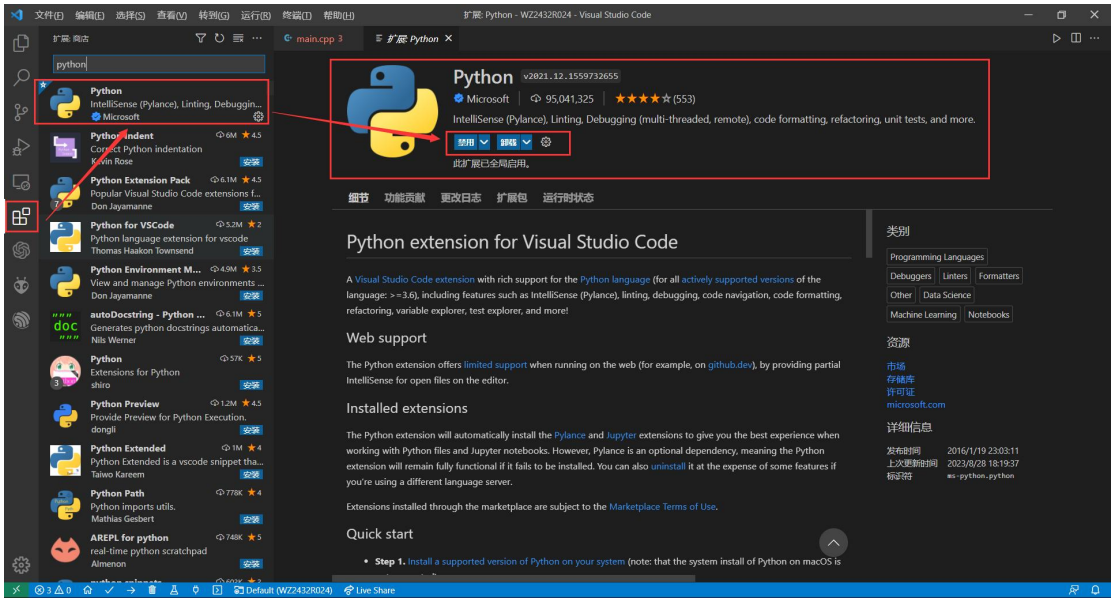
PlatformIO

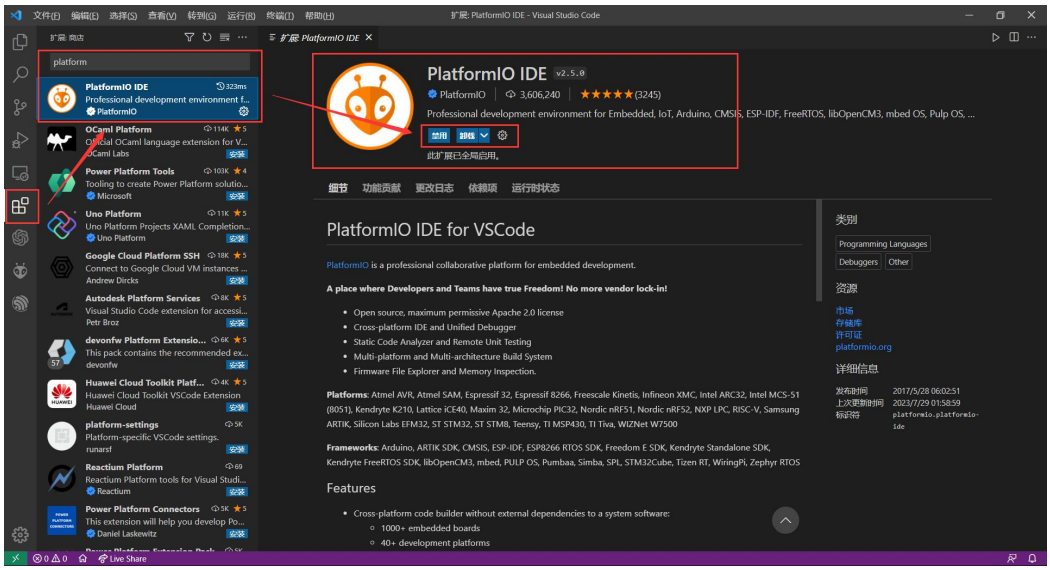
WZ2432R024 or WZ2432R028 or WZ2432R035

Take the WZ2432R024 as an example

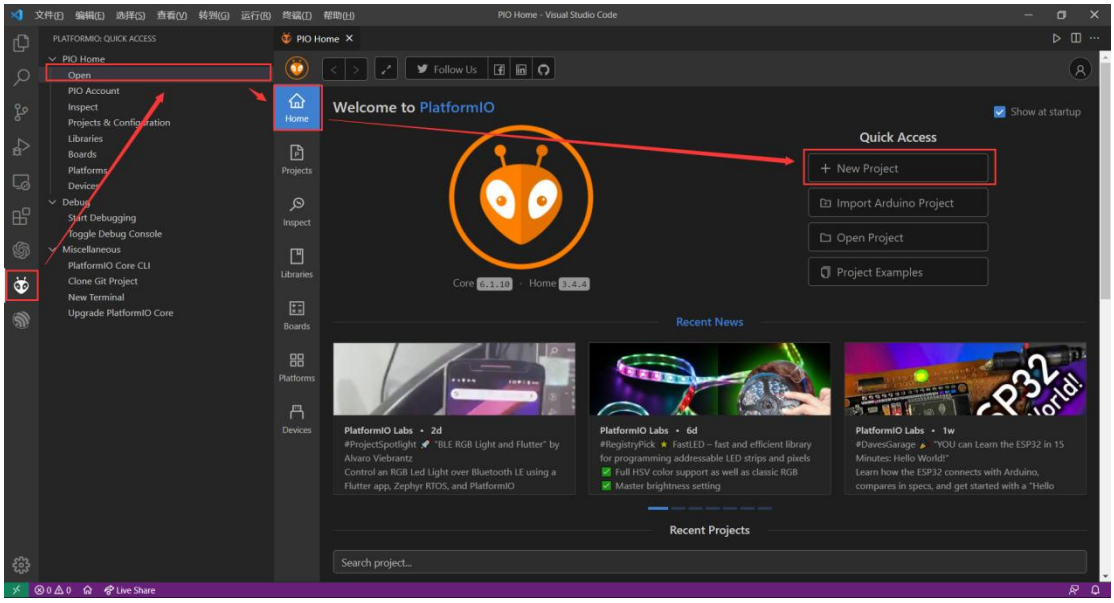
First open the VScode to check if the python is installed

