ESP32-S3-LCD-2

waveshare.com/wiki/ESP32-S3-LCD-2

ESP32-S3-LCD-2

2inch, 240×320, SPI

Overview

Usage Instructions

ESP32-S3-LCD-2 currently provides two development tools and frameworks, **Arduino IDE** and **ESP-IDF**, providing flexible development options, you can choose the right development tool according to your project needs and personal habits.

Development Tools

Arduino IDE

Arduino IDE is an open source electronic prototyping platform, convenient and flexible, easy to get started. After a simple learning, you can start to develop quickly. At the same time, Arduino has a large global user community, providing an abundance of open source code, project examples and tutorials, as well as rich library resources, encapsulating complex functions, allowing developers to quickly implement various functions.

ESP-IDF

ESP-IDF, or full name Espressif IDE, is a professional development framework introduced by Espressif Technology for the ESP series chips. It is developed using the C language, including a compiler, debugger, and flashing tool, etc., and can be developed via the command lines or through an integrated development environment (such as Visual Studio Code with the Espressif IDF plugin). The plugin offers features such as code navigation, project management, and debugging, etc..

Each of these two development approaches has its own advantages, and developers can choose according to their needs and skill levels. Arduino are suitable for beginners and non-professionals because they are easy to learn and quick to get started. ESP-IDF is a better choice for developers with a professional background or high performance requirements, as it provides more advanced development tools and greater control capabilities for the development of complex projects.

Components Preparation

- ESP32-S3-LCD-2 x1
- OV5640 camera x1
- TF card x 1 (32GB and below)
- Card reader x1

Before operating, it is recommended to browse the table of contents to quickly understand the document structure. For smooth operation, please read the carefully to understand possible problems in advance. All resources in the document are provided with hyperlinks for easy download.

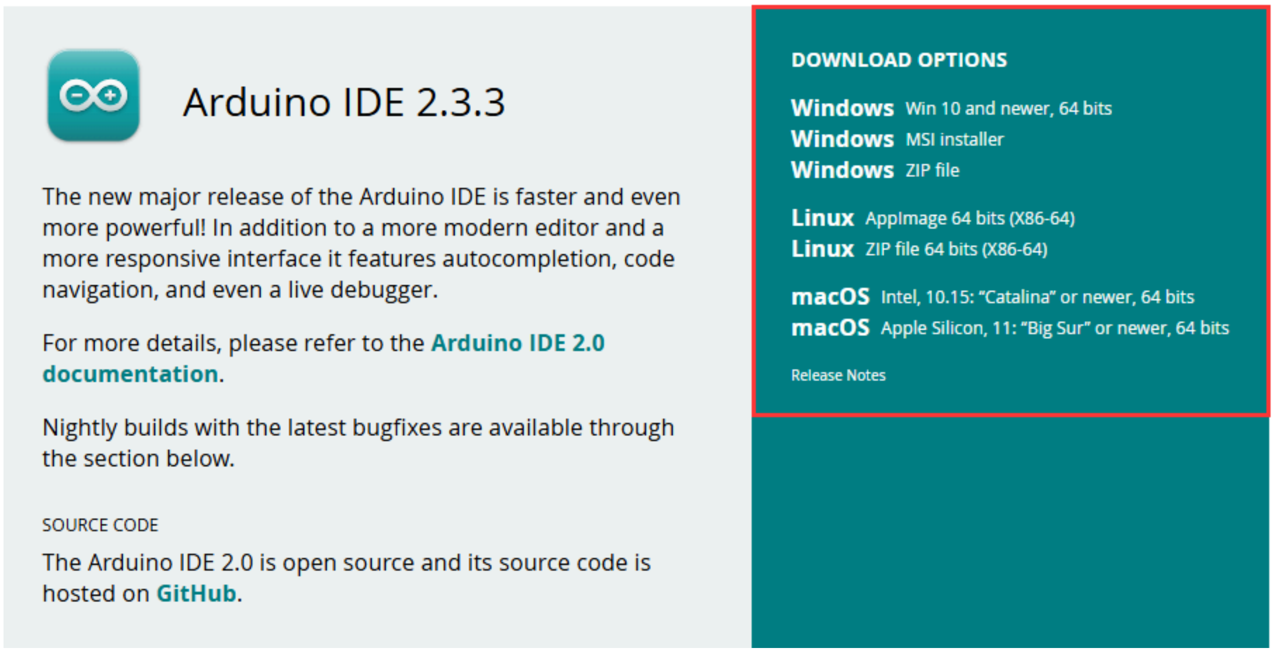
Working with Arduino

This chapter introduces setting up the Arduino environment, including the Arduino IDE, management of ESP32 boards, installation of related libraries, program compilation and downloading, as well as testing demos. It aims to help users master the development board and facilitate secondary development.

Environment Setup

Download and Install Arduino IDE

- Click to visit the Arduino official website, select the corresponding system and system bit to download



Arduino IDE 2.3.3

The new major release of the Arduino IDE is faster and even more powerful! In addition to a more modern editor and a more responsive interface it features autocompletion, code navigation, and even a live debugger.

For more details, please refer to the **Arduino IDE 2.0 documentation**.

Nightly builds with the latest bugfixes are available through the section below.

SOURCE CODE

The Arduino IDE 2.0 is open source and its source code is hosted on **GitHub**.

DOWNLOAD OPTIONS

Windows Win 10 and newer, 64 bits
Windows MSI installer
Windows ZIP file

Linux AppImage 64 bits (X86-64)
Linux ZIP file 64 bits (X86-64)

macOS Intel, 10.15: "Catalina" or newer, 64 bits
macOS Apple Silicon, 11: "Big Sur" or newer, 64 bits

Release Notes

- Run the installer and install all by default

Install ESP32 Development Board

- Before using ESP32-related motherboards with the Arduino IDE, you must first install the software package for the **esp32 by Espressif Systems** development board
- According to **board installation requirement**, it is generally recommended to use **Install Online**. If online installation fails, use **Install Offline**.
- For the installation tutorial, please refer to Arduino board manager tutorial

Board name	Board installation requirement	Version number requirement
esp32 by Espressif Systems	"Install Offline" / "Install Online"	≥3.0.0

ESP32-S3-LCD-2 required development board installation description

Install Library

- When installing Arduino libraries, there are usually two ways to choose from: **Install online** and **Install offline**. **If the library installation requires offline installation, you must use the provided library file**
  For most libraries, users can easily search and install them through the online library manager of the Arduino software. However, some open-source libraries or custom libraries are not synchronized to the Arduino Library Manager, so they cannot be acquired through online searches. In this case, users can only manually install these libraries offline.
- For library installation tutorial, please refer to Arduino library manager tutorial
- **ESP32-S3-LCD-2 library file path:**

```
..\ESP32-S3-LCD-2-Demo\Arduino\libraries
```

Library Name	Description	Version	Library Installation Requirement
lvgl	Graphical library	v8.4.0	"Install Online" (requires copying the demos folder to src)
GFX_Library_for_Arduino	LCD driver library	v1.5.0	"Install Online"
FastIMU	IUM driver library	v1.2.6	"Install Online"
OneButton	Button driver library	v2.6.1	"Install Online"

ESP32-S3-LCD-2 library file installation description

Run the First Arduino Demo

If you are just getting started with ESP32 and Arduino, and you don't know how to create, compile, flash, and run Arduino ESP32 programs, then please expand and take a look. Hope it can help you!

