1.54inch LCD Module

- Operating voltage: 3.3V/5V
- Interface: SPI
- LCD type: IPS
- Controller: ST7789
- Resolution: 240(H)RGB x 240(V)
- Display area: 27.72（H）x 27.72（V）mm
- Pixel size: 0.1155（H）x 0.1155（V）mm
- Dimension: 50 x 35(mm)

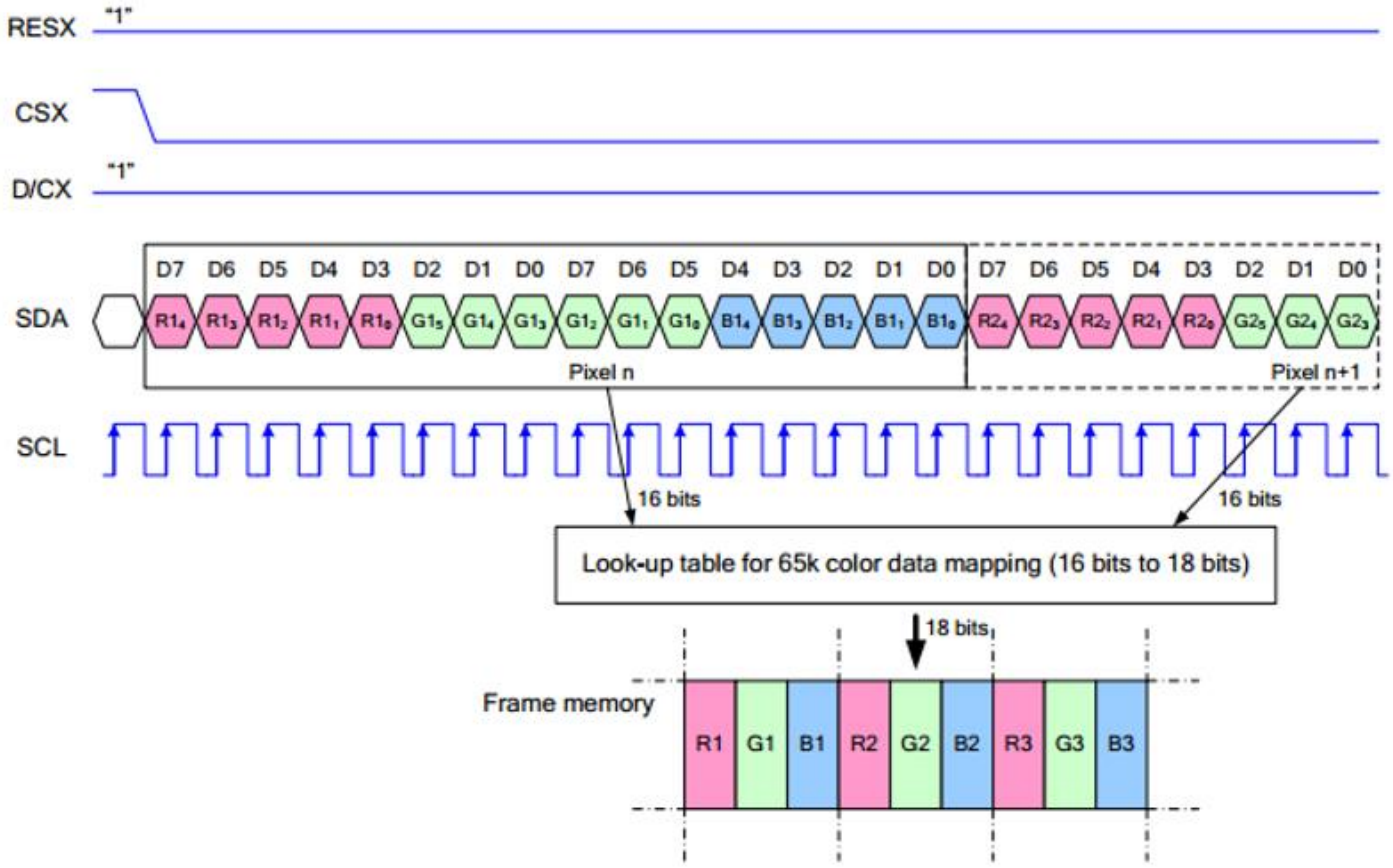
Pinout

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 ST7789VW is a single-chip controller/driver for 262K-color, graphic type TFT-LCD. It consists of 240 source line and 320 gate line driving circuits. The resolution of this LCD is 240(H)RGB x 240(V), it supports horizontal mode and vertical mode, and it doesn't use all the RAM of controller.
This LCD accepts 8-bits/9-bits/16-bits/18-bits parallel interface, that are RGB444, RGB565, RGB666. The color format used in demo codes is RGB565.
This LCD use 4-lines SPI interface for reducing GPIO and fast speed.LCD

Working Protocol



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.

