4.0inch SPI Module MSP4020&MSP4021 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 ST7796S，分辨率为 320x480，带有触摸功能（可选）。该模块包含有 LCD 显示屏，背光控制电路以及触摸屏控制电路。

产品特点

- 4.0 寸彩屏，支持 RGB 65K 色显示，显示色彩丰富
- 480X320 高清分辨率，可选触摸功能
- 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	RGB 65K 彩色
SKU	带触摸：MSP4021
	不带触摸：MSP4020
尺寸	4.0(inch)
类型	TFT
驱动芯片	ST7796S
分辨率	480*320 (Pixel)
模块接口	4-wire SPI interface
有效显示区域	55.68x83.52 (mm)
模块尺寸	61.74x108.04(mm)
触摸屏类型	电阻触摸屏
触摸 IC	XPT2046

工作温度	-10℃~60℃
存储温度	-20℃~70℃
工作电压	3.3V / 5V
功耗	待定
产品重量（含包装）	带触摸：71g
	不带触摸：58g

接口说明

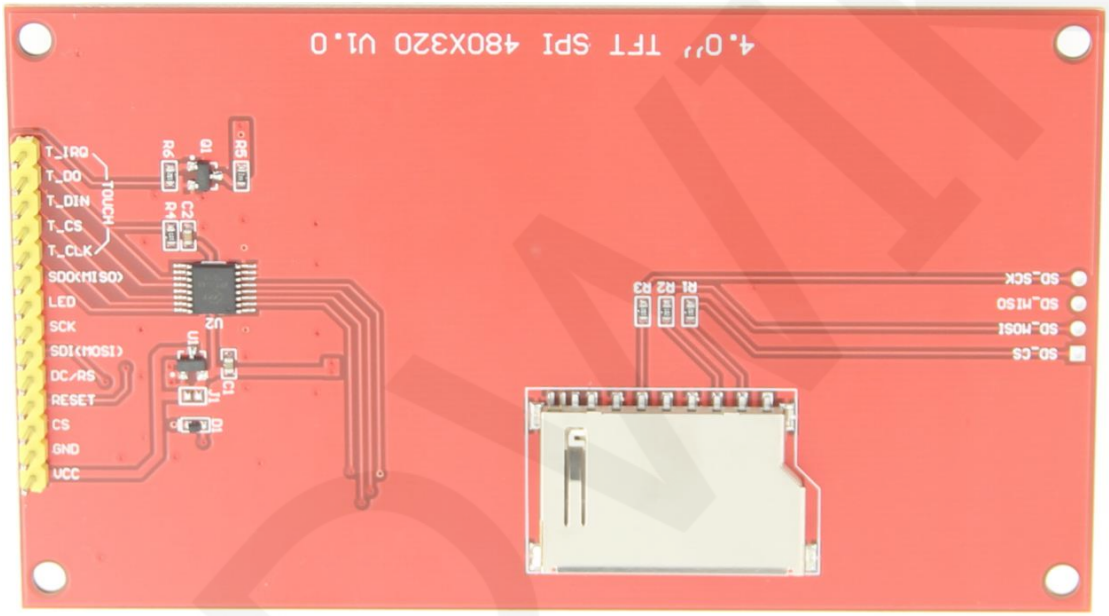


图1. 引脚丝印图

标号	模块引脚	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（低电平使能）
4	RESET	LCD 复位信号（低电平复位）
5	DC/RS	LCD 命令/数据选择信号（高电平：数据，低电平：命令）
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）
以下为触摸屏引脚，如果不带触摸或者不需要触摸功能，可以不接		
10	T_CLK	触摸屏 SPI 总线时钟信号
11	T_CS	触摸屏片选信号（低电平使能）
12	T_DIN	触摸屏 SPI 总线输入信号
13	T_DO	触摸屏 SPI 总线输出信号
14	T_IRQ	触摸屏触摸中断信号（检测到触摸时为低电平）

