1.28inch LCD Module

Introduction

1.28inch LCD Display Module, IPS Screen, 65K RGB Colors, 240×240 Resolution, SPI Interface

Specification

- Operating voltage: 3.3V/5V
- Interface: SPI
- LCD type: IPS
- Controller: GC9A01
- Resolution: 240(H)RGB x 240(V)
- Display size: Φ32.4mm
- Pixel size: 0.135（H）x0.135（V）mm
- Dimension: 40.4×37.5(mm) Φ37.5(mm)

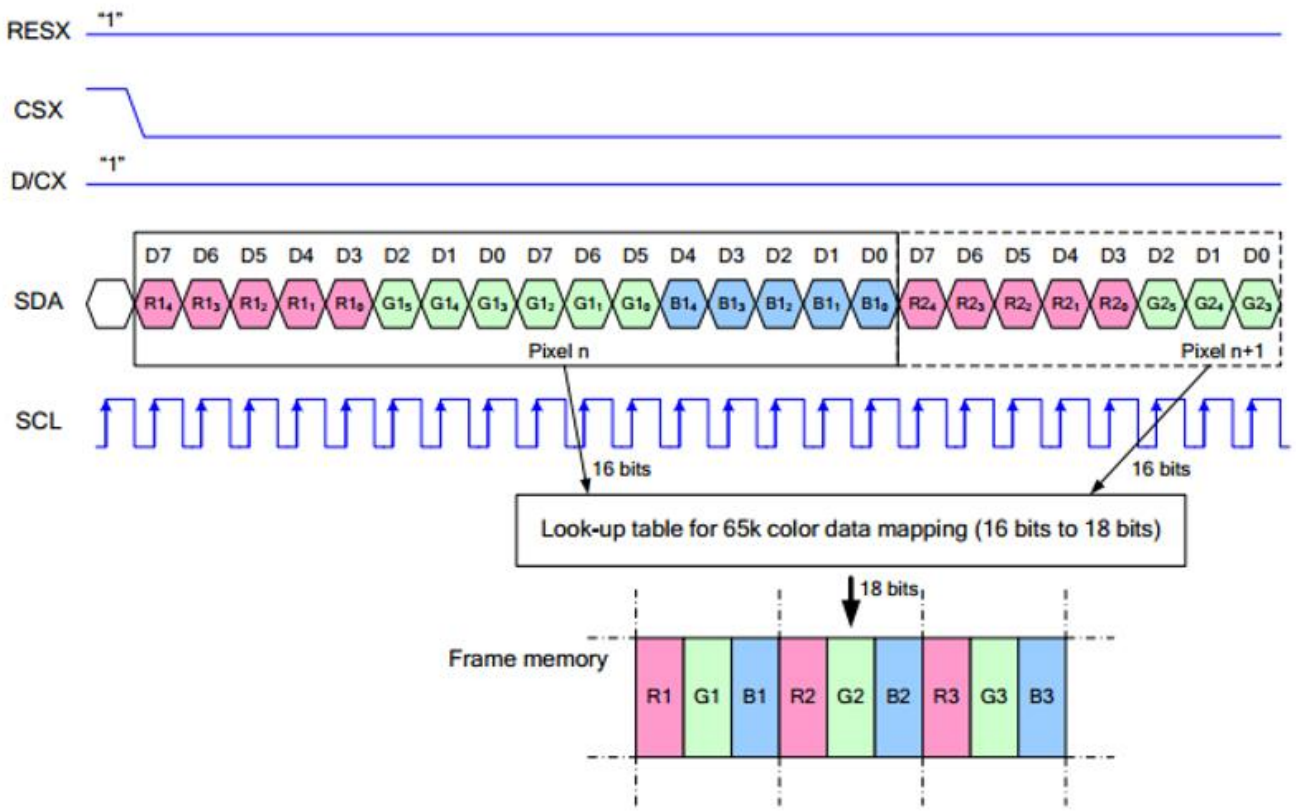
Pinout

PIN	Description
VCC	3.3V/5V Power input
GND	Ground
DIN	SPI data input
CLK	SPI clock input
CS	Chip selection, low active
DC	Data/Command control
RST	Reset
BL	Backlight

LCD and the controller

- The driver used in this LCD is GC9A01, with a resolution of 240RGB×240 dots and 129600 bytes of GRAM inside. This LCD supports 12-bits/16-bits/18-bits data bus by MCU interface, which are RGB444, RGB565, RGB666.
- For most LCD controllers, the communication method of the controller can be configured, they are usually using 8080 parallel interface, 3-line SPI, 4-line SPI, and other communication methods. This LCD uses a 4-line SPI interface for reducing GPIO and fast speed.LCD
- If you are wondering which point is the first pixel of the screen (because the screen is round), you can understand it as a square screen with an inscribed circle drawn in it, and it only displays the content in this inscribed circle. The pixels in other locations are simply discarded (just like most round smartwatches on the market)

Working Protocol


RESX
"1"
CSX
D/CX
"1"
SDA
D7 D6 D5 D4 D3 D2 D1 D0 D7 D6 D5 D4 D3 D2 D1 D0 D7 D6 D5 D4 D3 D2 D1 D0
R1₄ R1₃ R1₂ R1₁ R1₀ G1₅ G1₄ G1₃ G1₂ G1₁ G1₀ B1₄ B1₃ B1₂ B1₁ B1₀ R2₄ R2₃ R2₂ R2₁ R2₀ G2₅ G2₄ G2₃
Pixel n
Pixel n+1
SCL
16 bits
16 bits
Look-up table for 65k color data mapping (16 bits to 18 bits)
18 bits
Frame memory
R1 G1 B1 R2 G2 B2 R3 G3 B3

Note: Different from the traditional SPI protocol, the data line from the slave to the master is hidden since the device only has display requirement.
RESX Is the reset pin, it should be low when powering the module and be higher at other times; ;
CSX is slave chip select, when CS is low, the chip is enabled.
D/CX is data/command control pin, when DC = 0, write command, when DC = 1, write data
SDA is the data pin for transmitting RGB data, it works as the MOSI pin of SPI interface;
SCL worka s the SCLK pins of SPI interface.
SPI communication has data transfer timing, which is combined by CPHA and CPOL.
CPOL determines the level of the serial synchronous clock at idle state. When CPOL = 0, the level is Low. However, CPOL has little effect to the transmission.
CPHA determines whether data is collected at the first clock edge or at the second clock edge of serial synchronous clock; when CPHL = 0, data is collected at the first clock edge.
There are 4 SPI communication modes. SPI0 is commonly used, in which CPHL = 0, CPOL = 0.