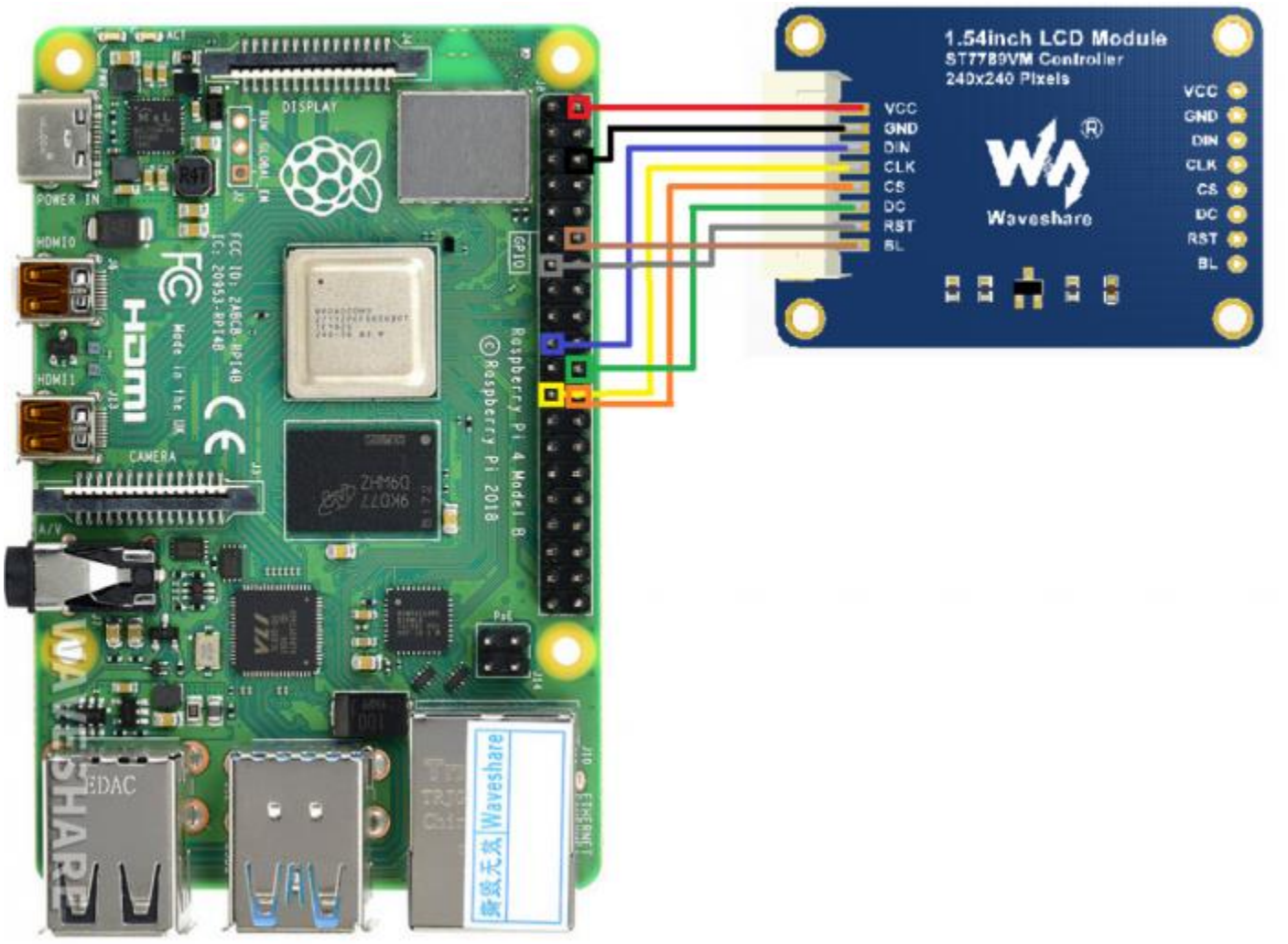
Hardware connection

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

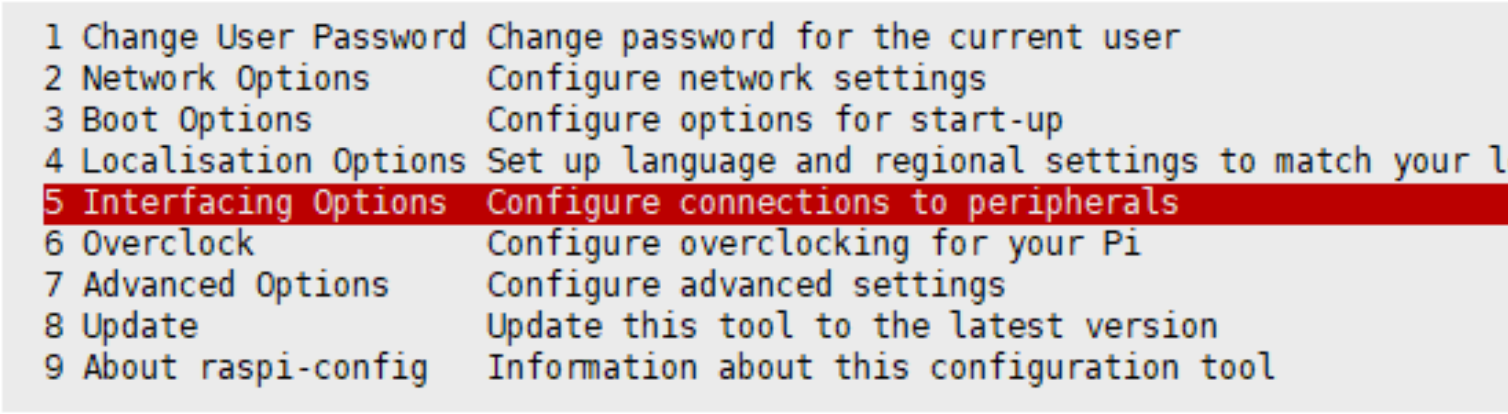
The color of actual cable may be different with the figure here, please connect them according to the pins instead of color.

