4.0inch Arduino Mega2560 8&16BIT Module MAR4018

用户手册

产品概述

该产品为一款 4.0 寸 TFT LCD 模块，拥有 800x480 分辨率，支持 16BIT RGB 65K 色显示，内部驱动 IC 为 NT35510，支持 8 位和 16 位并口通信，默认采用 16 位并口通信方式。该模块包含有 LCD 显示屏、电阻触摸屏、SD 卡插槽以及 PCB 底板等部件，支持 SD 卡扩展功能，可以直插到 Arduino MEGA2560 开发板上使用，还可以用于 C51 和 STM32 平台。

产品特点

- 4.0 寸彩屏，支持 16BIT RGB 65K 色显示，显示色彩丰富
- 800x480 分辨率，显示效果清晰
- 支持 8 位和 16 位并行总线传输，传输速度快
- 板载 5V/3.3V 电平转换 IC，兼容 5V/3.3V 工作电压
- 支持 Arduino Mage2560 直插式使用
- 支持触摸功能
- 支持 SD 卡功能扩展
- 提供 Arduino 库和丰富的示例程序
- 可用于 C51 和 STM32 平台并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	MAR4018
尺寸	4.0(inch)
类型	TFT
驱动芯片	NT35510

分辨率	800*480 (Pixel)
模块接口	8Bit or 16Bit parallel interface
有效显示区域	86.40x51.84(mm)
模块尺寸	58.65x108.48 (mm)
背光	8 chip HighLight white LEDs paralle
工作温度	-10℃~60℃
存储温度	-20℃~70℃
工作电压	5V
功耗	待定
产品重量(含包装)	60g

接口说明

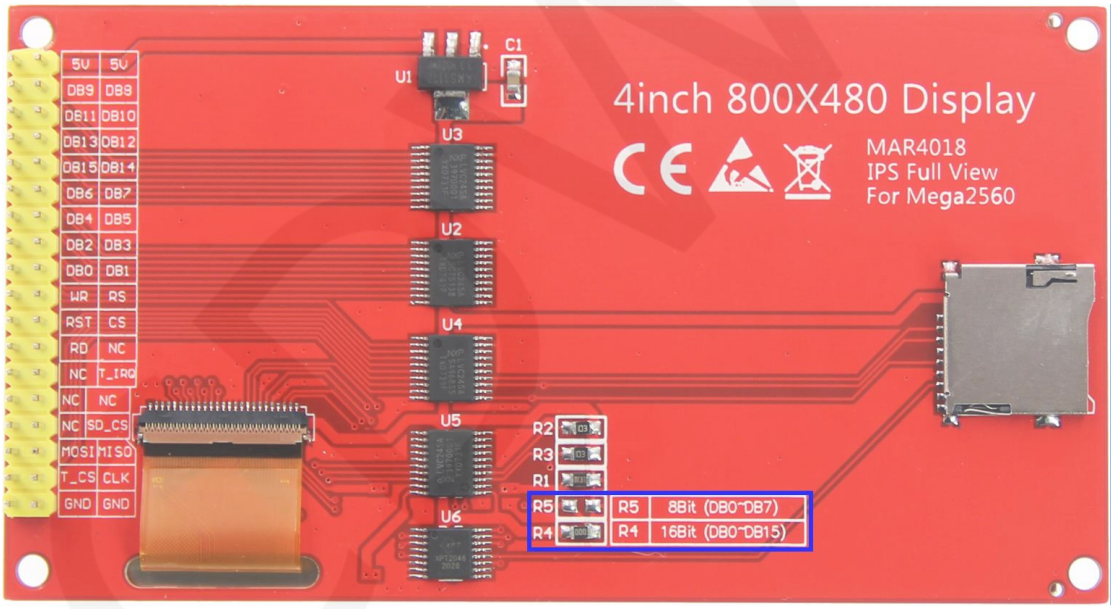


图 1. 模块引脚丝印图

注意：

1. 该模块硬件支持8位和16位并口数据总线模式切换（如上面图1中蓝框所示），具体说明如下：

   A. 将R4焊接0欧电阻或者直接短接，并将R5断开：选择16位数据总线模式（默认），使用DB0~DB15数据引脚

B. 将R5焊接0欧电阻或者直接短接，并将R4断开：选择8位数据总线模式，使用DB0~DB7数据引脚

重要说明：

1. 以下引脚序号1~30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：LCD_CS在我们模块上是20脚，可能在不同尺寸裸屏上是x脚。
2. 关于VCC供电电压：如果您购买的是带PCB底板模块，VCC/VDD供电需要接5V（模块已集成超低压差5V转3.3V电路），如果您购买的是液晶屏裸屏，切记只能接3.3V。
3. 关于背光电压：带PCB底板的模块均已接入3.3V，不需要再手动接入。如果您购买的是裸屏，则 LEDA接3.0V-3.3V，LEDKx接地即可。

序号	模块引脚	引脚说明
1	5V	电源引脚
2	5V	
3	DB8	数据总线高8位引脚
4	DB9	
5	DB10	
6	DB11	
7	DB12	
8	DB13	
9	DB14	
10	DB15	
11	DB7	数据总线低8位引脚
12	DB6	
13	DB5	
14	DB4	
15	DB3	
16	DB2	
17	DB1	
18	DB0	
19	RS	液晶屏寄存器/数据选择引脚（高电平：数据，低电平：寄存器）