Hardware connection

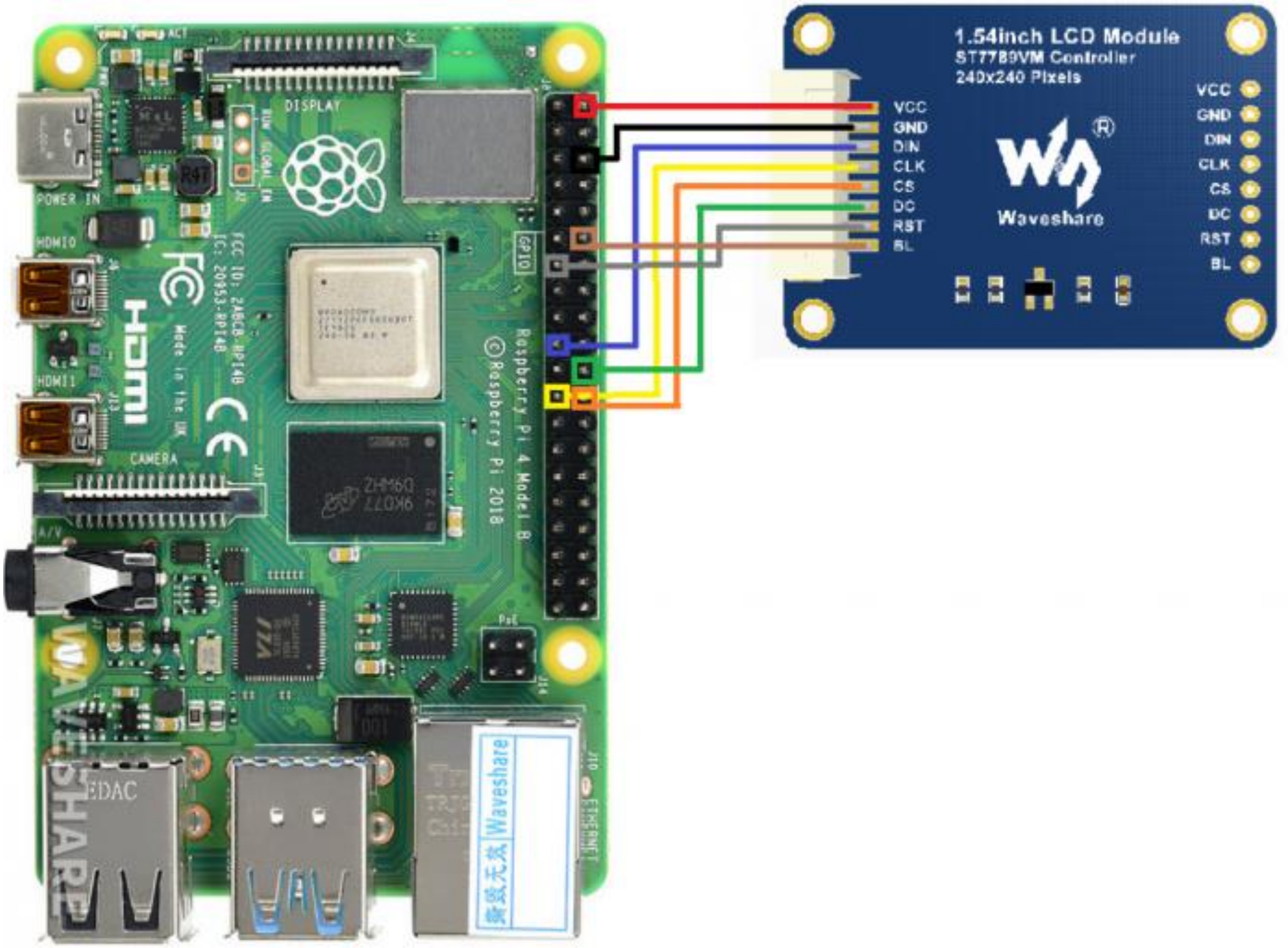
Please connect the LCD to your Raspberry Pi by the 8PIn cable according to the table below

Connect to Raspberry Pi

LCD	Raspberry Pi	
	BCM2835	Board
VCC	5V	5V
GND	GND	GND

DIN	MOSI	19
CLK	SCLK	23
CS	CE0	24
DC	25	22
RST	27	13
BL	18	12

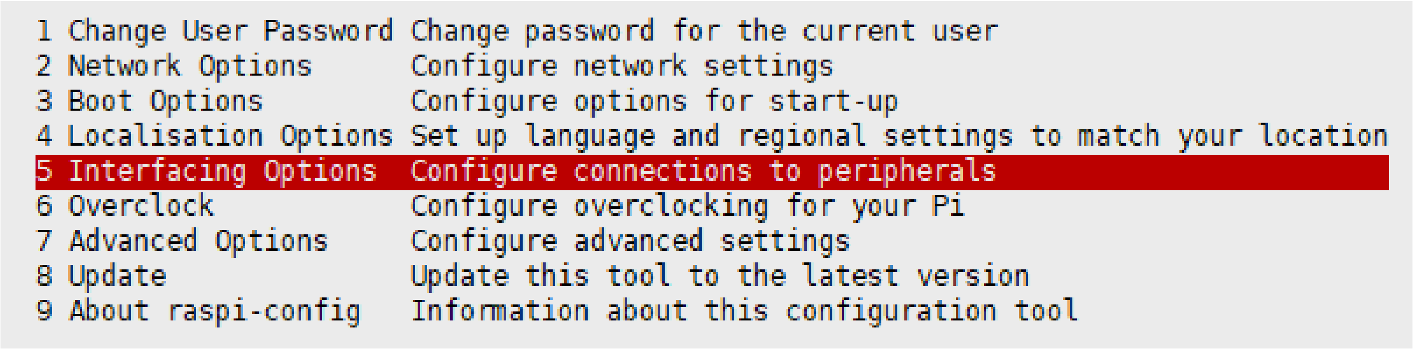
The color of actual cable may be different with the figure here, please connect them according to the pins instead of color.