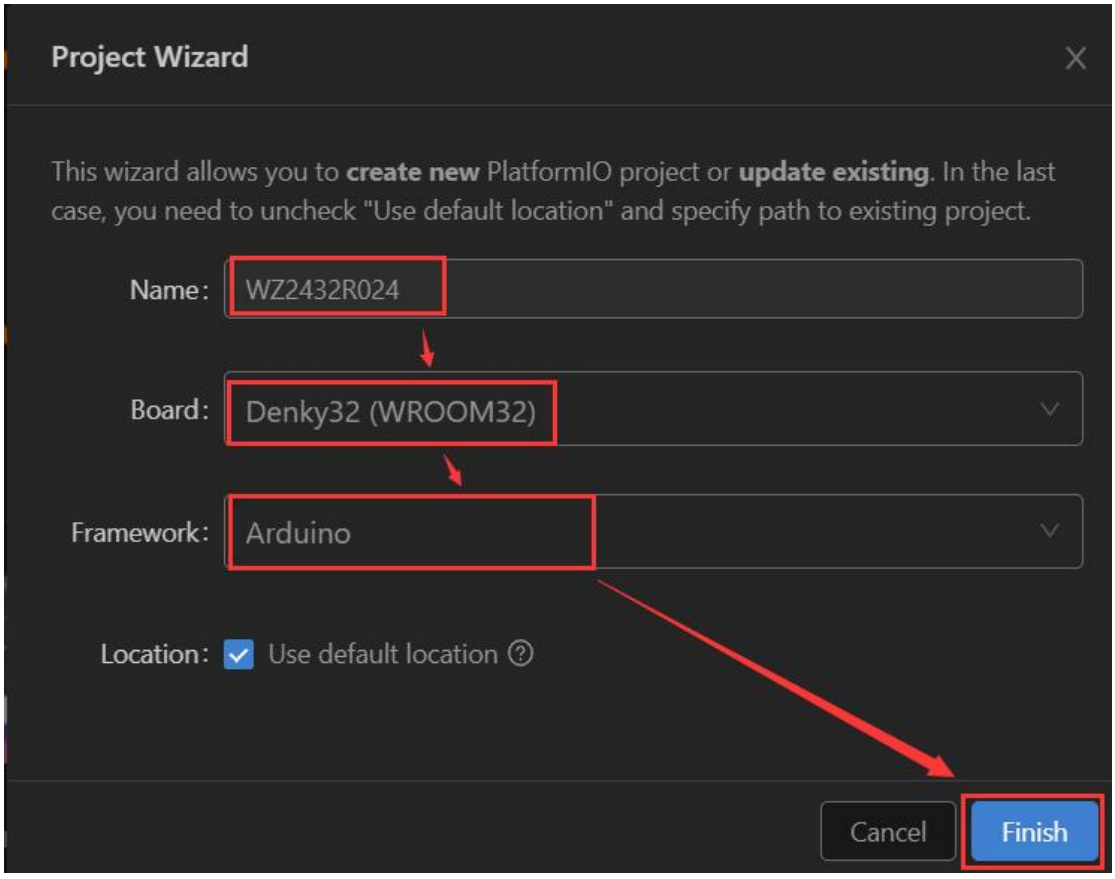


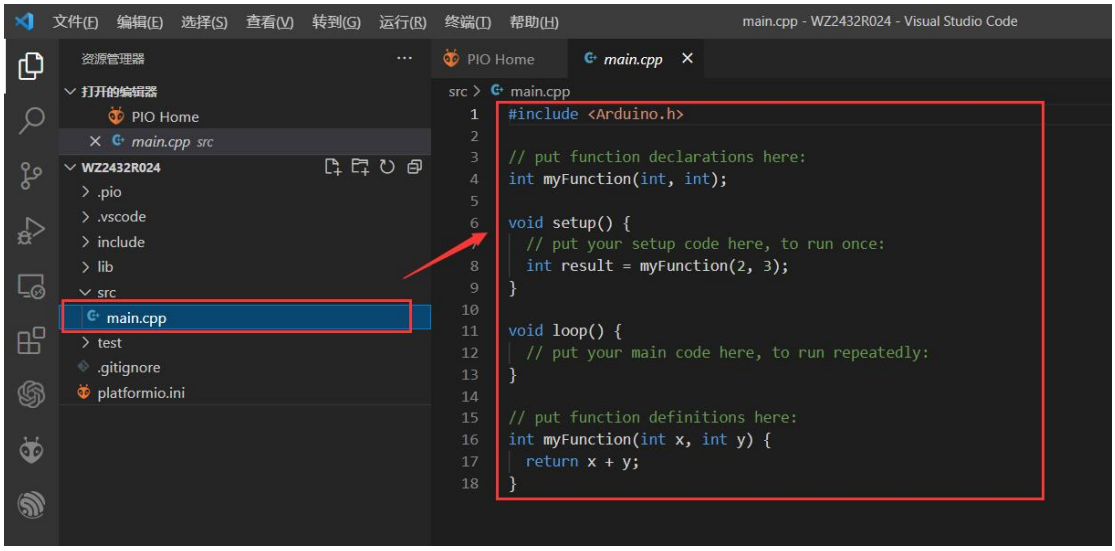
Open the VScode to download the PlatformIO



