

2.2inch SPI Module MSP2202 用户手册

产品概述

该款 LCD 模块采用 4 线制 SPI 通信方式，驱动 IC 为 ILI9341，分辨率为 240x320。该模块包含有 LCD 显示屏，背光控制电路。

产品特点

- 2.2 寸彩屏，支持 65K 色显示，显示色彩丰富
- 240X320 高清分辨率，显示清晰
- 采用 SPI 串行总线，只需几个 IO 即可点亮显示
- 带 SD 卡槽方便扩展实验
- 提供丰富的示例程序
- 军工级工艺标准,长期稳定工作
- 提供底层驱动技术支持

产品参数

名称	描述
显示颜色	65K 彩色
SKU	MSP2202
尺寸	2.2(inch)
类型	TFT
驱动芯片	ILI9341
分辨率	320*240 (Pixel)
模块接口	4-wire SPI interface
有效显示区域	33.84x45.12 (mm)
模块尺寸	40.10x67.20 (mm)
视角	>60°
工作温度	-20℃~70℃
存储温度	-30℃~80℃

工作电压	3.3V / 5V
功耗	约为 90mA
产品重量	25(g)

接口说明

标号	PIN	引脚说明
1	VCC	LCD 电源正(3.3V~5V)
2	GND	LCD 电源地
3	CS	LCD 片选信号
4	RESET	LCD 复位信号
5	DC/RS	LCD 寄存器/数据选择信号
6	SDI(MOSI)	LCD SPI 总线写数据信号
7	SCK	LCD SPI 总线时钟信号
8	LED	背光控制信号（高电平点亮，如不需要控制，请接 3.3V）
9	SDO(MISO)	LCD SPI 总线读数据信号（如果不需要，可以不接）

硬件配置

该 LCD 模块硬件电路包含两大部分：LCD 显示控制电路、背光控制电路。

LCD 显示控制电路用于控制 LCD 的引脚，包括控制引脚和数据传输引脚。

背光控制电路用于控制背光亮和灭，当然如果不需要控制背光，可以不使用该电路，直接将背光控制引脚接到 3.3V 电源上。

另外 STC12C5A60S2 的硬件 SPI 功能需要进行电平转换（3.3V 转 5V）才能正常工作，所以引脚需要外一个电平转换模块。

工作原理

1、ITI9341 控制器简介

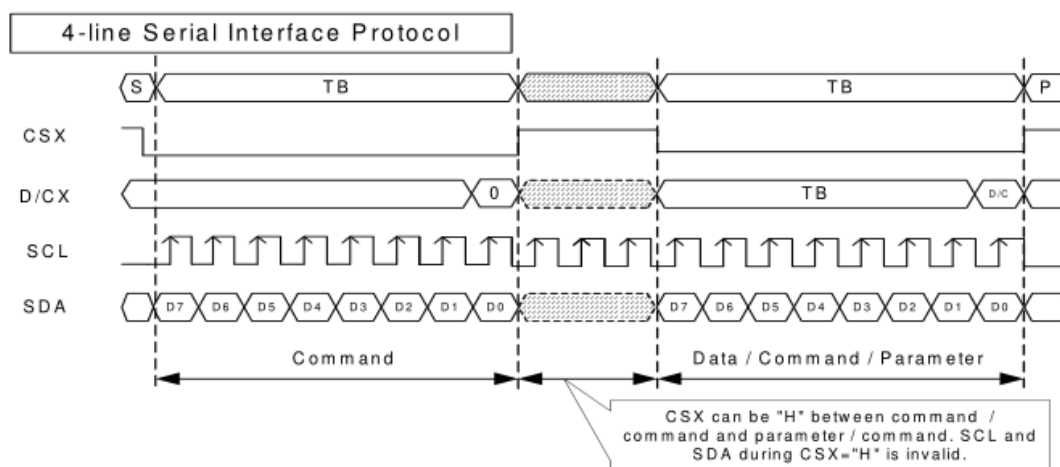
ITI9341 控制器支持的最大分辨率为 240*320，拥有一个 172800 字节大小的 GRAM。同时

