emWin的移植说明：

解压缩后，使用工程

\stemwinban\STemWin\_Library\_V1.1.2\Project\STM3210C-EVAL\Standalone\EWARM、\STemWinDemo.eww

# 1.GUI介绍

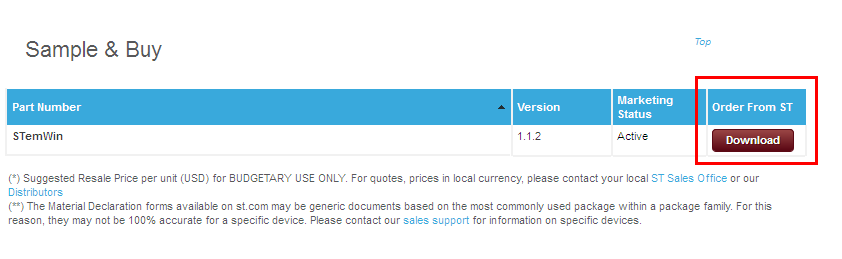
Uc/GUI->

emWin->

STemWin

低版本的叫**UCGUI**，高版本叫**emWin**

下载地址：<http://www.st.com/web/en/catalog/tools/PF259225>



# 2.STemWin流程说明

## 1）初始化

LCD和Touch均要初始化配置，包括控制端口和数据端口，这部分工作在驱动的C文件中。

其中LCD初始化函数为：lcd\_config。LCD的初始化工作在LCDConf.c文件中的LCD\_X\_DisplayDriver()函数中完成。

## 2）运行时调用时间函数

在GUI\_X.c中，需要在主程序中产生一个对OS\_TimeMS变量操作的程序，使该变量每隔1ms加1.

在timer5.c中，

Tim初始化修改：

TIM\_TimeBaseStructure.TIM\_Period = (10 - 1);

TIM\_TimeBaseStructure.TIM\_Prescaler = (7200 - 1);

添加代码：

extern int OS\_TimeMS;

extern void GUI\_TOUCH\_Exec(void);

void TIM5\_IRQHandler(void)

{

//if (TIM\_GetITStatus(TIM5, TIM\_IT\_Update) != RESET)

{

TIM\_ClearITPendingBit(TIM5, TIM\_IT\_Update);

/\* 这里添加自己的代码\*/

OS\_TimeMS++; //1.1ms更新GUI时间OS\_TimeMS

if(OS\_TimeMS % 10 == 0)

{

GUI\_TOUCH\_Exec(); //2.10ms调用GUI\_TOUCH\_Exec

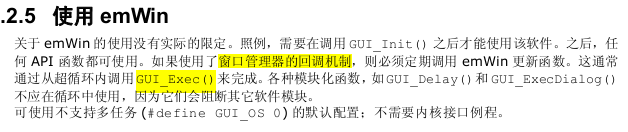
}

}

}

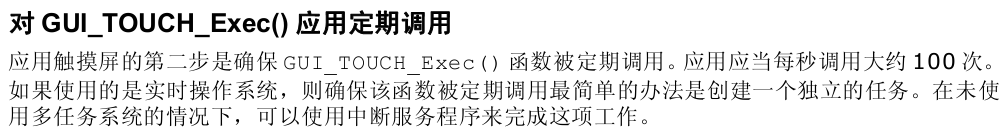
## 3）运行时调用 的函数

（1）GUI\_Exec() 在GUI初始化后，在GUI\_Delay()中自动运行。



(2) GUI\_TOUCH\_Exec()

在timer.c的定时中断中调用，每隔10ms



# 3.文件代码说明及修改实现

与emWIn的GUI有关的配置文件在emWin\_Config文件夹下。



与LCD和触摸屏有关的驱动文件在drivers文件夹下。



其中touch.c 为不使用SPI总线（采用模拟SPI）的驱动，touch\_spi.c为使用SPI总线的驱动。

与demo运行有关的文件在emWin\_Demo文件夹下。

与主程序中运行计算器有关的文件在 src文件夹下：

运行emWin还需要一个库文件（IAR和MDK），在Libraries\STemWinLibrary522\Lib文件夹下：



运行时调用的时间函数GUI\_X.c,在\Libraries\STemWinLibrary522\OS文件夹下：



运行时主程序，在src文件夹下：



# 4.液晶模块的驱动配置

液晶模块的驱动很多，常用ili9325 SSD1289,不同的驱动模块初始化代码不同，运行时的操作也不同，其模块的驱动可通过读取寄存器0x0000获得。

例程针对以上不同模块作了兼容处理，首先读取ID

DeviceIdCode = lcd\_read\_reg(0x0000);

