

STemWin5.22 移植笔记

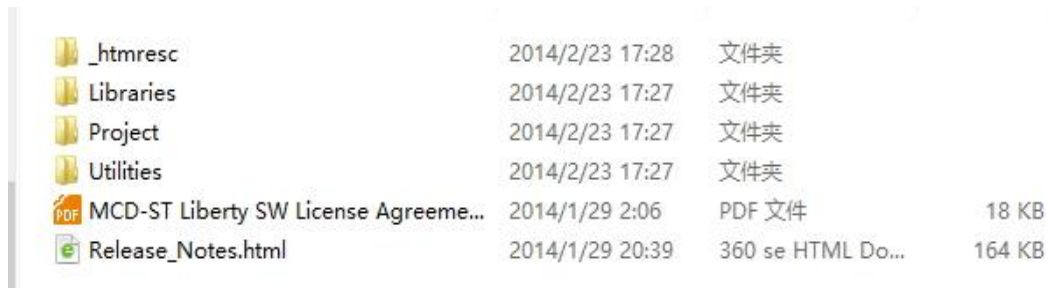
网上关于 emwin 的资料很少，我在移植的时候查了很多资料，对我一个感觉是好乱，有的代码改动的太多，这会让像我这种菜鸟无从下手，并且来源很乱，这让我决心写下这个笔记，来让新手快速入门 emwin，高手当然飘过哦，这只限于我这种菜鸟罢了。

emWin 是 segger 公司出的一款图形化界面，非常好看，大家所熟悉的 ucGUI 也是这个公司开发的，ucGUI 是 XP 的，而 emWin 是 win7 哦！而移植也要比 ucGUI 简单的多，没有 ucGUI 那么麻烦。好废话少说，如正题。

1、获取库文件

首先，emWin 库来源有两个，一个是 MDK(KEIL)软件目录下的，另外一个 ST 公司的 emWin，这里我们用第二个，因为 MDK 目录下的 emWin 最新版本对于大家不是好获得 (MDK5.0 才是 5.22，MDK4.7 下是 5.16 的)，而 ST 公司的那个好下载。
<http://www.st.com/web/en/catalog/tools/PF259225#> 这个是下载地址哦，目前最新的是 5.22 的版本，打开页面，点击下面的 Download，即可下载。

解压缩下载的压缩包，打开文件夹，我们看到



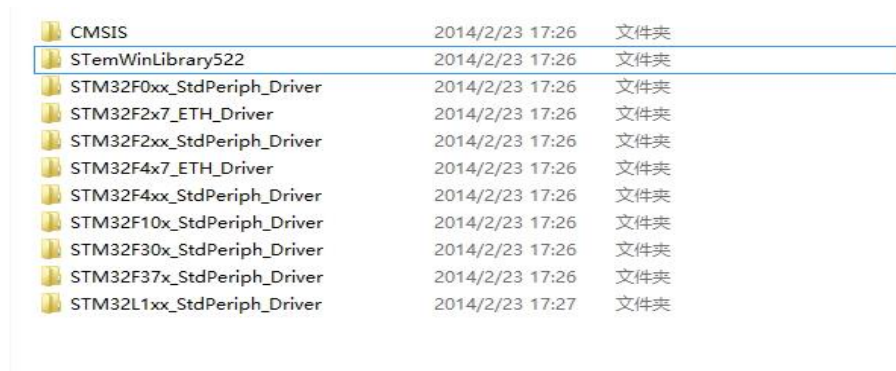
| | | | |
|---------------------------------------|-----------------|-------------------|--------|
| _htmresc | 2014/2/23 17:28 | 文件夹 | |
| Libraries | 2014/2/23 17:27 | 文件夹 | |
| Project | 2014/2/23 17:27 | 文件夹 | |
| Utilities | 2014/2/23 17:27 | 文件夹 | |
| MCD-ST Liberty SW License Agreeeme... | 2014/1/29 2:06 | PDF 文件 | 18 KB |
| Release_Notes.html | 2014/1/29 20:39 | 360 se HTML Do... | 164 KB |