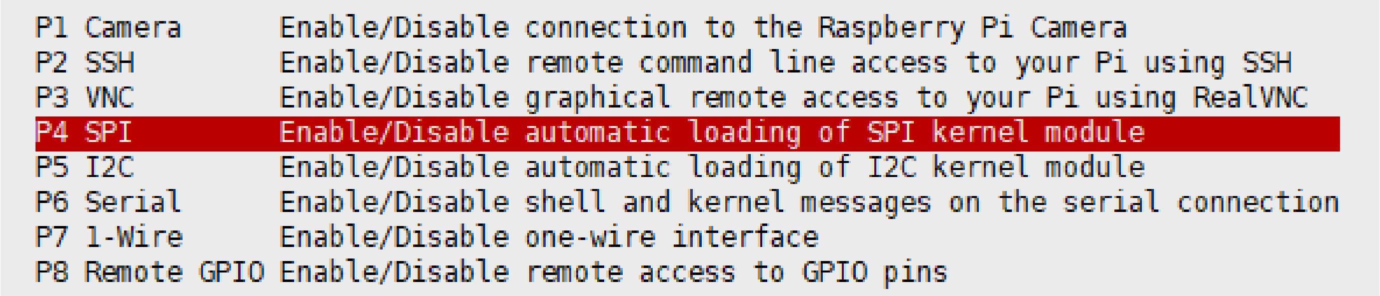
Enable SPI interface

- Open terminal, use command to enter the configuration page

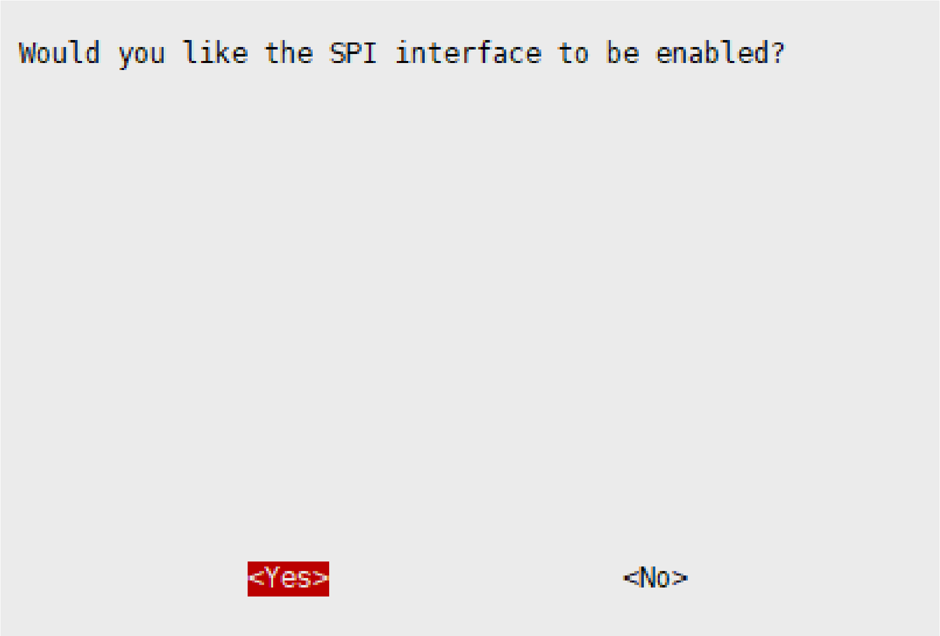
```
sudo raspi-config
Choose Interfacing Options -> SPI -> Yes  to enable SPI interface
```



```
1 Change User Password Change password for the current user
2 Network Options      Configure network settings
3 Boot Options         Configure options for start-up
4 Localisation Options Set up language and regional settings to match your location
5 Interfacing Options  Configure connections to peripherals
6 Overclock            Configure overclocking for your Pi
7 Advanced Options     Configure advanced settings
8 Update               Update this tool to the latest version
9 About raspi-config   Information about this configuration tool
```



```
P1 Camera      Enable/Disable connection to the Raspberry Pi Camera
P2 SSH         Enable/Disable remote command line access to your Pi using SSH
P3 VNC         Enable/Disable graphical remote access to your Pi using RealVNC
P4 SPI         Enable/Disable automatic loading of SPI kernel module
P5 I2C         Enable/Disable automatic loading of I2C kernel module
P6 Serial      Enable/Disable shell and kernel messages on the serial connection
P7 1-Wire      Enable/Disable one-wire interface
P8 Remote GPIO Enable/Disable remote access to GPIO pins
```



```
Would you like the SPI interface to be enabled?
```

Reboot Raspberry Pi：

```
sudo reboot
```

Please make sure that SPI interface was not used by other devices

Install Libraries

- Install BCM2835 libraries

```
wget http://www.airspayce.com/mikem/bcm2835/bcm2835-1.68.tar.gz
```

```
tar zxvf bcm2835-1.68.tar.gz
cd bcm2835-1.68/
sudo ./configure
sudo make
sudo make check
sudo make install
#For more details, please refer to http://www.airspayce.com/mikem/bcm2835/
```

- Install wiringPi libraries

```
sudo apt-get install wiringpi
```

For the version of the Raspberry Pi system after May 2019 (the OS version earlier than this date doesn't need to be executed), an upgrade may be required：

```
wget https://project-downloads.drogon.net/wiringpi-latest.deb
sudo dpkg -i wiringpi-latest.deb
gpio -v
#You will get 2.52 information if you install it correctly
```

- Install Python libraries

```
#python2
sudo apt-get update
sudo apt-get install python-pip
sudo apt-get install python-pil
sudo apt-get install python-numpy
sudo pip install RPi.GPIO
sudo pip install spidev
#python3
sudo apt-get update
sudo apt-get install python3-pip
sudo apt-get install python3-pil
sudo apt-get install python3-numpy
sudo pip3 install RPi.GPIO
sudo pip3 install spidev
```

Download Examples

Open Raspberry Pi terminal and run the following command

```
sudo apt-get install p7zip-full
sudo wget  https://www.waveshare.net/w/upload/a/a8/LCD_Module_RPI_code.7z
```

```
7z x LCD_Module_RPI_code.7z -O./LCD_Module_code
cd LCD_Module_code/RaspberryPi/
```

Run the demo codes

Please go into the RaspberryPi directory (demo codes) first and run the commands in terminal

C codes

- Re-compile the demo codes

```
cd c
sudo make clean
sudo make -j 8
```

This examples are made for multi-dusplay, you can input the type of the LCD when using.

```
sudo ./main <<type of LCD>>
```

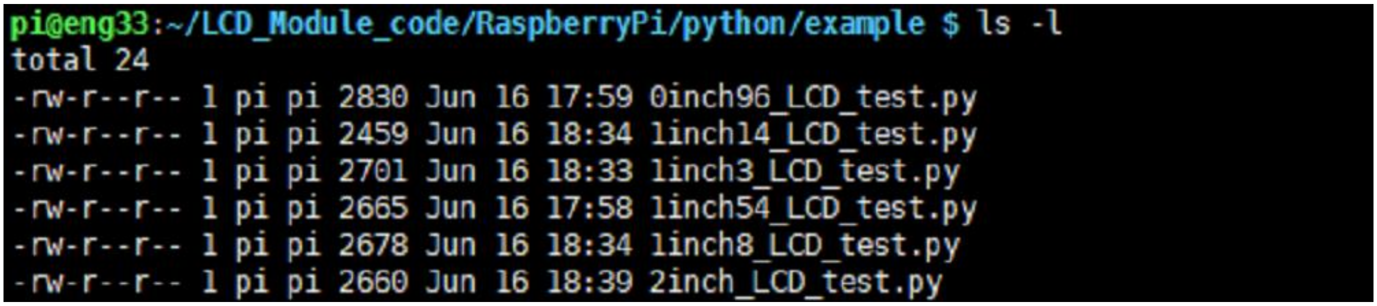
Use the command according to LCD: :

```
sudo ./main 0.96
sudo ./main 1.14
sudo ./main 1.28
sudo ./main 1.3
sudo ./main 1.54
sudo ./main 1.8
sudo ./main 2
```

python

- Enter the python directory and run ls -al

```
cd python/examples
ls -l
```



```
pi@eng33:~/LCD_Module_code/RaspberryPi/python/example $ ls -l
total 24
-rw-r--r-- 1 pi pi 2830 Jun 16 17:59 0inch96_LCD_test.py
-rw-r--r-- 1 pi pi 2459 Jun 16 18:34 1inch14_LCD_test.py
-rw-r--r-- 1 pi pi 2701 Jun 16 18:33 1inch3_LCD_test.py
-rw-r--r-- 1 pi pi 2665 Jun 16 17:58 1inch54_LCD_test.py
-rw-r--r-- 1 pi pi 2678 Jun 16 18:34 1inch8_LCD_test.py
-rw-r--r-- 1 pi pi 2660 Jun 16 18:39 2inch_LCD_test.py
```

You can check all the files which are listed in type:

0inch96_LCD_test.py	0.96inch LCD example
1inch14_LCD_test.py	1.14inch LCD example
1inch28_LCD_test.py	1.28inch LCD example
1inch3_LCD_test.py	1.3inch LCD example
1inch54_LCD_test.py	1.54inchLCD example
1inch8_LCD_test.py	1.8inch LCD example
2inch_LCD_test.py	2inch LCD example