20	WR	液晶屏写控制引脚
21	CS	液晶屏片选控制引脚（低电平有效）
22	RST	液晶屏复位控制引脚（低电平有效）
23	NC	无定义，保留
24	RD	液晶屏读控制引脚
25	T_IRQ	触摸屏中断控制引脚（低电平触发）
26	NC	无定义，保留
27	NC	
28	NC	
29	SD_CS	扩展引用：SD卡片选引脚
30	NC	无定义，保留
31	MISO	SPI总线输入引脚（扩展应用）
32	MOSI	SPI总线输出引脚（扩展应用）
33	CLK	SPI总线时钟引脚
34	T_CS	触摸屏片选引脚（低电平有效）
35	GND	电源地
36	GND	

硬件配置

该 LCD 模块硬件电路包含五大部分:LCD 显示控制电路、电平转换电路、SD 卡控制电路、触摸屏控制电路以及 8 位和 16 位数据总线模式切换电路。

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。

电平转换电路用于进行 5V/3.3V 转换，使模块可以兼容 3.3V/5V 电源。

SD 卡控制电路用于 SD 卡功能扩展，控制 SD 卡的识别，读取及写入。

触摸屏控制电路用于控制触摸屏中断获取，数据采样，AD 转换，数据发送等。

8 位和 16 位数据总线模式切换电路用于切换数据总线类型（8 位模式和 16 位模式），具体说明见以上图 1 中红框内所示或者查阅模块电路原理图。

工作原理

1、NT35510 控制器简介

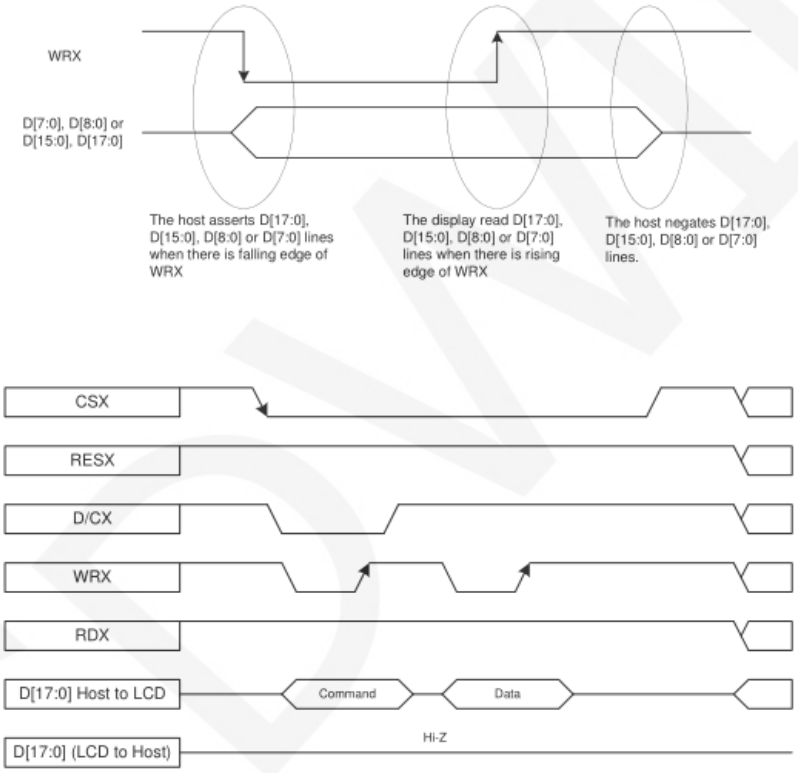
NT35510 控制器是一款用于 TFT LCD 的驱动 IC，其支持多种分辨率：480*864、480*854、

480*800、480*720、480*640 以及 480*1024(需扩展内存)。它拥有一个 1244160 字节大小的内存，可以支持 MDDI 接口、MIPI 接口、16 位/18 位/24 位 RGB 接口、8 位/16 位/18 位/24 位并口、SPI 以及 I2C 接口。其支持 8、65K、262K 以及 16.7M 的 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。

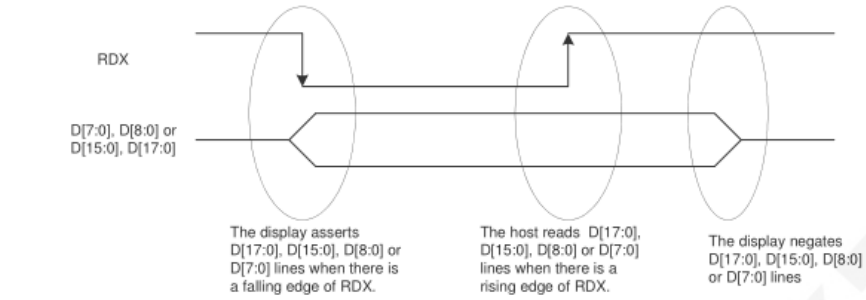
本模块使用 16 位并口传输数据，使用 16bit（RGB565）来控制一个像素点显示，因此可以每个像素点显示颜色多达 65K 种。像素点地址设置按照行列的顺序进行，递增递减方向由扫描方式决定。NT35510 显示方法按照先设置地址再设置颜色值进行。

2、并口通信简介

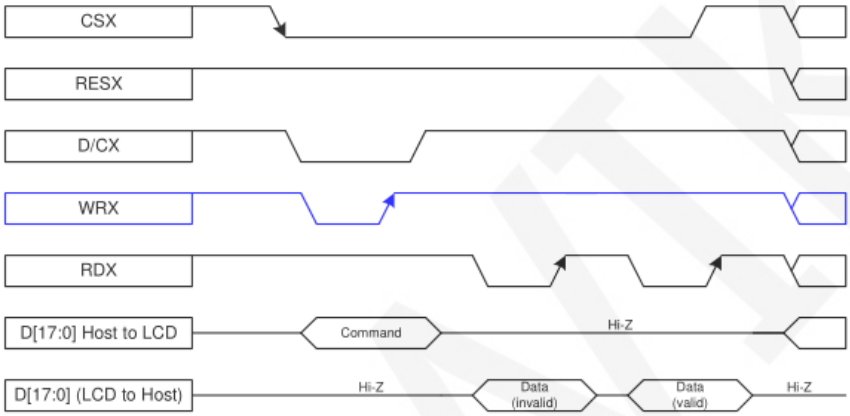
并口通信写模式时序如下图所示：



并口通信读模式时序如下图所示：



Note: RDX is an unsynchronized signal (It can be stopped).