硬件配置

该 LCD 模块硬件电路包含三大部分：LCD 显示控制电路、触摸屏控制电路以及背光控制电路。

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。

触摸屏控制电路可控制触摸屏触摸相应以及触摸坐标读取（触摸屏可选）。

背光控制电路用于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。

工作原理

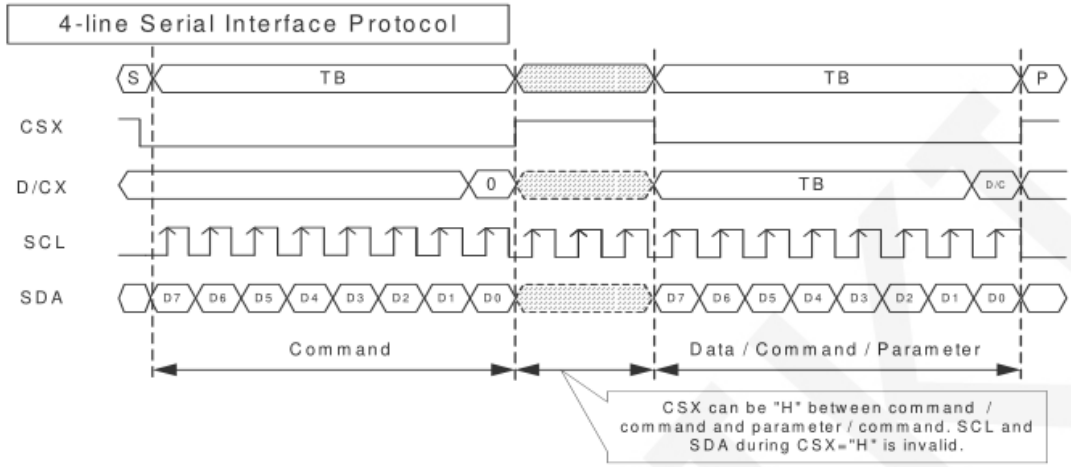
1、ST7796S 控制器简介

ST7796S 控制器支持的最大分辨率为 320*480，拥有一个 345600 字节大小的 GRAM。同时支持 8 位、9 位、16 位、18 位以及 24 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。ST7796S 还支持 65K、262K、16.7M 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。

ST7796S 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。ST7796S 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：

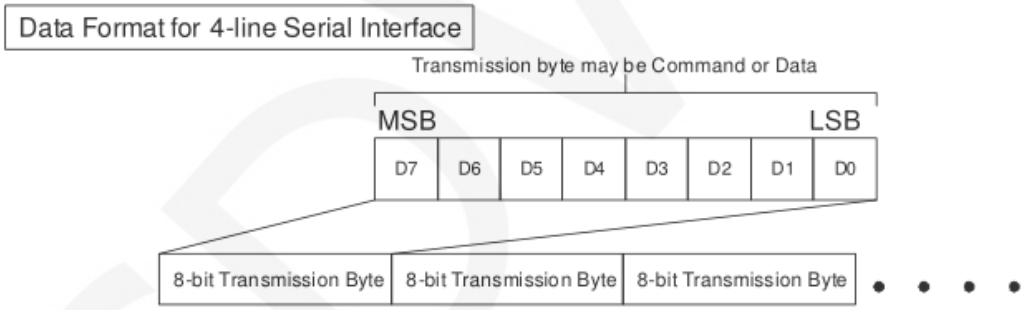


CSX 为从机片选， 仅当 CSX 为低电平时，芯片才会被使能。

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、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	SDO(MISO)	12
2	LED	A0
3	SCK	13
4	SDI(MOSI)	11
5	DC/RS	A3
6	RESET	A4
7	CS	A5
8	GND	GND
9	VCC	5V/3.3V
10	T_IRQ	6
11	T_DO	4
12	T_DIN	5
13	T_CS	2
14	T_CLK	3

Arduino MEGA2560单片机测试程序接线说明		
序号	模块引脚	对应MEGA2560开发板接线引脚
1	SDO(MISO)	50
2	LED	A0
3	SCK	52
4	SDI(MOSI)	51
5	DC/RS	A3
6	RESET	A4
7	CS	A5