- Run the example

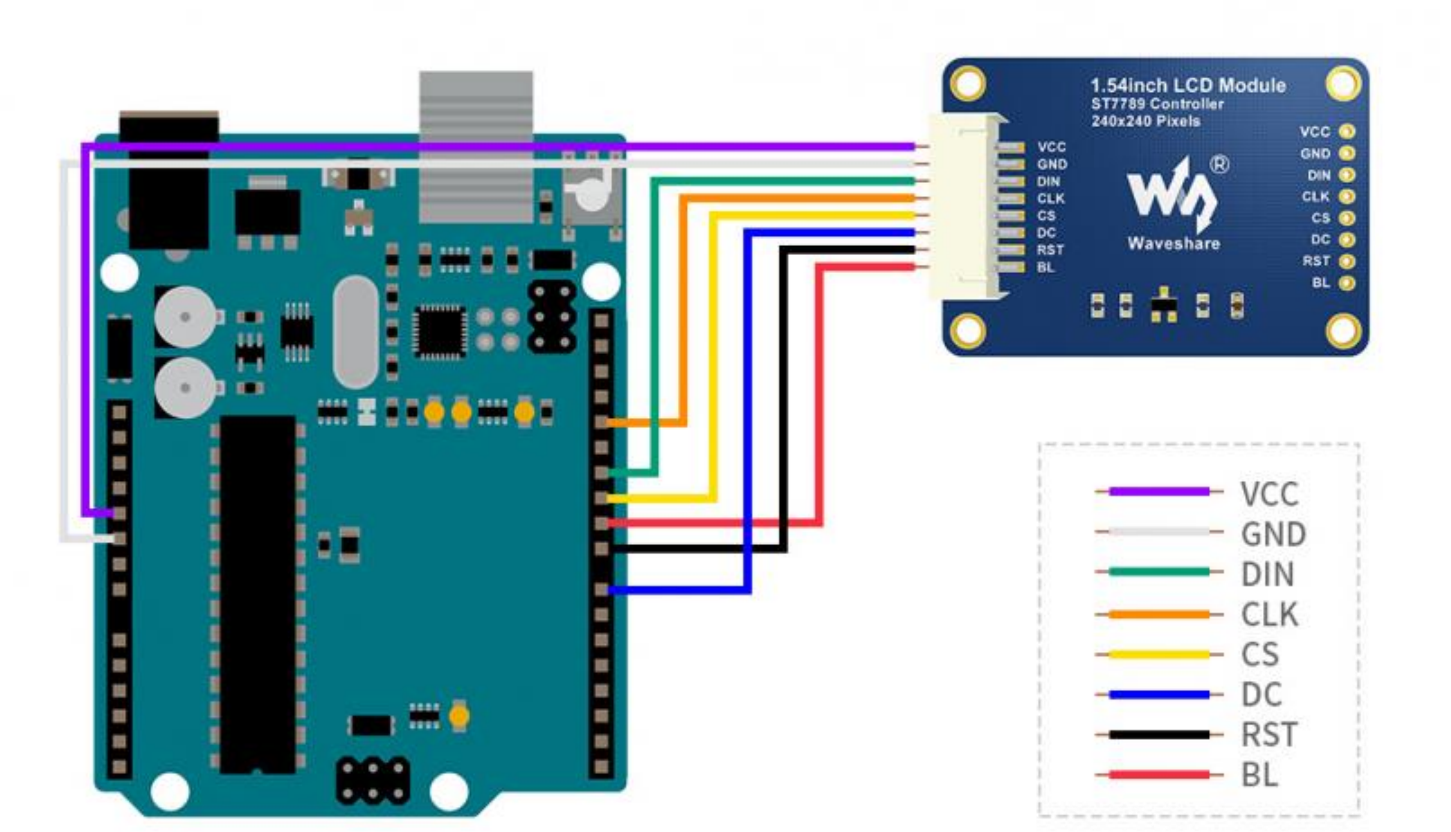
```
# python2
sudo python 0inch96_LCD_test.py
sudo python 1inch14_LCD_test.py
sudo python 1inch28_LCD_test.py
sudo python 1inch3_LCD_test.py
sudo python 1inch54_LCD_test.py
sudo python 1inch8_LCD_test.py
sudo python 2inch_LCD_test.py
# python3
sudo python3 0inch96_LCD_test.py
sudo python3 1inch14_LCD_test.py
sudo python3 1inch28_LCD_test.py
sudo python3 1inch3_LCD_test.py
sudo python3 1inch54_LCD_test.py
sudo python3 1inch8_LCD_test.py
sudo python3 2inch_LCD_test.py
```

The examples are tested in Arduino UNO, if you want to use other versions of the Arduino, you need to change the connection according to the actual boards.

Hardware Connection

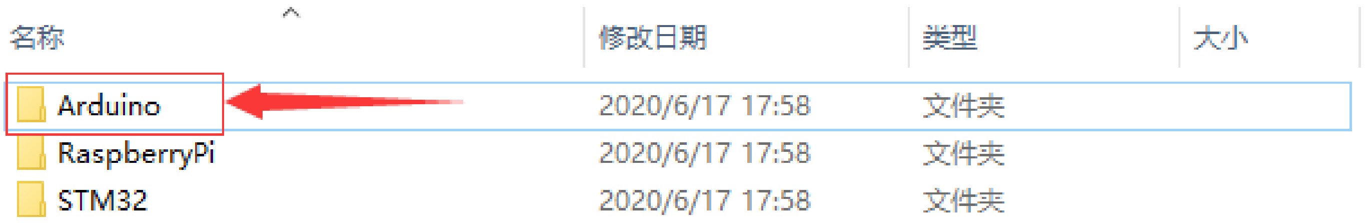
Arduino UNO连接引脚对应关系	
LCD	UNO
VCC	5V/3.3V
GND	GND
DIN	D11
CLK	D13
CS	D10
DC	D7

RST	D8
BL	D9



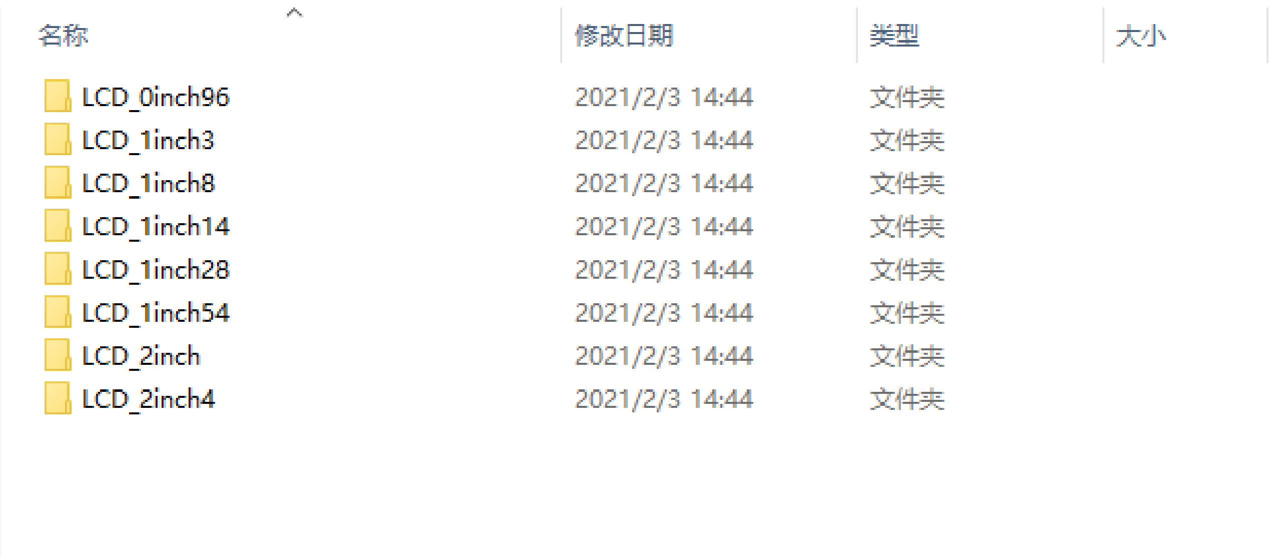
Run the example

Download the demo codes and unzip it. The Arduino project is located in ~/Arduino/...