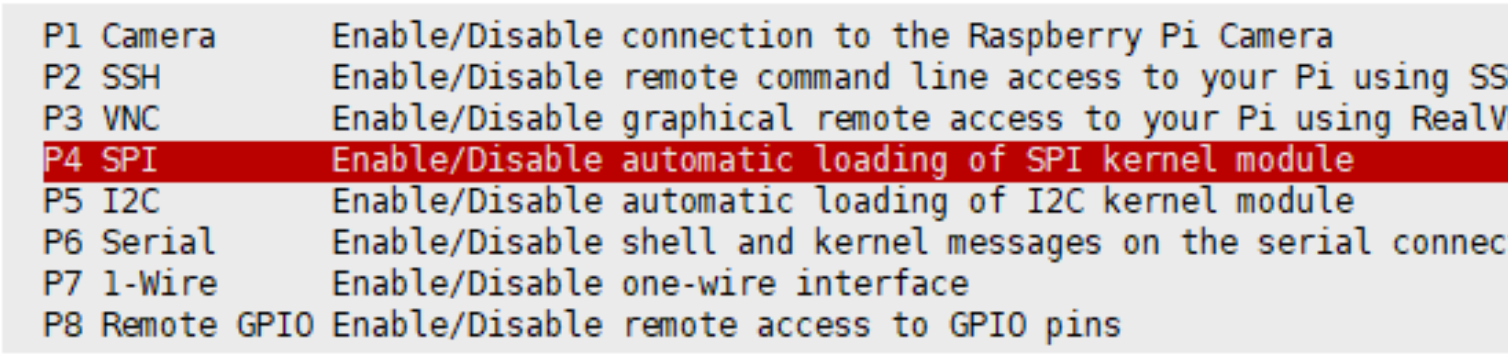


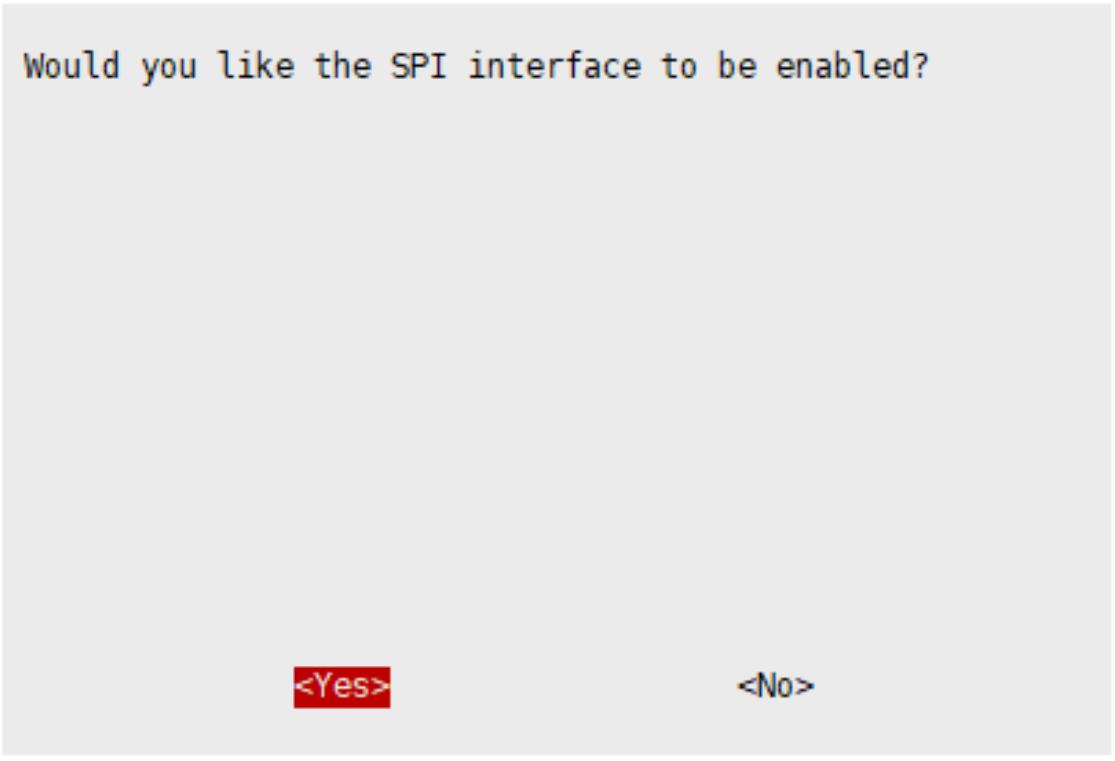
Enable SPI interface

- Open terminal, use command to enter the configuration page

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







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/bcm2835-1.60.tar.gz
tar zxvf bcm2835-1.60.tar.gz
cd bcm2835-1.60/
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 Pi 4, you need to update it:
cd /tmp
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.com/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>>
```

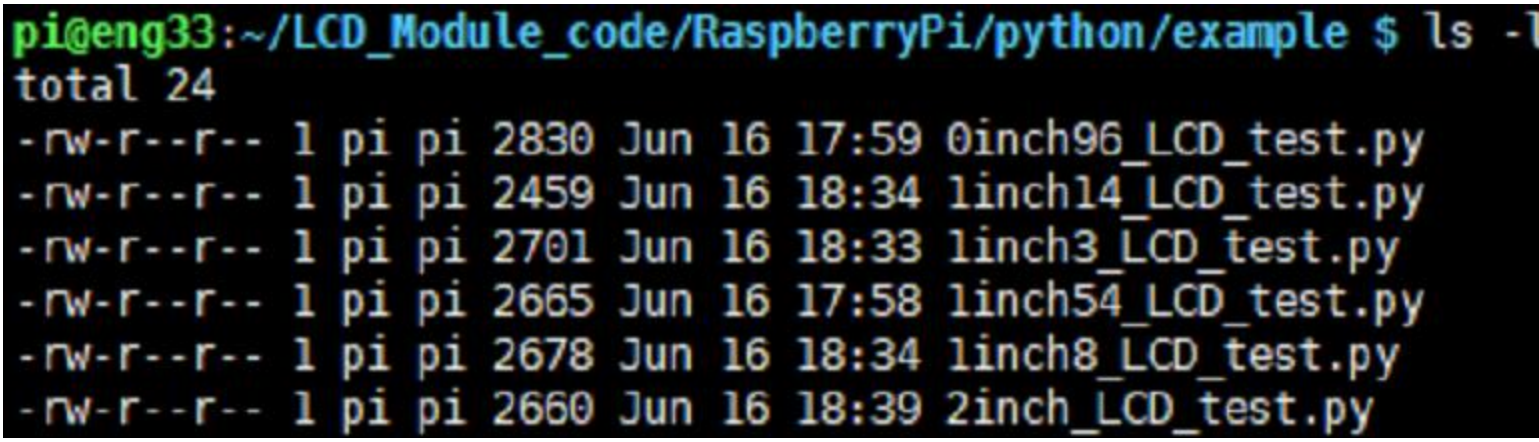
Use the command according to LCD: :

```
sudo ./main 0.96
sudo ./main 1.14
sudo ./main 1.3
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
```



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
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 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 1inch3_LCD_test.py
sudo python3 1inch54_LCD_test.py
sudo python3 1inch8_LCD_test.py
sudo python3 2inch_LCD_test.py
```

FBCP Transplant

The Framebuffer uses a memory area to store the display content, and changes the data in the memory to change the display content.

There is an open-source project on github: fbcp-ili9341. Compared with other fbcp projects, this project uses partial refresh and DMA to achieve a refresh rate of up to 60fps.

Compile and Run

```
sudo apt-get install cmake -y
```

```
cd ~
wget https://www.waveshare.com/w/upload/f/f9/Waveshare_fbcp.7z
sudo apt-get install p7zip-full
7z x Waveshare_fbcp.7z
cd waveshare_fbcp
mkdir build
cd build
cmake [options] ..
make -j
sudo ./fbcp
```

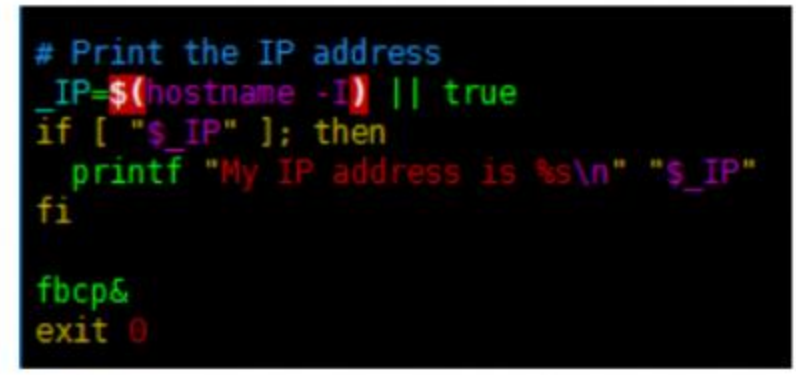
Replace the above cmake [options] .. according to the LCD Module you are using.

```
#0.96inch LCD Module
cmake -DSPI_BUS_CLOCK_DIVISOR=20 -DWAVESHARE_0INCH96_LCD=ON -DBACKLIGHT_CONTROL=ON -
DSTATISTICS=0 ..

#1.14inch LCD Module
cmake -DSPI_BUS_CLOCK_DIVISOR=20 -DWAVESHARE_1INCH14_LCD=ON -DBACKLIGHT_CONTROL=ON -
DSTATISTICS=0 ..

#1.3inch LCD Module
cmake -DSPI_BUS_CLOCK_DIVISOR=20 -DWAVESHARE_1INCH3_LCD=ON -DBACKLIGHT_CONTROL=ON -
DSTATISTICS=0 ..

#1.54inch LCD Module
cmake -DSPI_BUS_CLOCK_DIVISOR=20 -DWAVESHARE_1INCH54_LCD=ON -DBACKLIGHT_CONTROL=ON -
DSTATISTICS=0 ..

#1.8inch LCD Module
cmake -DSPI_BUS_CLOCK_DIVISOR=20 -DWAVESHARE_1INCH8_LCD=ON -DBACKLIGHT_CONTROL=ON -
DSTATISTICS=0 ..

#2inch LCD Module
cmake -DSPI_BUS_CLOCK_DIVISOR=20 -DWAVESHARE_2INCH_LCD=ON -DBACKLIGHT_CONTROL=ON -
DSTATISTICS=0 ..

#2.4inch LCD Module
cmake -DSPI_BUS_CLOCK_DIVISOR=20 -DWAVESHARE_2INCH4_LCD=ON -DBACKLIGHT_CONTROL=ON -
DSTATISTICS=0 ..
```

