OLED 屏的驱动与使用

文/杜洋

【OLED 时代来了】

一提到 OLED，我想无人不知。智能手机的迅猛发展，使得屏幕显示技术不断推陈出新。虽然最新报出三星将产量更新颖的 QLED 屏幕，但在没有上市之前智能手机屏幕从显示原理上仍只有 LCD 和 OLED 两种。在三星和 LG 两家公司的高端手机上都采用了 OLED 屏幕，其他手机生产商还是采用传统的 LCD 屏幕。其两者最大的区别就是：LCD 屏幕是由灯管或 LED 做背光照明，再通过 LCD 屏通过或阻断光线达到显示目的。OLED 是有机发光二极管，它和我们常用的 LED 是近亲。OLED 不需要背光灯，其上面每一个像素点都能独立发光。综合来看 OLED 显示技术功耗更低、效果更有优势。我曾用过一款手机，采用的是 SUPER OLED 屏幕，无论从色彩还是可视角度都非常出众。不过现在仅有几家大公司才能生产 OLED 屏幕，产量不足，价格昂贵。

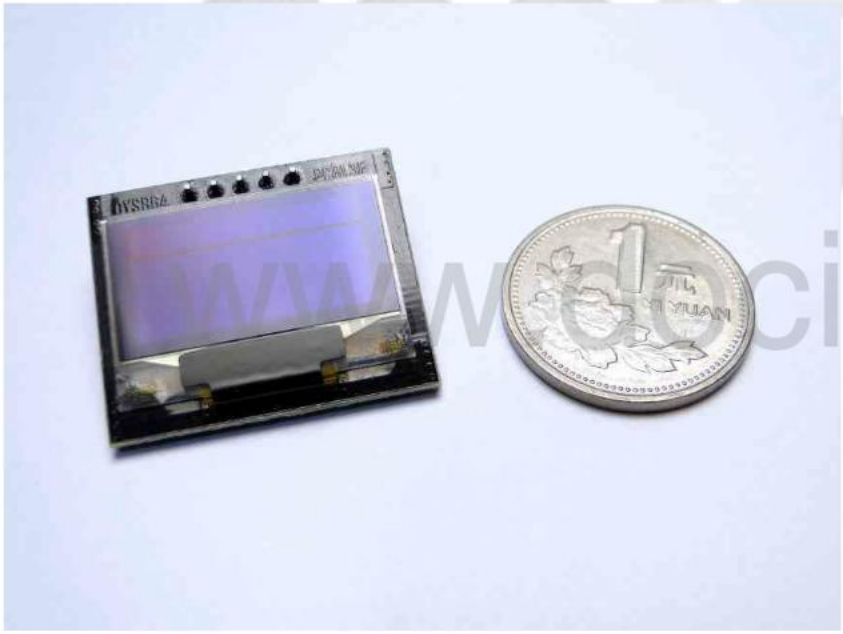
OLED 屏供应手机市场都不足够，作为一名电子爱好者更不敢奢求有一天能亲手把玩它了。记得 3 年前，我在北京工作的时候，就曾在电子市场看到久闻未见的 OLED 屏幕。之所以让我一眼就识别出来，是因那柜台里每一块屏幕都只是一片薄薄的玻璃，发着光的文字和图案在上面不停切换，周围没有背光灯，没有电路板，只有一片玻璃。旁人也许并无感想，但对我这个电子发烧友来说，这种震撼不亚于第一次看到没有“大包”的液晶电视。站在柜台前看了半天，老板见势走过来。我指着那片 1 元硬币大小的屏幕问多少钱，老板一本正经地说：“这款小的便宜，你要多少？”我说只要 1 个。他有些不太高兴的说：“一个呀，280 元”。听闻此言，二话没说，转身就跑。只是一块单色的小屏幕，如果是 LCD 液晶屏顶多 80 元，就因为是 OLED 就多出 200 元。可是又忍不住对它的神奇所吸引，好想能早点把玩。