CSX 为片选信号，用于开启和禁止并口通信，低电平有效

RESX 为外部复位信号，低电平有效

D/CX 为数据或者命令选择信号，1-写数据或者命令参数，0-写命令

WRX 为写数据控制信号

RDX 为读数据控制信号

D[X:0]为并口数据位，共有 8 位、9 位、16 位、18 位四种类型

当进行写入操作时，在已经复位的基础上，先设置数据或者命令选择信号，然后将片选信号拉低，接下来从主机输入需要写入的内容，然后将写数据控制信号拉低再拉高，数据在写控制信号的上升沿会被写入到液晶屏控制 IC，最后将片选信号拉高，一次数据写入操作完成。

当进入读操作时，在已经复位的基础上，先将片选信号拉低，然后将数据或者命令选择信号拉高，接下来将读数据控制信号拉低，然后从液晶屏控制 IC 读取数据，再将读数据控制信号拉高，数据在读数据控制信号上升沿会被读取出来，最后将片选信号拉高，一次数据读取操作完成

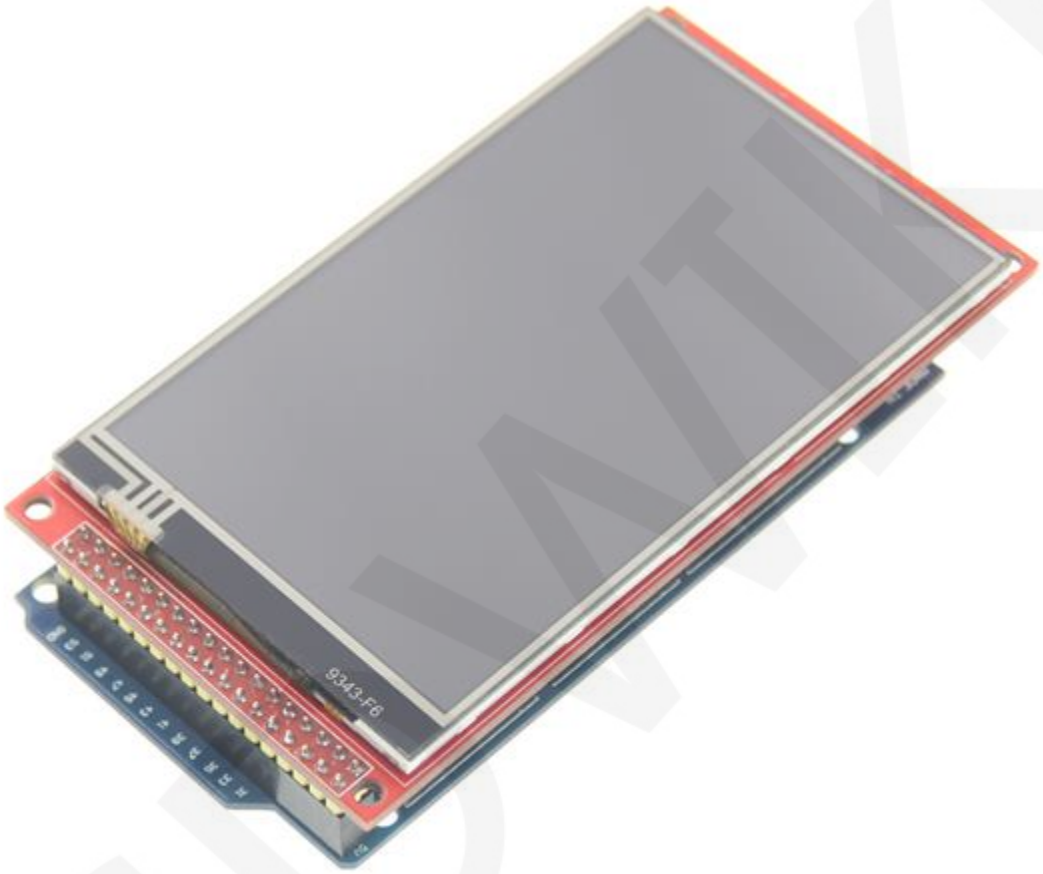
使用说明

1、Arduino 使用说明

接线说明：

引脚标注见接口说明。

此模块可以直接插入 Mega2560 中使用，不需要再手动接线。



与Mega2560直插图

Arduino MEGA2560单片机测试程序直插说明			
序号	模块引脚	对应MEGA2560开发板直插引脚	
		8位模式	16位模式
1	5V	5V	
2	5V		
3	DB8	没使用	22

