Important Information

Thank you for choosing Freenove products!

Getting Started

If you have not yet downloaded the zip file, associated with this kit, please do so now and unzip it.

Get Support and Offer Input

Freenove provides free and responsive product and technical support, including but not limited to:

- Product quality issues
- Product use and build issues
- Questions regarding the technology employed in our products for learning and education
- Your input and opinions are always welcome
- We also encourage your ideas and suggestions for new products and product improvements

For any of the above, you may send us an email to:

support@freenove.com

Safety and Precautions

Please follow the following safety precautions when using or storing this product:

- Keep this product out of the reach of children under 6 years old.
- This product should be **used only when there is adult supervision present** as young children lack necessary judgment regarding safety and the consequences of product misuse.
- This product contains small parts and parts, which are sharp. This product contains electrically conductive parts. **Use caution with electrically conductive parts near or around power supplies, batteries and powered (live) circuits.**
- When the product is turned ON, activated or tested, some parts will move or rotate. **To avoid injuries to hands and fingers keep them away from any moving parts!**
- It is possible that an improperly connected or shorted circuit may cause overheating. **Should this happen, immediately disconnect the power supply or remove the batteries and do not touch anything until it cools down!** When everything is safe and cool, review the product tutorial to identify the cause.
- Only operate the product in accordance with the instructions and guidelines of this tutorial, otherwise parts may be damaged or you could be injured.
- Store the product in a cool dry place and avoid exposing the product to direct sunlight.
- After use, always turn the power OFF and remove or unplug the batteries before storing.

About Freenove

Freenove provides open source electronic products and services worldwide.

Freenove is committed to assist customers in their education of robotics, programming and electronic circuits so that they may transform their creative ideas into prototypes and new and innovative products. To this end, our services include but are not limited to:

- Educational and Entertaining Project Kits for Robots, Smart Cars and Drones
- Educational Kits to Learn Robotic Software Systems for Arduino, Raspberry Pi and micro: bit
- Electronic Component Assortments, Electronic Modules and Specialized Tools
- **Product Development and Customization Services**

You can find more about Freenove and get our latest news and updates through our website:

http://www.freenove.com

sale@freenove.com

Copyright

FREENOVE™

Contents

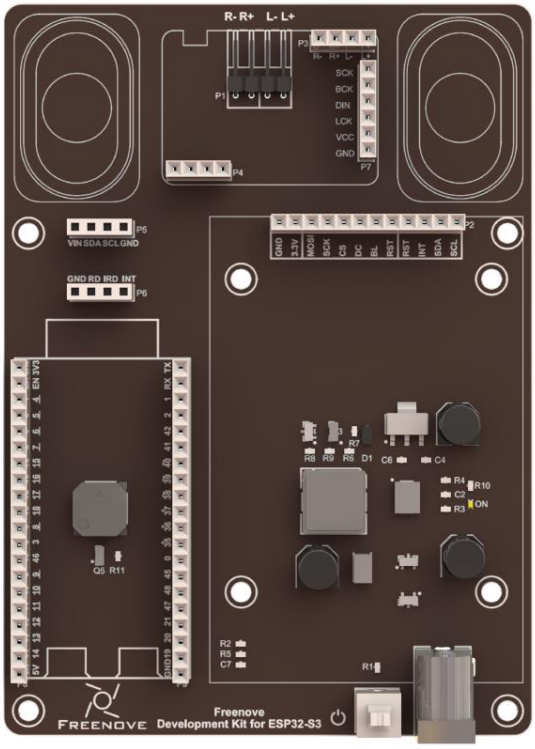
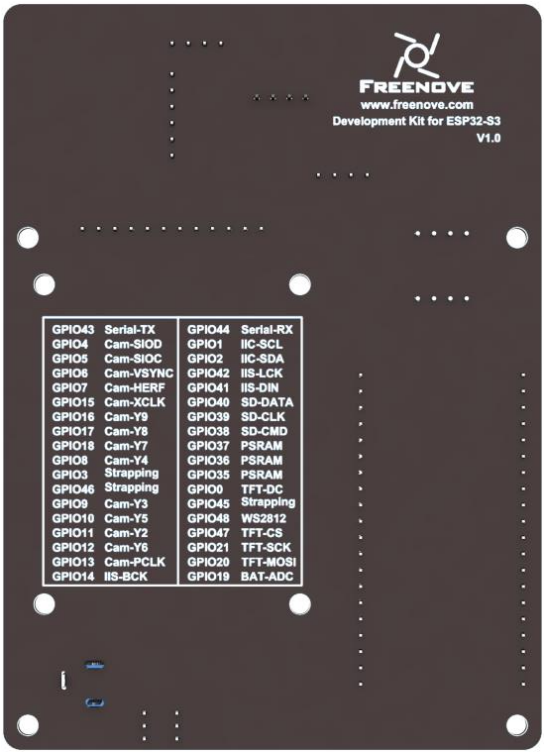
Important Information 1
Contents 1
List 4
ESP32-S3 WROOM Shield 4
Machinery Parts 4
Electronic Parts 5
Tools 5
Preface 6
ESP32-S3 WROOM 6
Notes for GPIO 8
CH343 (Importance) 10
Programming Software 18
Environment Configuration 21
Library Installation 25
Chapter 0 Assembly 29
Installation of Brass Standoffs 29
Installation of Screen 29
Installation of Speakers 31
Installation of the Audio Module 31
Connection of Camera 32
Connection of ESP32-S3 WROOM 33
Connection of MAX30102 34
Chapter1 ADC Test 35
Project 1.1 Read the Voltage of Power 35
Chapter 2 WS2812 40
Project 2.1 WS2812 40
Chapter 3 Camera Web Server 44
Project 3.1 Camera Web Server 44
Chapter 4 Read and Write the SDcard 53
Project 4.1 SDMMC Test 53
Chapter 5 Play SD card music 64

Project 5.1 SDMMC Music 64
Chapter 6 MAX30102 72
Project 6.1 MAX30102 72
Chapter 7 Drving Freenove 2.8-Inch Screen 79
Project 7.1 Screen Display Function 79
Chapter 8 Touch of Freenove 2.8-Inch Screen 88
Project 8.1 Touch Function of the Screen 88
Chapter 9 LVGL 93
Project 9.1 LVGL Test 93
Chapter 10 LVGL Lable 101
Project 10.1 LVGL Lable 101
Chapter 11 LVGL Button 107
Project 11.1 LVGL Button 107
Chapter 12 LVGL Slider 115
Project 12.1 LVGL Slider 115
Chapter 13 LVGL Img 124
Project 13.1 LVGL Img 124
Chapter 14 LVGL Imgbtn 134
Project 14.1 LVGL Imgbtn 134
Chapter 15 LVGL Camera 140
Project 15.1 LVGL Camera 140
Chapter 16 LVGL Picture 151
Project 16.1 LVGL Picture 151
Chapter 17 LVGL Music 160
Project 17.1 LVGL Music 160
Chapter 18 LVGL Heartrate 177
Project 18.1 LVGL Heartrate 177
Chapter 19 LVGL Multifunctionality 188
Project 19.1 LVGL Multifunctionality 188

What's next? 199
End of the Tutorial 199

List

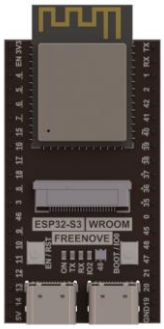
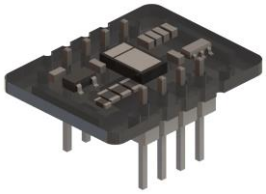
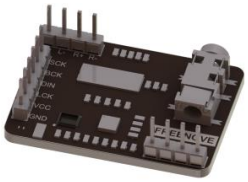
ESP32-S3 WROOM Shield

Top	Bottom
	

Machinery Parts

M3*8+3 Brass Standoff x5	M3*5 Brass Standoff x5	M3*4 Screw x9
		

Electronic Parts

Screen 	ESP32-S3 WROOM 	Camera 
MAX30102 	Audio Converter& Amplifier 	Speaker*2 
Card Reader 	SD Card 	9V Battery Cable 
FPC camera cable 		Extension board for camera 

Tools

Type-C 	3mm Cross Screwdriver 

Preface

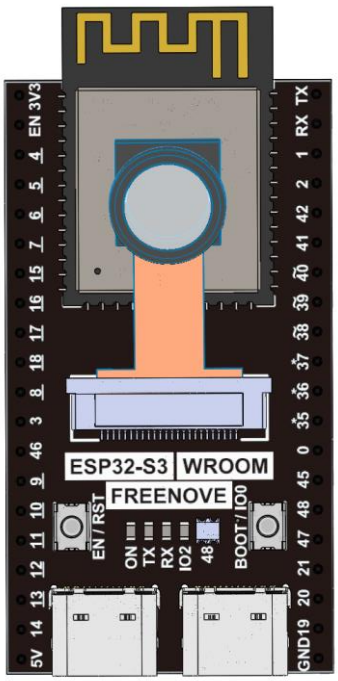
ESP32-S3 WROOM

ESP32-S3-WROOM-1 has launched a total of two antenna packages, PCB on-board antenna and IPEX antenna respectively. The PCB on-board antenna is an integrated antenna in the chip module itself, so it is convenient to carry and design. The IPEX antenna is a metal antenna derived from the integrated antenna of the chip module itself, which is used to enhance the signal of the module.

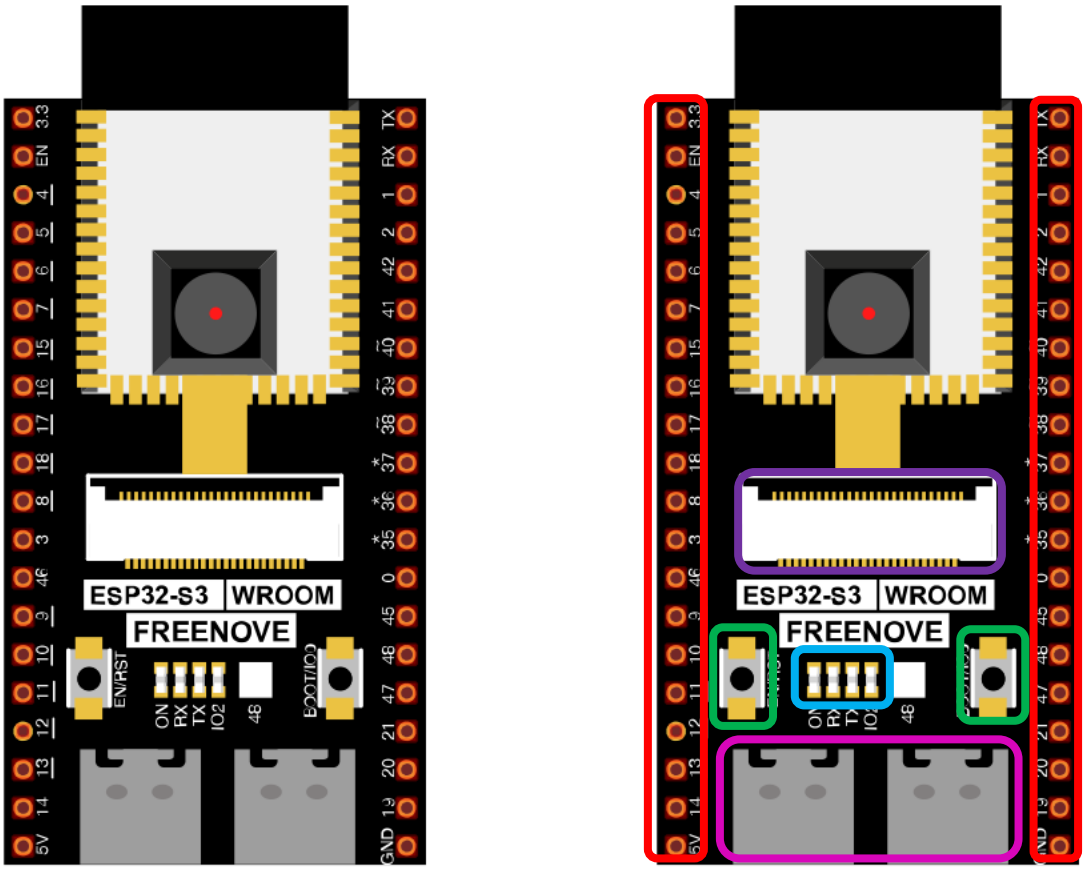
PCB on-board antenna	IPEX antenna
	

In this tutorial, the ESP32-S3 WROOM is designed based on the PCB on-board antenna-packaged ESP32-S3-WROOM-1 module.

ESP32-S3 WROOM



The hardware interfaces of ESP32-S3 WROOM are distributed as follows:



Compare the left and right images. We've boxed off the resources on the ESP32-S3 WROOM in different colors to facilitate your understanding of the ESP32-S3 WROOM.

Box color	Corresponding resources introduction
	GPIO pins
	LED indicators
	Camera interface
	Reset button, Boot mode selection button
	USB ports

For more information, please visit: https://www.espressif.com.cn/sites/default/files/documentation/esp32-s3-wroom-1_wroom-1u_datasheet_en.pdf.

GPIO pins of ESP32-S3 WROOM can be used to interface with external devices and control peripheral circuits.
In the following projects, we only use USB cable to power ESP32-S3 WROOM by default. But you can still power the ESP3-S3 WROOM Shield with external power supply, with a voltage range of 6-12V and current of over 0.5A.

Notes for GPIO

PSRAM Pin

The module on the ESP32-S3-WROOM board utilizes the ESP32-S3R8 chip, which comes with 8MB of external Flash. When using the OPI PSRAM, it should be noted that GPIO35-GPIO37 on the ESP32-S3-WROOM board will not be available for other purposes. However, when OPI PSRAM is not used, GPIO35-GPIO37 on the board can be used as normal GPIO.

ESP32-S3R8 / ESP32-S3R8V	In-package PSRAM (8 MB, Octal SPI)
SPICLK	CLK
SPICS1	CE#
SPID	DQ0
SPIQ	DQ1
SPIWP	DQ2
SPIHD	DQ3
GPIO33	DQ4
GPIO34	DQ5
GPIO35	DQ6
GPIO36	DQ7
GPIO37	DQS/DM

SDcard Pin

An SDcard slot is integrated on the back of the ESP32-S3-WROOM board, and we can use GPIO38-GPIO40 of ESP32-S3-WROOM to drive SD card.

The SDcard of ESP32-S3-WROOM uses SDMMC, a 1-bit bus driving method, which is integrated in the Arduino IDE, and we can call the "SD_MMC.h" library to drive it. For more details, please refer to the SDcard chapter in this tutorial.

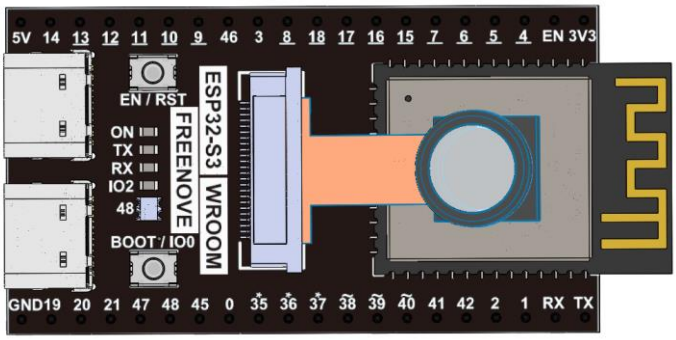
USB Pin

In Micropython, GPIO19 and GPIO20 are dedicated to the USB function of the ESP32S3 and cannot be used for other functions.

Please note that in this product, GPIO19 is used to read the ADC value of the external power supply, and therefore should not be used for USB functions to avoid conflicts.

Cam Pin

When using the camera of our ESP32-S3 WROOM, please check the pins of it. Pins with underlined numbers are used by the camera function, if you want to use other functions besides it, please avoid using them.



CAM_Pin	GPIO_pin
SIOD	GPIO4
SIOC	GPIO5
CSI_VYSNC	GPIO6
CSI_HREF	GPIO7
CSI_Y9	GPIO16
XCLK	GPIO15
CSI_Y8	GPIO17
CSI_Y7	GPIO18
CSI_PCLK	GPIO13
CSI_Y6	GPIO12
CSI_Y2	GPIO11
CSI_Y5	GPIO10
CSI_Y3	GPIO9
CSI_Y4	GPIO8

If you have any questions regarding GPIO information, you can click here to navigate back to the ESP32-S3 WROOM and view specific GPIO details.
Or check: https://www.espressif.com/sites/default/files/documentation/esp32-s3_datasheet_en.pdf.

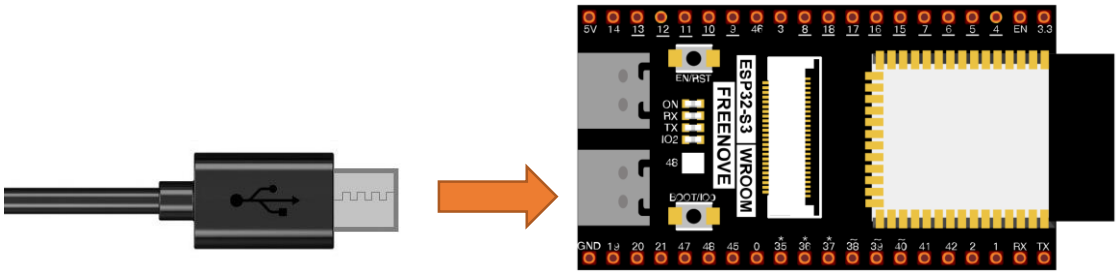
CH343 (Importance)

ESP32-S3 WROOM uses CH343 to download code. Therefore, before using the device, it is necessary to install the CH343 driver on your computer.

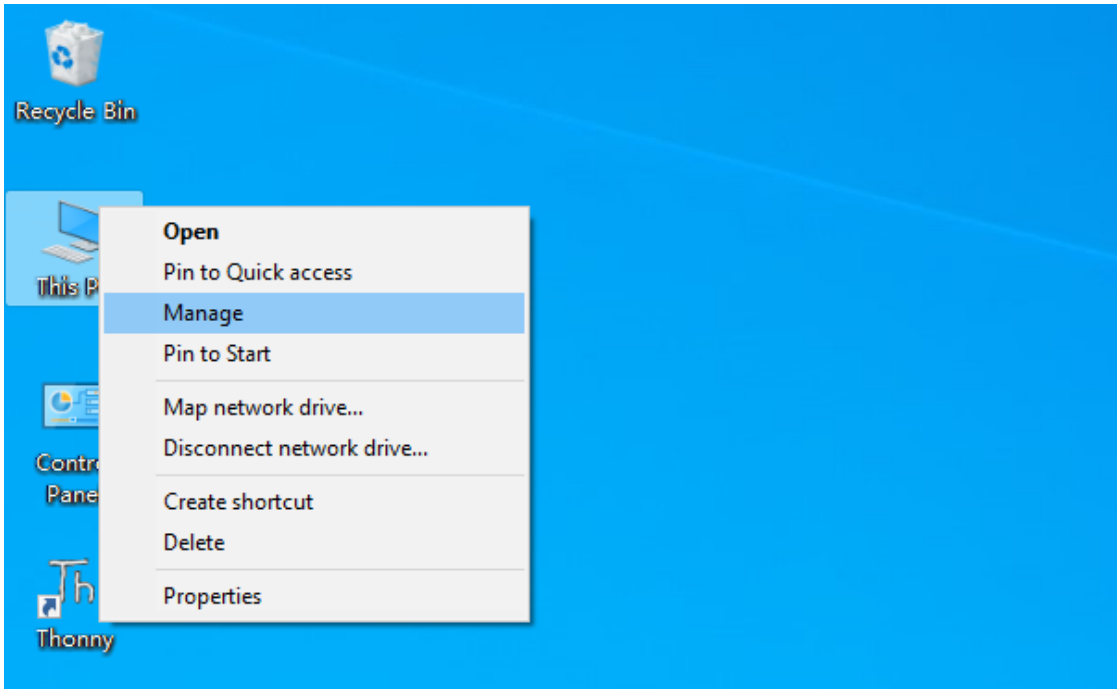
Windows

Check whether CH343 has been installed

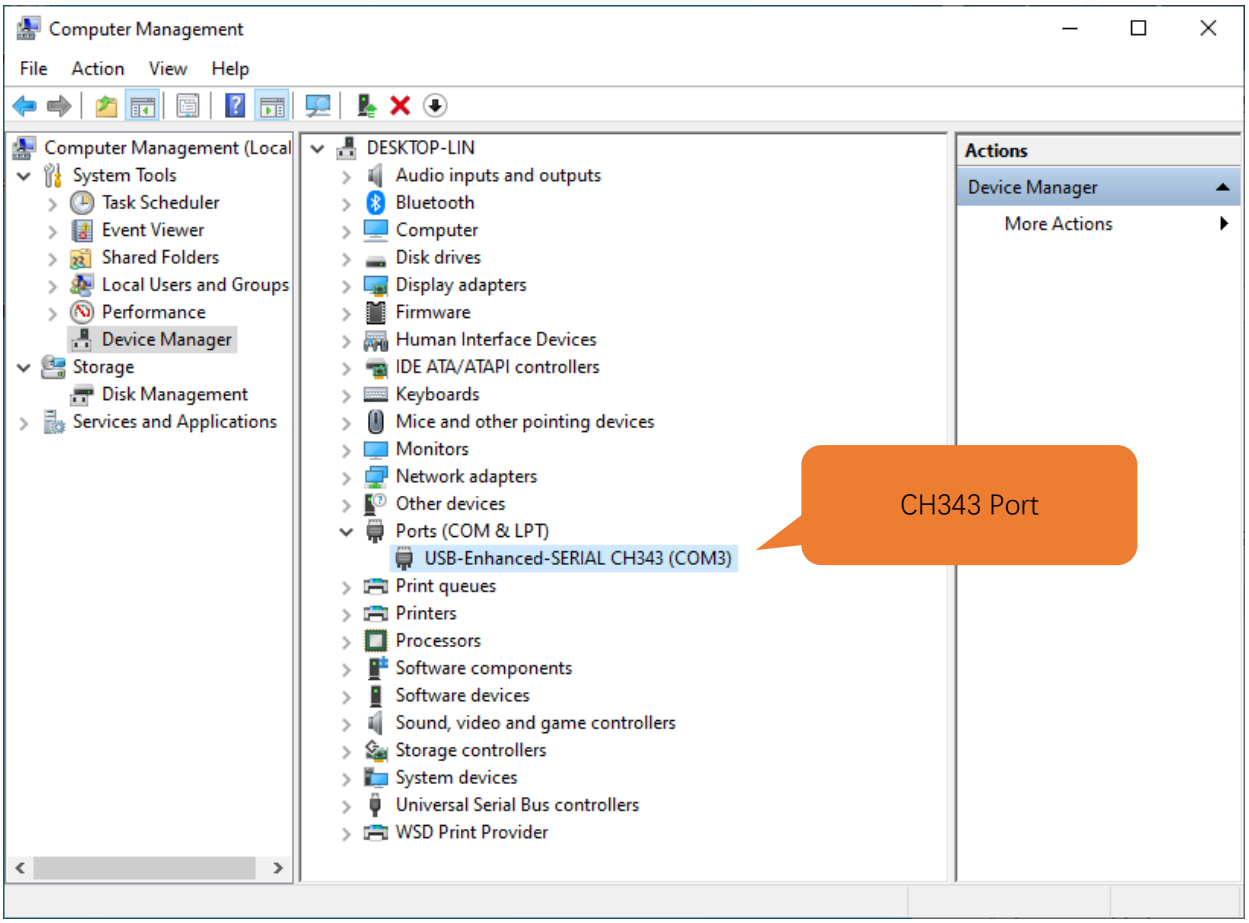
1. Connect your computer and ESP32-S3 WROOM with a USB cable.



2. Turn to the main interface of your computer, select “This PC” and right-click to select “Manage”.

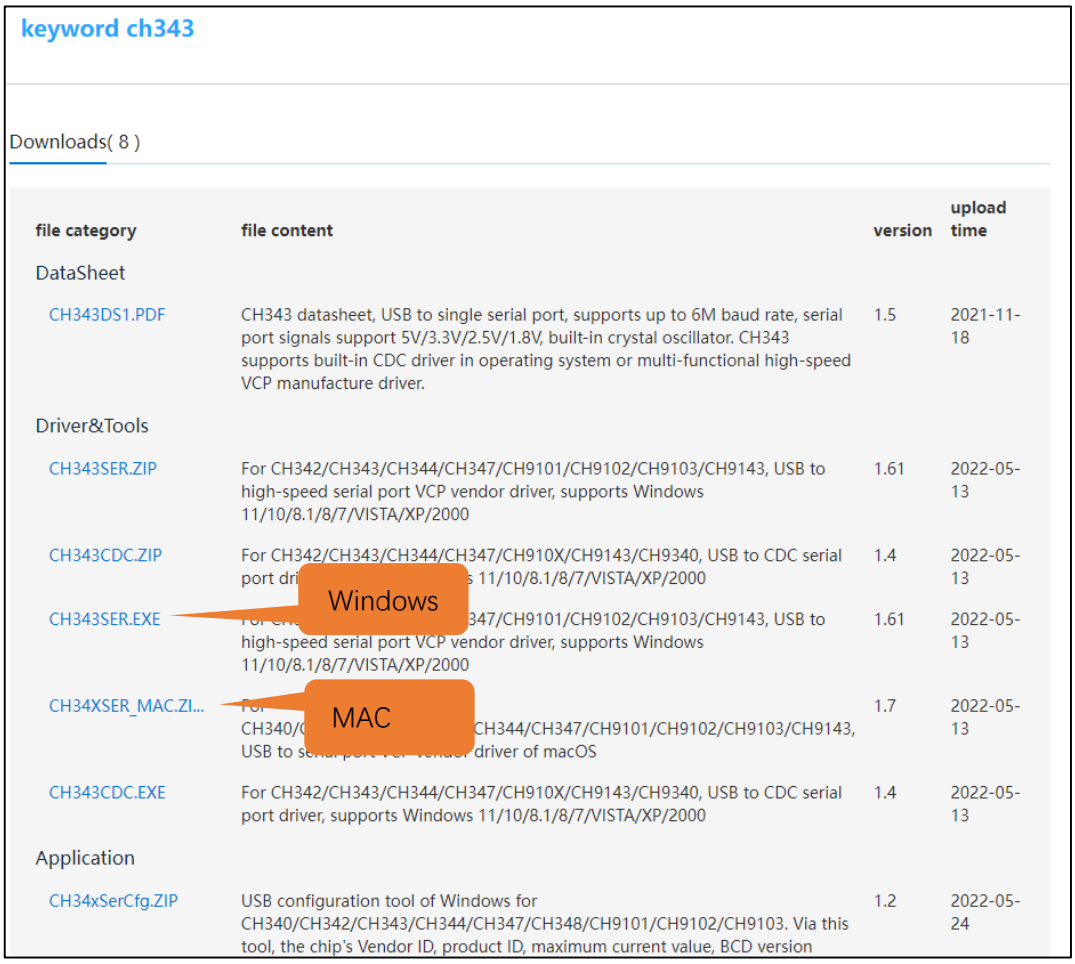


3. Click "Device Manager". If your computer has installed CH343, you can see"USB-Enhances-SERIAL CH343 (COMx)". And you can click here to move to the next step.

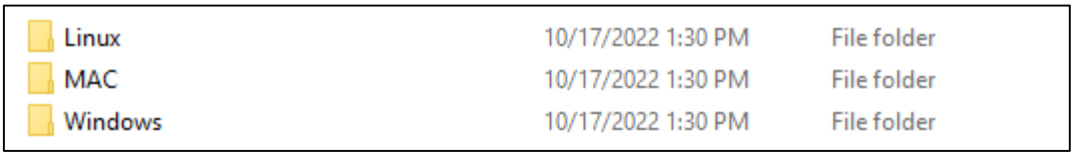


Installing CH343

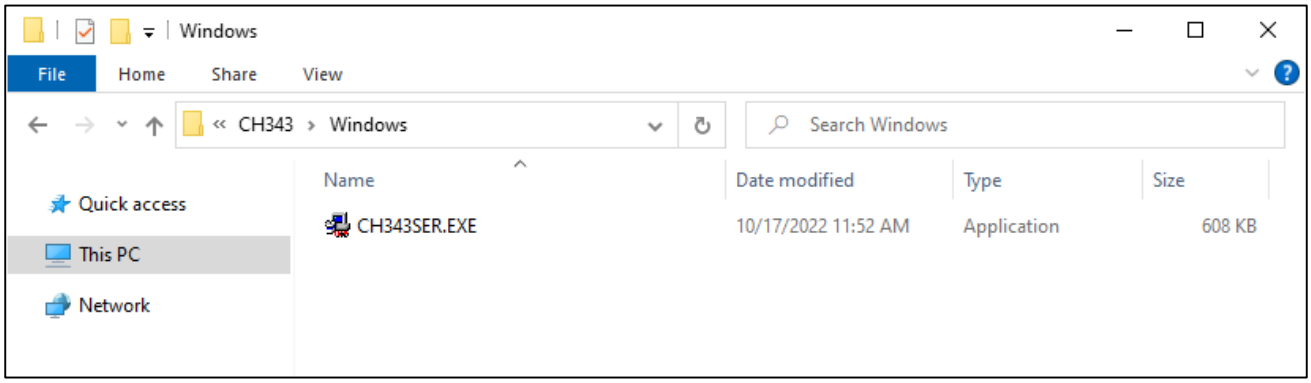
1. First, download CH343 driver, click http://www.wch-ic.com/search?t=all&q=ch343 to download the appropriate one based on your operating system.



