1.5inch IIC OLED Module
用户手册

OLED 简介

OLED 即有机发光二级管（Organic Light-Emitting Diode, OLED)。OLED 显示技术具有自发光、广视角、几乎无穷高的对比度、较低耗电、极高反应速度、可用于挠曲性面板、使用温度范围广、构造及制程较简单等优点，被认为是下一代的平面显示器新兴应用技术。

OLED 显示和传统的 LCD 显示不同,其可以自发光,所以不需要背光灯,这使得 OLED 显示屏相对于 LCD 显示屏尺寸更薄，同时显示效果更优。

产品特点

- 超大可视角度:大于 160°(显示屏中可视角度最大的一种屏幕)
- 超低功耗:正常显示 0.06w(远低于 TFT 显示屏)
- 宽电压供电(3V~5V),兼容 3.3V 和 5V 电平逻辑,无需电平转换芯片
- IIC 接口只需 2 个 IO 轻松点亮
- 工作温度范围为工业级(-20℃~70℃)
- 军工级工艺标准,长期稳定工作
- 提供丰富的多平台例程,提供底层驱动技术支持
- 支持 16 位灰度显示

产品参数

名称	描述
显示颜色	白色
SKU	MSP1503
尺寸	1.5(inch)
类型	OLED
OLED 驱动芯片	SSD1327
分辨率	128*128 (Pixel)
模块接口	IIC，①-GND,②-VCC,③-SCL,④-SDA
有效显示区域	26.86x26.86(mm)
模块尺寸	45.50x34.30(mm)

视角	>160°
工作温度	-20℃~70℃
存储温度	-30℃~80℃
工作电压	3.3V / 5V
功耗	全亮约为 25mA，全灭约为 1.5mA。
产品重量	15(g)

接口说明

标号	PIN	引脚说明
1	GND	OLED 电源地
2	VCC	OLED 电源正(3.3V~5V)
3	SCL	OLED IIC 总线时钟信号
4	SDA	OLED IIC 总线数据信号