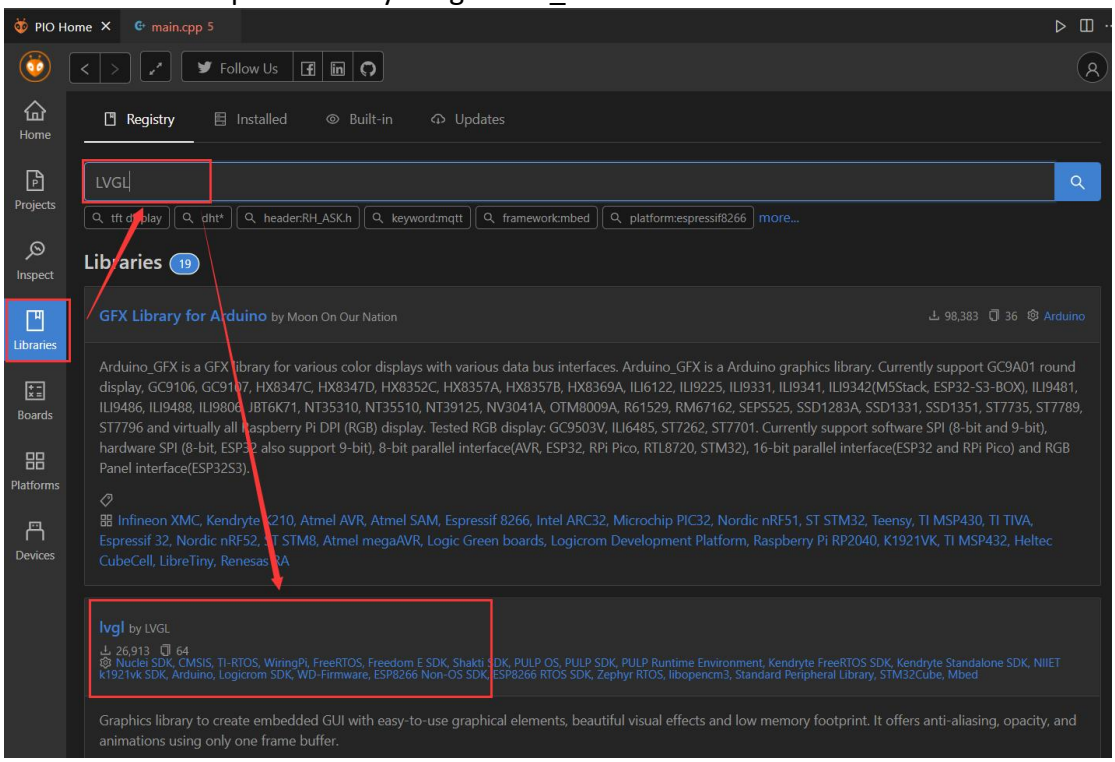
Create new projects

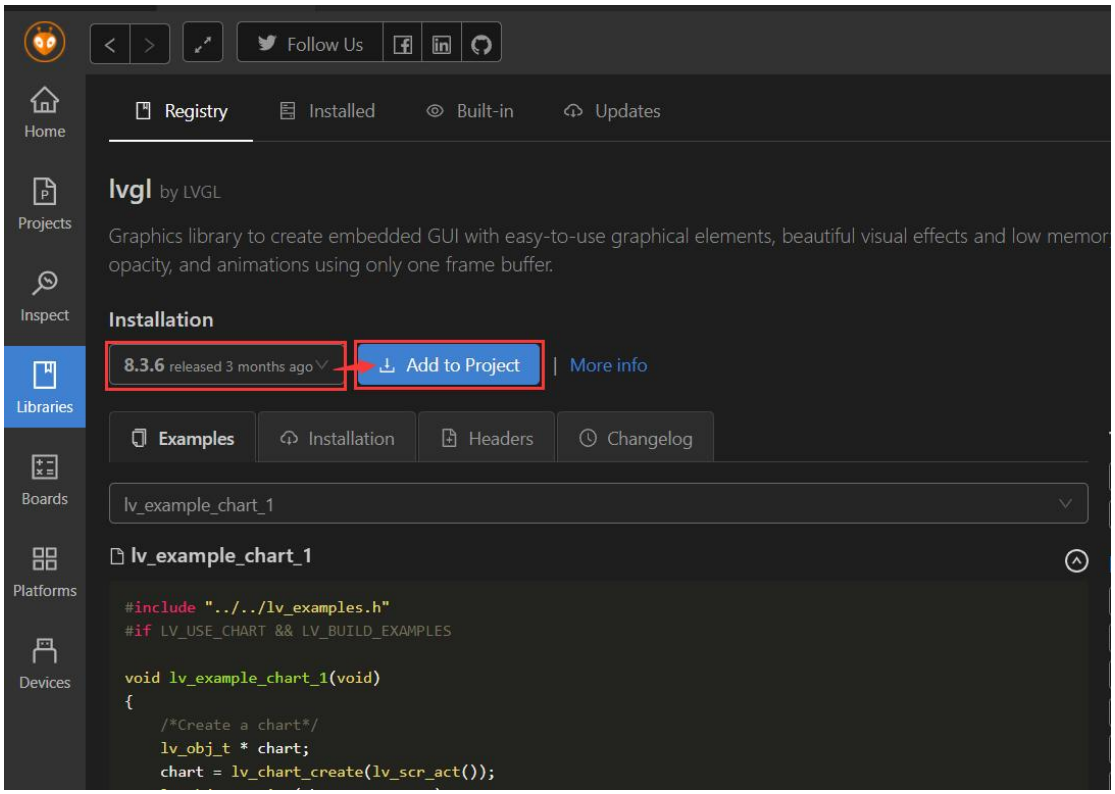


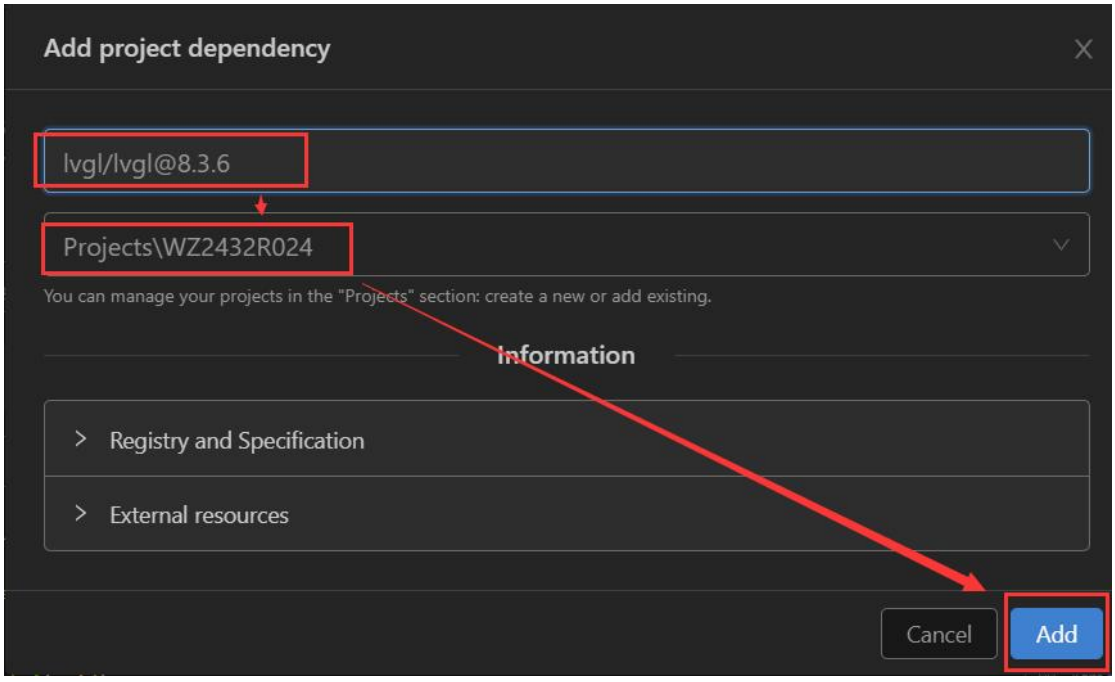