8	**GND**	GND
9	**VCC**	5V/3.3V
10	**T_IRQ**	49
11	**T_DO**	47
12	**T_DIN**	48
13	**T_CS**	45
14	**T_CLK**	46

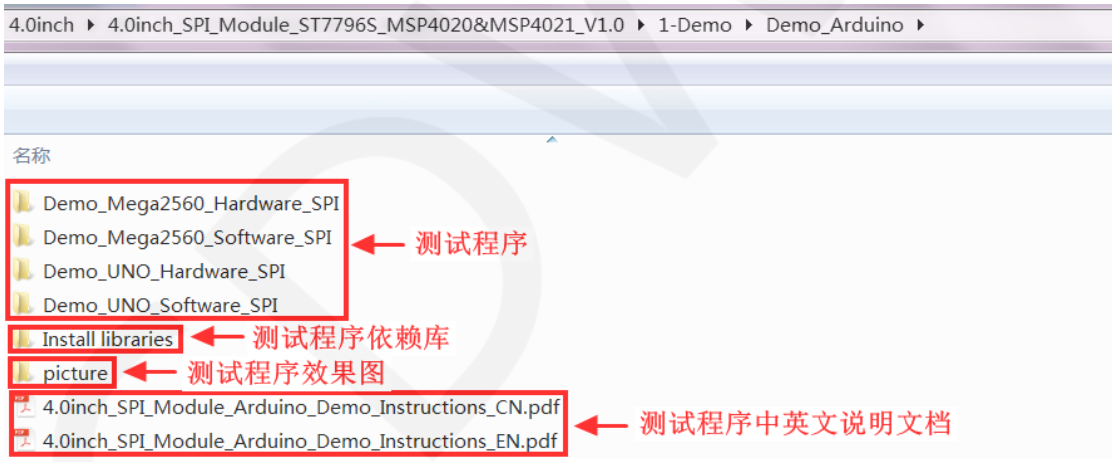
操作步骤：

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（默认的 Arduino 工程目录为 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；

2、C51 使用说明

接线说明：

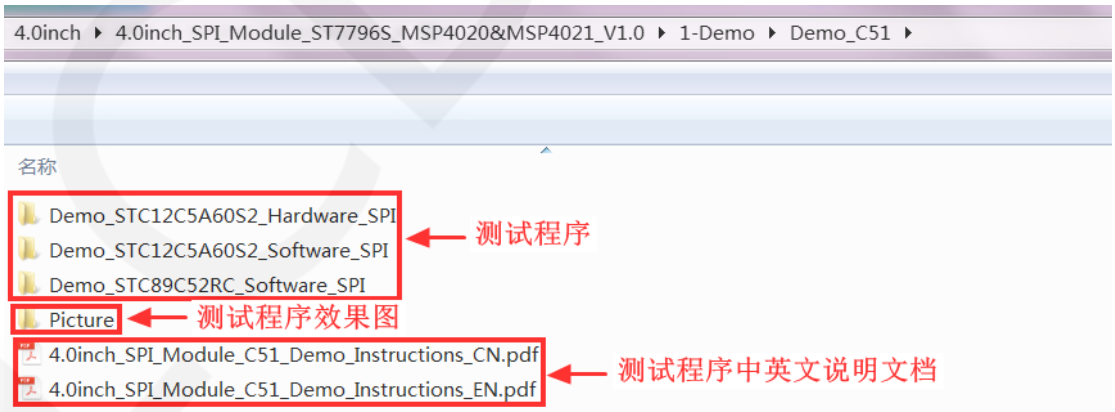
STC89C52RC和STC12C5A60S2单片机测试程序接线说明		
序号	引脚丝印	对应STC89/STC12开发板接线
1	VCC	5V/3.3V
2	GND	GND
3	CS	P13
4	RESET	P33
5	DC/RS	P12
6	SDI(MOSI)	P15
7	SCK	P17
8	LED	P32
9	SDO(MISO)	P16
10	T_CLK	P36
11	T_CS	P37
12	T_DIN	P34
13	T_DO	P35
14	T_IRQ	P40

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、 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、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：

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**
1	VCC	5V/3.3V
2	GND	GND
3	CS	PB11
4	RESET	PB12
5	DC/RS	PB10
6	SDI(MOSI)	PB15
7	SCK	PB13
8	LED	PB9
9	SDO(MISO)	PB14
10	T_CLK	PC0
11	T_CS	PC13
12	T_DIN	PC3
13	T_DO	PC2
14	T_IRQ	PC10

