1. Introduction to Testing Platform

Development Board : ESP32-WROOM-32E devKit

MCU : ESP32-32E module

Frequency : 240MHz

2. Pin connection instructions

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

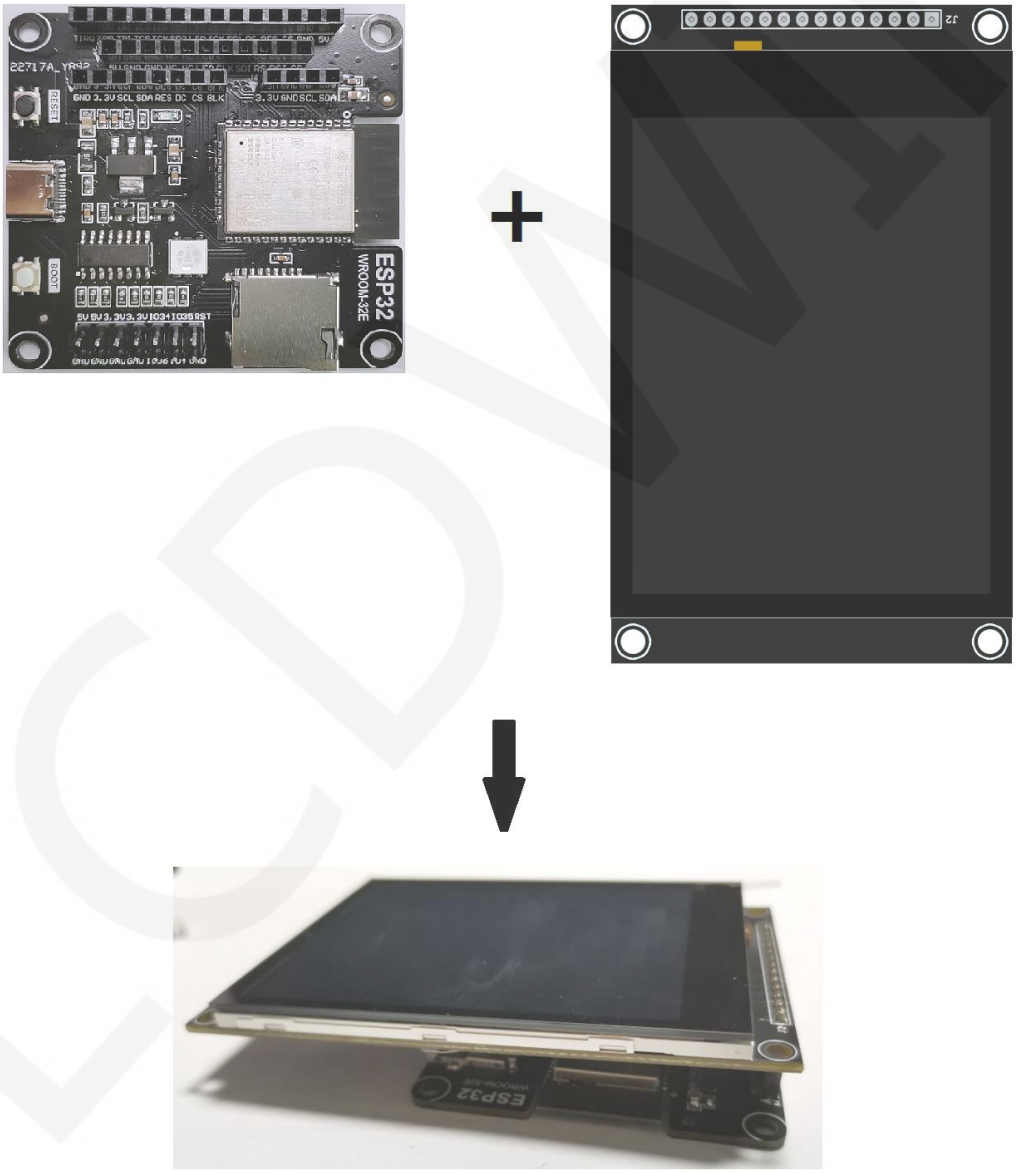


Figure 1: Module Inline ESP 32-32E Development Board

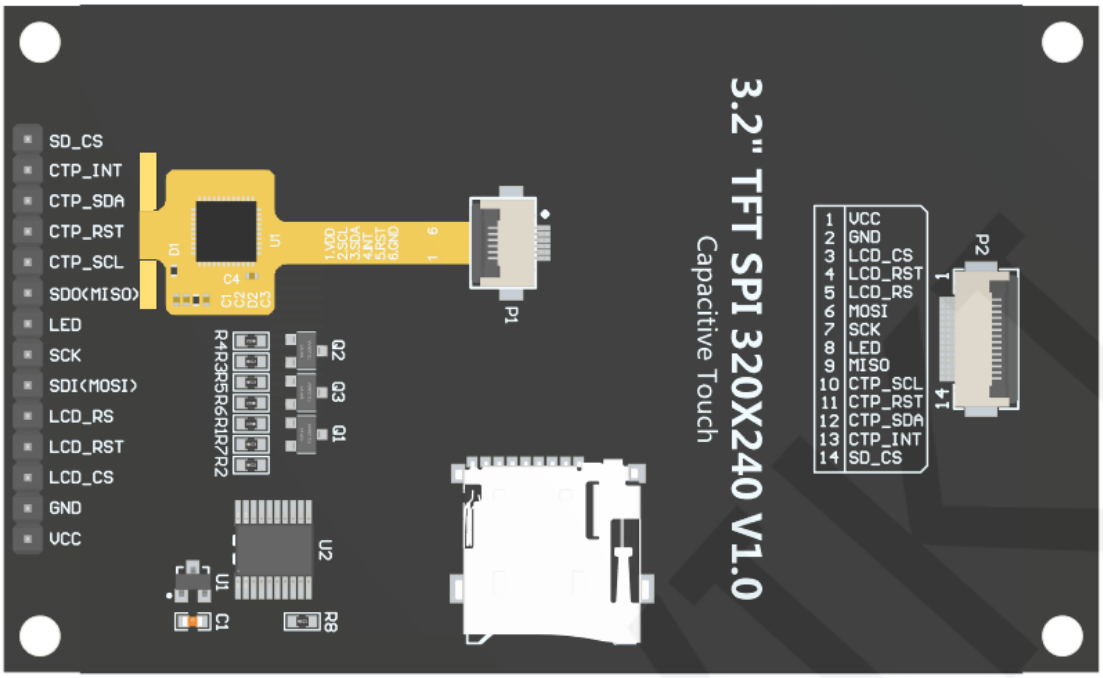


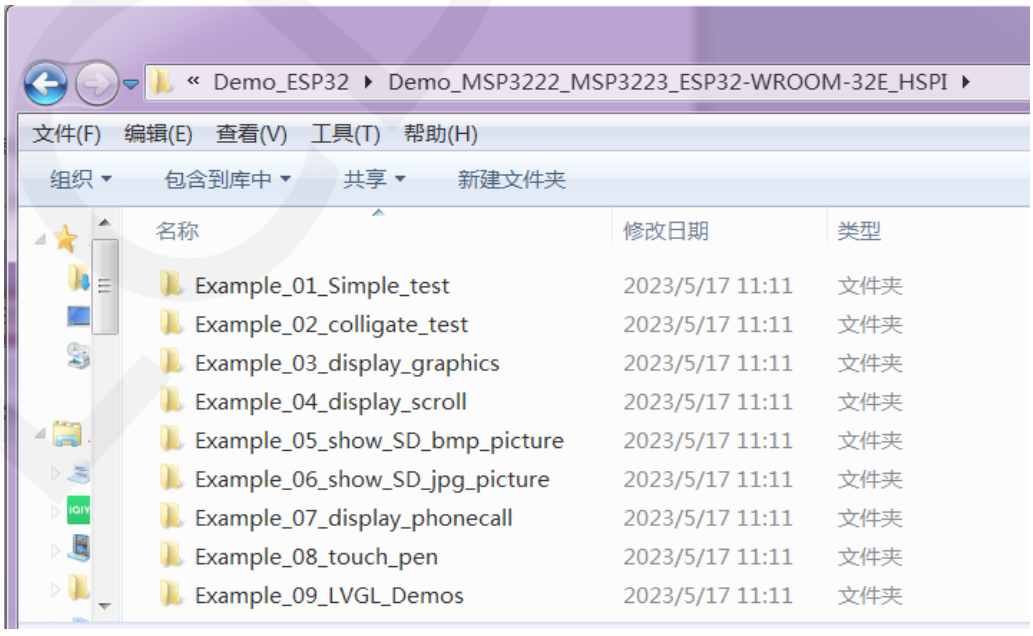
Figure 2 Module Back Pins

ESP32-32E Test Program Pin Direct Insertion Instructions			
Number	**Module pins**	**Corresponding ESP32-32E development board wiring pins**	**Remarks**
1	**VCC**	5V	LCD power positive
2	**GND**	GND	LCD Power ground
3	**LCD_CS**	IO15	LCD selection control signal, Low level active
4	**LCD_RST**	IO27	LCD reset control signal, Low level reset
5	**LCD_RS**	IO2	LCD command / data selection control signal High level: data, low level: command
6	**SDI(MOSI)**	IO13	SPI bus write data signal(SD card and LCD screen used together)
7	**SCK**	IO14	SPI bus clock signal(SD card and LCD screen used together)
8	**LED**	IO21	LCD backlight control signal (If you need control, please connect the pins. If you don't need control, you can skip it)

9	**SDO(MISO)**	IO12	SPI bus read data signal (SD card and LCD screen used together)
10	**CTP_SCL**	IO25	Capacitive touch screen IIC bus clock signal (modules without touch screens do not need to be connected)
11	**CTP_RST**	IO33	Capacitor touch screen reset control signal, low-level reset (modules without touch screens do not need to be connected)
12	**CTP_SDA**	IO32	Capacitive touch screen IIC bus data signal (modules without touch screens do not need to be connected)
13	**CTP_INT**	IO39	Capacitor touch screen IIC bus touch interrupt signal, when generating touch, input low level to the main control (modules without touch screens do not need to be connected)
14	**SD_CS**	IO22	SD card selection control signal, low level active (without SD card function, can be disconnected)

3. Demo Function Description

This sample program uses the ESP32 hardware HSPI bus, which is located in **Demo_MSP3222_MSP3223_ESP32-WROOM-32E_HSPI** directory, as shown in the following figure:



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_ show_ SD_ bmp_ Picture is a BMP image display program that displays BMP format images within SD;

F、Example_ 06_ show_ SD_ jpg_ Picture is a JPG image display program that displays images in jpg format within SD;

G. Example_ 07_ display_ Phonecall is a touch testing program for telephone dialing, which simulates the dialing function through touch;

H. Example_ 08_ touch_ Pen is a touch stroke test program that draws on the LCD screen through touch;

K. Example_ 09_ LVGL_ Demos is an LVGL example display program that allows you to experience the powerful UI design features of LVGL. The bin file for this example has been extracted and can be directly burned using the corresponding tool.