Libraries：里面就有我们需要的 STemWin 库

Project：这个是 ST 的工程文件，以官方的 eval 板子建的工程

Utilities：这个是文件系统和 eval 板子的硬件驱动程序

我们打开 Libraries 文件夹，看到 StemWinLibrary522 了，先莫激动，这个我们先到这里。



| | | |
|----------------------------|-----------------|-----|
| CMSIS | 2014/2/23 17:26 | 文件夹 |
| STemWinLibrary522 | 2014/2/23 17:26 | 文件夹 |
| STM32F0xx_StdPeriph_Driver | 2014/2/23 17:26 | 文件夹 |
| STM32F2x7_ETH_Driver | 2014/2/23 17:26 | 文件夹 |
| STM32F2xx_StdPeriph_Driver | 2014/2/23 17:26 | 文件夹 |
| STM32F4x7_ETH_Driver | 2014/2/23 17:26 | 文件夹 |
| STM32F4xx_StdPeriph_Driver | 2014/2/23 17:26 | 文件夹 |
| STM32F10x_StdPeriph_Driver | 2014/2/23 17:26 | 文件夹 |
| STM32F30x_StdPeriph_Driver | 2014/2/23 17:26 | 文件夹 |
| STM32F37x_StdPeriph_Driver | 2014/2/23 17:26 | 文件夹 |
| STM32L1xx_StdPeriph_Driver | 2014/2/23 17:27 | 文件夹 |

2、库文件加到工程里

首先，我们借用原子的触摸程序(战舰的哦)，先将前面找到的库文件夹复制到工程文件夹下，**注意**：工程里的 lcd.c 和 lcd.h 改为 ili93xx.c 和 ili93xx.h 后重新加入到工程里，LCD_Init () 也改为 LCDx_Init ()

下面的程序由 ili93xx.h 中剪切到 ili93xx.c 中，至于为什么做以上工作，移植过 ucGUI 的再知道不过了，这里就不多说了，不知道的可以百度哦。

typedef struct

```

{
    u16 LCD_REG;
    u16 LCD_RAM;
} LCD_TypeDef;
#define LCD_BASE ((u32)(0x6C000000 | 0x000007FE))
#define LCD ((LCD_TypeDef *) LCD_BASE)

```

如图：

| | | |
|-------------------|-----------------|---------------------|
| CORE | 2014/2/26 21:05 | 文件夹 |
| HARDWARE | 2014/2/26 21:05 | 文件夹 |
| OBJ | 2014/2/26 21:05 | 文件夹 |
| STemWinLibrary522 | 2014/2/26 21:05 | 文件夹 |
| STM32F10x_FWLib | 2014/2/26 21:05 | 文件夹 |
| SYSTEM | 2014/2/26 21:05 | 文件夹 |
| USER | 2014/2/26 21:05 | 文件夹 |
| keilkill.bat | 2011/4/23 10:24 | Windows 批处理... 1 KB |