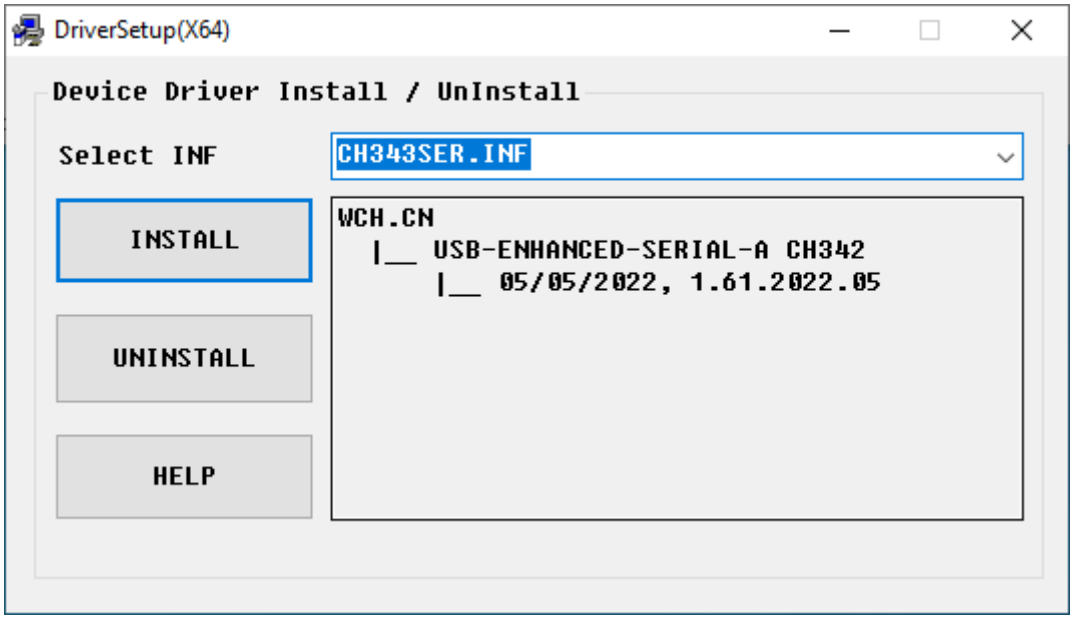
If you would not like to download the installation package, you can open **"Freenove_Ultimate_Starter_Kit_for_ESP32_S3/CH343"**, we have prepared the installation package.



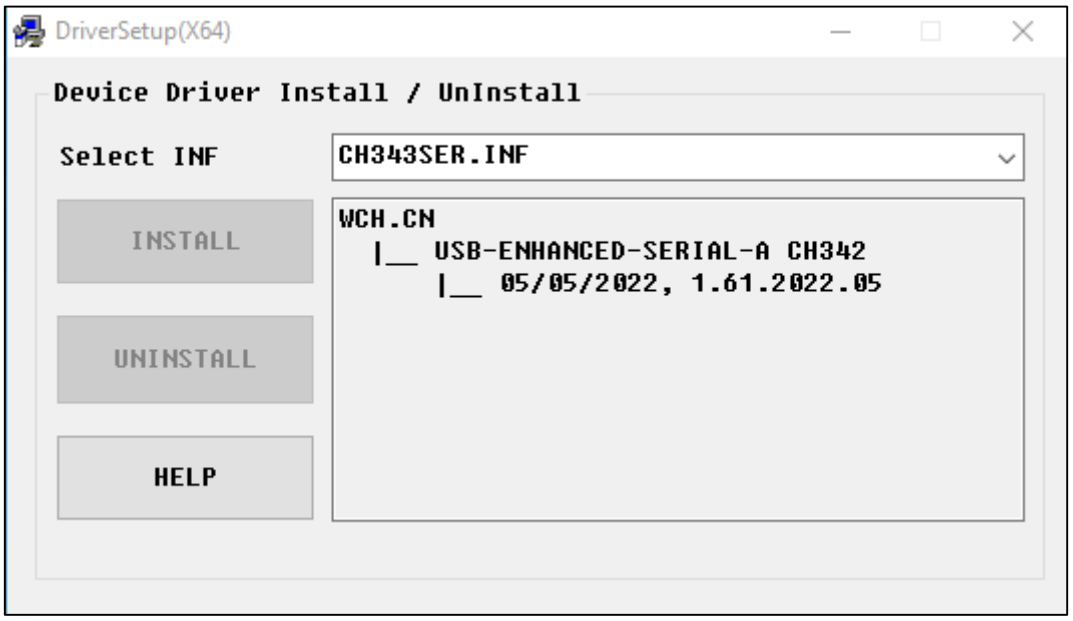
2. Open the folder "**Freenove_Ultimate_Starter_Kit_for_ESP32_S3/CH343/Windows/**"



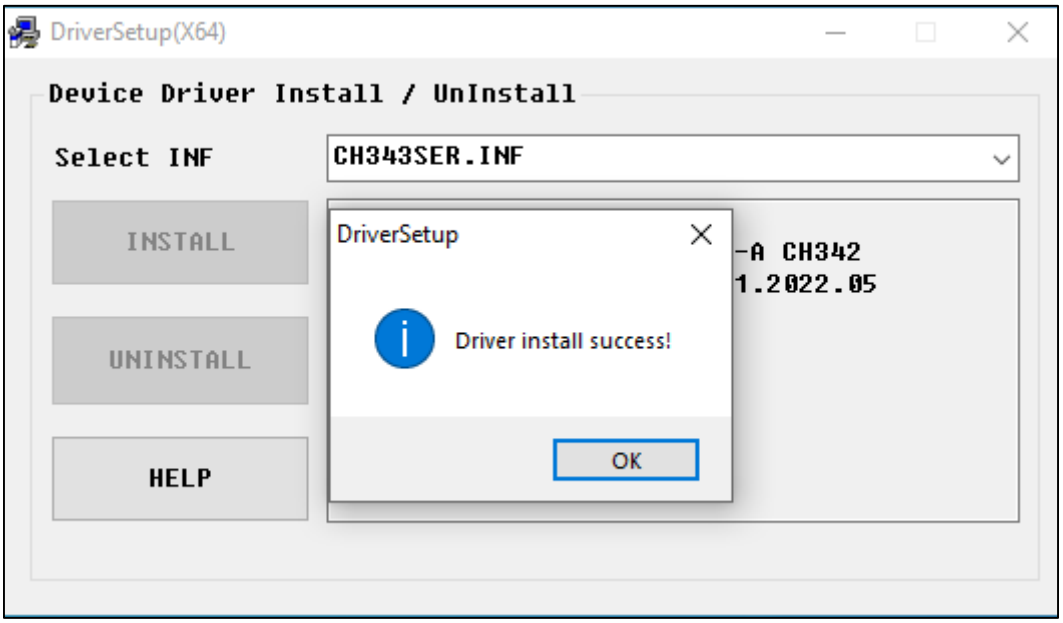
3. Double click "**CH343SER.EXE**".



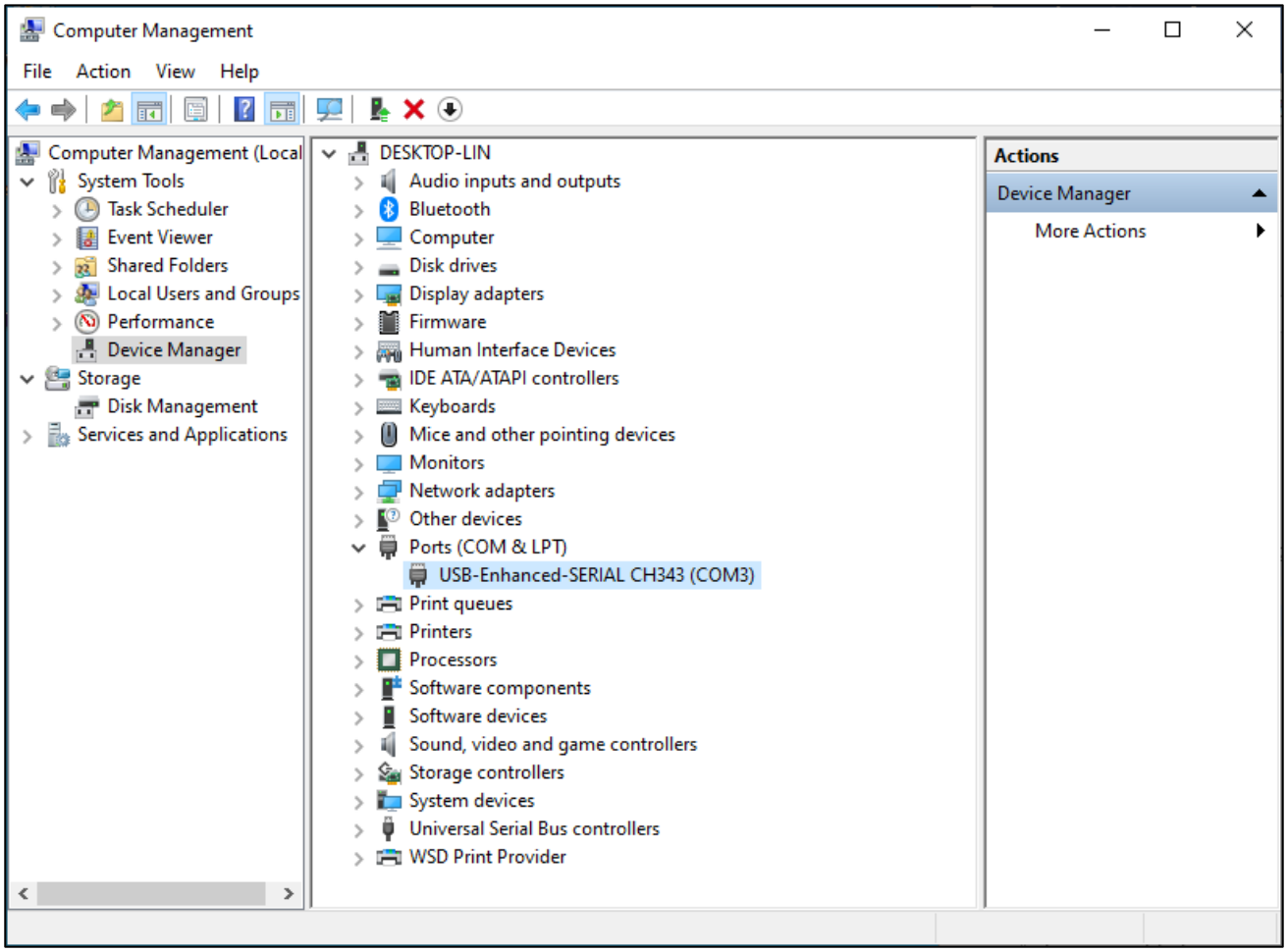
4. Click "INSTALL" and wait for the installation to complete.



5. Install successfully. Close all interfaces.



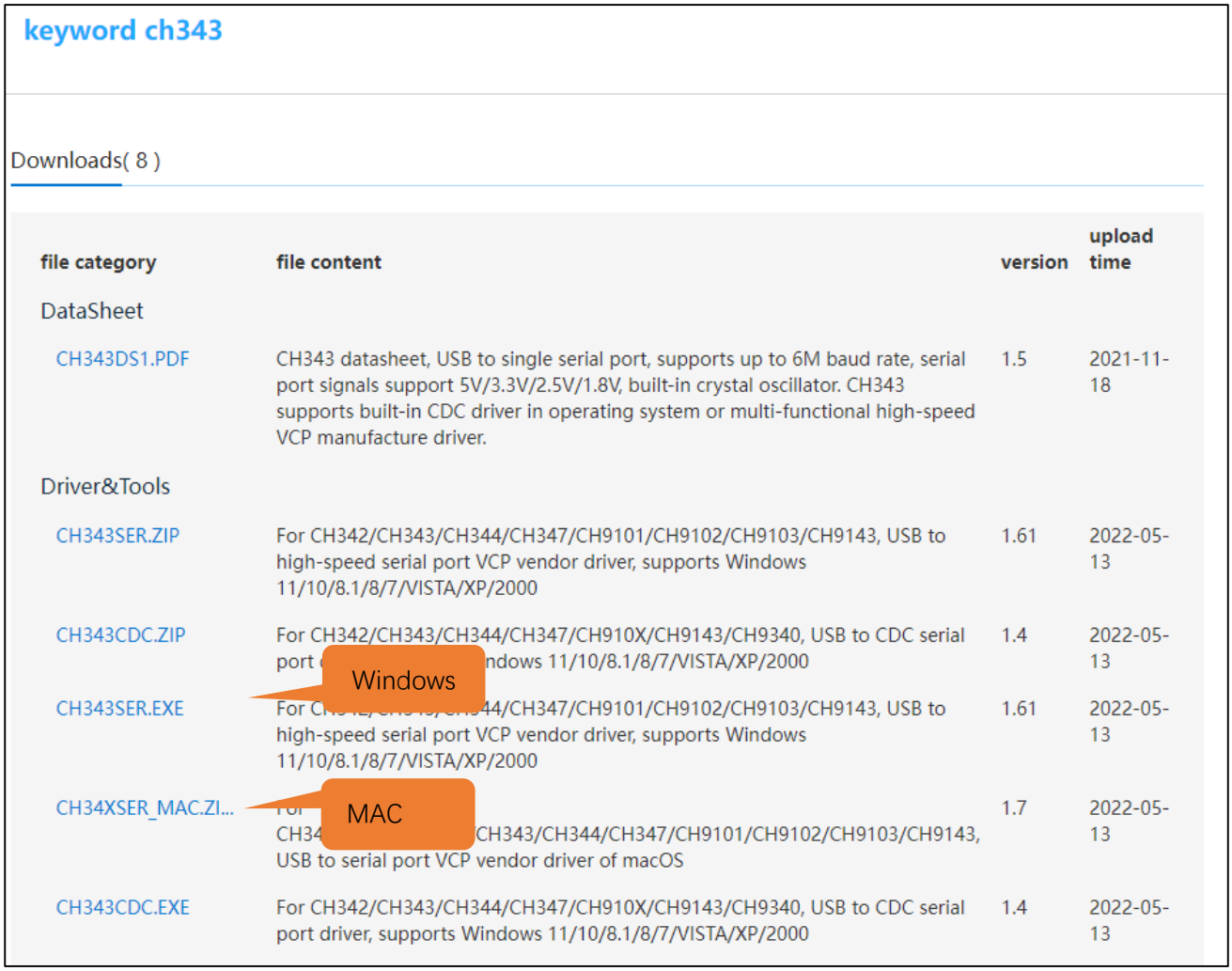
6. When ESP32-S3 WROOM is connected to computer, select "This PC", right-click to select "Manage" and click "Device Manager" in the newly pop-up dialog box, and you can see the following interface.



7. So far, CH343 has been installed successfully. Close all dialog boxes.

MAC

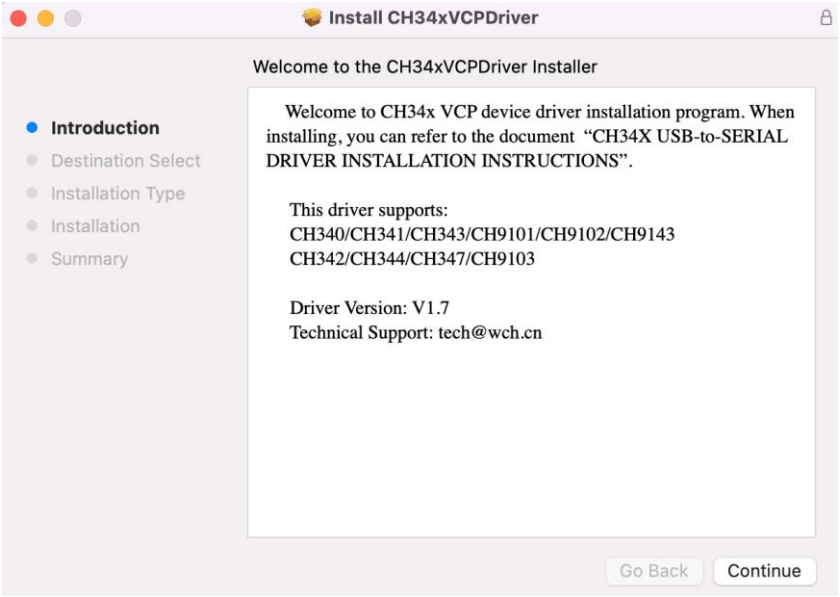
First, download CH343 driver. Click http://www.wch-ic.com/search?t=all&q=ch343 to download the appropriate one based on your operating system.



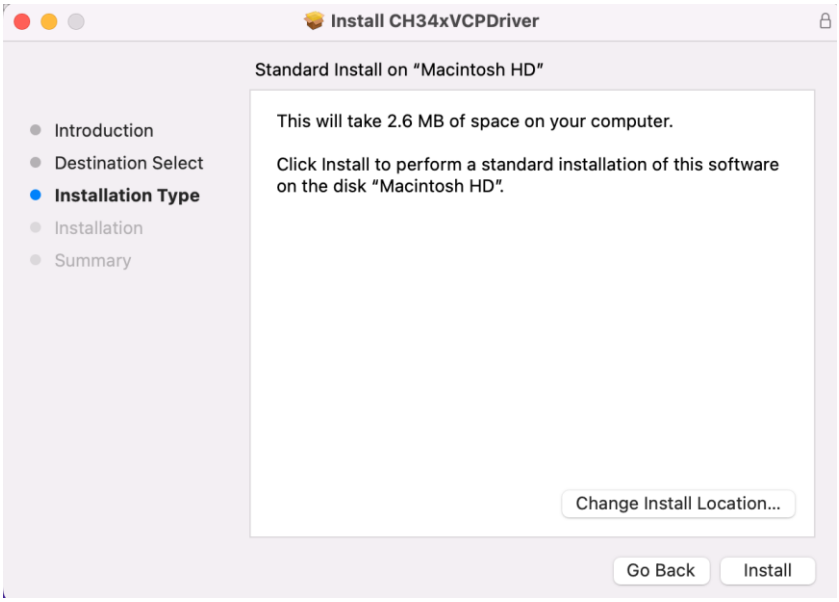
If you would not like to download the installation package, you can open **"Freenove_Ultimate_Starter_Kit_for_ESP32_S3/CH343"**. We have prepared the installation package.

Second, open the folder "**Freenove_Ultimate_Starter_Kit_for_ESP32_S3/CH343/MAC/**"

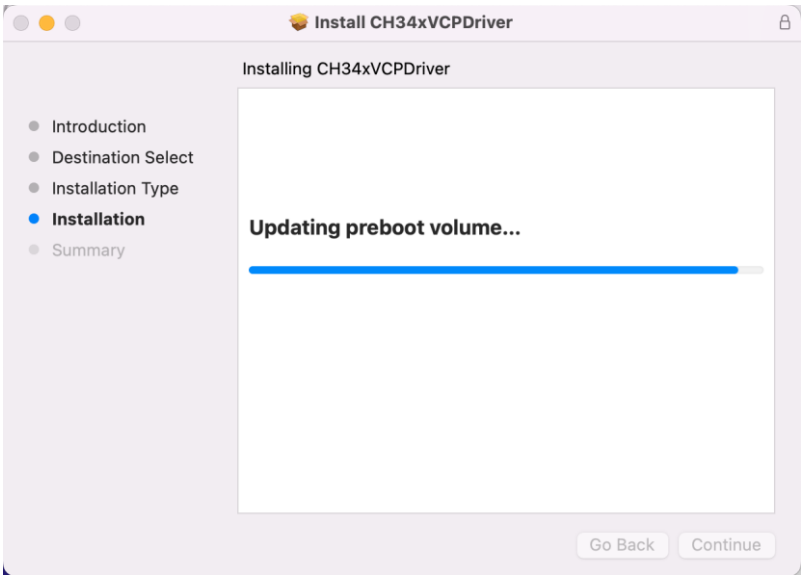
Third, click Continue.



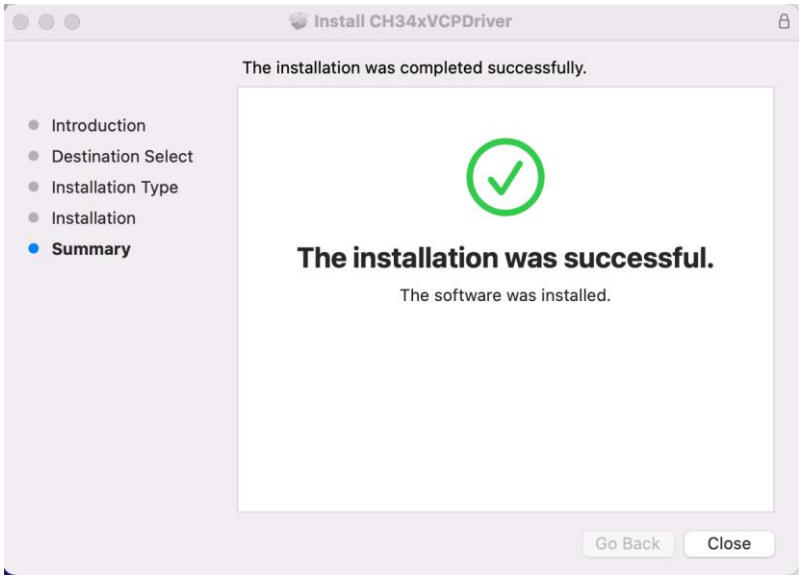
Fourth, click Install.



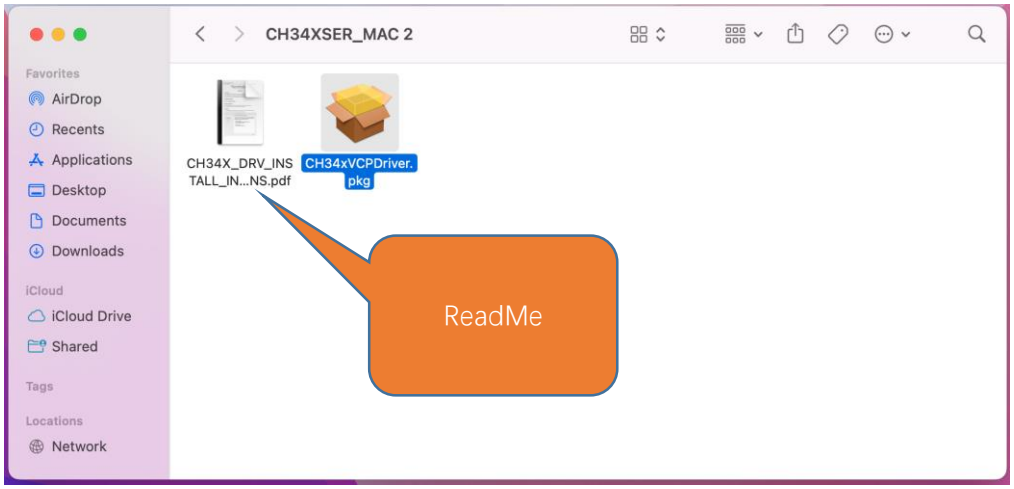
Then, waiting Finsh.



Finally, restart your PC.

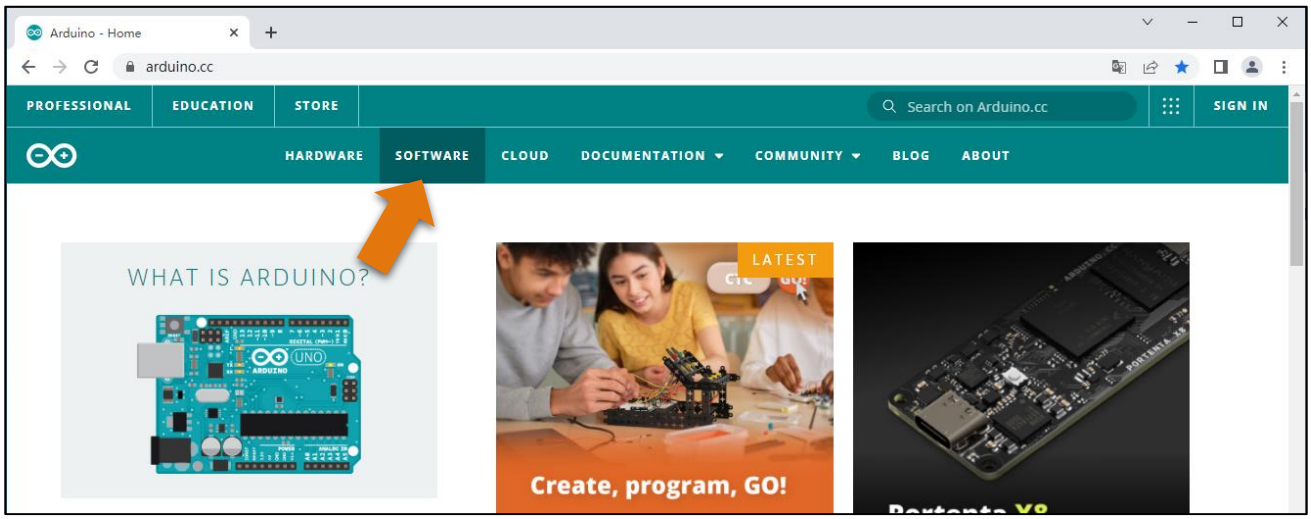


If it fails to be installed with the above steps, you can refer to readme.pdf to install it.

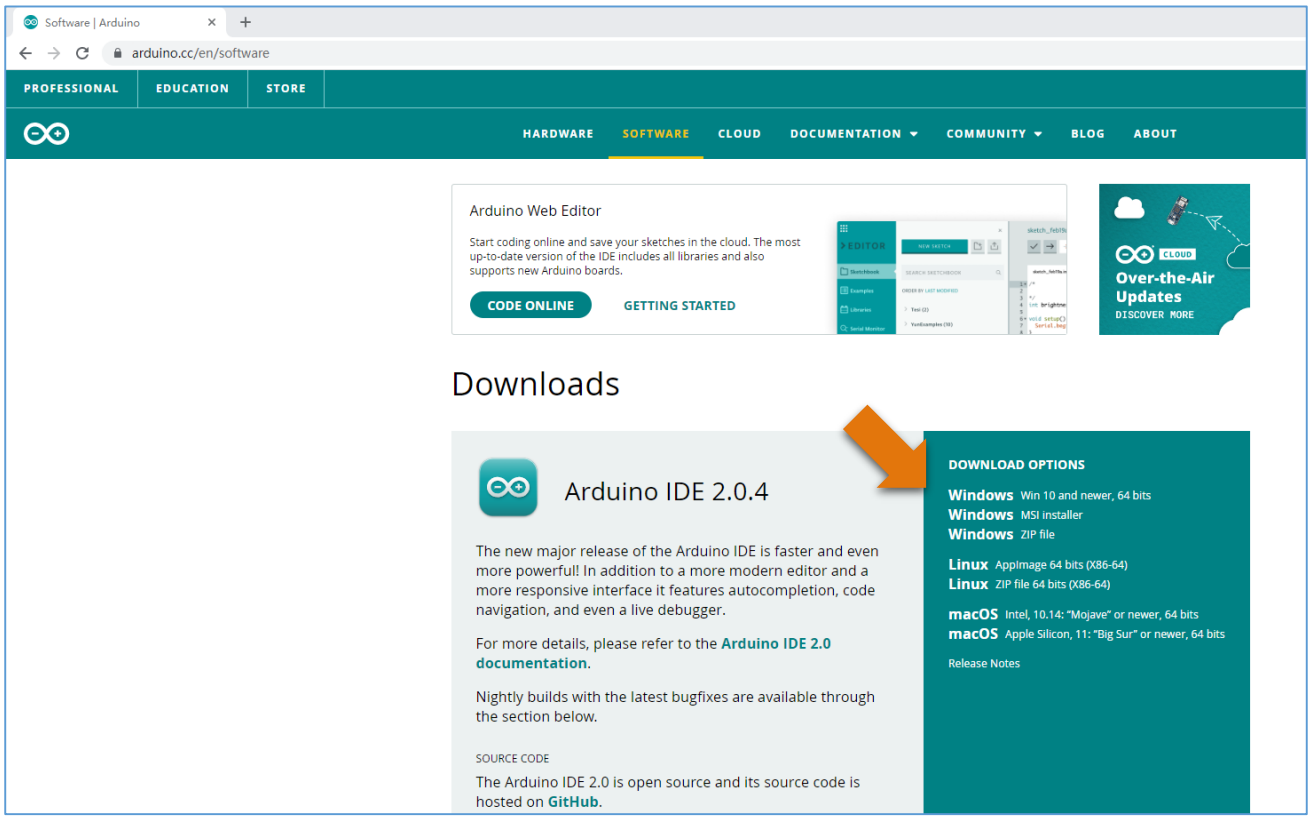


Programming Software

Arduino Software (IDE) is used to write and upload the code for Arduino Board.
First, install Arduino Software (IDE): visit https://www.arduino.cc, click "Download" to enter the download page.