4. Demo Usage Instructions

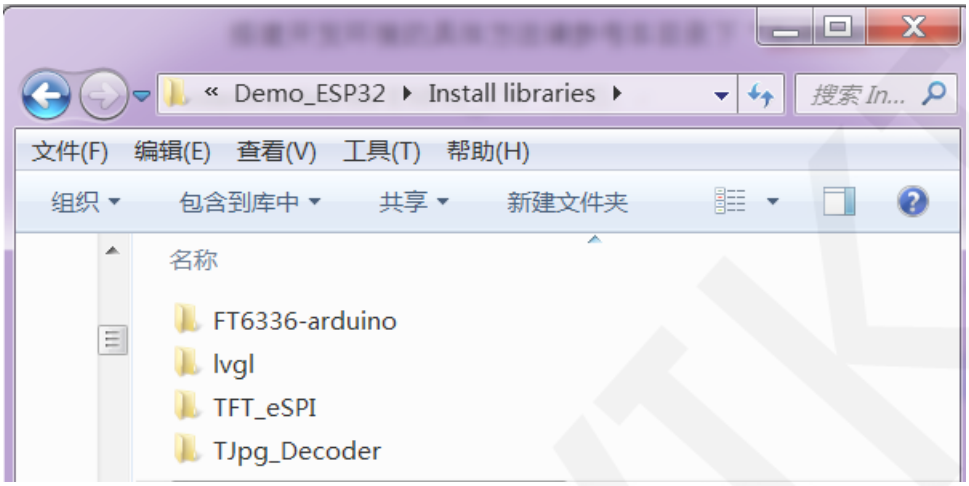
✧ Building Development Environment

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:

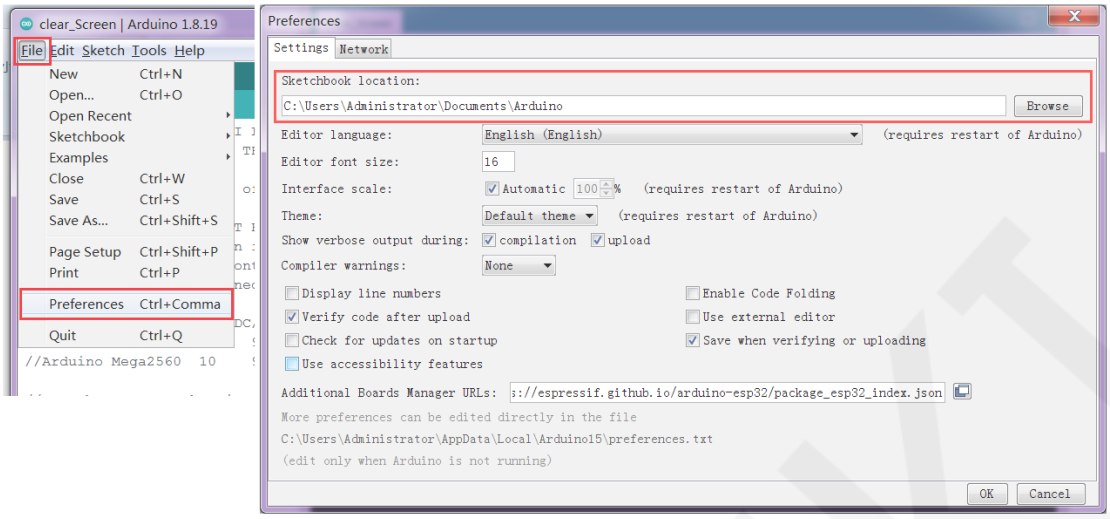
FT6336 arduino is the driver of FT6336 capacitive touch IC

LVgl is LVGL GUI graphics software library

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

TJpg_ Decoder is a JPG format image decoding library for the Arduino platform

These 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 (excluding FT6336 arduino) from Github at the following download address and then configured:

lvgl: https://github.com/lvgl/lvgl/tree/release/v8.3 (Only V8. x version can be used, V9. x version cannot be used)

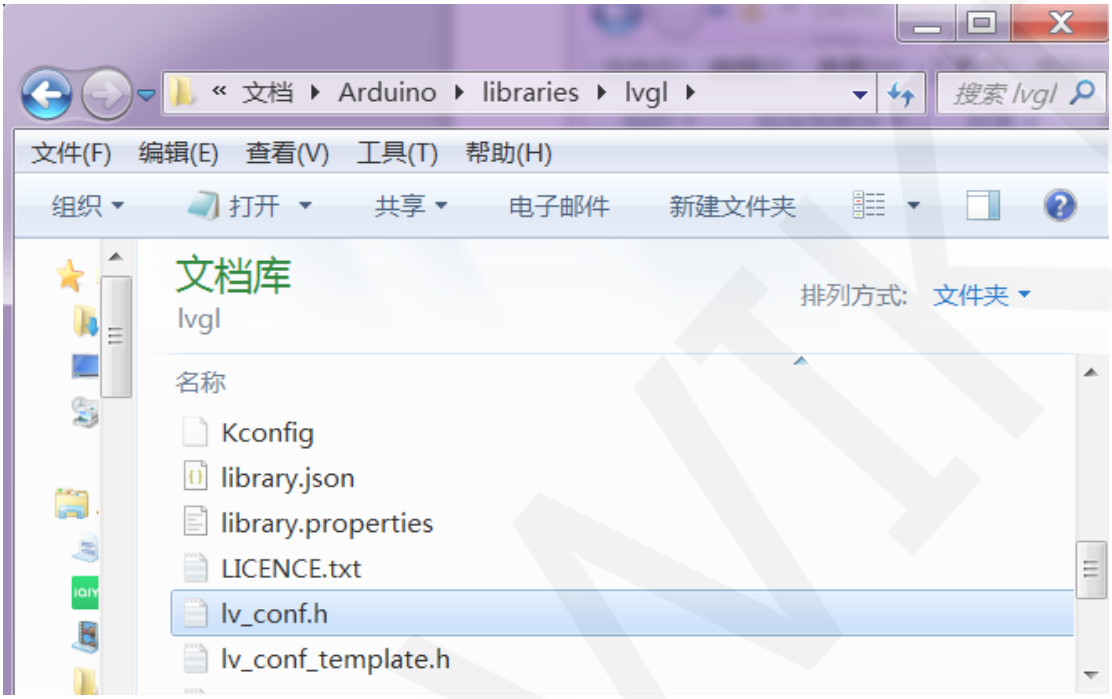
TFT_eSPI: https://github.com/Bodmer/TFT_eSPI