再根据不同的ID，运行相应的程序。

if(DeviceIdCode==0x8989)//SSD1289

{…

}

else if(DeviceIdCode==0x9325)// ili9325

{…

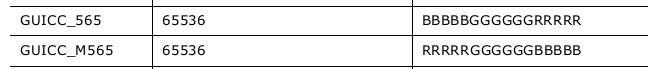
}

P P

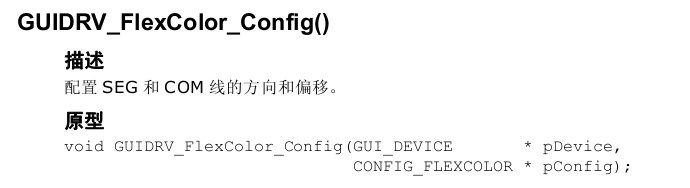
P862



P229



P838



配置在LCDConf.c 中LCD\_X\_Config函数中进行：

/\* Set display driver and color conversion\*/

if ( getLcdIdCode()==0x8989)

{

pDevice = GUI\_DEVICE\_CreateAndLink(GUIDRV\_FLEXCOLOR, GUICC\_M565, 0, 0);

}

else if ( getLcdIdCode()==0x9325)

{

pDevice = GUI\_DEVICE\_CreateAndLink(GUIDRV\_FLEXCOLOR, GUICC\_565, 0, 0);

}

/\* Display driver configuration, required for Lin-driver\*/

LCD\_SetSizeEx (0, XSIZE\_PHYS , YSIZE\_PHYS);

LCD\_SetVSizeEx(0, VXSIZE\_PHYS, VYSIZE\_PHYS);

/\* Touch driver configuration\*/

if ( getLcdIdCode()==0x8989)

{

GUI\_TOUCH\_Calibrate(GUI\_COORD\_X, 0, 240, TOUCH\_AD\_TOP, TOUCH\_AD\_BOTTOM);

GUI\_TOUCH\_Calibrate(GUI\_COORD\_Y, 0, 320, TOUCH\_AD\_LEFT, TOUCH\_AD\_RIGHT);

}

else if ( getLcdIdCode()==0x9325)

{

GUI\_TOUCH\_Calibrate(GUI\_COORD\_X, 0, 240, TOUCH\_AD\_BOTTOM, TOUCH\_AD\_TOP);

GUI\_TOUCH\_Calibrate(GUI\_COORD\_Y, 0, 320, TOUCH\_AD\_RIGHT, TOUCH\_AD\_LEFT);

}

/\* Display driver Orientation configuration\*/

if ( getLcdIdCode()==0x8989)

{

Config.Orientation = GUI\_SWAP\_XY;

GUIDRV\_FlexColor\_Config(pDevice, &Config);

}

else if ( getLcdIdCode()==0x9325)

{

Config.Orientation = GUI\_SWAP\_XY|GUI\_MIRROR\_Y;

GUIDRV\_FlexColor\_Config(pDevice, &Config);

}

/\* Set controller and operation mode\*/

PortAPI.pfWrite16\_A0 = LcdWriteReg;

PortAPI.pfWrite16\_A1 = LcdWriteData;

PortAPI.pfWriteM16\_A1 = LcdWriteDataMultiple;

PortAPI.pfReadM16\_A1 = LcdReadDataMultiple;

if ( getLcdIdCode()==0x8989)

{

GUIDRV\_FlexColor\_SetFunc(pDevice, &PortAPI, GUIDRV\_FLEXCOLOR\_F66702, GUIDRV\_FLEXCOLOR\_M16C0B16);

}

else if ( getLcdIdCode()==0x9325)

{

GUIDRV\_FlexColor\_SetFunc(pDevice, &PortAPI, GUIDRV\_FLEXCOLOR\_F66708, GUIDRV\_FLEXCOLOR\_M16C0B16);

}

}

# 5. GUIDEMO\_Main循环操作

\_Main();

//GUI清屏:

GUI\_Clear();

//显示鼠标:

#if (GUI\_SUPPORT\_CURSOR | GUI\_SUPPORT\_TOUCH)

GUI\_CURSOR\_Show();

#endif

// 建立配置控制和信息窗口

xSize = LCD\_GetXSize();

ySize = LCD\_GetYSize();

\_hDialogControl = GUI\_CreateDialogBox(\_aFrameWinControl, GUI\_COUNTOF(\_aFrameWinControl), &\_cbFrameWinControl, WM\_HBKWIN, xSize - CONTROL\_SIZE\_X, ySize - CONTROL\_SIZE\_Y);

\_hDialogInfo = GUI\_CreateDialogBox(\_aFrameWinInfo, GUI\_COUNTOF(\_aFrameWinInfo), &\_cbFrameWinInfo, WM\_HBKWIN, (xSize >> 1) - 1, 0);

WM\_HideWindow(\_hDialogInfo);

// 显示 Intro

//

WM\_InvalidateWindow(\_hDialogControl);

WM\_DisableMemdev(WM\_HBKWIN);

GUI\_Exec();

WM\_EnableMemdev(WM\_HBKWIN);

GUIDEMO\_Intro();

//Demo一个个地显示

for (\_iDemo = 0; \_GUIDemoConfig.apFunc[\_iDemo]; \_iDemo++) {

\_ClearHalt();

GUIDEMO\_UpdateControlText();

(\*\_GUIDemoConfig.apFunc[\_iDemo])();

\_iDemoMinor = 0;

\_Pressed = 0;

}

//

// 清屏

//

WM\_DeleteWindow(\_hDialogControl);

WM\_DeleteWindow(\_hDialogInfo);

#if (GUI\_SUPPORT\_CURSOR | GUI\_SUPPORT\_TOUCH)

GUI\_CURSOR\_Hide();

#endif

# 6.练习

1）修改例程，添加AD转换数据显示，将以下语句

/\* Start Demo \*/

//GUIDEMO\_Main();

换成

GUI\_DispChars('/', 53);

for(int i = 0; i < 240; i += 8)

{

GUI\_DispCharAt('/', 312, i);

GUI\_DispCharAt('/', 1, i);

}

GUI\_DispChars('/', 52);

GUI\_SetColor(GUI\_RED);

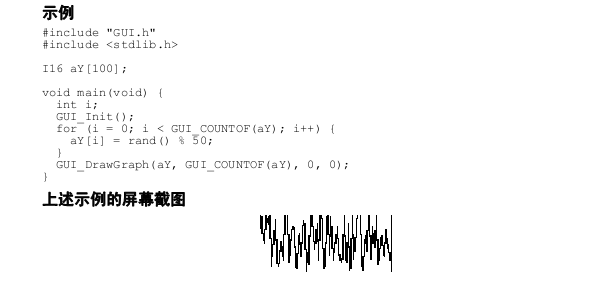
GUI\_SetFont(&GUI\_Font24B\_ASCII);

GUI\_DispStringAt("Data:",10,110);

将AD转换的数据显示在后面。

2）将AD后的数据用图画出来。参考《emWin手册》P137





答案：

在main.c中添加：

/\*每一帧线图点的个数\*/

#define DATA\_NUM 10

/\*简单延时函数\*/

static void Delay(unsigned int t)

{

while(t--) ;

}

/\*显示线图变量定义\*/

uint16\_t adcvalue;

uint16\_t Precent = 0;

uint32\_t Voltage = 0;

char str[50] = {0x00};

uint16\_t value[DATA\_NUM] = {0};

ADC\_Config();

/\* AD转换线图显示 \*/

/\* 显示外框 \*/

GUI\_DispChars('/', 53);

for(int i = 0; i < 240; i += 8)

{

GUI\_DispCharAt('/', 312, i);

GUI\_DispCharAt('/', 1, i);

}

GUI\_DispChars('/', 52);

GUI\_SetColor(GUI\_RED);

GUI\_SetFont(&GUI\_Font24B\_ASCII);

int i = 10;

while(1)

{

/\* 读取ADC的值 \*/

for(int j = 1; j < DATA\_NUM; j++ )

{

getadcvalue(&adcvalue);

value[j] = (adcvalue % 100);

}

/\* 显示线图（DATA\_NUM个点） \*/

GUI\_DrawGraph(value,DATA\_NUM,i,70);

/\* DATA\_NUM个点组成一个数组，要把上一个数组的最后一个赋值下一个数组第一个 \*/

i += (DATA\_NUM-1);

value[0] = (adcvalue % 100);

if(i > 300)

{

/\* 一帧图画完后，要擦除 \*/

GUI\_SetColor(GUI\_BLACK);

GUI\_FillRect(10,10,310,230);

GUI\_SetColor(GUI\_RED);

i = 10;

}

/\*显示测量值\*/

Precent = (adcvalue\*100/0x1000);

Voltage = adcvalue \* 3300/0x1000;

sprintf(str,"\r\nAD:0x%X , Per:%d% , V:%d.%dV \n\r",

adcvalue, Precent, (uint32\_t)Voltage/1000, Voltage%1000);

GUI\_DispStringAt(str,0,150);

/\*延时\*/

Delay(1000000);