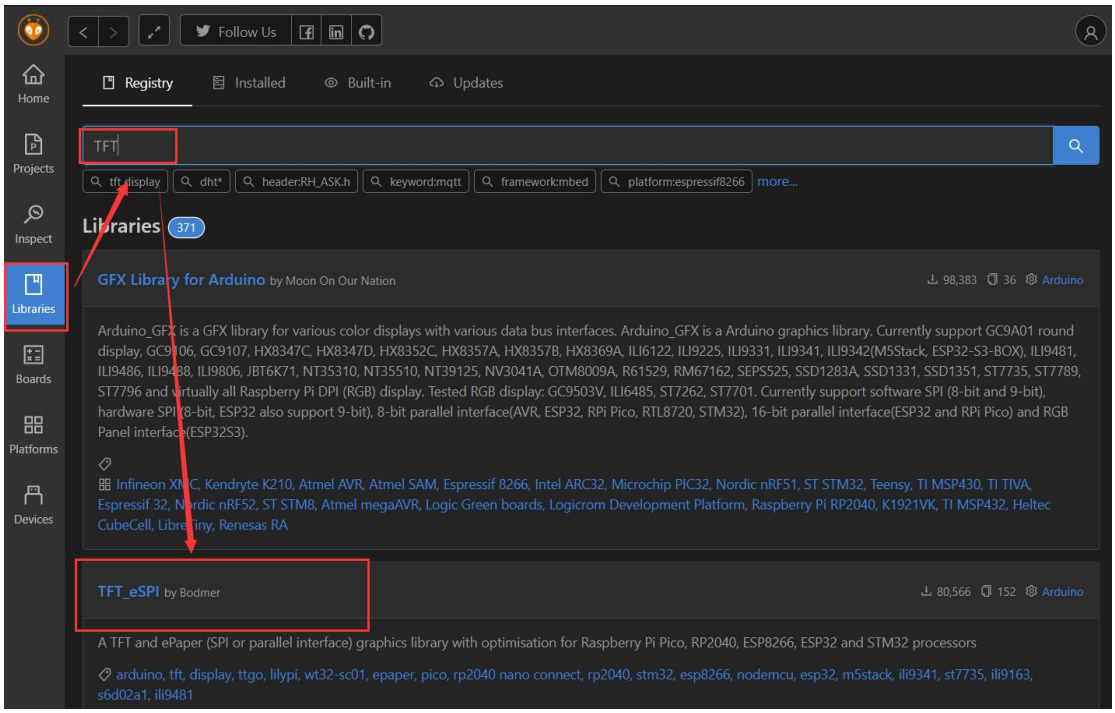


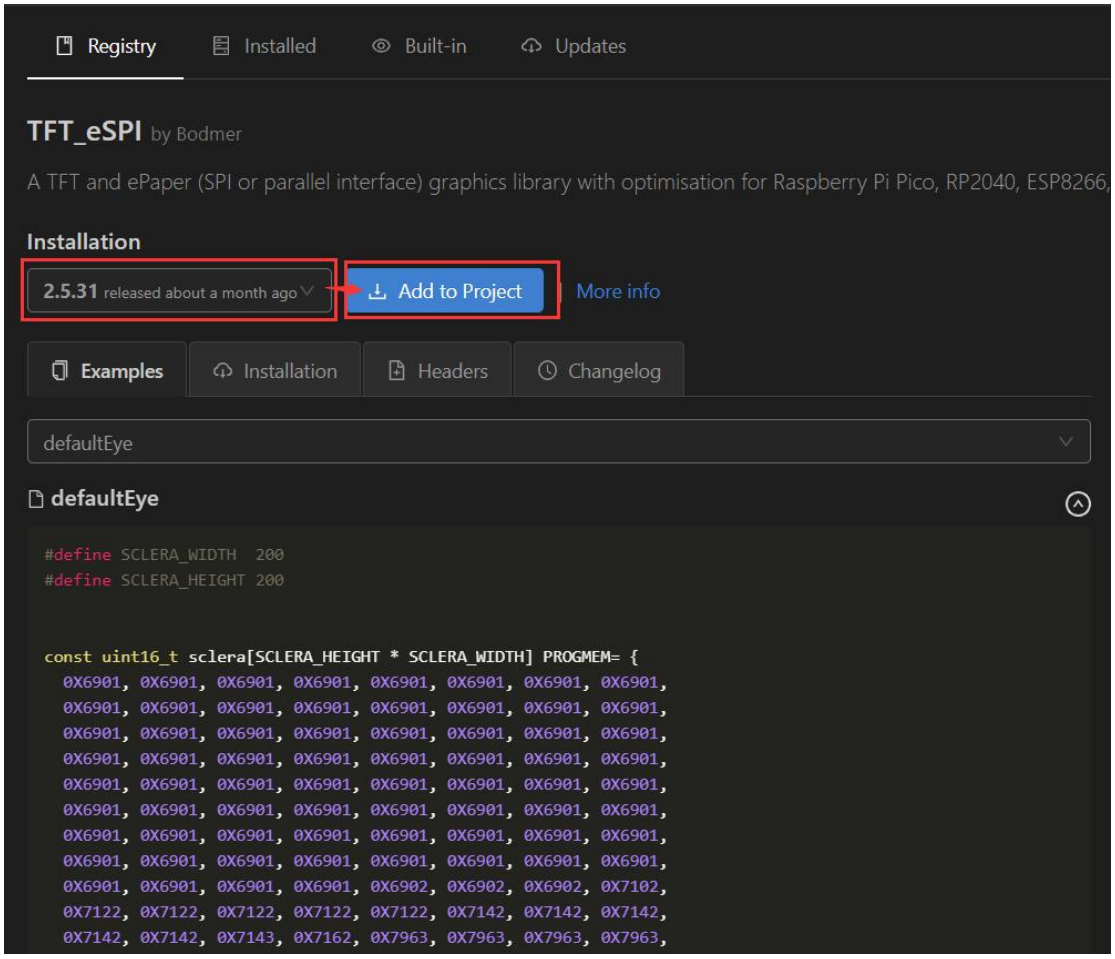
Download the required library（lvgl、TFT_eSPI）