硬件配置

该 OLED 模块使用 IIC 接口，模块背面有个可以选择焊接的电阻，通过电阻焊接选择来选择不同 IIC 从设备地址，如下图所示：

如红圈内所示，如果电阻焊接在 0x78 上，则表示 IIC 的从设备地址为 0x78，如果焊接在 0x7A 上，则表示 IIC 的从设备地址为 0x7A。

工作原理

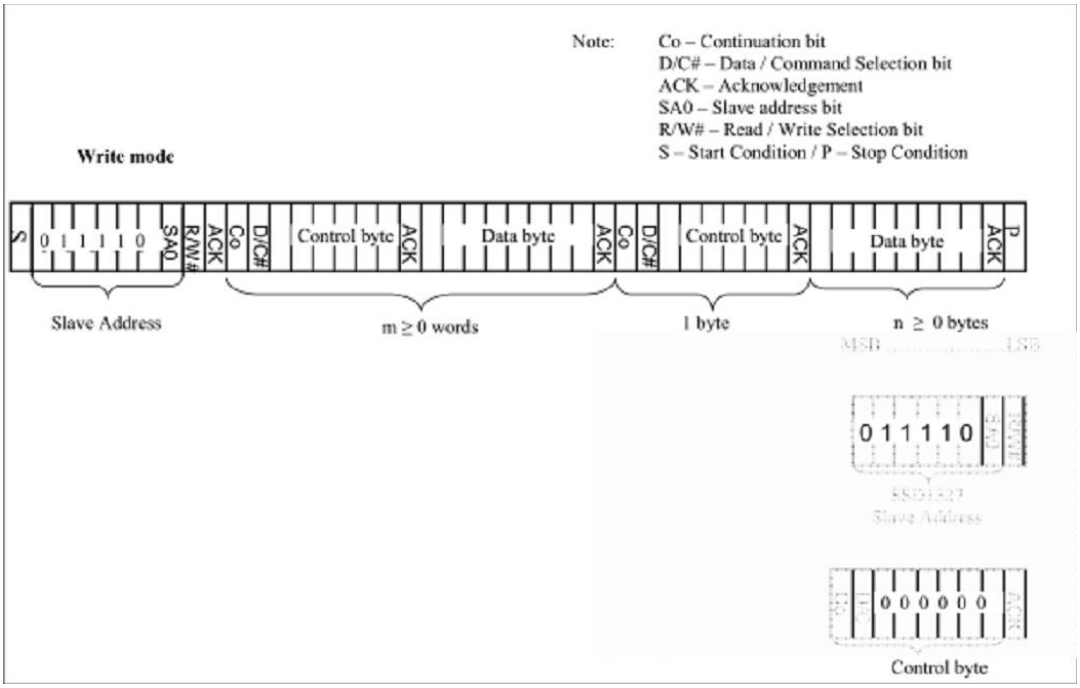
1、SSD1327 控制器简介

该 OLED 的控制 IC 为 SSD1327。SSD1327 控制器支持的最大分辨率为 128*128,同时支持 8bit 8080 并行、8bit 6800 并行、三线 SPI、四线 SPI 以及 IIC 控制，由于并行控制会浪费 IO 口，三线 SPI 不常用,因此四线 SPI 以及 IIC 控制用的最广泛，此 OLED 模块就是使用 IIC 控制。

SSD1327 控制器使用 4bit 来控制一个像素点显示，因此可以每个像素点亮度可以显示 16 个等级，这样控制器就可以支持 16 位灰度显示。每个 Byte 水平方式控制两个像素点，因此每设置一个水平坐标，控制两个像素点。

2、IIC 通信协议简介

IIC 总线写模式数据格式如下图所示：



详情见 SSD1327 datasheet 第 23 页

首先发送一个 7 个字节的从设备地址+一个字节的读写位，然后等待从设备的响应，这个与大部分的 I2C 通信是类似的，改 7 位地址可以通过 DC 管脚来控制；

然后再发送一个数据位+一个命令数据位+6 个控制自己字节，实际上有用的就是命令数据为，该位设置为 0 就是写命令，设置为 1 就是写数据；然后再发送 8 位的数据。这样一次控制传输就算完成了。

使用说明

1、Arduino 使用说明

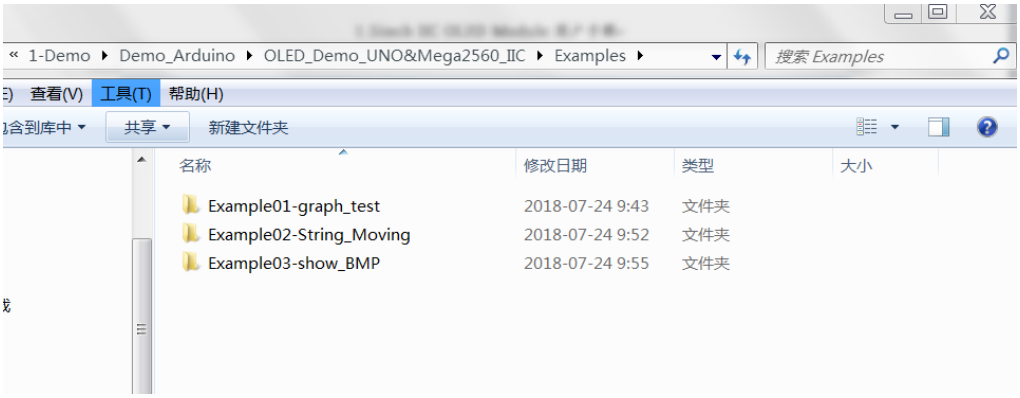
接线说明：

OLED 模块引脚	对应 UNO 单片机接线引脚	对应 Mega2560 单片机接线引脚
VCC	5V/3.3V	5V/3.3V
GND	GND	GND
SCL	A5	21
SDA	A4	20