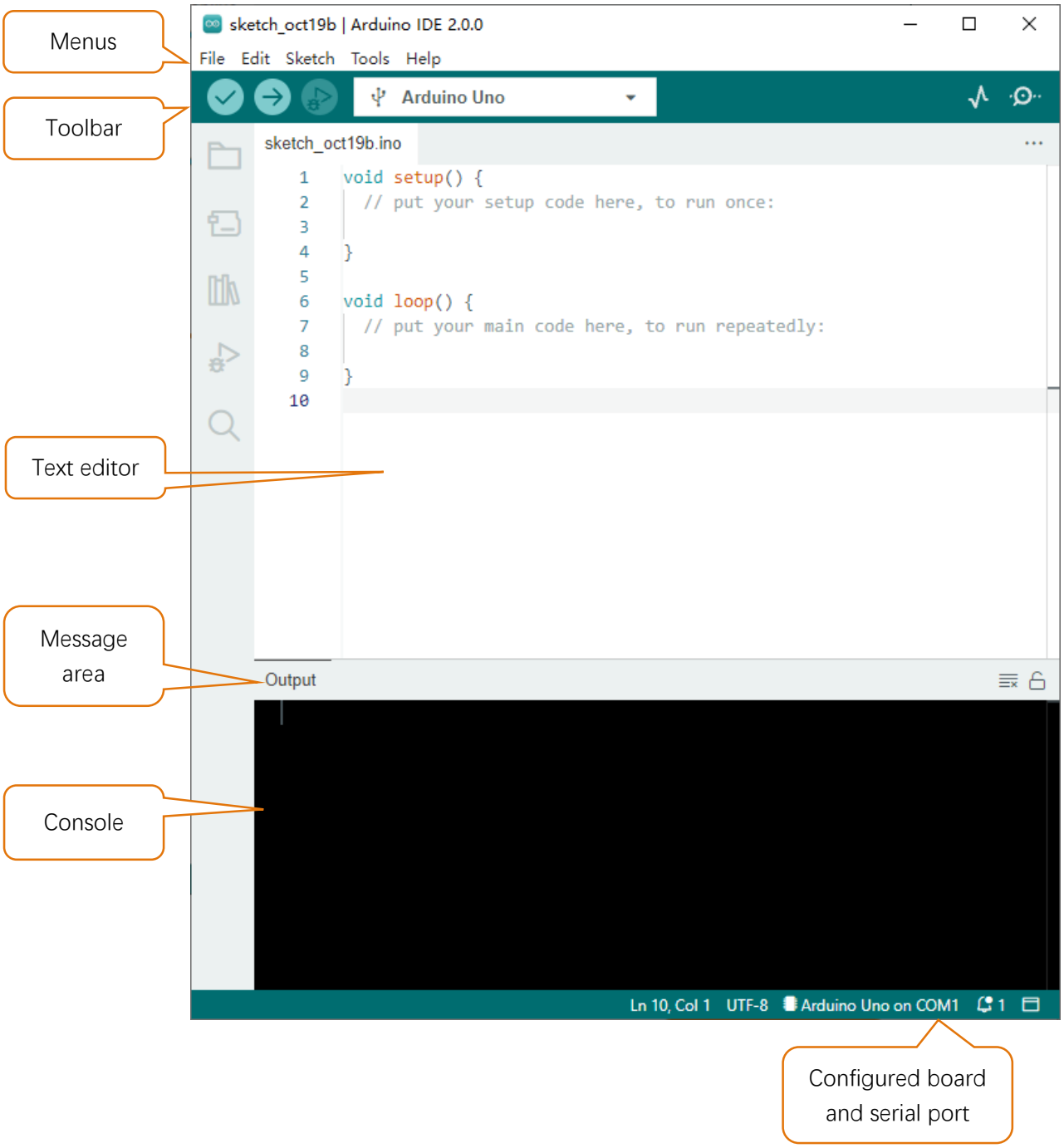


Select and download corresponding installer according to your operating system. If you are a windows user, please select the "Windows" to download and install it correctly.



After the download completes, run the installer. For Windows users, there may pop up an installation dialog box of driver during the installation process. When it popes up, please allow the installation.
After installation completes, an Arduino Software shortcut will be generated in the desktop. Run the Arduino Software.

The interface of Arduino Software is as follows:



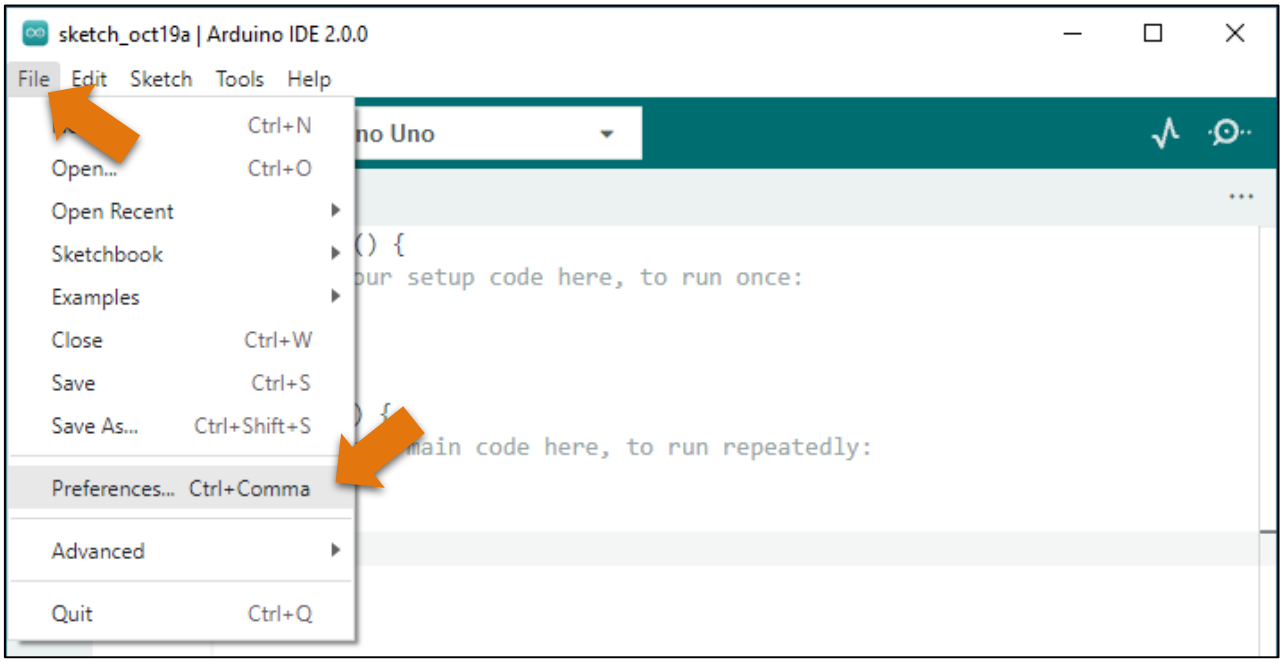
Programs written with Arduino Software (IDE) are called **sketches**. These sketches are written in the text editor and saved with the file extension**.ino**. The editor features text cutting/pasting and searching/replacing. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom right-hand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

	Verify Check your code for compile errors.
	Upload Compile your code and upload them to the configured board.
	Debug Debug code running on the board. (Some development boards do not support this function)
	Development board selection Configure the support package and upload port of the development board.
	Serial Plotter Receive serial port data and plot it in a discounted graph.
	Serial Monitor Open the serial monitor.

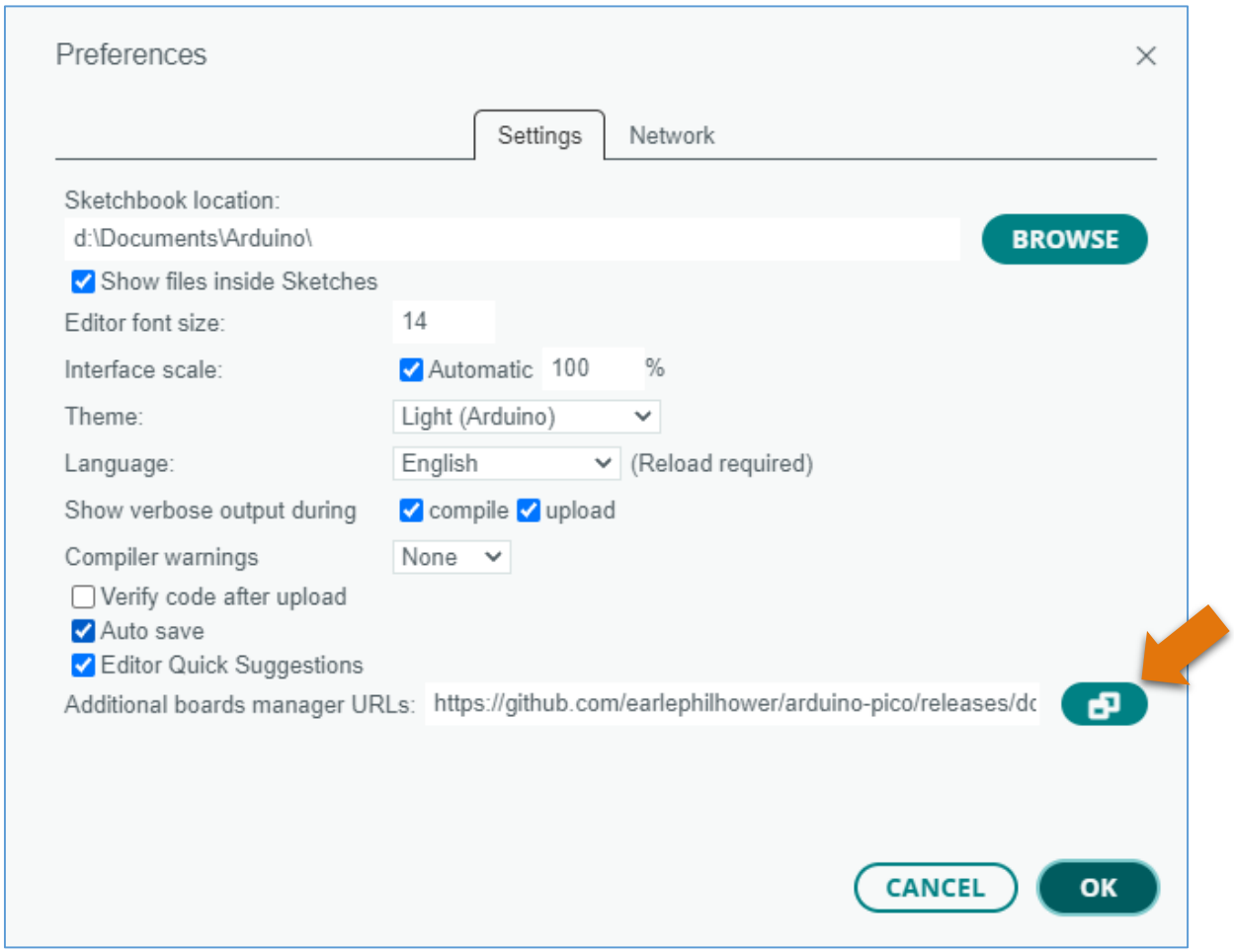
Additional commands are found within the five menus: File, Edit, Sketch, Tools, Help. The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.

Environment Configuration

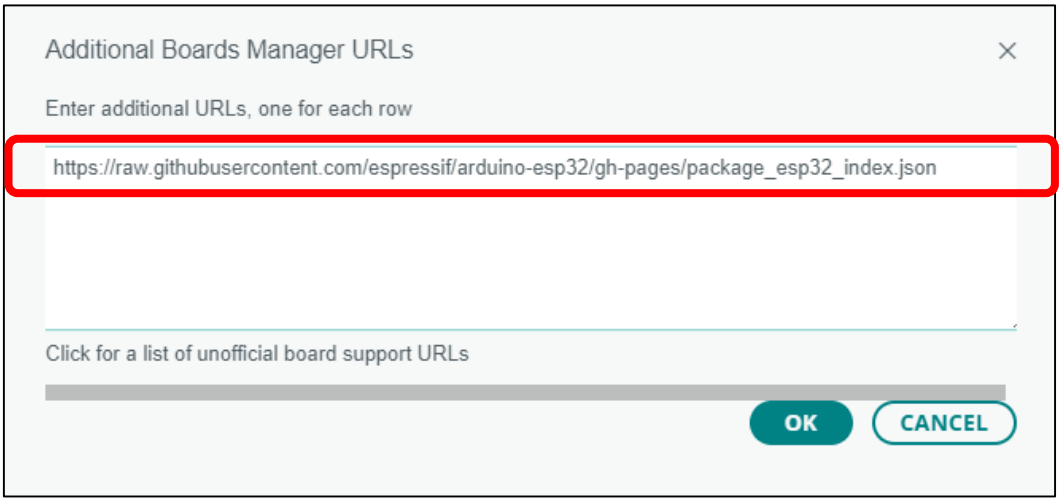
First, open the software platform arduino, and then click File in Menus and select Preferences.



Second, click on the symbol behind "Additional Boards Manager URLs"

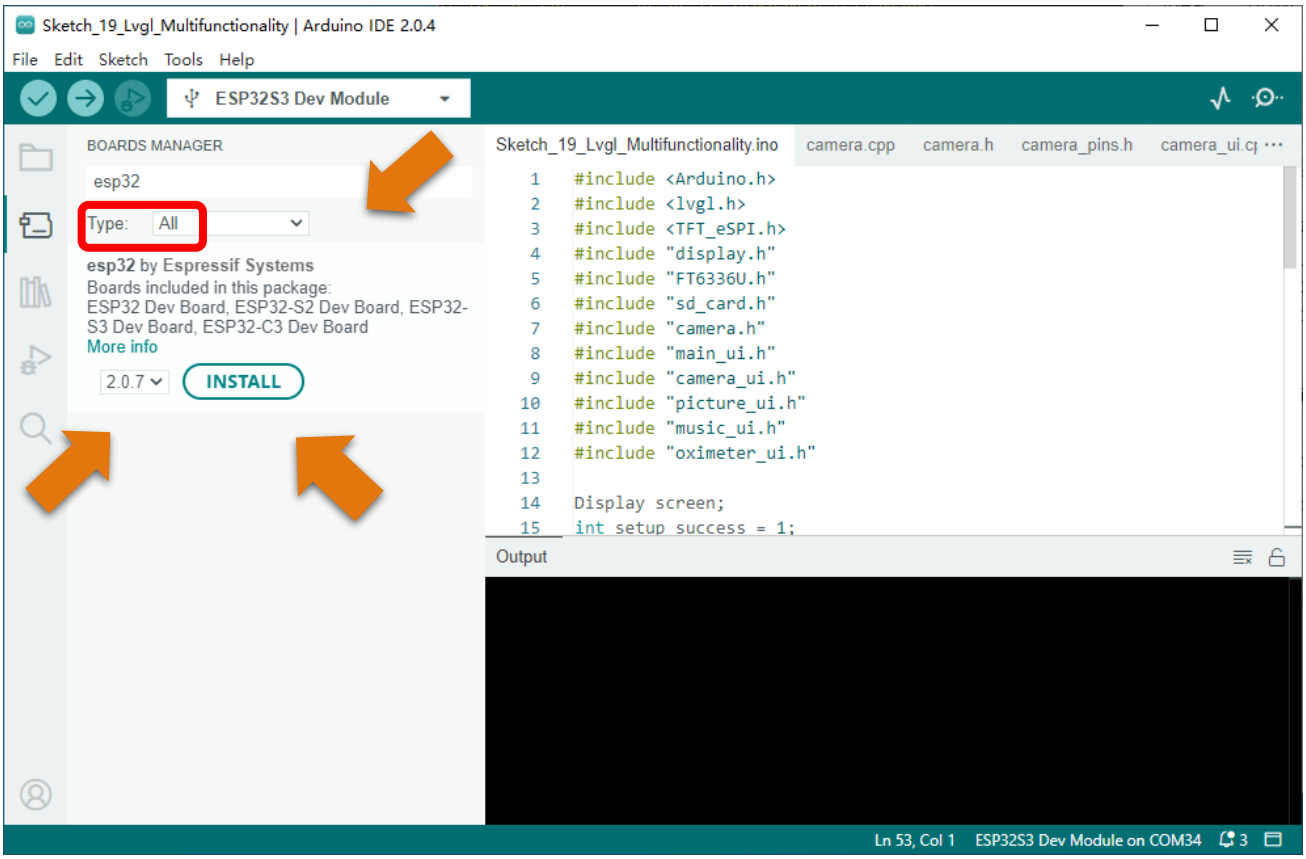


Third, fill in https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json in the new window, click OK, and click OK on the Preferences window again.



Note: if you copy and paste the URL directly, you may lose the "-". Please check carefully to make sure the link is correct.

Fourth, click "Boards Manager". Enter “esp32” in Boards manager, select 2.0.7, and click “INSTALL”.

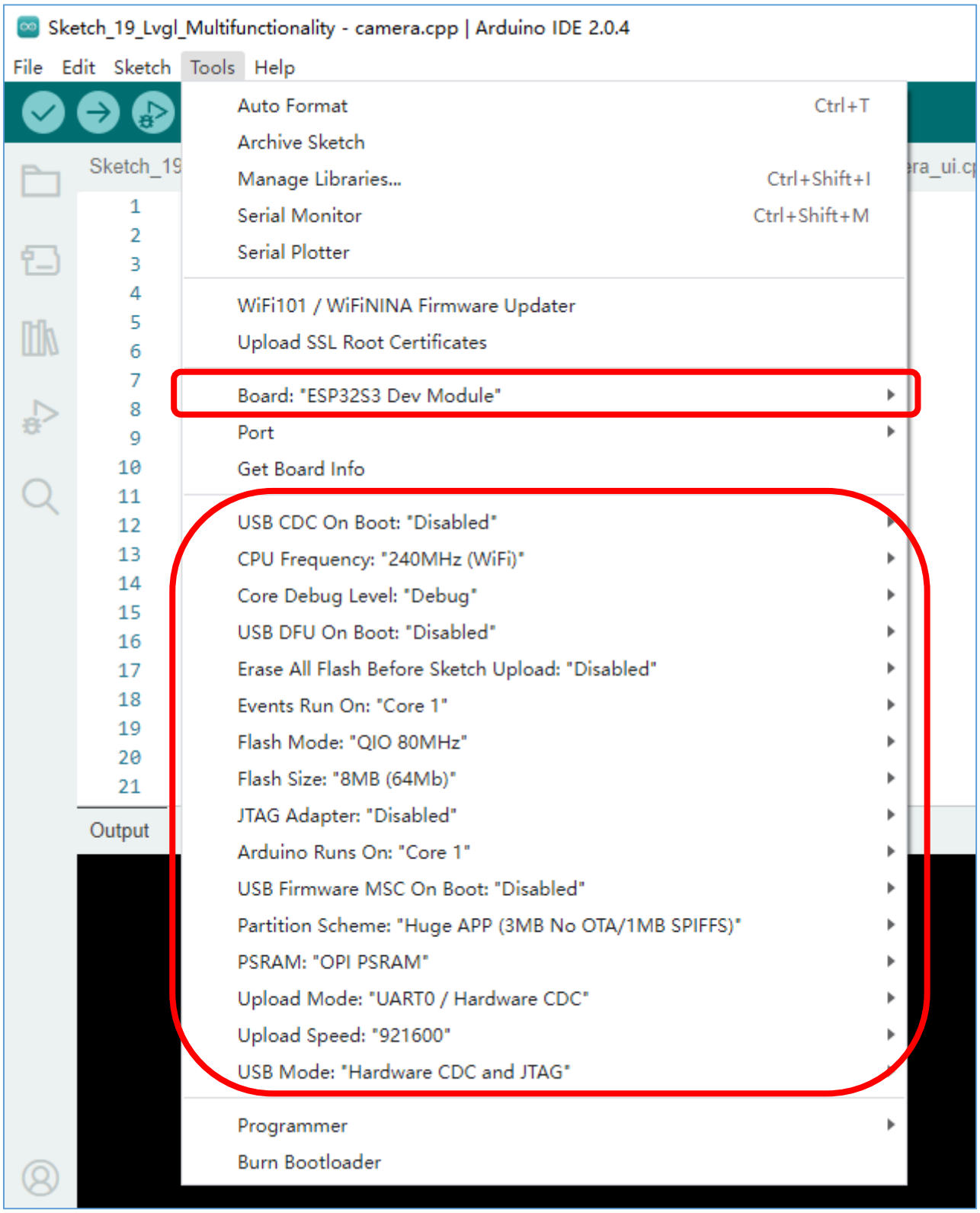


Arduino will download these files automaticly. Wait for the installation to complete.

Output

```
Downloading packages
esp32:xtensa-esp32-elf-gcc@esp-2021r2-patch5-8.4.0
esp32:xtensa-esp32s2-elf-gcc@esp-2021r2-patch5-8.4.0
esp32:xtensa-esp32s3-elf-gcc@esp-2021r2-patch5-8.4.0
esp32:riscv32-esp-elf-gcc@esp-2021r2-patch5-8.4.0
esp32:openocd-esp32@v0.11.0-esp32-20221026
esp32:esptool_py@4.5
esp32:mkspiffs@0.2.3
esp32:mklittlefs@3.0.0-gnu12-dc7f933
esp32:esp32@2.0.7
Installing esp32:xtensa-esp32-elf-gcc@esp-2021r2-patch5-8.4.0
Configuring tool.
esp32:xtensa-esp32-elf-gcc@esp-2021r2-patch5-8.4.0 installed
Installing esp32:xtensa-esp32s2-elf-gcc@esp-2021r2-patch5-8.4.0
Configuring tool.
esp32:xtensa-esp32s2-elf-gcc@esp-2021r2-patch5-8.4.0 installed
Installing esp32:xtensa-esp32s3-elf-gcc@esp-2021r2-patch5-8.4.0
Configuring tool.
esp32:xtensa-esp32s3-elf-gcc@esp-2021r2-patch5-8.4.0 installed
Installing esp32:riscv32-esp-elf-gcc@esp-2021r2-patch5-8.4.0
Configuring tool.
esp32:riscv32-esp-elf-gcc@esp-2021r2-patch5-8.4.0 installed
Installing esp32:openocd-esp32@v0.11.0-esp32-20221026
Configuring tool.
esp32:openocd-esp32@v0.11.0-esp32-20221026 installed
Installing esp32:esptool_py@4.5
Configuring tool.
esp32:esptool_py@4.5 installed
Installing esp32:mkspiffs@0.2.3
Configuring tool.
esp32:mkspiffs@0.2.3 installed
Installing esp32:mklittlefs@3.0.0-gnu12-dc7f933
Configuring tool.
esp32:mklittlefs@3.0.0-gnu12-dc7f933 installed
Installing platform esp32:esp32@2.0.7
Configuring platform.
Platform esp32:esp32@2.0.7 installed
```

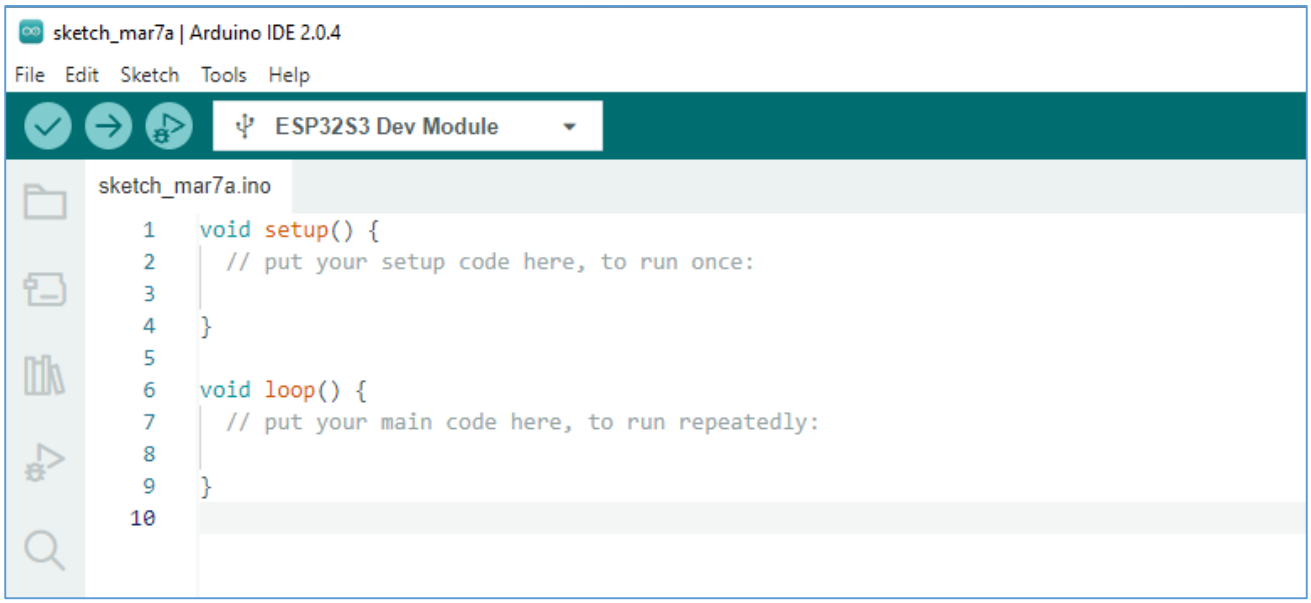
When finishing installation, click Tools in the Menus again and select Board: "ESP32S3 Dev Module", and then you can see information of ESP32-S3.



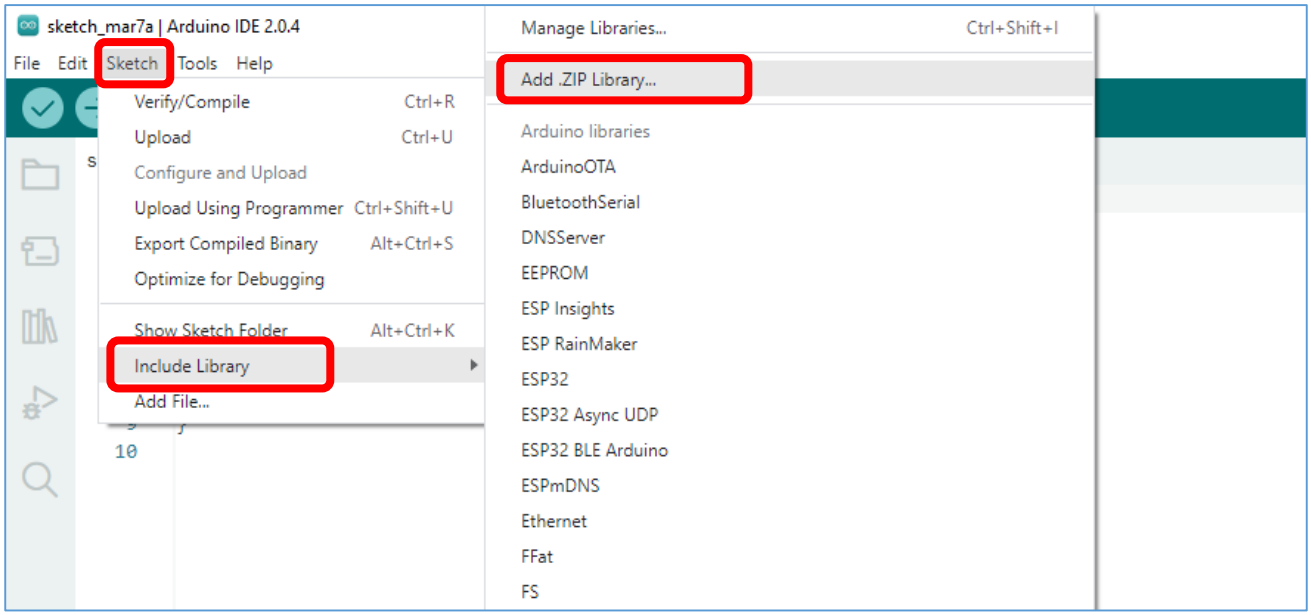
Library Installation

Before starting the learning process, it is necessary to install some libraries in advance to enable the code to be compiled properly. For convenience, we have already packaged these libraries and placed them in the Freenove Development Kit for ESP32-S3/Libraries folder. Please refer to the following steps to install these libraries into the Arduino IDE.