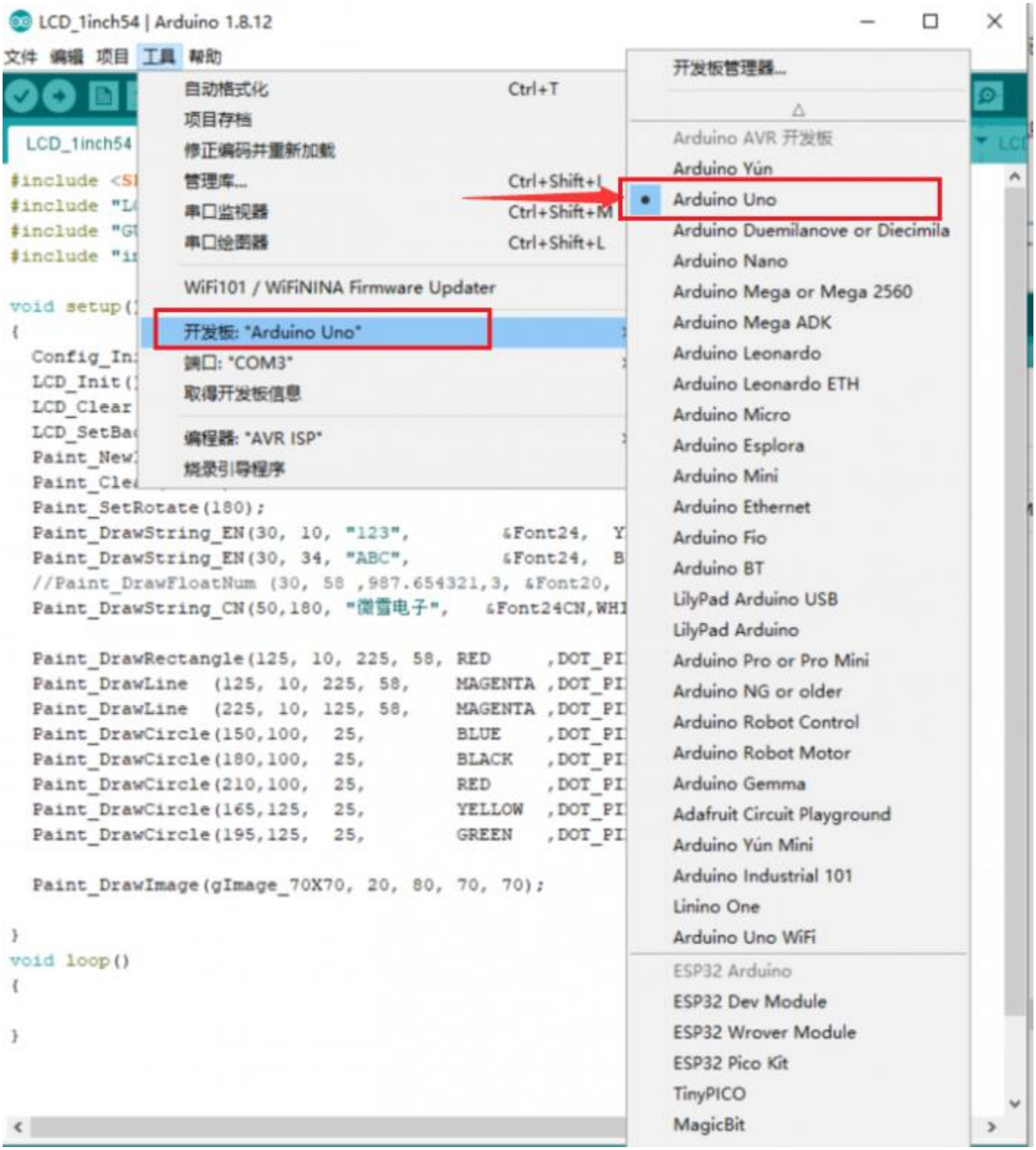
名称	修改日期	类型	大小
Arduino	2020/6/17 17:58	文件夹	
RaspberryPi	2020/6/17 17:58	文件夹	
STM32	2020/6/17 17:58	文件夹	

Run the project according to the actual display type



名称	修改日期	类型	大小
LCD_0inch96	2021/2/3 14:44	文件夹	
LCD_1inch3	2021/2/3 14:44	文件夹	
LCD_1inch8	2021/2/3 14:44	文件夹	
LCD_1inch14	2021/2/3 14:44	文件夹	
LCD_1inch28	2021/2/3 14:44	文件夹	
LCD_1inch54	2021/2/3 14:44	文件夹	
LCD_2inch	2021/2/3 14:44	文件夹	
LCD_2inch4	2021/2/3 14:44	文件夹	

For examples: 1.54inch LCD Module. Enter the LCD_1inch54 directory and run the LCD_1inch54.ino file
Run the project and choose Arduino UNO as Board



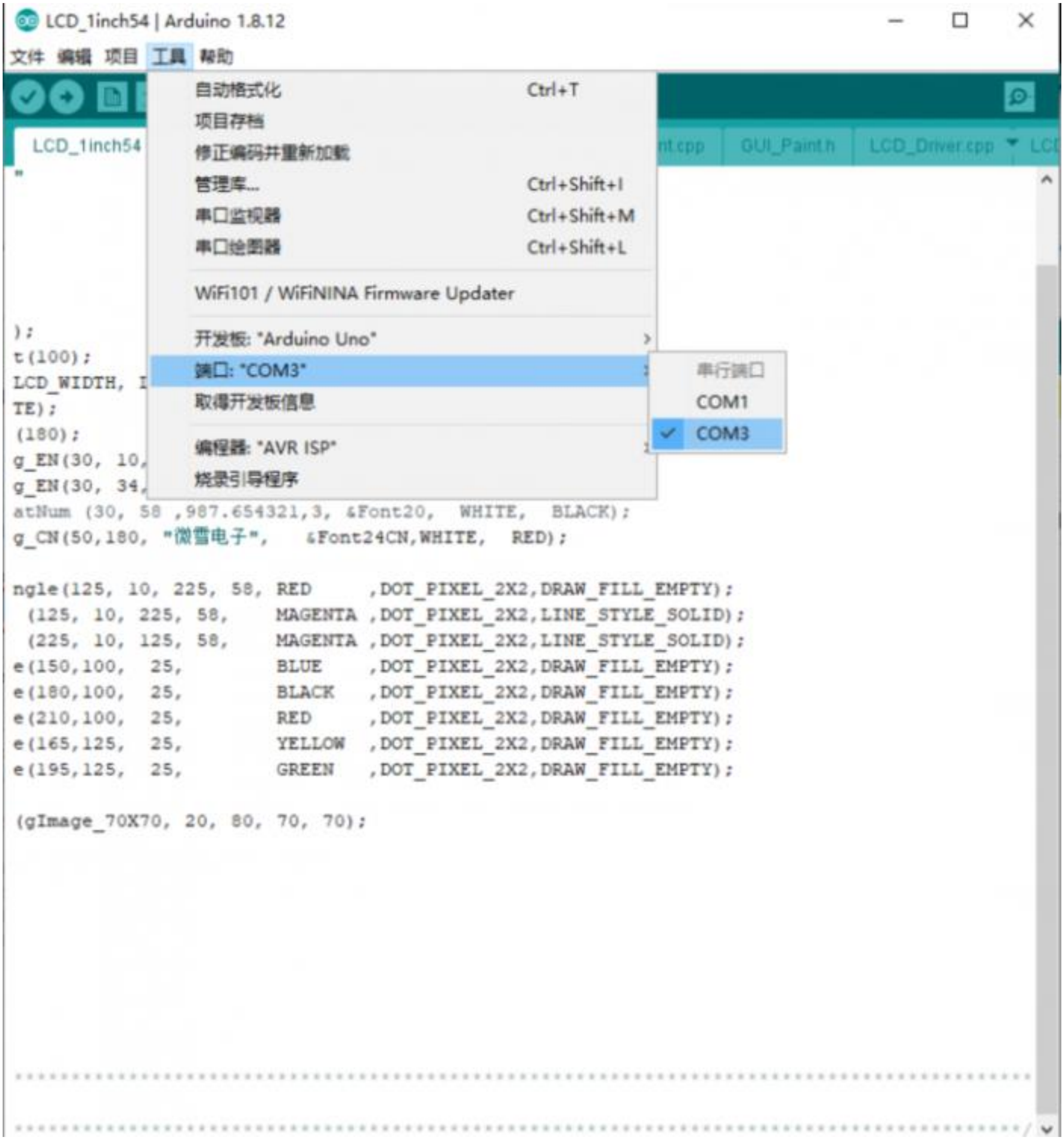
开发板: "Arduino Uno"