操作步骤：

A、按照上述接线说明，将 OLED 模块和单片机连接起来，然后上电；

B、打开“1-Demo\Demo_Arduino\OLED_Demo_UNO&Mega2560_IIC\Examples”目录，选择想要测试的示例，如下图所示：



C、使用 Arduino IDE 软件对示例进行编译和下载（Arduino IDE 软件具体使用方法见 Arduino_IDE_Use_Illustration_CN.pdf）；

D、下载成功后，OLED 模块如果正常显示字符和图形，则说明程序运行成功；

2、RaspberryPI 使用说明

接线说明：

使用的GPIO库: bcm2835

Raspberry Pi GPIO map如下:

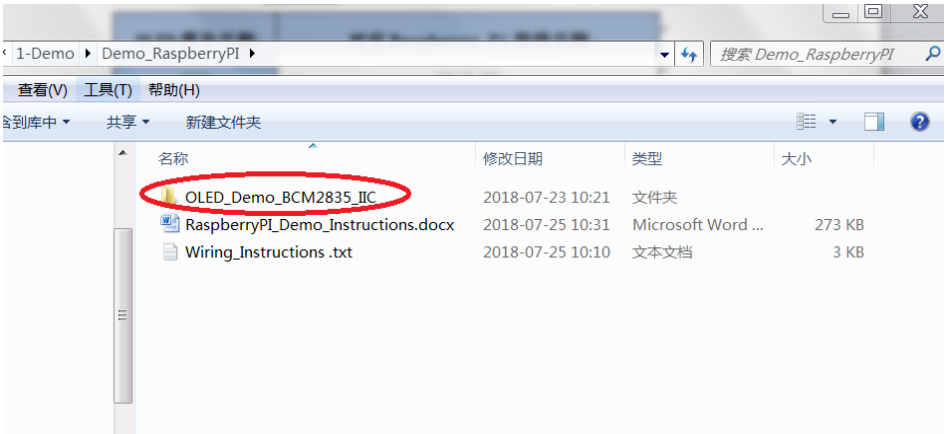
wiringPi 编码	BCM 编码	功能名	物理引脚 BOARD编码		功能名	BCM 编码	wiringPi 编码
		3.3V	1	2	5V		
8	2	SDA.1	3	4	5V		
9	3	SCL.1	5	6	GND		
7	4	GPIO.7	7	8	TXD	14	15
		GND	9	10	RXD	15	16
0	17	GPIO.0	11	12	GPIO.1	18	1
2	27	GPIO.2	13	14	GND		
3	22	GPIO.3	15	16	GPIO.4	23	4
		3.3V	17	18	GPIO.5	24	5
12	10	MOSI	19	20	GND		
13	9	MISO	21	22	GPIO.6	25	6
14	11	SCLK	23	24	CE0	8	10
		GND	25	26	CE1	7	11
30	0	SDA.0	27	28	SCL.0	1	31
21	5	GPIO.21	29	30	GND		
22	6	GPIO.22	31	32	GPIO.26	12	26
23	13	GPIO.23	33	34	GND		
24	19	GPIO.24	35	36	GPIO.27	16	27
25	26	GPIO.25	37	38	GPIO.28	20	28
		GND	39	40	GPIO.29	21	29

OLED 模块引脚	对应 Raspberry Pi 接线引脚
VCC	5V/3.3V
GND	GND
SCL	5(物理引脚编码)(BCM:3,SCL.1)
SDA	3(物理引脚编码)(BCM:2,SDA.1)

操作步骤：

A、按照上述接线说明将 OLED 模块和 Raspberry Pi 开发板连接起来，接好网线后上电；

B、打开“1-Demo\Demo_RaspberryPI”目录，将整个 OLED_Demo_BCM2835_IIC 目录拷贝到 Raspberry Pi 开发板系统里，如下图所示：



C、在 Raspberry Pi 开发板系统里进行示例编译并运行（具体 Raspberry Pi 开发板操作步骤见 Raspberrypi_Use_Illustration_CN.pdf）；