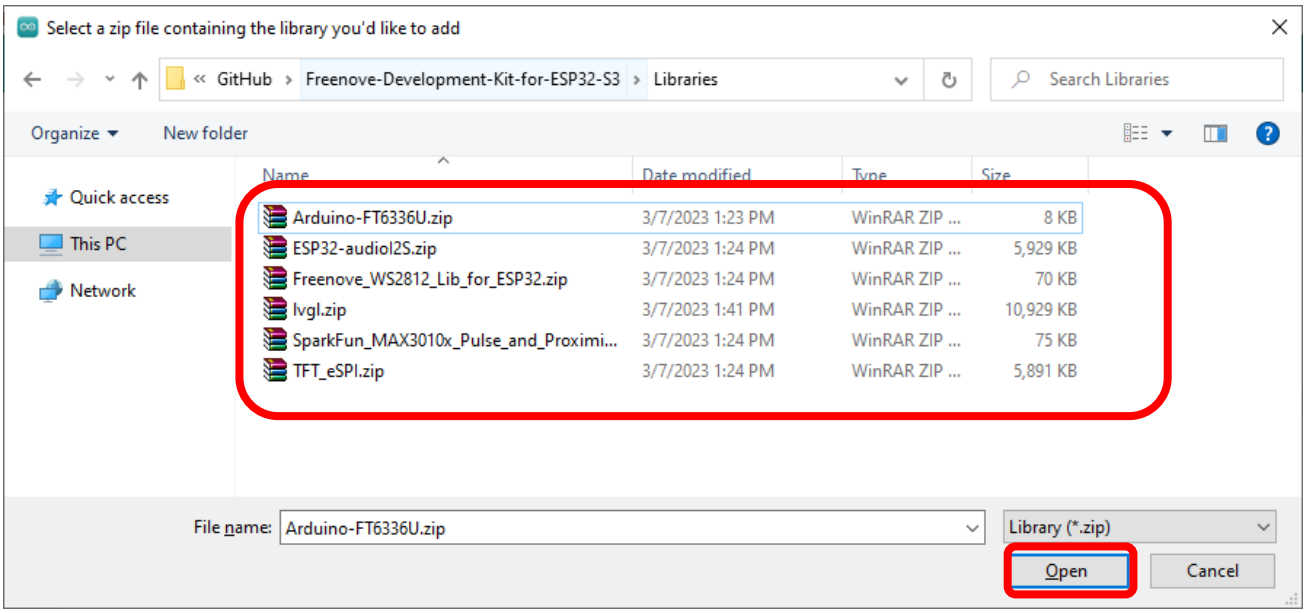
1. Open Arduino IDE.



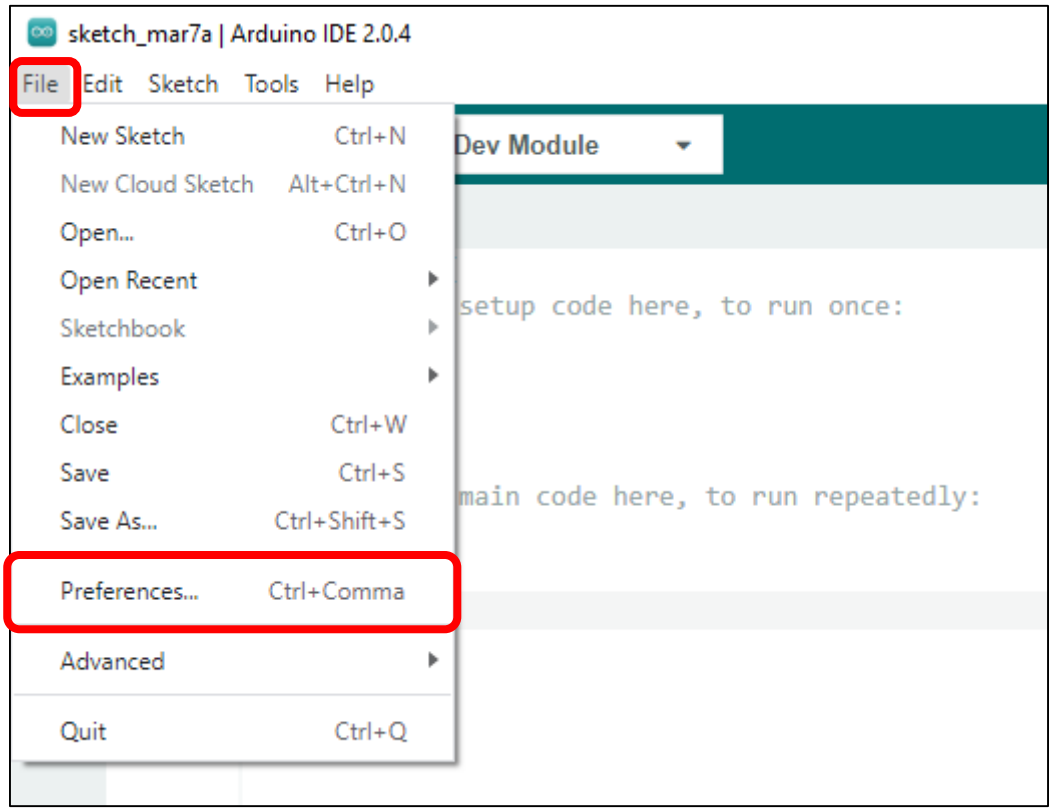
2. Select Sketch->Include Library->Add .ZIP library….



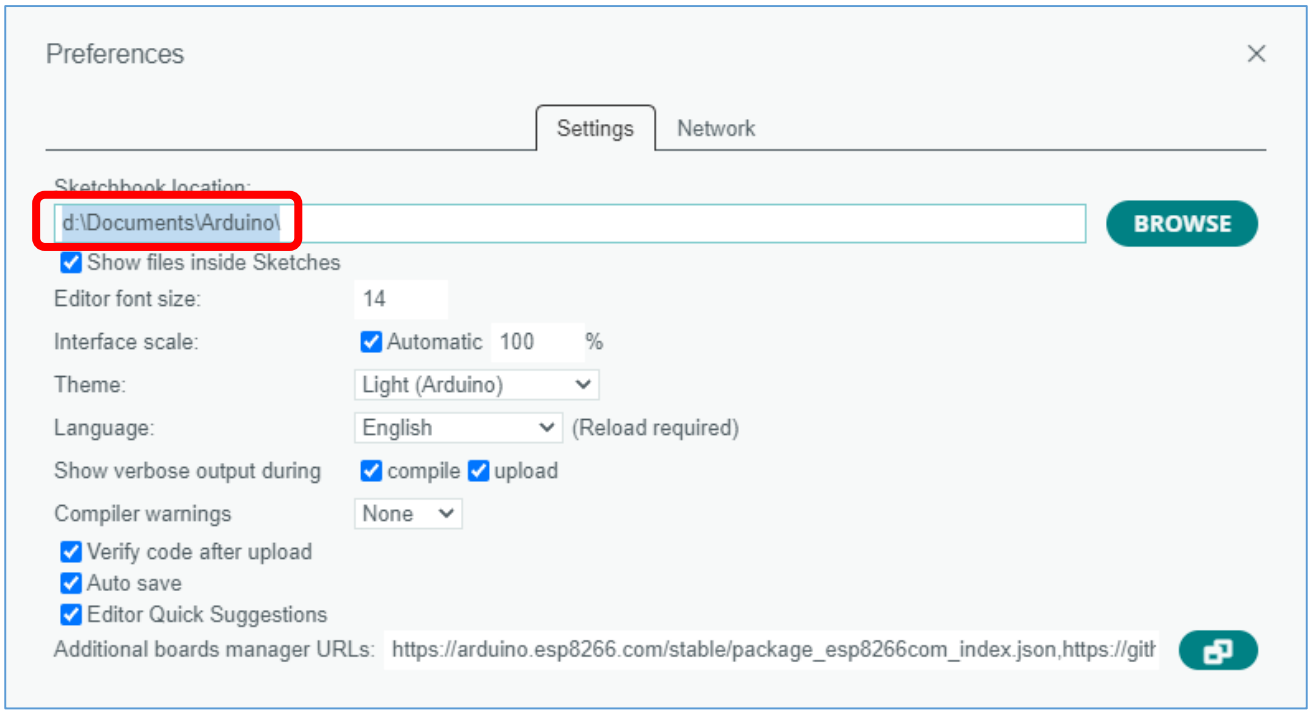
3. On the newly pop-up window, select the files from the **Freenove-Development-Kit-for-ESP32-S3/Libraries**. Click Open to install the library.



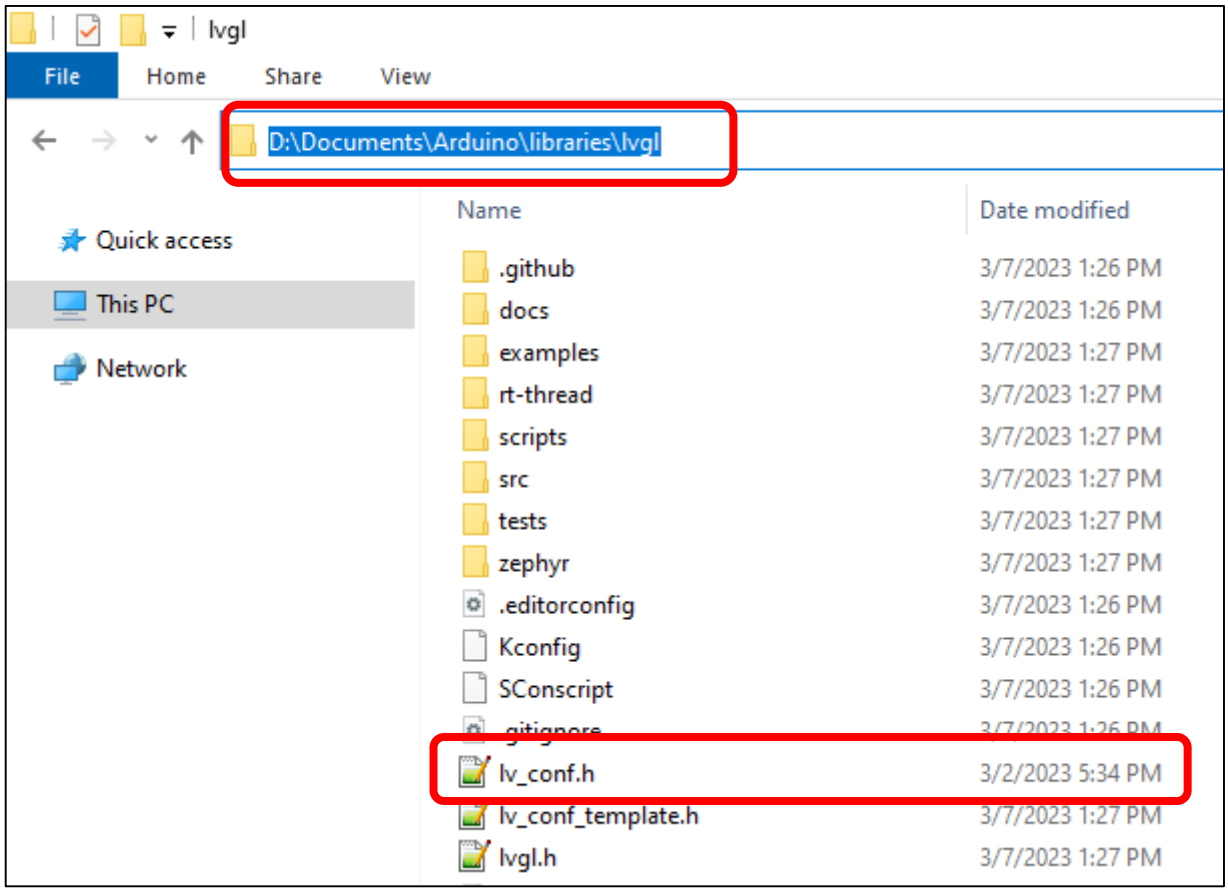
4. **Repear the above steos until all the six libraries are installed to Arduino.**
5. Click File->Preferences.



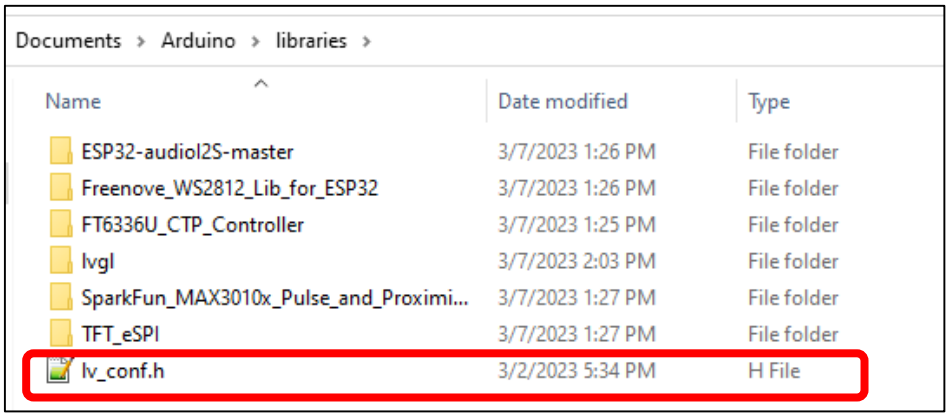
6. On the newly pop-up window, you can see Sketchbook location bar. Copy this location.



7. Paste and find the location on Windows computer. Enter lvgl folder.

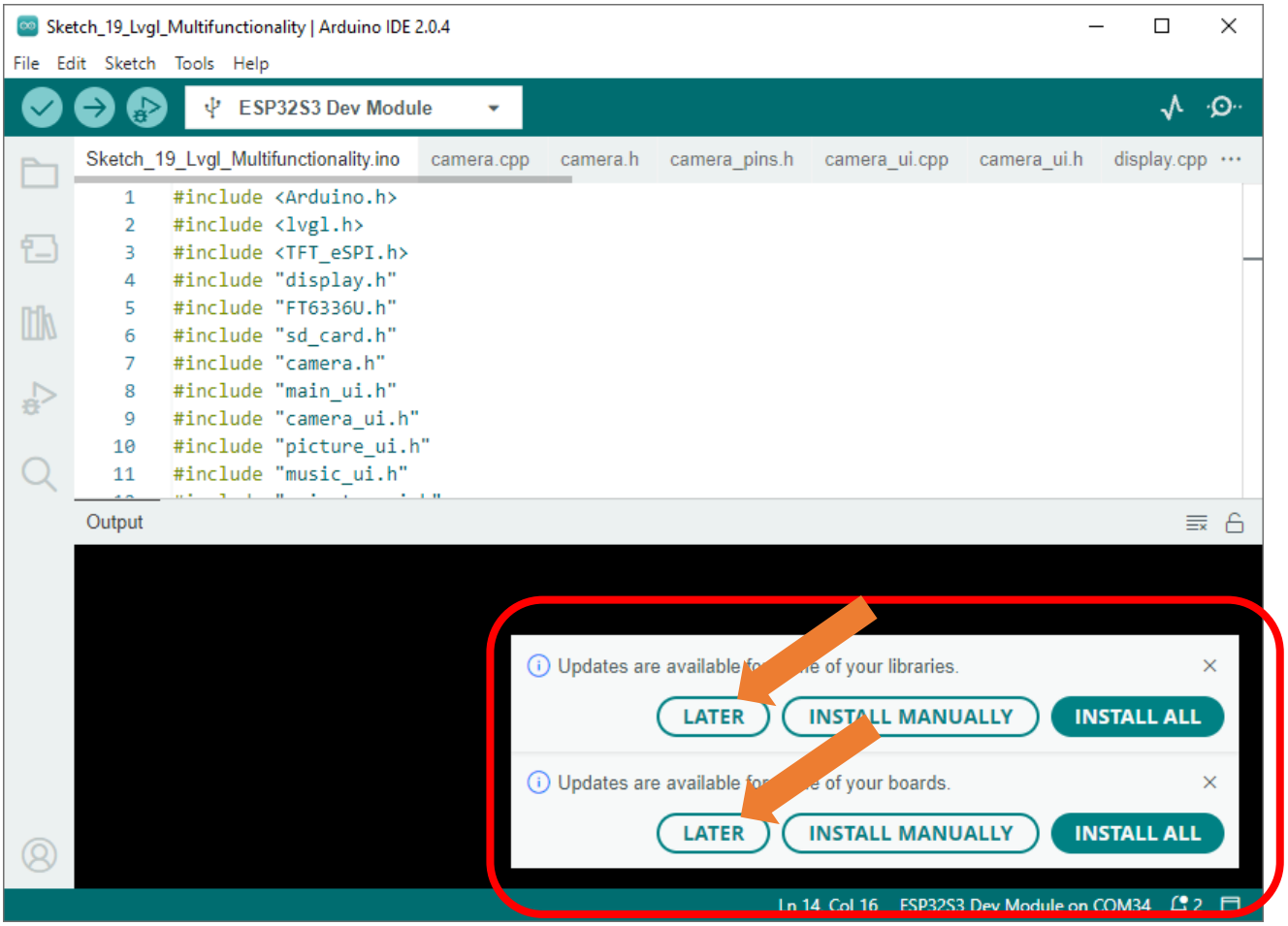


8. Move lv_conf.h in the lvgl folder to the same level directory as the lvgl folder.



9. So far, all libraries have been installed.

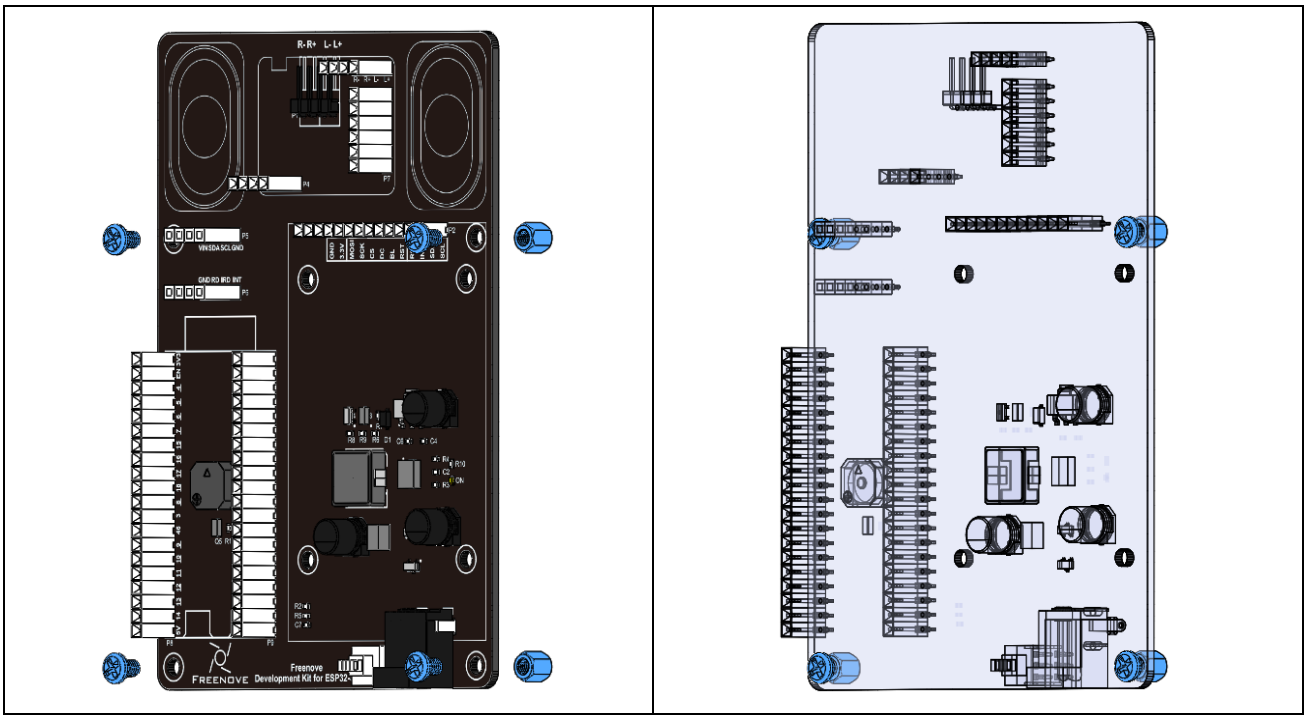
Note: Some libraries are not the latest version. Please do not update them even if it prompts every time you open the IDE. Just click LATER. Otherwise, it may lead the compilation to fail.



Chapter 0 Assembly

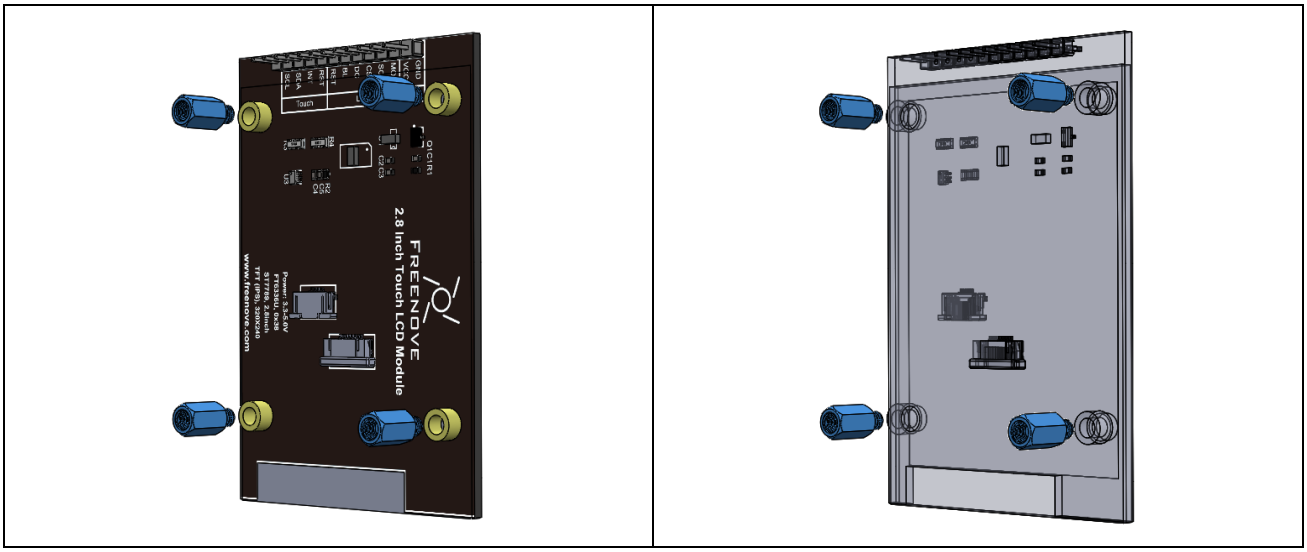
Installation of Brass Standoffs

Mount four M3*5 brass standoffs to the ESP32-S3 WROOM shield with four M3*4 screws, as shown below:

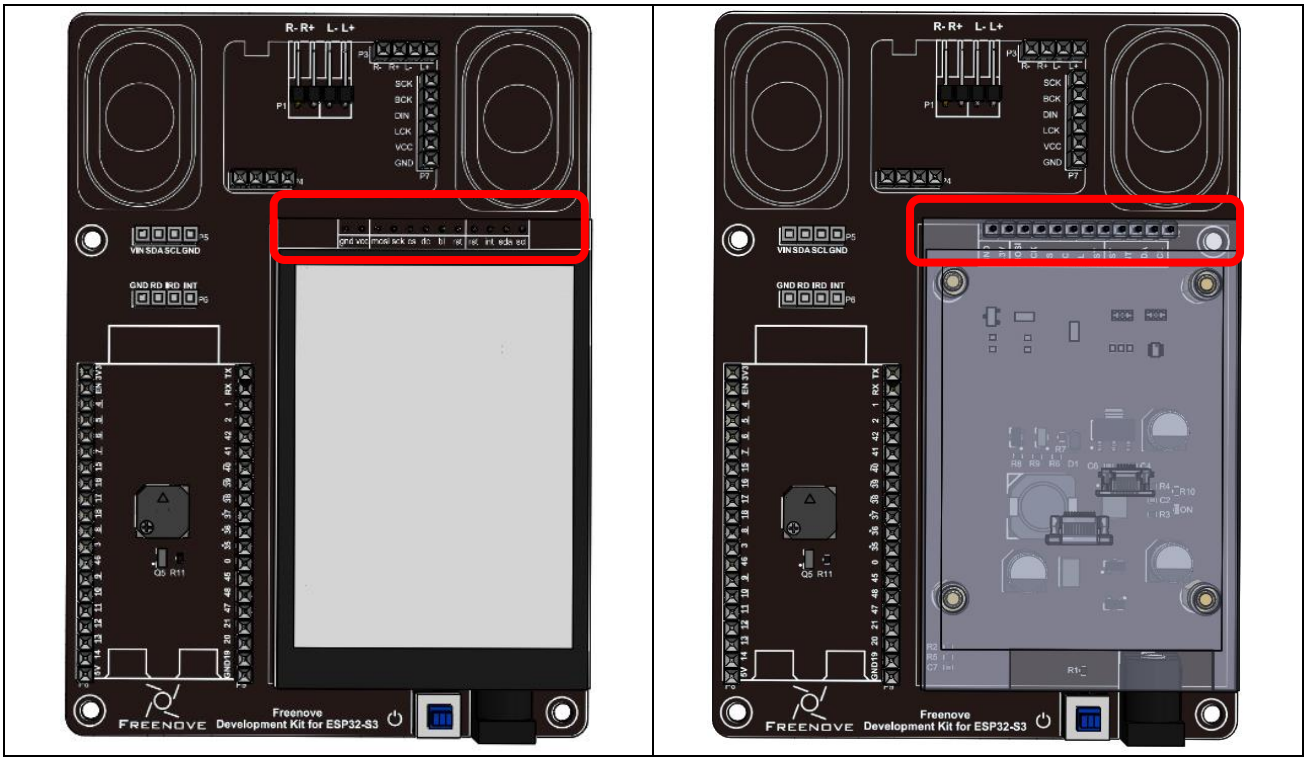


Installation of Screen

Mount four M3*8+3 single-pass standoffs to the back of the screen.