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**
1	VCC	5V/3.3V
2	GND	GND
3	CS	PB11
4	RESET	PB12
5	DC/RS	PB10
6	SDI(MOSI)	PB15
7	SCK	PB13

8	LED	PB9
9	SDO(MISO)	PB14
10	T_CLK	PC0
11	T_CS	PC13
12	T_DIN	PC3
13	T_DO	PC2
14	T_IRQ	PC10

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
1	VCC	5V/3.3V
2	GND	GND
3	CS	PB15
4	RESET	PB12
5	DC/RS	PB14
6	SDI(MOSI)	PB5
7	SCK	PB3
8	LED	PB13
9	SDO(MISO)	PB4
10	T_CLK	PB0
11	T_CS	PC5
12	T_DIN	PF11
13	T_DO	PB2
14	T_IRQ	PB1

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	VCC	5V/3.3V
2	GND	GND
3	CS	PD11

4	RESET	PD12
5	DC/RS	PD5
6	SDI(MOSI)	PF9
7	SCK	PF7
8	LED	PD6
9	SDO(MISO)	PF8
10	T_CLK	PH6
11	T_CS	PI8
12	T_DIN	PI3
13	T_DO	PG3
14	T_IRQ	PH11

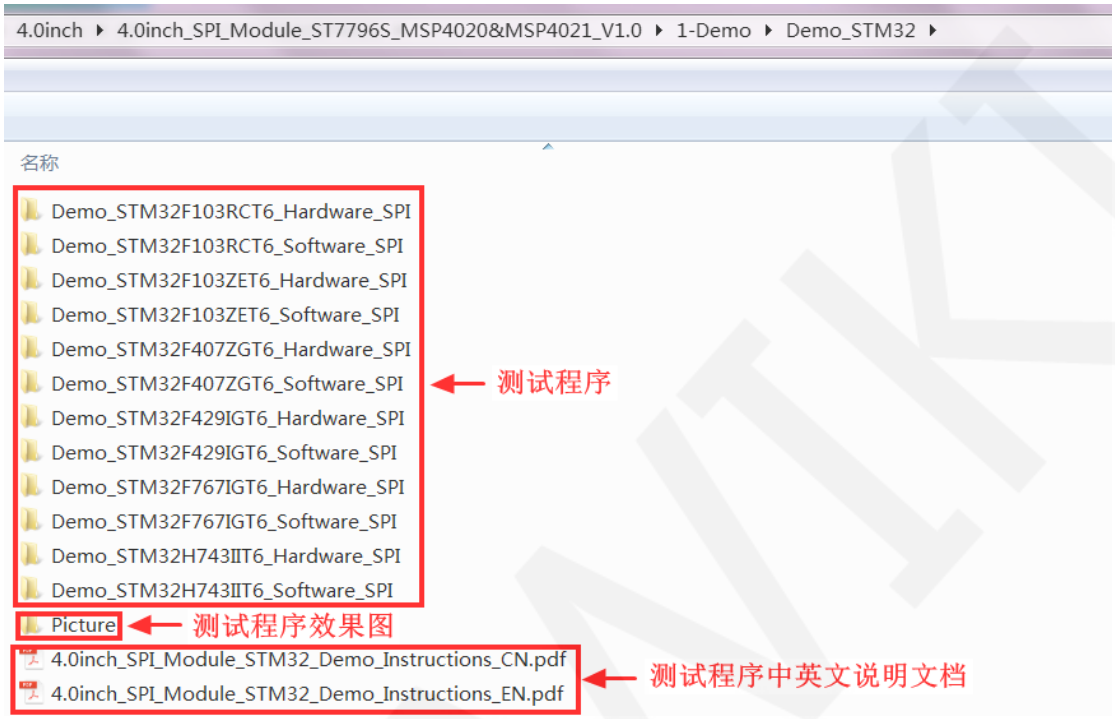
STM32F767IGT6、STM32H743IIT6单片机测试程序接线说明		
序号	**引脚丝印**	**对应Apollo STM32F4/F7开发板接线**
1	VCC	5V/3.3V
2	GND	GND
3	CS	PD11
4	RESET	PD12
5	DC/RS	PD5
6	SDI(MOSI)	PB15
7	SCK	PB13
8	LED	PD6
9	SDO(MISO)	PB14
10	T_CLK	PH6
11	T_CS	PI8
12	T_DIN	PI3
13	T_DO	PG3
14	T_IRQ	PH11