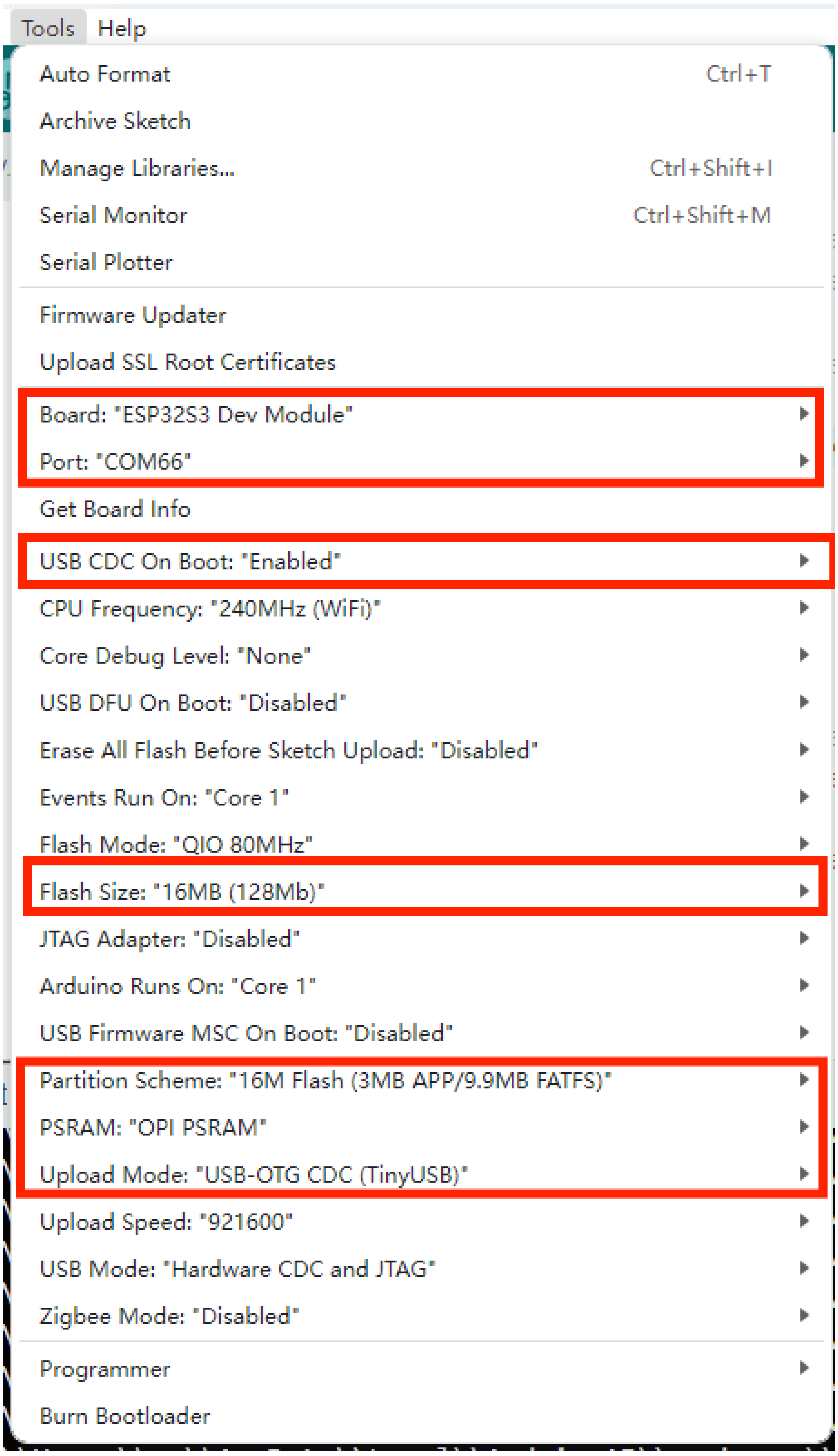
Demo

Demo	Basic Description	Dependency Library
01_factory	Comprehensive testing program	lvgl, GFX_Library_for_Arduino, FastIMU, OneButton
02_gfx_helloworld	HelloWorld is displayed on the screen	GFX_Library_for_Arduino
03_sd_card_test	Test TF card	--

04_qmi8658_output	Serial port prints QMI8658 data	FastIMU
05_lvgl_qmi8658	Use the LVGL library to display QMI8658 data	lvgl, FastIMU
06_lvgl_battery	Use the LVGL library to display the battery voltage	lvgl, GFX_Library_for_Arduino
07_lvgl_brightness	Use the LVGL library to control and display screen brightness	lvgl, GFX_Library_for_Arduino
08_lvgl_example	lvgl demos	lvgl, GFX_Library_for_Arduino
09_lvgl_camera	Use the LVGL library to display the camera images	lvgl, GFX_Library_for_Arduino
10_camera_web_server	Display the camera images on the web page	--

ESP32-S3-LCD-2 demos

Arduino Project Parameter Setting


Tools Help

Auto Format Ctrl+T
Archive Sketch
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter
Firmware Updater
Upload SSL Root Certificates
Board: "ESP32S3 Dev Module"
Port: "COM66"
Get Board Info
USB CDC On Boot: "Enabled"
CPU Frequency: "240MHz (WiFi)"
Core Debug Level: "None"
USB DFU On Boot: "Disabled"
Erase All Flash Before Sketch Upload: "Disabled"
Events Run On: "Core 1"
Flash Mode: "QIO 80MHz"
Flash Size: "16MB (128Mb)"
JTAG Adapter: "Disabled"
Arduino Runs On: "Core 1"
USB Firmware MSC On Boot: "Disabled"
Partition Scheme: "16M Flash (3MB APP/9.9MB FATFS)"
PSRAM: "OPI PSRAM"
Upload Mode: "USB-OTG CDC (TinyUSB)"
Upload Speed: "921600"
USB Mode: "Hardware CDC and JTAG"
Zigbee Mode: "Disabled"
Programmer
Burn Bootloader

01_factory

【Demo description】

This demo tests the functionality of the ESP32-S3-LCD-2 onboard module, the information of each module will be displayed on the screen, and the user can switch between pages through the touch screen

【Hardware connection】

- Connect the board to the computer
- Plug the OV5640 camera into the 24Pin socket on the board (if it is not available, it will not affect the use of other functions)
- Insert the TF card into the card slot (if it is not available, it will not affect the use of other functions)

【Code analysis】

Hardware initialization

```
Serial.begin(115200);
bsp_i2c_init();          //Initialize I2C bus, commonly used to connect sensors,
displays, etc.
bsp_lv_port_init();      //Initialize the LVGL graphics library port to output graphics
to display devices such as LCD
bsp_spi_init();          //Initialize SPI bus, commonly used for communication with
sensors, external storage, screens, etc.
```

LVGL graphics library configuration

```
bsp_lv_port_run();       //Run LVGL-related drivers or loops, such as refreshing the
display, etc.
if (lvgl_lock(-1)) {     //Lock LVGL resources to ensure they do not conflict with
other tasks when updating the UI
  lvgl_ui_init();        //Initialize UI, possibly including layout and control
settings
  lvgl_unlock();         //Unlock LVGL, allowing other tasks to access LVGL
}
```

Peripheral initialization

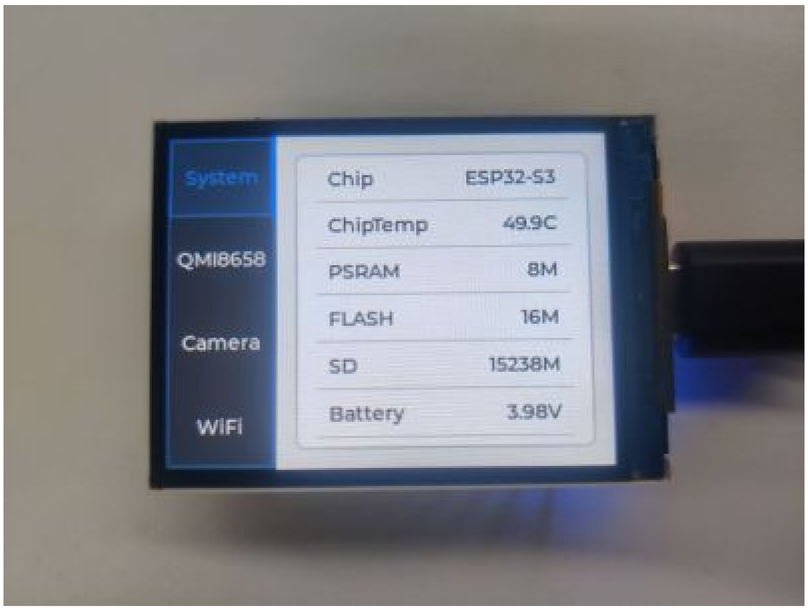
```
app_qmi8658_init();                     //Initialize QMI8658 sensor (such as
accelerometer, gyroscope, etc.)
app_system_init();                      //Perform system-level initialization, such as
clock, memory, reset, etc
app_camera_init();                      //Initialize the camera module, prepare for
image capture
app_wifi_init(sta_ssid, sta_pass);      //Initialize Wi-Fi module, connect to specified
Wi-Fi network
```

Application layer tasks running

```
app_qmi8658_run();       //Start QMI8658 sensor-related tasks, such as data
acquisition, processing, etc.
app_system_run();        //Start system tasks to ensure that system functions are
running properly
app_camera_run();        //Start camera-related tasks, usually to obtain image data and
process or transmit it
app_wifi_run();          //Start Wi-Fi related tasks, such as connection management,
data transmission, etc.
```

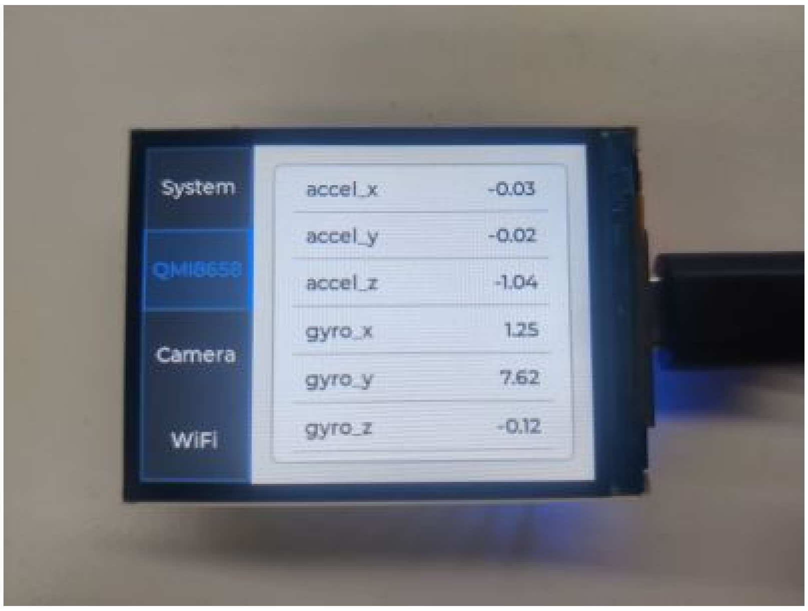
【Result demonstration】

- Click the BOOT button to be down, double click the BOOT button to be up, and long press the BOOT button to OK
- System interface



Chip	ESP32-S3
ChipTemp	49.9C
PSRAM	8M
FLASH	16M
SD	15238M
Battery	3.98V

QMI8658 interface, where you can see the acceleration and angular velocity data of the acquired X, Y, and Z axes



accel_x	-0.03
accel_y	-0.02
accel_z	-1.04
gyro_x	1.25
gyro_y	7.62
gyro_z	-0.12

Camera interface, where you can see the images captured by the camera

WiFi interface, showing the WiFi name and password to be connected, IP will be displayed after successful connection

02_gfx_helloworld

【Demo description】

This demo demonstrates the ESP32-S3-LCD-2 using the GFX_Library_for_Arduino library to drive the screen and display HelloWorld on the screen.