时间是把无情刀，能杀人，也能杀价。如今的 OLED 技术已经广泛被使用，产量高了，价格自然就降低了。从前上百元的 OLED 屏幕，现在只要几十元。还有各种尺寸大小、颜色和分辨率。如此一来，我们每一位电子爱好者都有能力玩转 OLED，体验新型显示技术所带来的乐趣，OLED 屏的 DIY 时代已经到来了！对于我而言，OLED 可是低功耗便携设备的优质显示屏，不用它来制作一款便携产品实在太可惜了。经过几个月的创新构想，我正在开发一款 OLED 屏幕的运动型手表，这款手表不仅能显示日期和时间，还有计步功能。更为重要的是，这是一款电子 DIY 的运动型手表，每位电子爱好者都能根据自己的需要订制显示内容。可以写上对某人的祝福，提醒某些重要纪念日。相信这款手表会带给电子爱好者一片新的制作领域。到时我也会撰文介绍这款作品。不过在手表问世之前，我想把刚刚研究明白的 OLED 驱动及中英文显示的方法与大家分享，希望大家能通过我的文章学会 OLED 的使用。

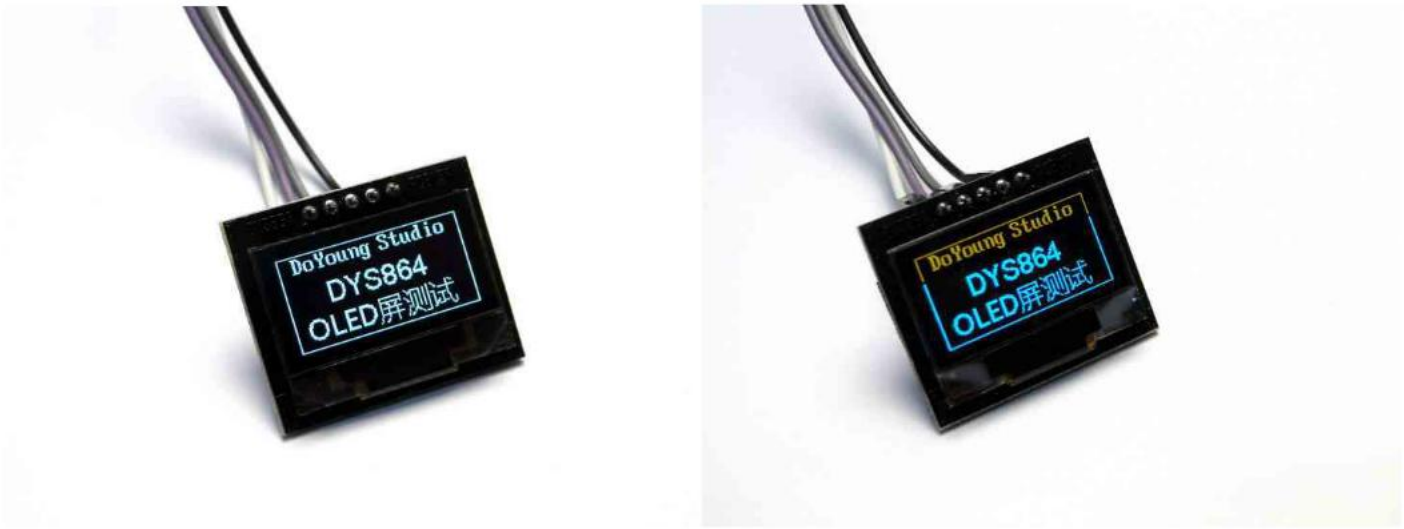
【OLED 屏的硬件连接】

OLED 屏虽然说是新鲜事物，但依然遵循、沿用 LCD 显示屏的设计方案。一来是降低开发成本，二来也让曾使用 LCD 屏的人快速上手。OLED 屏有全彩色和单色屏两种，前者用在手机等 3C 产品上，后者用在工业、医疗、商业的嵌入式产品上。在工业、医疗、商业领域，早就采用了 LCD 显示屏，比如我们熟悉的 LCD1602、LCD12864 等。但 LCD 屏的体积大、背光功耗大、环境要求高，而新的 OLED 屏正好可以解决这些问题，OLED 屏的生产商是希望自己的 OLED 屏可以取代传统 LCD 屏，所以 OLED 屏被设计成可兼容原来 LCD 屏的规格，如尺寸、分辨率、安装方式。原来 LCD 屏常用的分辨率有 128x64，那 OLED 屏也照旧。在屏幕驱动控制上，OLED 屏生产商也尽量与传统驱动电路的指令保持一致。相同功能的直接照搬，有差异的尽量靠近。所以请大家放心，学会 OLED 屏的驱动和显示并不难。

