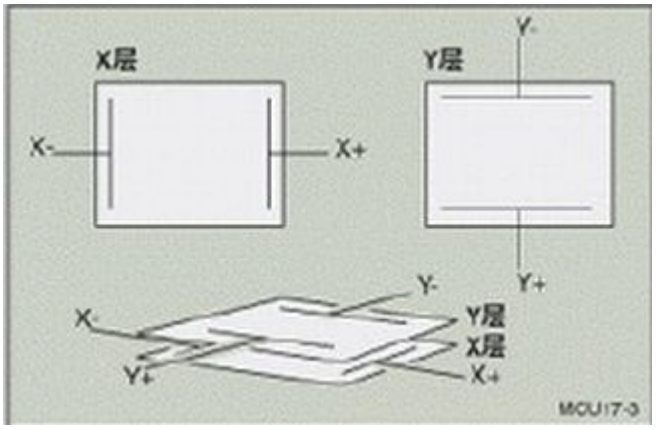
Copyright© 2009-2015 青风电子社区 ALL rights reserved

作者： 青风
出品论坛: www.qfv8.com
淘宝店：http://qfv5.taobao.com
QQ 技术群：241364123
硬件平台：QF-STM32F030 开发板

2.14 TFT 触摸的使用

2.14.1 原理分析

目前电子行业上，触摸屏被大量使用，触摸屏能够大大提高了人机交互的友好性，在消费电子，工业控制上大量使用。这一节我们将讨论下如何使用触摸屏。在液晶屏上有一层触摸屏贴在 TFT 液晶上，引出 4 个输出管脚 X+,X-,Y+,Y-,实际上这种触摸屏就是 4 线的电阻屏，这四个管脚的表示如下图所示：



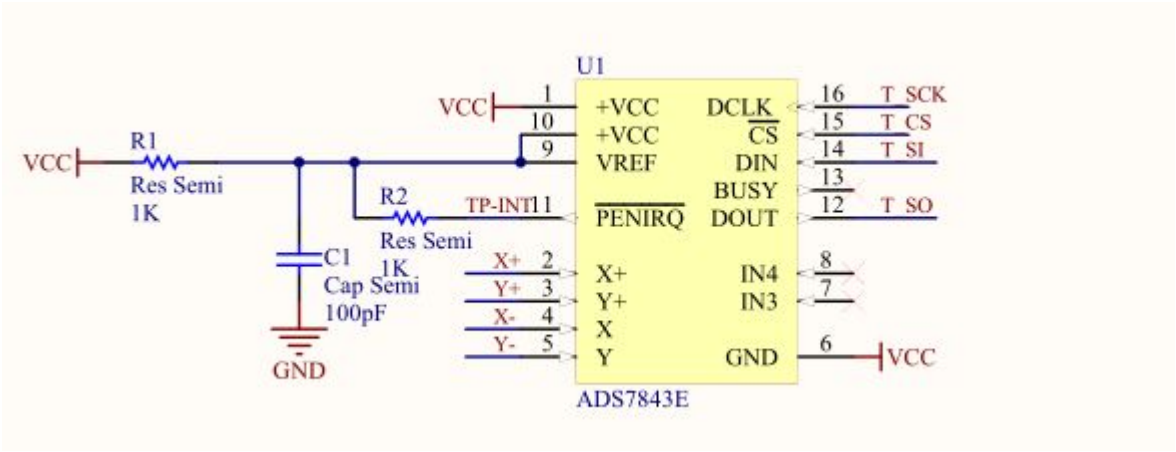
我们要做的工作就是把这四线信号转换成对应液晶屏的位置，这个工作还是比较麻烦的。首先电阻屏出来的信号是模拟信号，我们首先要把这个信号通过 ADC 转换成数字信号，然后在转换成对应的 X,Y 坐标。然后在对应的坐标上面画点就可以了。原理其实也很简单，下面就来设计我们的触摸屏驱动了。