【Hardware connection】

Connect the board to the computer

【Code analysis】

Create an object bus of the Arduino_ESP32SPI class to configure the SPI bus GPIO, and create an object gfx of the Arduino_ST7789 class to drive the ST7789 display

```
Arduino_DataBus *bus = new Arduino_ESP32SPI(
  EXAMPLE_PIN_NUM_LCD_DC /* DC */, EXAMPLE_PIN_NUM_LCD_CS /* CS */,
  EXAMPLE_PIN_NUM_LCD_SCLK /* SCK */, EXAMPLE_PIN_NUM_LCD_MOSI /* MOSI */,
EXAMPLE_PIN_NUM_LCD_MISO /* MISO */);

/* More display class: https://github.com/moononournation/Arduino_GFX/wiki/Display-
Class */
Arduino_GFX *gfx = new Arduino_ST7789(
  bus, EXAMPLE_PIN_NUM_LCD_RST /* RST */, EXAMPLE_LCD_ROTATION /* rotation */, true /*
IPS */,
  EXAMPLE_LCD_H_RES /* width */, EXAMPLE_LCD_V_RES /* height */);
```

【Result demonstration】

03_sd_card_test

【Demo description】

This demo demonstrates how to use ESP32-S3-LCD-2 to test the read and write functions of the TF card

【Hardware connection】

- Connect the board to the computer
- Insert the TF card into the card slot **(TF card needs to be formatted as FAT32)**

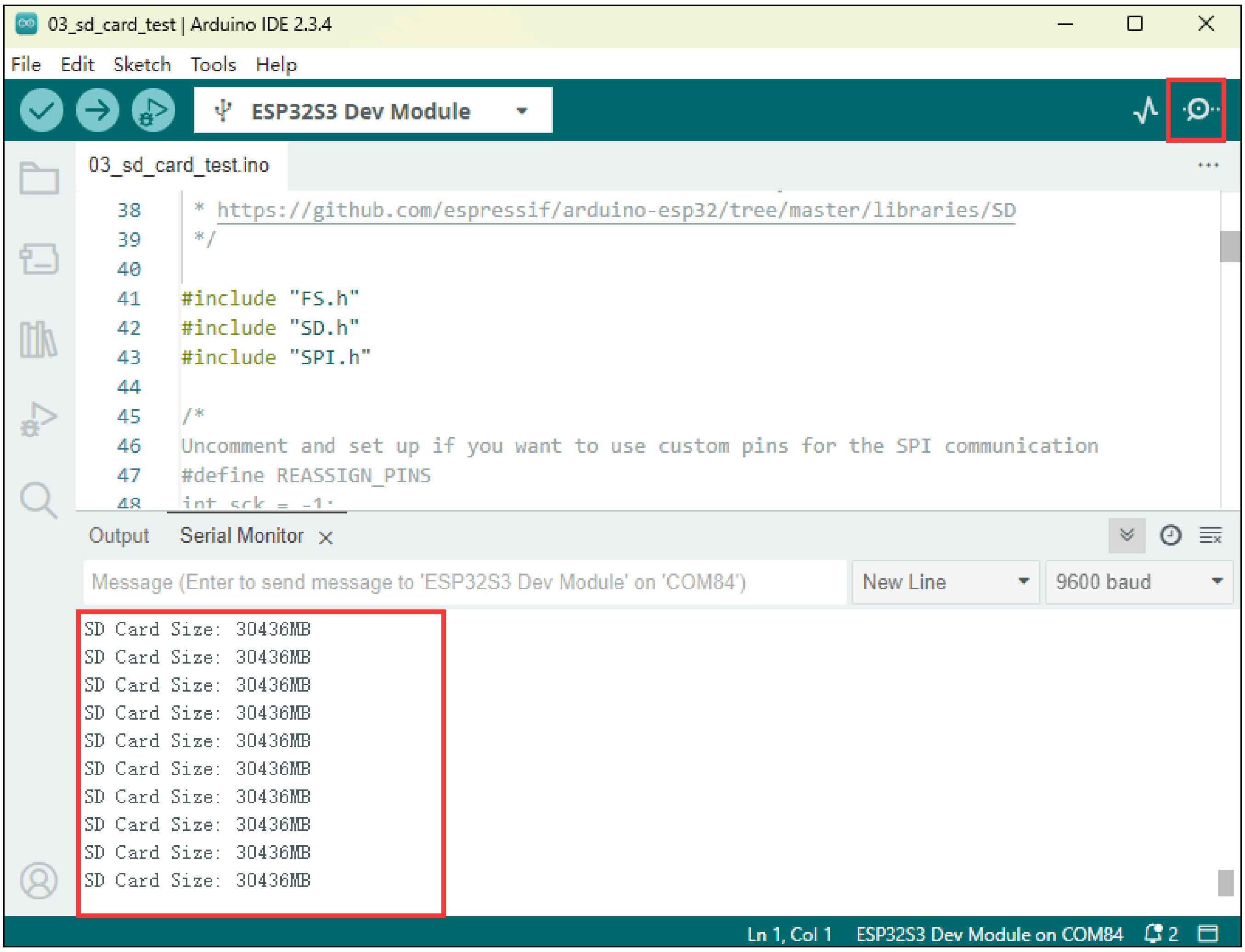
【Code analysis】

SPI interface initialization and TF card initialization

```
SPI.begin(sck, miso, mosi, cs);
if (!SD.begin(cs)) {
  Serial.println("Card Mount Failed");
  return;
}
```

【Result demonstration】

Open the serial port monitor


03_sd_card_test | Arduino IDE 2.3.4
File Edit Sketch Tools Help
ESP32S3 Dev Module
03_sd_card_test.ino
Output
Serial Monitor
Message (Enter to send message to 'ESP32S3 Dev Module' on 'COM84')
New Line
9600 baud
Ln 1, Col 1 ESP32S3 Dev Module on COM84 2

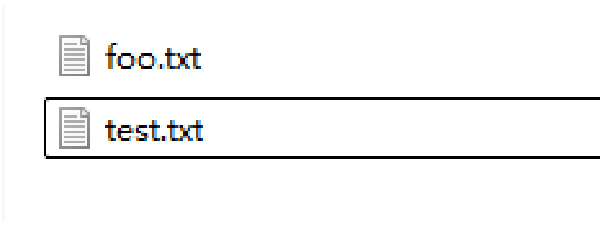
```
 * https://github.com/espressif/arduino-esp32/tree/master/libraries/SD
 */

#include "FS.h"
#include "SD.h"
#include "SPI.h"

/*
Uncomment and set up if you want to use custom pins for the SPI communication
#define REASSIGN_PINS
int sck = -1;
```

```
SD Card Size: 30436MB
SD Card Size: 30436MB
SD Card Size: 30436MB
SD Card Size: 30436MB
SD Card Size: 30436MB
SD Card Size: 30436MB
SD Card Size: 30436MB
SD Card Size: 30436MB
SD Card Size: 30436MB
SD Card Size: 30436MB
```

Insert the TF card into the computer, and you can find two more files: test.txt and foo.test. The content of the foo.txt is Hello World!, and the content of the test.txt is empty



04_qmi8658_output

【Demo description】

This demo demonstrates how to use ESP32-S3-LCD-2 to acquire QMI8658 data and print using serial port