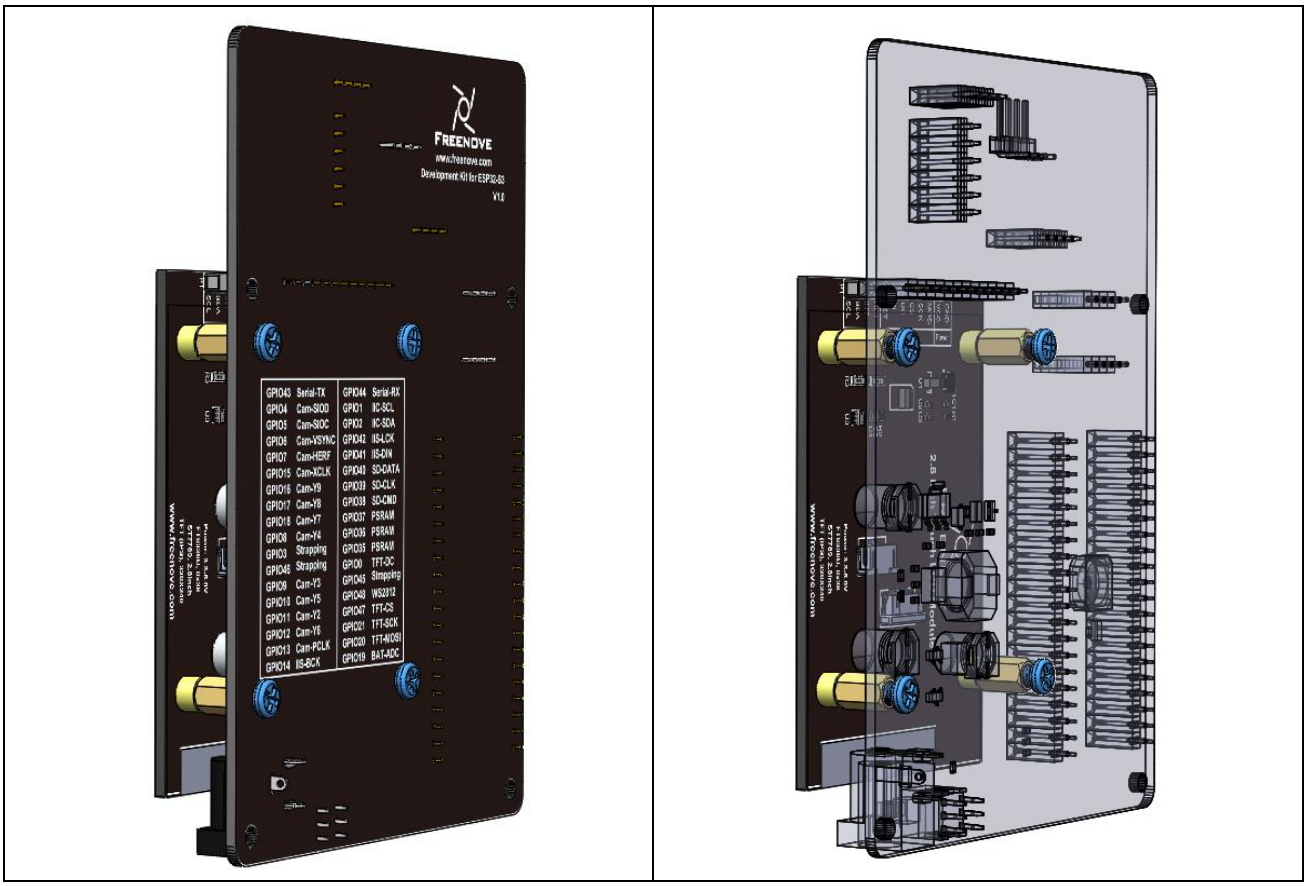


Plug the screen to ESP32-S3 WROOM Shield.



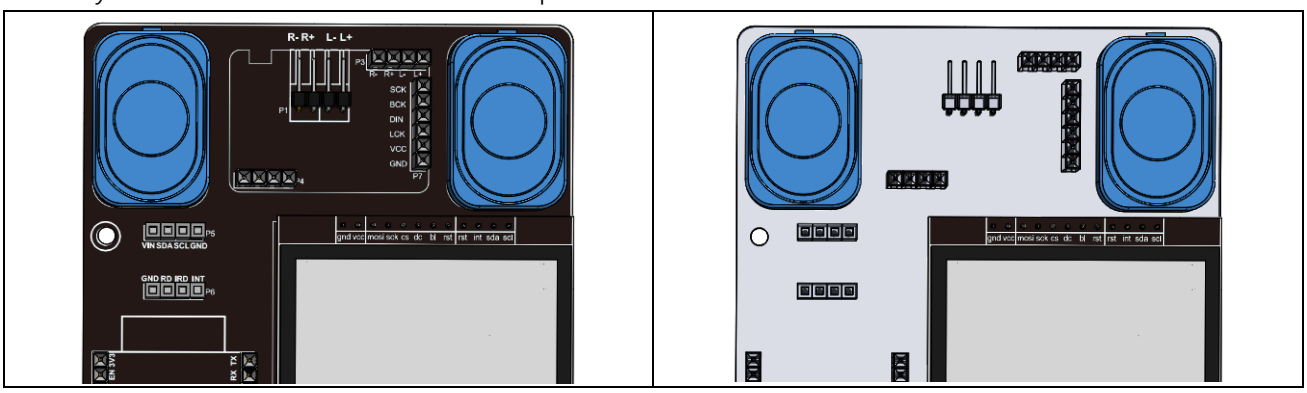
It should be noted that when plugging the screen, you should have the pins in line with each other to avoid misalignment.

Fix the screen to ESP32-S3 WROOM Shield with four M3*4 screws.

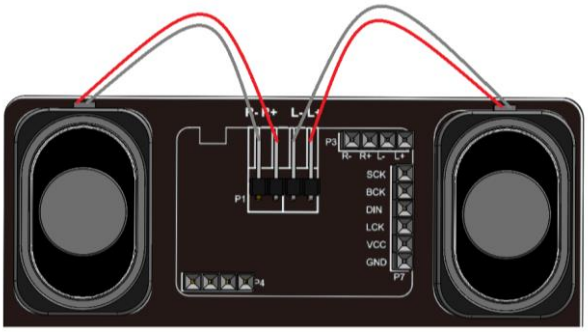


Installation of Speakers

Carefully tear the sticker on the back of the speakers and attach them to the ESP32-S3 WROOM Shield.

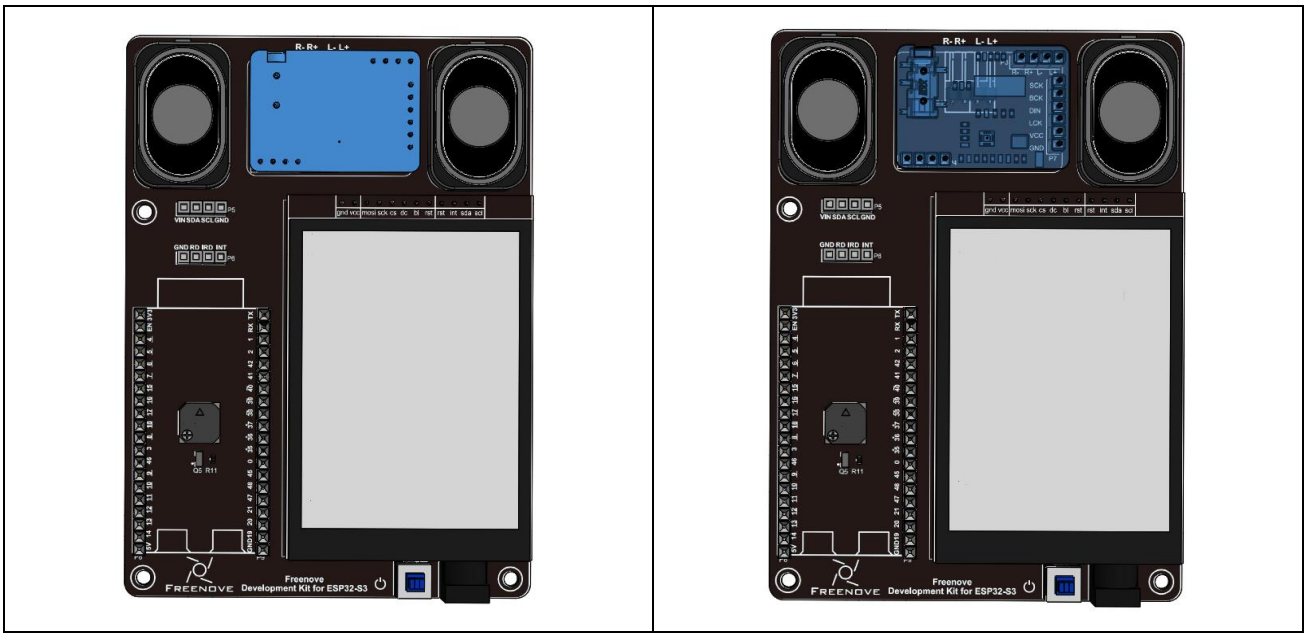


Connect the wires of the speakers to ESP32-S3 WROOM Shield, as shown below:



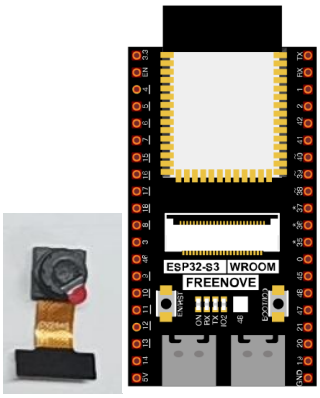
Installation of the Audio Module

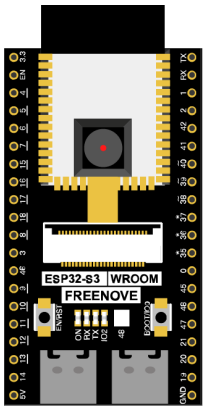
Plug the audio module to the shield.



Connection of Camera

Gently pull up the camera connector and insert the camera module, and then press it down until it clicks into place.

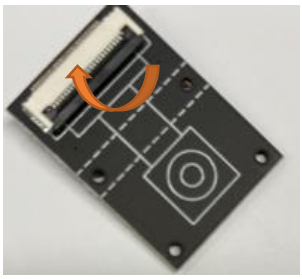


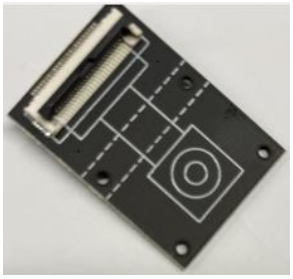


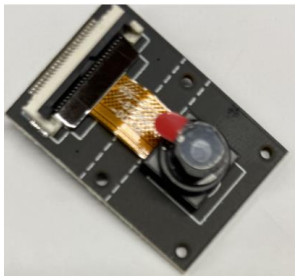
Note:

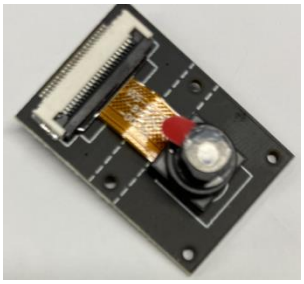
All cameras in this tutorial are mounted on the esp32s3 board. If you want to extend it, you can mount the camera onto the expansion pad and use the fpc cable to expand.

Connect the camera to its extension board.







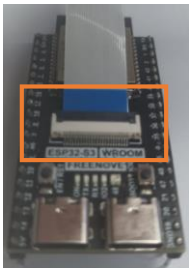


Plug one end of the camera cable into the camera extension board. **Pay attention to blue side.**

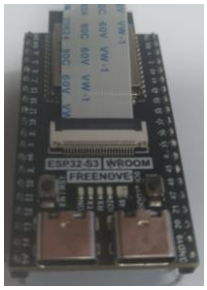
√

×

Plug the other end into the ESP32.



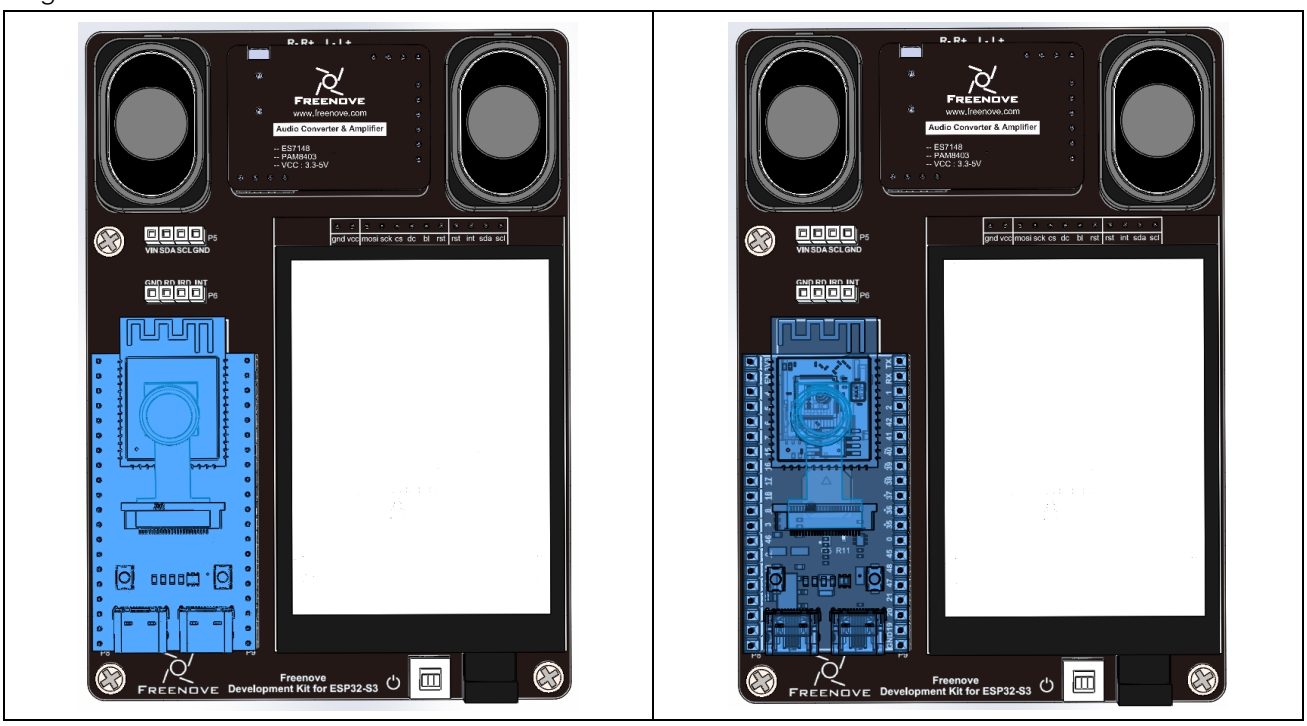
√



×

Connection of ESP32-S3 WROOM

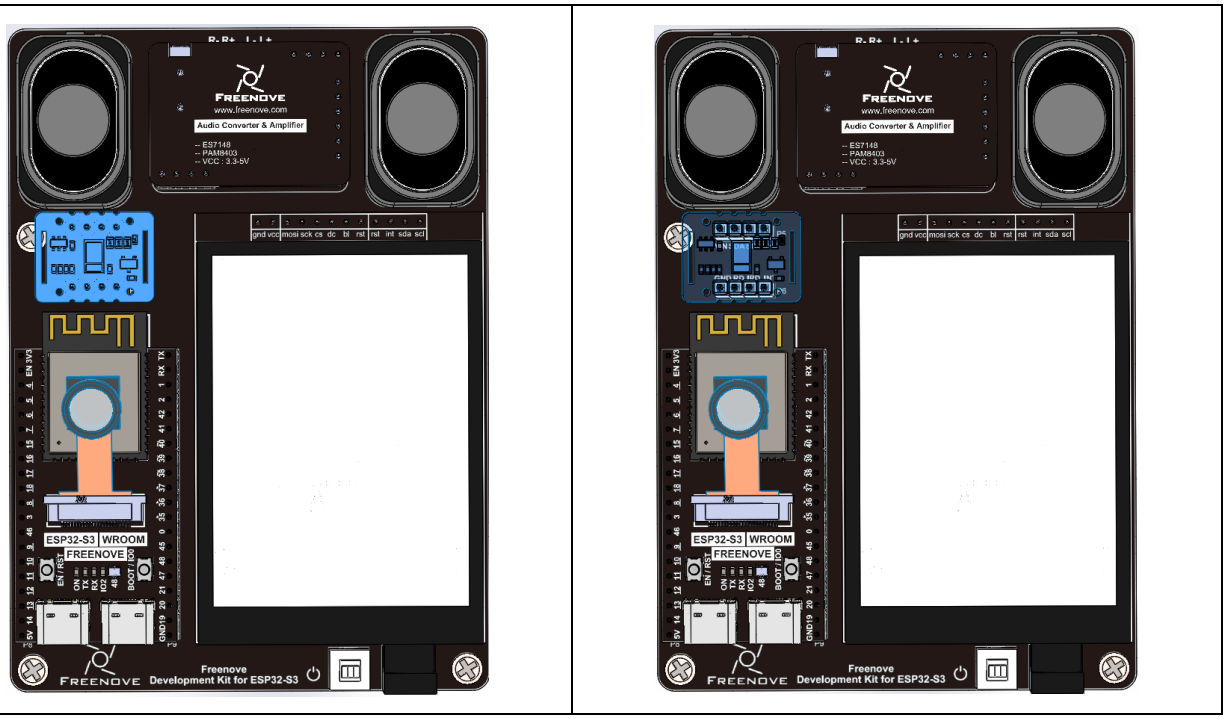
Plug the ESP32-S3 WROOM to the shield.



When installing the ESP32-S3 WROOM, ensure that the ESP32-S3 wroom is not improperly installed.

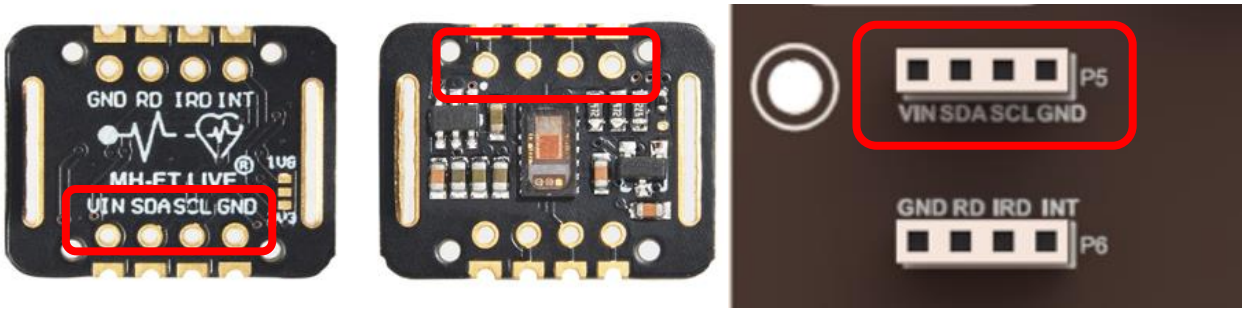
Connection of MAX30102

Plug the MAX30102to the shield.



Important notes:

The MAX30102 module should not be connected reversely, otherwise it may affect the circuit, or even cause damages. It must be connected strictly according to the texts on the board. Make sure the pins are connected pin to pin.



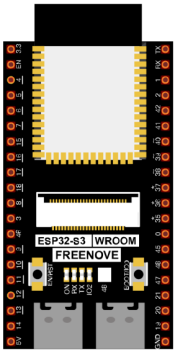
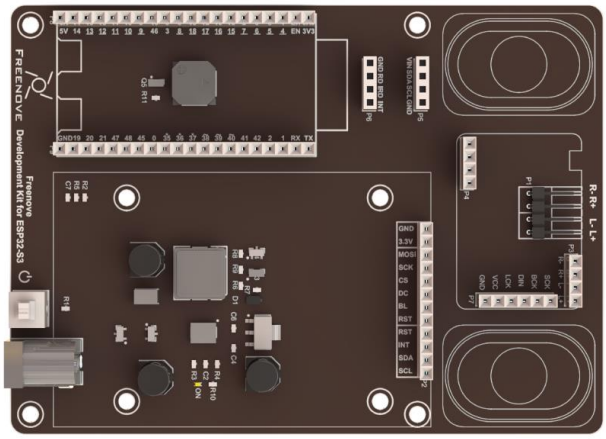
Chapter1 ADC Test

In this chapter, we will learn how to use ESP32-S3 to read analog signals.

Project 1.1 Read the Voltage of Power

In this project, we will use the ADC function of ESP32-S3 to read the voltage value of the power and print it out through the serial monitor.

Component List

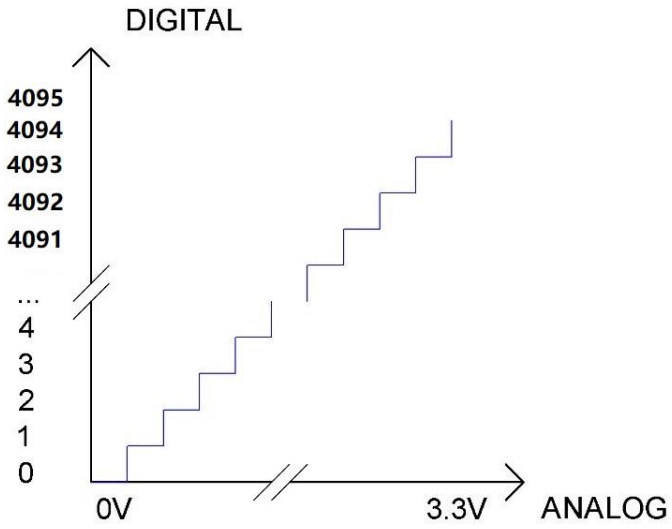
ESP32-S3 WROOM x1 	ESP32-S3 WROOM Shield x1 	
9V Battery Cable x1 	9V battery x1 (Not included in the kit, prepared by yourself) 	Type-C cable x1 

Please Note: Batteries need to be prepared by yourself. We do not provide batteries. ESP32-S3 WROOM Shield features with a DC005 interface, whose size is 5.5x2.1cm. If using an external power supply, ensure that its voltage ranges between 6-12V and that the current is greater than 0.5A.

Related knowledge

ADC

An ADC is an electronic integrated circuit used to convert analog signals such as voltages to digital or binary form consisting of 1s and 0s. The range of our ADC on ESP32-S3 is 12 bits, which means it can represent a resolution of up to 2^12=4096, which is the number of discrete values that the ADC can represent within its range (3.3V). The range of analog values corresponds to ADC values. So the more bits the ADC has, the denser the partition of analog will be and the greater the precision of the resulting conversion.

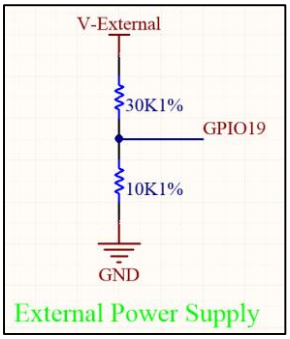


Subsection 1: the analog in rang of 0V---3.3/4095 V corresponds to digital 0;
Subsection 2: the analog in rang of 3.3/4095 V---2*3.3 /4095V corresponds to digital 1;
...
The following analog will be divided accordingly.
The conversion formula is as follows:

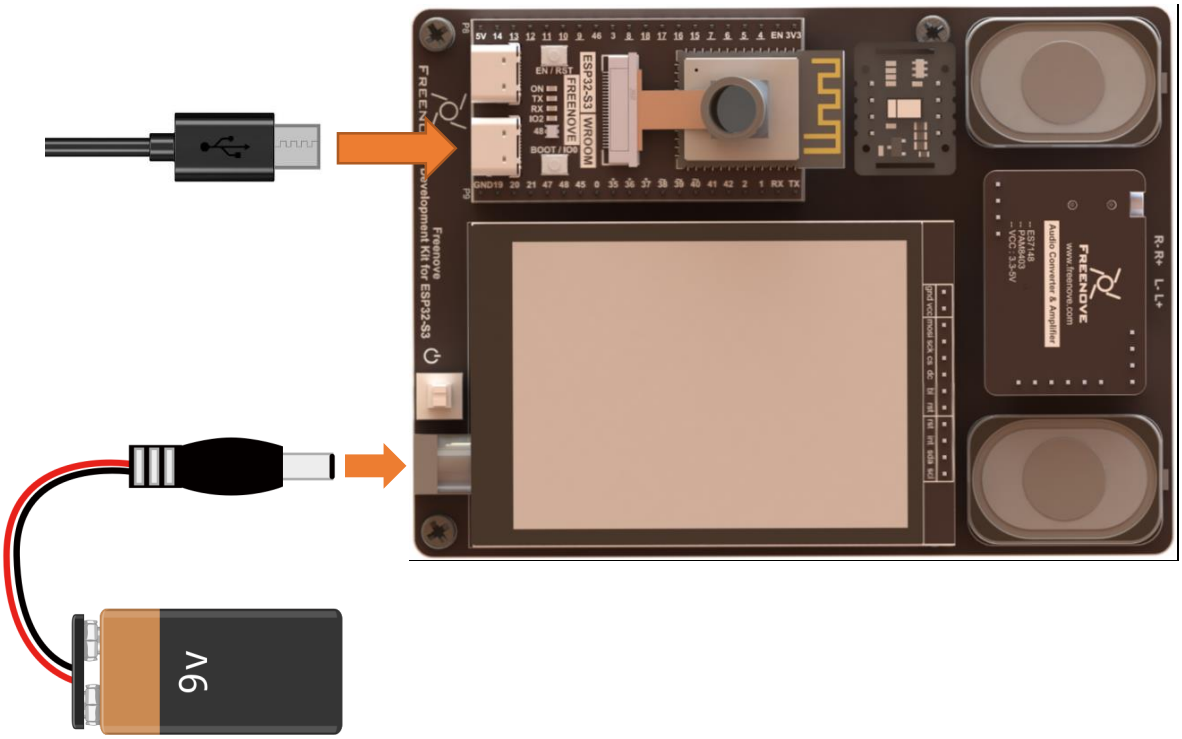
$$ADC\ \ Value = \frac{\text{Analog Voltage}}{3.3} * 4095$$

Circuit

Schematic diagram

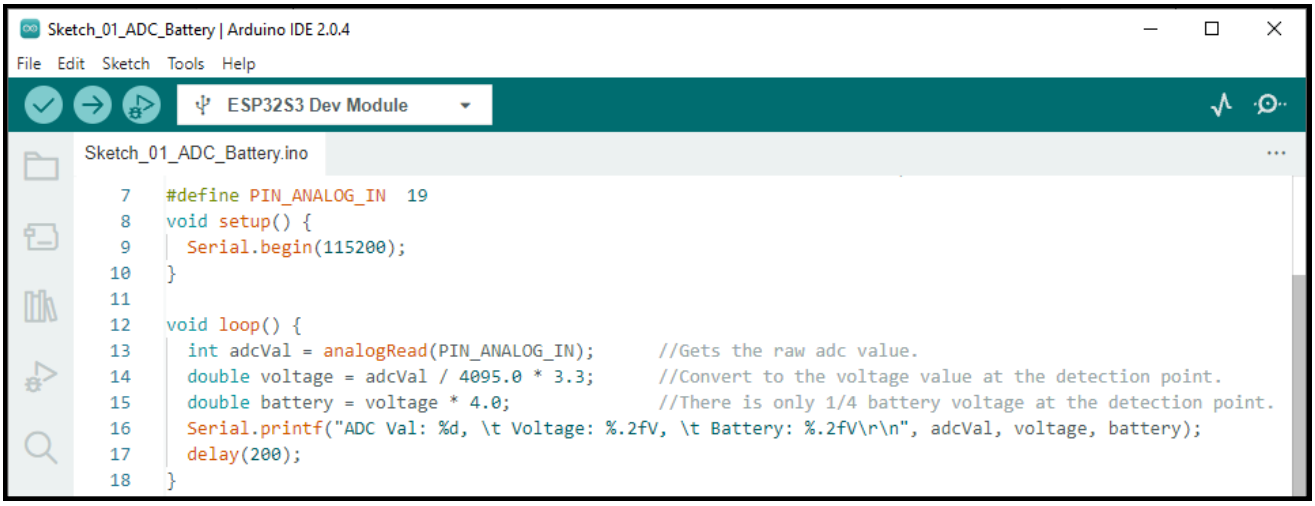


Hardware connection. If you need any support, please feel free to contact us via: support@freenove.com

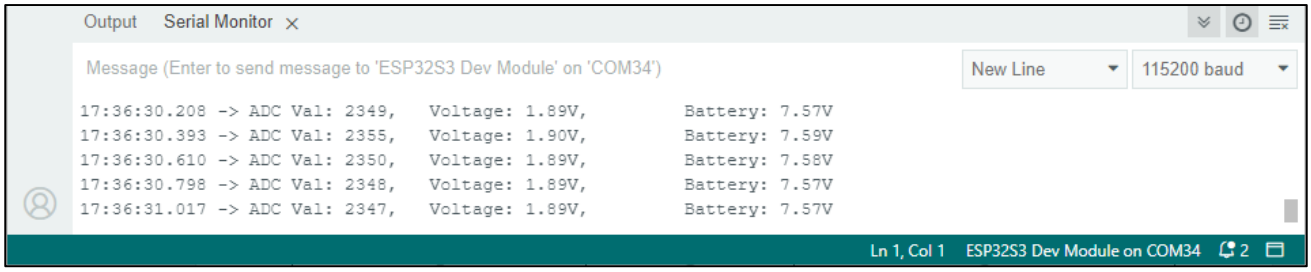


Sketch

Sketch_01_ADC_Battery



Download the code to ESP32-S3 WROOM, open the serial monitor, and set the baud rate to 115200. Turn on the power switch, as shown in the following figure.