支持 8 位、9 位、16 位、18 位并口数据总线，还支持 3 线制和 4 线制 SPI 串口。由于并行控制需要大量的 IO 口，所以最常用的还是 SPI 串口控制。ITI9341 还支持 65K、262K RGB 颜色显示，显示色彩很丰富，同时支持旋转显示和滚动显示以及视频播放，显示方式多样。

ITI9341 控制器使用 16bit (RGB565) 来控制一个像素点显示，因此可以每个像素点显示颜色多达 65K 种。像素点地址设置按照行列的顺序进行，递增递减方向由扫描方式决定。ITI9341 显示方法按照先设置地址再设置颜色值进行。

2、SPI 通信协议简介

4 线制 SPI 总线写模式时序如下图所示：

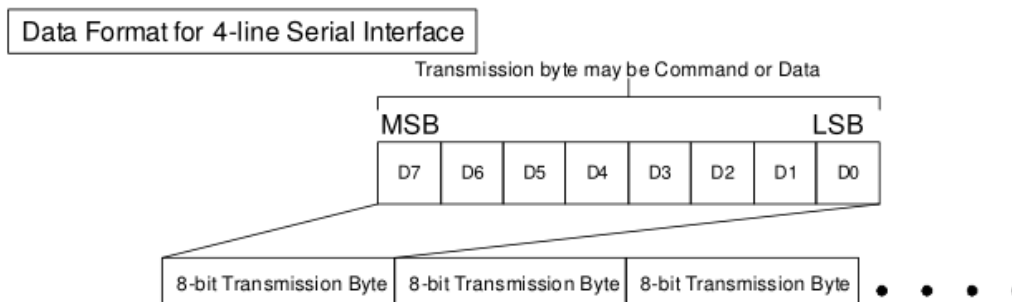


CSX 为从机片选，仅当 CSX 为低电平时，芯片才会被使能。

D/CX 为芯片的数据/命令控制引脚，当 DCX 为低电平时写命令，为高电平时写数据

SCL 为 SPI 总线时钟，每个上升沿传输 1bit 数据；

SDA 为 SPI 传输的数据，一次传输 8bit 数据，数据格式如下图所示：



高位在前，先传输。

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

议没有很多的影响；

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

使用说明

1、C51 使用说明

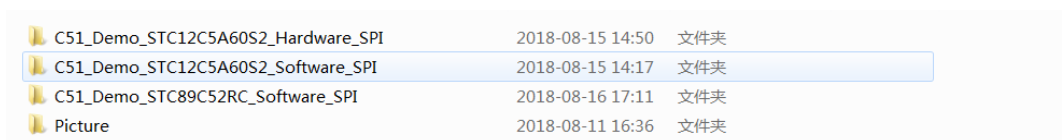
接线说明：

STC89C52RC和STC12C5A60S2单片机测试程序接线说明			
序号	引脚丝印	对应STC89/STC12开发板接线	备注
1	VCC	5V/3.3V	液晶屏电源正(3.3V~5V)
2	GND	GND	液晶屏电源地
3	CS	P13	液晶屏片选控制信号
4	RESET	P33	液晶屏复位控制信号
5	DC/RS	P12	液晶屏寄存器/数据选择控制信号
6	SDI (MOSI)	P15	液晶屏SPI总线写数据信号
7	SCK	P17	液晶屏SPI总线时钟信号
8	LED	P32	液晶屏背光控制信号（高电平点亮，如不需要控制或者使用STC89C52RC，请接3.3V）
9	SDO (MISO)	P16	液晶屏SPI总线读数据信号（如果不需要，可以不接）

操作步骤：

A、按照上述接线说明将 LCD 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、LCD 模块如果正常显示字符和图形，则说明程序运行成功；