【Hardware connection】

Connect the board to the computer

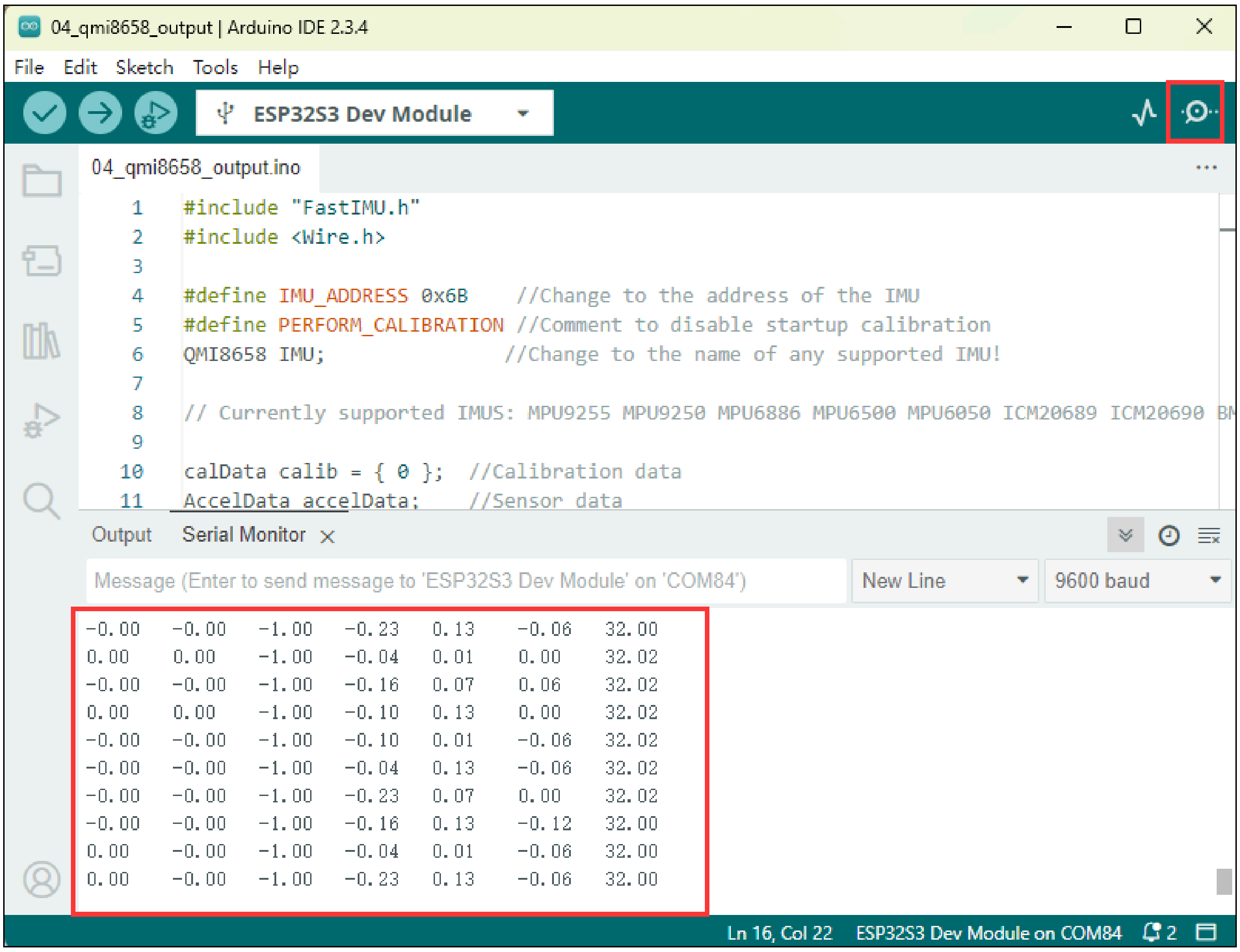
【Code analysis】

Initialize QMI8658

```
int err = IMU.init(calib, IMU_ADDRESS);
if (err != 0) {
  Serial.print("Error initializing IMU: ");
  Serial.println(err);
  while (true) {
    ;
  }
}
```

【Result demonstration】

Open the serial port monitor to see the printed x, y, z axis accel and gyro as well as the qmi8658 internal temperature sensor data


04_qmi8658_output | Arduino IDE 2.3.4
File Edit Sketch Tools Help
ESP32S3 Dev Module
04_qmi8658_output.ino
Output
Serial Monitor
New Line
9600 baud
Ln 16, Col 22
ESP32S3 Dev Module on COM84
2

```
#include "FastIMU.h"
#include <Wire.h>

#define IMU_ADDRESS 0x6B    //Change to the address of the IMU
#define PERFORM_CALIBRATION //Comment to disable startup calibration
QMI8658 IMU;               //Change to the name of any supported IMU!

// Currently supported IMUS: MPU9255 MPU9250 MPU6886 MPU6500 MPU6050 ICM20689 ICM20690 BM

calData calib = { 0 };  //Calibration data
AccelData accelData;    //Sensor data
```

Message (Enter to send message to 'ESP32S3 Dev Module' on 'COM84')

```
-0.00   -0.00   -1.00   -0.23   0.13    -0.06   32.00
0.00    0.00    -1.00   -0.04   0.01    0.00    32.02
-0.00   -0.00   -1.00   -0.16   0.07    0.06    32.02
0.00    0.00    -1.00   -0.10   0.13    0.00    32.02
-0.00   -0.00   -1.00   -0.10   0.01    -0.06   32.02
-0.00   -0.00   -1.00   -0.04   0.13    -0.06   32.02
-0.00   -0.00   -1.00   -0.23   0.07    0.00    32.02
-0.00   -0.00   -1.00   -0.16   0.13    -0.12   32.00
0.00    -0.00   -1.00   -0.04   0.01    -0.06   32.00
0.00    -0.00   -1.00   -0.23   0.13    -0.06   32.00
```

05_lvgl_qmi8658

【Demo description】

This demo demonstrates how to use ESP32-S3-LCD-2 to acquire QMI8658 data and display it using the lvgl library

【Hardware connection】

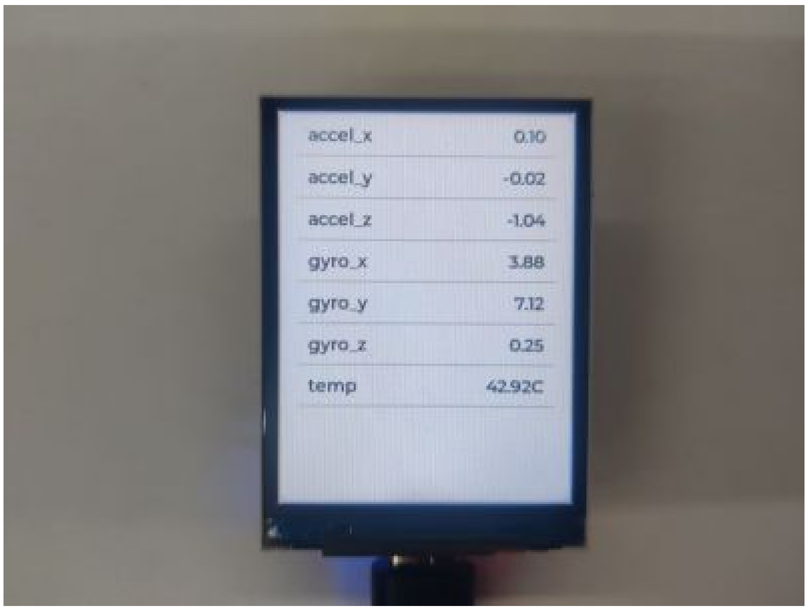
Connect the board to the computer

【Code analysis】

Initialize the UI and create a 100ms timer to obtain data from QMI8658

```
lvgl_qmi8658_ui_init(lv_scr_act());
```

【Result demonstration】



accel_x	0.10
accel_y	-0.02
accel_z	-1.04
gyro_x	3.88
gyro_y	7.12
gyro_z	0.25
temp	42.92C

06_lvgl_battery

【Demo description】

This demo demonstrates how to use ESP32-S3-LCD-2 to display battery voltage and ADC values on the screen using the lvgl library

【Hardware connection】

- Connect the board to the computer
- Insert the battery into the battery holder

【Code analysis】

Initialize the UI and create a 1000ms timer to obtain the data of the adc and convert the data into a voltage

```
lvgl_battery_ui_init(lv_scr_act());
```

【Result demonstration】

07_lvgl_brightness

【Demo description】

This demo demonstrates how to use ESP32-S3-LCD-2 to display the screen brightness on the screen using the lvgl library, and the screen brightness can be controlled through the slider

【Hardware connection】

Connect the board to the computer

【Code analysis】

Initialize the UI, and create a slider value change callback, when the value of the slider changes, modify the screen brightness

```
lvgl_brightness_ui_init(lv_scr_act());
```

【Result demonstration】

08_lvgl_example

【Demo description】

This demo demonstrates how to use ESP32-S3-LCD-2 to run the lvgl demos

【Hardware connection】

Connect the board to the computer

【Precautions】

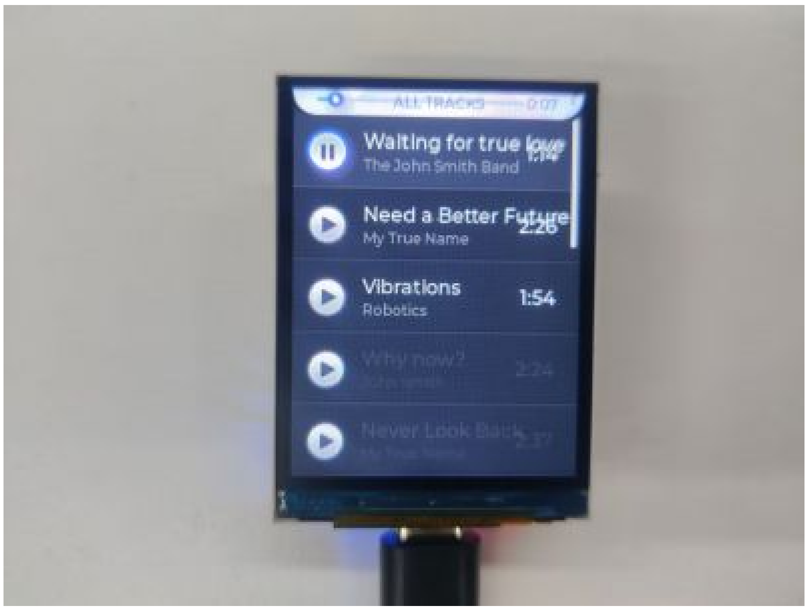
If the lvgl library is installed online, you need to copy the demos folder to src