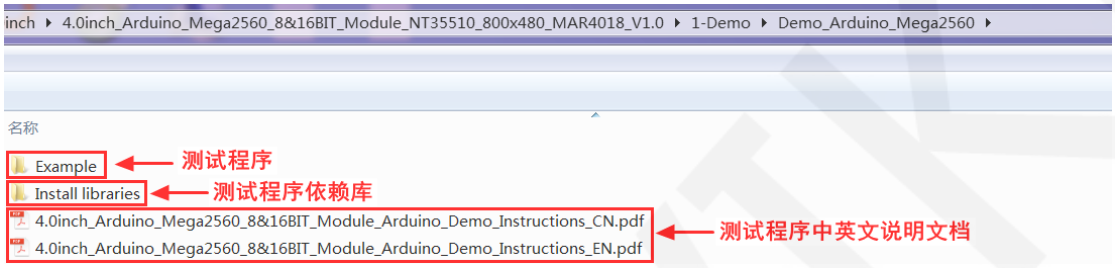
4	DB9		23
5	DB10		24
6	DB11		25
7	DB12		26
8	DB13		27
9	DB14		28
10	DB15		29
11	DB7	30	
12	DB6	31	
13	DB5	32	
14	DB4	33	
15	DB3	34	
16	DB2	35	
17	DB1	36	
18	DB0	37	
19	RS	38	
20	WR	39	
21	CS	40	
22	RST	41	
23	NC	没使用	
24	RD	43	
25	T_IRQ	44	
26	NC	没使用	
27	NC		
28	NC		
29	SD_CS	48	
30	NC	没使用	
31	MISO	50	
32	MOSI	51	
33	T_CS	53	
34	CLK	52	
35	GND	GND	
36	GND		

操作步骤：

A、按照上述接线说明将 LCD 模块直插到 Arduino 单片机上，并上电；

B、将测试程序包中 Install libraries 目录下的依赖库拷贝到 Arduino 工程目录的 libraries 文件夹下（如果不需要依赖库，则不需要拷贝）；

C、选择需要测试的 Arduino 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）



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：

引脚标注见接口说明。

STC89C52RC单片机测试程序接线说明			
序号	模块引脚	对应STC89开发板接线引脚	
		8位模式	16位模式
1	5V	5V	
2	5V		
3	DB8	不需要接	P20
4	DB9		P21
5	DB10		P22
6	DB11		P23
7	DB12		P24

8	DB13		P25
9	DB14		P26
10	DB15		P27
11	DB7	P37	
12	DB6	P36	
13	DB5	P35	
14	DB4	P34	
15	DB3	P33	
16	DB2	P32	
17	DB1	P31	
18	DB0	P30	
19	RS	P12	
20	WR	P11	
21	CS	P13	
22	RST	P14	
23	NC	不需要接	
24	RD	P10	
25	T_IRQ	不需要接（不能测试触摸）	
26	NC	不需要接	
27	NC		
28	NC		
29	SD_CS	不需要接	
30	NC	不需要接	
31	MISO	不需要接（不能测试触摸）	
32	MOSI	不需要接（不能测试触摸）	
33	CLK	不需要接（不能测试触摸）	
34	T_CS	不需要接（不能测试触摸）	
35	GND	GND	
36	GND		

STC12C5A60S2单片机测试程序接线说明			
序号	模块引脚	对应STC12开发板接线引脚	
		8位模式	16位模式
1	5V	5V	
2	5V		
3	DB8	不需要接	P20
4	DB9		P21
5	DB10		P22
6	DB11		P23
7	DB12		P24
8	DB13		P25
9	DB14		P26
10	DB15		P27
11	DB7	P07	
12	DB6	P06	
13	DB5	P05	
14	DB4	P04	
15	DB3	P03	
16	DB2	P02	
17	DB1	P01	
18	DB0	P00	
19	RS	P12	
20	WR	P11	
21	CS	P13	
22	RST	P33	
23	NC	不需要接	
24	RD	P10	
25	T_IRQ	P40	
26	NC	不需要接	
27	NC		

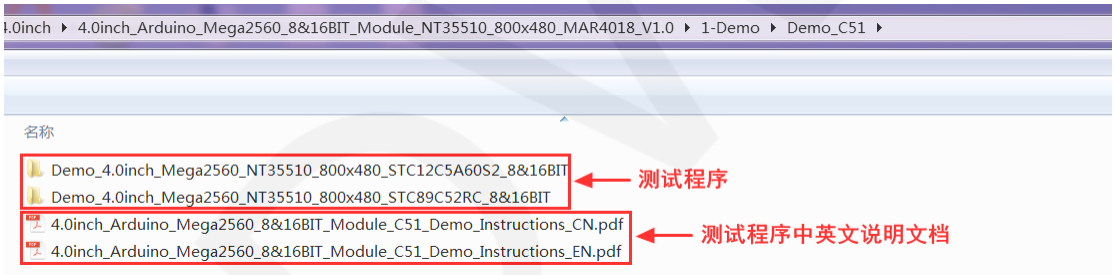
28	NC	
29	SD_CS	不需要接
30	NC	不需要接
31	MISO	P35
32	MOSI	P34
33	CLK	P36
34	T_CS	P37
35	GND	GND
36	GND	

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：

引脚标注见接口说明。

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	
		8位模式	16位模式
1	5V	5V	
2	5V		
3	DB8	不需要接	PB8
4	DB9		PB9
5	DB10		PB10
6	DB11		PB11
7	DB12		PB12
8	DB13		PB13
9	DB14		PB14
10	DB15		PB15
11	DB7	PB7	
12	DB6	PB6	
13	DB5	PB5	
14	DB4	PB4	
15	DB3	PB3	
16	DB2	PB2	
17	DB1	PB1	
18	DB0	PB0	
19	RS	PC8	
20	WR	PC7	
21	CS	PC9	
22	RST	PC4	
23	NC	不需要接	
24	RD	PC6	
25	T_IRQ	PC1	
26	NC	不需要接	

