

1. Introduction to Testing Platform

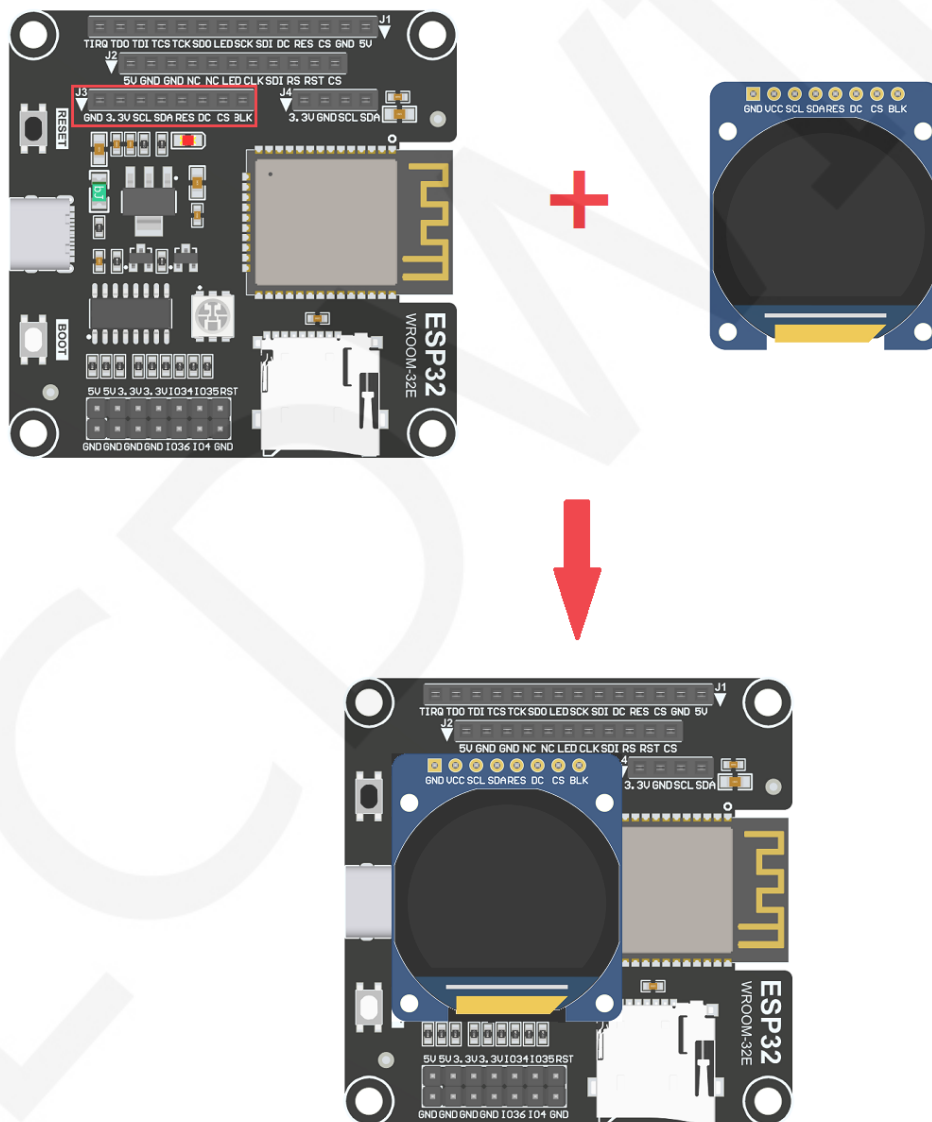
Development Board : ESP32-WROOM-32E devKit

