1.28inch 4-line-SPI IPS Module MSP1281 用户手册

产品概述

该产品为一款 1.28 寸圆形 IPS 显示模块，拥有 240x240 分辨率。采用 4 线制 SPI 通信方式，内部驱动 IC 为 GC9A01。该模块包含有 IPS 显示屏和 PCB 底板。

产品特点

- 1.28 寸圆形 IPS 彩屏，支持 RGB 65K 色显示，显示色彩丰富
- 240X240 分辨率，显示清晰
- 超大可视角度（全角度），显示颜色不失真
- 采用 4 线制 SPI 串行总线，只需几个 IO 即可点亮显示
- 提供丰富的 Arduino、STM32、C51 以及 MSP430 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	RGB 65K 彩色
SKU	MSP1281
尺寸	1.28(inch)
类型	TFT
驱动芯片	GC9A01
分辨率	240*240 (Pixel)
模块接口	4-line SPI interface
有效显示区域	直径：32.40 (mm)
触摸屏类型	无触摸屏
触摸 IC	无触摸 IC
模块尺寸	38.03x45.80(mm)
视角	全角度

工作温度	-10℃~60℃
存储温度	-20℃~70℃
工作电压	3.3V
功耗	待定
产品重量（含包装）	11(g)

接口说明

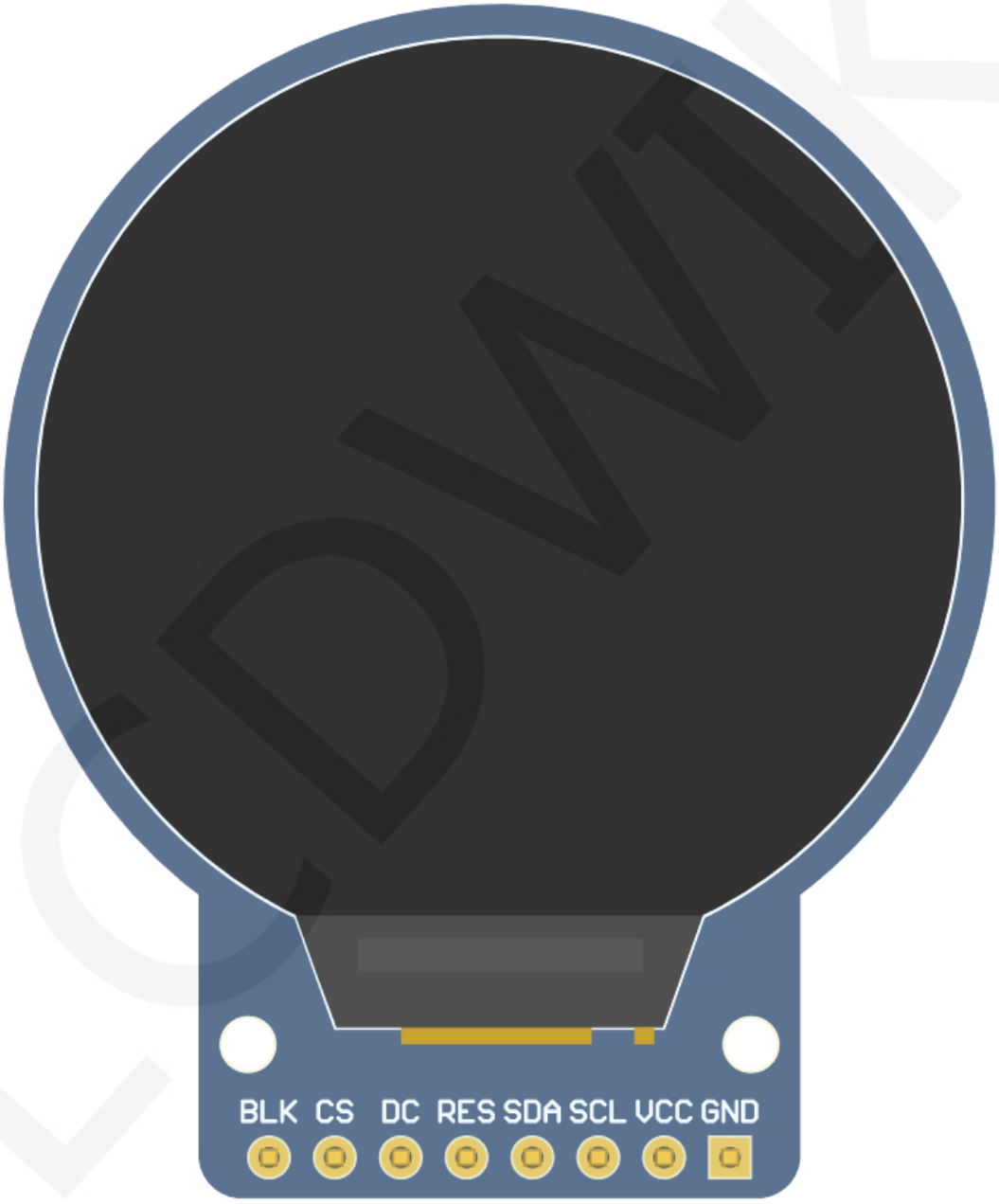


图1. 模块引脚丝印图

重要说明：

1. 以下引脚序号1~8是指我司带PCB底板的模块引脚编号，如果您购买的是裸屏，请参考裸屏规格书的引脚定义，按照信号类型来参考接线而不是直接根据下面的模块引脚编号来接线，举例：DC在我们模块上是6脚，可能在不同尺寸裸屏上是x脚。
2. 关于VCC供电电压：该IPS显示模块可接3.3V或者5V。
3. 关于背光电压：带PCB底板的模块均已集成三极管背光控制电路，只需BL引脚输入高电平或者PWM波则背光点亮。如果您购买的是裸屏，则LEDAx接3.0V-3.3V，LEDKx接地即可。

标号	模块引脚	引脚说明
1	GND	液晶屏电源地
2	VCC	液晶屏电源正(3.3V/5V)
3	SCL	液晶屏SPI总线时钟信号
4	SDA	液晶屏SPI总线写数据信号
5	RES	液晶屏复位控制信号（低电平复位，模块有复位电路，该引脚可不接）
6	DC	液晶屏寄存器/数据选择控制信号 （低电平：寄存器，高电平：数据）
7	CS	液晶屏片选控制信号（低电平使能）
8	BLK	液晶屏背光控制信号（高电平点亮，如不需要控制，请接3.3V）