接下来就是拿出你的剪刀了，首先把主程序里的触摸都剪掉，剩下下面的就行啦！

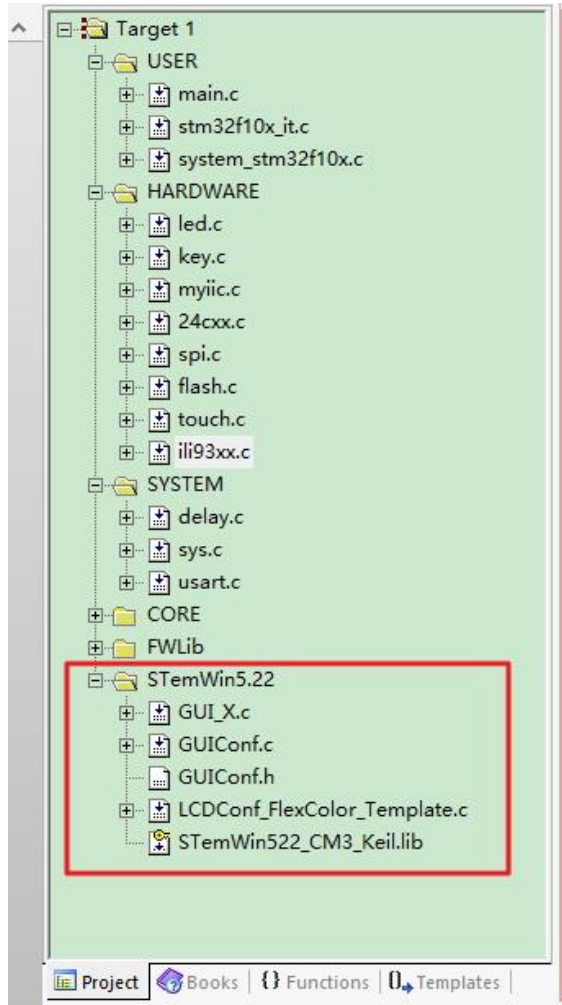
```

int main(void)
{
    u8 i=0;
    delay_init();
    NVIC_Configuration();
    uart_init(9600);
    LED_Init();
    LC。Dx_Init();
    KEY_Init();
    POINT_COLOR=RED;
    while(1)
    {

        i++;
        if(i==20)
        {
            i=0;
            LED0=!LED0;
        }delay_ms(20);
    }
}

```

接着就是把文件加到工程你去那，下图是工程截图：



红框中间的文件路径：STemWinLibrary522\Config

GUI_X.c 的路径：STemWinLibrary522\OS

接下来就是 emWin 的库啦，哈哈，打开 Lib 文件夹，看到那么多的文件，是不是晕了，我们仔细看看，ST 给 M0 M3 M4 的都分配了库，还很了编译软件和有系统的，简直是分的很好，显然我们用 STemWin522_CM3_Keil.lib 这个库，好了，工程就到此就弄好了。

3、修改程序，启动 emWin

接下来的工作就麻烦点喽，大家淡定点，其实也就一会儿的功夫喽！

首先动 GUIConf.c 中的 GUI_NUMBYTES 为 1024*50，50 可以小点，不要太大，太大编译器会编译会错误的，

接下来是 LCDConf_FlexColor_Template.c 这个文件，先将我们的 ili93xx.h 头文件包含进来，接下来加入以下两行程序，至于后面的地址为什么会这样，这是 LCD 显示里的哦

```
#define LCD_REG_ADDRESS      *(__IO uint16_t*)(0x6C000000)
#define LCD_DATA_ADDRESS    *(__IO uint16_t*)(0x6C000800)
```

再就是下面的几个程序

```
static void LcdWriteReg(U16 Data) {
    // ... TBD by user
    LCD_REG_ADDRESS=Data;
```

```

}
static void LcdWriteData(U16 Data) {
    // ... TBD by user
    LCD_DATA_ADDRESS=Data;
}

static void LcdWriteDataMultiple(U16 * pData, int NumItems) {
    while (NumItems--) {
        // ... TBD by user
        LCD_DATA_ADDRESS=*pData++;
    }
}

static void LcdReadDataMultiple(U16 * pData, int NumItems) {
    *pData = LCD_DATA_ADDRESS;
    while (NumItems--) {
        // ... TBD by user
        *pData++=LCD_DATA_ADDRESS;
    }
}

```

此时重要的函数来喽 void LCD_X_Config(void)

```
Config.Orientation = GUI_SWAP_XY | GUI_MIRROR_Y;
```

这句是显示方向的，默认的是这个，弄好后，下进去板子，看是否正常显示，不正常显示可以试试该这里哦。PS:9341 的能行，但是 6804 的不行，看哪位能解决哦！

```
GUIDRV_FlexColor_SetFunc(pDevice,          &PortAPI,          GUIDRV_FLEXCOLOR_F66709,
GUIDRV_FLEXCOLOR_M16C0B16);
```