When there is no external power supply connected, the voltage measured on GPIO19 pin is around 4V, which is supplied to V-External through the circuitry on the ESP32-S3 WROOM Shield from the USB power supply. This is a normal phenomenon.

The following is the code:

```
#define PIN_ANALOG_IN  19
void setup() {
  Serial.begin(115200);
}

void loop() {
  int adcVal = analogRead(PIN_ANALOG_IN);//Gets the raw adc value.
  double voltage = adcVal/4095.0*3.3;//Convert to the voltage value at the detection point.
  double battery = voltage * 4.0;//There is only 1/4 battery voltage at the detection point.
  Serial.printf("ADC Val: %d, \t Voltage: %.2fV, \t Battery: %.2fV\r\n", adcVal, voltage, battery);
  delay(200);
}
```

In loop(), use the analogRead() function to obtain the input ADC value of the potentiometer, calculate the voltage value of the Power according to the formula in the previous knowledge point, and print it out through the serial port.

```
7    int adcVal = analogRead(PIN_ANALOG_IN);//Gets the raw adc value.
8    double voltage = adcVal/4095.0*3.3;//Convert to the voltage value at the detection point.
9    double battery = voltage * 4.0;//There is only 1/4 battery voltage at the detection point.
10   Serial.printf("ADC Val: %d, \t Voltage: %.2fV, \t Battery: %.2fV\r\n", adcVal, voltage,
     battery);
```

Reference

uint16_t analogRead(*uint8_t* pin);

Reads the value from the specified analog pin. Return the analog reading on the pin. (0-4095 for 12 bits).

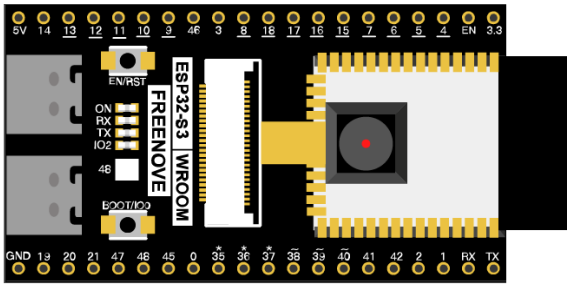
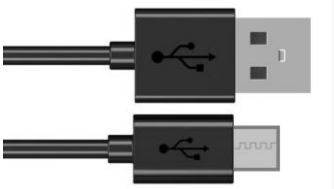
Chapter 2 WS2812

In this chapter, we will learn to use the onboard WS2812. Make the RGB LEDs to emit various color via a pin.

Project 2.1 WS2812

Learn the basic usage of WS2812 and use it to flash red, green, blue and white.

Component List

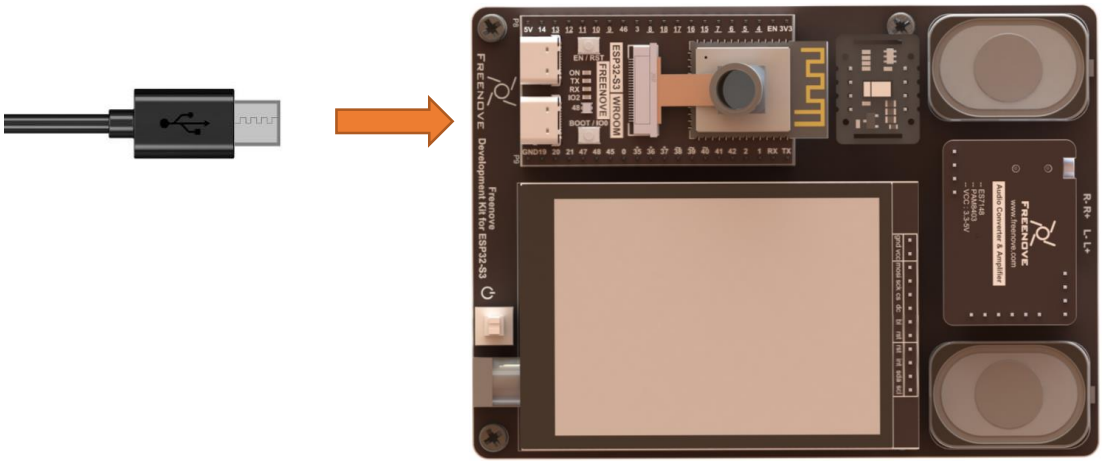
ESP32-S3 WROOM x1	USB cable x1
	

Related knowledge

A WS2812 integrates 3 LEDs of red, green and blue colors, and each LED supports 256 levels of brightness adjustment, which means WS2812 can emit 2^24=16,777,216 different colors.

Circuit

Connect Freenove ESP32-S3 to your computer using the USB cable.

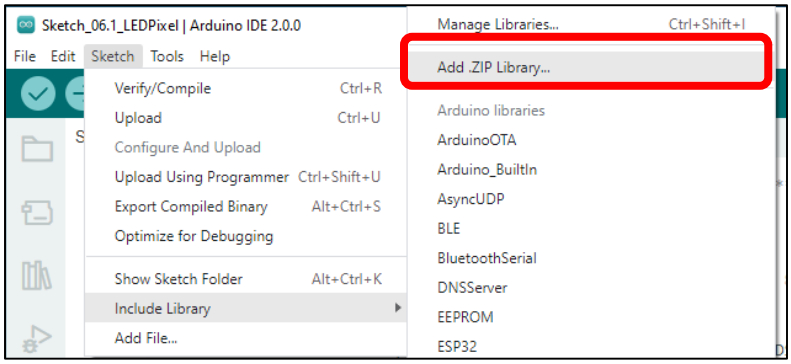


Sketch

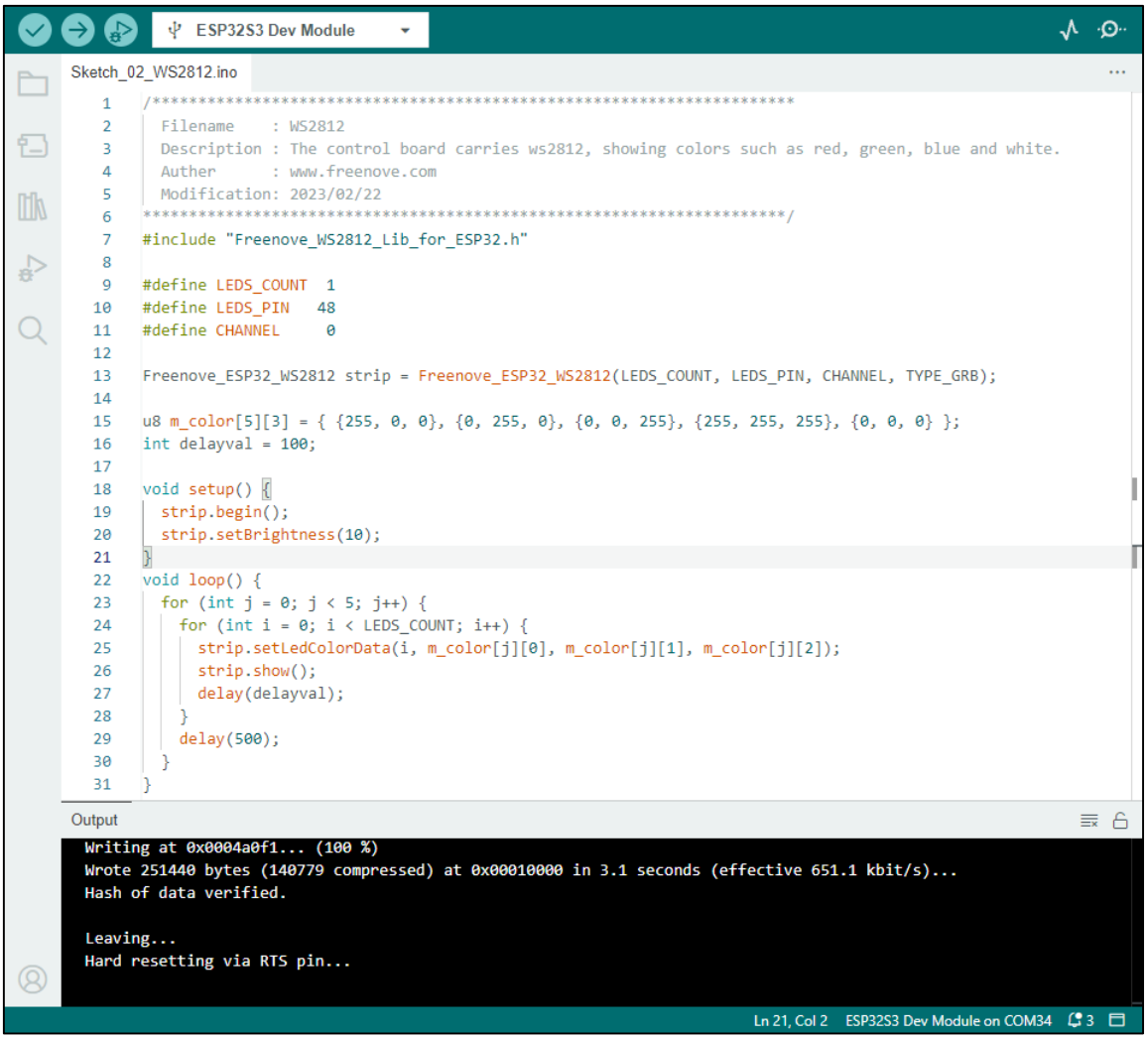
This code uses a library named "Freenove_WS2812_Lib_for_ESP32". If you have not yet installed it, please do so first.

How to install the library

Open Arduino IDE, click Sketch→Include Library→Add .ZIP Library. In the pop-up window, find the file named "./Libraries/Freenove_WS2812_Lib_for_ESP32.Zip" which locates in this directory, and click OPEN.



Sketch_02_WS2812



Download the code to ESP32-S3 WROOM and LED WS2812 begins to light up in red, green, blue, white and bla

The following is the program code:

```
#include "Freenove_WS2812_Lib_for_ESP32.h"

#define LEDS_COUNT  1  // The number of led
#define LEDS_PIN    48 // define the pin connected to the Freenove 8 led strip
#define CHANNEL     0  // RMT channel

Freenove_ESP32_WS2812 strip = Freenove_ESP32_WS2812(LEDS_COUNT, LEDS_PIN, CHANNEL, TYPE_GRB);

u8 m_color[5][3] = { {255, 0, 0}, {0, 255, 0}, {0, 0, 255}, {255, 255, 255}, {0, 0, 0} };
int delayval = 100;

void setup() {
    strip.begin();
    strip.setBrightness(10);
}
void loop() {
    for (int j = 0; j < 5; j++) {
        for (int i = 0; i < LEDS_COUNT; i++) {
            strip.setLedColorData(i, m_color[j][0], m_color[j][1], m_color[j][2]);
            strip.show();
            delay(delayval);
        }
        delay(500);
    }
}
```

To use some libraries, first you need to include their header file.

```
#include "Freenove_WS2812_Lib_for_ESP32.h"
```

Define the pins connected to the ws2812, the number of LED,s and RMT channel values.

```
#define LEDS_COUNT  1  // The number of led
#define LEDS_PIN    48 // define the pin connected to the Freenove 8 led strip
#define CHANNEL     0  // RMT channel
```

Use the above parameters to create a WS2812 object strip.

```
Freenove_ESP32_WS2812 strip = Freenove_ESP32_WS2812(LEDS_COUNT, LEDS_PIN, CHANNEL, TYPE_GRB);
```

Define the color values to be used, such as red, green, blue, white, and black.

```
u8 m_color[5][3] = { {255, 0, 0}, {0, 255, 0}, {0, 0, 255}, {255, 255, 255}, {0, 0, 0} };
```

Define a variable to set the time interval for each led to light up. The smaller the value is, the faster it will light up.

```
int delayval = 50;
```

Initialize strip() in setup() and set the brightness.

```
  strip.begin();
  strip.setBrightness(10);
```

In the loop(), there are two "for" loops, the internal for loop is to light the LED one by one, and the external one to switch colors. strip.setLedColorData() is used to set the color, but it does not change immediately. Only when strip.show() is called will the color data be sent to the LED to change the color.

```
17      for (int j = 0; j < 5; j++) {
18          for (int i = 0; i < LEDS_COUNT; i++) {
19              strip.setLedColorData(i, m_color[j][0], m_color[j][1], m_color[j][2]);
20              strip.show();
21              delay(delayval);
22          }
23          delay(500);
24      }
```

Reference

`Freenove_ESP32_WS2812(u16 n = 8, u8 pin_gpio = 2, u8 chn = 0, LED_TYPE t = TYPE_GRB)`

Constructor to create a ws2812 object.

Before each use of the constructor, please add "**#include "Freenove_WS2812_Lib_for_ESP32.h**"

Parameters

n: The number of led.

pin_gpio: A pin connected to an led.

Chn: RMT channel, which has eight channels, 0-7, and uses channel by default. This means that you can use eight ws2812 modules for the display at the same time, and these modules do not interfere with each other

t: Types of LED.

TYPE_RGB: The sequence of ws2812 module loading color is red, green and blue.
TYPE_RBG: The sequence of ws2812 module loading color is red, blue and green.
TYPE_GRB: The sequence of ws2812 module loading color is green, red and blue.
TYPE_GBR: The sequence of ws2812 module loading color is green, blue and red.
TYPE_BRG: The sequence of ws2812 module loading color is blue, red and green.
TYPE_BGR: The sequence of ws2812 module loading color is blue, green and red.

`void begin(void);`

Initialize the LEDPixel object

```
void setLedColorData (u8 index, u8 r, u8 g, u8 b);
void setLedColorData (u8 index, u32 rgb);
void setLedColor (u8 index, u8 r, u8 g, u8 b);
void setLedColor (u8 index, u32 rgb);
```

Set the color of led with order number n.

`void show(void);`

Send the color data to the led and display the set color immediately.

`void setBrightness(uint8_t);`

Set the brightness of the LED.

If you want to learn more about this library, you can visit the following website:
https://github.com/Freenove/Freenove_WS2812_Lib_for_ESP32

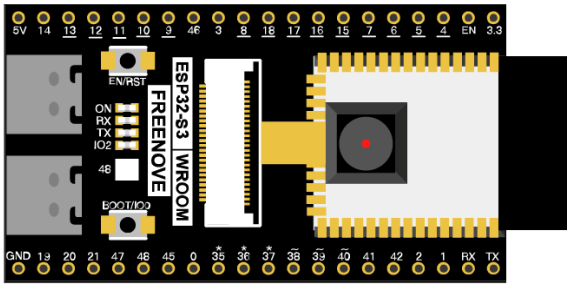
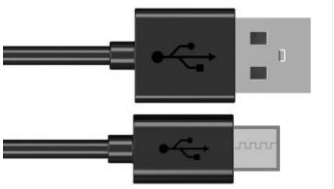
Chapter 3 Camera Web Server

In this section, we take ESP32-S3's video function as an example to study.

Project 3.1 Camera Web Server

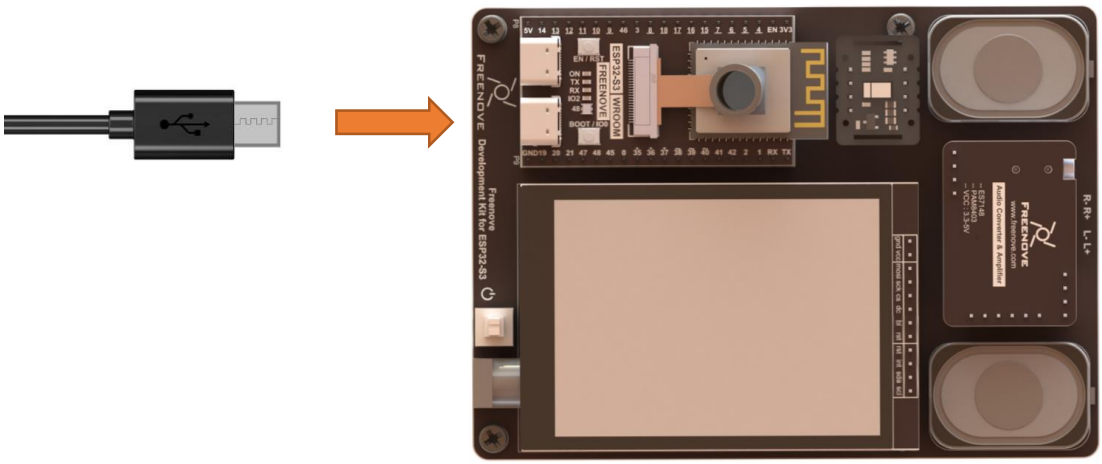
Connect ESP32-S3 using USB and check its IP address through serial monitor. Use web page to access IP address to obtain video and image data.

Component List

ESP32-S3 WROOM x1 	USB cable x1 

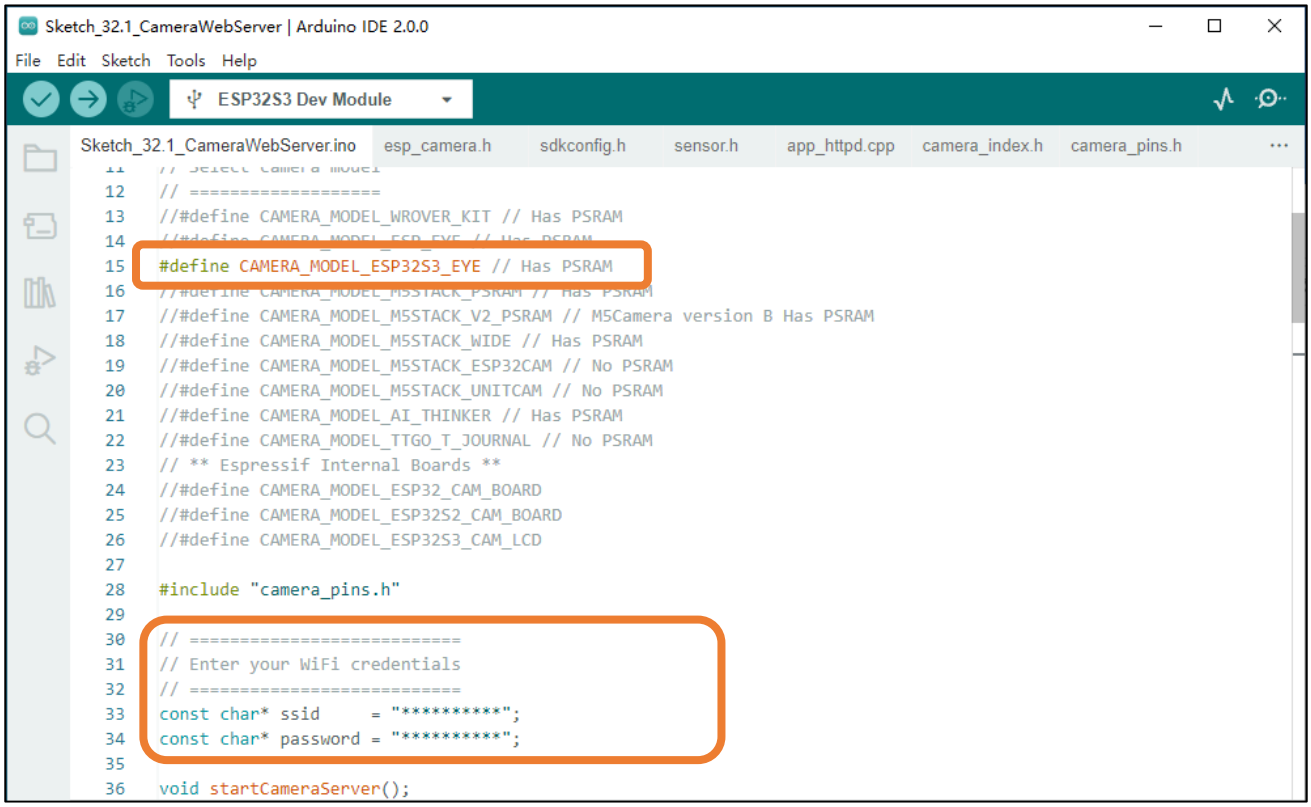
Circuit

Connect Freenove ESP32-S3 to the computer using the USB cable.



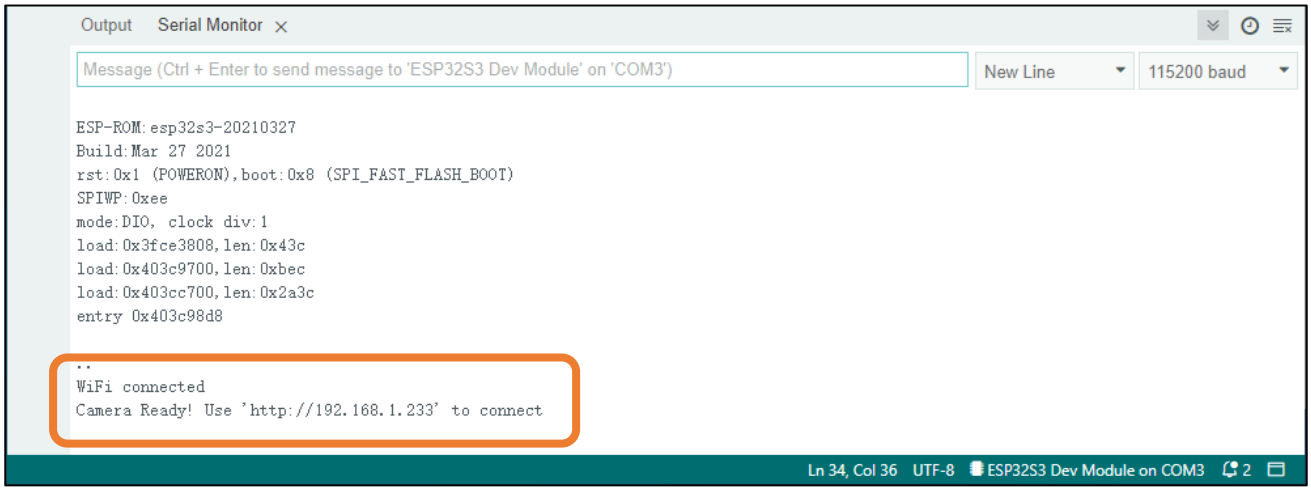
Sketch

Sketch_03_CameraWebServer



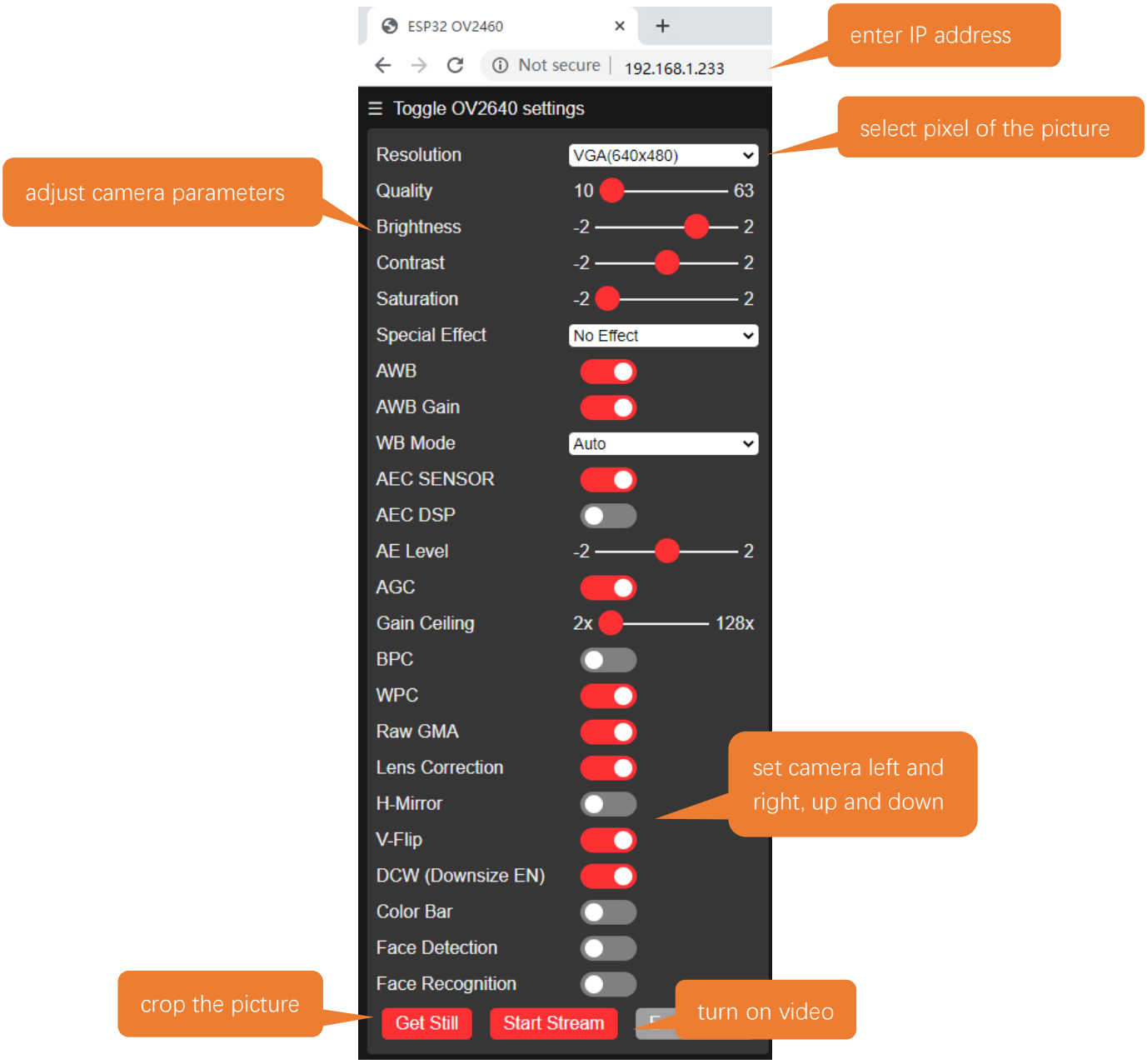
Before running the program, please modify your router's name and password marked in the illustration above to make sure that your Sketch can compile and work successfully.

Compile and upload codes to ESP32-S3, open the serial monitor and set the baud rate to 115200, and the serial monitor will print out a network address.

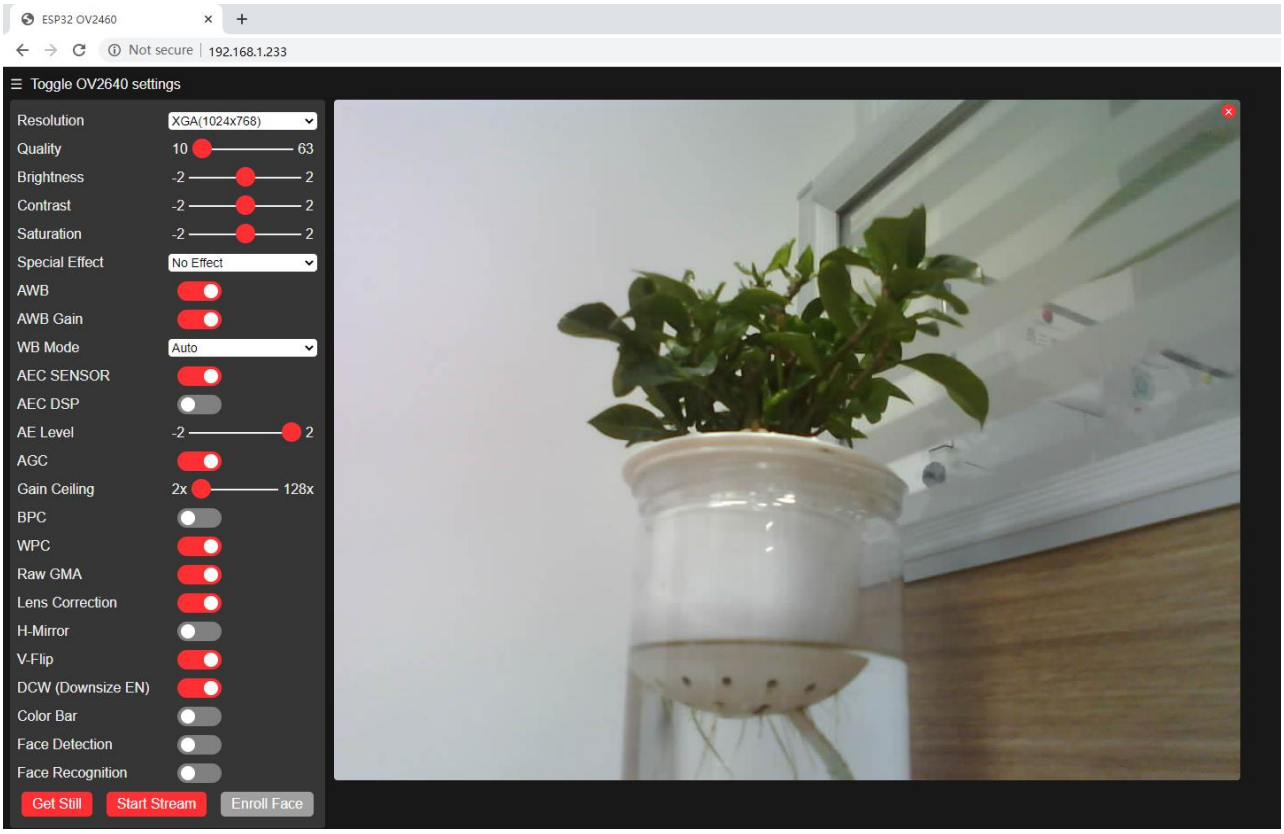


If your ESP32-S3 is in the process of connecting to router (dots are printed continuousl), but the IP address does not show up, please re-check whether the router name and password have been entered correctly and press the reset key on ESP32-S3 WROOM to wait for a successful connection prompt.