操作步骤：

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、 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

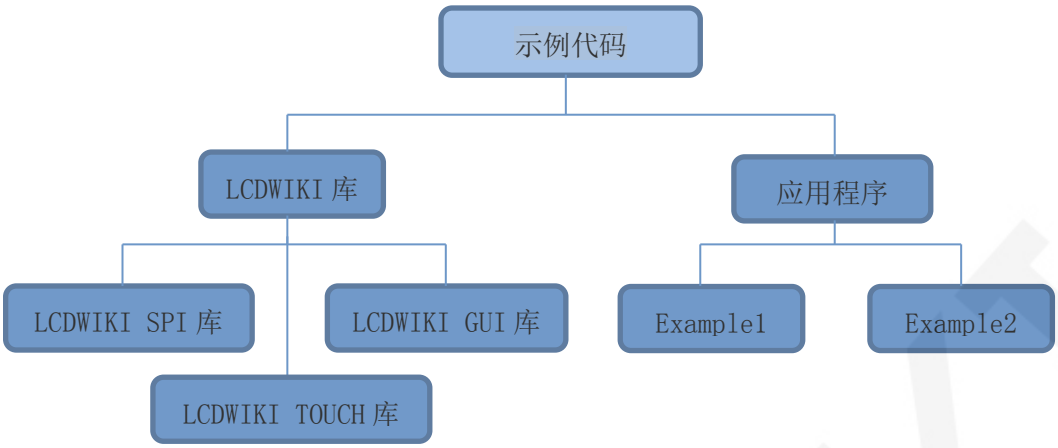
D、LCD 模块如果正常显示字符和图形，则说明程序运行成功；

软件说明

1、代码架构

A、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 库包含三部分内容:LCDWIKI_SPI 库、LCDWIKI_GUI 库以及 LCDWIKI_TOUCH 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。

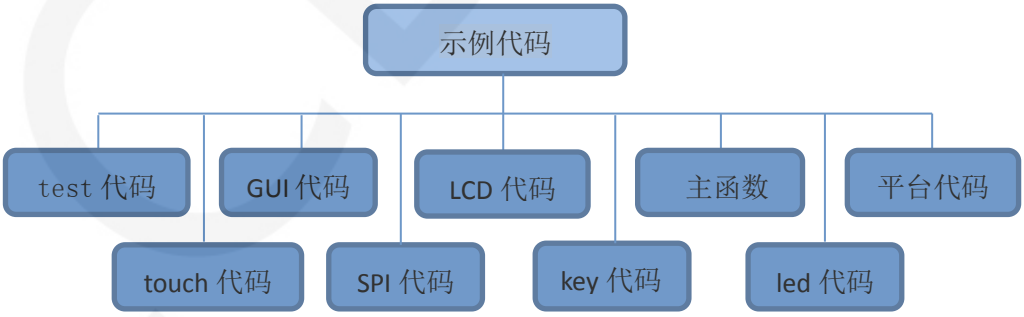
LCDWIKI_GUI 为中间层库，负责使用底层库提供的 API 实现图形的绘制，字符显示。

LCDWIKI_TOUCH 为触摸屏底层库，主要负责触摸中断检测，触摸数据采样和 AD 转换以及触摸数据发送。

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。

B、C51 和 STM32 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；

触摸屏相关的操作都包含在 touch 代码中；

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、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 ILI9341
#define CS   A5
#define CD   A3
#define RST  A4
#define MOSI  11
#define MISO  12
#define SCK   13
#define LED  A0   //if you don't need to control the L