27	NC	
28	NC	
29	SD_CS	不需要接
30	NC	不需要接
31	MISO	PC2
32	MOSI	PC3
33	CLK	PC0
34	T_CS	PC13
35	GND	GND
36	GND	

STM32F103ZET6单片机测试程序接线说明			
序号	模块引脚	对应Elite STM32开发板接线引脚（使用FSMC总线）	
		8位模式	16位模式
1	5V	5V	
2	5V		
3	DB8	不需要接	PE11
4	DB9		PE12
5	DB10		PE13
6	DB11		PE14
7	DB12		PE15
8	DB13		PD8
9	DB14		PD9
10	DB15		PD10
11	DB7	PE10	
12	DB6	PE9	
13	DB5	PE8	
14	DB4	PE7	
15	DB3	PD1	

16	DB2	PD0
17	DB1	PD15
18	DB0	PD14
19	RS	PG0
20	WR	PD5
21	CS	PG12
22	RST	单片机复位引脚
23	NC	不需要接
24	RD	PD4
25	T_IRQ	PF10
26	NC	不需要接
27	NC	
28	NC	
29	SD_CS	不需要接
30	NC	不需要接
31	MISO	PB2
32	MOSI	PF9
33	CLK	PB1
34	T_CS	PF11
35	GND	GND
36	GND	

STM32F407VGT6单片机测试程序接线说明			
序号	模块引脚	对应STM32F407VxT6开发板接线引脚(使用FSMC总线)	
		8位模式	16位模式
1	5V	5V	
2	5V		
3	DB8	不需要接	PE11

4	DB9		PE12
5	DB10		PE13
6	DB11		PE14
7	DB12		PE15
8	DB13		PD8
9	DB14		PD9
10	DB15		PD10
11	DB7	PE10	
12	DB6	PE9	
13	DB5	PE8	
14	DB4	PE7	
15	DB3	PD1	
16	DB2	PD0	
17	DB1	PD15	
18	DB0	PD14	
19	RS	PD11	
20	WR	PD5	
21	CS	PD7	
22	RST	单片机复位引脚	
23	NC	不需要接	
24	RD	PD4	
25	T_IRQ	PB1	
26	NC	不需要接	
27	NC		
28	NC		
29	SD_CS	不需要接	
30	NC	不需要接	
31	MISO	PB2	
32	MOSI	PC4	
33	CLK	PB0	

34	T_CS	PC13
35	GND	GND
36	GND	

STM32F407ZGT6单片机测试程序接线说明			
序号	模块引脚	对应Explorer STM32开发板接线引脚（使用FSMC总线）	
		8位模式	16位模式
1	5V	5V	
2	5V		
3	DB8	不需要接	PE11
4	DB9		PE12
5	DB10		PE13
6	DB11		PE14
7	DB12		PE15
8	DB13		PD8
9	DB14		PD9
10	DB15		PD10
11	DB7	PE10	
12	DB6	PE9	
13	DB5	PE8	
14	DB4	PE7	
15	DB3	PD1	
16	DB2	PD0	
17	DB1	PD15	
18	DB0	PD14	
19	RS	PF12	
20	WR	PD5	
21	CS	PG12	
22	RST	单片机复位引脚	

23	NC	不需要接
24	RD	PD4
25	T_IRQ	PB1
26	NC	不需要接
27	NC	
28	NC	
29	SD_CS	不需要接
30	NC	不需要接
31	MISO	PB2
32	MOSI	PF11
33	CLK	PB0
34	T_CS	PC13
35	GND	GND
36	GND	

STM32F429IGT6、STM32F767IGT6、STM32H743IIT6
单片机测试程序接线说明

序号	模块引脚	对应Apollo STM32F4/F7开发板接线引脚（使用FSMC总线）	
		8位模式	16位模式
1	5V	5V	
2	5V		
3	DB8	不需要接	PE11
4	DB9		PE12
5	DB10		PE13
6	DB11		PE14
7	DB12		PE15
8	DB13		PD8
9	DB14		PD9

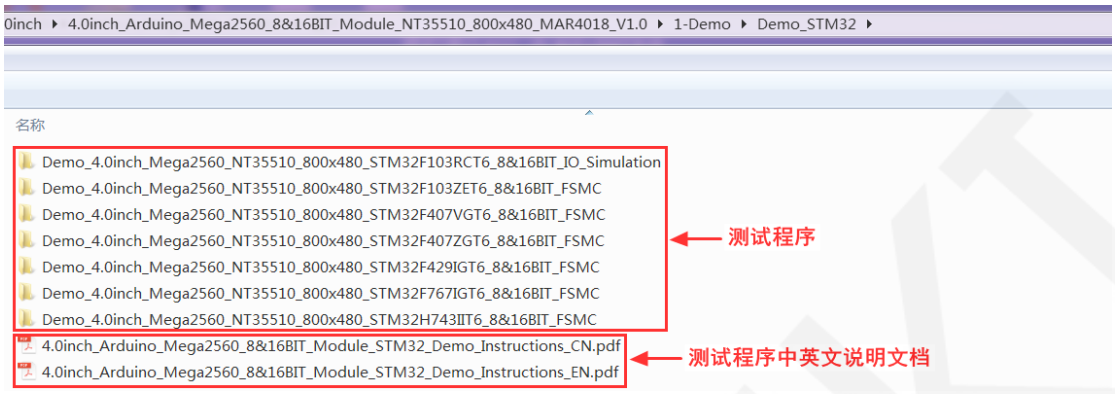
10	DB15		PD10
11	DB7	PE10	
12	DB6	PE9	
13	DB5	PE8	
14	DB4	PE7	
15	DB3	PD1	
16	DB2	PD0	
17	DB1	PD15	
18	DB0	PD14	
19	RS	PD13	
20	WR	PD5	
21	CS	PD7	
22	RST	单片机复位引脚	
23	NC	不需要接	
24	RD	PD4	
25	T_IRQ	PH7	
26	NC	不需要接	
27	NC		
28	NC		
29	SD_CS	不需要接	
30	NC	不需要接	
31	MISO	PG3	
32	MOSI	PI3	
33	CLK	PH6	
34	T_CS	PI8	
35	GND	GND	
36	GND		