MCU : ESP32-32E module

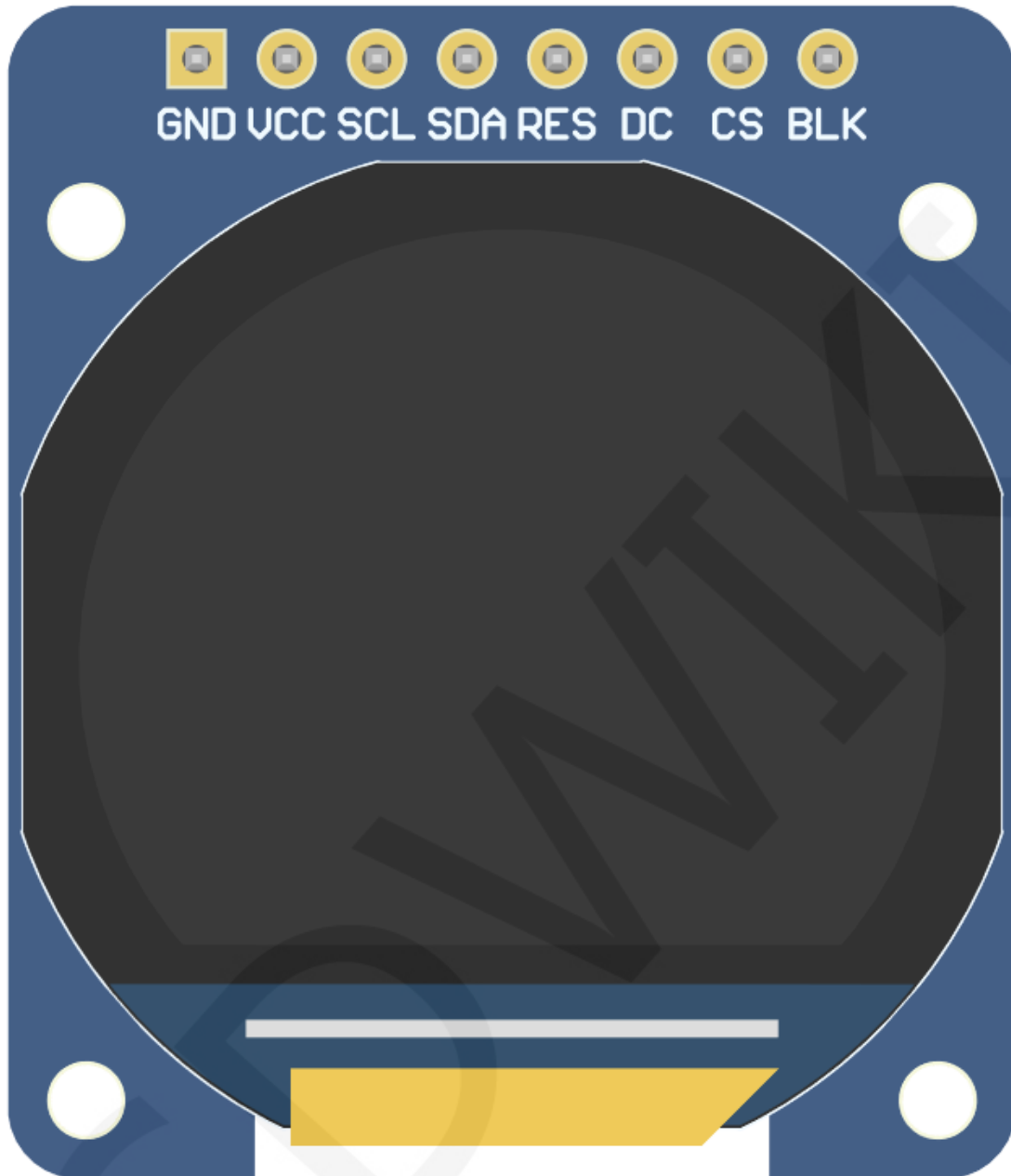
Frequency : 240MHz

2. Pin connection instructions

The display module can be directly plugged into the ESP32-32E development board, as shown in the following figure:



Picture1. Module inline ESP32-32E development board



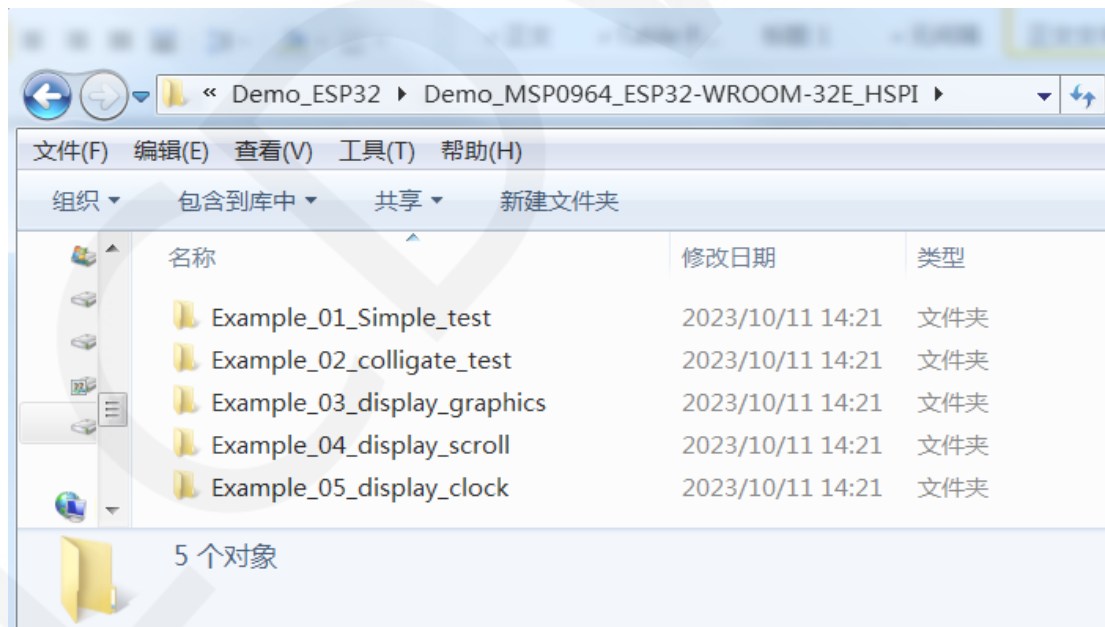
Picture 2. Module front pin diagram

ESP32-32E Test Program Pin Direct Insertion Instructions			
Number	Module pins	Corresponding ESP32-32E development board wiring pins	Remarks
1	GND	GND	LCD Power ground
2	VCC	5V/3.3V	LCD power positive(It is recommended to connect to 5V. When connected to 3.3V, the backlight brightness will be slightly dim)

3	SCL	IO14	LCD SPI bus clock signal
4	SDA	IO13	LCD SPI bus write data signal
5	RES	IO27	LCD reset control signal, Low level reset
6	DC	IO2	LCD command / data selection control signal High level: data, low level: command
7	CS	IO15	LCD selection control signal, Low level active
8	BLK	IO21	LCD backlight control signal (If you need control, please connect the pins. If you don't need control, you can skip it)

3. Demo Function Description

This sample program uses the ESP32 hardware HSPI bus, which is located in **Demo_MSP0964_ESP32-WROOM-32E_HSPI** directory, as shown in the following figure:



✧ Description of sample program content

A. Example_01_Simple_Test is a screen brushing test program, which does not

- rely on any software library;
- B. Example_02_colligate_Test is a comprehensive testing program that displays graphics, lines, and counts program runtime;
 - C. Example_03_display_Graphics is a graphic display testing program that displays various graphics;
 - D. Example_04_display_Scroll is a scrolling test program that displays text scrolling;
 - E. Example_05_display_Scroll is a scrolling test that displays text scrolling;

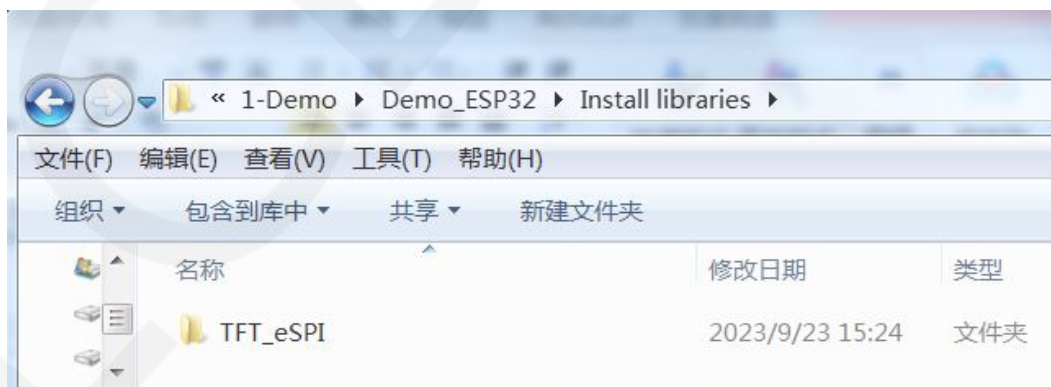
4. Demo Usage Instructions

✧ Building Development Environment

For specific methods of building a development environment, please refer to the "Arduino_development_environment_construction_for-ESP32-EN" document in this directory.

✧ Installing software library

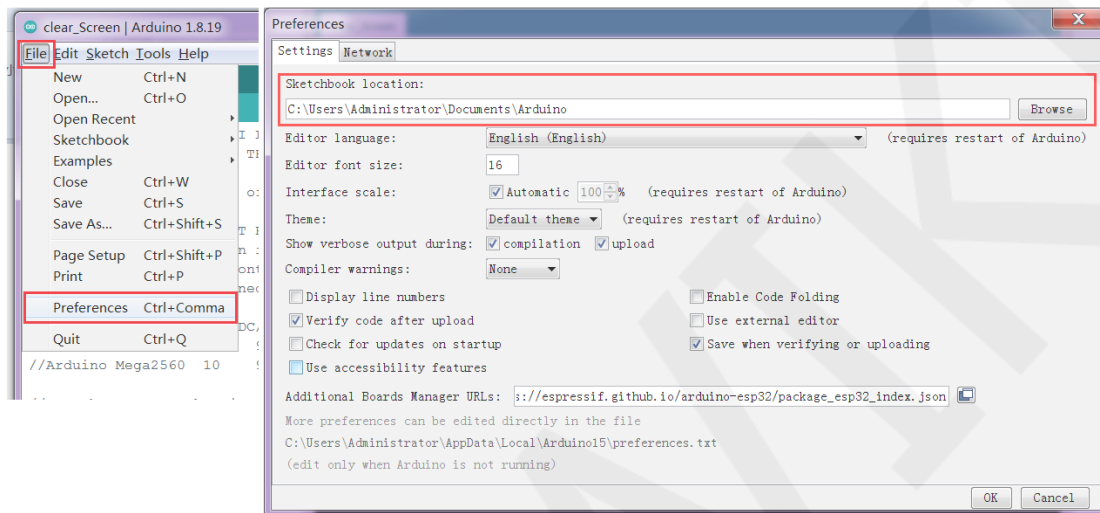
After the development environment is set up, the software library used by the sample program needs to be copied to the project library directory so that the sample program can be called. The software library is located in the **Install libraries** directory, as shown in the following figure:



Among them:

TFT_eSPI is an Arduino graphics library for TFT-LCD LCD screens, supporting multiple platforms and LCD driver ICs

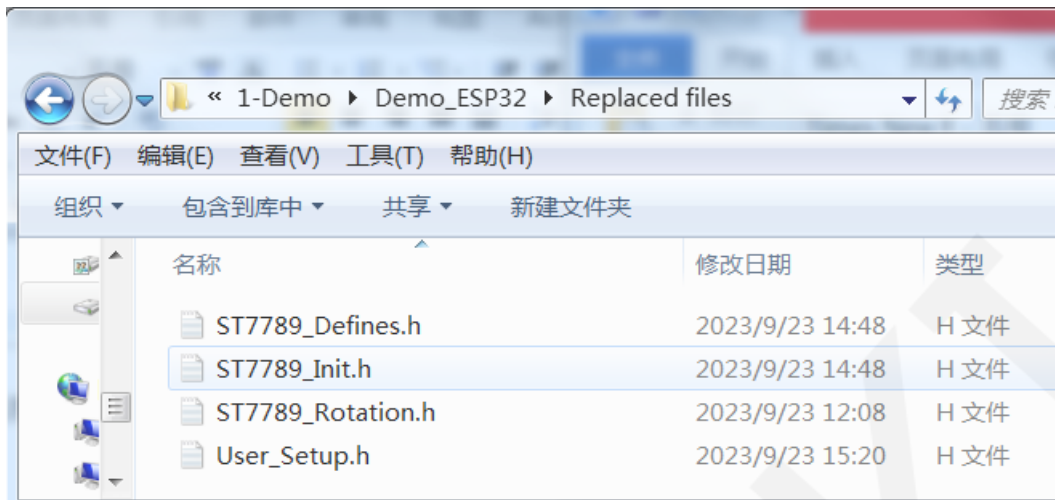
The software library have been configured and can be directly copied to the project library directory for use. The default path for the engineering library directory is **C:\Users\Administrator\Documents\Arduino\libraries**. You can also change the project library directory: open the Arduino IDE software, click **File ->Preferences**, and reset the **Sketchbook location** in the pop-up interface, as shown in the following figure:



If you do not want to use the already configured library, you can download the latest version of the library from Github at the following download address and then configured:

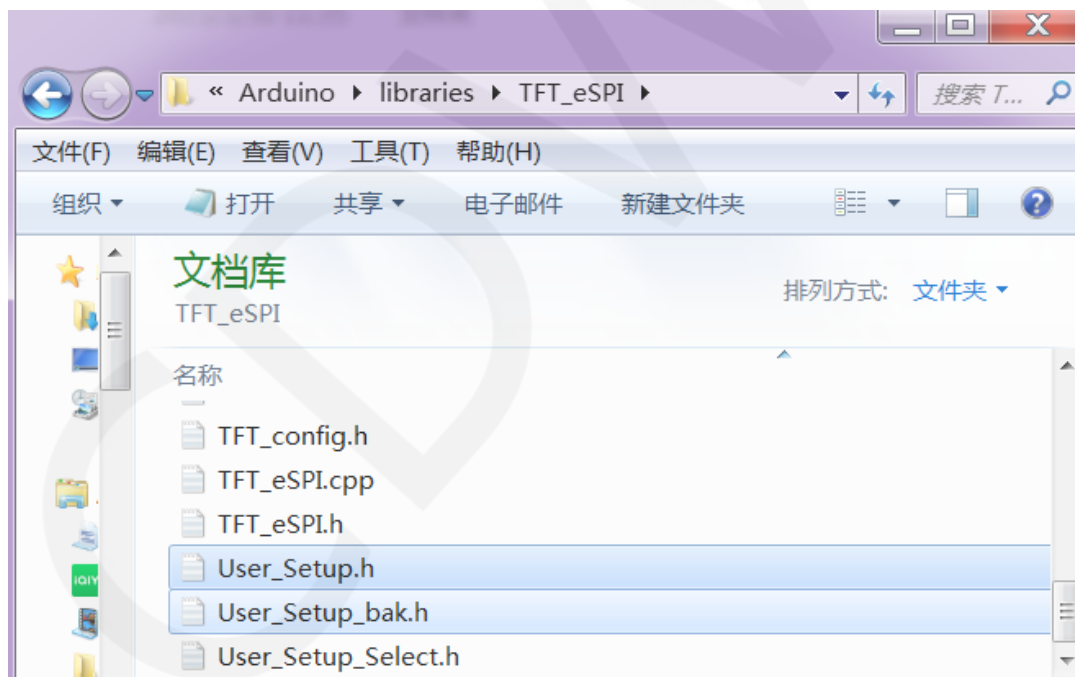
https://github.com/Bodmer/TFT_eSPI

After the library download is completed, unzip it (for easy differentiation, rename the unzipped library folder, as shown in the Install libraries directory), and then copy it to the engineering library directory. Next, proceed with library configuration. The files that need to be replaced are located in the **Replaced files** directory, as shown in the following figure:



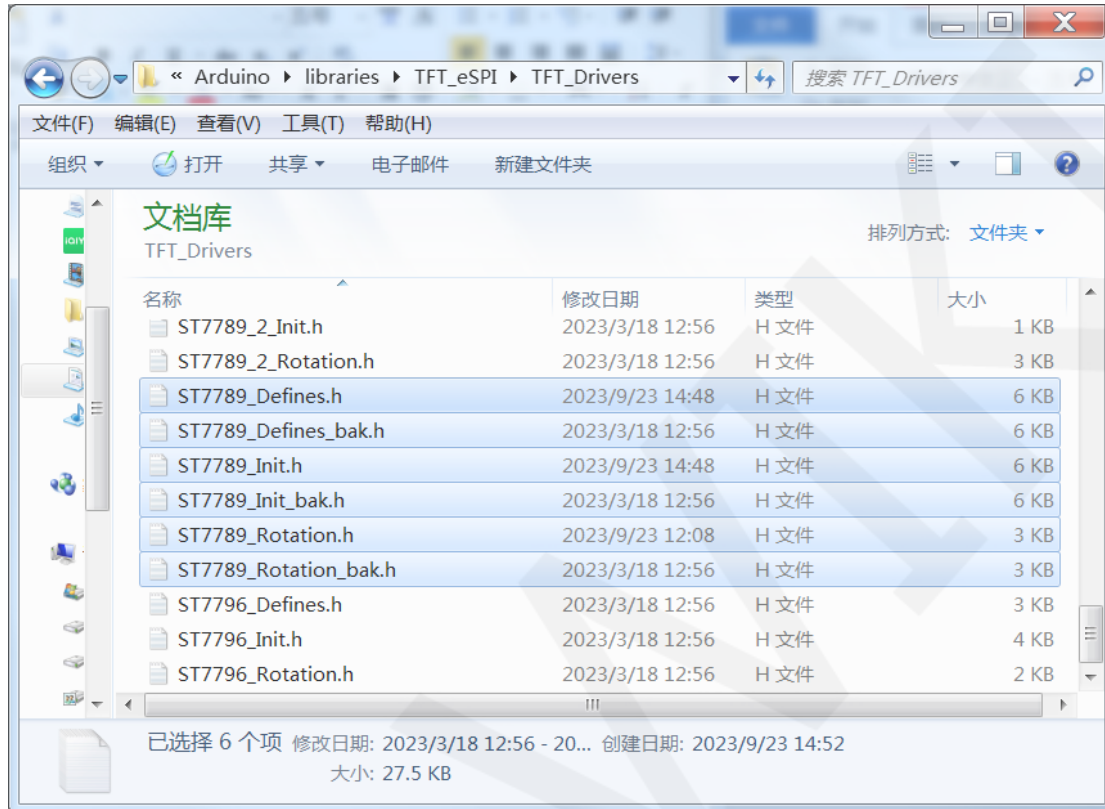
TFT_eSPI library configuration:

First rename the **User_Setup.h** file which is in the top-level directory of the **TFT_eSPI** library of the engineering library directory to **User_Setup_bak.h**, then copy the **User_Setup.h** file which is in the **Replaced files** directory to the top-level directory of the **TFT_eSPI** library, As shown in the following figure:



First, set the TFT in the engineering library directory_ ESPI Library TFT_ ST7789 in the Drivers directory_ Init. h, ST7789_ Rotation. h, ST7789_ Define. h These three files are renamed as ST7789 respectively_ Init. h_ Bak. h, ST7789_ Rotation_ Bak. h, ST7789_ Definitions_ Bak. h, and then replace ST7789 in the Replaced files

directory_ Init. h, ST7789_ Rotation. h, ST7789_ Define. h three copies to TFT in the engineering library directory_ ESPI Library TFT_ Drivers directory, as shown in the following figure:

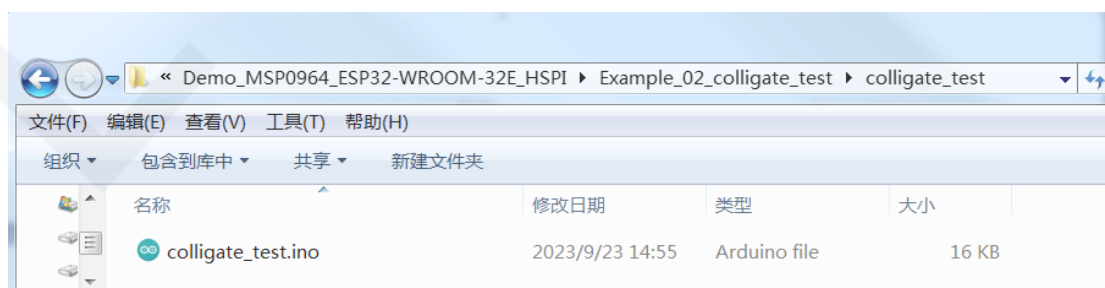


❖ Compile and Run Programs

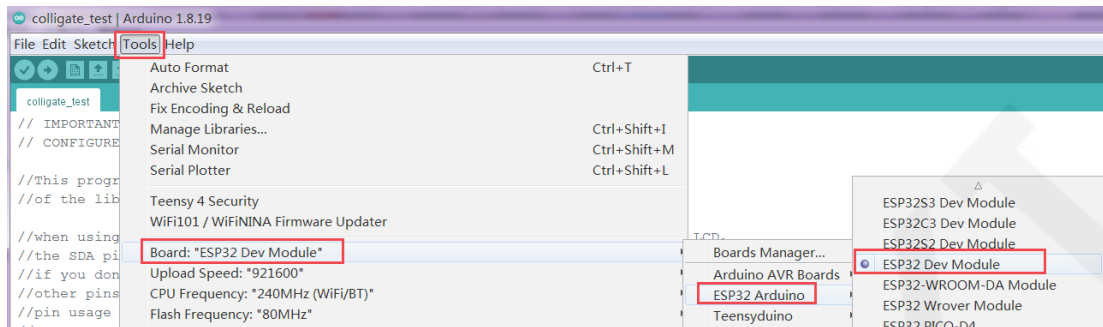
After the library installation is completed, the sample program can be compiled and run as follows:

- A. Plug the display module directly into the ESP32 development board, and connect the development board to a PC to power on;
- B. Open Any sample program in the

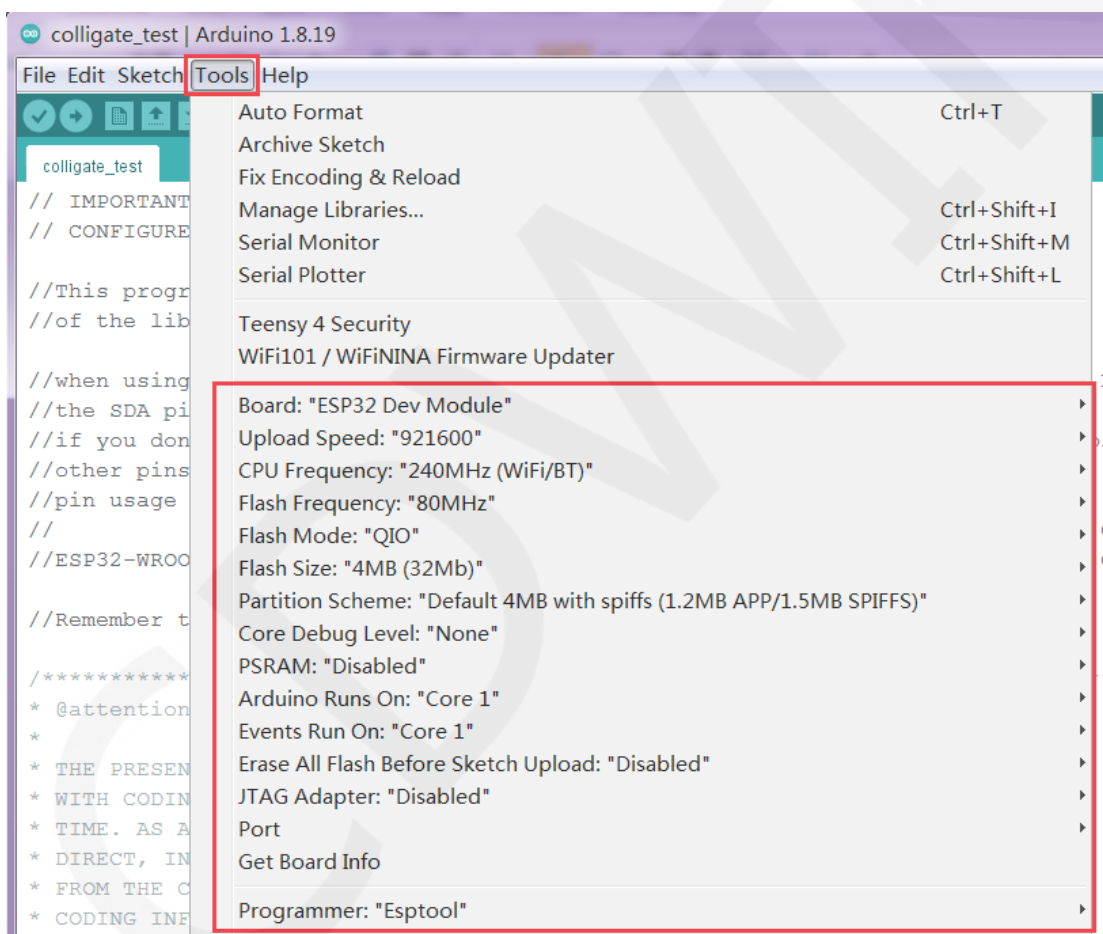
Demo_MSP0962_MSP0963_ESP32-WROOM-32E_HSPI directory, as shown in the following figure (using the colligate test test program as an example):