3、STM32 使用说明

接线说明：

由于不同的开发板引脚位置不一样，而且预留外接的引脚也不一样（有些开发板没有将需要的引脚外接），为了方便接线，所以每种开发板的接线引脚不一致。

STM32F103RCT6单片机测试程序接线说明			
序号	引脚丝印	对应MiniSTM32开发板接线	备注
1	VCC	5V/3.3V	液晶屏电源正(3.3V~5V)
2	GND	GND	液晶屏电源地
3	CS	PB11	液晶屏片选控制信号
4	RESET	PB12	液晶屏复位控制信号
5	DC/RS	PB10	液晶屏寄存器/数据选择控制信号
6	SDI (MOSI)	PB15	液晶屏SPI总线写数据信号
7	SCK	PB13	液晶屏SPI总线时钟信号
8	LED	PB9	液晶屏背光控制信号（高电平点亮，如不需要控制，请接3.3V）
9	SDO (MISO)	PB14	液晶屏SPI总线读数据信号（如果不需要，可以不接）

STM32F103ZET6单片机测试程序接线说明			
序号	引脚丝印	对应Elite STM32开发板接线	备注
1	VCC	5V/3.3V	液晶屏电源正(3.3V~5V)
2	GND	GND	液晶屏电源地
3	CS	PB11	液晶屏片选控制信号
4	RESET	PB12	液晶屏复位控制信号
5	DC/RS	PB10	液晶屏寄存器/数据选择控制信号
6	SDI (MOSI)	PB15	液晶屏SPI总线写数据信号
7	SCK	PB13	液晶屏SPI总线时钟信号
8	LED	PB9	液晶屏背光控制信号（高电平点亮，如不需要控制，请接3.3V）
9	SDO (MISO)	PB14	液晶屏SPI总线读数据信号（如果不需要，可以不接）

STM32F407ZGT6单片机测试程序接线说明

序号	引脚丝印	对应Explorer STM32F4开发板接线	备注
1	VCC	5V/3.3V	液晶屏电源正(3.3V~5V)
2	GND	GND	液晶屏电源地
3	CS	PB15	液晶屏片选控制信号
4	RESET	PB12	液晶屏复位控制信号
5	DC/RS	PB14	液晶屏寄存器/数据选择控制信号
6	SDI (MOSI)	PB5	液晶屏SPI总线写数据信号
7	SCK	PB3	液晶屏SPI总线时钟信号
8	LED	PB13	液晶屏背光控制信号（高电平点亮，如不需要控制，请接3.3V）
9	SDO (MISO)	PB4	液晶屏SPI总线读数据信号（如果不需要，可以不接）

STM32F429IGT6单片机测试程序接线说明

序号	引脚丝印	对应Apollo STM32F4/F7开发板接线	备注
1	VCC	5V/3.3V	液晶屏电源正(3.3V~5V)
2	GND	GND	液晶屏电源地
3	CS	PD11	液晶屏片选控制信号
4	RESET	PD12	液晶屏复位控制信号
5	DC/RS	PD5	液晶屏寄存器/数据选择控制信号
6	SDI (MOSI)	PF9	液晶屏SPI总线写数据信号
7	SCK	PF7	液晶屏SPI总线时钟信号
8	LED	PD6	液晶屏背光控制信号（高电平点亮，如不需要控制，请接3.3V）
9	SDO (MISO)	PF8	液晶屏SPI总线读数据信号（如果不需要，可以不接）

操作说明：

- A、按照上述接线说明将 LCD 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：

Demo_STM32F103RCT6_Hardware_SPI	2018-08-06 17:13	文件夹
Demo_STM32F103RCT6_Software_SPI	2018-08-06 19:20	文件夹
Demo_STM32F103ZET6_Hardware_SPI	2018-08-10 13:41	文件夹
Demo_STM32F103ZET6_Software_SPI	2018-08-10 13:41	文件夹
Demo_STM32F407ZGT6_Hardware_SPI	2018-08-09 10:20	文件夹
Demo_STM32F407ZGT6_Software_SPI	2018-08-09 10:20	文件夹
Demo_STM32F429IGT6_Hardware_SPI	2018-08-08 19:41	文件夹
Demo_STM32F429IGT6_Software_SPI	2018-08-09 10:56	文件夹
Picture	2018-08-11 16:36	文件夹