D、OLED 模块如果正常显示字符和图形，则说明程序运行成功；

3、C51 使用说明

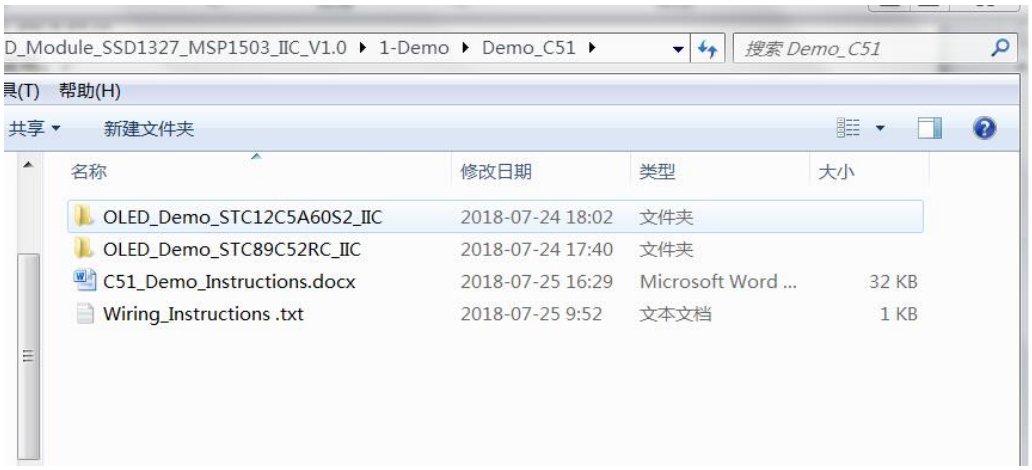
接线说明：

OLED 模块引脚	对应 Raspberry Pi 接线引脚
VCC	5V/3.3V
GND	GND
SCL	P23
SDA	P24

操作步骤：

A、按照上述接线说明将 OLED 模块和 C51 单片机连接起来，并上电；

B、打开“1-Demo\Demo_C51”目录，根据单片机型号选择测试示例，如下图所示：



C、打开所选的示例工程，进行编译和下载（C51keil 具体操作方法见 C51_Keil&stc-isp_Use_Illustration_CN.pdf）；

D、OLED 模块如果正常显示字符和图形，则说明程序运行成功；

4、STM32 使用说明

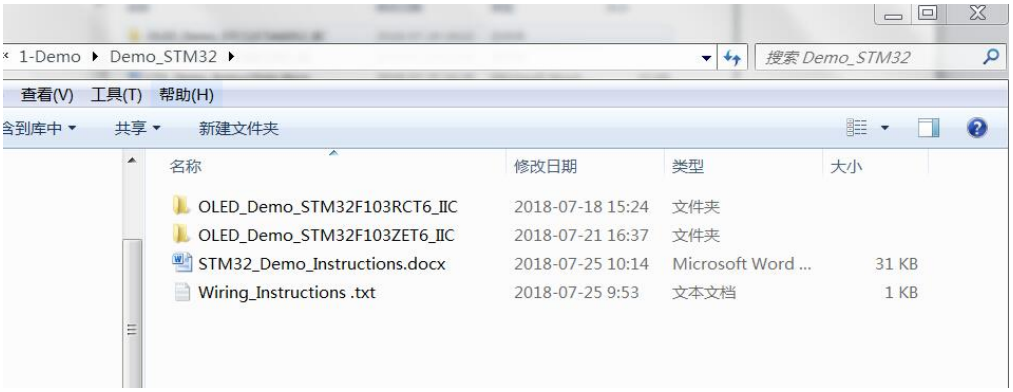
接线说明：

OLED 模块引脚	对应 Raspberry Pi 接线引脚
VCC	5V/3.3V
GND	GND
SCL	PB13
SDA	PB14

操作说明：

A、 按照上述接线说明将 OLED 模块和 STM32 单片机连接起来，并上电；

B、 打开“1-Demo\Demo_STM32”目录，根据单片机型号选择测试示例，如下图所示：



C、 打开所选的示例工程，进行编译和下载（STM32keil 具体操作方法见 STM32_Keil_Use_Illustration_CN.pdf）；

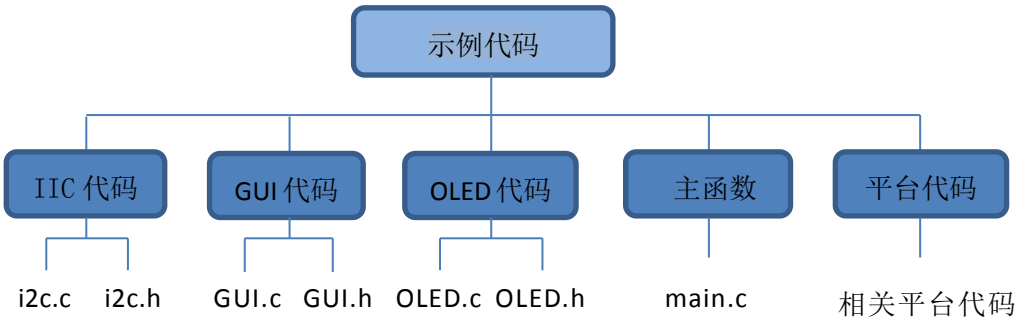
D、OLED 模块如果正常显示字符和图形，则说明程序运行成功；

软件说明

1、代码架构

A、Arduino 测试示例需要依赖 U8glib 库，底层部分都是 U8glib 实现；

B、C51、STM32 以及 Raspberry Pi 测试示例代码架构如下：



IIC 初始化以及相关的操作都包含在 IIC 代码中；

OLED 初始化以及相关的操作都包含在 OLED 代码中；

画点、线、图形以及中英文字符显示相关的操作都包含在 GUI 代码中；

主函数实现应用程序运行；

平台代码因平台而异；

2、IIC 从设备地址修改

A、Arduino 需要在 U8glib 库里面修改 IIC 从设备地址，打开

“U8glib\utility\u8g_com_arduino_ssd_i2c.c”文件，找到如下内容：

```
#include "u8g.h"