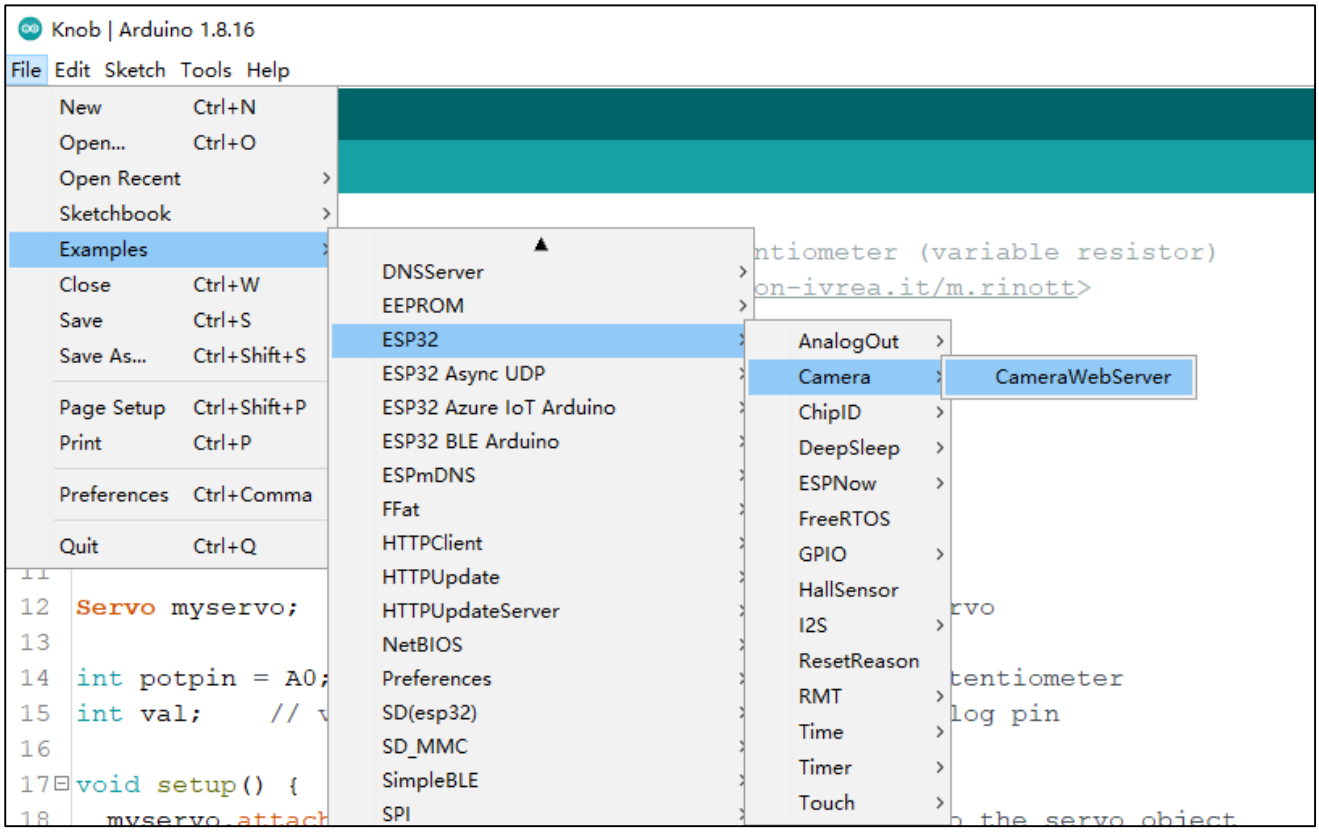
Open a web browser, enter the IP address printed by the serial monitor in the address bar, and access it. Taking the Google browser as an example, here is what the browser prints out after successful access to ESP32-S3's IP.



Click on Start Stream. The result is as shown below.



Note: If sketch compilation fails due to ESP32-S3 support package, try to run the example sketch as per the image below. This sketch is the same as the one described in the tutorial above.



The following is the main program code. You need include other code files in the same folder when write your own code.

```
#include "esp_camera.h"
#include <WiFi.h>

// ===================
// Select camera model
// ===================
//#define CAMERA_MODEL_WROVER_KIT // Has PSRAM
//#define CAMERA_MODEL_ESP_EYE // Has PSRAM
#define CAMERA_MODEL_ESP32S3_EYE // Has PSRAM
//#define CAMERA_MODEL_M5STACK_PSRAM // Has PSRAM
//#define CAMERA_MODEL_M5STACK_V2_PSRAM // M5Camera version B Has PSRAM
//#define CAMERA_MODEL_M5STACK_WIDE // Has PSRAM
//#define CAMERA_MODEL_M5STACK_ESP32CAM // No PSRAM
//#define CAMERA_MODEL_M5STACK_UNITCAM // No PSRAM
//#define CAMERA_MODEL_AI_THINKER // Has PSRAM
//#define CAMERA_MODEL_TTGO_T_JOURNAL // No PSRAM
// ** Espressif Internal Boards **
//#define CAMERA_MODEL_ESP32_CAM_BOARD
//#define CAMERA_MODEL_ESP32S2_CAM_BOARD
//#define CAMERA_MODEL_ESP32S3_CAM_LCD

#include "camera_pins.h"

// ===========================
// Enter your WiFi credentials
// ===========================
const char* ssid     = "********";
const char* password = "********";

void startCameraServer();

void setup() {
  Serial.begin(115200);
  Serial.setDebugOutput(true);
  Serial.println();

  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
```

```
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.frame_size = FRAMESIZE_SXGA;
  config.pixel_format = PIXFORMAT_JPEG; // for streaming
  config.grab_mode = CAMERA_GRAB_WHEN_EMPTY;
  config.fb_location = CAMERA_FB_IN_PSRAM;
  config.jpeg_quality = 12;
  config.fb_count = 1;

  // if PSRAM IC present, init with UXGA resolution and higher JPEG quality
  // for larger pre-allocated frame buffer.
  if(psramFound()){
    config.jpeg_quality = 10;
    config.fb_count = 2;
    config.grab_mode = CAMERA_GRAB_LATEST;
  } else {
    // Limit the frame size when PSRAM is not available
    config.frame_size = FRAMESIZE_HVGA;
    config.fb_location = CAMERA_FB_IN_DRAM;
  }

  // camera init
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;
  }

  sensor_t * s = esp_camera_sensor_get();
  // initial sensors are flipped vertically and colors are a bit saturated
  s->set_vflip(s, 1); // flip it back
  s->set_brightness(s, 1); // up the brightness just a bit
```

```
87    s->set_saturation(s, -1); // lower the saturation
88
89    WiFi.begin(ssid, password);
90    WiFi.setSleep(false);
91
92    while (WiFi.status() != WL_CONNECTED) {
93      delay(500);
94      Serial.print(".");
95    }
96    Serial.println("");
97    Serial.println("WiFi connected");
98
99    startCameraServer();
100
101   Serial.print("Camera Ready! Use 'http://");
102   Serial.print(WiFi.localIP());
103   Serial.println("' to connect");
104 }
105
106 void loop() {
107   // Do nothing. Everything is done in another task by the web server
108   delay(10000);
109 }
```

Add procedure files and API interface files related to ESP32-S3 camera.

```
1   #include "esp_camera.h"
2   #include <WiFi.h>
…
9   #define CAMERA_MODEL_ESP32S3_EYE // Has PSRAM
…
11  #include "camera_pins.h"
```

Enter the name and password of your router.

```
13  const char *ssid     = "********";  //input your wifi name
14  const char *password = "********";  //input your wifi passwords
```

Initialize serial port, set baud rate to 115200; open the debug and output function of the serial.

```
21    Serial.begin(115200);
22    Serial.setDebugOutput(true);
23    Serial.println();
```

Configure parameters including interface pins of the camera. Note: It is not recommend changing them.

```
37    camera_config_t config;
38    config.ledc_channel = LEDC_CHANNEL_0;
39    config.ledc_timer = LEDC_TIMER_0;
40    config.pin_d0 = Y2_GPIO_NUM;
41    config.pin_d1 = Y3_GPIO_NUM;
42    config.pin_d2 = Y4_GPIO_NUM;
```

```
43    config.pin_d3 = Y5_GPIO_NUM;
44    config.pin_d4 = Y6_GPIO_NUM;
45    config.pin_d5 = Y7_GPIO_NUM;
46    config.pin_d6 = Y8_GPIO_NUM;
47    config.pin_d7 = Y9_GPIO_NUM;
48    config.pin_xclk = XCLK_GPIO_NUM;
49    config.pin_pclk = PCLK_GPIO_NUM;
50    config.pin_vsync = VSYNC_GPIO_NUM;
51    config.pin_href = HREF_GPIO_NUM;
52    config.pin_sscb_sda = SIOD_GPIO_NUM;
53    config.pin_sscb_scl = SIOC_GPIO_NUM;
54    config.pin_pwdn = PWDN_GPIO_NUM;
55    config.pin_reset = RESET_GPIO_NUM;
56    config.xclk_freq_hz = 20000000;
57    config.frame_size = FRAMESIZE_SXGA;
58    config.pixel_format = PIXFORMAT_JPEG; // for streaming
59    config.grab_mode = CAMERA_GRAB_WHEN_EMPTY;
60    config.fb_location = CAMERA_FB_IN_PSRAM;
61    config.jpeg_quality = 12;
62    config.fb_count = 1;
```

ESP32-S3 connects to the router and prints a successful connection prompt. If it does not connect succssfully, press the reset key on the ESP32-S3 WROOM.

```
89    WiFi.begin(ssid, password);
90    WiFi.setSleep(false);
91
92    while (WiFi.status() != WL_CONNECTED) {
93      delay(500);
94      Serial.print(".");
95    }
96    Serial.println("");
97    Serial.println("WiFi connected");
```

Open the video streams server function of the camera and print its IP address via serial port.

```
99    startCameraServer();
100
101   Serial.print("Camera Ready! Use 'http://");
102   Serial.print(WiFi.localIP());
103   Serial.println("' to connect");
```

Configure the display image information of the camera.

The set_vflip() function sets whether the image is flipped 180°, with 0 for no flip and 1 for flip 180°.

The set_brightness() function sets the brightness of the image, with values ranging from -2 to 2.

The set_saturation() function sets the color saturation of the image, with values ranging from -2 to 2.

```
36    sensor_t * s = esp_camera_sensor_get();
37    s->set_vflip(s, 1);        //flip it back
38    s->set_brightness(s, 1);   //up the blightness just a bit
39    s->set_saturation(s, -1);  //lower the saturation
```

Modify the resolution and sharpness of the images captured by the camera. The sharpness ranges from 10 to 63, and the smaller the number, the sharper the picture. The larger the number, the blurrier the picture. Please refer to the table below.

```
    config.frame_size = FRAMESIZE_VGA;
    config.jpeg_quality = 10;
```

Reference

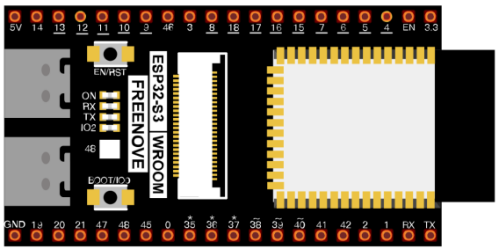
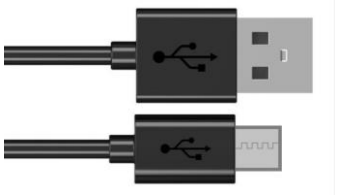
Image resolution	Sharpness	Image resolution	Sharpness
FRAMESIZE_96X96	96x96	FRAMESIZE_HVGA	480x320
FRAMESIZE_QQVGA	160x120	FRAMESIZE_VGA	640x480
FRAMESIZE_QCIF	176x144	FRAMESIZE_SVGA	800x600
FRAMESIZE_HQVGA	240x176	FRAMESIZE_XGA	1024x768
FRAMESIZE_240X240	240x240	FRAMESIZE_HD	1280x720
FRAMESIZE_QVGA	320x240	FRAMESIZE_SXGA	1280x1024
FRAMESIZE_CIF	400x296	FRAMESIZE_UXGA	1600x1200

Chapter 4 Read and Write the SDcard

An SDcard slot is integrated on the back of the ESP32-S3 WROOM. In this chapter, we learn how to use ESP32-S3 to read and write SDcard.

Project 4.1 SDMMC Test

Component List

ESP32-S3 WROOM x1 	USB cable x1 
SD card x1 	Card reader x1 (random color) 

Component knowledge

SD card read and write method

The ESP32-S3 offers two methods for accessing the SD card: SPI interface and SDMMC interface. The SPI mode requires 4 IOs to access the SD card, while the SDMMC interface supports both one-bit and four-bit bus modes. In the one-bit bus mode of SDMMC, only 3 IOs are required to access the SD card; while, the four-bit bus mode of SDMMC requires 6 IOs to access the SD card.

The above three methods can all be used to access the SD card, the difference of which lies in the speed. When accessing an SD card, the fastest reading and writing speed is achieved under the four-bit bus mode of SDMMC. The one-bit bus mode of SDMMC offers a slightly slower access speed of around 80% compared to the four-bit bus mode. The slowest access speed is observed with the SPI mode, which offers only around 50% of the speed of the four-bit bus mode of SDMMC.

For most applications, we recommend using the one-bit bus mode because it requires the least number of IO pins while still providing good performance and speed.

Format SD card

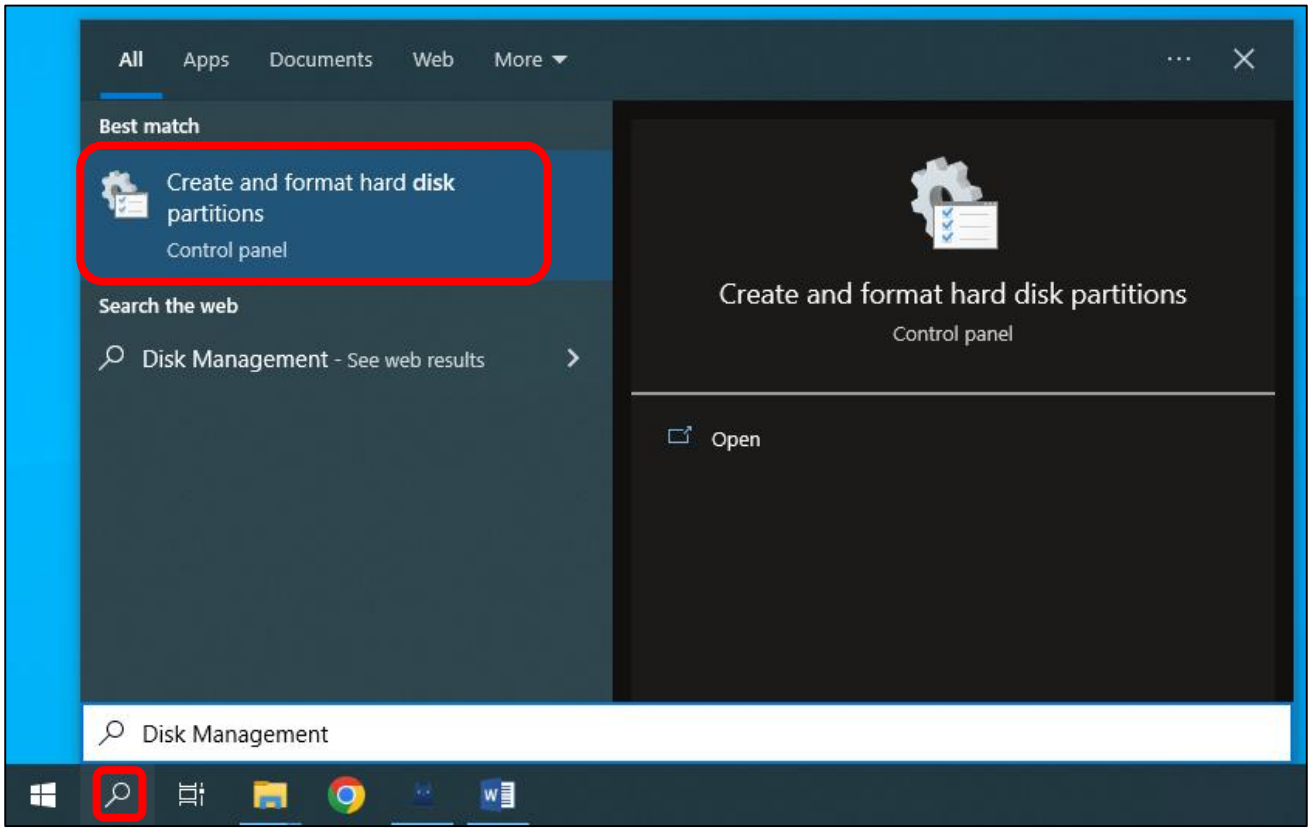
To begin the project, it is essential to prepare a blank SD card and a card reader. Before proceeding, we need to create a drive letter for the SD card and format it. The following steps provide a guide on how to accomplish this on different computer systems. Please follow the guide that corresponds to your computer.

Note that this step requires a card reader and a blank SD card. Please ensure that you have these components available before proceeding.

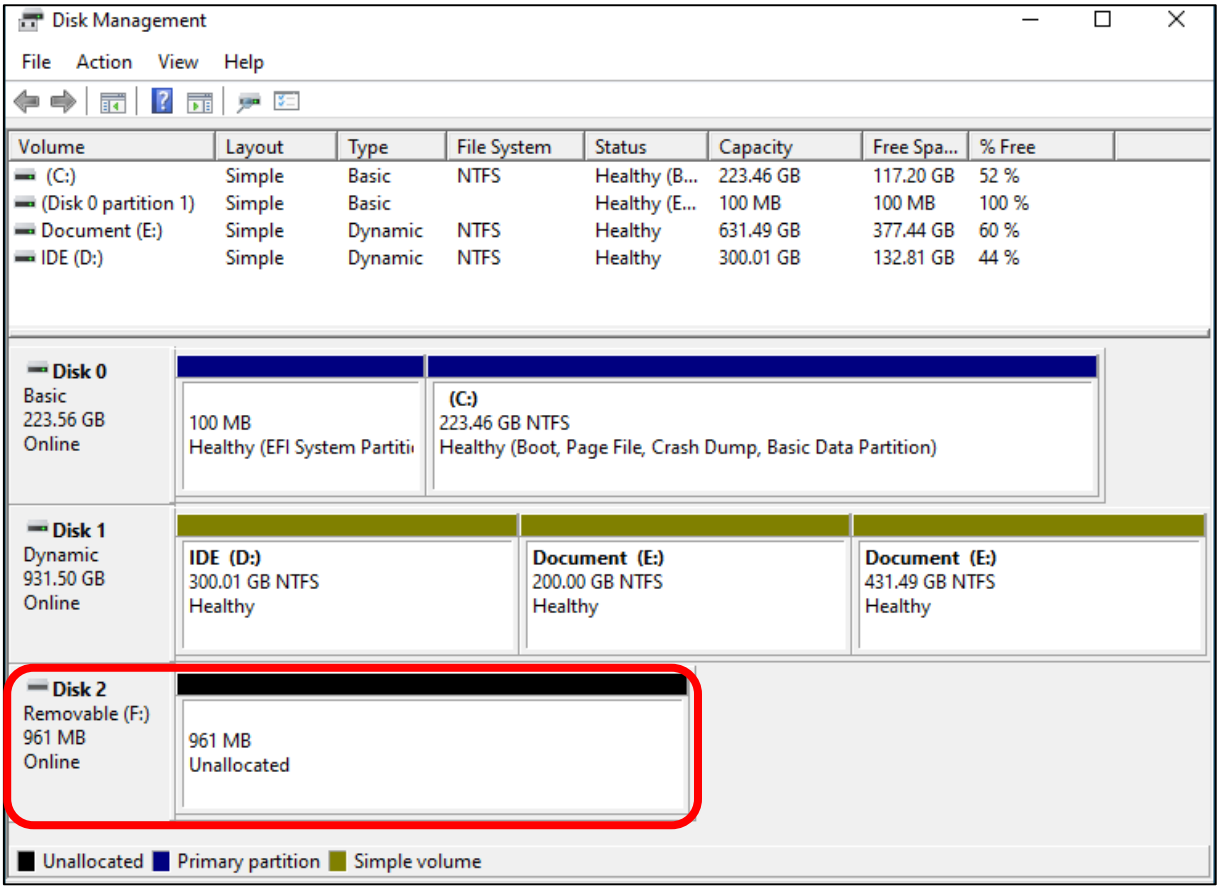
Windows

Insert the SD card into the card reader, and then insert the card reader into the computer.

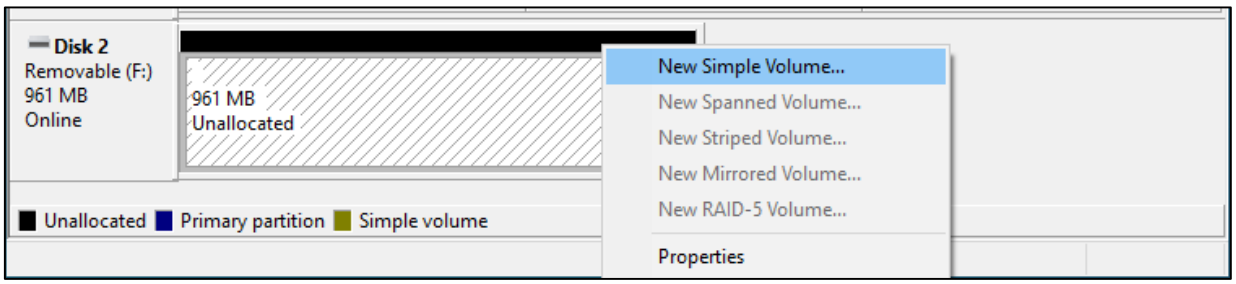
In the Windows search box, enter "Disk Management" and select "Create and format hard disk partitions".



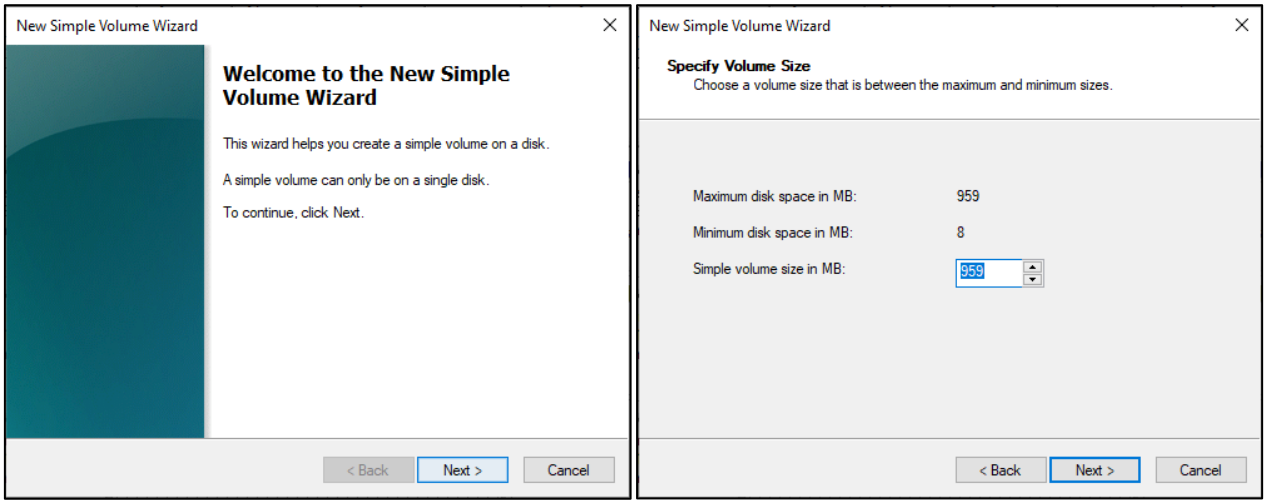
In the newly opened window, locate an unallocated volume with a size close to 1GB.



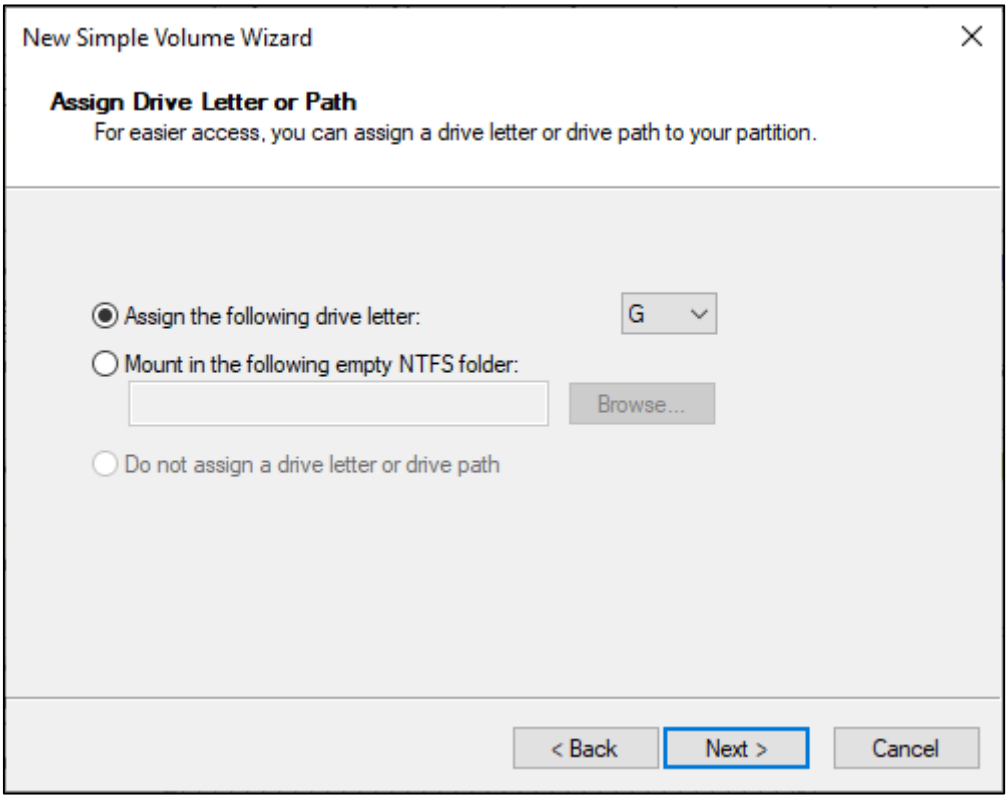
Click to select the volume, right-click and select "New Simple Volume".



