2.6inch Arduino UNO Module MAR2601

User Manual

Product Description

The Arduino UNO module is a 2.6-inch TFT LCD module with 320x240 resolution and 65K color display. It uses 8-bit line parallel port communication, and the driver IC is ILI9341. The module includes an LCD display, 5V~3.3V level conversion circuit, which can be directly plugged into the Arduino UNO and MEGA2560 development boards, and also supports SD card expansion function.

Product Features

- 2.6-inch color screen, support 16BIT RGB 65K color display, display rich colors

- 240x320 resolution for clear display

- 8-bit parallel bus transmission for fast transfer speed

- On-board 5V/3.3V level-shifting IC compatible with 5V/3.3V operating voltage

- Support Arduino UNO and MAGE2560 for direct plug-in use

- Support for touch function

- Support SD card function expansion

- Provide Arduino libraries and rich sample programs

- Military-grade process standards, long-term stable work

- Provide underlying driver technical support

Product Parameters

Name	Description
Display Color	RGB 65K color
SKU	MAR601
Screen Size	2.6(inch)
Type	TFT
Driver IC	ILI9341
Resolution	320*240 (Pixel)

Luminance	360Cd/m2
Module Interface	8Bit parallel interface
Active Area	39.60x52.80(mm)
Module PCB Size	52.70x78.22 (mm)
Operating Temperature	-20℃~60℃
Storage Temperature	-30℃~70℃
Operating Voltage	3.3V / 5V
Power Consumption	0.98w
Product Weight(including package)	35(g)

Interface Description
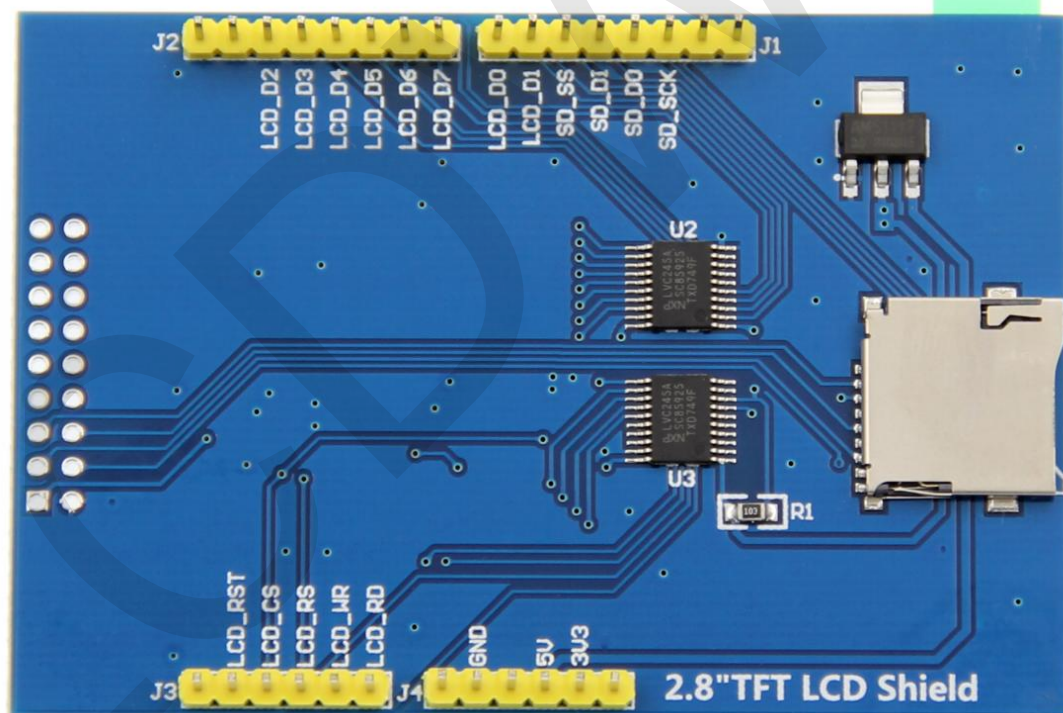
Module pin silkscreen image

Note: Pins that are not marked with silkscreen are not used.