【Code analysis】

Select the lvgl demos to run

```
lv_demo_widgets();
// lv_demo_benchmark();
// lv_demo_keypad_encoder();
// lv_demo_music();
// lv_demo_stress();
```

【Result demonstration】



09_lvgl_camera

【Demo description】

This demo uses the LVGL library to display images captured from the camera on the ESP32-S3-LCD-2 screen

【Hardware connection】

- Connect the board to the computer
- Insert the TF card into the card slot

【Code analysis】

Create a task specifically for acquiring camera images

```
xTaskCreatePinnedToCore(camera_task, "camera_task_task", 1024 * 3, NULL, 1, NULL, 0);
```

Obtain camera images and update display

```
    camera_fb_t *pic;
    lv_img_dsc_t img_dsc;
    img_dsc.header.always_zero = 0;
    img_dsc.header.w = 480;
    img_dsc.header.h = 320;
    img_dsc.data_size = 320 * 480 * 2;
    img_dsc.header.cf = LV_IMG_CF_TRUE_COLOR;
    img_dsc.data = NULL;
    // lv_img_set_src(img_camera, &pic);
    while (1)
    {
        pic = esp_camera_fb_get();

        if (NULL != pic)
        {
            img_dsc.data = pic->buf;
            if (lvgl_lock(-1))
            {
                lv_img_set_src(img_camera, &img_dsc);
                lvgl_unlock();
            }
        }
        esp_camera_fb_return(pic);
        vTaskDelay(pdMS_TO_TICKS(1));
    }
```

【Result demonstration】

10_camera_web

【Demo description】

This demo demonstrates how to use ESP32-S3-LCD-2 to drive the camera. After connecting to WiFi, the program will create a web server, and users can access it by entering the device's IP address in the browser. Web pages can display images from cameras and support settings such as resolution and mirroring

【Hardware connection】

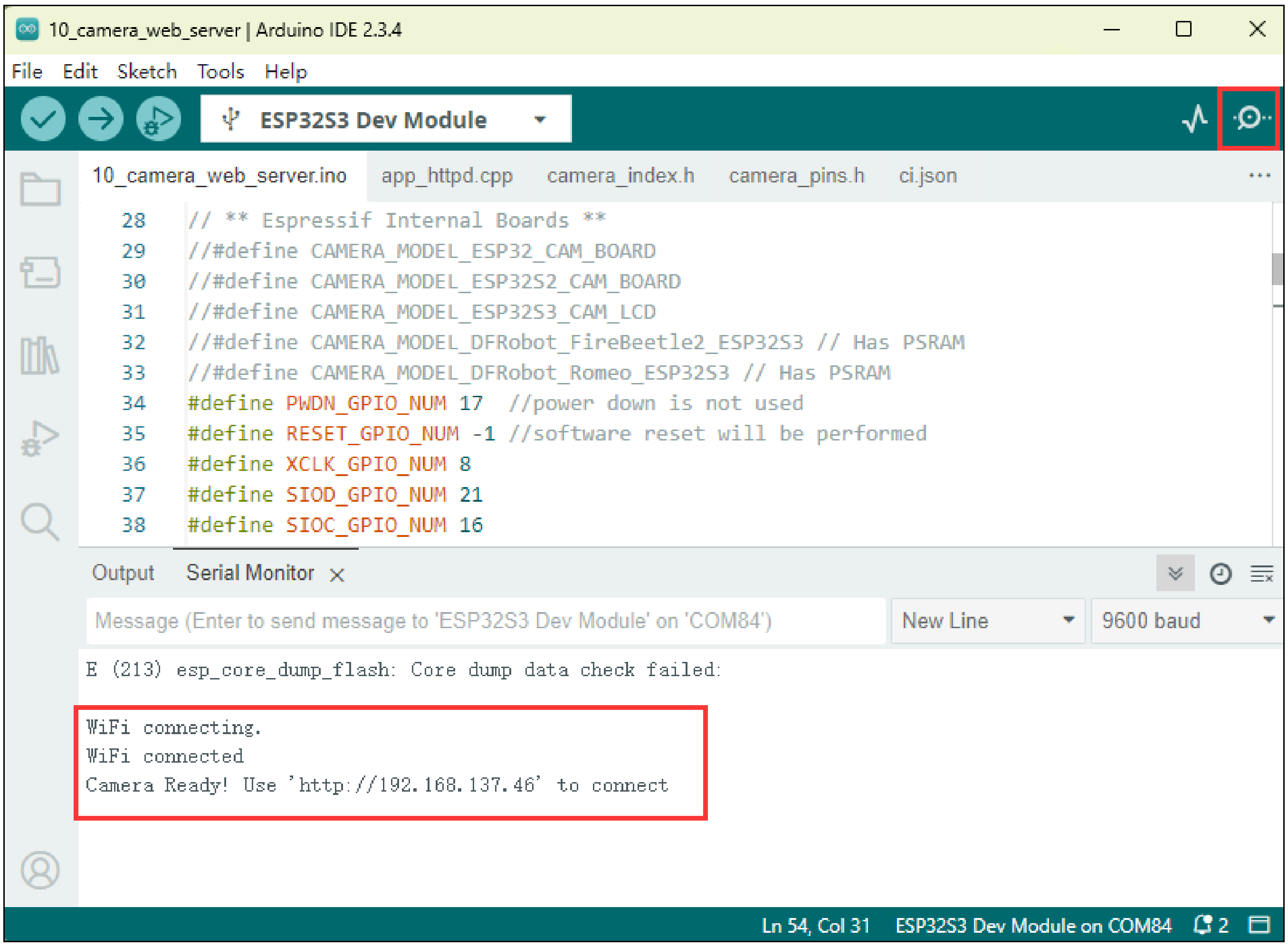
- Connect the board to the computer
- Plug the OV5640 camera into the 24Pin socket on the board

【Code analysis】

- **char sta_ssid[] = "waveshare";** : The name of the WiFi to be connected
- **char sta_pass[] = "12345678";** : The WiFi password to be connected

【Result demonstration】

Open the serial port terminal after connecting to WiFi to see the IP address



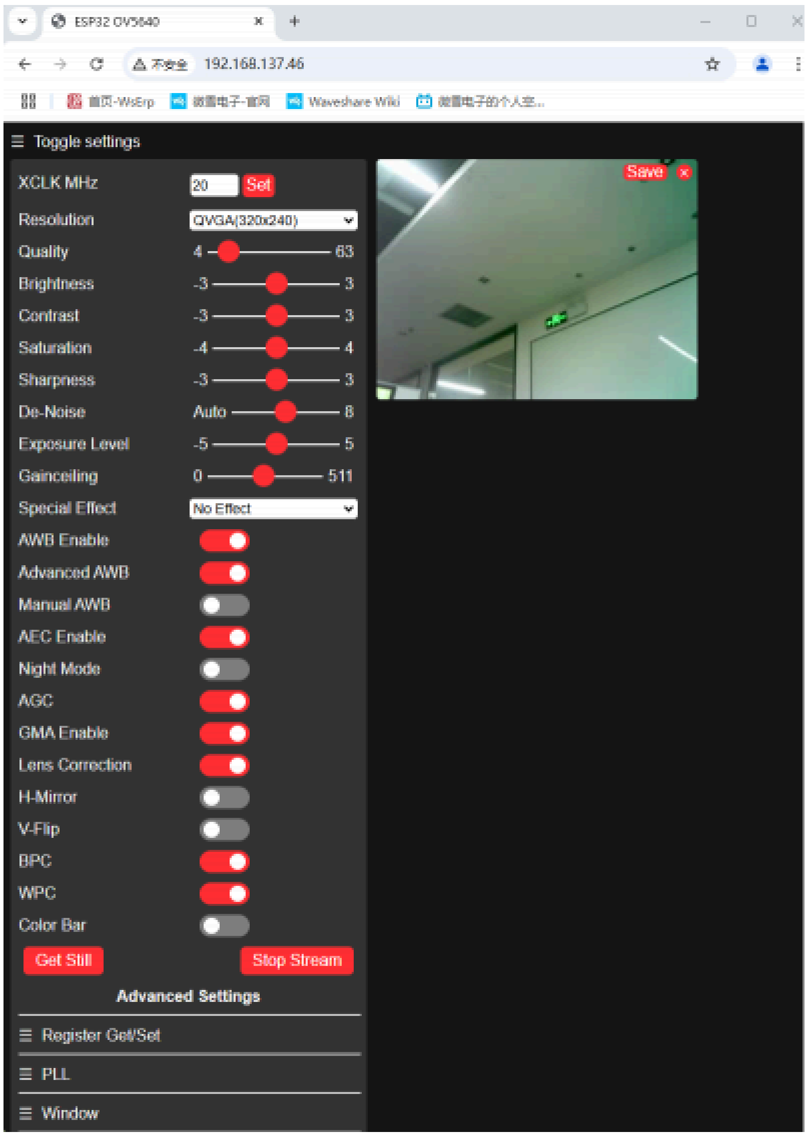
```
// ** Espressif Internal Boards **
//#define CAMERA_MODEL_ESP32_CAM_BOARD
//#define CAMERA_MODEL_ESP32S2_CAM_BOARD
//#define CAMERA_MODEL_ESP32S3_CAM_LCD
//#define CAMERA_MODEL_DFRobot_FireBeetle2_ESP32S3 // Has PSRAM
//#define CAMERA_MODEL_DFRobot_Romeo_ESP32S3 // Has PSRAM
#define PWDN_GPIO_NUM 17  //power down is not used
#define RESET_GPIO_NUM -1 //software reset will be performed
#define XCLK_GPIO_NUM 8
#define SIOD_GPIO_NUM 21
#define SIOC_GPIO_NUM 16
```