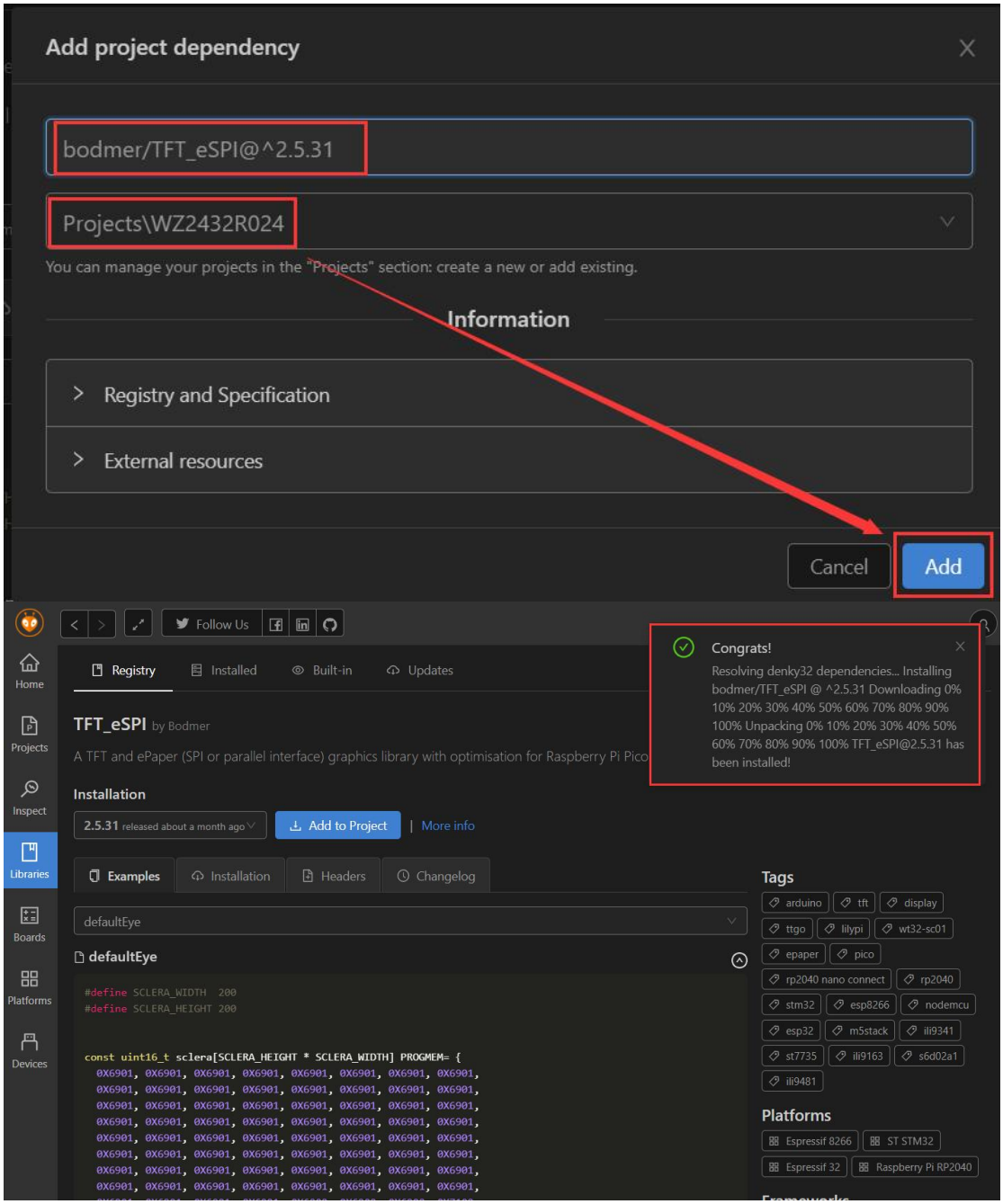



Follow Us
Home
Projects
Inspect
Libraries
Boards
Platforms
Devices
Registry
Installed
Built-in
Updates
lvgl by LVGL
Graphics library to create embedded GUI with easy-to-use graphical elements, beautiful visual effects and low memor
opacity, and animations using only one frame buffer.
Installation
8.3.6 released 3 months ago
Add to Project
More info
Examples
Installation
Headers
Changelog
lv_example_chart_1
lv_example_chart_1
#include "../../lv_examples.h"
#if LV_USE_CHART && LV_BUILD_EXAMPLES
void lv_example_chart_1(void)
{
/*Create a chart*/
lv_obj_t * chart;
chart = lv_chart_create(lv_scr_act());

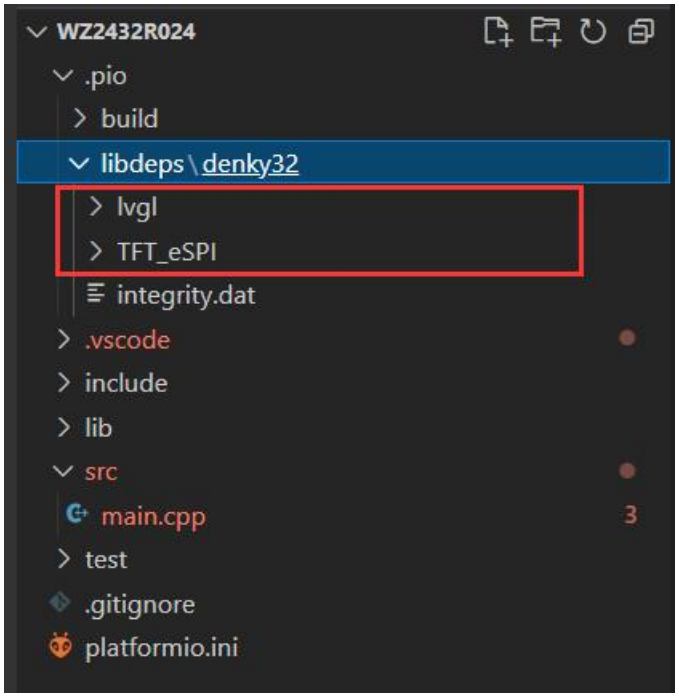

Add project dependency
lvgl/lvgl@8.3.6
Projects\WZ2432R024
You can manage your projects in the "Projects" section: create a new or add existing.
Information
Registry and Specification
External resources
Cancel
Add


Follow Us
Home
Projects
Inspect
Libraries
Boards
Platforms
Devices
Registry
Installed
Built-in
Updates
TFT
tft_display
dht*
header:RH_ASK.h
keyword:mqtt
framework:mbed
platform:espressif8266
more...
Libraries 371
GFX Library for Arduino by Moon On Our Nation
98,383 36 Arduino
Arduino_GFX is a GFX library for various color displays with various data bus interfaces. Arduino_GFX is a Arduino graphics library. Currently support GC9A01 round display, GC9106, GC9107, HX8347C, HX8347D, HX8352C, HX8357A, HX8357B, HX8369A, ILI6122, ILI9225, ILI9331, ILI9341, ILI9342(M5Stack, ESP32-S3-BOX), ILI9481, ILI9486, ILI9488, ILI9806, JBT6K71, NT35310, NT35510, NT39125, NV3041A, OTM8009A, R61529, RM67162, SEPS525, SSD1283A, SSD1331, SSD1351, ST7735, ST7789, ST7796 and virtually all Raspberry Pi DPI (RGB) display. Tested RGB display: GC9503V, ILI6485, ST7262, ST7701. Currently support software SPI (8-bit and 9-bit), hardware SPI (8-bit, ESP32 also support 9-bit), 8-bit parallel interface(AVR, ESP32, RPi Pico, RTL8720, STM32), 16-bit parallel interface(ESP32 and RPi Pico) and RGB Panel interface(ESP32S3).
Infineon XMC, Kendryte K210, Atmel AVR, Atmel SAM, Espressif 8266, Intel ARC32, Microchip PIC32, Nordic nRF51, ST STM32, Teensy, TI MSP430, TI TIVA, Espressif 32, Nordic nRF52, ST STM8, Atmel megaAVR, Logic Green boards, Logicrom Development Platform, Raspberry Pi RP2040, K1921VK, TI MSP432, Heltec CubeCell, Libre Tiny, Renesas RA
TFT_eSPI by Bodmer
80,566 152 Arduino
A TFT and ePaper (SPI or parallel interface) graphics library with optimisation for Raspberry Pi Pico, RP2040, ESP8266, ESP32 and STM32 processors
arduino, tft, display, ttgo, lilypi, wt32-sc01, epaper, pico, rp2040 nano connect, rp2040, stm32, esp8266, nodemcu, esp32, m5stack, ili9341, st7735, ili9163, s6d02a1, ili9481