Number	Module Pin	Pin Description
1	5V	Power positive 5V pin
2	3V3	Power positive 3.3V pin
3	GND	Power ground pin
4	LCD_D0	8-bit data bus pin
5	LCD_D1	
6	LCD_D2	
7	LCD_D3	
8	LCD_D4	
9	LCD_D5	
10	LCD_D6	
11	LCD_D7	
12	LCD_RST	LCD reset control pin
13	LCD_CS	LCD chip select control pin
14	LCD_RS	LCD register / data selection control pin
15	LCD_WR	LCD write control pin
16	LCD_RD	LCD read control pin
17	SD_SS	Extended function: SD card selection control pin
18	SD_DI	Extended function: SD card input pin
19	SD_DO	Extended function: SD card output pin
20	SD_SCK	Extended function: SD card clock control pin

Hardware Configuration

The LCD module hardware circuit comprises three parts: LCD display control circuit, level conversion circuit, SD card control circuit.

LCD display control circuit for controlling the pins of the LCD, including control pins and data transfer pins.

Level shifting circuit for 5V/3.3V conversion, making the module compatible with 3.3V/5V power supply.

SD card control circuit is used for SD card function expansion, controlling SD card identification, reading and writing.

working principle

## 1. Introduction to ITI9341 Controller

The ITI9341 controller supports a maximum resolution of 240*320 and has a 172800-byte GRAM. It also supports 8-bit, 9-bit, 16-bit, and 18-bit parallel port data buses. It also supports 3-wire and 4-wire SPI serial ports. Since the supported resolution is relatively large and the amount of data transmitted is large, the parallel port transmission is adopted, and the transmission speed is fast. ITI9341 also supports 65K, 262K RGB color display, display color is very rich, while supporting rotating display and scroll display and video playback, display in a variety of ways.

The ITI9341 controller uses 16bit (RGB565) to control a pixel display, so it can display up to 65K colors per pixel. The pixel address setting is performed in the order of rows and columns, and the incrementing and decreasing direction is determined by the scanning mode. The ITI9341 display method is performed by setting the address and then setting the color value.
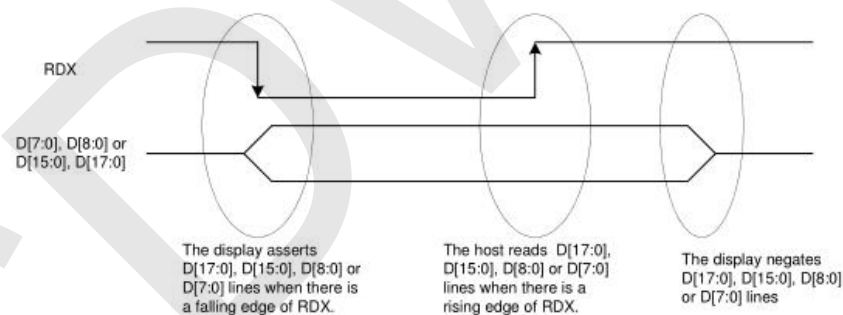
2. Introduction to parallel port communication

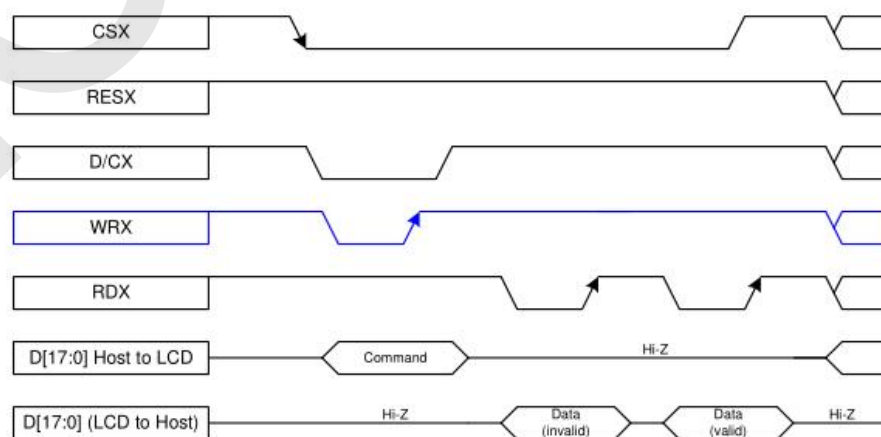The parallel port communication write mode timing is as shown below:
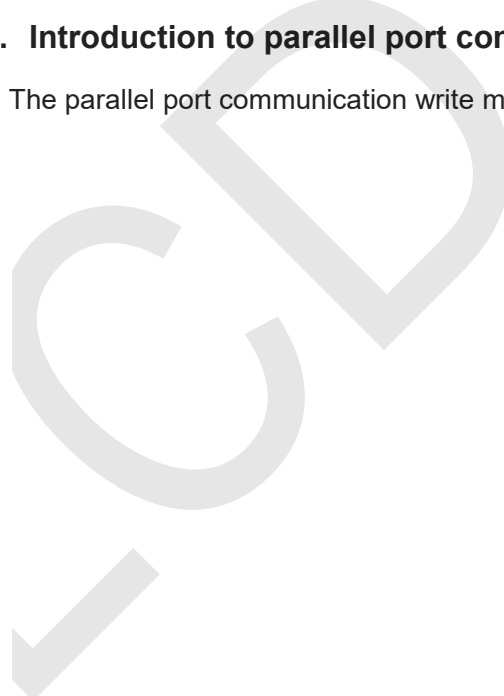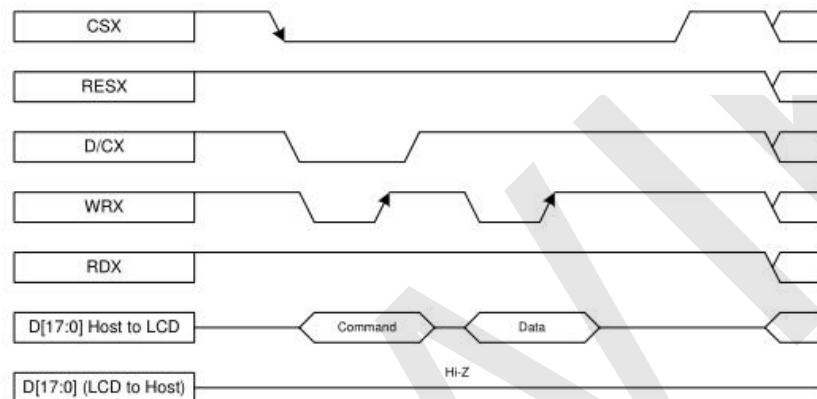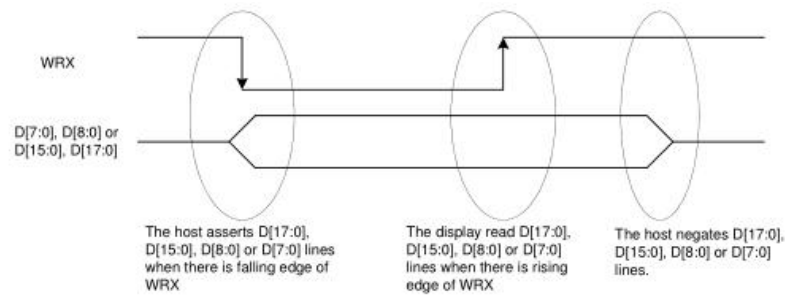
The timing of the parallel port communication read mode is shown in the figure

below:

CSX is a chip select signal for enabling and disabling parallel port communication, active low

RESX is an external reset signal, active low

D/CX is the data or command selection signal, 1-write data or command parameters, 0-write command

WRX is a write data control signal

RDX is a read data control signal

D[X:0] is a parallel port data bit, which has four types: 8-bit, 9-bit, 16-bit, and 18-bit.

When performing a write operation, on the basis of the reset, first set the data or command selection signal, then pull the chip select signal low, then input the content to be written from the host, and then pull the write data control signal low. When pulled high, data is written to the LCD control IC on the rising edge of the write control signal. Finally, the chip select signal is pulled high and a data write operation is completed.

When entering the read operation, on the basis of the reset, first pull the chip select signal low, then pull the data or command select signal high, then pull the read data control signal low, and then read the data from the LCD control IC. And then The read data control signal is pulled high, and the data is read out on the rising edge of the read data control signal. Finally, the chip select signal is pulled high, and a data read operation is completed.

Instructions for use

1. Arduino instructions

Wiring instructions:

See the interface description for pin assignments.

This module can be directly inserted into the Arduino UNO and Mega2560, no need to manually wire, as shown below:

**UNO directly inserted picture**

Mega2560 directly inserted picture

Direct insertion instructions for Arduino UNO and MEGA2560 microcontroller test program pins		
Number	Module Pin	Corresponding to UNO and MEGA2560 development board direct plug pins
1	5V	5V
2	3V3	3.3V
3	GND	GND
4	LCD_D0	8
5	LCD_D1	9
6	LCD_D2	2
7	LCD_D3	3
8	LCD_D4	4
9	LCD_D5	5
10	LCD_D6	6
11	LCD_D7	7
12	LCD_RST	A4
13	LCD_CS	A3
14	LCD_RS	A2
15	LCD_WR	A1
16	LCD_RD	A0
17	SD_SS	10
18	SD_DI	11
19	SD_DO	12
20	SD_SCK	13

Operating Steps：

A. Insert the LCD module directly into the Arduino MCU according to the above wiring instructions, and power on;

B. Copy the dependent libraries in the Install libraries directory of the test package to the libraries folder of the Arduino project directory (if you do not need to depend on the libraries, you do not need to copy them);

C. Open the directory where the Arduino test program is located and select the

example you want to test, as shown below:

(Please refer to the test program description document in the test package for the

test program description)
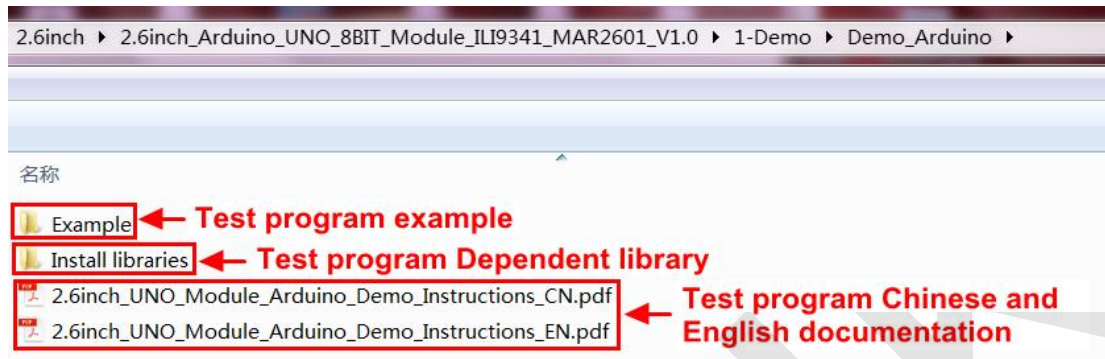
D. Open the selected sample project, compile and download.

The specific operation methods for the Arduino test program relying on library

copy, compile and download are as follows:

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

E. If the LCD module displays characters and graphics normally, the program runs

successfully；

2. C51 instructions

Wiring instructions:

See the interface description for pin assignments.

STC12C5A60S2 microcontroller test program wiring instructions		
Number	**Module Pin**	**Corresponding to STC12 development board wiring pin**
1	**5V**	5V
2	**3V3**	3V3
3	**GND**	GND
4	**LCD_D0**	P20
5	**LCD_D1**	P21
6	**LCD_D2**	P22
7	**LCD_D3**	P23
8	**LCD_D4**	P24
9	**LCD_D5**	P25

10	LCD_D6	P26
11	LCD_D7	P27
12	LCD_RST	P33
13	LCD_CS	P13
14	LCD_RS	P12
15	LCD_WR	P11
16	LCD_RD	P10
17	SD_SS	No need to connect
18	SD_DI	No need to connect
19	SD_DO	No need to connect
20	SD_SCK	No need to connect

Operating Steps：

A. Connect the LCD module and the C51 MCU according to the above wiring instructions, and power on;

B. Open the directory where the C51 test program is located and select the example to be tested, as shown below:

(Please refer to the test program description document for test program description)

C. Open the selected test program project, compile and download;

detailed description of the C51 test program compilation and download can be found in the following document:

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

D. If the LCD module displays characters and graphics normally, the program runs successfully；

3. STM32 instructions

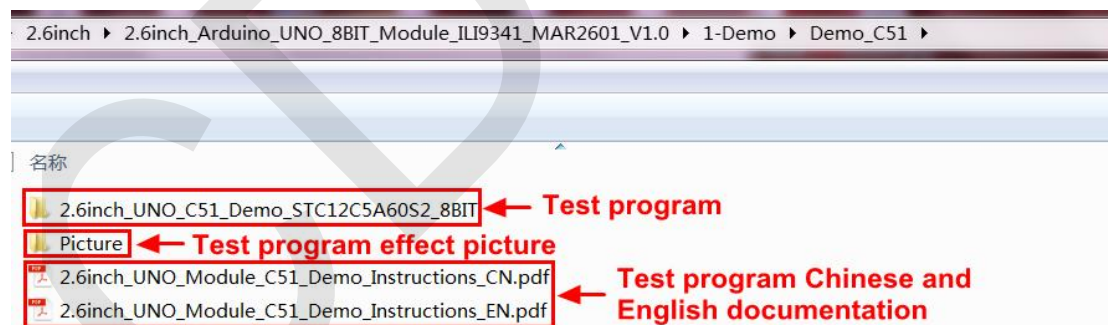**Wiring instructions:**

See the interface description for pin assignments.

STM32F103RCT6 microcontroller test program wiring		
Number	**Module Pin**	**Corresponding to MiniSTM32 development board wiring pin**
1	**5V**	5V
2	**3V3**	3.3V
3	**GND**	GND
4	**LCD_D0**	PB8
5	**LCD_D1**	PB9
6	**LCD_D2**	PB10
7	**LCD_D3**	PB11
8	**LCD_D4**	PB12
9	**LCD_D5**	PB13
10	**LCD_D6**	PB14
11	**LCD_D7**	PB15
12	**LCD_RST**	PC10
13	**LCD_CS**	PC9
14	**LCD_RS**	PC8
15	**LCD_WR**	PC7
16	**LCD_RD**	PC6
17	**SD_SS**	No need to connect
18	**SD_DI**	No need to connect
19	**SD_DO**	No need to connect
20	**SD_SCK**	No need to connect

STM32F103ZET6 microcontroller test program wiring		
Number	**Module Pin**	**Corresponding to Elite STM32 development board wiring pin**
1	**5V**	5V
2	**3V3**	3.3V
3	**GND**	GND

4	LCD_D0	PF8
5	LCD_D1	PF9
6	LCD_D2	PF10
7	LCD_D3	PF11
8	LCD_D4	PF12
9	LCD_D5	PF13
10	LCD_D6	PF14
11	LCD_D7	PF15
12	LCD_RST	PC10
13	LCD_CS	PC9
14	LCD_RS	PC8
15	LCD_WR	PC7
16	LCD_RD	PC6
17	SD_SS	No need to connect
18	SD_DI	No need to connect
19	SD_DO	No need to connect
20	SD_SCK	No need to connect

STM32F407ZGT6 microcontroller test program wiring		
Number	**Module Pin**	**Corresponding to Explorer STM32F4 development board wiring pin**
1	5V	5V
2	3V3	3.3V
3	GND	GND
4	LCD_D0	PG8
5	LCD_D1	PG9
6	LCD_D2	PG10
7	LCD_D3	PG11
8	LCD_D4	PG12
9	LCD_D5	PG13
10	LCD_D6	PG14
11	LCD_D7	PG15
12	LCD_RST	PC10

13	LCD_CS	PC9
14	LCD_RS	PC8
15	LCD_WR	PC7
16	LCD_RD	PC6
17	SD_SS	No need to connect
18	SD_DI	No need to connect
19	SD_DO	No need to connect
20	SD_SCK	No need to connect

STM32F429IGT6 microcontroller test program wiring		
Number	**Module Pin**	**Corresponding to Apollo STM32F4/F7 development board wiring pin**
1	5V	5V
2	3V3	3.3V
3	GND	GND
4	LCD_D0	PE8
5	LCD_D1	PE9
6	LCD_D2	PE10
7	LCD_D3	PE11
8	LCD_D4	PE12
9	LCD_D5	PE13
10	LCD_D6	PE14
11	LCD_D7	PE15
12	LCD_RST	PC10
13	LCD_CS	PC9
14	LCD_RS	PC8
15	LCD_WR	PC7
16	LCD_RD	PC6
17	SD_SS	No need to connect
18	SD_DI	No need to connect
19	SD_DO	No need to connect
20	SD_SCK	No need to connect

Operating Steps：

A. Connect the LCD module and the STM32 MCU according to the above wiring instructions, and power on;

B. Open the directory where the STM32 test program is located and select the example to be tested, as shown below:

(Please refer to the test program description document for test program description)
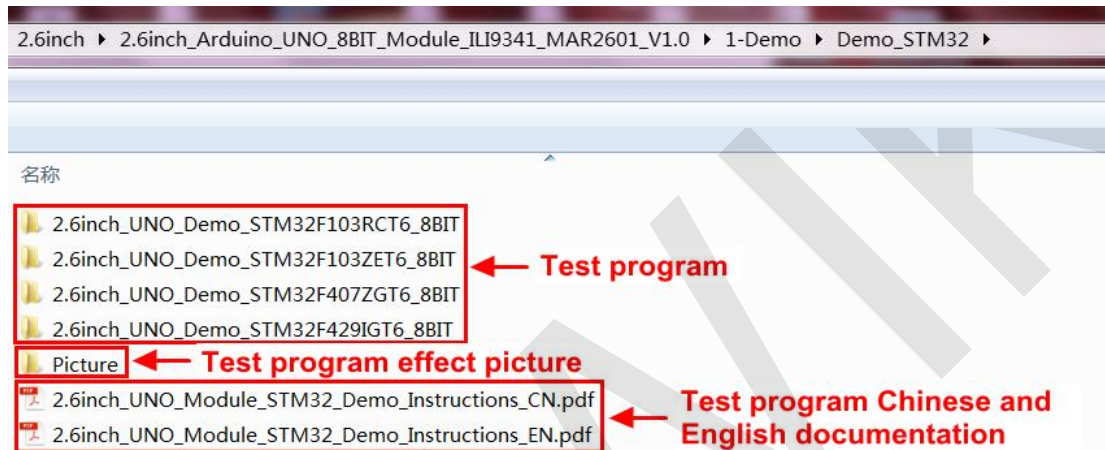
C. Open the selected test program project, compile and download;

detailed description of the STM32 test program compilation and download can be found in the following document:

http://www.lcdwiki.com/res/PublicFile/STM32_Keil_Use_Illustration_EN.pdf

D. If the LCD module displays characters and graphics normally, the program runs successfully；
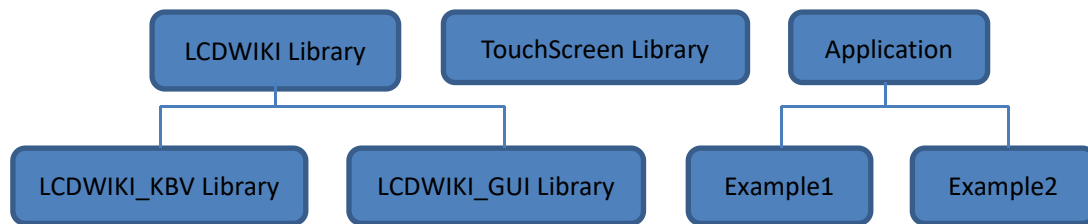
Software Description