Message (Enter to send message to 'ESP32S3 Dev Module' on 'COM84')

```
E (213) esp_core_dump_flash: Core dump data check failed:

WiFi connecting.
WiFi connected
Camera Ready! Use 'http://192.168.137.46' to connect
```

- Use the browser to open the IP address for serial port printing
- Click Start Stream to see the image of the camera



Toggle settings

XCLK MHz	20 Set
Resolution	QVGA(320x240)
Quality	4 — 63
Brightness	-3 — 3
Contrast	-3 — 3
Saturation	-4 — 4
Sharpness	-3 — 3
De-Noise	Auto — 8
Exposure Level	-5 — 5
Gainceiling	0 — 511
Special Effect	No Effect
AWB Enable	✓
Advanced AWB	✓
Manual AWB	○
AEC Enable	✓
Night Mode	○
AGC	✓
GMA Enable	✓
Lens Correction	✓
H-Mirror	○
V-Flip	○
BPC	✓
WPC	✓
Color Bar	○

Advanced Settings

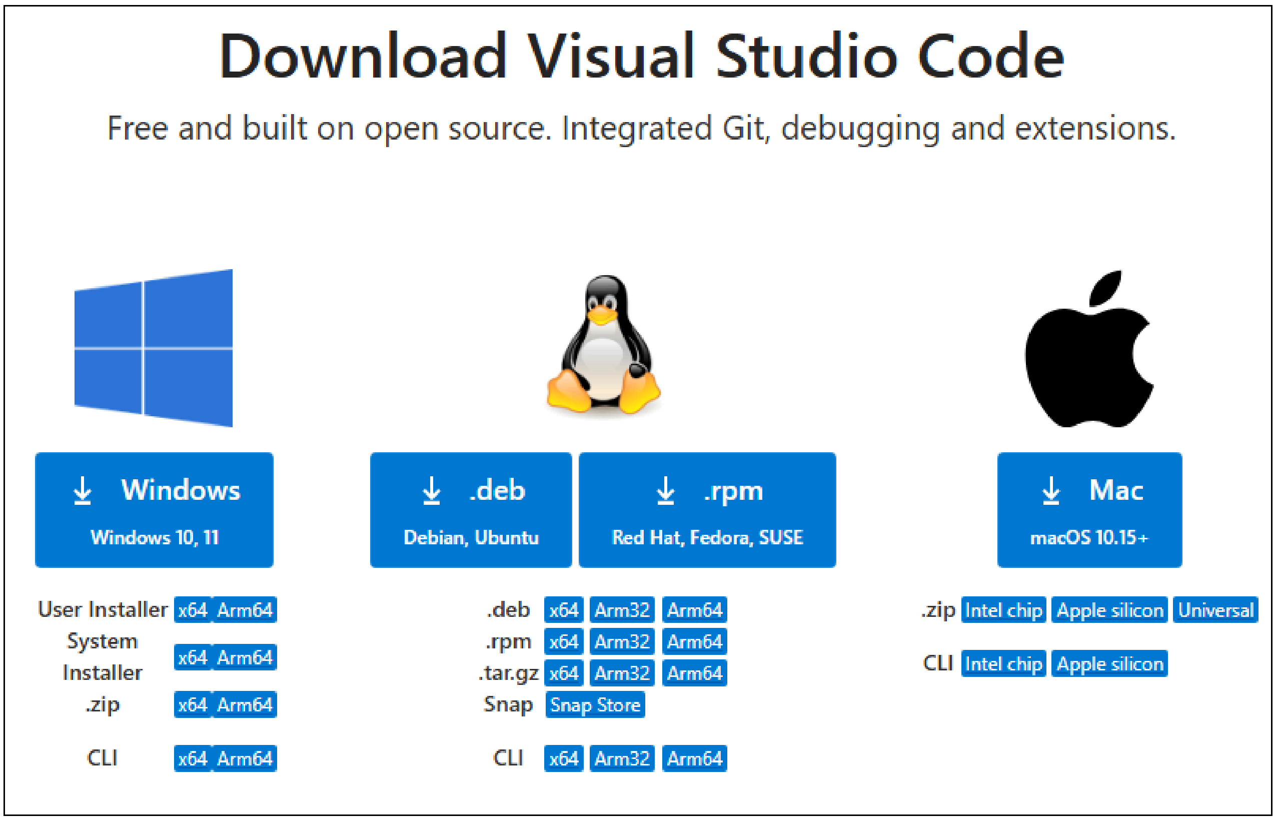
Working with ESP-IDF

This chapter introduces setting up the ESP-IDF environment setup, including the installation of Visual Studio and the Espressif IDF plugin, program compilation, downloading, and testing of demos, to assist users in mastering the development board and facilitating secondary development.

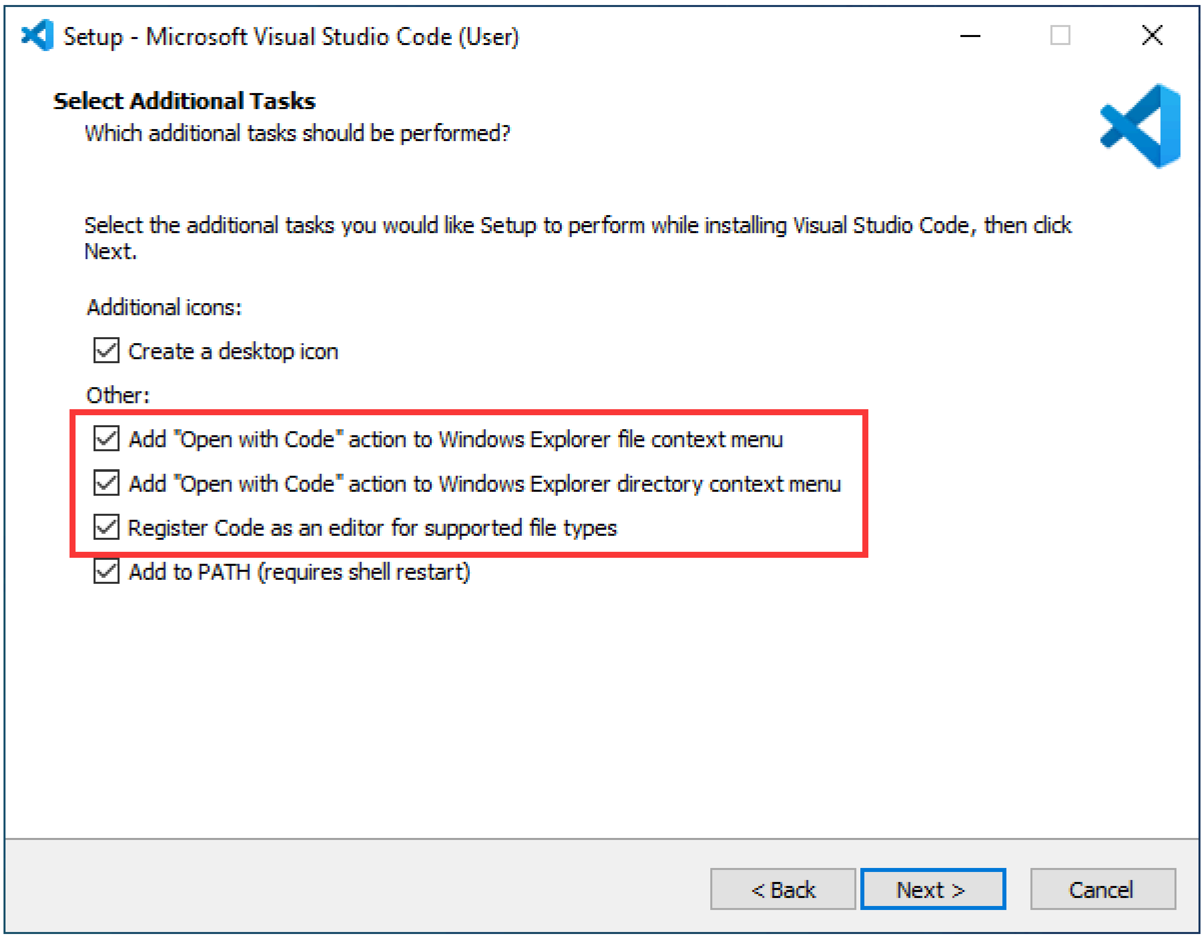
Environment Setup

Download and install Visual Studio

- Open the download page of VScode official website, choose the corresponding system and system bit to download



- After running the installation package, the rest can be installed by default, but here for the subsequent experience, it is recommended to check boxes 1, 2, and 3



Setup - Microsoft Visual Studio Code (User)

Select Additional Tasks
Which additional tasks should be performed?