Auto-start when Power on

```
sudo cp ~/Fbcp-ili9341/build/fbcp-ili9341 /usr/local/bin/fbcp
sudo nano /etc/rc.local
```

And then add fbcp& before exit 0, as the picture below.



Set the display resolution

Set the user interface display size in the /boot/config.txt file.

```
sudo nano /boot/config.txt
```

Then add the following lines at the end of the config.txt.

```
hdmi_force_hotplug=1
hdmi_cvt=[options]
hdmi_group=2
hdmi_mode=1
hdmi_mode=87
display_rotate=0
```

Replace the above hdmi_cvt=[options] according to the LCD Module you are using.

```
#2.4inchinch LCD Module & 2inchinch LCD Module
hdmi_cvt=640 480 60 1 0 0 0

#1.8inch LCD Module
hdmi_cvt=400 300 60 1 0 0 0

#1.3inch LCD Module & 1.54inch LCD Module
hdmi_cvt=300 300 60 1 0 0 0

#1.14inch LCD Module
hdmi_cvt=300 170 60 1 0 0 0

#0.96inch LCD Module
hdmi_cvt=300 150 60 1 0 0 0
```

【Note】If you are using Raspberry Pi 4B, you need to comment out the following lines on the [pi4] part. The modification is as below:

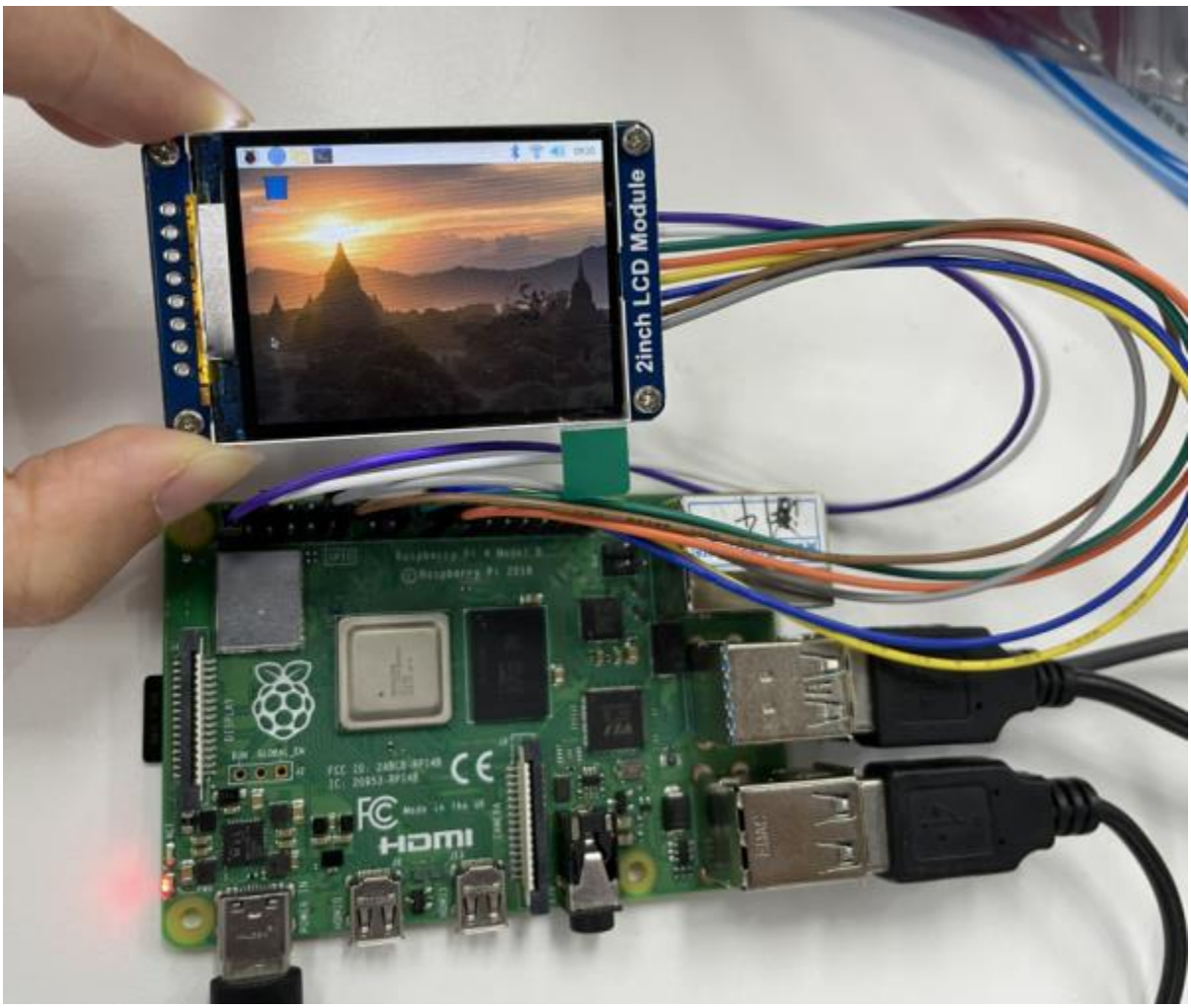
```
[pi4]
# Enable DRM VC4 V3D driver on top of the dispmanx display stack
#dtoverlay=vc4-fkms-v3d
```

```
#max_framebuffers=2
```

And then reboot the system

```
sudo reboot
```

After rebooting the system, the Raspberry Pi OS user interface will be displayed.