本文介绍的这款 DYS864 显示屏模块，是一款 0.96 英寸的 OLED 屏。模块由一片 OLED 玻璃屏幕和一个驱动 PCB 板组成。屏幕有单白色和黄绿双色两种颜色可选，其中黄绿双色并不是传统上理解的双色屏。而是把屏幕分成上下两部分，上面 1/4 是黄色，下面 3/4 是蓝色。模块分辨率是 128x64，内置了升压电路（OLED 点亮显示 8~14V 的高电压）和复位电路，所以只要 3~5V 电源输入电压即可。采用 I2C 总线通过，电路连接简洁。0.96 英寸的大小与 1 元硬币差不多。小体积带来小的功耗，只要 3mA 左右的电流就能让它显示内容，而点亮一支 LED 灯还需要 20mA 的电流呢。把它放在用电池供电的小制作上，再完美不过了。

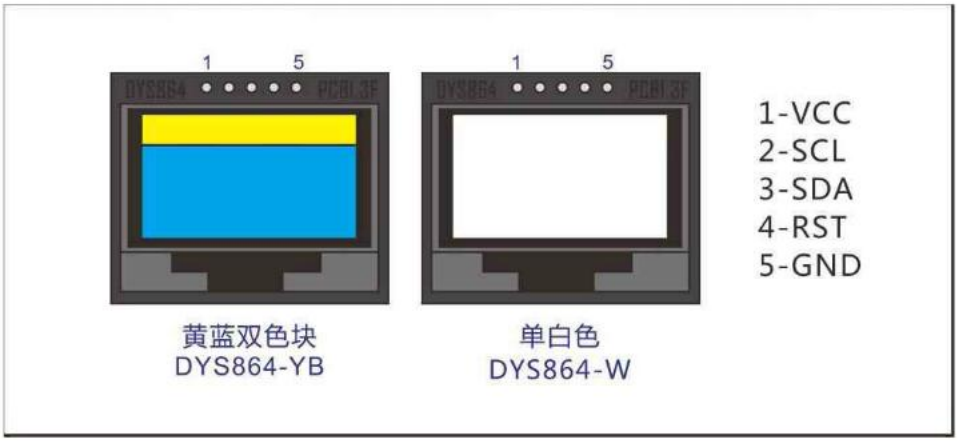


【图 1】DYS864 模块的体积小巧

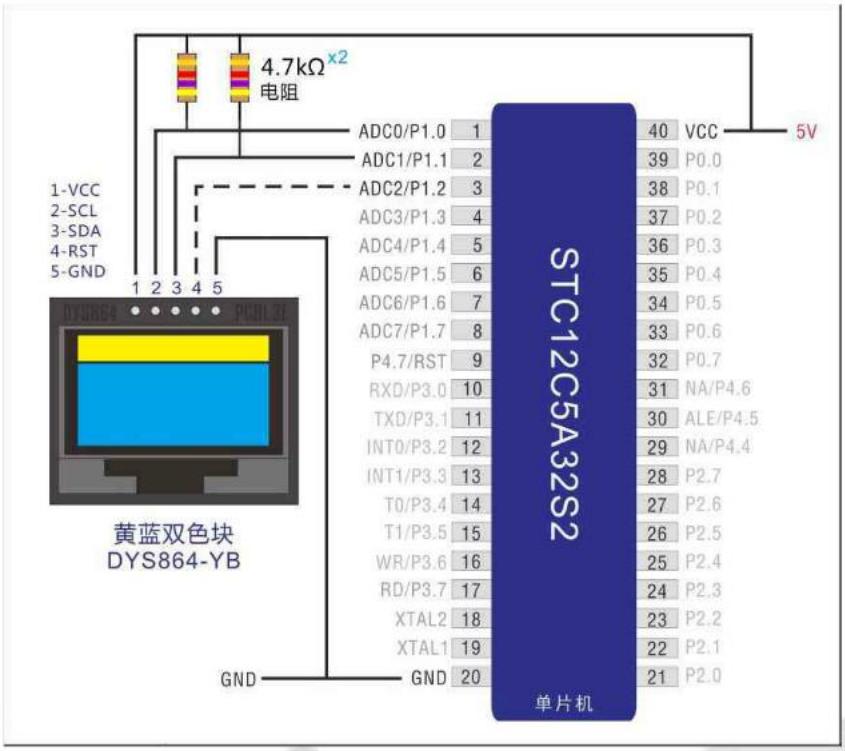


【图 2】两种颜色 OLED 屏的显示效果

你可以把 OLED 屏上的像素点理解成 LED 灯，小小玻璃片上嵌入了 8192（128x64）个 LED，对应的 LED 点亮或熄灭，则形成图案和文字。想一想，用单片机驱动 LED 灯阵列时通常要专用的驱动芯片，那 OLED 屏上的“LED 灯”自然也要有一个驱动芯片。可是你并不会发现有什么芯片，那是因为生产商把驱动芯片嵌到玻璃片当中了，就在显示区域的正下方。这种设计工艺叫 COG，很多 LCD 液晶屏也采用过这种设计。OLED 屏驱动芯片的型号有很多种，我们这款用的是 SH1106 芯片，你能在网上搜索到这款芯片的数据手册。驱动芯片的引脚非常多，主要是用来连接 OLED 屏阵列上的“LED 灯”，留给用户的接口并不多。DYS864 模块的 PCB 上已经将驱动芯片的外围电路做好了，留给你使用的只有电源、I2C 总线和复位接口。你不需要了解复杂的驱动电路设计，只要把 DYS864 与单片机连接，专心于编程部分即可。【图 3】是 DYS864 模块的接口定义，【图 4】是模块与单片机连接的电路图。