驱动 ST7565 显示汉字以及画点

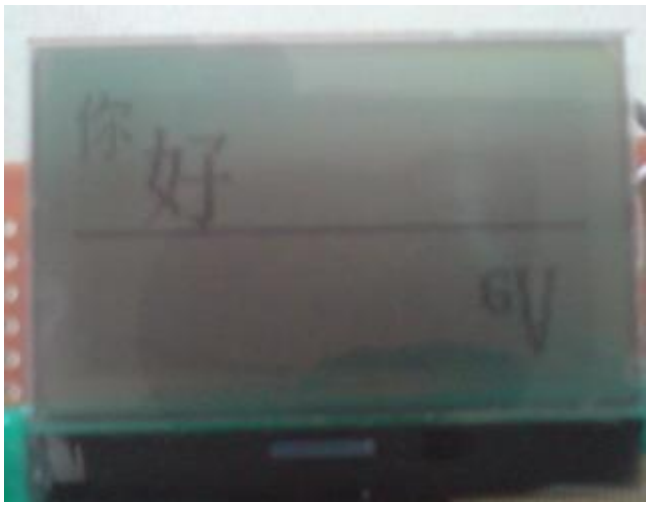
最近需要用 ST7565 来显示汉字以及画点，发现网上关于 ST7565 驱动显示文字的例子也不少，不过画图方面的例子就很少了。ST7565 是比较常见的 128*64 的 LCD，我这里使用模拟 SPI 来写 ST7565，ST7565 是“纵向 8 点下高位”类型的 LCD，难点在于页（Y 轴）。

如下图，这里 Y=6

纵向LCD		6的二进制的倒向
0		0
1		0
2		0
3		0
4		0
5		0
6	■	1
7		0

一个 8 位数据对应 LCD 纵向的 8 个格子，例如：要在 Y=6 地方亮一个点，把二进制 0100 0000 写到 ST7565 里，实际在纵向倒数第二个点显示一个点。

理论解释完了，接下来看看本例中实现的结果，看下图：



接下来贴上代码，由于每个厂家的 ST7565 的屏的接线都会有区别，所以这里就不给出写 ST7565 的实现，SPI_Write（）就是写 ST7565 函数，通过 LCD_CMD/LCD_CANVAS 来区分是写控制命令还是图像数据。这里的字模来自 Lcmzimo 字模工具。