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、 选择需要测试的 STM32 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 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

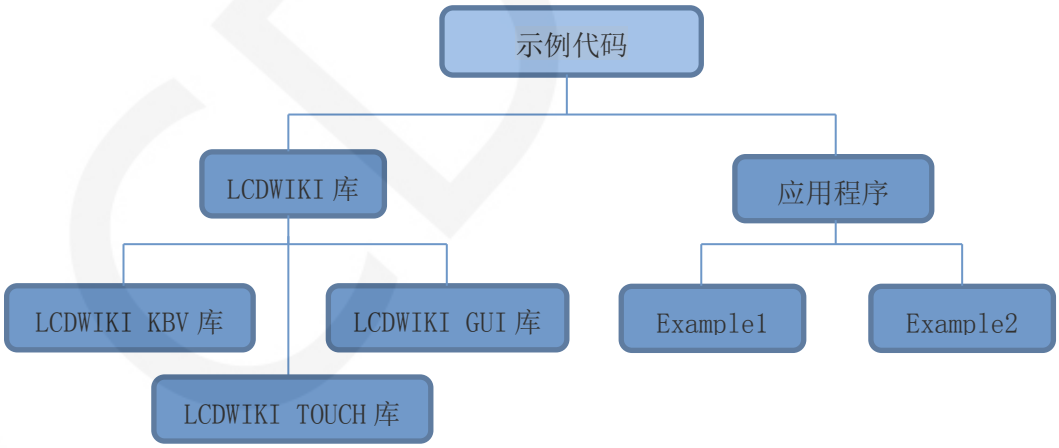
D、LCD 模块如果正常显示字符和图形，则说明程序运行成功；

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_KBV 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_KBV 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。

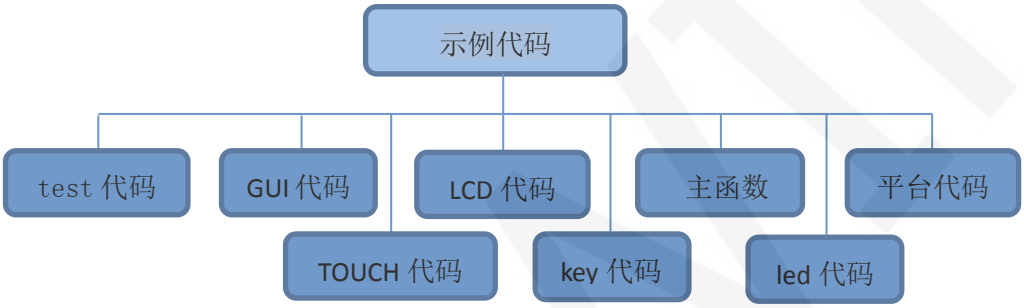
LCDWIKI_GUI 为中间层库，负责使用底层库提供的 API 实现图形的绘制，字符显示。

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 初始化以及相关的斌并口写数据操作都包含在 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；

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、GPIO 定义说明

A、Arduino 测试程序 GPIO 定义说明

模块是直插到 Arduino mage2560 上使用的，所以不允许修改 GPIO 口定义。

B、C51 测试程序 GPIO 定义说明

C51 测试程序 lcd 屏 GPIO 定义放在 lcd.h 文件里，如下图所示（以 STC12C5A60S2 单片机测试程序为例）：

```
//IO连接
#define  LCD_DataPortH P2      //高8位数据口,8位模式下只使用高8位
#define  LCD_DataPortL P0      //低8位数据口 ,8位模式下低8位可以不
sbit LCD_RS = P1^2;       //数据/命令切换
sbit LCD_WR = P1^1;      //写控制
//sbit LCD_RD = P1^0;         //读控制
sbit LCD_CS = P1^3;    //片选
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
```

并口引脚定义需要选择整套 GPIO 口组，如 P0，P2 等，这样传输数据时，操作方便。

其他引脚可以定义成任何空闲的 GPIO。

触摸屏 GPIO 口定义放在 touch.h 里，如下图所示（只有 12C5A60S2 才可以测试触摸）

```
//IO连接
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 FSMC 测试程序 lcd 屏 GPIO 由 FSMC 总线定义，相关的定义方法可以查阅 FSMC 总线说明资料，其 GPIO 定义放在 lcd.h 文件里如下图所示（以 STM32F103ZET6 单片机 FSMC 测试程序为例）：

```
//////////////////////////////////////////////////////////////////////////////
//-----------------LCD端口定义----------------
#define LED         0          //背光控制引脚        PB0