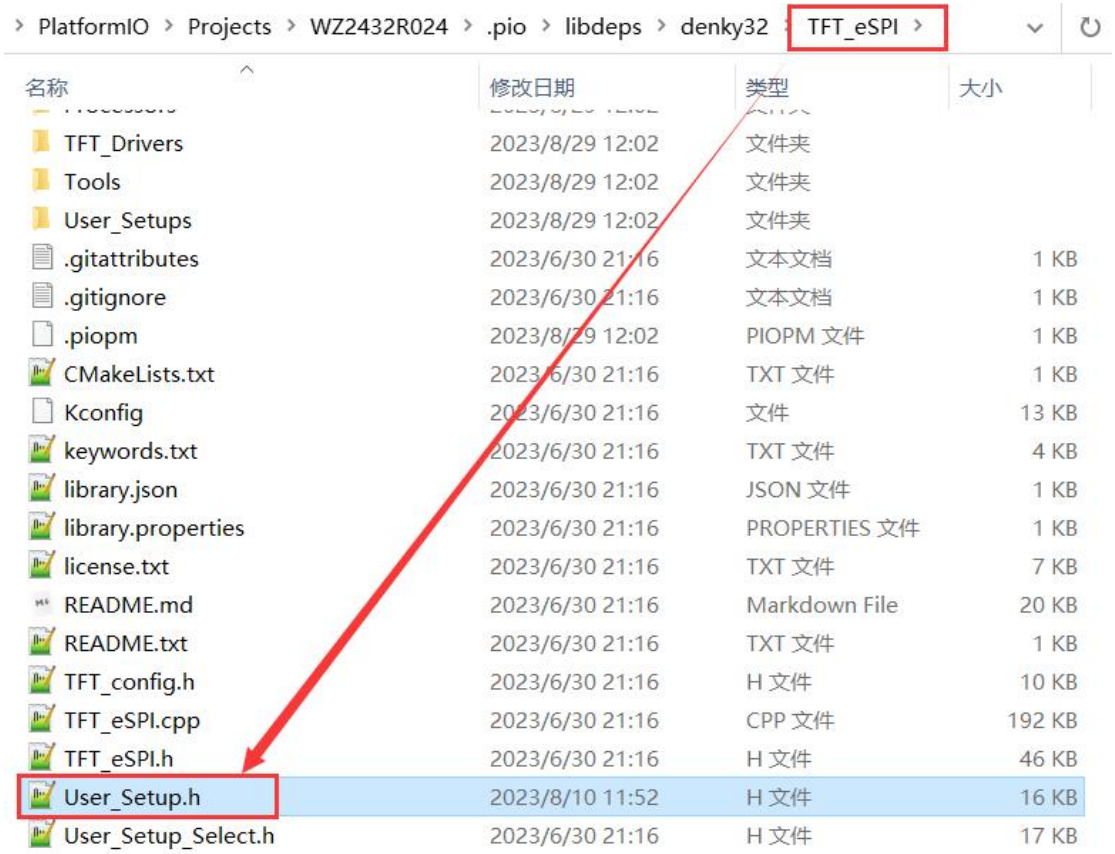

Registry
Installed
Built-in
Updates
TFT_eSPI by Bodmer
A TFT and ePaper (SPI or parallel interface) graphics library with optimisation for Raspberry Pi Pico, RP2040, ESP8266,
Installation
2.5.31 released about a month ago
Add to Project
More info
Examples
Installation
Headers
Changelog
defaultEye
defaultEye
#define SCLERA_WIDTH 200
#define SCLERA_HEIGHT 200
const uint16_t sclera[SCLERA_HEIGHT * SCLERA_WIDTH] PROGMEM= {
0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901,
0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901,
0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901,
0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901,
0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901,
0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901,
0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901,
0X6901, 0X6901, 0X6901, 0X6901, 0X6902, 0X6902, 0X6902, 0X7102,
0X7122, 0X7122, 0X7122, 0X7122, 0X7122, 0X7142, 0X7142, 0X7142,
0X7142, 0X7142, 0X7143, 0X7162, 0X7963, 0X7963, 0X7963, 0X7963,



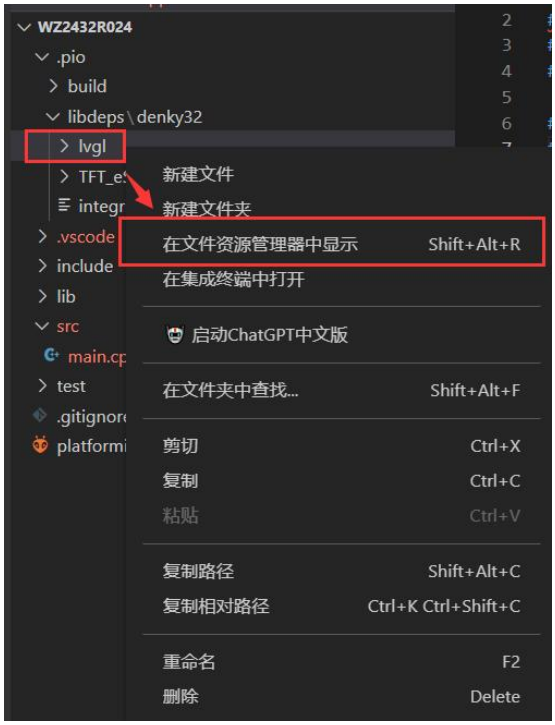
We can see that the library has been added successfully!