端口: "COM3"

编程器: "AVR ISP"

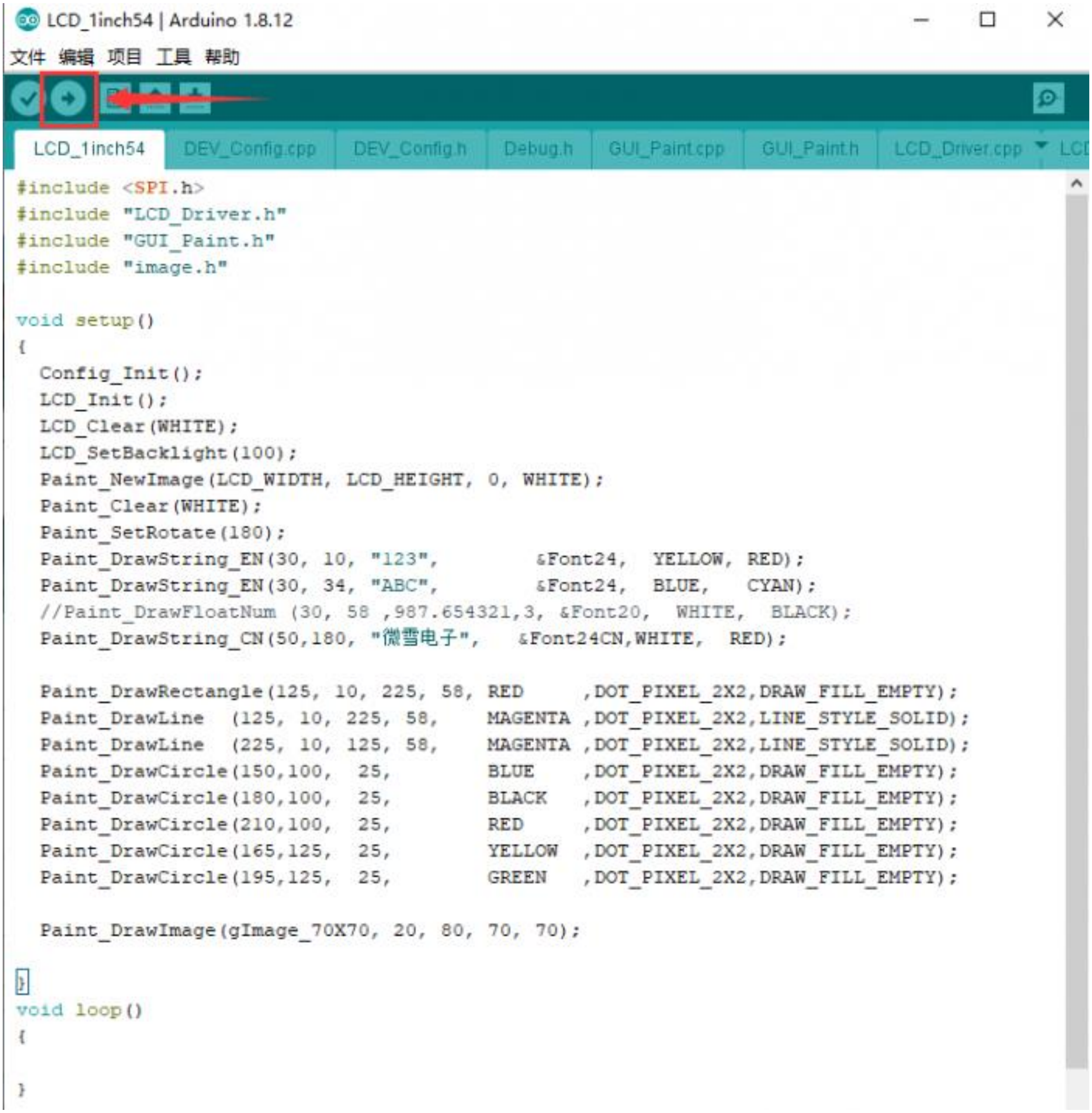
```
#include <S
#include "L
#include "G
#include "i

void setup(
{
  Config_In
  LCD_Init(
  LCD_Clear
  LCD_SetBa
  Paint_New
  Paint_Cle
  Paint_SetRotate(180);
  Paint_DrawString_EN(30, 10, "123",        &Font24,  Y
  Paint_DrawString_EN(30, 34, "ABC",        &Font24,  B
  //Paint_DrawFloatNum (30, 58 ,987.654321,3, &Font20,
  Paint_DrawString_CN(50,180, "微雪电子",   &Font24CN,WHI

  Paint_DrawRectangle(125, 10, 225, 58, RED     ,DOT_PI
  Paint_DrawLine  (125, 10, 225, 58,    MAGENTA ,DOT_PI
  Paint_DrawLine  (225, 10, 125, 58,    MAGENTA ,DOT_PI
  Paint_DrawCircle(150,100,  25,        BLUE    ,DOT_PI
  Paint_DrawCircle(180,100,  25,        BLACK   ,DOT_PI
  Paint_DrawCircle(210,100,  25,        RED     ,DOT_PI
  Paint_DrawCircle(165,125,  25,        YELLOW  ,DOT_PI
  Paint_DrawCircle(195,125,  25,        GREEN   ,DOT_PI

  Paint_DrawImage(gImage_70X70, 20, 80, 70, 70);

}
void loop()
{

}
```