TJpg_Decoder: https://github.com/Bodmer/TJpg_Decoder

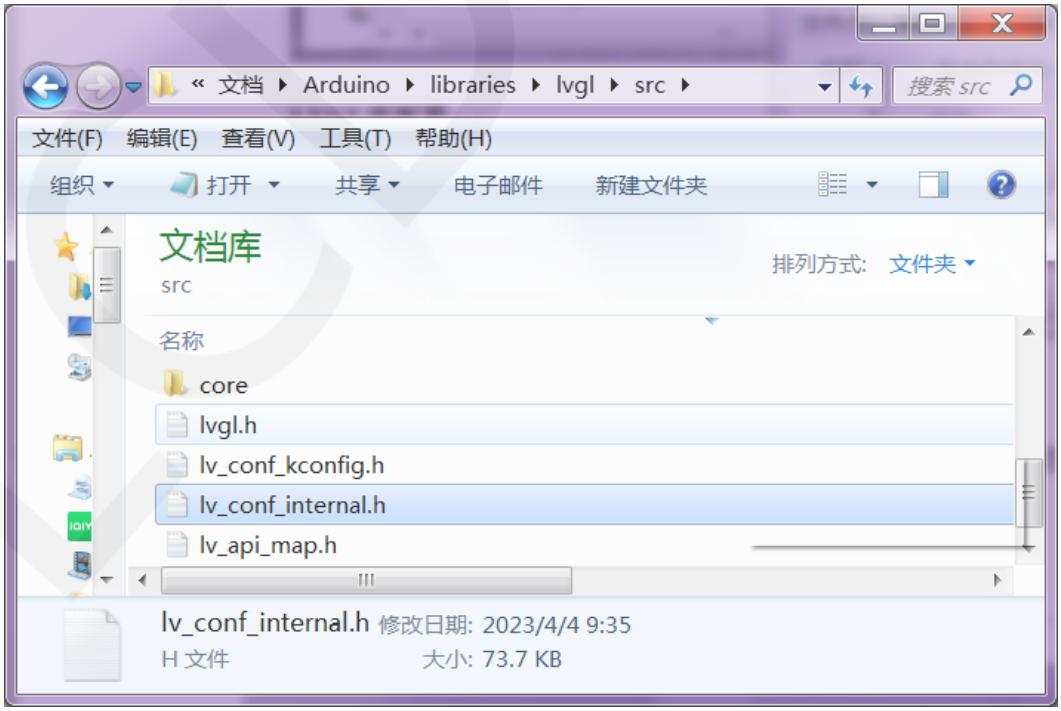
After the library download is completed, unzip it (for easy differentiation, rename the unzipped library folder, as shown in the Install libraries directory), and then copy it to the engineering library directory. Next, proceed with library configuration. The files that need to be replaced are located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

LVGL library configuration:

Copy the **lv_conf.h** file which is in the **Replace files** directory to the top-level directory of the lvgl library in the engineering library directory,As shown in the following figure:



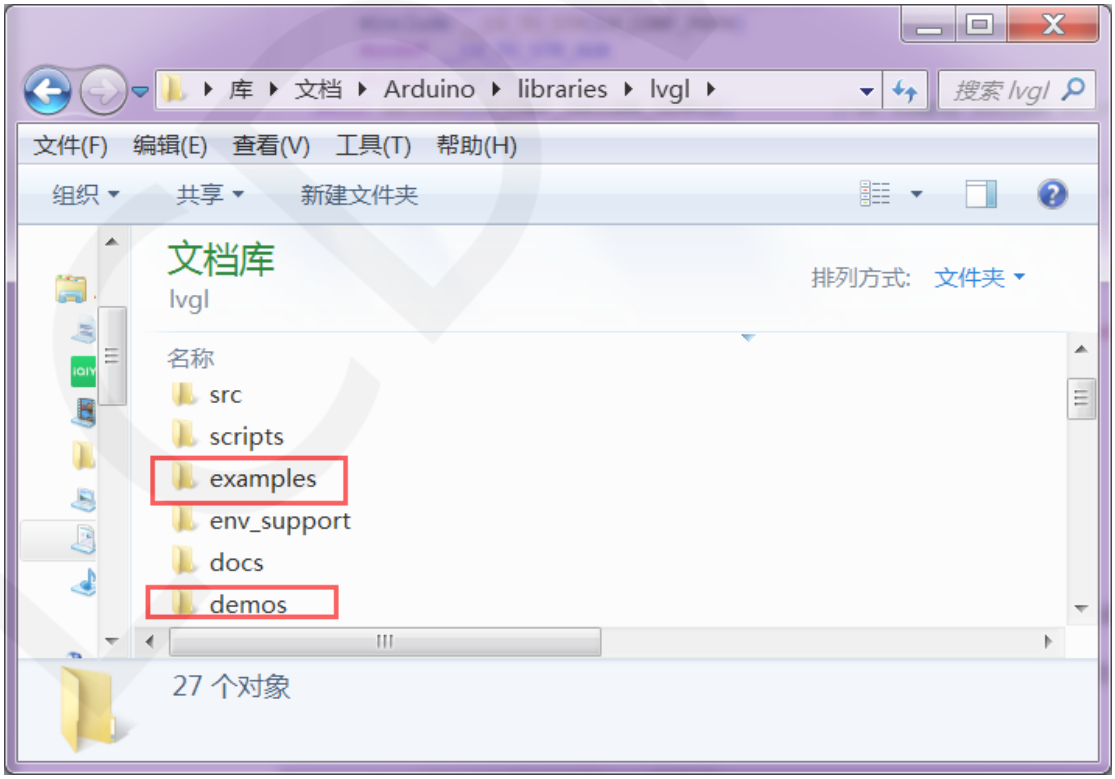
Open the **lv_conf_internal.h** file which is in the Lvgl library **src** directory under the engineering library directory,As shown in the following figure:



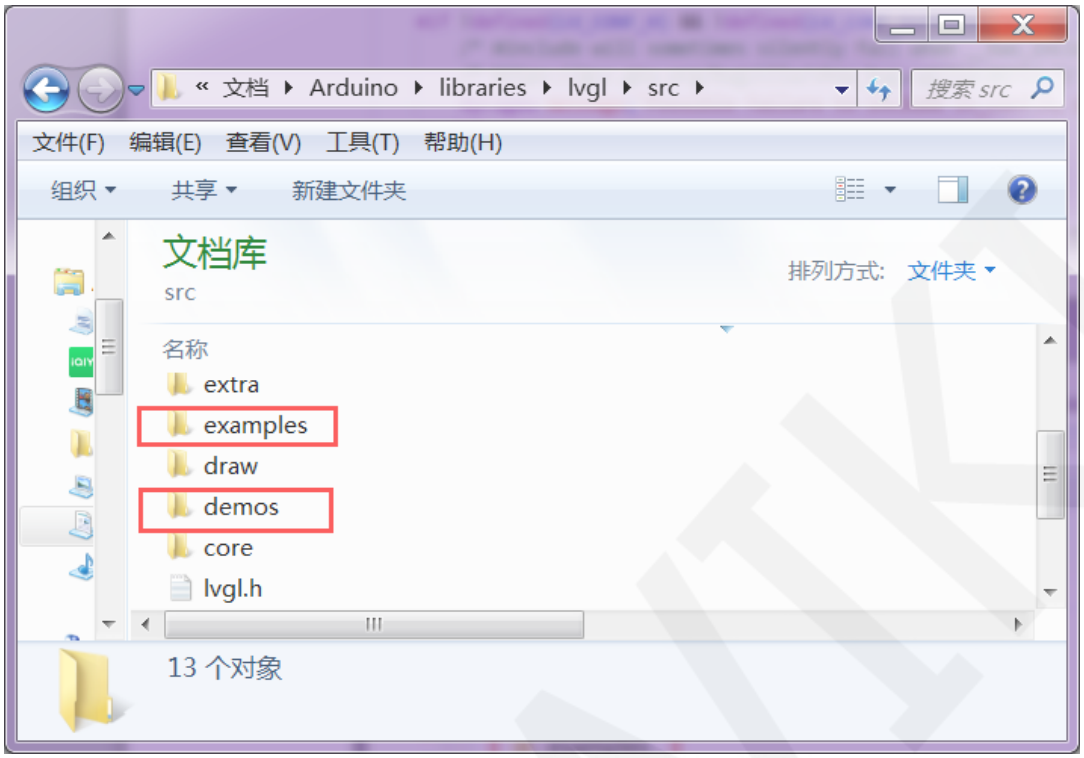
After opening the file, modify the content of line **41** as shown in the following figure (from "**../../lv_conf. h**" to "**../lv_conf. h**"), and save after the modifications are completed.

```
/*If lv_conf.h is not skipped include it*/
#ifndef LV_CONF_SKIP
    #ifdef LV_CONF_PATH                         /*If there is a path defined for lv_conf.h
        #define __LV_TO_STR_AUX(x) #x
        #define __LV_TO_STR(x) __LV_TO_STR_AUX(x)
        #include __LV_TO_STR(LV_CONF_PATH)
        #undef __LV_TO_STR_AUX
        #undef __LV_TO_STR
    #elif defined(LV_CONF_INCLUDE_SIMPLE)       /*Or simply include lv_conf.h is enabled*/
        #include "lv_conf.h"
    #else
        #include "../lv_conf.h"                 /*Else assume lv_conf.h is next to the lvgl fo
    #endif
    #if !defined(LV_CONF_H) && !defined(LV_CONF_SUPPRESS_DEFINE_CHECK)
        /* #include will sometimes silently fail when __has_include is used */
        /* https://gcc.gnu.org/bugzilla/show_bug.cgi?id=80753 */
        #pragma message("Possible failure to include lv_conf.h, please read the comment in th
    #endif
#endif
```

Copy the examples and demos directories under the engineering library directory to the src directory under the lvgl library. These two directories are shown in the following figure in the lvgl library:

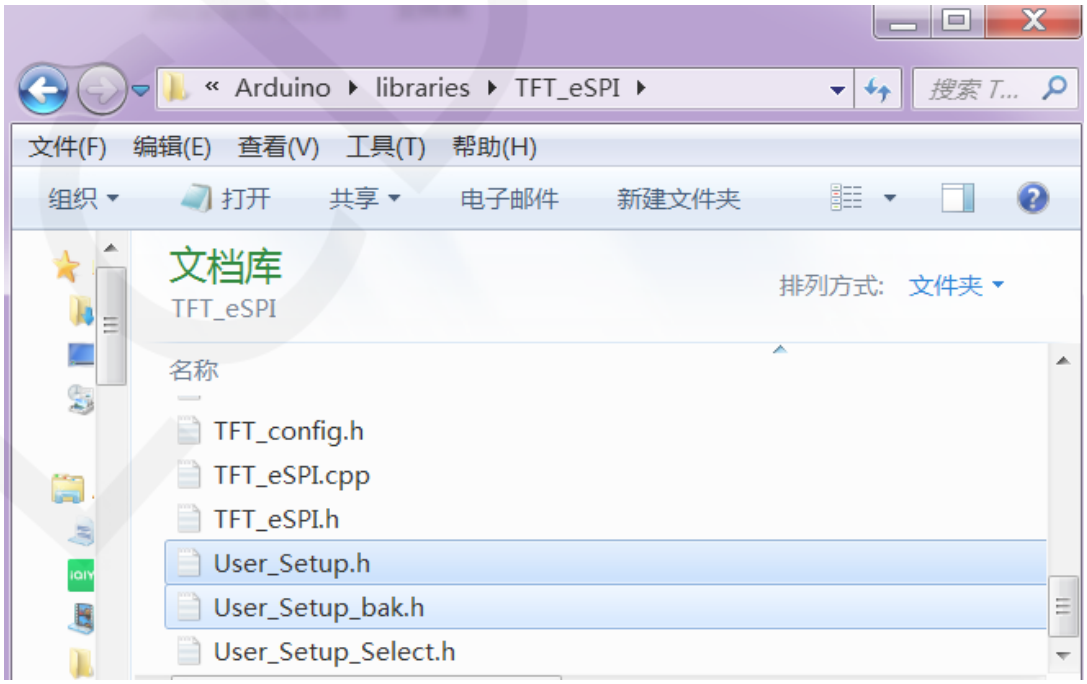


The directory status after copying:



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:



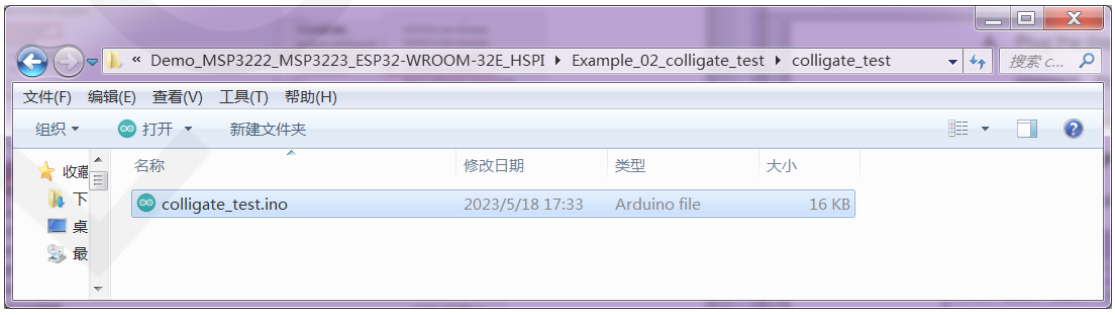
First rename the **ILI9341_Init.h** file which is in the **TFT_Drivers** directory of the the **TFT_eSPI** engineering library directory,then copy the **ILI9341_Init.h** file to the **TFT_Drivers** directory of the the **TFT_eSPI** engineering library directory, as shown in the following figure:

✧ Compile and Run Programs

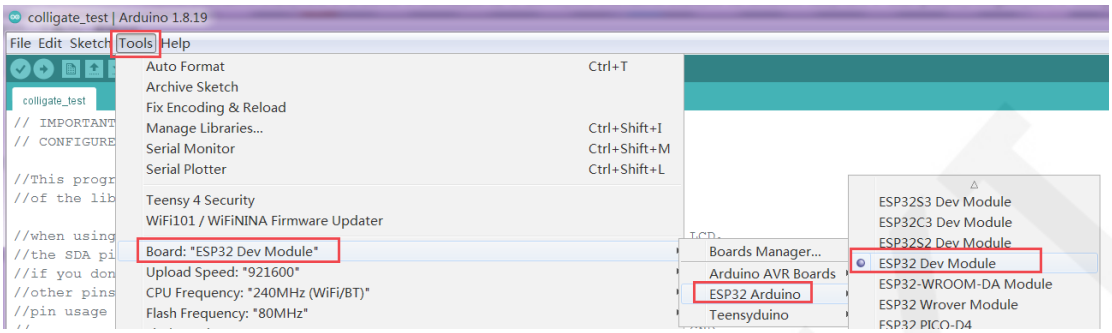
After the library installation is completed, the sample program can be compiled and run as follows:

A. Plug the display module directly into the ESP32 development board, and connect the development board to a PC to power on;