//QDtech全系列模块采用了三极管控制背光亮灭，用户也可以接PWM调节背光亮度
#define LCD_LED PBout(LED) //LCD背光
//如果使用官方库函数定义下列底层，速度将会下降到14帧每秒，建议采用我司推荐方法
//以下IO定义直接操作寄存器，快速IO操作，刷屏速率可以达到28帧每秒！

//LCD地址结构体
typedef struct
{
  vu16 LCD_REG;
  vu16 LCD_RAM;
} LCD_TypeDef;

//使用NOR/SRAM的 Bank1.sector4,地址位HADDR[27,26]=11 A10作为数据命令区分线
//注意设置时STM32内部会右移一位对其!
#define LCD_BASE        ((u32)(0x6C000000 | 0x000007FE))
#define LCD             ((LCD_TypeDef *) LCD_BASE)
```

STM32 IO 模拟测试程序 lcd 屏 GPIO 定义放在 lcd.h 文件里，如下图所示（以 STM32F103RCT6 单片机 IO 模拟测试程序为例）：

```
//////////////////////////////////////////////////////////////////////////////
//------------------LCD端口定义----------------
#define GPIO_TYPE  GPIOC  //GPIO组类型
#define LED        10         //背光控制引脚       PC10
#define LCD_CS     9        //片选引脚            PC9
#define LCD_RS     8        //寄存器/数据选择引脚 PC8
#define LCD_RST    4        //复位引脚            PC4
#define LCD_WR     7        //写引脚              PC7
#define LCD_RD     6        //读引脚              PC6

//PB0~15,作为数据线
//注意：如果使用8位模式数据总线，则液晶屏的数据高8位是接到MCU的高8位总
//举例：如果接8位模式则本示例接线为液晶屏DB10-DB17对应接至单片机GPIOB_
//举例：如果是16位模式：DB0-DB7分别接GPIOB_Pin0-GPIOB_Pin7,DB10-DB17对
#define DATAOUT(x) GPIOB->ODR=x; //数据输出
#define DATAIN     GPIOB->IDR;   //数据输入
```

数据并口引脚定义需要选择整套 GPIO 口组，如 PB，传输数据时，操作方便。

其他引脚可以定义成任何空闲的 GPIO。

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 IO 模拟测试程序为例）：

```
//与触摸屏芯片连接引脚
//与触摸屏芯片连接引脚
#define PEN  PCin(1)     //PC1  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
```

如果使用 IO 模拟测试程序，则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。

如果使用 FSMC 测试程序，则触摸屏 GPIO 不能修改，因为要采用直插方法连接，开发板上 GPIO 引脚固定。

3、并口通信代码实现

A、Arduino 测试程序并口通信代码实现

如果使用 8 位模式相关的代码在 LCDWIKI_KBV 库的 mcu_8bit_magic.h 文件里实现，如下图所示：

```
#define CMASK    0xFF
#define write8(d) {\
        PORTC = d;WR_STROBE;}
#define read8(dst) {\
        RD_ACTIVE; DELAY7; \
        dst = PINC;RD_IDLE;}
#define setWriteDir() {DDRC |= CMASK;}
#define setReadDir()  {DDRC &= ~CMASK;}
```

如果使用 16 位模式相关的代码在 LCDWIKI_KBV 库的 mcu_16bit_magic.h 文件里实现，如下图所示：

```
// Data write strobe, ~2 instructions and always inline
#define WR_STROBE { WR_ACTIVE; WR_IDLE; }
#define RD_STROBE {RD_IDLE; RD_ACTIVE;RD_ACTIVE;RD_ACTIVE;}
#define write16(x) { write_16(x) }
#define read16(dst) { read_16(dst) }
#define writeCmd8(x){ CD_COMMAND; write8(x); CD_DATA;  }
#define writeData8(x){  write8(x) }
#define writeCmd16(x){ CD_COMMAND; write16(x); CD_DATA; }
#define writeData16(x){ write16(x) }

#define write_16(x)    { PORTA = (x) >> 8; PORTC = x; WR_STROBE;}
#define write8(x)     { PORTC = x; WR_STROBE;}
```

B、C51 测试程序并口通信代码实现

相关的代码在 LCD.c 文件里实现，如下图所示：

```
void LCD_write(u8 HVAL,u8 LVAL)
{
  LCD_CS = 0;
  LCD_WR = 0;
  LCD_DataPortH = HVAL;
  LCD_DataPortL = LVAL;
  LCD_WR = 1;
  LCD_CS = 1;
}

u16 LCD_read(void)
{
  u16 d;
  LCD_CS = 0;
  LCD_RD = 0;
  delay_us(1); //delay 1 us
  d = LCD_DataPortH;
  d = (d<<8)|LCD_DataPortL;
  LCD_RD = 1;
  LCD_CS = 1;
  return d;
}
```

实现了 8、16 位命令以及 8、16 位数据传输。

C、STM32 测试程序并口通信代码实现

STM32 测试程序并口通信代码都放在 LCD.c 文件里实现。

FSMC 测试程序实现如下图所示：

```
u16 LCD_read(void)
{
  vu16 data;  //防止被优化
  data=LCD->LCD_RAM;
  return data;
}

/*****************************************************************************
 * @name       :void LCD_WR_REG(u16 data)
 * @date       :2018-08-09
 * @function   :Write an 16-bit command to the LCD screen
 * @parameters :data:Command value to be written
 * @retvalue   :None
*****************************************************************************/
void LCD_WR_REG(u16 data)
{
  LCD->LCD_REG=data; //写入要写的寄存器序号
}