GUIDRV_FLEXCOLOR_F66709 这个东西很重要，移植的时候我就栽在这里啦，这是对控制芯片的支持，具体请看最新手册 5.22 的（P991），也有中文的 5.12，不过 5.22 支持的很多。

最后就是 main 函数喽

```
RCC_AHBPeriphClockCmd(RCC_AHBPeriph_CRC, ENABLE);
```

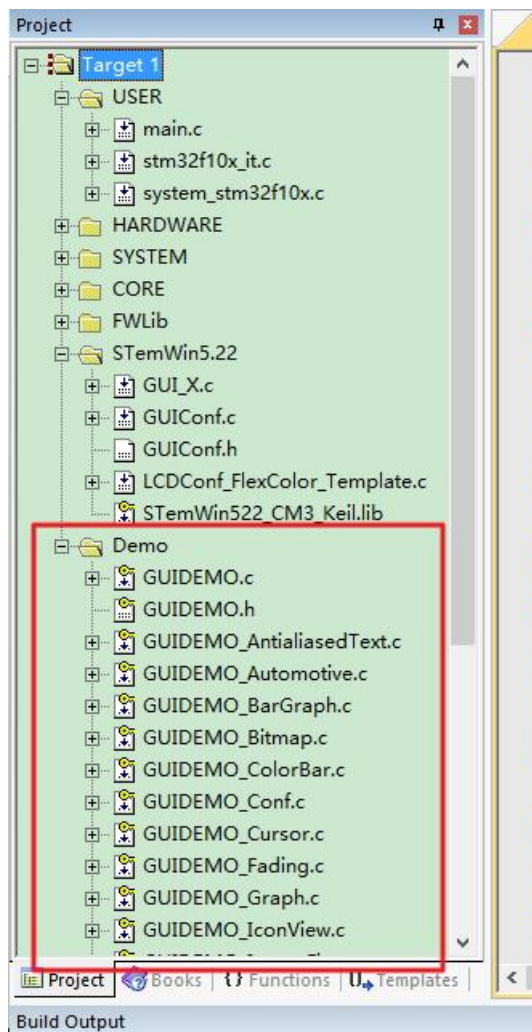
```
GUI_Init();
```

```
GUI_DispString("Hello STemWin!");
```

记住这句一定要加上 RCC_AHBPeriphClockCmd(RCC_AHBPeriph_CRC, ENABLE); 不知道是因为什么，大家加上就行喽，哈哈，现在编译把程序下到板子里去哦，哈哈 是不是成功显示了。来张照片。哈哈，先歇会儿，后续再写上 demo 的和触屏的。

4、加入 demo，让 emWin 炫起来

首先，工程文件加入 demo 的文件，如下图



加入头文件 `gui_demo.h` 然后再主函数里加入 `GUIDEMO_Main();` 好，编译程序，下进去，惊喜吗？看到经典的开始画面了，但是没有动，右下脚没有像别人的视频里的滚动啊！哈哈，莫急，因为我们还没有启动 `emWin` 的心跳哦！

有的是用 `systick`，但是我没有实验成功，而是用定时器用的，其实都是一样，在 `HARDWARE` 里加入 `gui_time.c` 和 `gui_time.h` 文件，具体源码见工程喽，其实就是循环调用 `OS_TimeMS++`；让 `emWin` 跳动起来。别忘了，主函数要调用定时器的初始化哦！我们再编译，程序跑起，哈哈，是不是动了。