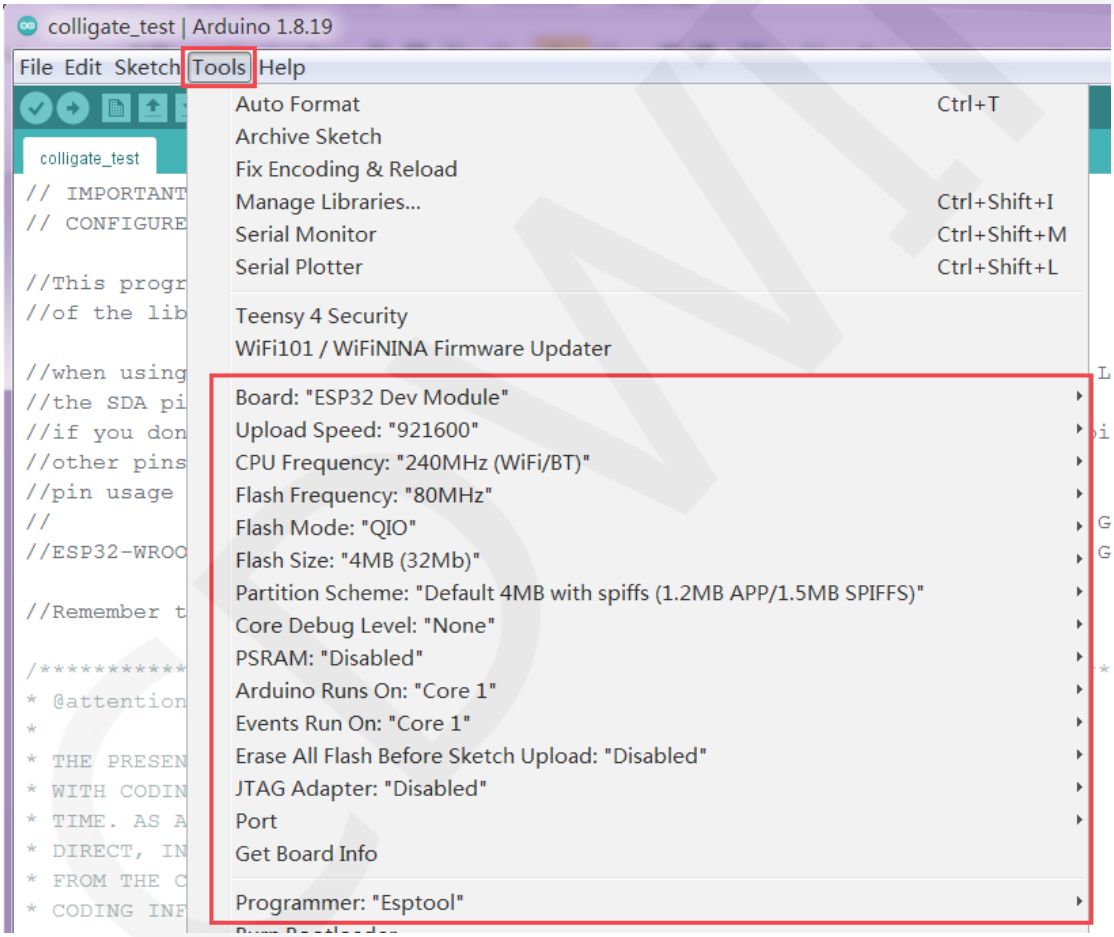
B. Open Any sample program in the **Demo_MSP3222_MSP3223_ESP32-WROOM-32E_HSPI** directory, as shown in the following figure (using the colligate test test program as an example):



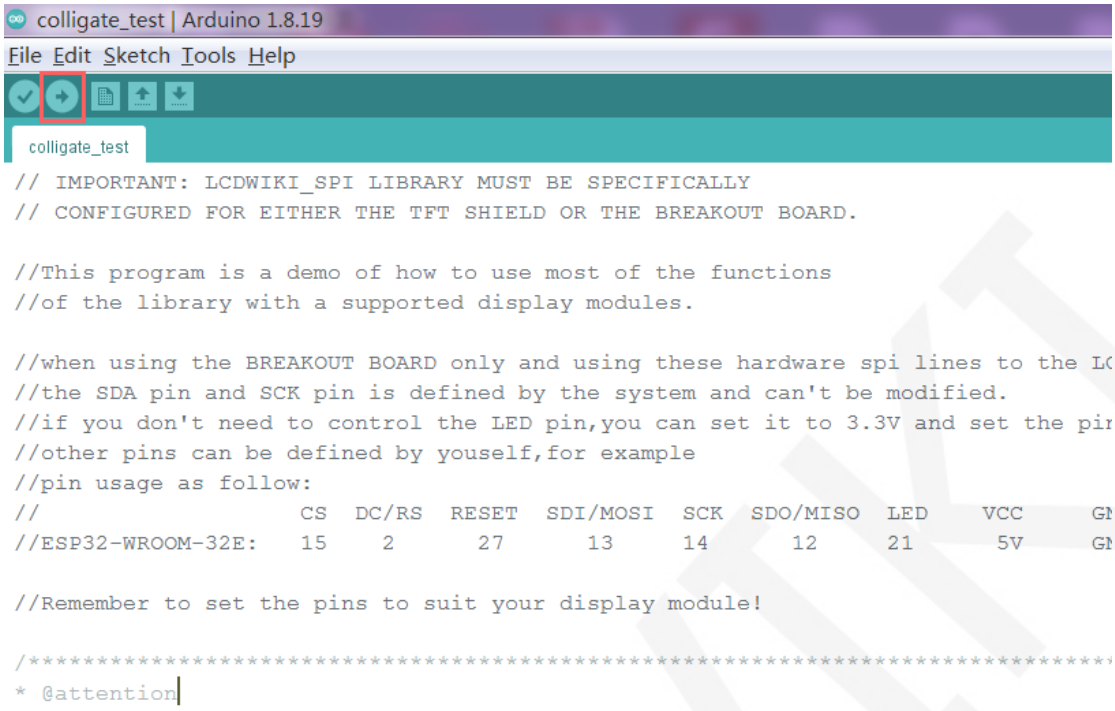
C. After opening the sample program, select the ESP32 device, as shown in the following figure:



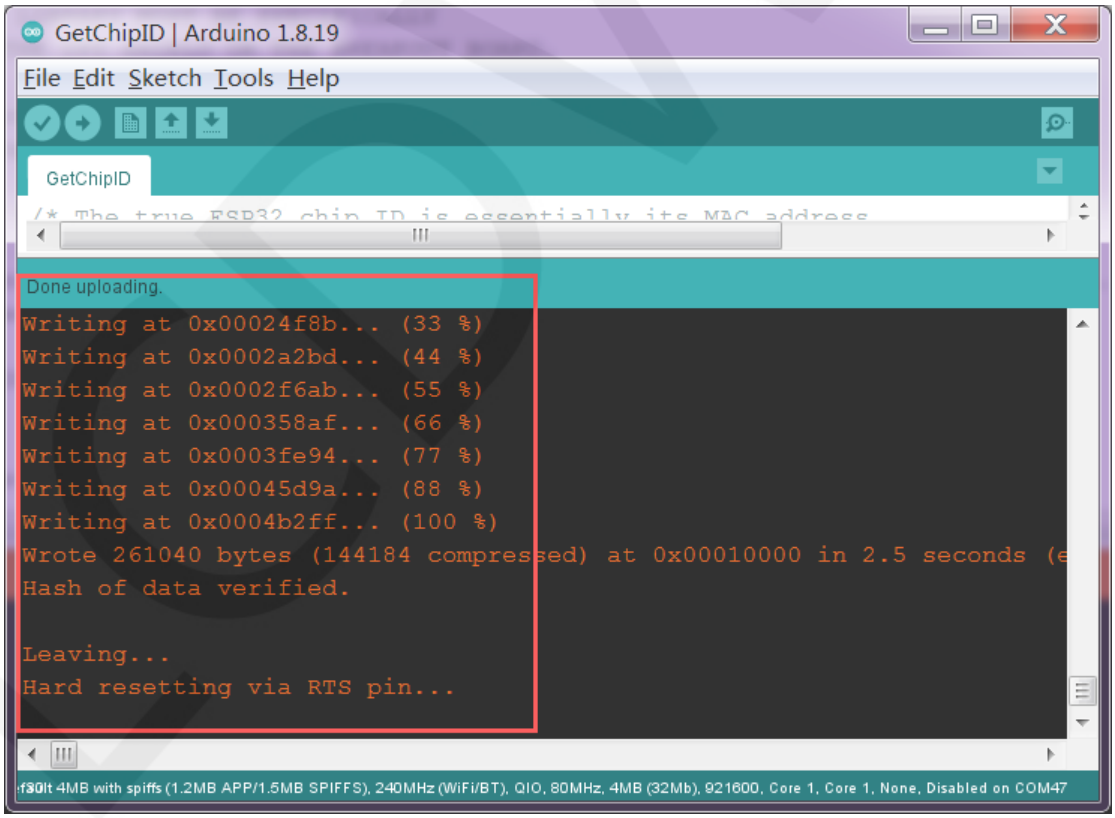
D. Configure ESP32 Flash, PSRAM, ports, etc. as shown in the following figure:



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



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

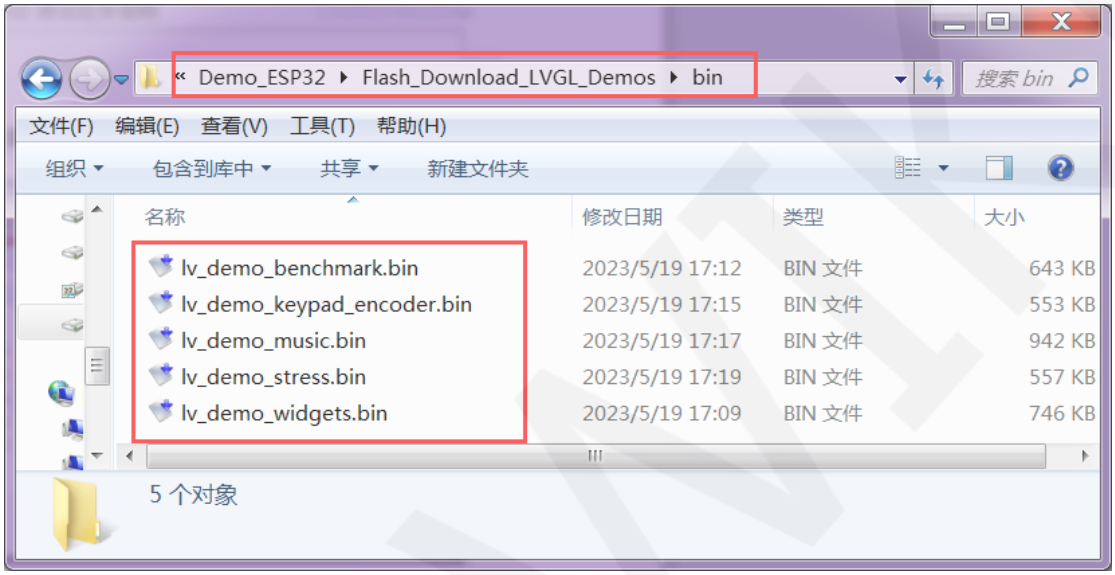


G. If the display module displays content, it indicates that the program has run successfully.

✧ LVGL example bin file burning

Due to the long compilation time of the LVGL sample program, the compiled bin file has been extracted and can be directly burned using the flash download tool.

Bin file located in **Demo_ESP32\Flash_Download_LVGL_Demos\bin** directory,as shown in the following figure:



Using the **flash_download_tool** can burn in the **Demo_ESP32\Flash_Download_LVGL_Demos** directory, as shown in the following figure: