



1.44inch LCD HAT

用户手册

产品概述

本产品是 1.44 寸电阻屏模块，分辨率为 128x128，带有内部控制器，使用 SPI 接口通信，与树莓派 Zero 大小一致，已封装好基本函数，可以选择点的大小，显得粗细虚实，是否填充圆、矩形，以及显示英文字符。

产品特性

类型：TFT
 接口：SPI
 控制芯片：ST7735S
 色阶指数：256K
 分辨率：128*128 (Pixel)
 产品尺寸：65*30.2 (mm)
 显示尺寸：25.5*26.5 (mm)
 像素大小：0.129 (W) *0.219 (H) (MM)
 工作温度：-30°C ~ +85°C

接口说明

功能引脚	树莓派接口 (BCM)	描述
KEY1	P21	按键 1GPIO
KEY2	P20	按键 2GPIO
KEY3	P16	按键 3GPIO
摇杆 Up	P6	摇杆上
摇杆 Down	P19	摇杆下
摇杆 Left	P5	摇杆左
摇杆 Right	P26	摇杆右
摇杆 Press	P13	摇杆按下
SCLK	P11/SCLK	SPI 时钟线
MOSI	P10/MOSI	SPI 数据线
DC	P25	数据/命令选择
CS	P8/CE0	片选
RST	P27	复位
BL	P24	背光

工作原理

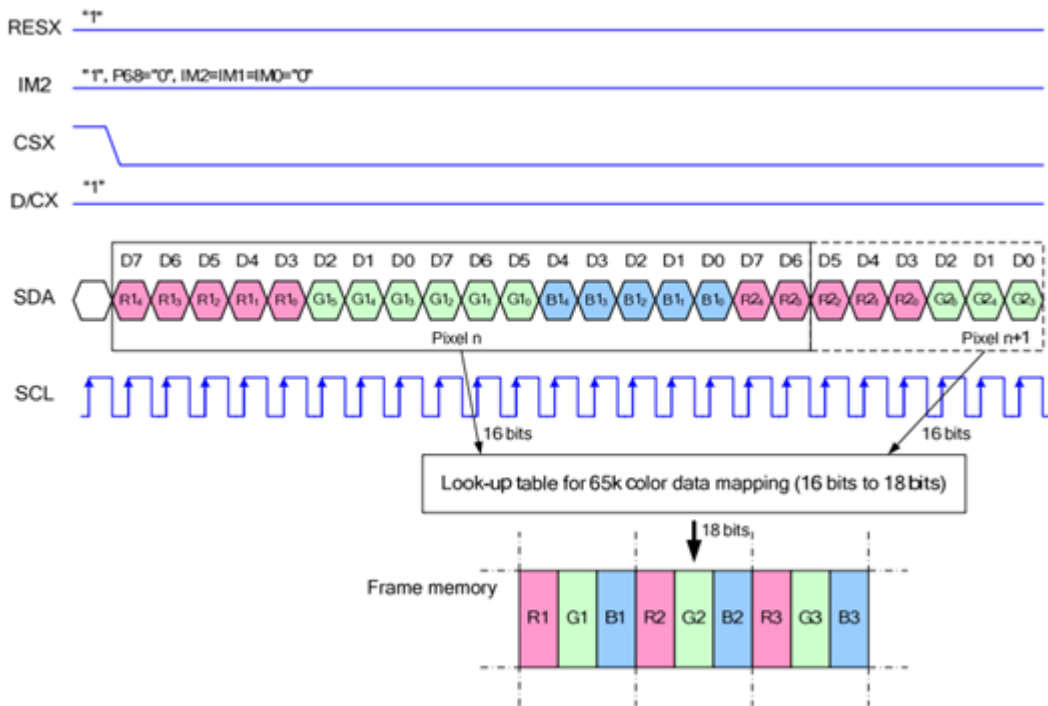
1. 工作原理

ST7735S 是一款 132*162 像素的 LCD，而本产品为 128*128 像素的 LCD，因此在显示上做了一些处理：水平方向从第二个像素点开始显示，这样就可以保证显示的时候 LCD 中 RAM 对应的位置与实际位置是一致的。