1. Code Architecture

A. Arduino code architecture description

The code architecture is shown below:
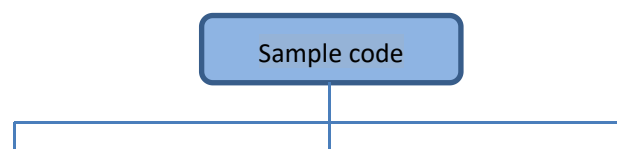
Arduino's test program code consists of three parts: LCDWIKI library, TouchScreen library and application code;

The LCDWIKI library contains two parts: LCDWIKI_KBV library and LCDWIKI_GUI library;

The application contains several test examples, each with different test content;

LCDWIKI_KBV is the underlying library, which is associated with hardware. It is mainly responsible for operating registers, including hardware module initialization, data and command transmission, pixel coordinates and color settings, display mode configuration, etc;

LCDWIKI_GUI is the middle layer library, which is responsible for drawing graphics and displaying characters using the API provided by the underlying library;

TouchScreen is the underlying library of touch screens, mainly responsible for touch interrupt detection, touch data sampling and AD conversion,Touch data transmission;

The application is to use the API provided by the LCDWIKI library and the TouchScreen library to write some test examples and implement Some aspect of the test function;

B. C51 and STM32 code architecture description

The code architecture is shown below:
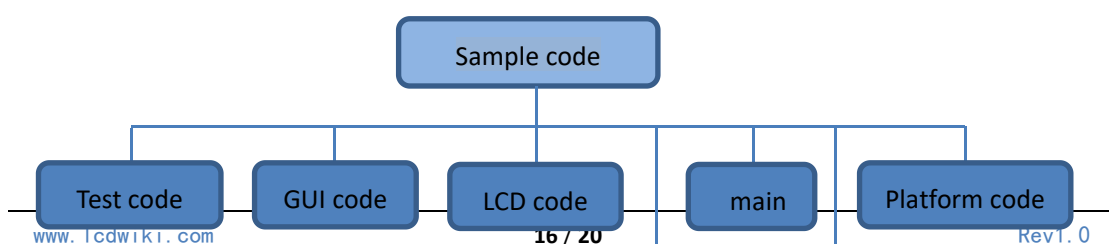
Key code   Led code

The Demo API code for the main program runtime is included in the test code;

LCD initialization and related bin parallel port write data operations are included in the

LCD code;

Drawing points, lines, graphics, and Chinese and English character display related

operations are included in the GUI code;

The main function implements the application to run;

Platform code varies by platform;

The key processing related code is included in the key code (the C51 platform does

not have a button processing code);

The code related to the led configuration operation is included in the led code;

2. GPIO definition description

A. Arduino test program GPIO definition description

The module is plugged into the Arduino UNO and Mage2560, so it is not allowed to

modify the GPIO port definition.

B. C51 test program GPIO definition description

C51 test program lcd screen GPIO definition is placed in the lcd.h file, as shown

below:

```
//IO connect
#define  LCD_DataPortH P2     //High 8-bit data port,Only use the upper 8 bits in 8-bit mode
#define  LCD_DataPortL P0     //Low 8-bit data port,The lower 8 bits in 8-bit mode can be le
sbit LCD_RS = P1^2;      //Data/command switching
sbit LCD_WR = P1^1;      //Write control
sbit LCD_RD = P1^0;      //read control
sbit LCD_CS = P1^3;    //Chip Select
sbit LCD_RESET = P3^3;       //reset
//sbit LCD_BL=P3^2;    //Backlight control,If you do not need control, connect 3.3V
```

Parallel pin definition needs to select the whole set of GPIO port groups, such as P0,

P2, etc., so that when transferring data, the operation is convenient.Other pins can be

defined as any free GPIO.