//touch screen paramters define
#define TCS   2
#define TCLK  3
#define TDOUT 4
#define TDIN  5
#define TIRQ  6
```

液晶屏GPIO定义

触摸屏GPIO定义

注意：使用触摸功能的测试程序才定义触摸屏 GPIO。

B、C51 测试程序 GPIO 定义说明

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。

触摸屏相关的 GPIO 定义放在 touch.h 文件里面，如下图所示（以 STC12C5A60S2 单片机测试程序为例）：

```
sfr    P4     = 0xC0;
sbit DCLK     =     P3^6;
sbit TCS      =     P3^7;
sbit DIN      =     P3^4;
sbit DOUT     =     P3^5;
sbit Penirq   =     P4^0;    //检测触摸屏响应信号
```

触摸屏的 GPIO 定义都可以修改，可以定义成其他任何空闲的 GPIO。

如果单片机没有 P4 GPIO 组，可以把 penirq 定义成其他 GPIO。

C、STM32 测试程序 GPIO 定义说明

STM32 的液晶屏非 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）：

```
#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（以 STM32F103RCT6 单片机测试程序为例）：

```
#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（以 STM32F103RCT6 单片机测试程序为例）：

```
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。

STM32 触摸屏相关的 GPIO 定义放在 touch.h 文件里面，如下图所示（以 STM32F103RCT6 单片机测试程序为例）：

```
//与触摸屏芯片连接引脚
//与触摸屏芯片连接引脚
#define PEN  PCin(10)    //PC10  INT
#define DOUT PCin(2)     //PC2  MISO     PC2--PB14
#define TDIN PCout(3)    //PC3  MOSI     PC3--PB15
#define TCLK PCout(0)    //PC0  SCLK   PC0--PB13
#define TCS  PCout(13)   //PC13 CS
```

修改括号里面的值即可，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

A、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。

B、C51 和 STM32 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。

5、触摸屏校准说明

A、Arduino 测试程序触摸屏校准说明

Arduino 触摸屏校准需要先运行 touch_screen_calibration 程序，然后根据提示进行校准，校准合格后，需要将屏幕显示的校准参数写入 LCDWIKI_TOUCH 库的 cali_para.h 文件里面，如下图所示：

```
//for resolution 240x320,the calibration parameter is 663,-13,894,-30
//for resolution 320x480,the calibration parameter is 852,-14,1284,-30

#define XFAC      852 //663
#define XOFFSET   (-14) //(-13)
#define YFAC      1284 //894
#define YOFFSET   (-30)
```