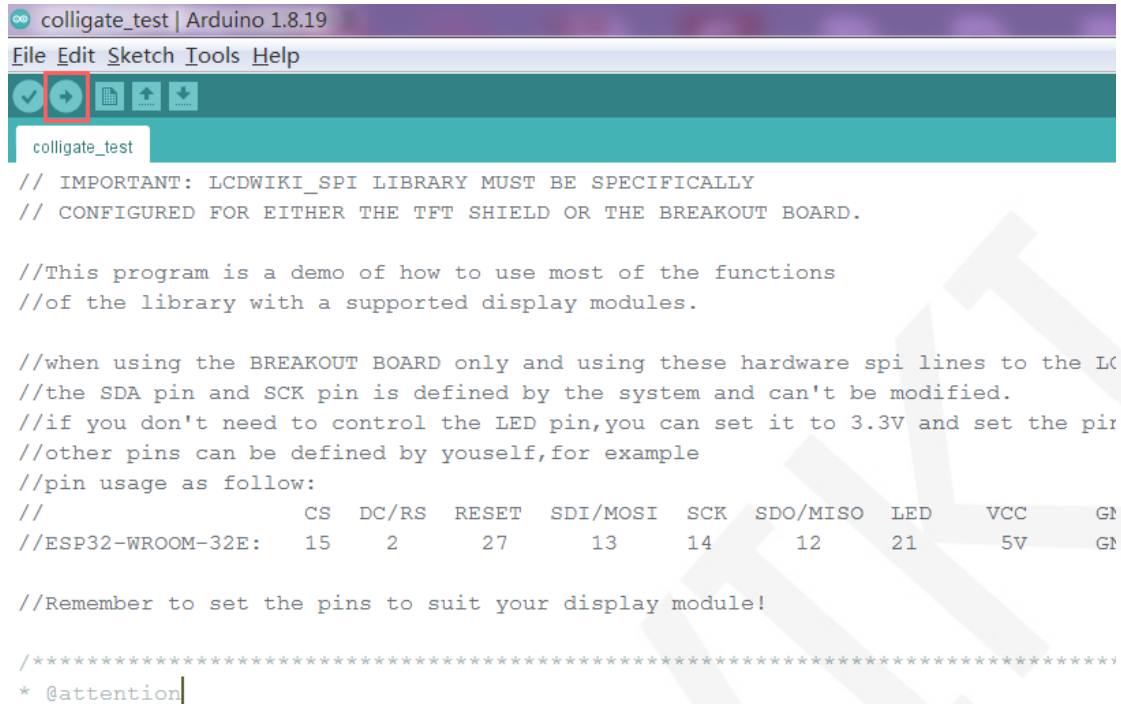
- C. After opening the sample program, select the ESP32 device, as shown in the following figure:



- D. Configure ESP32 Flash, PSRAM, ports, etc. as shown in the following figure:



- E. Click the **upload** button to compile and download the program, as shown in the following figure:



```

colligate_test | Arduino 1.8.19
File Edit Sketch Tools Help
colligate_test
// IMPORTANT: LCDWIKI_SPI LIBRARY MUST BE SPECIFICALLY
// CONFIGURED FOR EITHER THE TFT SHIELD OR THE BREAKOUT BOARD.

//This program is a demo of how to use most of the functions
//of the library with a supported display modules.

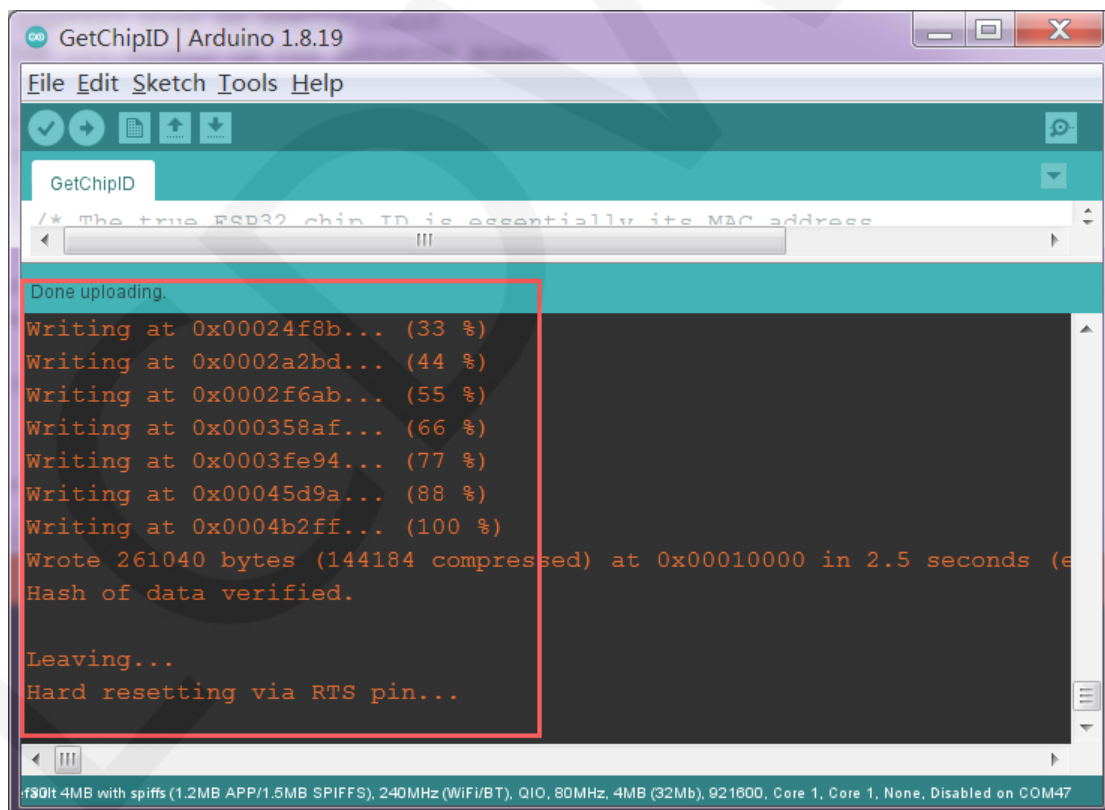
//when using the BREAKOUT BOARD only and using these hardware spi lines to the LCD
//the SDA pin and SCK pin is defined by the system and can't be modified.
//if you don't need to control the LED pin,you can set it to 3.3V and set the pin
//other pins can be defined by yourself,for example
//pin usage as follow:
//
//          CS DC/RS  RESET  SDI/MOSI  SCK  SDO/MISO  LED   VCC   GN
//ESP32-WROOM-32E:  15   2    27    13      14    12    21    5V   GN

//Remember to set the pins to suit your display module!

/*****
 * @attention

```

- F. If the following prompt appears, it indicates that the program has been compiled and downloaded successfully, and has already been run:



```

GetChipID | Arduino 1.8.19
File Edit Sketch Tools Help
GetChipID
/* The true ESP32 chip ID is essentially its MAC address

Done uploading.
Writing at 0x00024f8b... (33 %)
Writing at 0x0002a2bd... (44 %)
Writing at 0x0002f6ab... (55 %)
Writing at 0x000358af... (66 %)
Writing at 0x0003fe94... (77 %)
Writing at 0x00045d9a... (88 %)
Writing at 0x0004b2ff... (100 %)
Wrote 261040 bytes (144184 compressed) at 0x00010000 in 2.5 seconds (effective 102400 bytes/s)
Hash of data verified.

Leaving...
Hard resetting via RTS pin...

ESP32 4MB with spiiffs (1.2MB APP/1.5MB SPIFFS), 240MHz (WiFi/BT), QIO, 80MHz, 4MB (32Mb), 921600, Core 1, Core 1, None, Disabled on COM47

```

- G. If the display module displays content, it indicates that the program has run successfully.