Select the COM Port according to your Device Manager



```
"
);
t(100);
LCD_WIDTH, L
TE);
(180);
g_EN(30, 10,
g_EN(30, 34,
atNum (30, 58 ,987.654321,3, &Font20,  WHITE,  BLACK);
g_CN(50,180, "微雪电子",   &Font24CN,WHITE,  RED);

ngle(125, 10, 225, 58, RED     ,DOT_PIXEL_2X2,DRAW_FILL_EMPTY);
 (125, 10, 225, 58,    MAGENTA ,DOT_PIXEL_2X2,LINE_STYLE_SOLID);
 (225, 10, 125, 58,    MAGENTA ,DOT_PIXEL_2X2,LINE_STYLE_SOLID);
e(150,100,  25,        BLUE    ,DOT_PIXEL_2X2,DRAW_FILL_EMPTY);
e(180,100,  25,        BLACK   ,DOT_PIXEL_2X2,DRAW_FILL_EMPTY);
e(210,100,  25,        RED     ,DOT_PIXEL_2X2,DRAW_FILL_EMPTY);
e(165,125,  25,        YELLOW  ,DOT_PIXEL_2X2,DRAW_FILL_EMPTY);
e(195,125,  25,        GREEN   ,DOT_PIXEL_2X2,DRAW_FILL_EMPTY);

(gImage_70X70, 20, 80, 70, 70);
```

Compile and download it to your board



```
#include <SPI.h>
#include "LCD_Driver.h"
#include "GUI_Paint.h"
#include "image.h"

void setup()
{
  Config_Init();
  LCD_Init();
  LCD_Clear(WHITE);
  LCD_SetBacklight(100);
  Paint_NewImage(LCD_WIDTH, LCD_HEIGHT, 0, WHITE);
  Paint_Clear(WHITE);
  Paint_SetRotate(180);
  Paint_DrawString_EN(30, 10, "123",        &Font24,  YELLOW, RED);
  Paint_DrawString_EN(30, 34, "ABC",        &Font24,  BLUE,   CYAN);
  //Paint_DrawFloatNum (30, 58 ,987.654321,3, &Font20,  WHITE,  BLACK);
  Paint_DrawString_CN(50,180, "微雪电子",   &Font24CN,WHITE,  RED);

  Paint_DrawRectangle(125, 10, 225, 58, RED     ,DOT_PIXEL_2X2,DRAW_FILL_EMPTY);
  Paint_DrawLine  (125, 10, 225, 58,    MAGENTA ,DOT_PIXEL_2X2,LINE_STYLE_SOLID);
  Paint_DrawLine  (225, 10, 125, 58,    MAGENTA ,DOT_PIXEL_2X2,LINE_STYLE_SOLID);
  Paint_DrawCircle(150,100,  25,        BLUE    ,DOT_PIXEL_2X2,DRAW_FILL_EMPTY);
  Paint_DrawCircle(180,100,  25,        BLACK   ,DOT_PIXEL_2X2,DRAW_FILL_EMPTY);
  Paint_DrawCircle(210,100,  25,        RED     ,DOT_PIXEL_2X2,DRAW_FILL_EMPTY);
  Paint_DrawCircle(165,125,  25,        YELLOW  ,DOT_PIXEL_2X2,DRAW_FILL_EMPTY);
  Paint_DrawCircle(195,125,  25,        GREEN   ,DOT_PIXEL_2X2,DRAW_FILL_EMPTY);

  Paint_DrawImage(gImage_70X70, 20, 80, 70, 70);

}
void loop()
{

}
```

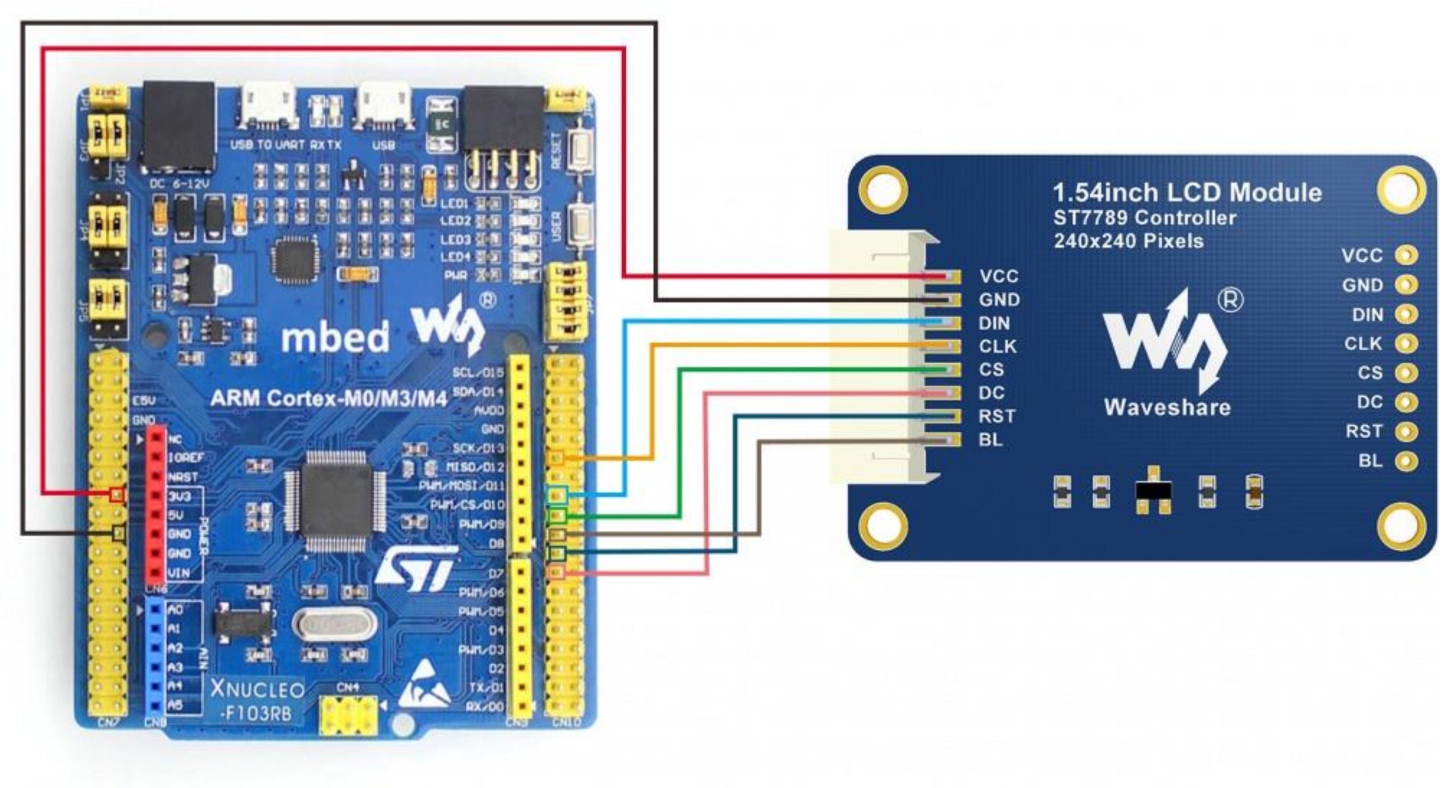
Hardware Coonnection

The examples are based on STM32F103RBT6 as well as the connection table. If you want to use other MCU, you need to port the project and change the connection according to the actual hardware.