欸，但是还有个问题，怎么就是一个画面，不往下去了呢？这是我们还没有开启 `demos` 的开关了，打开 `GUIDEMO.h` 文件，配置如下图，记住后面每个项目的开关要关掉很多的，不然会有错误的。我开了 8 个，到黑圈那里就好了。

```

97  ****
98  */
99  #if 1 // Show all demos
100 #ifndef SHOW_GUIDEMO_AATEXT
101     #define SHOW_GUIDEMO_AATEXT (1)
102 #endif
103 #ifndef SHOW_GUIDEMO_AUTOMOTIVE
104     #define SHOW_GUIDEMO_AUTOMOTIVE (1)
105 #endif
106 #ifndef SHOW_GUIDEMO_BARGRAPH
107     #define SHOW_GUIDEMO_BARGRAPH (1)
108 #endif
109 #ifndef SHOW_GUIDEMO_BITMAP
110     #define SHOW_GUIDEMO_BITMAP (1)
111 #endif
112 #ifndef SHOW_GUIDEMO_COLORBAR
113     #define SHOW_GUIDEMO_COLORBAR (1)
114 #endif
115 #ifndef SHOW_GUIDEMO_CURSOR
116     #define SHOW_GUIDEMO_CURSOR (1)
117 #endif
118 #ifndef SHOW_GUIDEMO_FADING
119     #define SHOW_GUIDEMO_FADING (1)
120 #endif
121 #ifndef SHOW_GUIDEMO_GRAPH
122     #define SHOW_GUIDEMO_GRAPH (1)
123 #endif
124 #ifndef SHOW_GUIDEMO_ICONVIEW
125     #define SHOW_GUIDEMO_ICONVIEW (0)
126 #endif

```

再跑起程序，哈哈，这回就真的动了。高兴啊，见到 win7 风格的画面，我为以后的界面充满了信息。

5、启动 touch，人机界面更美好

这是我们移植的最后一步啦，我们先看中文手册上是怎么说的（P765），见下图

22.4.2.1 设置模拟触摸屏

触控面板的准备应遵循以下步骤：

- 应用硬件程序
- 对 GUI_TOUCH_Exec() 应用定期调用
- 用示波器验证工作是否正常
- 通过示例来确定校准值
- 使用上一步确定的值，在初始化程序 LCD_X_Config() 中添加一个对 GUI_TOUCH_Calibrate() 的调用

下面对每一步进行详细描述。

那我们就按照这个步骤来喽。首先是硬件程序，由于我们用的 5.22 版本没有关于 touch 的 C 文件，我便从 5.16 那儿拷过来了 GUI_X_Touch_Analog.c，把这个文件加入到 StemWin5.22 组里，因为直接就用的原子的触摸了，我们就直接调用 TP_Read_XY2(&x,&y);这个函数，把他放进 GUI_TOUCH_X_MeasureX(void)和 GUI_TOUCH_X_MeasureY(void)这两个函数里，其实最后由这个函数 GUI_TOUCH_Exec()循环调用，来一直读 AD 的值。哦，说到这儿，把 GUI_TOUCH_Exec()这个函数也要放进前一步建的 gui_time.c 文件里，具体见源码喽!这就把前两步弄完了。

第三步没搞过，直接第四步吧。用示例确定校准值？？迷惑，哪里的示例，怎么确定，哈哈，听我一步步说来。这里的值就是取得 A/D 转换器的最小值和最大值。emWin 需要用这些值来把测量结果转换为以像素表示的触摸位。这 4 个值为：

| 值 | 获取方法 |
|---------------------|-----------------------|
| GUI_TOUCH_AD_TOP | 按下触摸屏的顶部，写下 Y 轴模拟输入值。 |
| GUI_TOUCH_AD_BOTTOM | 按下触摸屏的底部，写下 Y 轴模拟输入值。 |
| GUI_TOUCH_AD_LEFT | 按下触摸屏的左侧，写下 X 轴模拟输入值。 |
| GUI_TOUCH_AD_RIGHT | 按下触摸屏的右侧，写下 X 轴模拟输入值。 |

emWin 的示例文件夹中有一个小程序，可用来获取触控面板的这些值。该程序位于文件夹 Sample\Tutorial 下，程序名称为 TOUCH_Sample.c。在硬件上运行该示例。其输出应类似于下侧的屏幕截图。

```
Measurement of
A/D converter values
Analog input:
x:0423, y:0386
Position:
x:0093, y:0043
```