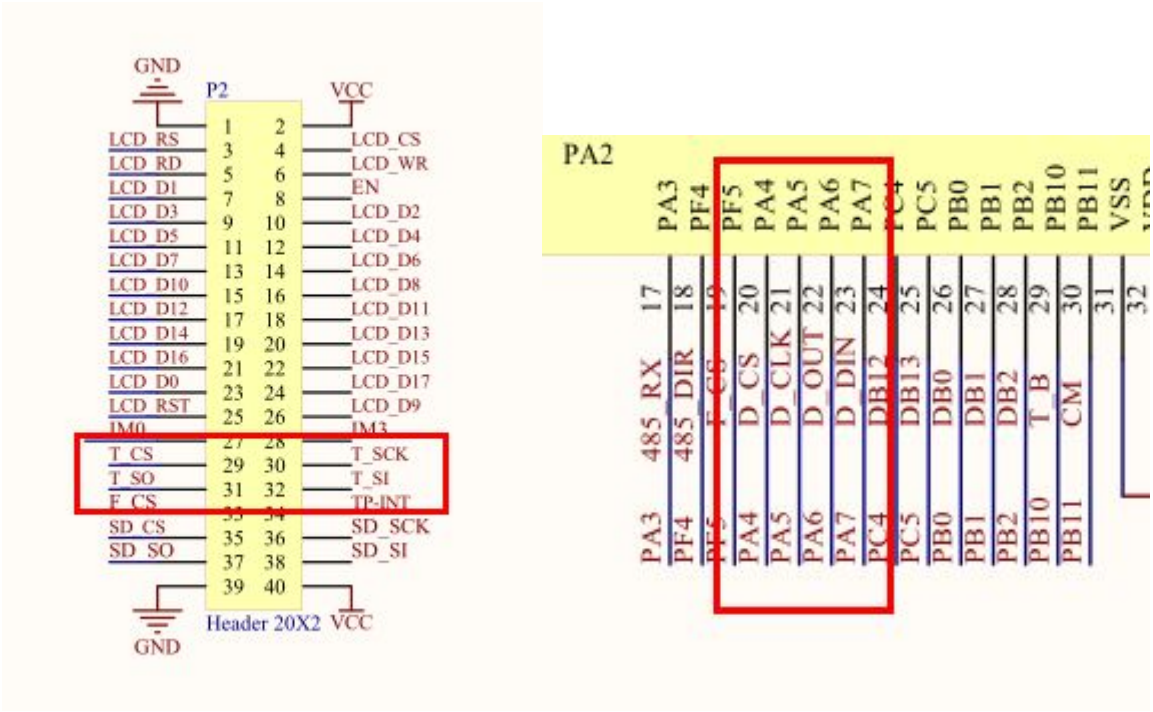
2.14.2 硬件准备：

我们要把 阻屏出来的信号通过 ADC 转换成数字信号，就需要使用到 AD,AD 转换的方式采用差分输入，我们可以直接接开发板上的 ADC，当然为了减小 MCU 的负担，我们增加外部的 adc，采用外部 ADC 转换，转换的结果通过 SPI 接口输入到 MCU 中去。开发板采用 12 位的 ADC 芯片 ADC7843 或者 XTP2046，两种通用。硬件电路

Copyright© 2009-2015 青风电子社区 ALL rights reserved

如下图所示：





端口配置：

硬件连接：// ----------

// PB10---T-INT
// PA4----T-CS
// PA5----T-CLK
// PA6----T-OUT
// PA7----T-DIN
// SPI1-- AF0
// ----------

说明：其中 T-CS :ADC 芯片片选引脚，低电平时有效。
T-CLK :ADC 芯片时钟信号引脚。

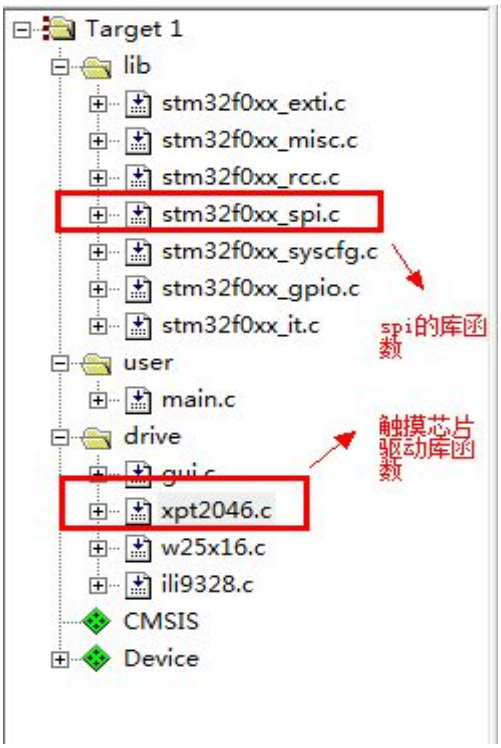
Copyright© 2009-2015 青风电子社区 ALL rights reserved

T-OUT，T-DIN：ADC 芯片 SPI 通信引脚。

T-INT：ADC 中断采样引脚。

2.19.3 软件准备：

我们在 keil5.0 中建立工程项目如下所示，其中 XPT2046.C 文件就是程序员需要编写的 ADC 芯片驱动文件：



我们编写程序的时候首先其实应该手动画一下程序流程图，第一步要执行什么，第二步要执行什么，这样整个程序的框架就给搭建起来了，后面在来往框架里面填物料。教程画流程图比较麻烦，我就直接和大家讨论了：

按照上面分析触摸屏使用步骤，第一步应该是把阻屏出来的信号通过 ADC 转换成数字信号，当片选 ADC 后，ADC 启动转换，转换后的信号需要通过 SPI 接口被我们读取，那么 SPI 接口就应该被初始化，初始化我们的硬件接口，这个在代码里有，可以直接查找代码。初始化后我们就开始读取 ADC 转换的值了：

```
01.    uint8_t    WR_Cmd(uint8_t cmd)
02.  {     /* Wait for SPI1 Tx buffer empty */
03.        while (SPI_I2S_GetFlagStatus(SPI1, SPI_I2S_FLAG_TXE) == RESET);
04.        /* Send SPI1 data */
05.        SPI_SendData8(SPI1,cmd);
06.        /* Wait for SPI1 data reception */
07.        while (SPI_I2S_GetFlagStatus(SPI1, SPI_I2S_FLAG_RXNE) == RESET);
08.        /* Read SPI1 received data */
09.        return SPI_ReceiveData8(SPI1);
10.        }
```

Copyright© 2009-2015 青风电子社区 ALL rights reserved

11.