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

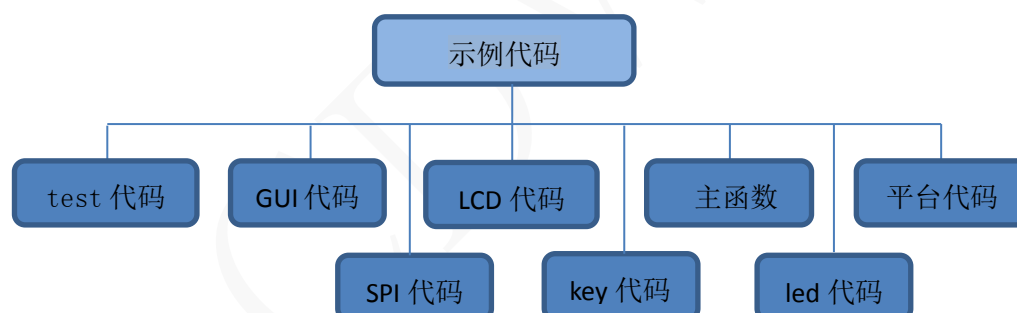
STM32_Keil_Use_Illustration_CN.pdf）；

D、LCD 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、C51 和 STM32 测试示例代码架构如下：



主程序运行时的 Demo API 代码包含在 test 代码中；

LCD 初始化以及相关的操作都包含在 LCD 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；

SPI 初始化及配置相关的操作包含在 SPI 代码中；

按键处理相关的代码都包含在 key 代码中（C51 平台没有按键处理代码）；

led 配置操作相关的代码都包含在 led 代码中；

2、软件 SPI 和硬件 SPI 说明

该 LCD 模块分别提供了软件 SPI 和硬件 SPI 示例代码（STC89C52RC 除外，因为其没有硬件 SPI 功能），两台示例代码在显示内容上没有任何区别，但是如下方面有区别：

A、显示速度

硬件 SPI 明显比软件 SPI 要快，这是由硬件决定的。

B、GPIO 定软件

软件 SPI 全部控制引脚都要定义，可以使用任何空闲引脚，硬件 SPI 的数据和时钟信号引脚是固定的（因平台而异），其他控制引脚要自己定义，也可以使用任何空闲引脚。

C、初始化

软件 SPI 初始化时，只需要对用于引脚定义的 GPIO 进行初始化（C51 平台不需要），

硬件 SPI 初始化时，需要对相关的控制寄存器以及数据寄存器进行初始化。

3、模块 GPIO 定义修改

A、C51 的液晶屏相关的 GPIO 定义放在 lcd.h 文件里面，如下图所示：

```
//IO连接
sbit LCD_RS = P1^2;      //数据/命令切换
sbit LCD_SDI = P1^5;      //SPI写
sbit LCD_SDO = P1^6;      //SPI读
sbit LCD_CS = P1^3;      //片选
sbit LCD_CLK = P1^7;      //SPI时钟
sbit LCD_RESET = P3^3;    //复位
sbit LCD_BL=P3^2;        //背光控制，如果不需要控制，接3.3V
```

如果使用软件 SPI，所有引脚定义都可以修改，可以定义成其他任何空闲的 GPIO。

如果使用硬件 SPI，LCD_BL、LCD_RS、LCD_CS 以及 LCD_RST 引脚定义可以修改，可以定义成其他任何空闲的 GPIO。LCD_CLK 和 LCD_SDI 不需要定义。

B、STM32 的液晶屏非 SPI 的 GPIO 定义放在 lcd.h 里面，如下图所示：

```
#define LED      13      //背光控制引脚
#define CS       15      //片选引脚
#define RS       14      //寄存器/数据选择引脚
#define RST      12      //复位引脚
```

所有引脚定义都可以修改，可以定义成其他任何空闲的 GPIO。

STM32 的液晶屏 SPI 的 GPIO 定义放在 spi.h 里面，如下图所示：

```
#define SCLK          3 //PB13--->>TFT --SCL/SCK
#define MISO          4
#define MOSI          5 //PB15 MOSI--->>TFT --SDA/DIN
```