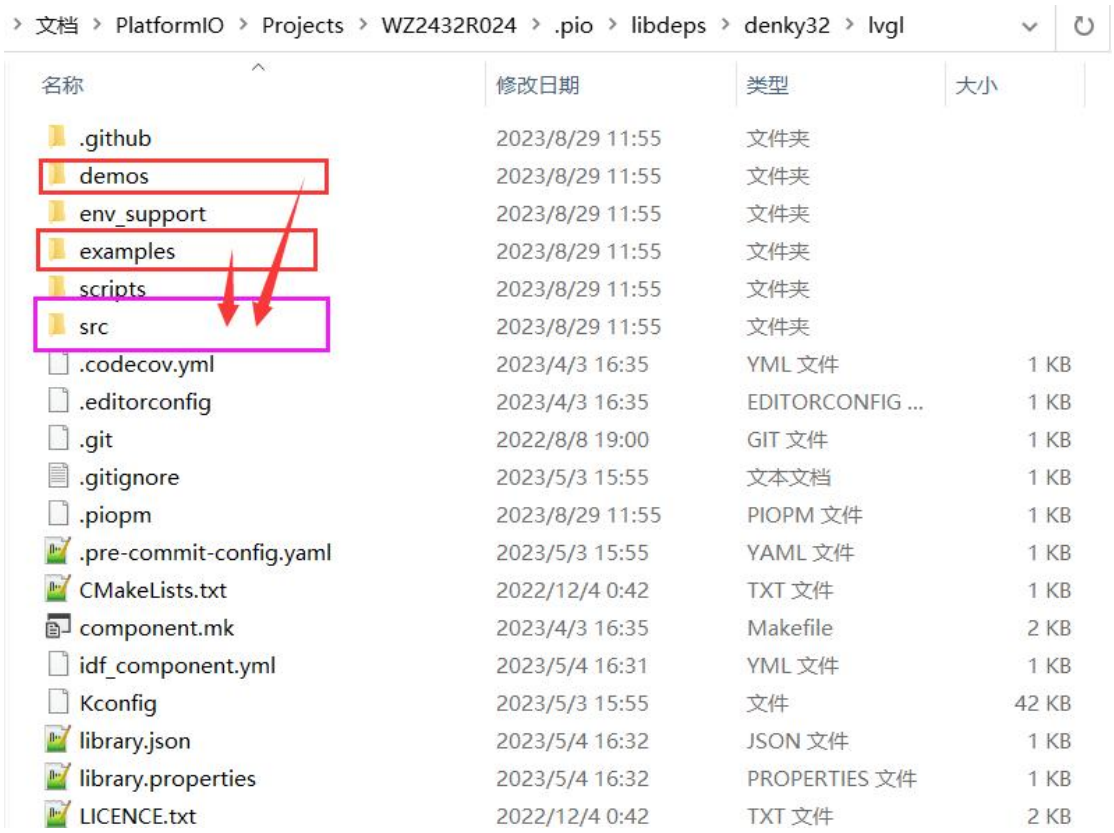


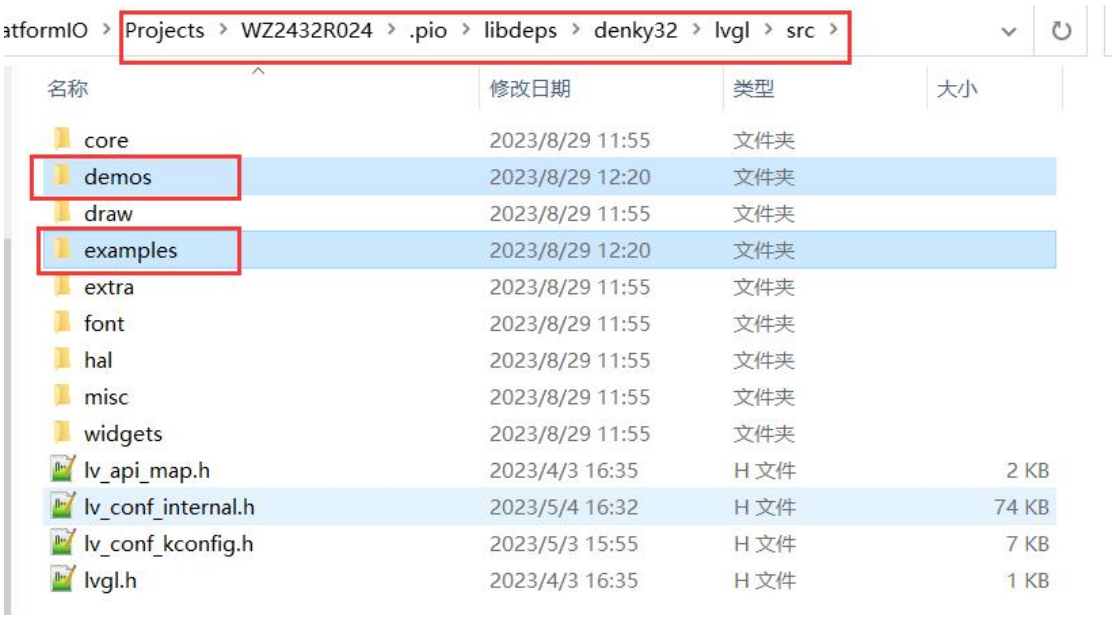
Next, we want to configure the TFT _ eSPI library to replace the User_Setup.h in the library with the file we provide



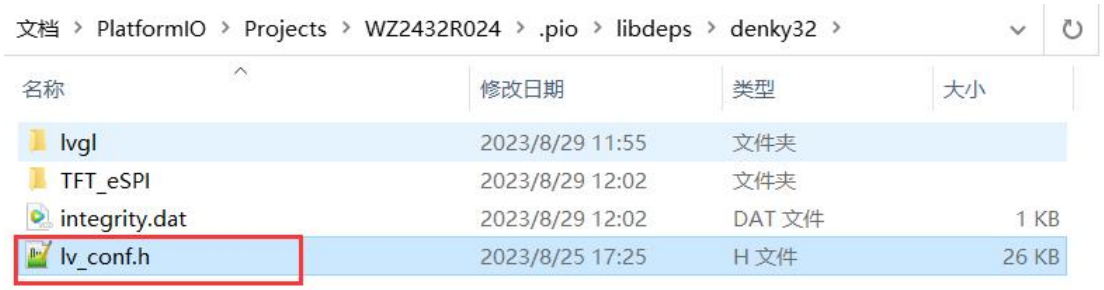
Next, we want to configure the lvgl library, right-click to open the folder directory, and put the demo and examples folders into the src folder!