上面的代码就是读一个字节，我们可以判定是读取 X 还是 Y 轴，代码如下所示：

```
12.  uint16_t ADS_Read_XY(uint8_t xy)
13.  {
14.      uint16_t i, j;
15.      uint16_t buf[READ_TIMES];
16.      uint16_t sum=0;
17.      uint16_t temp;
18.      for(i=0;i<READ_TIMES;i++)
19.      {
20.          buf[i]=ADS_Read_AD(xy);
21.      }
22.      for(i=0;i<READ_TIMES-1; i++)//
23.      {
24.          for(j=i+1;j<READ_TIMES;j++)
25.          {
26.              if(buf[i]>buf[j])//
27.              {
28.                  temp=buf[i];
29.                  buf[i]=buf[j];
30.                  buf[j]=temp;
31.              }
32.          }
33.      }
34.      sum=0;
35.      for(i=LOST_VAL;i<READ_TIMES-LOST_VAL;i++)sum+=buf[i];
36.      temp=sum/(READ_TIMES-2*LOST_VAL);
37.      return temp;
38.  }
```

之后一次读取一次转换的 x,y 值：

```
39.  uint8_t Read_ADS(uint16_t *x,uint16_t *y)
40.  {
41.      uint16_t xtemp,ytemp;
42.      xtemp=ADS_Read_XY(CMD_RDX);
43.      ytemp=ADS_Read_XY(CMD_RDY);

44.      if(xtemp<100||ytemp<100)return 1;//
45.      *x=xtemp;
46.      *y=ytemp;
47.      return 0;//
48.  }
49.
```

当此时读取出来的数字量我们还需要转换成,X,Y 在液晶屏幕上对应的坐标轴这里面用了一个比较简单的代码，直接平均分配，没用使用算法，参数需要我们手动校准：

Copyright© 2009-2015 青风电子社区 ALL rights reserved

```
50. void Change_XY(void)
51. {
52.     Pen_Point.X_Coord=(240-(Pen_Point.X_ADC-108)/7.462);
53.     Pen_Point.Y_Coord=(320-(Pen_Point.Y_ADC-136)/5.584);
54. }
```

然后我们就要判定什么时候起的这个转换坐标轴的操作了，应该是有手触摸发生的时候，也就是 ADC 中断管脚有信号的时候，引起一个外部中断，当这个中断发生了，我们认为有 ADC 进行了信号转换，那么就需要启动转换坐标轴的过程了：

```
55. uint8_t Read_Once(void)
56. {
57.       touch_flag=0;
58.     Pen_Point.Pen_Sign=Pen_Up;
59.     if(Read_ADS2(&Pen_Point.X_ADC,&Pen_Point.Y_ADC)==0)
60.     {
61.         while(SPI_TOUHC_INT==0);
62.         Change_XY();
63.         return 0;
64.     }
65.     else return 1;
66. }
```

上面的过程都完成了，我们就可以调用来操控我们的触摸屏了，首先进行初始化，初始化相应的端口和中断：

```
67.
68. void Touch_Init(void)
69. {
70.     XPT_Init();//spi 管脚初始化
71.     TOUCH_Int();//触摸中断设置
72.     touch_flag=0; //中断标志位初始化
73.     ADS_Read_AD(CMD_RDX);
74.     ADS_Read_AD(CMD_RDY);
75. }
```

主函数其实很简单了，调用初始化后，直接判断什么时候发生了触摸，然后转换触摸发生的 X,Y 坐标位置，在该位置进行画点操作：

```
76. if(touch_flag==1)
77.         {
78.             touch_flag==0;
79.             if(Read_Continue()==0)
80.             {
81.                 if((Pen_Point.Y_Coord>100)&&(Pen_Point.Y_Coord<270))
82.                 {
83.                     if(Read_Continue()==0)
84.                     {
    if((Pen_Point.X_Coord>20)&&(Pen_Point.X_Coord<220)&&
```

Copyright© 2009-2015 青风电子社区 ALL rights reserved

```
85.     (Pen_Point.Y_Coord>100)&&(Pen_Point.Y_Coord<270))
86.                     LCD_Draw5Point(Pen_Point.X_Coord, Pen_Point.Y_Coord, DrawPenColor);
87.                                     }
88.                             }
```

Copyright© 2009-2015 青风电子社区 ALL rights reserved