/*****************************************************************************
 * @name       :void LCD_WR_DATA(u16 data)
 * @date       :2018-08-09
 * @function   :Write an 16-bit data to the LCD screen
 * @parameters :data:data value to be written
 * @retvalue   :None
*****************************************************************************/
void LCD_WR_DATA(u16 data)
{
  LCD->LCD_RAM=data; //写入要写的数据
}
```

IO 模拟测试程序实现如下图所示：

```
void LCD_write(u16 VAL)
{
  LCD_CS_CLR;
  DATAOUT(VAL);
  LCD_WR_CLR;
  LCD_WR_SET;
  LCD_CS_SET;
}

u16 LCD_read(void)
{
  u16 data;
  LCD_CS_CLR;
  LCD_RD_CLR;
  delay_us(1);//延时1us
  data = DATAIN;
  LCD_RD_SET;
  LCD_CS_SET;
  return data;
}
```

IO 模拟测试程序实现了 8、16 位命令以及 8、16 位数据写入和读取。

FSMC 测试程序实现了 16 位命令以及 16 位数据写入和读取。

4、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：

```
#define XFAC       1307 //663 //852
#define XOFFSET    -30//(-13) //(-14)
#define YFAC       2062//894 //1284
#define YOFFSET    -17//(-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：

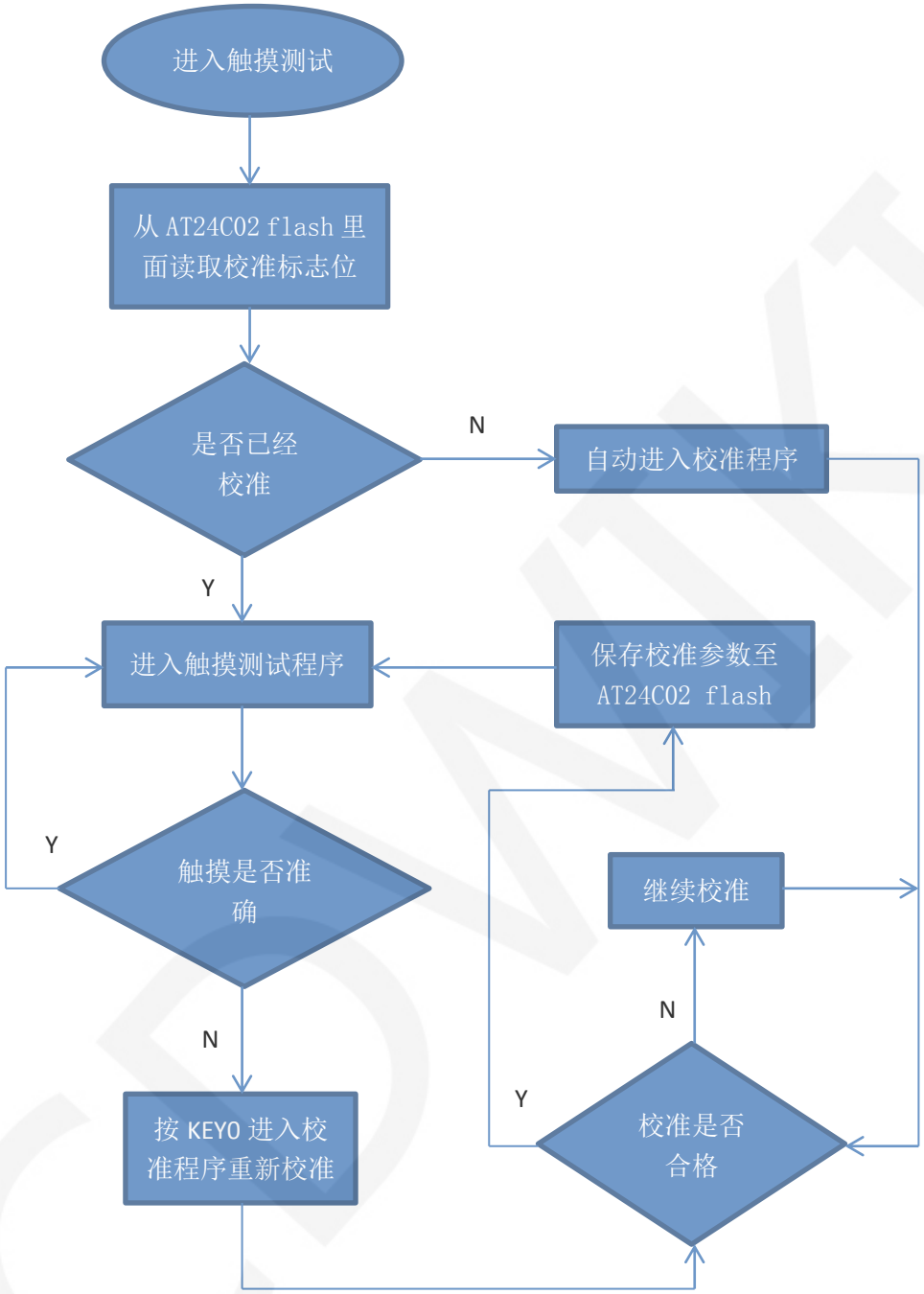
```
//循环进行各项测试
while(1)
{
  main_test();     //测试主界面
  Test_Color();      //简单刷屏填充测试
  Test_FillRec();    //GUI矩形绘图测试
  Test_Circle();     //GUI画圆测试
  Test_Triangle();    //GUI三角形填充测试
  English_Font_test();//英文字体示例测试
  Chinese_Font_test();//中文字体示例测试
  Pic_test();      //图片显示示例测试
  Rotate_Test();
//不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。