#define I2C_SLA      (0x3c*2)
//#define I2C_CMD_MODE   0x080
#define I2C_CMD_MODE     0x000
#define I2C_DATA_MODE    0x040
```

I2C_SLA 即为 IIC 从设备地址，修改此宏定义即可。

B、 C51 和 STM32 在 i2c.h 文件中修改 IIC 从设备地址，如下图所示：

```
#define IIC_SLAVE_ADDR 0x78  //IIC slave device address

void IIC_Start();
void IIC_Stop();
void Write_IIC_Command(unsigned char IIC_Command);
```

IIC_SLAVE_ADDR 即为 IIC 从设备地址，修改此宏定义即可。

C、 Raspberry Pi 在 i2c.c 文件里修改 IIC 从设备地址，包含在 IIC 初始化函数里面，如下图所示：

```
31: void IIC_init(void)
32: {
33:     bcm2835_i2c_begin();
34:     bcm2835_i2c_setSlaveAddress(0x3c);  //7 bits i2c address
35:     bcm2835_i2c_set_baudrate(1000000);  //1M I2C rate
36: }
```

bcm2835_i2c_setSlaveAddress 函数就是设置 IIC 从设备地址，地址设置后需要乘以 2。

3、模拟 IIC 的 GPIO 修改

A、Arduino 和 Raspberry Pi 使用硬件 IIC 功能，所以 IIC 的 GPIO 由系统设置，不能更改；

B、C51 和 STM32 使用软件 IIC 功能，GPIO 口可以自己设置。在 i2c.h 文件里面修改 IIC 的 GPIO 配置。

C51 平台 IIC 的 GPIO 配置如下图所示：

```
sbit OLED_SCL=P2^3; //IIC时钟信号引脚
sbit OLED_SDIN=P2^4; //IIC数据信号引脚

#define OLED_SCLK_Clr() OLED_SCL=0
#define OLED_SCLK_Set() OLED_SCL=1

#define OLED_SDIN_Clr() OLED_SDIN=0
#define OLED_SDIN_Set() OLED_SDIN=1
```