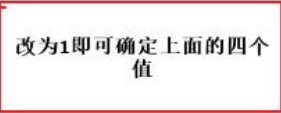


接下来我们在 LCDConf_FlexColor_Template.c 文件中加入上面提到的四个值

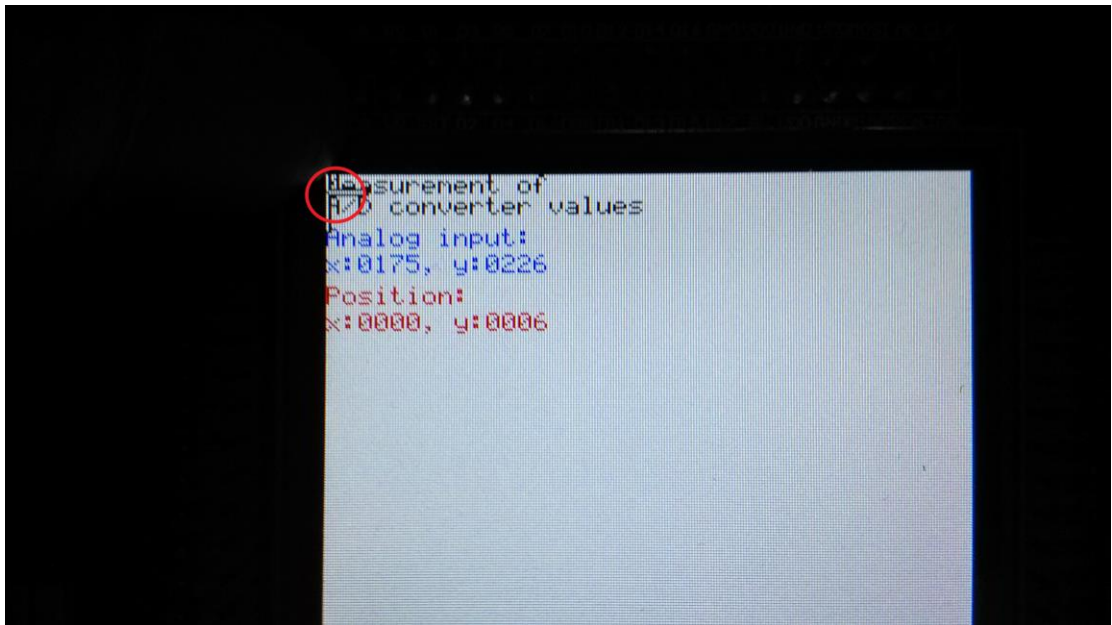
```
#define TOUCH_AD_TOP 221
#define TOUCH_AD_BOTTOM 3900
#define TOUCH_AD_LEFT 160
#define TOUCH_AD_RIGHT 3883
```

后面的数字只是临时写的，然后我们再新建 gui_touch.c 文件，这就是 TOUCH_Sample.c 文件的程序，我们拷贝过来到 gui_touch.c 中，如下图所示