Connect to STM32F103RBT6	
LCD	STM32
VCC	3.3V/5V
GND	GND

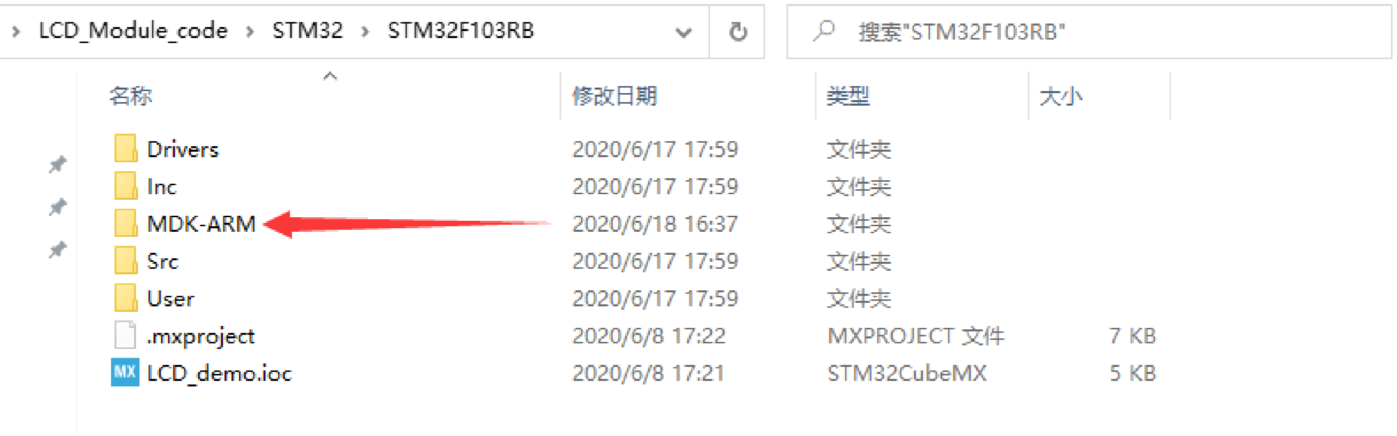
DIN	PA7
CLK	PA5
CS	PB6
DC	PA8
RST	PA9
BL	PC7

Use Waveshare XNUCLEO-F103RB as examples



About the examples

The examples use HAL libraries. Download demo codes, unzip, and find the STM32 projects. Open LCD_demo.uvprojx which is located in STM32\STM32F103RBT6\MDK-ARM directory by Keil project

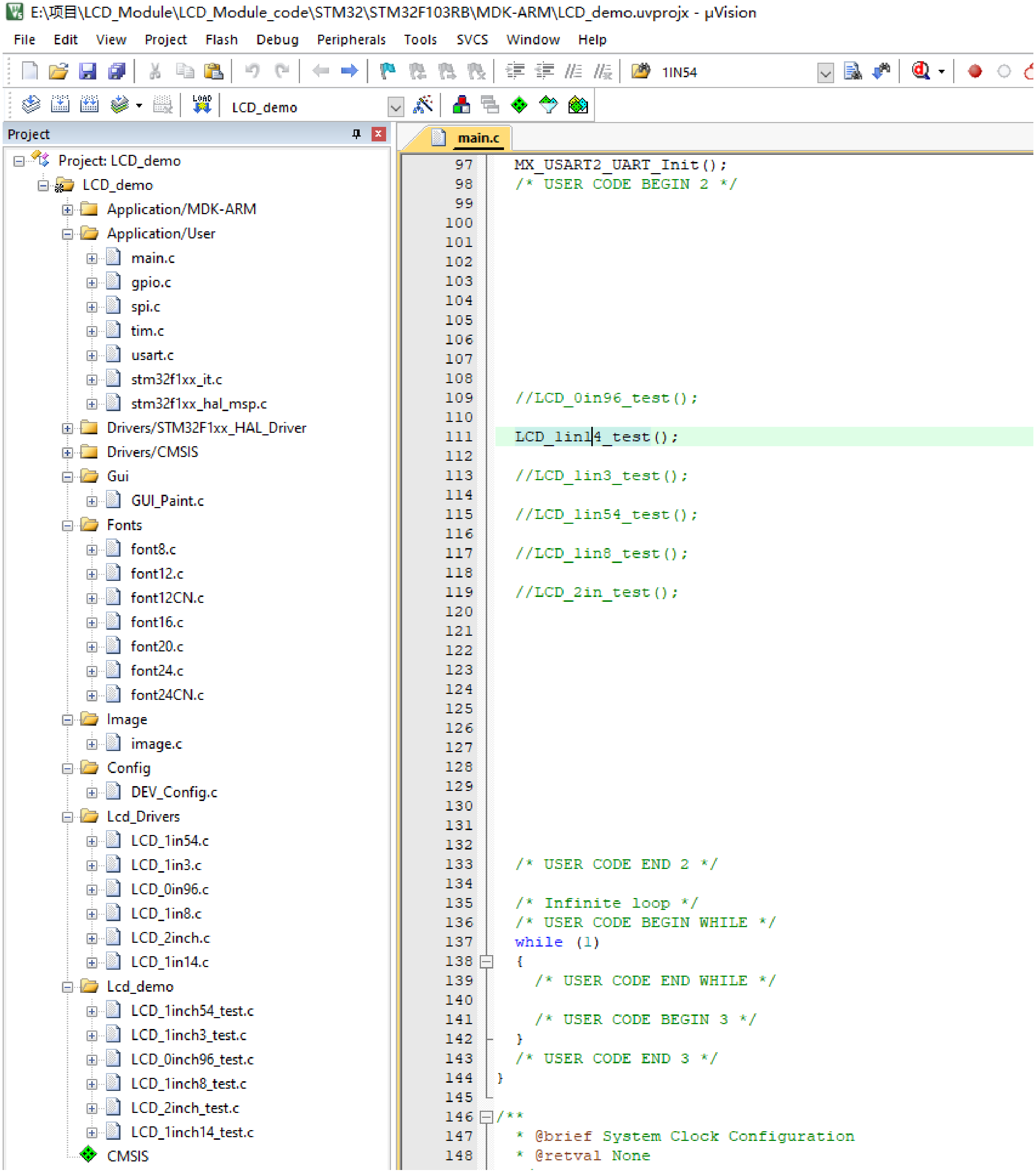


› LCD_Module_code › STM32 › STM32F103RB

搜索"STM32F103RB"

名称	修改日期	类型	大小
Drivers	2020/6/17 17:59	文件夹	
Inc	2020/6/17 17:59	文件夹	
MDK-ARM	2020/6/18 16:37	文件夹	
Src	2020/6/17 17:59	文件夹	
User	2020/6/17 17:59	文件夹	
.mxproject	2020/6/8 17:22	MXPROJECT 文件	7 KB
LCD_demo.ioc	2020/6/8 17:21	STM32CubeMX	5 KB

Open main.c file, you can configure the types for actual displays, recompile the project and download it to your board.



```
  MX_USART2_UART_Init();
  /* USER CODE BEGIN 2 */










  //LCD_0in96_test();

  LCD_1in14_test();

  //LCD_1in3_test();

  //LCD_1in54_test();

  //LCD_1in8_test();

  //LCD_2in_test();













  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}

/**
  * @brief System Clock Configuration
  * @retval None
```

LCD_0in96_test()	0.96inch LCD example
LCD_1in14_test()	1.14inch LCD example
LCD_1in28_test()	1.28inch LCD example
LCD_1in3_test()	1.3inch LCD example
LCD_1in54_test()	1.54inch LCD example

LCD_1in8_test()	1.8inch LCD example
LCD_2in_test()	2inchLCDexample

Resources

Documents

- Schematic
- GC9A01A manual

demo codes

- Demo codes