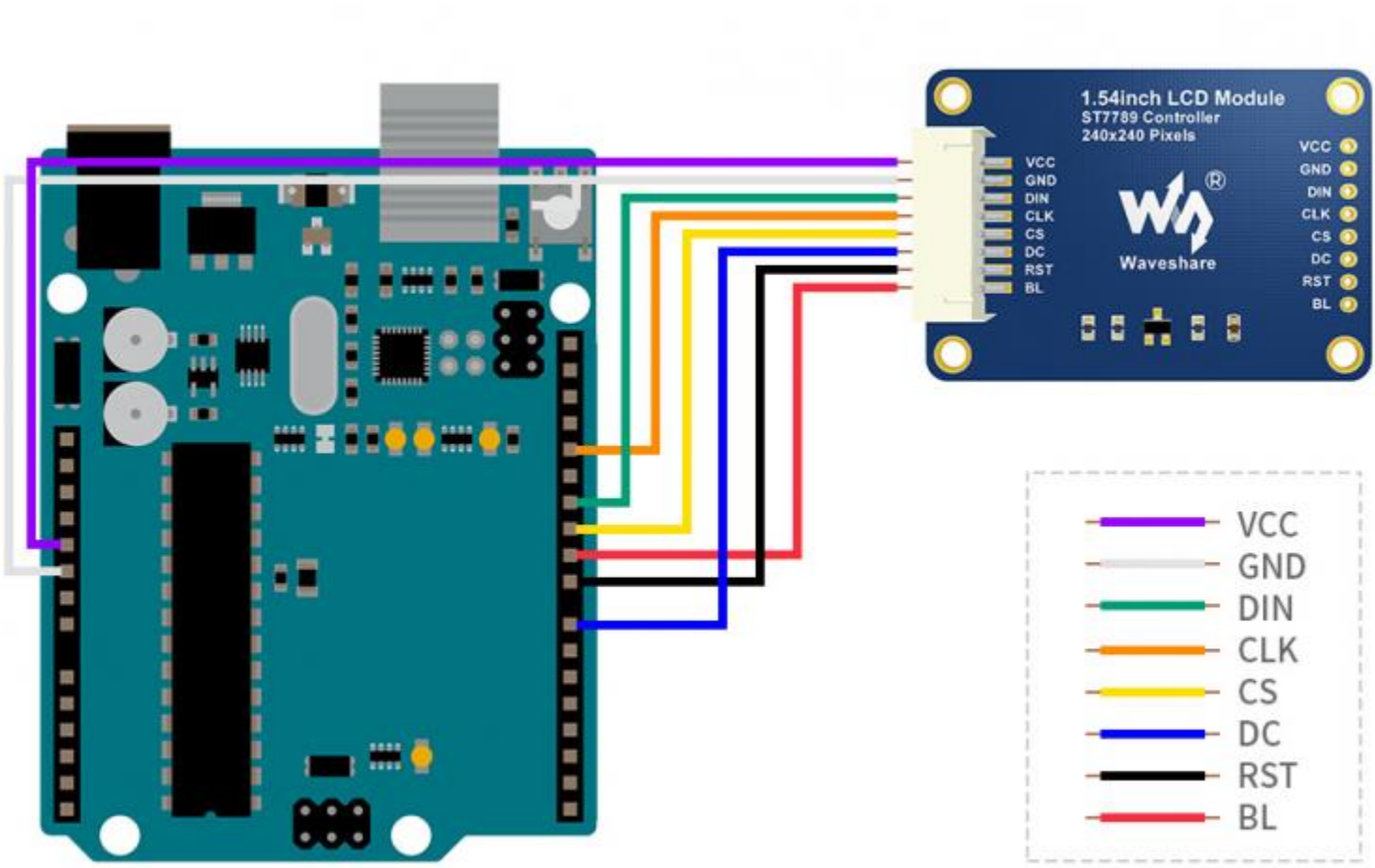


The examples are tested in Arduino UNO, if you want to use other Arduino UNO, you need to change the connection according to the actual boards.

Hardware Connection

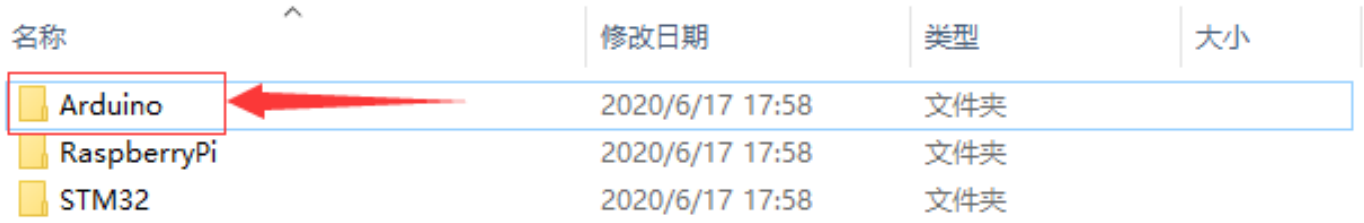
Arduino UNO连接引脚对应关系	
LCD	UNO
VCC	5V/3.3V
GND	GND
DIN	D11
CLK	D13
CS	D10
DC	D7