可以自己修改 OLED_SCL 和 OLED_SDIN 的值。

STM32 平台 IIC 的 GPIO 配置如下图所示：

```
//-----------------OLED IIC端口定义----------------
#define GPIO_TYPE GPIOB  //IIC GPIO类型
#define OLED_SCLK GPIO_Pin_13  //IIC时钟信号引脚
#define OLED_SDIN GPIO_Pin_14  //IIC数据信号引脚

#define OLED_SCLK_Clr() GPIO_ResetBits(GPIO_TYPE,OLED_SCLK)//CLK
#define OLED_SCLK_Set() GPIO_SetBits(GPIO_TYPE,OLED_SCLK)

#define OLED_SDIN_Clr() GPIO_ResetBits(GPIO_TYPE,OLED_SDIN)//DIN
#define OLED_SDIN_Set() GPIO_SetBits(GPIO_TYPE,OLED_SDIN)
```

可以自己修改 OLED_SCLK 和 OLED_SDIN 的值。如果要更改 GPIO 组，则需要修改 GPIO_TYPE 的值。

4、IIC 通信代码实现

A、Arduino 的 IIC 通信都是包含在 U8glib 库里面，具体实现可以去查阅 U8glib 库代码；

B、C51 和 STM32 的 IIC 通信代码在 i2c.c 中实现，如下图所示：

```
void Write_IIC_Byte(unsigned char IIC_Byte)
{
  unsigned char i;
  unsigned char m,da;
  da=IIC_Byte;
  OLED_SCLK_Clr();
  for(i=0;i<8;i++)
  {
      m=da;
    m=m&0x80;
    if(m==0x80)
    {OLED_SDIN_Set();}
    else OLED_SDIN_Clr();
      da=da<<1;
    OLED_SCLK_Set();
    OLED_SCLK_Clr();
    }
}
```

由代码可以发现，如果传输的数据位为 1 时，就将 IIC 数据信号引脚拉高，如果传输的数据位为 0 时，就将 IIC 数据信号引脚拉低。每一个时钟信号上升沿就发生 1bit 数据，

高位先传输，一次总共传输 8bit 数据。

C、Raspberry Pi 的 IIC 通信由 bcm2835 库实现，函数为 bcm2835_i2c_write;

常用软件

本套测试示例需要显示中英文、符号以及图片，所以要用到取模软件。取模软件有两种：Image2Lcd 和 PCtoLCD2002。这里只针对该套测试程序说明一下取模软件的设置。（具体使用方法见 PCtoLCD2002_Use_Illustration_CN.pdf、Image2Lcd_Use_Illustration_CN.pdf）

1、Arduino 平台取模软件设置

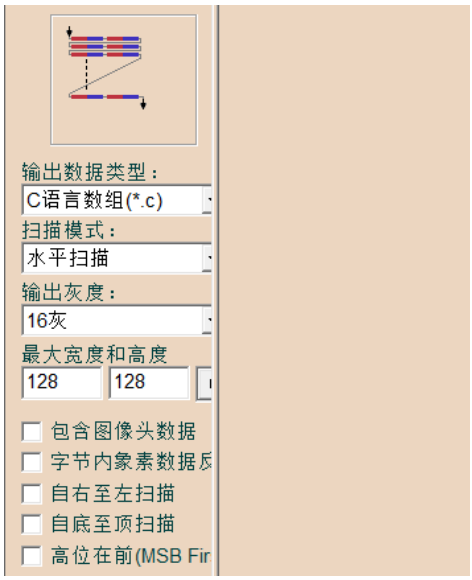
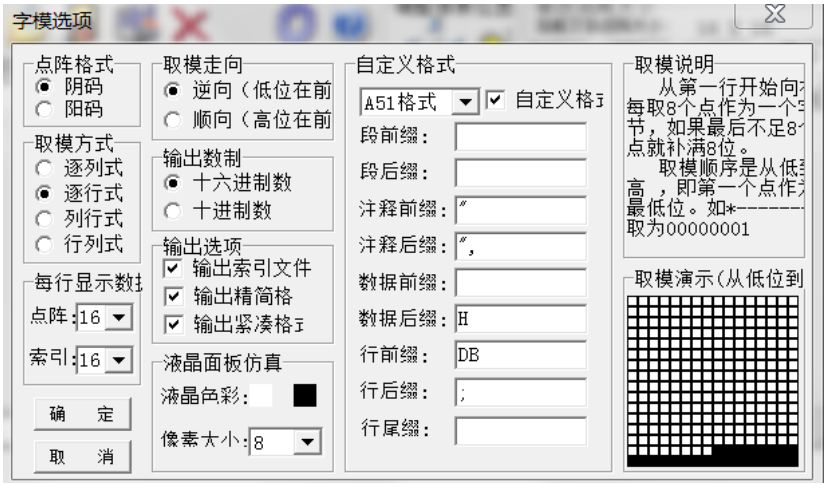