Select the additional tasks you would like Setup to perform while installing Visual Studio Code, then click Next.

Additional icons:

- [✓] Create a desktop icon

Other:

- [✓] Add "Open with Code" action to Windows Explorer file context menu
- [✓] Add "Open with Code" action to Windows Explorer directory context menu
- [✓] Register Code as an editor for supported file types
- [✓] Add to PATH (requires shell restart)

  - After the first two items are enabled, you can open VSCode directly by right-clicking files or directories, which can improve the subsequent user experience
  - After the third item is enabled, you can select VSCode directly when you choose how to open it

The environment setup is carried out on the Windows 10 system, Linux and Mac users can access ESP-IDF environment setup for reference

Install Espressif IDF Plugin

- It is generally recommended to use **Install Online**. If online installation fails due to network factor, use **Install Offline**
- For more information about how to install the Espressif IDF plugin, see Install Espressif IDF Plugin

Run the First ESP-IDF Demo

If you are just getting started with ESP32 and ESP-IDF, and you don't know how to create, compile, flash, and run ESP-IDF ESP32 programs, then please expand and take a look. Hope it can help you!

Demo

Demo	Basic Description
sd_card_test	Test TF card
lvgl_example	Display lvgl demos
lvgl_qmi8658	Use lvgl library to display qmi8658 data
lvgl_camera	Use lvgl library to display camera images
lvgl_battery	Use lvgl library to display battery voltage
lvgl_brightness	Use lvgl library to control and display screen brightness

ESP32-S3-LCD-2 demos

sd_card_test

【Demo description】

This demo demonstrates how to use ESP32-S3-LCD-2 to test the read and write functions of the TF card.

【Hardware connection】

- Connect the board to the computer
- Insert the TF card into the card slot

【Code analysis】

SPI initialization

```
sdmmc_host_t host = SDSPI_HOST_DEFAULT();

spi_bus_config_t bus_cfg = {
    .mosi_io_num = PIN_NUM_MOSI,
    .miso_io_num = PIN_NUM_MISO,
    .sclk_io_num = PIN_NUM_CLK,
    .quadwp_io_num = -1,
    .quadhd_io_num = -1,
    .max_transfer_sz = 4000,
};
ret = spi_bus_initialize(host.slot, &bus_cfg, SDSPI_DEFAULT_DMA);
if (ret != ESP_OK) {
    ESP_LOGE(TAG, "Failed to initialize bus.");
    return;
}
```

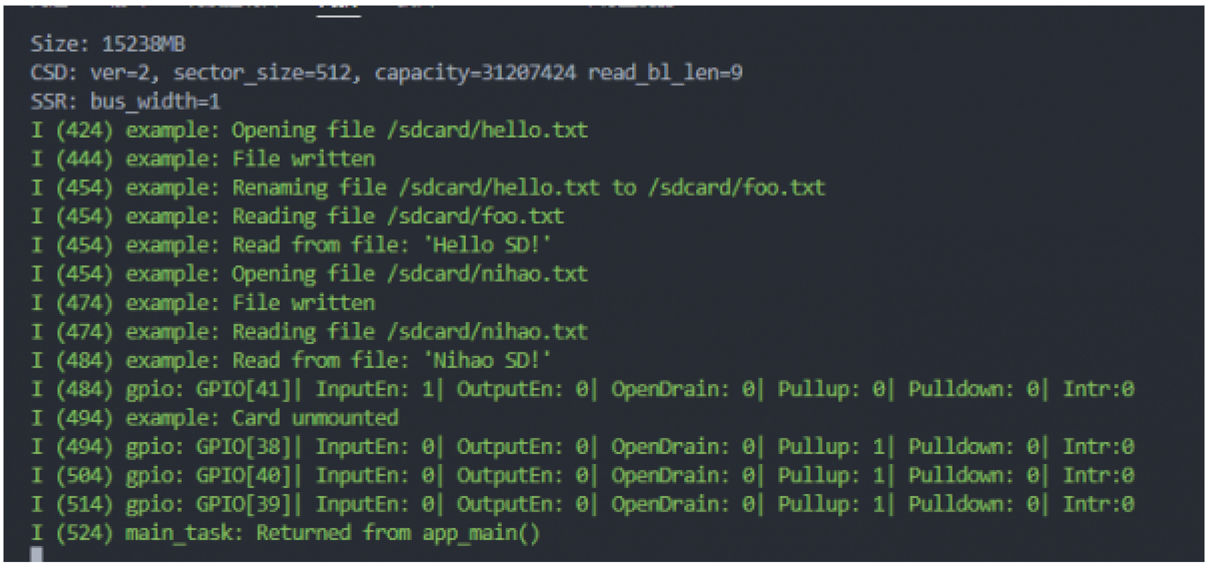
TF card initialization and mounting

```
    sdspi_device_config_t slot_config = SDSPI_DEVICE_CONFIG_DEFAULT();
    slot_config.gpio_cs = PIN_NUM_CS;
    slot_config.host_id = host.slot;

    ESP_LOGI(TAG, "Mounting filesystem");
    ret = esp_vfs_fat_sdspi_mount(mount_point, &host, &slot_config, &mount_config,
&card);
```

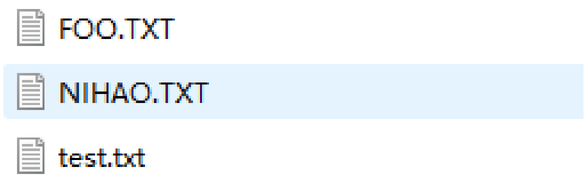
【Result demonstration】

Serial port monitor



```
Size: 15238MB
CSD: ver=2, sector_size=512, capacity=31207424 read_bl_len=9
SSR: bus_width=1
I (424) example: Opening file /sdcard/hello.txt
I (444) example: File written
I (454) example: Renaming file /sdcard/hello.txt to /sdcard/foo.txt
I (454) example: Reading file /sdcard/foo.txt
I (454) example: Read from file: 'Hello SD!'
I (454) example: Opening file /sdcard/nihao.txt
I (474) example: File written
I (474) example: Reading file /sdcard/nihao.txt
I (484) example: Read from file: 'Nihao SD!'
I (484) gpio: GPIO[41]| InputEn: 1| OutputEn: 0| OpenDrain: 0| Pullup: 0| Pulldown: 0| Intr:0
I (494) example: Card unmounted
I (494) gpio: GPIO[38]| InputEn: 0| OutputEn: 0| OpenDrain: 0| Pullup: 1| Pulldown: 0| Intr:0
I (504) gpio: GPIO[40]| InputEn: 0| OutputEn: 0| OpenDrain: 0| Pullup: 1| Pulldown: 0| Intr:0
I (514) gpio: GPIO[39]| InputEn: 0| OutputEn: 0| OpenDrain: 0| Pullup: 1| Pulldown: 0| Intr:0
I (524) main_task: Returned from app_main()
```

Insert the TF card into the computer, and you can find three more files: test.txt, FOO.TXT and NIHAO.TXT. Among them, the content of the FOO.TXT is Hello SD!, the content of the NIHAO.TXT is Hello SD!, and the content of the test.txt is empty



FOO.TXT
NIHAO.TXT
test.txt

【Precautions】

TF card needs to be formatted as FAT32

lvgl_example

【Demo description】

This demo demonstrates how to use ESP32-S3-LCD-2 to run the lvgl demos

【Hardware connection】

Connect the board to the computer

【Code analysis】

Initialization

```
lv_init();
display_init();
lv_port_disp_init();
lvgl_tick_timer_init(EXAMPLE_LVGL_TICK_PERIOD_MS);
bsp_brightness_init();
bsp_brightness_set_level(80);
```

Select the lvgl demos to run

```
lv_demo_widgets();
// lv_demo_benchmark();
// lv_demo_keypad_encoder();
// lv_demo_music();
// lv_demo_stress();
```

【Result demonstration】

lvgl_qmi8658

【Demo description】

This demo demonstrates how to use ESP32-S3-LCD-2 to acquire QMI8658 data and display it using the lvgl library

【Hardware connection】

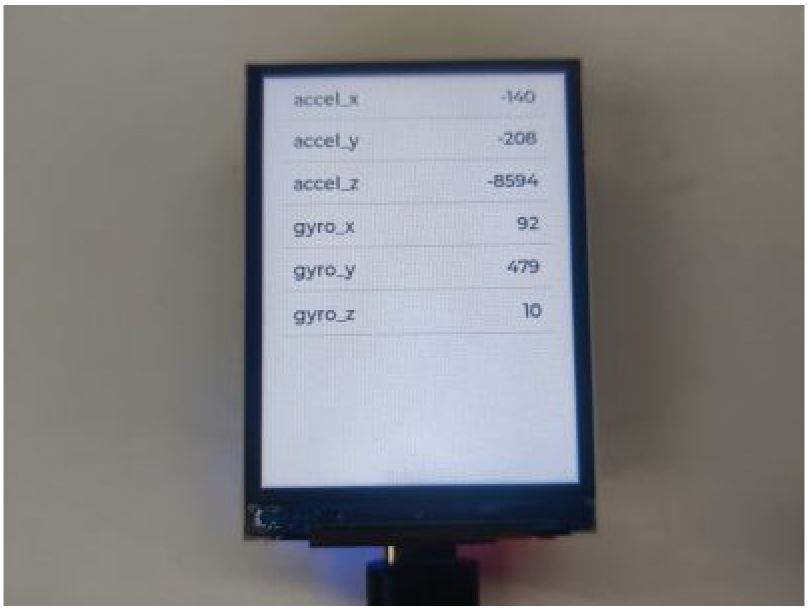
Connect the board to the computer

【Code analysis】

Initialize the UI and create a 100ms timer to obtain data from QMI8658

```
lvgl_qmi8658_ui_init(lv_scr_act());
```