RST	D8
BL	D9



Run the example

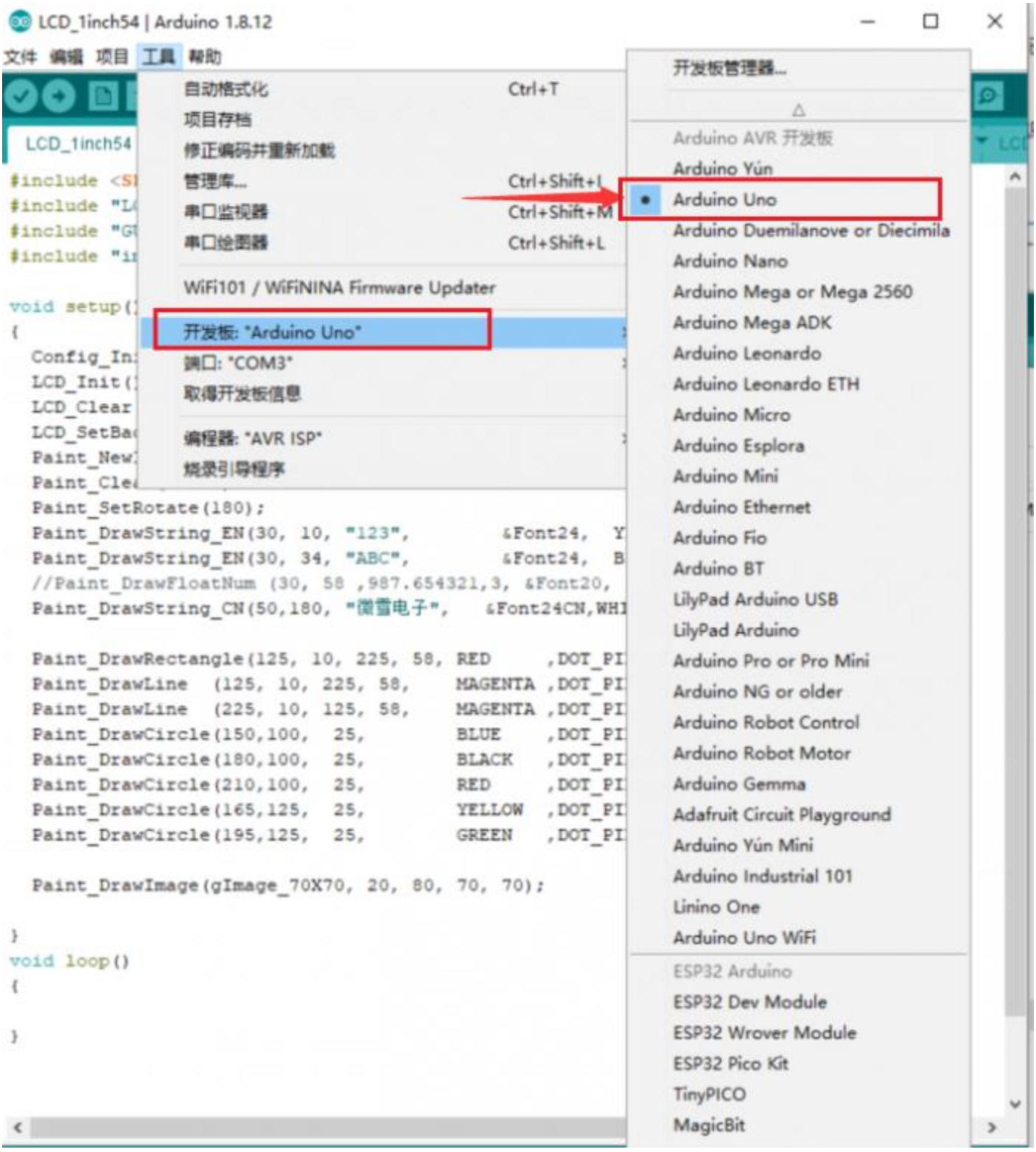
Download the demo codes and unzip it. The Arduino project is located in ~/Arduino/...



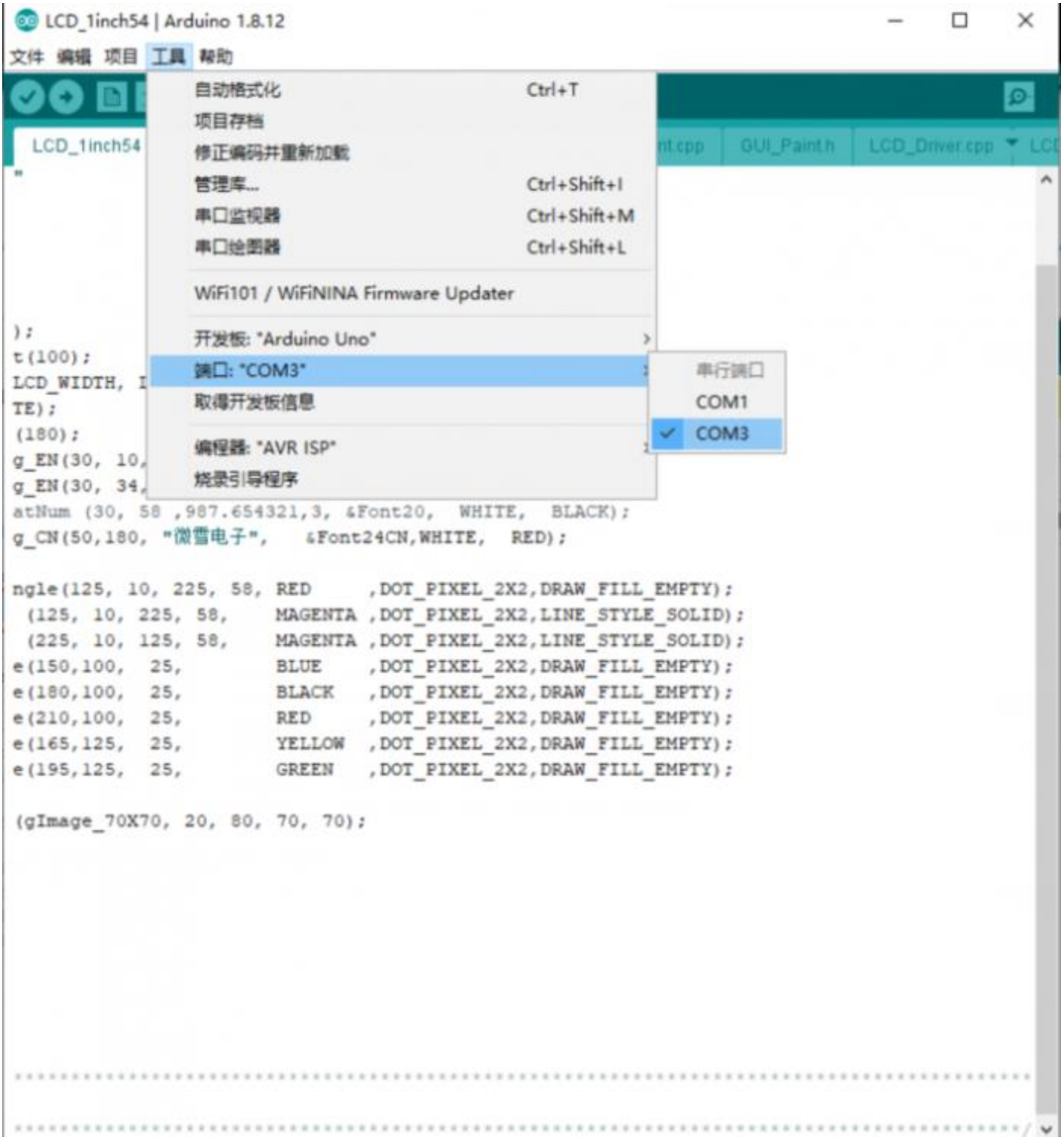
Run the project according to the actual display type

名称	修改日期	类型	大小
LCD_0inch96	2020/6/17 17:58	文件夹	
LCD_1inch3	2020/6/17 17:58	文件夹	
LCD_1inch8	2020/6/17 17:58	文件夹	
LCD_1inch14	2020/6/17 17:58	文件夹	
LCD_1inch54	2020/6/17 17:58	文件夹	
LCD_2inch	2020/6/17 17:58	文件夹	