view plaincopy to clipboardprint?

```
    //汉字 16*16 的定义

    unsigned int CHINESE_16_16 [] = {32/*数据总数*/, 16/*宽*/, 16/*高*/};

    //汉字 24*24 的定义

    unsigned int CHINESE_24_24 [] = {72/*数据总数*/, 24/*宽*/, 24/*高*/};

    //ASCII 16*8 的定义

    unsigned int ASCII_W8_H16 [] ={16/*数据总数*/, 8/*宽*/, 16/*高*/};

    //ASCII 24*12 的定义

    unsigned int ASCII_W12_H24 [] = {36/*数据总数*/, 12/*宽*/, 24/*高*/};

    // ******************************************************

    //画字函数

    //参数 x:X 轴坐标，0~127

    //参数 y_pag：纵向页数，0~7，每一页等于 8 个纵向像素

    //参数 font:font 的样式：{数据总数，高，宽}

    //参数 p：字模数组，当 p=LCD_CLEAR，则是清除指定区域

    // ******************************************************

    void LCD_PutChar（unsigned char x，unsigned char y_pag，unsigned int *font，unsigned char *p）

    {

    unsigned int size=font [0] ;//整个数组的大小

    unsigned int width=font [1] ;//字符的宽度

    //unsigned int height=font [2] ; 留着以后有用

    unsigned int pagindex=1;//记录 LCD 页指针去到的页数
```

```
unsigned int nextpage = width;
unsigned char i, pag, colh, coll;
pag = y_pag+0xb0;
colh = x>>4; /*取 y_pag 的高 4 位*/
colh = colh | 0xf0;
colh = colh & 0x1f;
coll = x & 0x0f; /*取 y_pag 的低 4 位*/
SPI_Write (colh, LCD_CMD) ;
SPI_Write (coll, LCD_CMD) ;
SPI_Write (pag, LCD_CMD) ;
for  (i=0;i<size;i++)
{
if  (i == nextpage) //当前页画完，则跳转到下一页继续画
{
SPI_Write (pag+pagindex, LCD_CMD) ;
SPI_Write (colh, LCD_CMD) ;
SPI_Write (coll, LCD_CMD) ;
pagindex++;//换到下一页
nextpage = pagindex * width;//定义下一页在 size 中的位置
}
if (p==0x00)
SPI_Write (0X00, LCD_CANVAS) ;
else
SPI_Write (*p++, LCD_CANVAS) ;
```

```
    }

    }



    // ******************************************************

    //画点函数

    //原理：x 直接设置列，Y/8=页数，Y%8=点在纵 8 格的位置，Y=0，Y|=BIT7，Y=Y》》（7-Y%8）

    //举例：（5，6），在列 5，Y 坐标在第 0 页的最后一点，即 Y=0100 0000（倒向的二进制） 等价于 Y=0，Y|=BIT7，Y 左移 1 位

    //参数 x:X 轴坐标，0~127

    //参数 y:Y 轴坐标，0~63

    //参数 ph：点的高度，为 0 时则为清除点

    // ******************************************************

    void LCD_DrawPoint（unsigned char x，unsigned char y，unsigned int ph）

    {

    unsigned char i，pag，colh，coll;

    pag = y/8;//判断 Y 所在的页

    pag = pag +0xb0;

    colh = x》》4; /*取 x 的高 4 位*/

    colh = colh | 0xf0;

    colh = colh & 0x1f;

    coll = x & 0x0f; /*取 x 的低 4 位*/

    SPI_Write（colh，LCD_CMD）;
```

```
    SPI_Write（coll，LCD_CMD）；

    SPI_Write（pag，LCD_CMD）；

    if（ph==LCD_CLEAR）

    {

    SPI_Write（LCD_CLEAR，LCD_CANVAS）；

    return;

    }

    unsigned int point=0;

    for（i=0;i<ph;i++）

    point|=（BIT7>>i）；//点加高

    point=point>>（8-ph-y%8）；//加高之后移位

    SPI_Write（point，LCD_CANVAS）；

    }

    unsigned char hz16_16［］={//“你”

    0x40，0x20，0xF8，0x07，0x40，0x20，0x18，0x0F，0x08，0xC8，0x08，
0x08，0x28，0x18，0x00，0x00，

    0x00，0x00，0xFF，0x00，0x00，0x08，0x04，0x43，0x80，0x7F，0x00，
0x01，0x06，0x0C，0x00，0x00

    }；

    unsigned char hz24_24［］={//“好”

    0x00，0x40，0x40，0x40，0xFF，0xFE，0x42，0x40，0xE0，0xE0，0x40，
0x00，0x08，0x08，0x08，0x08，

    0xC8，0x88，0x68，0x38，0x1C，0x08，0x00，0x00，0x00，0x00，0x70，
0x7F，0xCF，0x80，0x00，0xF0，

    0x7F，0x0F，0x10，0x10，0x10，0x10，0x10，0x10，0xFF，0xFF，0x10，
0x10，0x18，0x18，0x10，0x00，
```

```
    0x00, 0x40, 0x20, 0x10, 0x0C, 0x07, 0x03, 0x07, 0x1E, 0x1C, 0x00,
0x00, 0x20, 0x20, 0x60, 0xE0,

    0x7F, 0x3F, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};

    unsigned char ascii8_16 [] ={// -G-

    0xF0, 0xF8, 0x0C, 0x84, 0x84, 0x8C, 0x98, 0x00, 0x03, 0x07, 0x0C,
0x08, 0x08, 0x07, 0x0F, 0x00

    };

    unsigned char ascii12_24 [] ={// -V-

    0x04, 0xFC, 0xFC, 0x04, 0x00, 0x00, 0x00, 0x04, 0xFC, 0xFC, 0x04,
0x00, 0x00, 0x00, 0x3F, 0xFF,

    0xC0, 0x00, 0xC0, 0xFF, 0x3F, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x03, 0x1F, 0x3C, 0x1F, 0x03,

    0x00, 0x00, 0x00, 0x00,

    };

    unsigned char *hzcode;

    void main ()

    {

    WDTCTL=WDTPW+WDTHOLD; //停止 WDT

    LCD_Init () ;//初使化

    LCD_SetDisplay (LCD_CLEAR) ;

    //显示“你”

    hzcode= hz16_16;

    LCD_PutChar (0, 0, CHINESE_16_16, hzcode) ;

    //显示“好”

    hzcode= hz24_24;
```

```
LCD_PutChar（16，1，CHINESE_24_24，hzcode）；

//显示分割线

for（int x=0;x<127;x++）

LCD_DrawPoint（x，32，1）；

//显示“G”

hzcode= ascii8_16;

LCD_PutChar（100，5，ASCII_W8_H16，hzcode）；

//显示“V”

hzcode= ascii12_24;

LCD_PutChar（108，5，ASCII_W12_H24，hzcode）；

}
```