【Result demonstration】



accel_x	-140
accel_y	-208
accel_z	-8594
gyro_x	92
gyro_y	479
gyro_z	10

lvgl_camera

【Demo description】

This demo uses the LVGL library to display images captured from the camera on the ESP32-S3-LCD-2 screen

【Hardware connection】

- Connect the board to the computer
- Insert the TF card into the card slot

【Code analysis】

Create a task specifically for acquiring camera images

```
xTaskCreatePinnedToCore(camera_task, "camera_task_task", 1024 * 3, NULL, 1, NULL, 0);
```

Obtain camera images and update display

```
    camera_fb_t *pic;
    lv_img_dsc_t img_dsc;
    img_dsc.header.always_zero = 0;
    img_dsc.header.w = 480;
    img_dsc.header.h = 320;
    img_dsc.data_size = 320 * 480 * 2;
    img_dsc.header.cf = LV_IMG_CF_TRUE_COLOR;
    img_dsc.data = NULL;
    // lv_img_set_src(img_camera, &pic);
    while (1)
    {
        pic = esp_camera_fb_get();

        if (NULL != pic)
        {
            img_dsc.data = pic->buf;
            if (lvgl_lock(-1))
            {
                lv_img_set_src(img_camera, &img_dsc);
                lvgl_unlock();
            }
        }
        esp_camera_fb_return(pic);
        vTaskDelay(pdMS_TO_TICKS(1));
    }
```

【Result demonstration】

lvgl_battery

【Demo description】

This demo demonstrates how to use ESP32-S3-LCD-2 to display battery voltage and ADC values on the screen using the lvgl library

【Hardware connection】

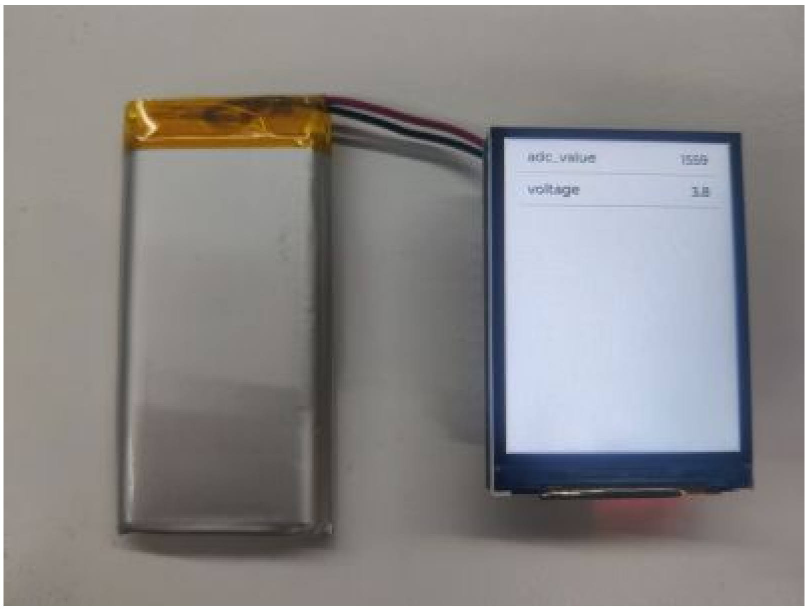
- Connect the board to the computer
- Insert the battery into the battery holder

【Code analysis】

Initialize the UI and create a 1000ms timer to obtain the data of the adc and convert the data into a voltage

```
lvgl_battery_ui_init(lv_scr_act());
```

【Result demonstration】



lvgl_brightness

【Demo description】

This demo demonstrates how to use ESP32-S3-LCD-2 to display the screen brightness on the screen using the lvgl library, and the screen brightness can be controlled through the slider

【Hardware connection】

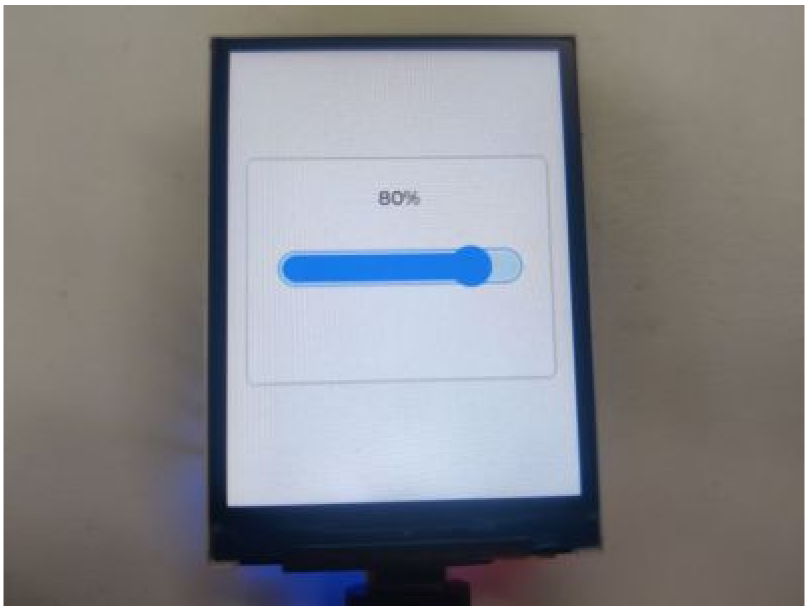
Connect the board to the computer

【Code analysis】

Initialize the UI, and create a slider value change callback, when the value of the slider changes, modify the screen brightness

```
lvgl_brightness_ui_init(lv_scr_act());
```

【Result demonstration】



Flash Firmware Flashing and Erasing

- **The current demo provides test firmware, which can be used to test whether the onboard device functions properly by directly flashing the test firmware**
- **bin file path:**

```
..\ESP32-S3-LCD-2-Demo\Firmware
```

Flash firmware flashing and erasing for reference

Resources

Schematic Diagram

ESP32-S3-LCD-2 Schematic diagram

Demo

ESP32-S3-LCD-2 Demo

Project Document

ESP32-S3-LCD-2 2D/3D file

Datasheets

ESP32-S3

- ESP32-S3 Technical Reference Manual
- ESP32-S3 Series Datasheet

Software Tools

Arduino

- Arduino IDE Official download link
- ESP32-Arduino official documentation
- Arduino-ESP32 offline component package

VScode

VScode official website

Firmware Flashing Tool

Flash_download_tool

Other Resource Links

- ESP32-Arduino official documentation
- LVGL official documentation

FAQ

Answer:

Refer to Arduino Project Parameter Setting.

Answer:

Long press the BOOT button, plug in the USB at the same time, then release the BOOT button, at this time the module can enter the download mode, which can solve most of the problems that can not be downloaded.

Answer:

It may be due to Flash blank and the USB port is not stable, you can long-press the BOOT button, press RESET at the same time, and then release RESET, and then release the BOOT button, at this time the module can enter the download mode to flash the firmware (demo) to solve the situation.

Answer:

It's normal for the first compilation to be slow, just be patient

Answer:

If there is a reset button on the development board, press the reset button; if there is no reset button, please power it on again

Answer:

- Some AppData folders are hidden by default and can be set to show.
- English system: Explorer->View->Check "Hidden items"
- Chinese system: File Explorer -> View -> Display -> Check "Hidden Items"

Answer:

Windows system:

①View through Device Manager: Press the Windows + R keys to open the "Run" dialog box; input devmgmt.msc and press Enter to open the Device Manager; expand the "Ports (COM and LPT)" section, where all COM ports and their current statuses will be listed.
②Use the command prompt to view: Open the Command Prompt (CMD), enter the "mode" command, which will display status information for all COM ports.
③Check hardware connections: If you have already connected external devices to the COM port, the device usually occupies a port number, which can be determined by checking the connected hardware.

Linux system:

①Use the dmesg command to view: Open the terminal.
①Use the ls command to view: Enter ls /dev/ttyS* or ls /dev/ttyUSB* to list all serial port devices.
③Use the setserial command to view: Enter setserial -g /dev/ttyS* to view the configuration information of all serial port devices.

Answer:

Install MAC Driver and flash again.

Answer:

Please refer to SquareLine Studio tutorial

Support

Technical Support

If you need technical support or have any feedback/review, please click the **Submit Now** button to submit a ticket, Our support team will check and reply to you within 1 to 2 working days. Please be patient as we make every effort to help you to resolve the issue.
Working Time: 9 AM - 6 PM GMT+8 (Monday to Friday)

Submit Now