其中模块的 1 和 5 脚是 VCC 和 GND，接 3V 或 5V 电源（我用的是 5V）。2 和 3 脚是 I2C 总线接口，与单片机的 I/O 接口连接。我使用的是 STC12C5A32S2 型单片机，其内部没有 I2C 的硬件控制器，要用软件模拟 I2C 总线。模拟程序部分我已经写好了，在源程序里直接调用。【图 4】中所画的两个 4.7KΩ电阻是 I2C 总线设计时必须的上拉电阻，它能让没有强上拉能力的单片机在模拟总线时保持很好的稳定性，不过我们使用的 STC12 系列单片机有强上拉能力，所以电阻可省去不接。最后，模块上的 4 脚（RST）是复位引脚，如果你想用单片机随时控制 OLED 模块重新复位初始化，可把第 4 脚连到 I/O 接口上，I/O 接口输出低电平时屏幕复位。不过在我们的使用当中用不到复位，所以第 4 脚可不接。如此一来，不需要接电阻，不需要接复位，把 2 根电源线和 2 根 I2C 总线接到单片机上就行了。如果你有其他 I2C 总线模块也可以挂在同一条总线上，电路制作十分简单！



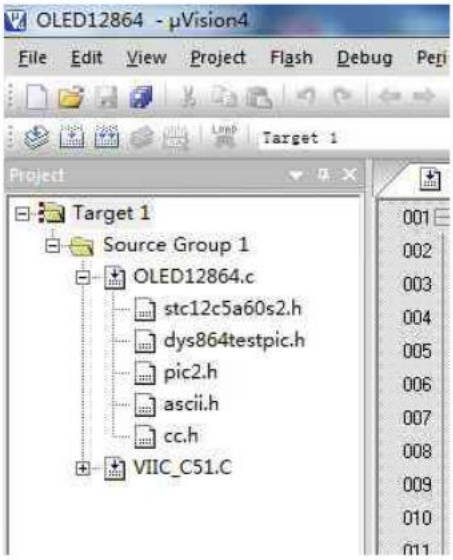
【图 3】DYS864 模块接口定义



【图 4】DYS864 模块与单片机连接电路图

【模块初始化与功能设置】

要想点亮 OLED 屏模块，编程部分是关键。幸好我已经写好了一套示例程序，你可以在《无线电》杂志的官方网站上找到下载链接。【图 5】这是名为“OLED12864”的一套 KEIL4 工程文件，打开工程文件可以看到名为 OLED12864.c 的主程序，还有 VIIC_C51.c 的 I2C 总线模拟程序。这里我们重点讲解 OLED12864.c。在主程序里还包含了 4 个头文件，stc12c5a60s2.h 是单片机头文件，dys864testpic.h 和 pic2.h 是全屏图片显示时所用的图片数据表，ascii.h 是显示英文、数字、符号用的数据表，cc.h 是显示汉字用的数据表。也就是说，接下来我会讲到如何用 DYS864 模块显示全屏图片、英文和汉字内容，但在此之前必须先让模块初始化，设定好屏幕的相关参数，然后才能开始显示。