B、C51 测试程序触摸屏校准说明

C51 的触摸屏校准需要执行 Touch_Adjust 测试项(只有 STC12C5A60S2 测试程序才有)，如下图所示：

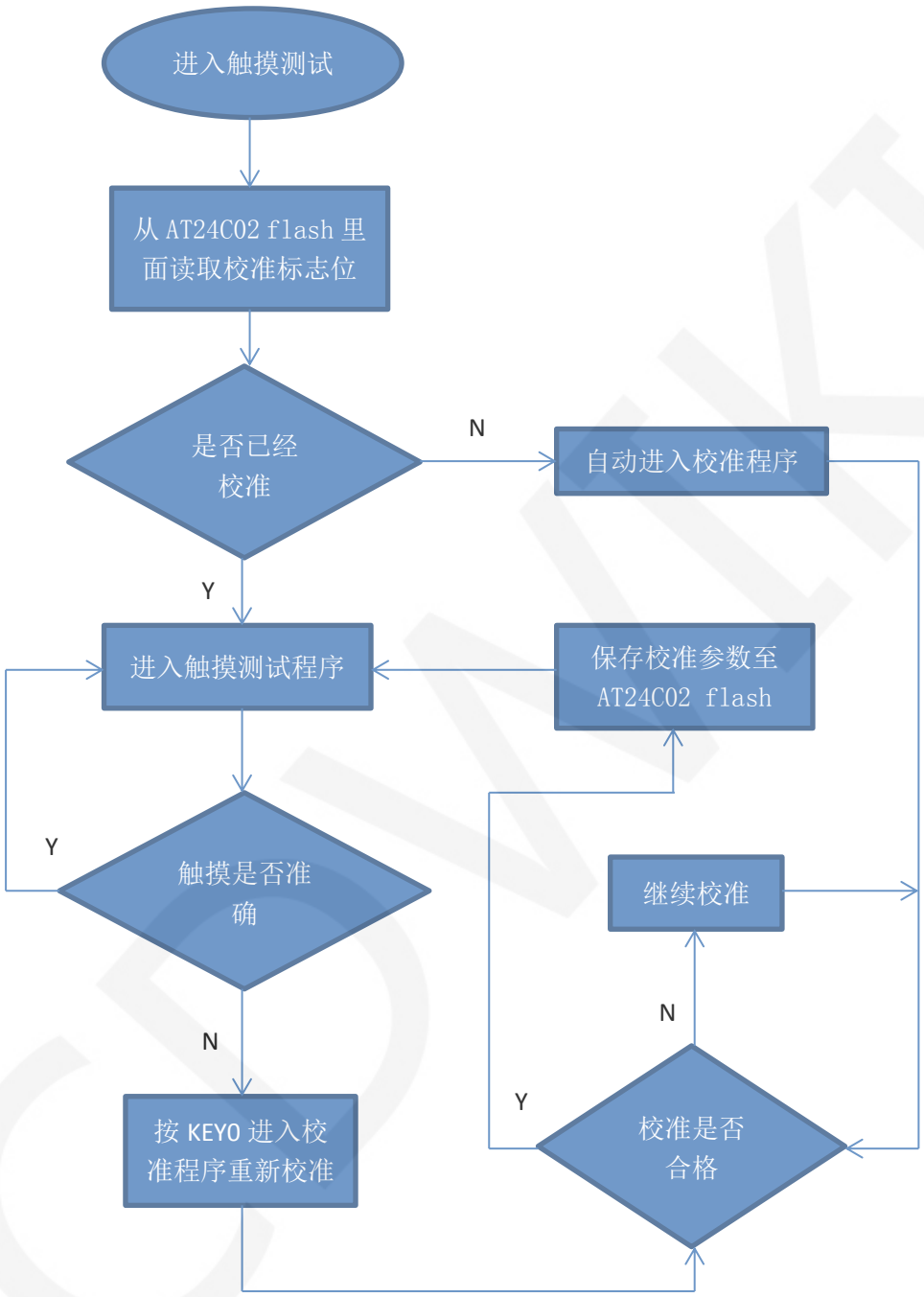
```
//循环进行各项测试
while(1)
{
  main_test();     //测试主界面
  Test_Color();      //简单刷屏填充测试
  Test_FillRec();    //GUI矩形绘图测试
  Test_Circle();     //GUI画圆测试
  Test_Triangle();    //GUI三角形填充测试
  English_Font_test();//英文字体示例测试
  Chinese_Font_test();//中文字体示例测试
  Pic_test();      //图片显示示例测试
  Rotate_Test();
//不使用触摸或者模块本身不带触摸，请屏蔽下面触摸屏测试
  Touch_Test();    //触摸屏手写测试
//需要触摸校准时，请将触摸手写测试屏蔽，将下面触摸校准测试项打开
//  Touch_Adjust();  //触摸校准
}
```

触摸校准合格后，需要将屏幕显示的校准参数保存在 touch.c 文件中，如下图所示：

```
//***因触摸屏批次不同等原因，默认的校准参数值可能会引起触摸
u16 vx=11738,vy=7736;   //比例因子，此值除以1000之后表示多少
u16 chx=3905,chy=246;//默认像素点坐标为0时的AD起始值
//***因触摸屏批次不同等原因，默认的校准参数值可能会引起触摸
```

C、STM32 测试程序触摸屏校准说明

STM32 触摸屏校准程序可以自动识别是否需要校准或者手动通过按键进入校准，此过程包含在触摸屏测试项中，校准标志和校准参数保存在 AT24C02 flash 里，需要时要从 flash 里面读取，校准流程如下图所示：


进入触摸测试
从 AT24C02 flash 里面读取校准标志位
是否已经校准
N
自动进入校准程序
Y
进入触摸测试程序
保存校准参数至 AT24C02 flash
Y
触摸是否准确
继续校准
N
N
Y
按 KEY0 进入校准程序重新校准
校准是否合格

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。