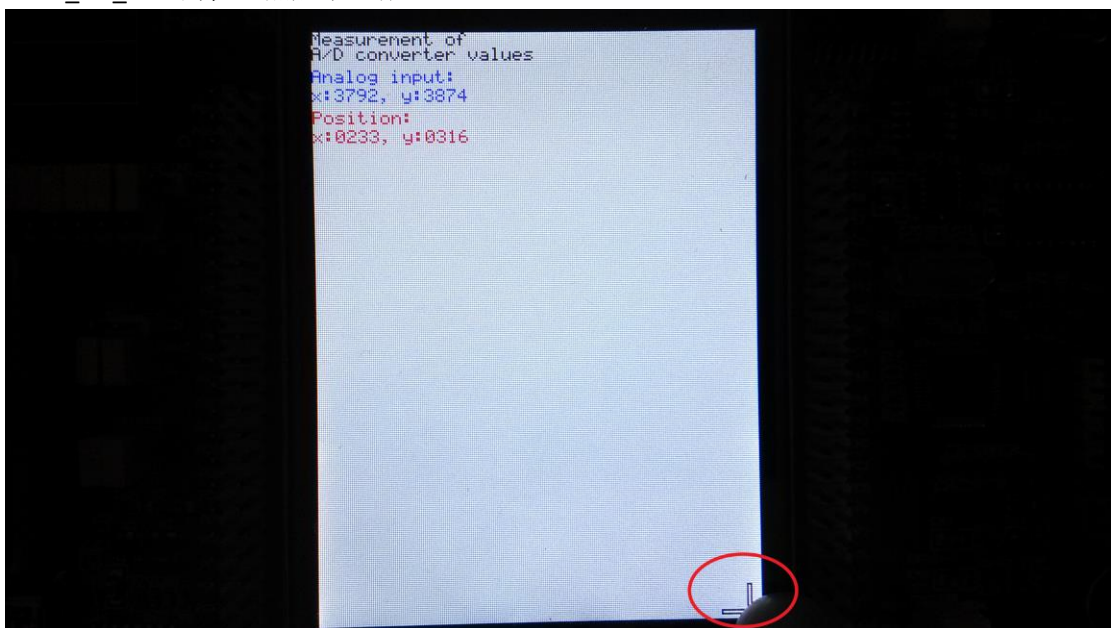
```
14 int main(void)
15 {
16     u8 i=0;
17     delay_init(); //延时函数初始化
18     NVIC_Configuration(); //设置NVIC中断分组2:2位抢占优先级，2位响应优先级
19     uart_init(9600); //串口初始化为9600
20     LED_Init(); //LED端口初始化
21     LCDx_Init();
22     KEY_Init();
23     TP_Init();
24     POINT_COLOR=RED; //设置字体为红色
25     TIM3_Init(2,36000-1);
26
27     RCC_AHBPeriphClockCmd(RCC_AHBPeriph_CRC, ENABLE); //不知道为什么一定要开crc时钟
28
29     GUI_Init(); //初始化GUI
30
31     GUI_DisString("Hello STemWin!"); //显示测试
32     #if 0
33     Touch_MainTask();
34     #else
35     GUIDEMO_Main();
36     #endif
37     while(1)
38     {
```



然后下载程序，界面如下图



这是确定左上角的值，记住，一定要触摸，蓝色的 X 就是 TOUCH_AD_LEFT 的值，Y 就是 TOUCH_AD_TOP 的值，然后下一张



同样触摸右下角，看到十字会移动到右下角，这是蓝色 X 的值为 TOUCH_AD_RIGHT，Y 值为 TOUCH_AD_BOTTOM，这样，前面提到的四个值就已经确定了，同时改掉 LCDConf_FlexColor_Template.c 里的那四个值。

最后一步啦，哈哈，见下图，在 LCD_X_Config 函数里调用 GUI_TOUCH_Calibrate 函数，这样，我们的触屏移植就弄完了。


```

180 void LCD_X_Config(void) {
181     GUI_DEVICE * pDevice;
182     CONFIG_FLEXCOLOR Config = {0};
183     GUI_PORT_API PortAPI = {0};
184     //
185     // Set display driver and color conversion
186     //
187     pDevice = GUI_DEVICE_CreateAndLink(GUIDRV_FLEXCOLOR, GUICC_565, 0, 0);
188     //
189     // Display driver configuration, required for Lin-driver
190     //
191     LCD_SetSizeEx (0, XSIZE_PHYS , YSIZE_PHYS);
192     LCD_SetVSizeEx(0, VXSIZE_PHYS, VYSIZE_PHYS);
193
194     GUI_TOUCH_Calibrate(GUI_COORD_X, 0, 240, TOUCH_AD_TOP, TOUCH_AD_BOTTOM);
195     GUI_TOUCH_Calibrate(GUI_COORD_Y, 0, 320, TOUCH_AD_LEFT, TOUCH_AD_RIGHT);
196     //

```

弄了好久了，寒假就开始了，到现在触屏终于可以啦，哈哈，发这个帖子也祝贺自己一下喽，哈哈，大神们指点意见哦！

作者：pizhihui

日期：2014 年 2 月 27 日

联系 QQ：912458257