【图 5】工程文件的内容

【程序 1】DYS864 模块初始化程序

```
void OLED_INIT (void){//DYS864 模块初始化程序
    unsigned char buf[]={//以下是初始设置参数
        0xae,//【0xae:关显示，0xaf:开显示】
        0xd5,0x80,//显示时钟频率（不要改动）
        0xa8,0x3f,//复用率（不要改动）
        0xd3,0x00,//显示偏移（不要改动）
        0x40,//显示开始线
        0x8d,0x14,//VCC 电源（不要改动）
        0xa1,//设置段重新映射（不要改动）
        0xc8,//COM 输出方式（不要改动）
        0xda,0x12,//COM 输出方式（不要改动）
        0x81,0xff,//对比度【指令：0x81，数据：0 到 0xff（最高）】
        0xd9,0xf1,//充电周期（不要改动）
        0xdb,0x30,//VCC 电压输出（不要改动）
        0x20,0x00,//水平寻址设置
        0xa4,//【0xa4:正常显示，0xa5:整体点亮】
        0xa6,//【0xa6:正常显示，0xa7:反色显示】
        0xaf//【0xae:关显示，0xaf:开显示】
    };
    RET=0;//OLED 复位
    DELAY_MS (50);
    RET=1; //复位脚放开
    DELAY_MS (50);
    ISendStr(OLED12864_ADD,COM,buf,25);//写入 25 个字节的初始设置参数
}
```

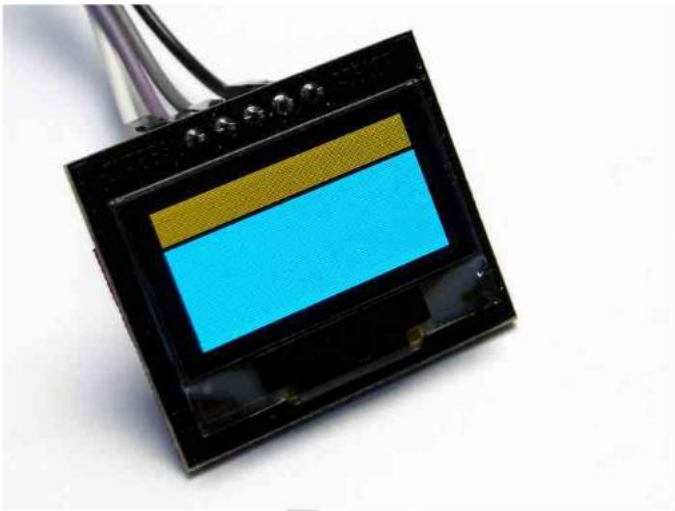

【程序 1】是 DYS864 模块的初始化程序，其内容是将 25 个指令写入 OLED 屏的驱动芯片。这些指令能让显示屏调整到正常的工作状态，就好像新买的手机需要先设置好铃声、音量、亮度一样，是一个初始的过程。【程序 1】里已经把每一个指令后面加了解释，有兴趣可以看一下，要想了解更深层的意思，就要看驱动芯片的数据手册了。虽然手册是英文的，但还是不难看懂的。但请注意，如果不明白的指令就不要改动。其中“0x81,0xff”（双字节指令，对比度调节）可以根据你的喜好调下对比度（也就是亮度），后一个字节的值从 0x00 到 0xff，一共 256 级对比度调整。写入 25 个字节数据之前，程序先给复位引脚一个低电平，让显示屏复位。不过上电时 DYS864 模块内部的复位电路已经使它复位一次了，所以不接复位引脚也没关系。写入指令后，显示屏就准备就绪，可以写入内容了。

【全屏图片显示方法】

和 LCD 屏幕一样，OLED 也能全屏显示单色图片。这里介绍两种全屏图片显示的方法，第一种是用同一个数据复位写满全屏。【程序 2】是单字节数据重复写入全屏的程序，它是写入 0xff 也就是所有点全亮的效果，如【图 6】所示。在【程序 2】中分析，实现起来并不难，只要用 for 循环，重复写入 0xff 这个数据 1024 次，每次写入后驱动芯片会自动将写入点移到下一格，所以我们不用考虑换行的问题。

【程序 2】单字节数据的全屏写入

```
//全屏固定像素写入示例
ISendStr2(OLED12864_ADD,COM,0xB0);//从第 1 页开始写入
for(i=0;i<1024;i++){//写入满屏 1024 个字节
    ISendStr2(OLED12864_ADD,DAT,0xff);//写入 0xff
}
```



【图 6】单字节数据全屏写入效果

第二种是将一幅 128x64 的单色无灰度图片转换成数据表，再从数据表中读取显示。这是常用的方式，多用于开机画面等静态内容的显示。单色图片的生成方法很简单，首先在网上找到想要的单色图片，再用画板软件或其他图片处理软件将像素变成 128x64，保存为单色 BMP 位图。然后用图片取模软件打开位图，转成数据表。【图 7】是我用一款常用的取模软件，将参数中的像素设置为 128 和 64，打开图片，然后点击“数据保存”。把文件保存为头文件（.h）。【图 8】所示是最终改好后，可编译的数据表文件，大家可以参考示例程序。

【图 7】设置并调整图片

```
001 //////////////////////////////////////////////////////////////////////
002 // Bitmap点阵数据表                                                 //
003 // 图片: E:\..萌八  246.bmp,纵向取模下高位,数据排列:从左到右从上到下   //
004 // 图片尺寸: 128 * 64                                               //
005 //////////////////////////////////////////////////////////////////////
006 unsigned char code PIC2[] =                    // 数据表
007 {
008       0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
009       0x00,0x00,0x00,0x00,0xFC,0xFE,0x0E,0x1C,
010       0x18,0x1C,0x0E,0xFE,0xFC,0x00,0x7C,0xFC,
011       0x80,0xFC,0xFC,0x00,0x00,0x00,0x7C,0xFE,
012       0xC6,0xC6,0xC6,0x00,0x40,0xE8,0xA8,0xF8,
013       0xF0,0x00,0xF8,0xF8,0x10,0x18,0x00,0x00,
014       0xC0,0xC0,0x00,0xC0,0xC0,0x00,0xC0,0xC0,
015       0x00,0x00,0x06,0x09,0x12,0x09,0x06,0x00,
016       0x00,0x00,0x00,0x00,0x00,0x00,0x80,0x40,
017       0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0C,
018       0x12,0x22,0x44,0x22,0x12,0x0C,0x00,0x00,
019       0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
020       0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
021       0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
022       0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
023       0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
```

【图 8】导出的图片数据表

有了图片数据表，接下来就是在主程序里调用数据表送入 OLED 屏显示。【程序 3】是两幅图片以 1 秒间隔切换显示的程序。首先是发送一条指令 0xB0，表示图片从第 1 页首位置开始显示。之后是 1024 次 for 循环，把数据表写入显示屏。在 for 循环里面是数据表发送程序，每一个图片都有自己的数据表名称（nBitmapDot[ ]和 PIC2[ ]），要显示那个图片，就改成对应的数据表名称就行了。这里注意，数据表的取模时一定要设置成 128x64 像素，不然与屏幕不匹配会产生乱码。【图 9】是小车图片的显示效果。

【程序 3】写入全屏图片的程序

```
ISendStr2(OLED12864_ADD,COM,0xB0);//从第 1 页开始写入
for(i=0;i<1024;i++){//写入整屏图案（DYS864 测试）
    ISendStr2(OLED12864_ADD,DAT,nBitmapDot[i]);//写入数据
}
DELAY_MS (1000);//显示停留 1 秒
ISendStr2(OLED12864_ADD,COM,0xB0); //从第 1 页开始写入
for(i=0;i<1024;i++){//写入整屏图案（MY CAR）
    ISendStr2(OLED12864_ADD,DAT,PIC2[i]); //写入数据
}
DELAY_MS (1000);//显示停留 1 秒
```

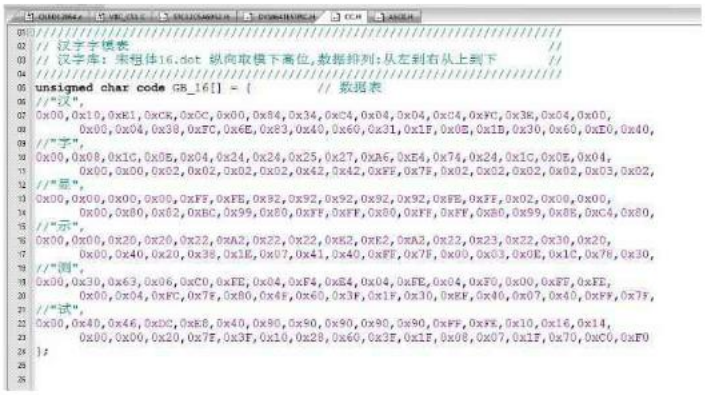
【图 9】写入全屏图片的效果

【中英文字符的显示方法】

除了全屏图片显示之外，在屏幕指定位置显示文字也是必须要掌握的。从前面的学习中能看出 OLED 与传统 LCD 屏的差异不大。但这款屏没有内部字库，不论是英文还是汉字，都要事先取模并导出数据表。大家可以把文字的显示理解成小像素的图片，一个汉字的像素是 16x16，一个英文或数字的像素是 8x16。虽然你也可以设计更大尺寸的字体，但字越大显示的内容越少，所以在开发之前你要多试验一下字体的大小。首先来为英文和数字所在的 ASCII 码取模吧。如【图 10】所示，在取模软件里选择 ASCII 码，字库选择 8x16，数据排列选择“从左到右从上到下”，取模方式选择“从向 8 点下高位”。点击“确定参数”后，会出现“ASC”的按键，点击可打开 ASCII 字库。把字库内容复制到头文件 ASCII.h 里，因为 ASCII 码字库内容并不多，全部复制到工程里也不会占太多空间。汉字的取模占用空间大，所以只能用什么字只取相应字的数据。【图 11】是汉字取模的设置参数和输出的数据表，注意在字库中要选择“粗宋 16 点阵”，只有这个字体在小屏幕上能清楚看到。然后在软件下方的输入框中输入你想要显示的汉字，如果汉字有重复，取模软件会自动只取一次。我们这里只是测试，于是我写了“汉字显示测试”几个字，然后点击“输入字串”按钮。在弹出的数据表窗口中可看到取出的数据，把数据表复制保存到 cc.h 文件中，结果就得到【图 12】所示的汉字内容数据。注意原数据表中的汉字部分要加//屏蔽。

【图 10】设置并导出 ASCII 码数据表

【图 11】设置并导出汉字数据表



```
//////////////////////////////////////////////////////////////////////
// 汉字字模表                                                       //
// 汉字库: 宋体16.dot 横向取模下高位,数据排列:从左到右从上到下     //
//////////////////////////////////////////////////////////////////////
unsigned char code GB_16[] = {          // 数据表
//"汉",
0x00,0x10,0xE1,0xCE,0x0C,0x00,0x84,0x34,0xC4,0x04,0x04,0x04,0xFC,0x3E,0x04,0x00,
     0x00,0x04,0x38,0xFC,0x6E,0x83,0x40,0x60,0x31,0x1F,0x0E,0x1B,0x30,0x60,0xE0,0x40,
//"字",
0x00,0x08,0x1C,0x0E,0x04,0x24,0x24,0x25,0x27,0xA6,0xE4,0x74,0x24,0x1C,0x0E,0x04,
     0x00,0x00,0x02,0x02,0x02,0x02,0x02,0x42,0x42,0xFF,0x7F,0x02,0x02,0x02,0x02,0x03,0x02,
//"显",
0x00,0x00,0x00,0x00,0xFF,0xFE,0x92,0x92,0x92,0x92,0x92,0xFE,0xFF,0x02,0x00,0x00,
     0x00,0x80,0x62,0xBC,0x99,0x80,0xFF,0xFF,0x80,0xFF,0xFF,0x80,0x99,0x8E,0xC4,0x80,
//"示",
0x00,0x00,0x20,0x20,0x22,0xA2,0x22,0x22,0x52,0xE2,0xA2,0x22,0x23,0x22,0x30,0x20,
     0x00,0x40,0x20,0x38,0x1E,0x07,0x41,0x40,0xFF,0x7F,0x00,0x03,0x0E,0x1C,0x78,0x30,
//"测",
0x00,0x30,0x63,0x06,0xC0,0xFE,0x04,0xF4,0xE4,0x04,0xFE,0x04,0xF0,0x00,0xFF,0xFE,
     0x00,0x04,0xFC,0x7F,0x80,0x4F,0x60,0x3F,0x1F,0x30,0xEF,0x40,0x07,0x40,0xFF,0x7F,
//"试",
0x00,0x40,0x46,0xDC,0xB8,0x40,0x90,0x90,0x90,0x90,0x90,0xFF,0xFE,0x10,0x16,0x14,
     0x00,0x00,0x20,0x7F,0x3F,0x10,0x28,0x60,0x3F,0x1F,0x08,0x07,0x1F,0x70,0xC0,0xF0
};
```