硬件配置

该 LCD 模块硬件电路包含五大部分：FPC 接口电路、3.3V 稳压电路、背光控制电路、排针接口、复位电路。

FPC 接口电路用于连接裸屏。

3.3V 稳压电路用于稳定输出 3.3V 电压。

背光控制电路用于用于控制背光亮和灭，如果不需控制背光。可接 3.3V 电源。

排针接口用于连接各种开发板。

复位电路用于模块上电复位。

工作原理

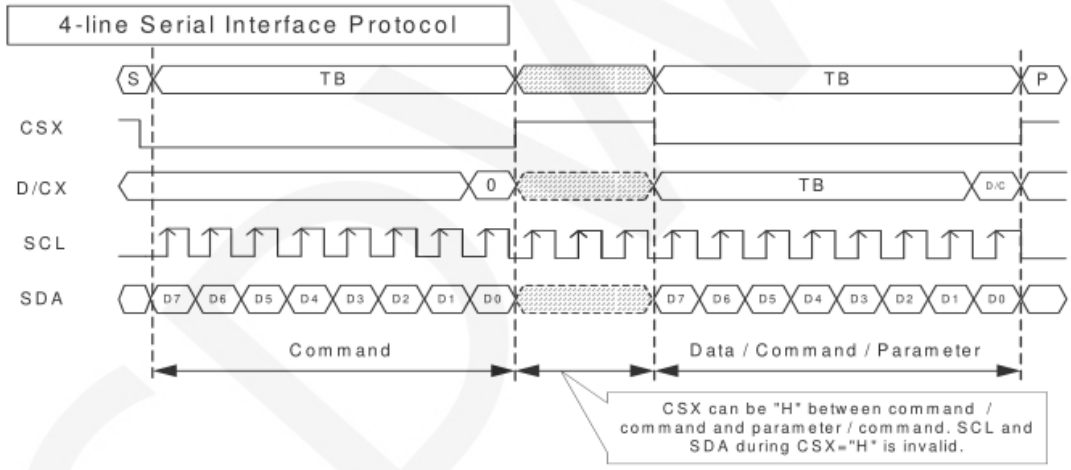
1、GC9A01 控制器简介

GC9A01 控制器支持的最大分辨率为 240*240,拥有一个 129600 字节大小的 GRAM。同时支持 8 位、9 位、12 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。ST7789 还支持 65K、262K RGB 颜色显示，显示色彩很丰富，同时支持旋转和滚动显示以及视频播放，显示方式多样。

GC9A01 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。GC9A01 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：

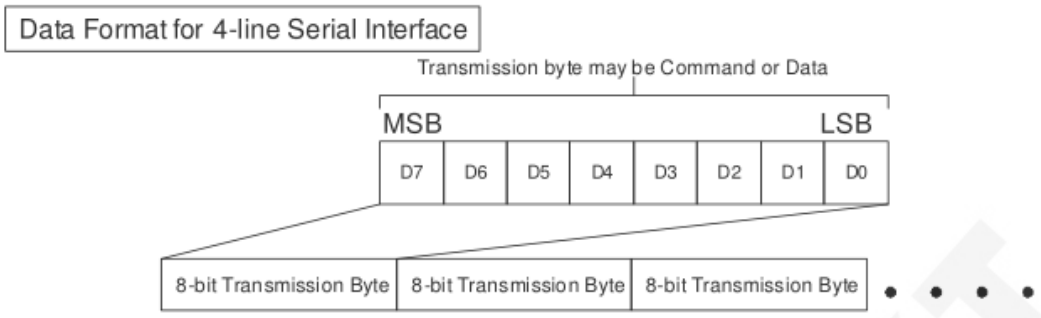


CSX 为从机片选， 仅当 CSX 为低电平时，芯片才会被使能。

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、STM32 使用说明

接线说明：

引脚标注见接口说明。

STM32F103C8T6单片机测试程序接线说明		
序号	**模块引脚**	**对应MiniSTM32开发板接线引脚**
1	**GND**	GND
2	**VCC**	3.3V/5V
3	**SCL**	PA5
4	**SDA**	PA7
5	**RES**	PB8
6	**DC**	PB7
7	**CS**	PB9
8	**BLK**	PB6

STM32F103RCT6单片机测试程序接线说明		
序号	模块引脚	对应MiniSTM32开发板接线引脚
1	GND	GND
2	VCC	3.3V/5V
3	SCL	PB13
4	SDA	PB15
5	RES	PB12
6	DC	PB10
7	CS	PB11
8	BLK	PB9

STM32F103ZET6单片机测试程序接线说明		
序号	引脚丝印	对应Elite STM32开发板接线引脚
1	GND	GND
2	VCC	3.3V/5V
3	SCL	PB13
4	SDA	PB15
5	RES	PB12
6	DC	PB10
7	CS	PB11
8	BLK	PB9

STM32F407ZGT6单片机测试程序接线说明		
序号	引脚丝印	对应Explorer STM32F4开发板接线引脚

1	GND	GND
2	VCC	3.3V/5V
3	SCL	PB3
4	SDA	PB5
5	RES	PB12
6	DC	PB14
7	CS	PB15
8	BLK	PB13

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
1	GND	GND
2	VCC	3.3V/5V
3	SCL	PF7
4	SDA	PF9
5	RES	PD12
6	DC	PD5
7	CS	PD11
8	BLK	PD6

STM32F767IGT6和STM32H743IIT6单片机测试程序接线说明		
序号	模块引脚	对应Apollo STM32F4/F7开发板接线
1	GND	GND
2	VCC	3.3V/5V

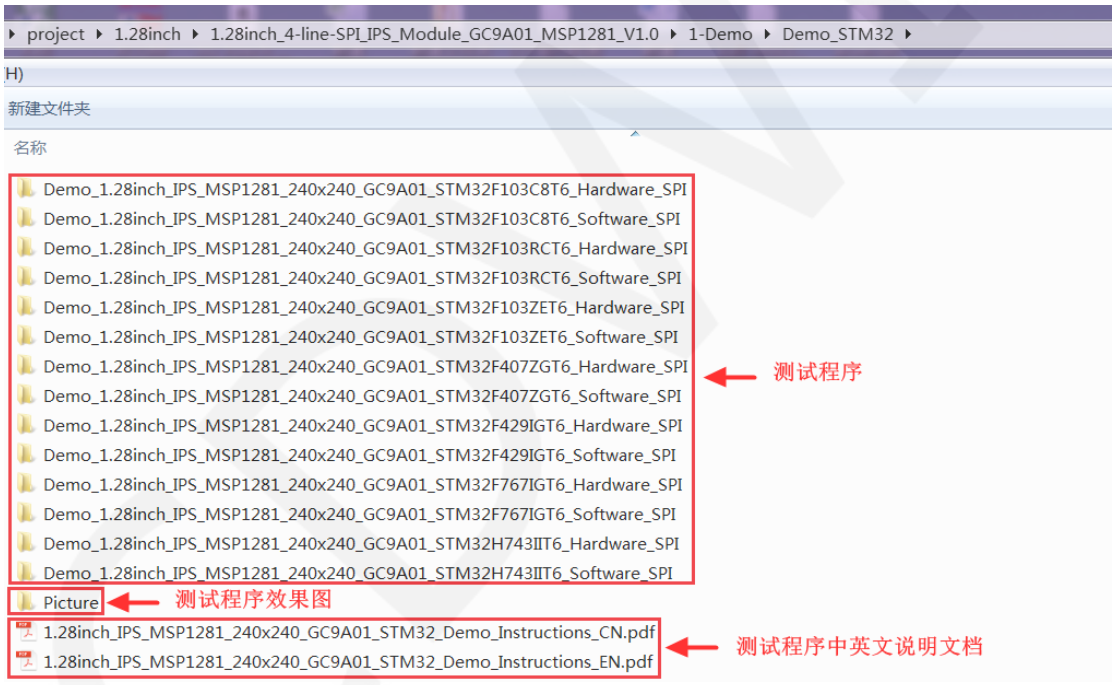
3	**SCL**	PB13
4	**SDA**	PB15
5	**RES**	PD12
6	**DC**	PD5
7	**CS**	PD11
8	**BLK**	PD6

操作步骤：

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

B、 根据单片机型号选择测试示例，如下图所示：

（测试程序说明请查阅测试程序包中测试程序说明文档）



C、打开所选的测试程序工程，进行编译和下载；

关于 STM32 测试程序编译和下载的详细说明见如下文档：

http://www.lcdwiki.com/res/PublicFile/STM32_Keil_Use_Illustration_CN.pdf

D、IPS 模块如果正常显示字符和图形，则说明程序运行成功；

2、C51 使用说明

接线说明：

引脚标注见接口说明。

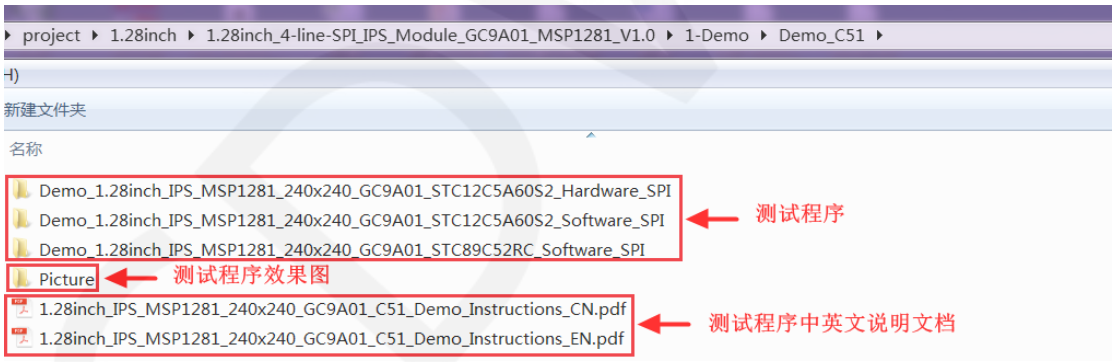
STC89C52RC和STC12C5A60S2单片机测试程序接线说明		
序号	模块引脚	对应STC89/STC12开发板接线引脚
1	GND	GND
2	VCC	3.3V/5V
3	SCL	P17
4	SDA	P15
5	RES	P33
6	DC	P12
7	CS	P13
8	BLK	P32

操作步骤：

A、 按照上述接线说明将 IPS 模块和 C51 单片机连接起来，并上电；

B、 选择需要测试的 C51 测试程序，如下图所示：

（测试程序说明请查阅测试程序包中测试程序说明文档）



C、打开所选的测试程序工程，进行编译和下载；

关于 C51 测试程序编译和下载的详细说明见如下文档：

http://www.lcdwiki.com/res/PublicFile/C51_Keil%26stc-isp_Use_Illustration_CN.pdf

D、IPS 模块如果正常显示字符和图形，则说明程序运行成功；

3、Arduino 使用说明

接线说明：

Arduino UNO单片机测试程序接线说明		
序号	模块引脚	对应UNO开发板接线引脚
1	GND	GND
2	VCC	3.3V/5V
3	SCL	13
4	SDA	11
5	RES	A4
6	DC	A3
7	CS	A2
8	BLK	A0

Arduino Mega2560单片机测试程序接线说明		
序号	模块引脚	对应UNO开发板接线引脚
1	GND	GND
2	VCC	3.3V/5V
3	SCL	52
4	SDA	51
5	RES	A4
6	DC	A3
7	CS	A2
8	BLK	A0

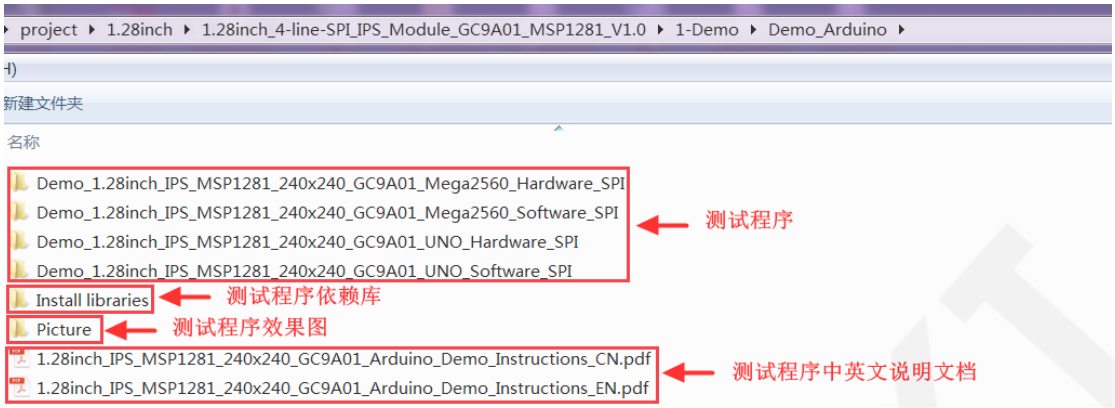
操作步骤：

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

B、将测试程序目录中 Install libraries 目录下的依赖库拷贝到 Arduino 工程目录的 libraries 文件夹下（默认为 C:\Users\Administrator\Documents\Arduino\libraries 如果不需要依赖库，则不需要拷贝）；

C、打开 Arduino 测试程序所在目录，选择需要测试的示例，如下图所示：

（测试程序说明请查阅测试程序包中测试程序说明文档）



D、打开所选的示例工程，进行编译和下载。

关于 Arduino 测试程序依赖库拷贝、编译和下载的具体操作方法见如下文档：

http://www.lcdwiki.com/res/PublicFile/Arduino_IDE_Use_Illustration_CN.pdf

E、LCD 模块如果正常显示字符和图形，则说明程序运行成功；

4、MSP430 使用说明

接线说明：

引脚标注见接口说明。

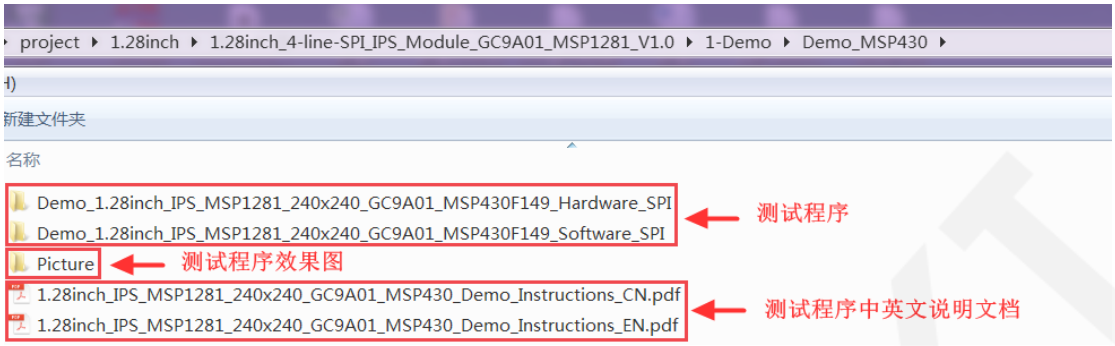
MSP430F149单片机测试程序接线说明		
序号	**模块引脚**	**对应MSP430开发板接线引脚**
1	**GND**	GND
2	**VCC**	3.3V/5V
3	**SCL**	P33
4	**SDA**	P31
5	**RES**	P22
6	**DC**	P21
7	**CS**	P23
8	**BLK**	P20

操作步骤：

A、 按照上述接线说明将 IPS 模块和 MSP430 单片机连接起来，并上电；

B、 选择需要测试的 MSP430 测试程序，如下图所示：

（测试程序说明请查阅测试程序包中测试程序说明文档）



C、打开所选的测试程序工程，进行编译和下载；

关于 MSP430 测试程序编译和下载的详细说明见如下文档：

http://www.lcdwiki.com/res/PublicFile/IAR_IDE%26MspFet_Use_Illustration_CN.pdf

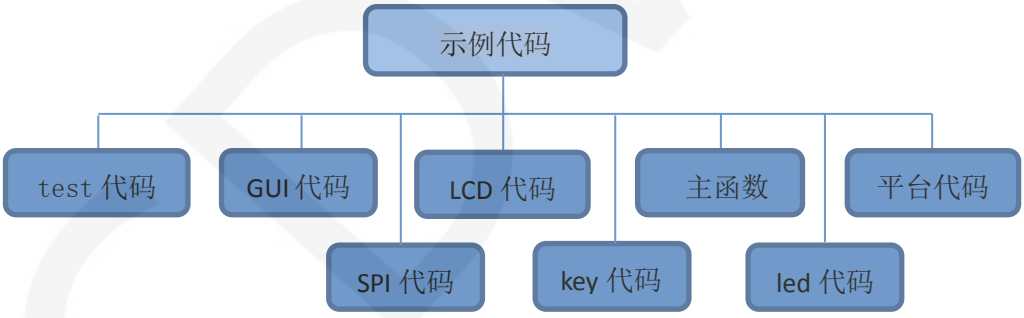
D、IPS 模块如果正常显示字符和图形，则说明程序运行成功；

软件说明

1、代码架构

A、C51、STM32 以及 MSP430 代码架构说明

代码架构如下图所示：



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；

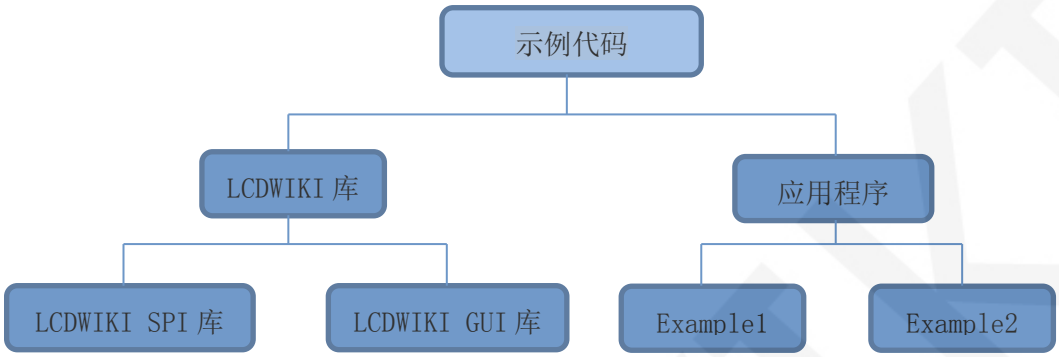
SPI 初始化及配置相关的操作包含在 SPI 代码中；

按键处理相关的代码都包含在 key 代码中(C51 和 MSP430 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（MSP430 平台没有 LED 代码）；

B、Arduino 代码架构说明

代码架构如下图所示：



Arduino 的测试程序代码由两部分组成：LCDWIKI 库和应用代码。

LCDWIKI 库包含两部分内容：LCDWIKI_SPI 库和 LCDWIKI_GUI 库。

应用程序包含几个测试示例，每个测试示例包含不同的测试内容。

LCDWIKI_SPI 为底层库，和硬件有关联，主要负责操作寄存器，包括硬件模块初始化，数据和命令传输，像素点坐标和颜色设置，显示方式配置等。

LCDWIKI_GUI 为中间层库，负责使用底层库提供的 API 实现图形的绘制，字符显示。

应用程序是利用 LCDWIKI 库提供的 API，编写一些测试示例，实现某方面的测试功能。

2、软件 SPI 和硬件 SPI 说明

该 IPS 模块分别提供了软件 SPI 和硬件 SPI 示例代码（STC89C52RC 只有软件 SPI 功能），两种示例代码在显示内容上没任何区别，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 定义

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，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 定义说明

A、STM32 测试程序 GPIO 定义说明

非 SPI 的 GPIO 定义放在 lcd.h 里面，如下图所示(以 STM32F103RCT6 单片机测试程序为例)：

```
//////////////////////////////////////////////////////////////////
//-----------------LCD端口定义----------------
#define GPIO_TYPE  GPIOB  //GPIO组类型
#define LED        9      //背光控制引脚        PB9
#define LCD_CS     11     //片选引脚            PB11
#define LCD_RS     10     //寄存器/数据选择引脚 PB10
#define LCD_RST    12     //复位引脚            PB12
```

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 测试程序，则不需要定义 SPI 的 GPIO。

如果使用软件 SPI 测试程序，则 SPI 的 GPIO 定义放在 spi.h 里面，如下图所示(以 STM32F103RCT6 单片机测试程序为例)：

```
#define LCD_CTRL        GPIOB    //定义TFT数据端口
#define SPI_SCLK        GPIO_Pin_13 //PB13--->>TFT --SCL/SCK
#define SPI_MISO        GPIO_Pin_14
#define SPI_MOSI        GPIO_Pin_15 //PB15 MOSI--->>TFT --SDA/DIN
```

所有引脚定义都可以修改，可以定义成其他任何空闲的 GPIO。

B、C51 测试程序 GPIO 定义说明

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。

C、Arduino 测试程序 GPIO 定义说明

Arduino 测试程序的液晶屏和触摸屏 GPIO 定义都单独放在每个应用程序里，也就是说每个应用程序可以根据需求灵活定义 GPIO。如下图所示（以 UNO 软件 SPI 测试程序为

例）：

```
//paramters define
#define MODEL ST7789
#define CS   A2
#define CD   A3
#define RST  A4
#define SDA  11
#define SCK  13
#define LED  A0   //if you don't need to control
```

D、MSP430 测试程序 GPIO 定义说明

非 SPI 的 GPIO 定义放在 lcd.h 里面，如下图所示：

```
//////////////////////////////////////////////////////////////////
//-----------------LCD端口定义----------------
#define LED      BIT0        //背光控制引脚   P20
#define LCD_RS   BIT1        //寄存器/数据选择引脚 P21
#define LCD_RST  BIT2        //复位引脚          P22
#define LCD_CS   BIT3        //片选引脚          P23
```

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 测试程序，则不需要定义 SPI 的 GPIO。

如果使用软件 SPI 测试程序，则 SPI 的 GPIO 定义放在 spi.h 里面，如下图所示：

```
//本测试程序使用的是软件SPI接口驱动
//SPI时钟信号以及SPI读、写信号引脚都可以更改

#define SPI_SCLK BIT3    //P33
#define SPI_MOSI BIT1    //P31
```

所有引脚定义都可以修改，可以定义成其他任何空闲的 GPIO。

4、SPI 通信代码实现

硬件 SPI 通信都是由系统实现好的，我们只需要操作寄存器，调用相关函数就可以了，具体说明请查阅 MCU 相关的说明文档。

A、STM32 测试程序 SPI 通信代码实现

软件 SPI 通信代码在 spi.c 中实现，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。

B、C51 测试程序 SPI 通信代码实现

软件 SPI 通信代码在 lcd.c 中实现，如下图所示：

```
void spi_write_byte(u8 d)
{
  u8 val=0x80;
  while(val)
  {
    if(d&val)
    {
      LCD_SDI = 1;
    }
    else
    {
      LCD_SDI = 0;
    }
    LCD_CLK = 0;
    LCD_CLK = 1;
    val>>=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。

C、Arduino 测试程序 SPI 通信代码实现

硬件 SPI 通信都是由系统实现好的，我们只需要操作寄存器，调用相关函数就可以了，具体说明请查阅 MCU 相关的说明文档。

软件 SPI 通信代码在 LCDWIKI_SPI 库的 LCDWIKI_SPI.cpp 文件里实现，如下图所示：

```
//spi write for hardware and software
void LCDWIKI_SPI::Spi_Write(uint8_t data)
{
    if(hw_spi)
    {
        SPI.transfer(data);
    }
    else
    {
        uint8_t val = 0x80;
        while(val)
        {
            if(data&val)
            {
                MOSI_HIGH;
            }
            else
            {
                MOSI_LOW;
            }
            CLK_LOW;
            CLK_HIGH;
            val >>= 1;
        }
    }
} // end spi_write
```

软件spi

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。

D、MSP430 测试程序 SPI 通信代码实现

软件 SPI 通信代码在 spi.c 中实现，如下图所示：

```
/*****************************************************************************
 * @name       :void  SPI_WriteByte(u8 Data)
 * @date       :2018-08-09
 * @function   :Write a byte of data using STM32's hardware SPI
 * @parameters :SPIx: SPI type,x for 1,2,3
                Byte:Data to be written
 * @retvalue   :Data received by the bus
*****************************************************************************/
void  SPI_WriteByte(u8 Data)
{
        unsigned char i=0;
        for(i=8;i>0;i--)
        {
          if(Data&0x80)
          SPI_MOSI_SET; //输出数据
      else SPI_MOSI_CLR;

      SPI_SCLK_CLR;
      SPI_SCLK_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。

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。

PCtoLCD2002 取模软件设置如下：

点阵格式选择**阴码**

取模方式选择**逐行式**

取模走向选择**顺向（高位在前）**

输出数制选择**十六进制数**

自定义格式选择 **C51 格式**

具体设置方法见如下网页：

http://www.lcdwiki.com/zh/%E3%80%90%E6%95%99%E7%A8%8B%E3%80%91%E4%B8%AD%E8%8B%B1%E6%96%87%E6%98%BE%E7%A4%BA%E5%8F%96%E6%A8%A1%E8%AE%BE%E7%BD%AE

Image2Lcd 取模软件设置如下图所示：

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。