}

注意：

1）以上添加代码不是简单的复制、粘贴，请思考后添加在相应的位置。

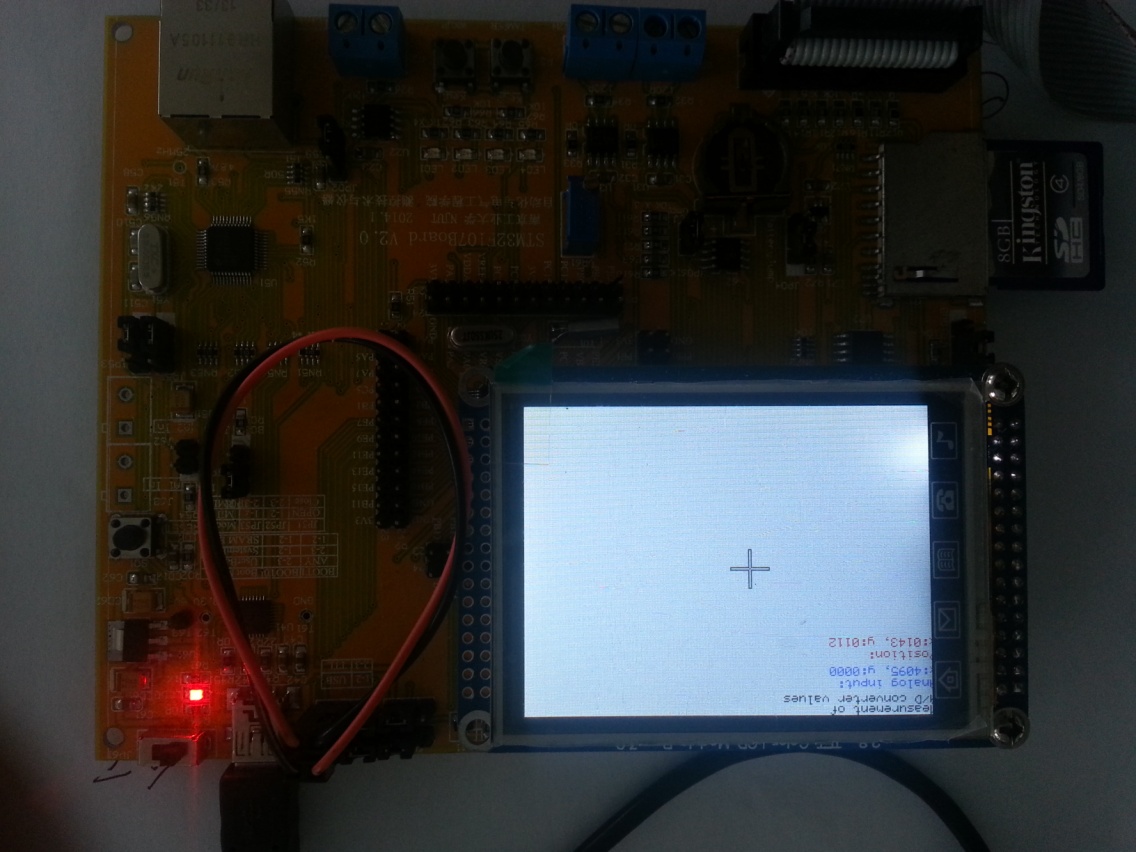
2）拷备和添加adc.c adc.h进入相应文件夹和工程中。

# 7.标定、运行demo、简易计算器

## 6.1 