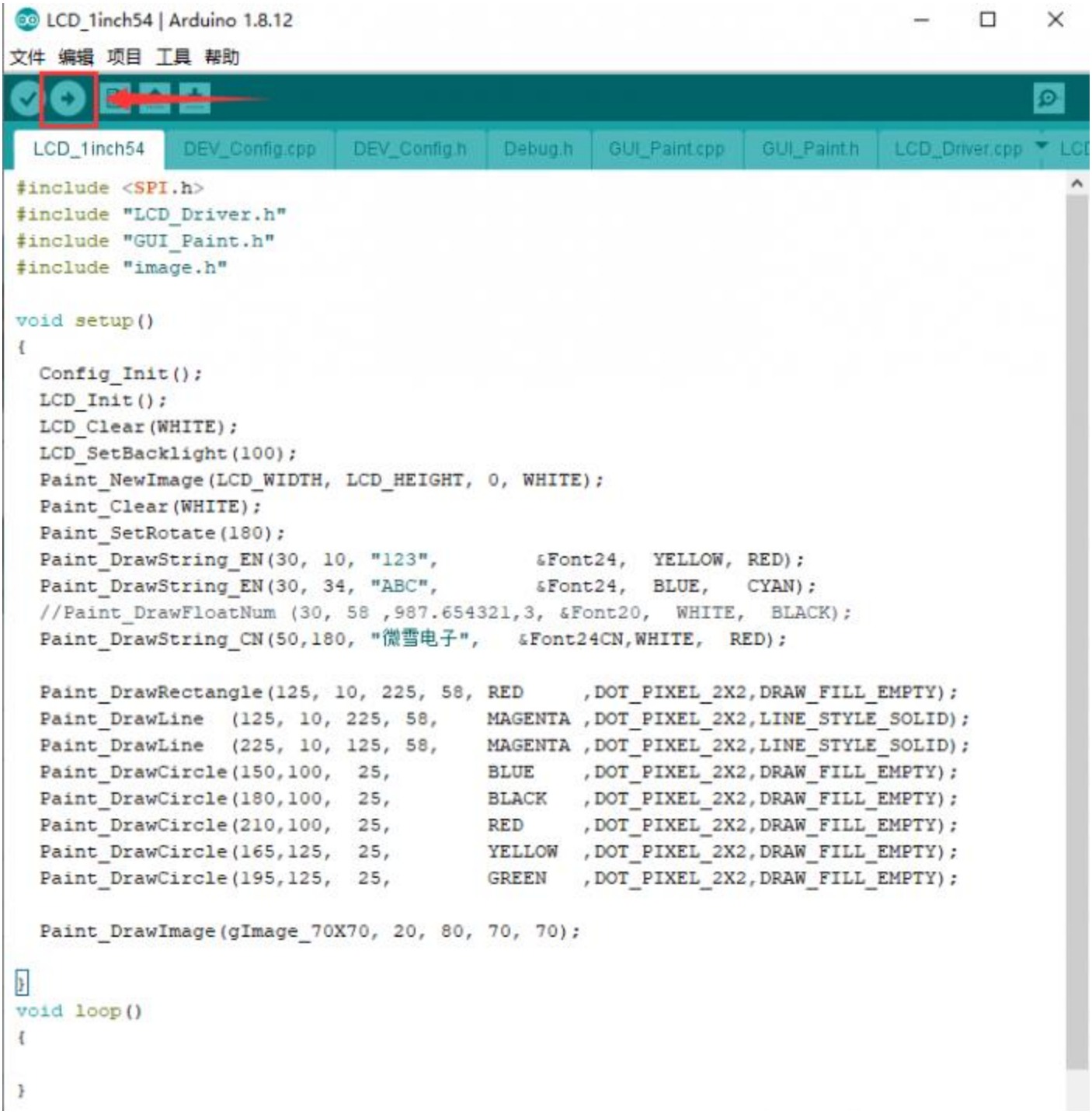
For examples: 1.54inch LCD Module. Enter the LCD_1inch54 directory and run the LCD_1inch54.ino file
Run the project and choose Arduino UNO as Board