Image2Lcd 软件需要设置为水平、自左向右、自上向下、低位在前扫描方式。



PCtoLCD2002 软件需要设置位逐行式、逆向扫描方式。

1、C51、STM32 以及 Raspberry Pi 平台取模软件设置

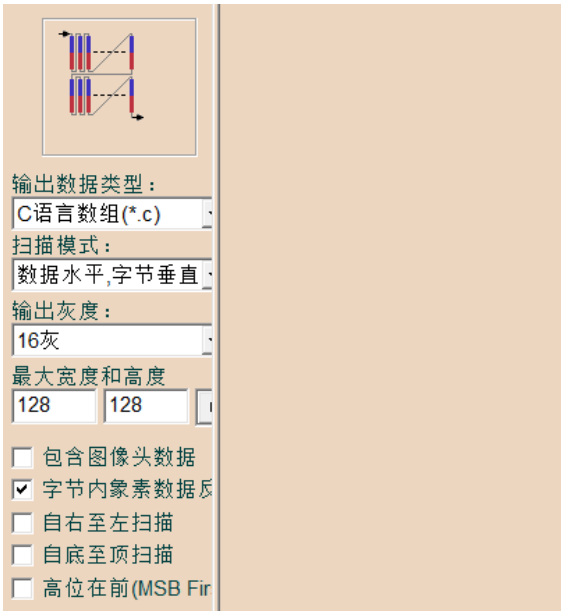
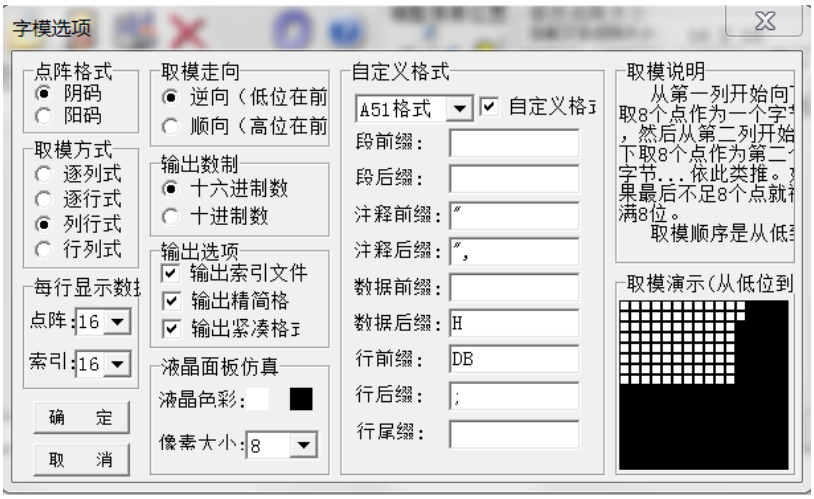


Image2Lcd 软件需要设置为数据水平，字节垂直、自左向右、自上向下、字节内低位在前扫描方式。



PCtoLCD2002 软件需要设置位列行式、逆向扫描方式。