如果使用软件 SPI，所有引脚定义都可以修改，可以定义成其他任何空闲的 GPIO。

如果使用硬件 SPI，这些引脚都不需要定义，同时需要在 lcd.c 文件里面的 LCD_GPIOInit 函数里将 SCLK、MISO 以及 MOSI 引脚初始化去掉，如下图所示：

```
void LCD_GPIOInit(void)
{
    GPIO_InitTypeDef GPIO_InitStructure;

    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOB ,ENABLE);

    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_3|GPIO_Pin_5|GPIO_Pin_12| GPIO_Pin_13|GPIO_Pin_14| GPIO_Pin_15;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT; //普通输出模式
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
    GPIO_InitStructure.GPIO_OType = GPIO_OType_PP; //推挽输出
    GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP; //上拉
    GPIO_Init(GPIOB, &GPIO_InitStructure);

    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_4;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN;
    GPIO_Init(GPIOB, &GPIO_InitStructure);
}
```

红圈里面的内容都要去掉。

4、SPI 通信代码实现

A、C51 和 STM32 的硬件 SPI 通信都是由系统实现好的，我们只需要操作寄存器，调用数据发生函数就可以了，具体说明请查阅 MCU 相关的说明文档。

B、C51 和 STM32 的软件 SPI 通信代码分别在 lcd.c 和 spi.c 中实现，软件 SPI 实现方法一样，如下图所示：

```
void SPIv_WriteData(u8 Data)
{
    unsigned char i=0;
    for(i=8;i>0;i--)
    {
        if(Data&0x80)
            LCD_SDA_SET; //输出数据
        else
            LCD_SDA_CLR;
        LCD_SCL_CLR;
        LCD_SCL_SET;
        Data<<=1;
    }
}
```

传输的数据位为 1，则将 SPI 数据引脚拉高，为 0，则将 SPI 数据引脚拉低，每次传输一个字节数据，高位在前，每个时钟上升沿传输 1 位数据。

STM32 和 C51 的硬件 SPI 的通信实现方法是一样的，都是通过操作 SPI 控制寄存器和数据寄存器，具体可以查看平台相关的示例。

常用软件

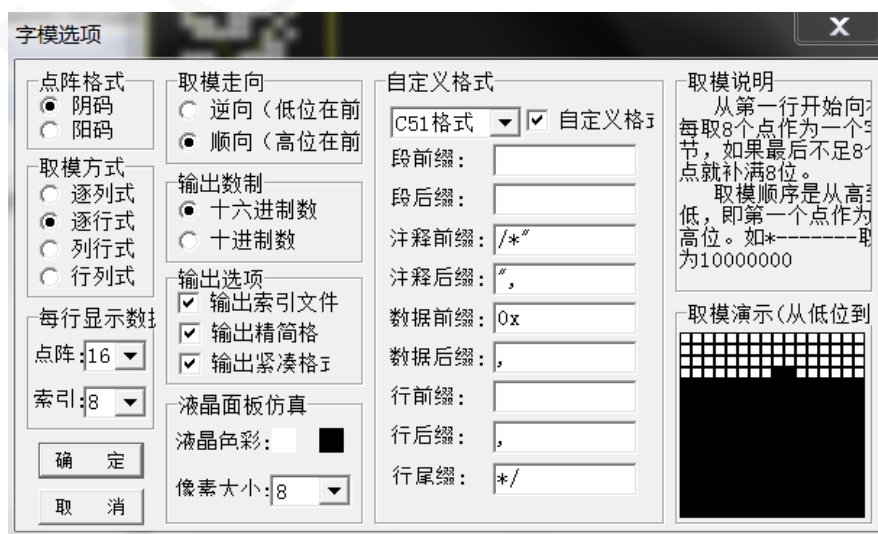
本套测试示例需要显示中英文、符号以及图片，所以要用到取模软件。取模软件有两种：Image2Lcd 和 PCtoLCD2002。这里只针对该套测试程序说明一下取模软件的设置。（具体使用方法见 PCtoLCD2002_Use_Illustration_CN.pdf、Image2Lcd_Use_Illustration_CN.pdf）

1、Image2Lcd 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。

2、PCtoLCD2002 取模软件设置



PCtoLCD2002 软件需要设置位逐行式、顺向扫描方式。