Select the COM Port according to your Device Manager



Compile and download it to your board



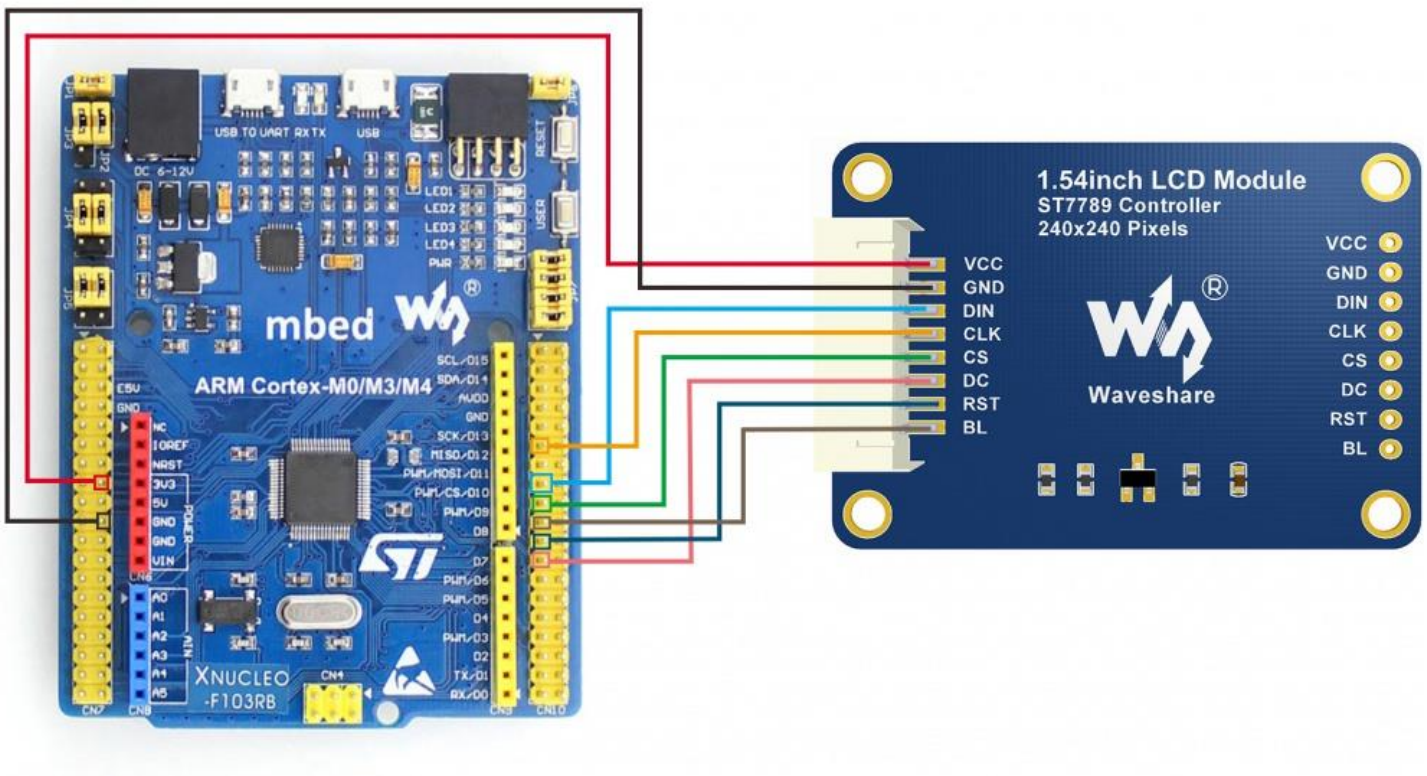
Hardware Coonnection

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

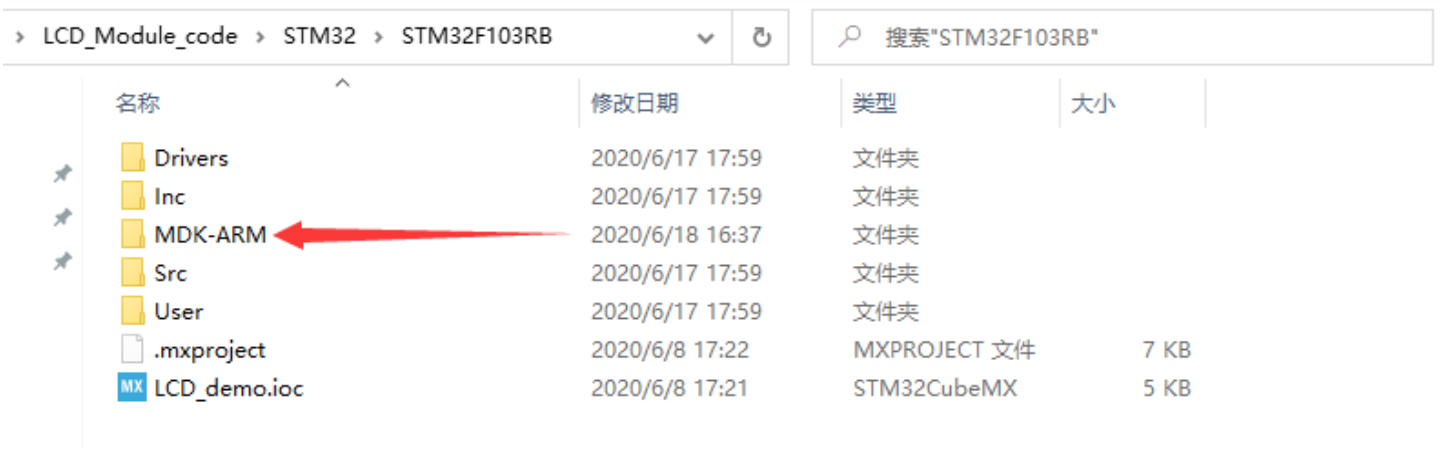
DIN	PA7
CLK	PA5
CS	PB6
DC	PA8
RST	PA9
BL	PC7

Use Waveshare XNUCLEO-F103RB as examples

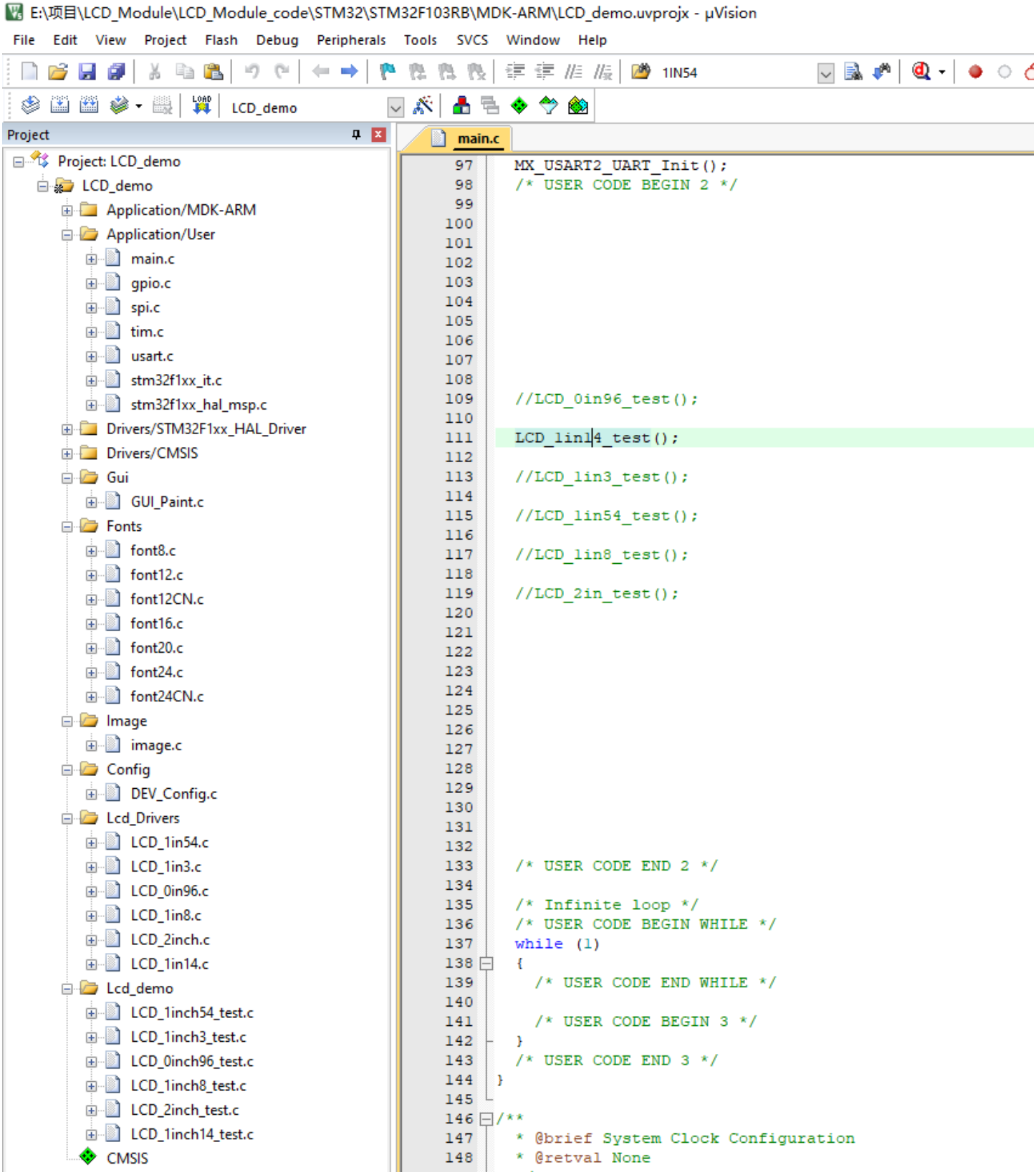


About the examples

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



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



LCD_0in96_test() 0.96inch LCD example

LCD_1in14_test() 1.14inch 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
- ST7789VW Datasheet

Demo codes

- Demo codes