触摸屏标定

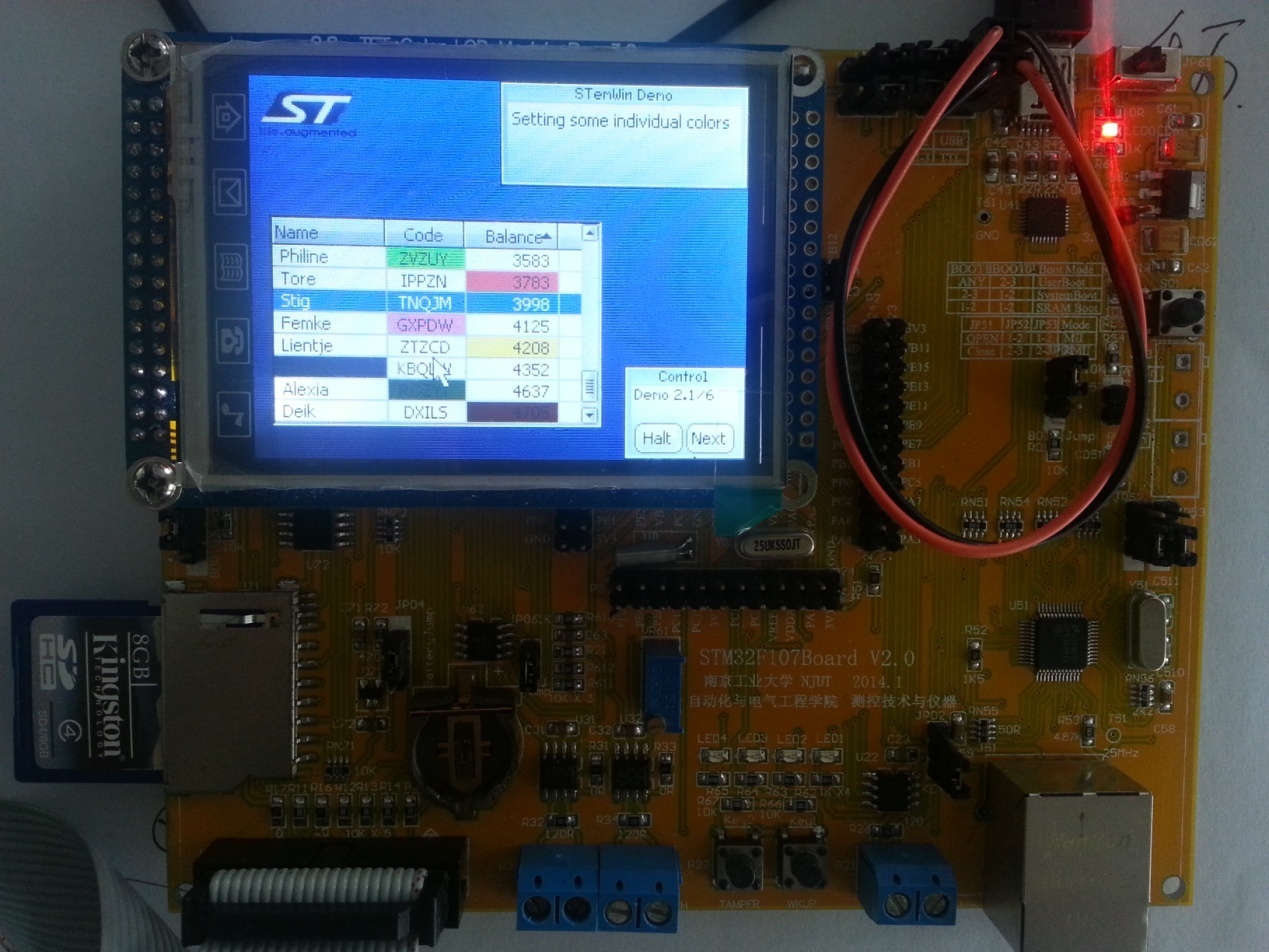
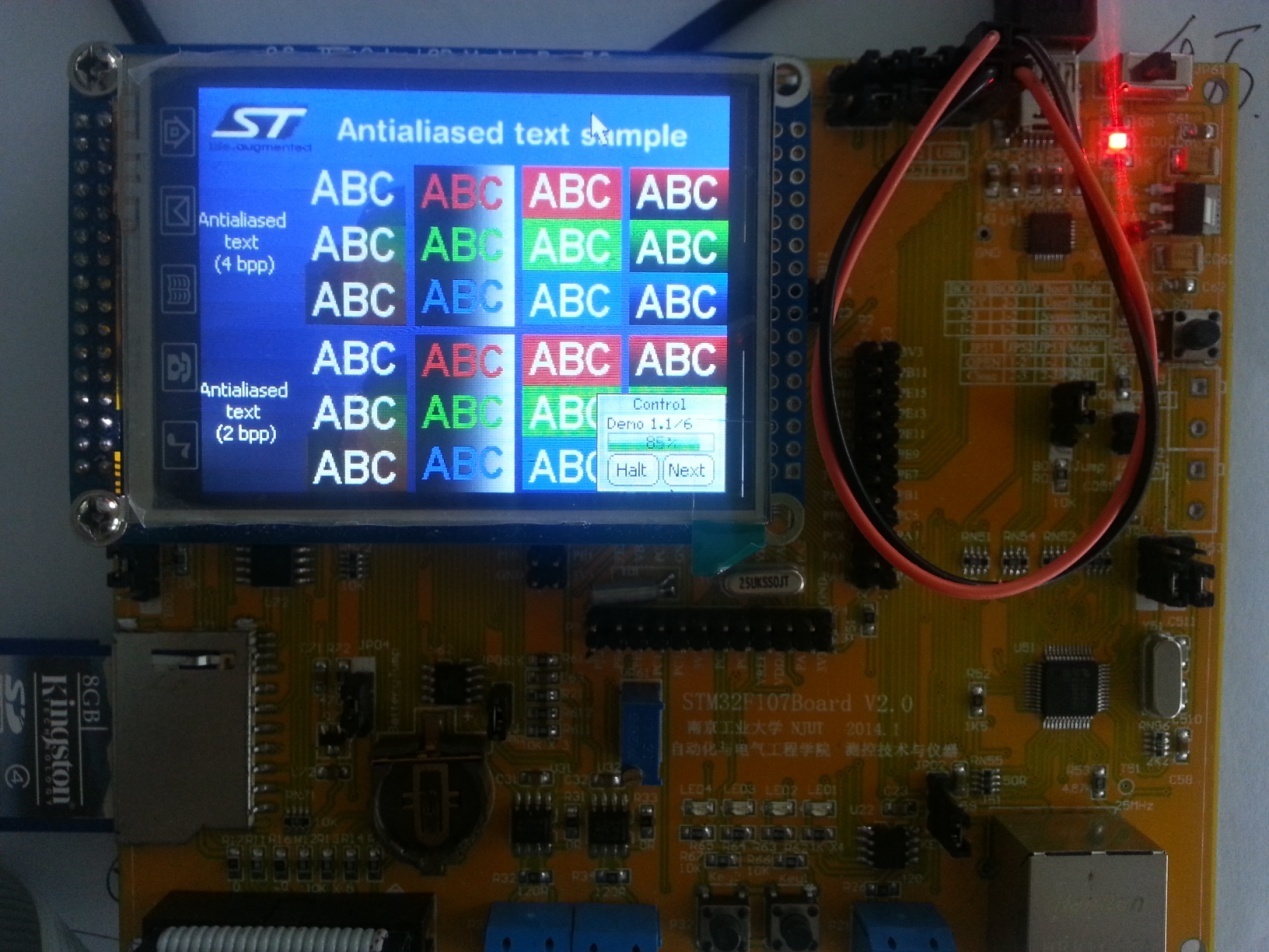
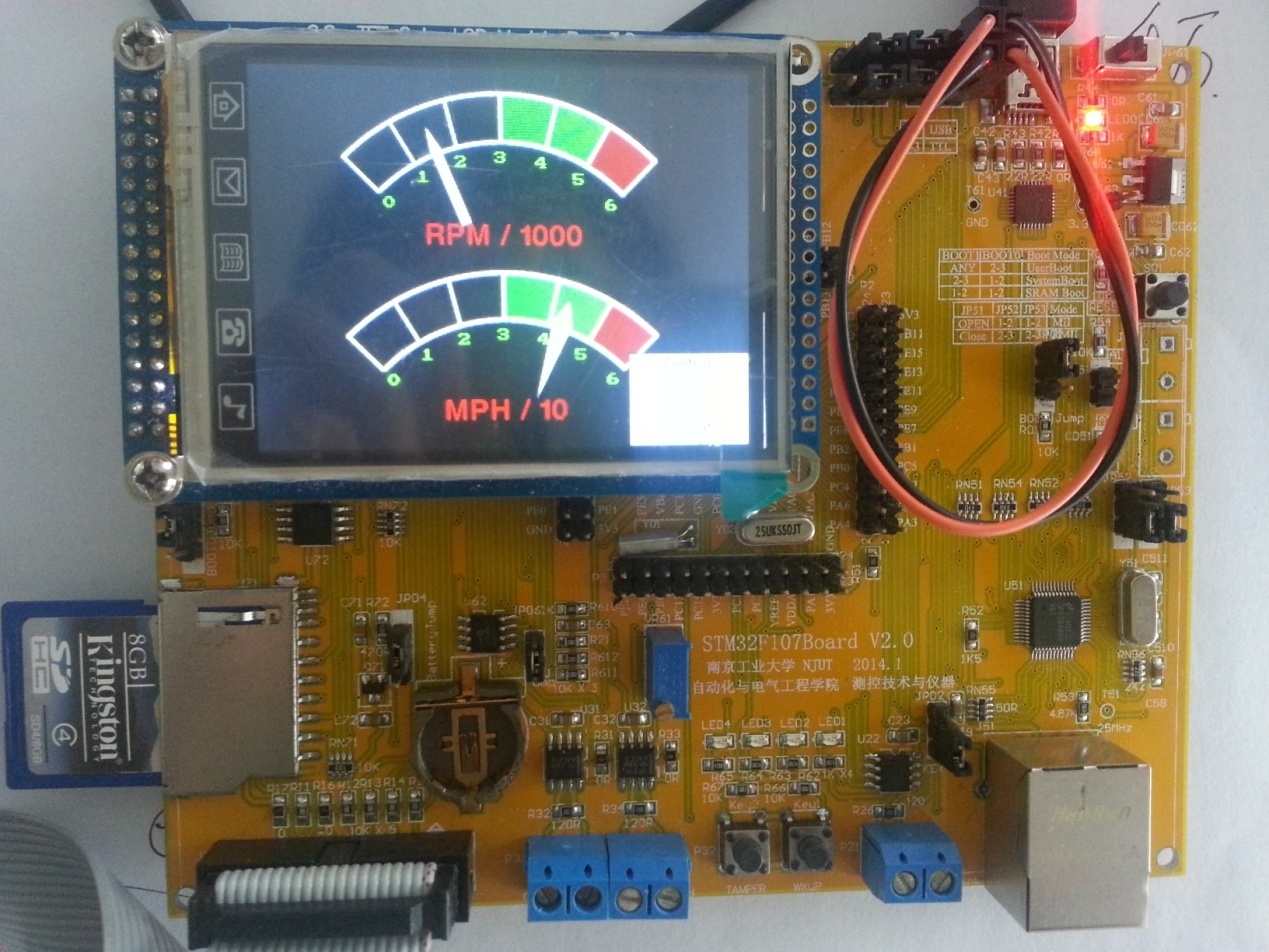
extern int touch\_calibrate();

touch\_calibrate();



## 6.2运行Demo

GUIDEMO\_Main();



## 6.3简易计算器（苗成诗编写）

extern int calculator\_main();

calculator\_main();