【图 12】导出到工程里的数据表格式

接下来是调用 ASCII 和汉字到屏幕上显示，这里我设计了 2 个子程序：lcm_ascii();和 lcm_cc();。分别用于 ASCII 及汉字的显示。【程序 4】是显示英文“OLED”和数字“0.96’”的程序，lcm_ascii();中需要 3 个参数，分别是显示页位置、显示列位置和文字在数据表中的序号。在【程序 4】中先设置了一个开始列位置 s=16 和开始页（行）位置 d=0xB2，表示从第 3 页左数第 16 列开始写文字。修改这两个初始值可以把文字放在任何位置。接下来是文字的写入，在参数中会看到“s+8*0”这样的算法。S 是开始位置，随后的每一个字符都向后窜 8*n 个列，这样就使文字从左到右依次排列了。参数最后一项中的“47、44、37、36”是在 ASCII.h 文件中，字符在数据表中的顺序号。怎么知道哪一个序号对应哪一个符号呢？这就要你给每个符号标上序号，想显示哪个字符就输入哪个序号。汉字的显示方法和 ASCII 码基本相同，只是汉字点用更大的像素位置。所以在汉字和英文混合显示时做特别注意它们的差异。【程序 5】是汉字显示的程序部分，可以看出在参数上 s+16*0 中的移动基数是 16 了，其他部分基本不变。

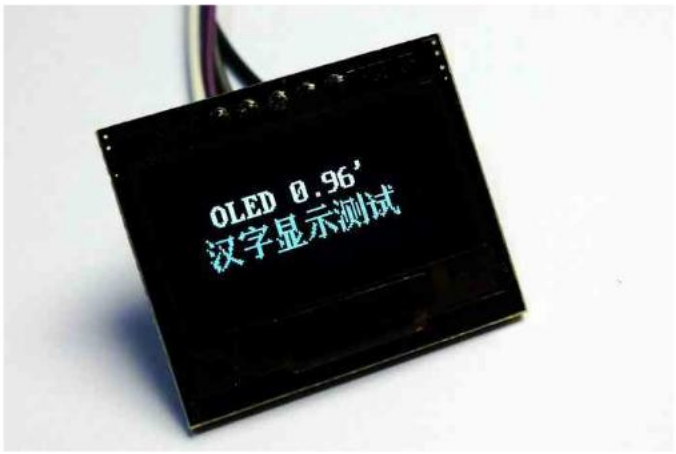
【程序 4】ASCII 码显示程序

```
s = 16;//列位置（0~255）
d = 0xB2;//页位置（0xB0~0xB7）
lcm_ascii(d,s+8*0,47);//OLED
lcm_ascii(d,s+8*1,44);
lcm_ascii(d,s+8*2,37);
lcm_ascii(d,s+8*3,36);

lcm_ascii(d,s+8*5,16); //0.96'
lcm_ascii(d,s+8*6,14);
lcm_ascii(d,s+8*7,25);
lcm_ascii(d,s+8*8,22);
lcm_ascii(d,s+8*9,7);
```

【程序 5】汉字显示程序

```
s = 8;//列位置（0~255）
d = 0xB4;// 页位置（0xB0~0xB7）
lcm_cc(d,s+16*0,0);//汉字显示测试
lcm_cc(d,s+16*1,1);//
lcm_cc(d,s+16*2,2);//
lcm_cc(d,s+16*3,3);//
lcm_cc(d,s+16*4,4);//
lcm_cc(d,s+16*5,5);//
```



【图 13】ASCII 码及汉字显示效果

【图 13】是 ASCII 与汉字分两行显示的效果。效果之所以很棒也得利于 OLED 屏自发光的特性，只有文字发光，很有未来科技感。同时你还可以自己定义新的字库，比如我正要使用 OLED 屏设计的运动手表，想让时间显示别具特色，需要设计每一个数字的形状，再取模导成数据表。其实 OLED 屏是个开放的显示模块，就像驱动 LED 点阵屏一样，可以做各种操作。同时它低功耗和小体积的特点，也能让你尽情发挥创意，制作出让人意想不到的新东西。我所知道的 OLED 屏驱动和显示技巧大概就是这些，你可以在《无线电》网站上下载到我写的源程序。没有想到 OLED 显示技术这么快就到了我们电子爱好者手里，热爱前沿显示技术的小家伙们，快来过把瘾吧。