C. STM32 test program GPIO definition description

STM32 test program lcd screen GPIO definition is placed in the lcd.h file, as shown below (to STM32F103RCT6 Test procedure as an example)：

```
/////////////////////////////////////////////////////////////
//----------------LCD port definition----------------
#define GPIO_TYPE   GPIOC  //GPIO group type
//#define LED      4       //Backlight control pin  PC4
#define LCD_CS    9        //Chip Select pin       PC9
#define LCD_RS    8        //Data/command switching pin PC8
#define LCD_RST  10        //reset pin             PC10
#define LCD_WR    7        //write pin             PC7
#define LCD_RD    6        //read pin              PC6
```

```
//PB0~15,As the data line
//note:If using an 8-bit mode data bus,Then the LCD
//Example:If connected to 8-bit mode, this example i
//Example:If it is 16-bit mode:DB0-DB7 are connected
#define DATAOUT(x) GPIOB->ODR=x; //data output
#define DATAIN     GPIOB->IDR;   //data input
```

Data parallel port pin definition needs to select a complete set of GPIO port groups, such as PB, when transferring data, it is convenient to operate.

Other pins can be defined as any free GPIO.

3. Parallel port communication code implementation

A. Arduino test program parallel port communication code implementation

The relevant code is implemented in the mcu_8bit_magic.h file of the LCDWIKI_KBV library, as shown in the figure below:

```
#define BMASK       0x03
#define DMASK       0xFC
#define write8(d) { PORTD = (PORTD & ~DMASK) | ((d) & DMASK); PORTB = (PORTB & ~BMASK) | ((d) & BMASK); WR_STROBE; }
// #define write16(d) {uint8_t h = (d)>>8, l = d; write8(d); write8(l);}
#define read8(dst) { RD_ACTIVE; DELAY7; dst = (PIND & DMASK) | (PINB & BMASK); RD_IDLE; }
// #define read16(dst) { uint8_t hi; read8(hi); read8(dst); dst |= (hi << 8); }

#define setWriteDir() { DDRD |=  DMASK; DDRB |=  BMASK; }
#define setReadDir()  { DDRD &= ~DMASK; DDRB &= ~BMASK; }


#define write16(d) { uint8_t h = (d)>>8, l = d; write8(h); write8(l); }
#define read16(dst) { uint8_t hi; read8(hi); read8(dst); dst |= (hi << 8); }
```

Implemented 8-bit and 16-bit commands and 8-bit and 16-bit data write and read.

B. C51 and STM32 test program parallel port communication code implementation

The relevant code is implemented in the LCD.c file as shown below:

```
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);//delay 1us
  data = DATAIN;
  LCD_RD_SET;
  LCD_CS_SET;
  return data;
}
```

Implemented 8-bit and 16-bit commands and 8-bit and 16-bit data write and read.

4. touch screen calibration instructions

A. Arduino test program touch screen calibration instructions

Arduino touch screen calibration needs to run the TouchScreen_Calibr program first (see the test program directory), and then calibrate according to the prompts. After the calibration is passed, the calibration parameters displayed on the screen need to be written into the corresponding test program, as shown in the following figure (using the touch_pen test program example):

```
//param calibration from kbv
#define TS_MINX 124 => LEFT
#define TS_MAXX 906 => RT

#define TS_MINY 83 => TOP
#define TS_MAXY 893 => BOT
```

B. C51 and STM32 test program touch screen calibration instructions

Since this module does not contain a dedicated touch IC, it is difficult to implement the touch function on the C51 and STM32.Therefore, the C51 and STM32 test programs do not have touch screen test items.

Common software

This set of test examples requires the display of Chinese and English, symbols and pictures, so the modulo software is used. There are two types of modulo software: Image2Lcd and PCtoLCD2002. Here is only the setting of the modulo software for the test program.

The **PCtoLCD2002** modulo software settings are as follows:

Dot matrix format select Dark code

the modulo mode select the progressive mode

Take the model to choose the direction (high position first)

Output number system selects hexadecimal number

Custom format selection C51 format

The specific setting method is as follows:

http://www.lcdwiki.com/Chinese_and_English_display_modulo_settings

Image2Lcd modulo software settings are shown below:

The Image2Lcd software needs to be set to horizontal, left to right, top to bottom, and low position to the front scan mode.