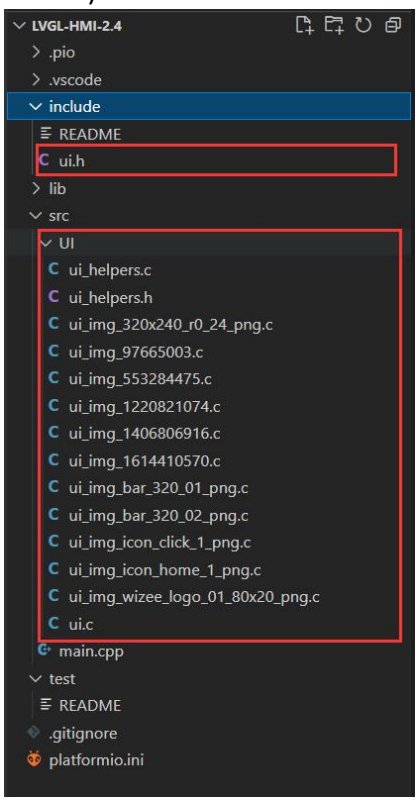




Place the lv_conf.h file under this directory again

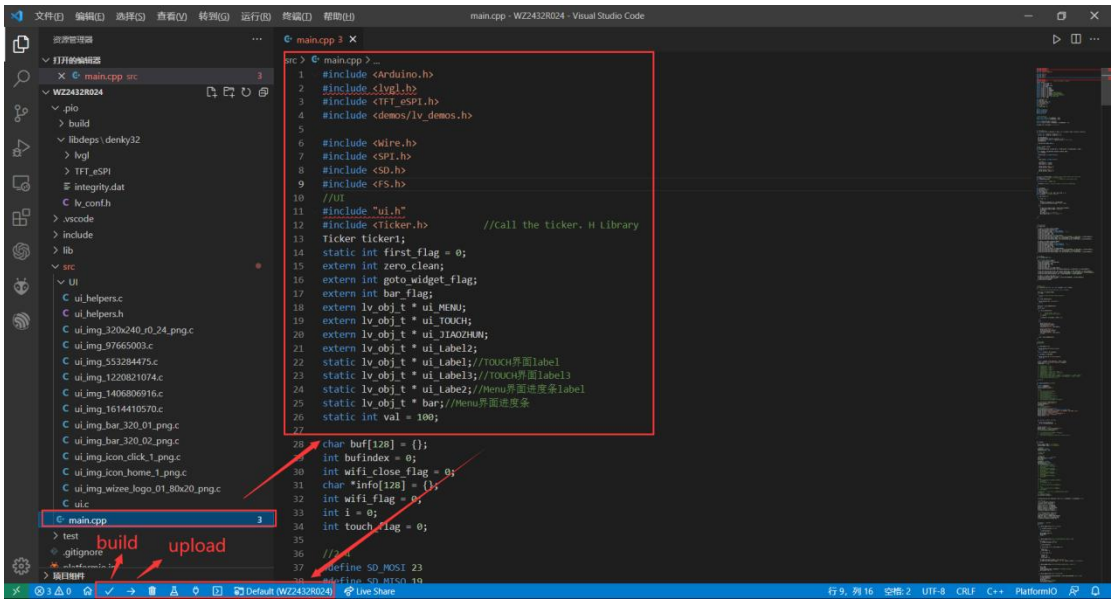


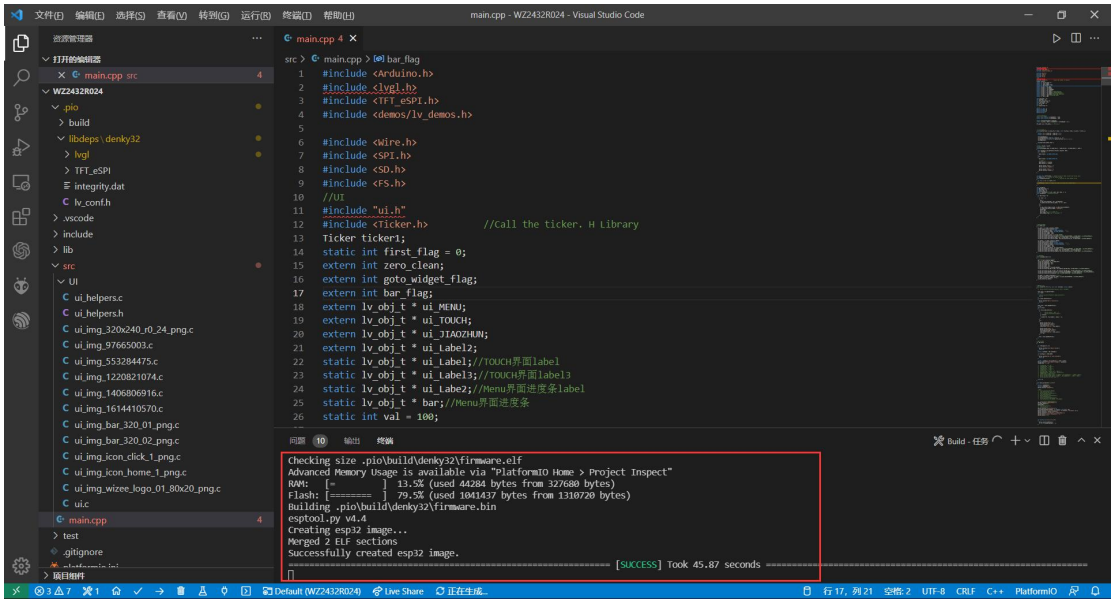
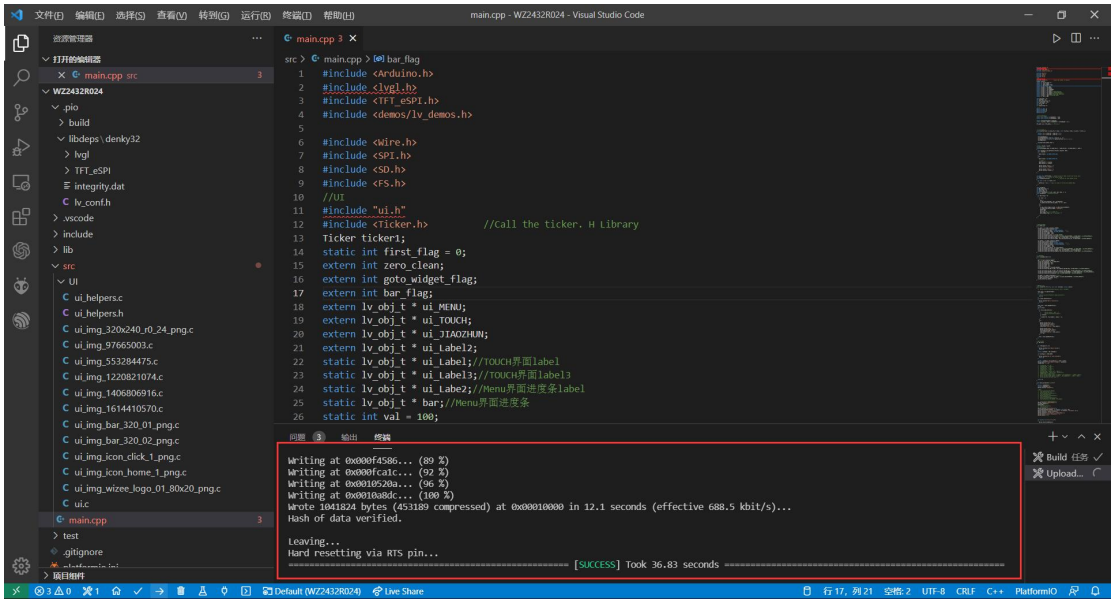
Next, let's configure our own UI files (the UI files are generated from the SquareLine Studio)



In the UI folder that will be generated. The c file is placed in the /src folder, and in the generated UI folder. Place the h file in the /include folder

At this time, we will complete all the configuration, write the code and start compiling the program





Next we began to burn the program, finished!