该 LCD 支持 12 位，16 位以及 18 位每像素的输入颜色格式，即 RGB444，RGB565，RGB666 三种颜色格式，本例程使用 RGB565 颜色格式。这也是比较常用的 RGB 格斯。

LCD 使用四线 SPI 通信接口，这样可以大大的节省 GPIO 口，同时通信速度也会比较快。

2. 通信协议



注：与传统的 SPI 协议不同的地方是：由于是只需要显示，故而将从机发往主机的数据线进行了隐藏，该表格详见 Datasheet Page 58.

RESX：复位，模块上电时拉低，通常情况下置 1；

IM2：控制模块数据通信方式，此处使用 SPI 通信

CSX：从机片选，仅当 CS 为低电平时，芯片才会被使能

D/CX：芯片的数据/命令控制引脚，当 DC=0 时写命令，当 DC=1 时写数据

SDA：传输的数据，即 RGB 数据

SCL：SPI 通信时钟

对于 SPI 通信而言，数据是有传输时序的，即时钟相位 (CPHA) 与时钟极性 (CPOL) 的组合：

CPOL 的高低决定串行同步时钟的空闲状态电平，CPOL=0，为低电平。CPOL 对传输协议的影响不大。

CPHA 的高低决定串行同步时钟是在第一时钟跳变沿还是第二个时钟跳变沿数据被采集，当 CPHL=0，在第一个跳变沿进行数据采集

这两者组合就成为四种 SPI 通信方式，国内通常使用 SPI0，即 CPHL=0，CPOL=0

从图中可以看出，当 SCLK 第一个下降沿时开始传输数据，一个时钟周期传输 8bit 数据，使用 SPI0，按位传输，高位在前，低位在后。

实验演示

本模块提供 BCM2835, WiringPi 以及 python 例程，并实现常用屏幕操作功能：画点，画线，画矩形，画圆及他们对应的大小，宽度，填充，显示应为字符。并提供了 5 种常见字体，以及画图等功能。

为了方便使用，这里对三种例程的使用镜像了详细的教程

1. 开启树莓派 SPI 功能

```
sudo raspi-config
```

选择 Advanced Options -> SPI -> yes

启动 SPI 内核驱动

2. 库的安装

关于树莓派库的安装详细见微雪课堂：

<http://www.waveshare.net/study/article-742-1.html>

此处详细介绍了 wiringPi、bcm2835、python 的安装

3. 使用

对于 BCM2835 与 WiringPi 而言只需把对应的例程拷贝进树莓派中国（可通过 samba 或者直接复制到 SD）即可，一下示例均复制到了树莓派 pi 用户目录下。

3.1 BCM2835 使用

运行 ls 命令，可见如下文件

```
pi@raspberrypi:~/bcm2835 $ ls
bin  Fonts  Makefile  obj  pic  tftlcd_lin44
```

其中 bin 文件夹中为项目生成的.o 文件，通常情况下我们是不需要管的

Fonts 文件夹中为 5 种常见字体。

Pic 文件夹中为显示的图片，请保证图片的像素为 128*128，否则可能显示不全。并且需要保证图片为.bmp 格式的。

Obj 文件夹中为项目文件，其中有 main.c, OLED_Driver.c 及.h, DEV_Config.c 及.h, 以及 OLED_GUI.c 及.h

main.c：主函数。需要注意的是，虽然定义了 OLED_ScanDir，这个是控制扫描方向的，但是此模块为树莓派专用，同时也为了程序的兼容性，此处虽然定义了函数，但是不会影响扫描方形。

DEV_Config.c：定义了树莓派的管脚及通信方式。

LCD_Driver.c：OLED 的驱动，通常情况下不需要做修改

LCD_GUI.c：常用的画点，线，图，字函数，通常情况下你只需要修改 GUI_Show()这个 jams 就，这个函数为显示调用函数。

LCD_BMP.c: 读取解析 bmp 图片文件，然后进行显示。

Makefile：工程的编译规则，如果更改了代码，需要先执行 make clear 清楚全部文件依赖以及生产的可执行文件，然后再执行 make，这样 makefile 就会自动编译整个项目从而生成可执行文件。

oled_1in44：可执行文件，通过 make 命令生成

用户使用的时候只需要执行 `sudo ./oled_1in44` 执行程序即可

3.2 WiringPi 使用

运行 ls 命令，可见如下文件

```
pi@raspberrypi:~/wiringpi $ ls
bin  Fonts  Makefile  obj  pic  tftlcd_1in44
```

WiringPi 与 BCM2835 文件目录相同，区别在于两点：

第一：WiringPi 是通过读写 linux 系统的设备文件操作，而 bcm2835 则是树莓派 cpu 芯片的库函数，操作的为寄存器，因此若先使用了 BCM2835 库，WiringPi 则会使用失败，此时重启系统可以运行

第二：由于第一个区别的原因，他们底层的配置就不一样，在 DEV_Config.c 中使用的为 WiringPi 及其相应的 wiringPiSPI 来提供底层接口。

同样，只需运行 `sudo ./tftlcd_1in44` 即可。

3.3 python：

运行 ls,可见文件如下：

```
pi@raspberrypi:~/python $ ls
LCD_1in44.py  LCD_Config.py  main.py  sky.bmp  time.bmp
```

LCD_1in44.py 中为 LCD 驱动程序；

LCD_Config.py 为硬件底层接口配置运行。

运行程序：sudo python main.py

注：对于有些树莓派系统可能并没有 image 这个库，运行：sudo apt-get install python-imaging 安装 python-imaging 库

Image 是 python 图像处理库，任何一副图像都是用一个 Image 对象表示，因此可以通过 new 方法来创建一张空白图片，大小与 LCD 的最大显示范围一致，通过 Draw 库来进行画图，最后把这个图片传递到 LCD 上，这里通过 Image.load()去读取点的 RGB888 数据，把他们数据转化 RGB565,通过每一点的扫描，达到整屏显示的效果。核心代码如下：

```
def LCD_ShowImage(self,Image,Xstart,Ystart):
    if (Image == None):
        return

    self.LCD_SetWindows ( 0, 0, self.LCD_Dis_Column , self.LCD_Dis_Page )
    Pixels = Image.load()
    for j in range(0, self.LCD_Dis_Page ):
        for i in range(0, self.LCD_Dis_Column ):
            Pixels_Color = ((Pixels[i, j][0] >> 3) << 11)|((Pixels[i, j][1] >> 2) << 5)|((Pixels[i, j][2] >> 3)#RGB Data
            self.LCD_SetColor(Pixels_Color , 1, 1)
```

3.4 启动运行

通过配置/etc/rc.local，使代码在树莓派启动时运行。

运行：

```
sudo vim /etc/rc.local
```

在 exit 0 前加上：

```
sudo python /home/pi/code/oled/demo.py &
```

需要注意的是：/home/pi/oled/demo.py 为您放置例程所在目录位置，可以通过命令：pwd 来获取。

还有务必在结尾加上&，否则您可能需要重装系统（无法通过 ctrl+c 终止进程，无法登录到树莓派用户下）

4. 关于兼容代码的移植

提供的 Demo，是一种通用程序，是可以兼容两种屏幕，他们之间的区别也就在于初始化显示区域的大小不同。

使用方式也是通过定义宏，来选择不同的初始设置。

在 LCD_Driver.h 其中：

```
#define LCD_1IN44
```

```
#define LCD_1IN8
```

这两个宏如名字一样分别代表着 1.44 寸和 1.8 寸，只需要给屏蔽不使用的那个屏对应的宏值就可以选择对应的初始化。

例：

```
#define LCD_1IN44
```

```
//#define LCD_1IN8
```

选择 1.44 寸的初始化配置，保存后通过 make clear 去清除依赖文件，再运行 make 生成可执行文件即可。