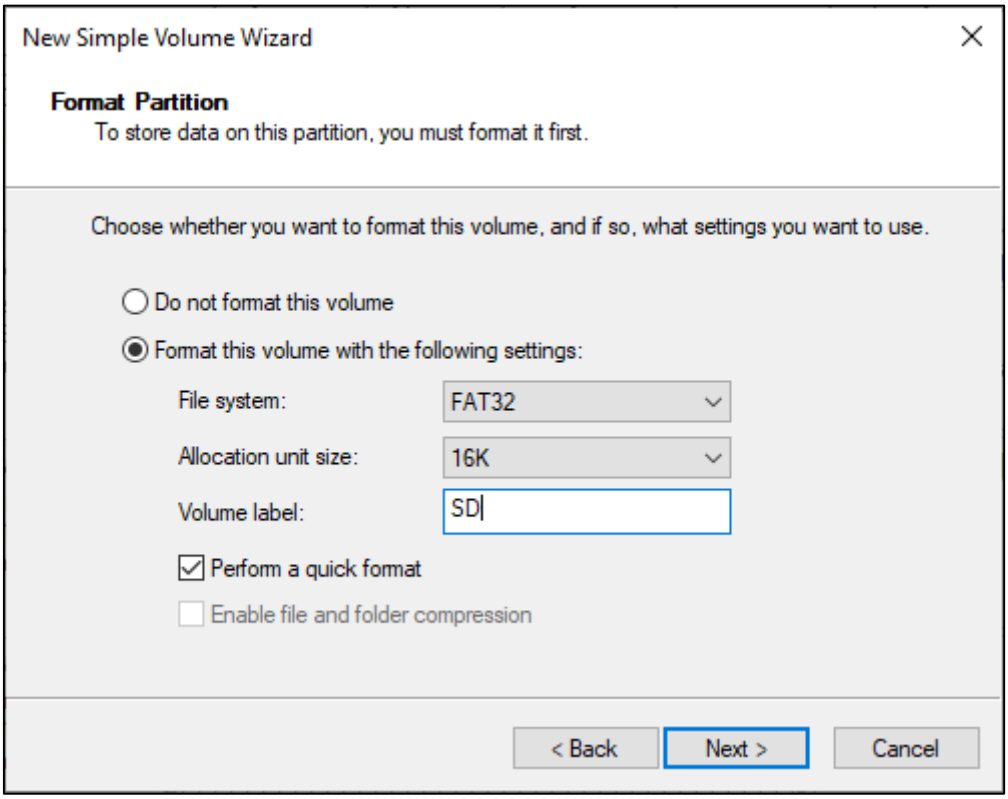
Click Next.



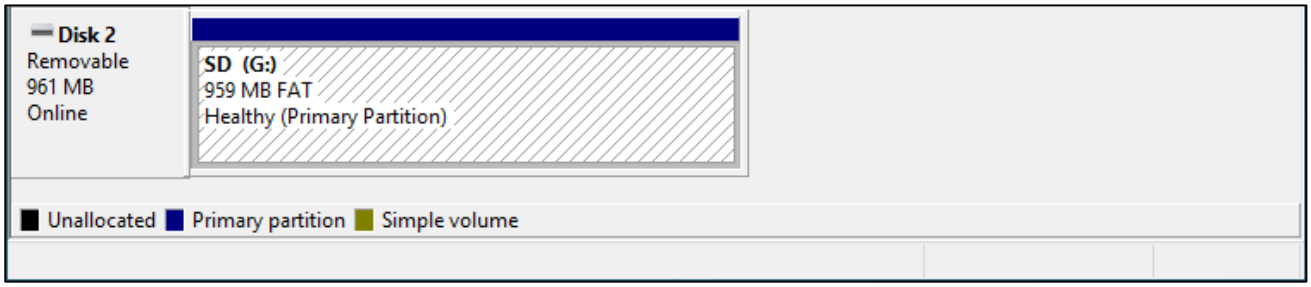
On the right-hand side, you can select a preferred drive letter for the SD card or simply proceed with the default setting by clicking on the "Next" button.



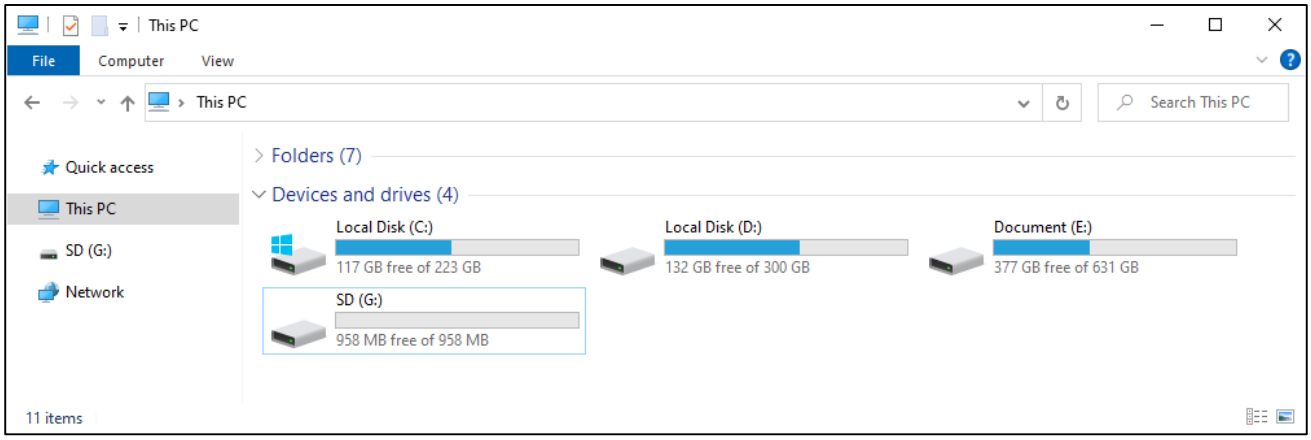
When formatting the SD card, select the file system as FAT (or FAT32) and set the allocation unit size to 16K. You can set the volume label to any name of your choice. Once you have made these selections, click on the "Next" button to proceed.



Click Finish. Wait for the SD card initialization to complete.

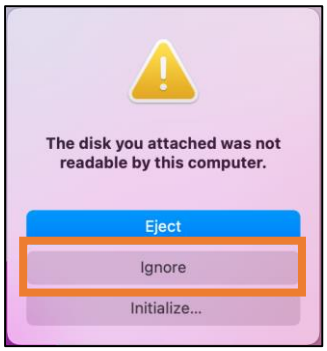


After completing the formatting process, you should be able to see the SD card in the "This PC" section of your computer.

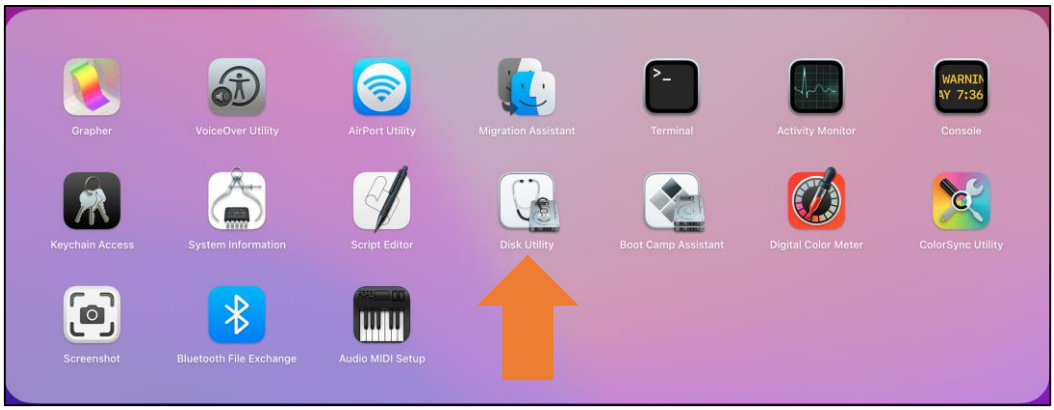


MAC

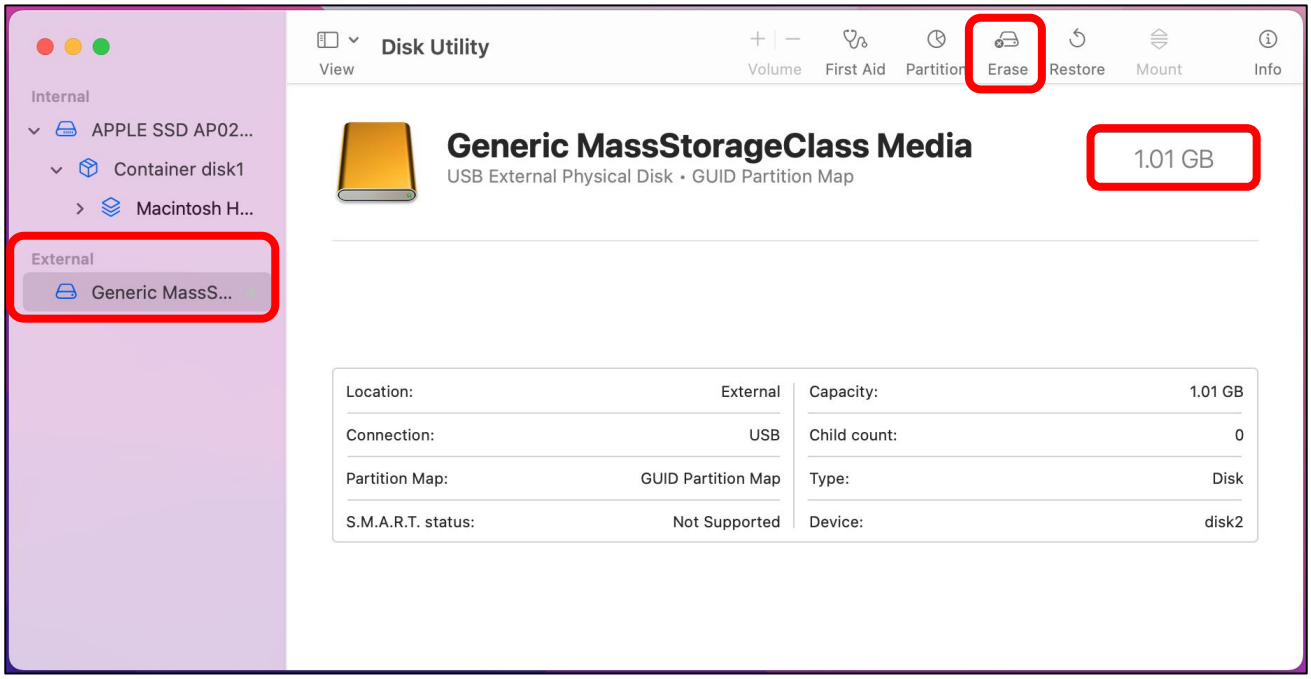
Insert the SD card into the card reader and then insert the card reader into your computer. Some computers may display a prompt with the following message. In this case, please click on the "Ignore" option to proceed.



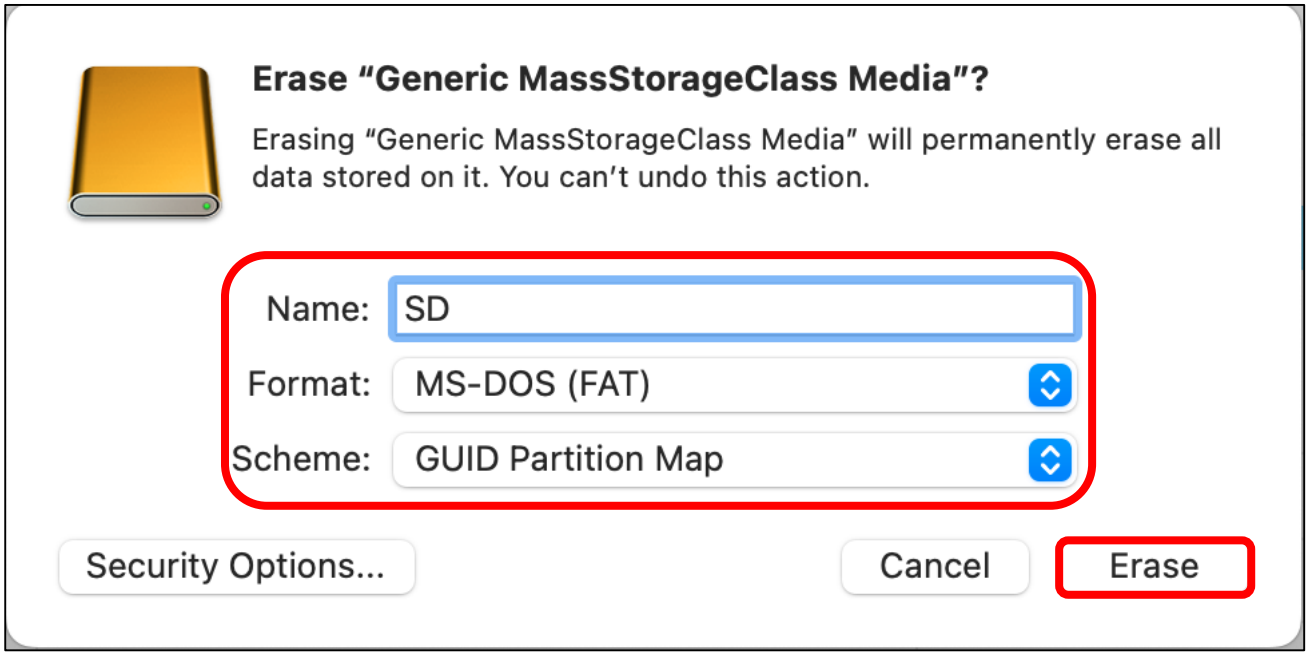
Find "Disk Utility" in the MAC system and click to open it.



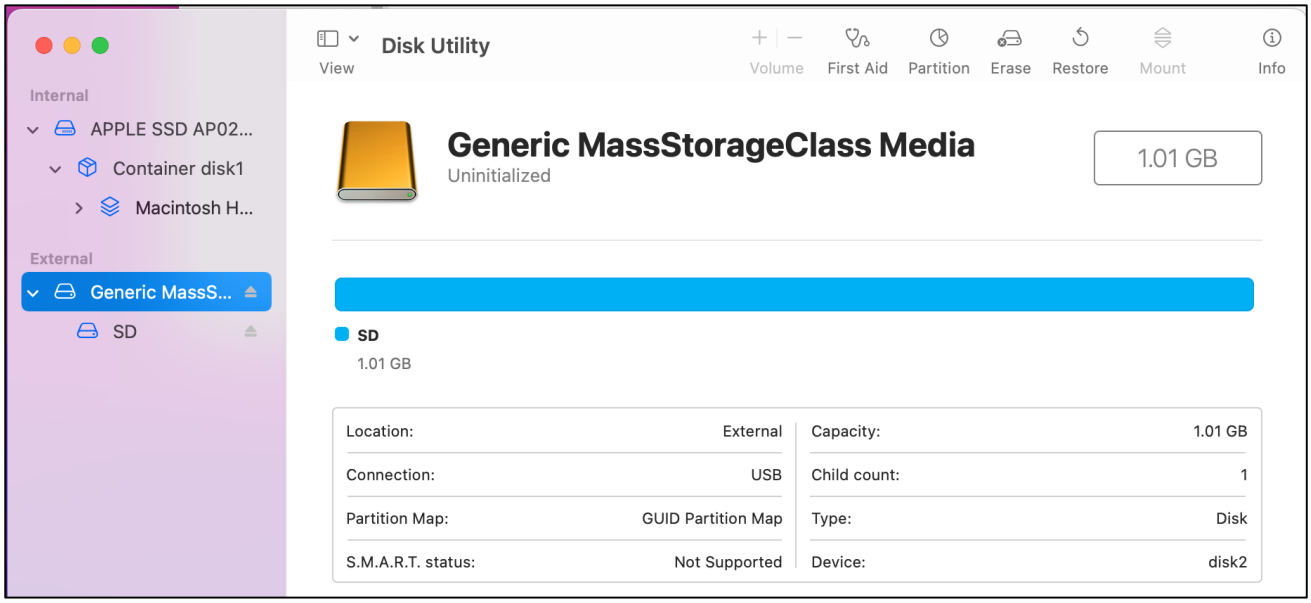
Select "Generic MassStorageClass Media", note that its size is about 1G. It is important to select the correct item to avoid accidentally erasing the wrong device. Once you have verified that you have selected the correct device, click on the "Erase" button to proceed with erasing the SD card.



Select the configuration as shown in the figure below, and then click "Erase".

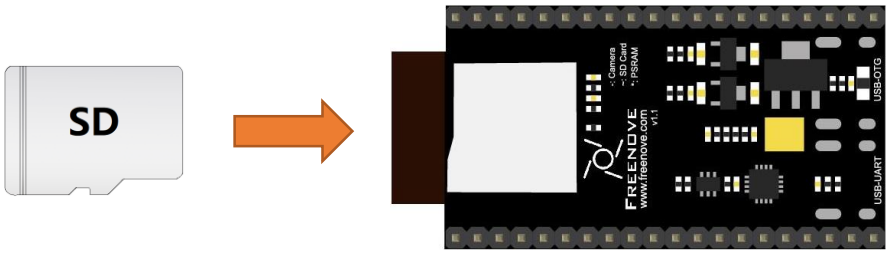


Please wait for the formatting process to complete. Once it is finished, the interface should resemble the image below. You should now be able to see a new disk named "SD" on your desktop.

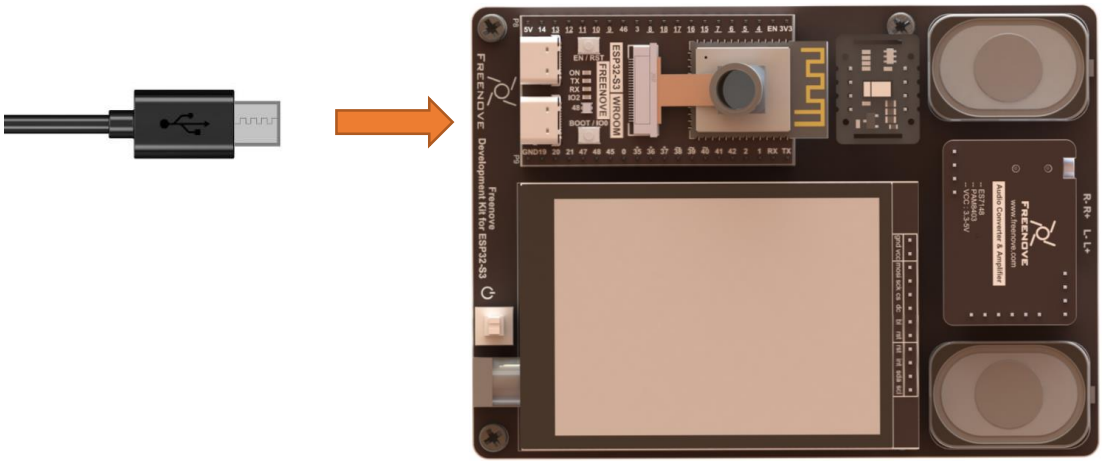


Circuit

Before connecting the USB cable, insert the SD card into the SD card slot on the back of the ESP32-S3.

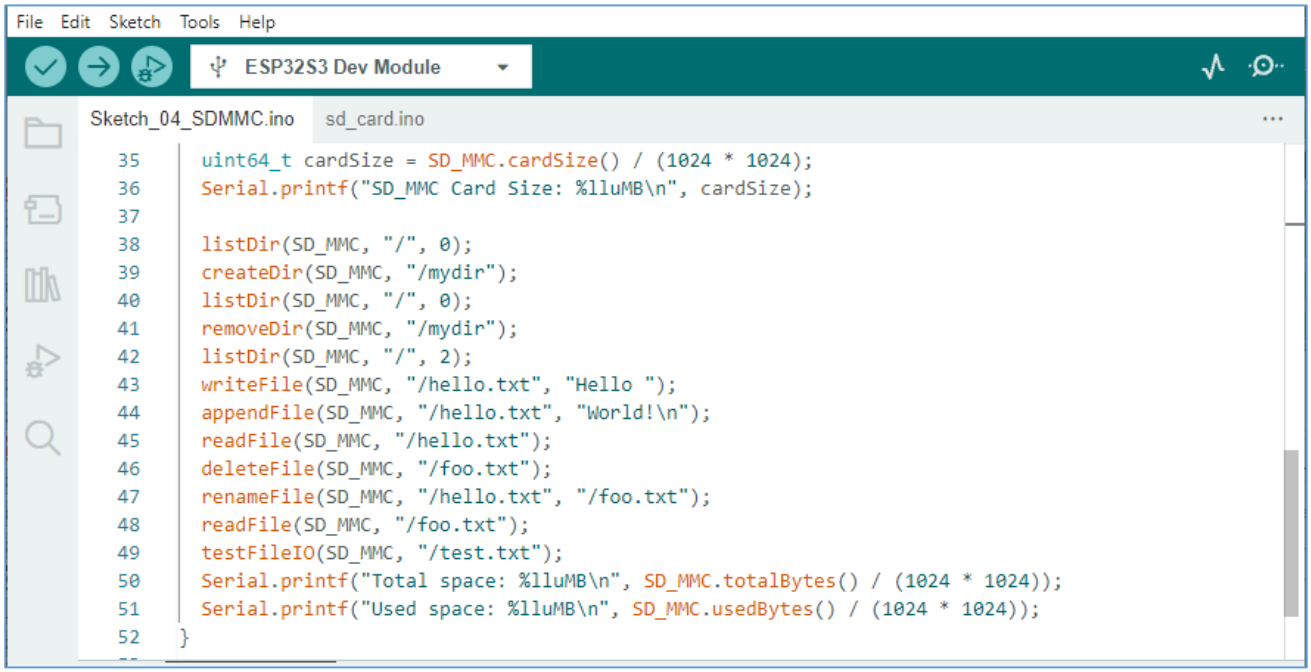


Connect Freenove ESP32-S3 to the computer using the USB cable.



Sketch

Sketch_04_SDMMC



Compile and upload the code to ESP32-S3-WROOM, open the serial monitor, and press the RST button on the board.

You can see the printout as shown below.

```
Listing directory: /
  DIR : System Volume Information
  FILE: test.txt  SIZE: 1048576
  FILE: foo.txt  SIZE: 13
  DIR : mydir
Removing Dir: /mydir
Dir removed
Listing directory: /
  DIR : System Volume Information
Listing directory: /System Volume Information
  FILE: WPSettings.dat  SIZE: 12
  FILE: IndexerVolumeGuid  SIZE: 76
  FILE: test.txt  SIZE: 1048576
  FILE: foo.txt  SIZE: 13
Writing file: /hello.txt
File written
Appending to file: /hello.txt
Message appended
Reading file: /hello.txt
Read from file: Hello World!
Deleting file: /foo.txt
File deleted
Renaming file /hello.txt to /foo.txt
File renamed
Reading file: /foo.txt
Read from file: Hello World!
1048576 bytes read for 549 ms
1048576 bytes written for 1172 ms
Total space: 960MB
Used space: 1MB
```

The following is the program code:

```
#include "FS.h"
#include "SD_MMC.h"

#define SD_MMC_CMD 38  //Please do not modify it.
#define SD_MMC_CLK 39  //Please do not modify it.
#define SD_MMC_D0 40   //Please do not modify it.

void setup() {
  Serial.begin(115200);
  SD_MMC.setPins(SD_MMC_CLK, SD_MMC_CMD, SD_MMC_D0);
  int error = SD_MMC.begin("/sdcard", true, true, SDMMC_FREQ_DEFAULT, 5);
  if (!error) {
    Serial.printf("Card Mount Failed: %d\r\n", error);
    return;
  }
  Serial.println("Card Mount Success.");
  uint8_t cardType = SD_MMC.cardType();

  if (cardType == CARD_NONE) {
    Serial.println("No SD_MMC card attached");
    return;
  }

  Serial.print("SD_MMC Card Type: ");
  if (cardType == CARD_MMC) {
    Serial.println("MMC");
  } else if (cardType == CARD_SD) {
    Serial.println("SDSC");
  } else if (cardType == CARD_SDHC) {
    Serial.println("SDHC");
  } else {
    Serial.println("UNKNOWN");
  }

  uint64_t cardSize = SD_MMC.cardSize() / (1024 * 1024);
  Serial.printf("SD_MMC Card Size: %lluMB\n", cardSize);

  listDir(SD_MMC, "/", 0);
  createDir(SD_MMC, "/mydir");
  listDir(SD_MMC, "/", 0);
  removeDir(SD_MMC, "/mydir");
  listDir(SD_MMC, "/", 2);
  writeFile(SD_MMC, "/hello.txt", "Hello ");
```

```
44    appendFile(SD_MMC, "/hello.txt", "World!\n");
45    readFile(SD_MMC, "/hello.txt");
46    deleteFile(SD_MMC, "/foo.txt");
47    renameFile(SD_MMC, "/hello.txt", "/foo.txt");
48    readFile(SD_MMC, "/foo.txt");
49    testFileIO(SD_MMC, "/test.txt");
50    Serial.printf("Total space: %lluMB\n", SD_MMC.totalBytes() / (1024 * 1024));
51    Serial.printf("Used space: %lluMB\n", SD_MMC.usedBytes() / (1024 * 1024));
52  }
53
54  void loop() {
55
56  }
```

Add the SD card drive header file.

```
1   #include "FS.h"
2   #include "SD_MMC.h"
```

The drive pins of the SD card are pre-defined and should not be modified as they are fixed. Altering the pins may result in errors or malfunctions while accessing the SD card.

```
4   #define SD_MMC_CMD 38 //Please do not modify it.
5   #define SD_MMC_CLK 39 //Please do not modify it.
6   #define SD_MMC_D0  40 //Please do not modify it.
```

Initialize the serial port function. Sets the drive pin for SDMMC one-bit bus mode.

```
9       Serial.begin(115200);
10      SD_MMC.setPins(SD_MMC_CLK, SD_MMC_CMD, SD_MMC_D0);
```

Set the mount point of the SD card, set SDMMC to one-bit bus mode, and set the read and write speed to 20MHz.

```
11    int error = SD_MMC.begin("/sdcard", true, true, SDMMC_FREQ_DEFAULT, 5);
12    if (!error) {
13      Serial.printf("Card Mount Failed: %d\r\n", error);
14      return;
15    }
```

Get the type of SD card and print it out through the serial port.

```
18    uint8_t cardType = SD_MMC.cardType();
19    if (cardType == CARD_NONE) {
20      Serial.println("No SD_MMC card attached");
21      return;
22    }
23
24    Serial.print("SD_MMC Card Type: ");
25    if (cardType == CARD_MMC) {
26      Serial.println("MMC");
27    } else if (cardType == CARD_SD) {
28      Serial.println("SDSC");
29    } else if (cardType == CARD_SDHC) {
```

```
30      Serial.println("SDHC");
31    } else {
32      Serial.println("UNKNOWN");
33    }
```

Call the listDir() function to read the folder and file names in the SD card, and print them out through the serial port. This function can be found in "sd_read_write.cpp".

```
38  listDir(SD_MMC, "/", 0);
```

Call createDir() to create a folder, and call removeDir() to delete a folder.

```
39  createDir(SD_MMC, "/mydir");
40  removeDir(SD_MMC, "/mydir");
```

Call writeFile() to write any content to the txt file. If there is no such file, create this file first.

Call appendFile() to append any content to txt.

Call readFile() to read the content in txt and print it via the serial port.

```
43  writeFile(SD_MMC, "/hello.txt", "Hello ");
44  appendFile(SD_MMC, "/hello.txt", "World!\n");
45  readFile(SD_MMC, "/hello.txt");
```

Call deleteFile() to delete a specified file.

Call renameFile() to copy a file and rename it.

```
46      deleteFile(SD_MMC, "/foo.txt");
47      renameFile(SD_MMC, "/hello.txt", "/foo.txt");
```

Call the testFileIO() function to test the time it takes to read 512 bytes and the time it takes to write 2048*512 bytes of data.

```
49      testFileIO(SD_MMC, "/test.txt");
```

Print the total size and used size of the SD card via the serial port.

```
50      Serial.printf("Total space: %lluMB\r\n", SD_MMC.totalBytes() / (1024 * 1024));
51      Serial.printf("Used space: %lluMB\r\n", SD_MMC.usedBytes() / (1024 * 1024));
```

If you are interesting in the implementation of functions, you can check them out here.

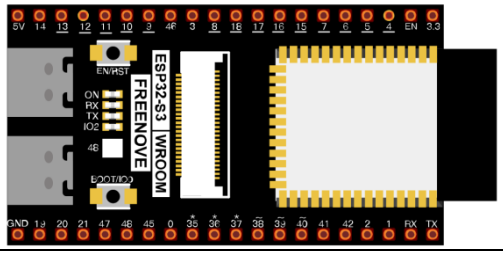
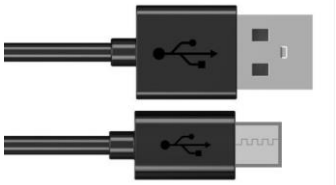
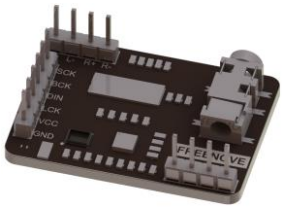
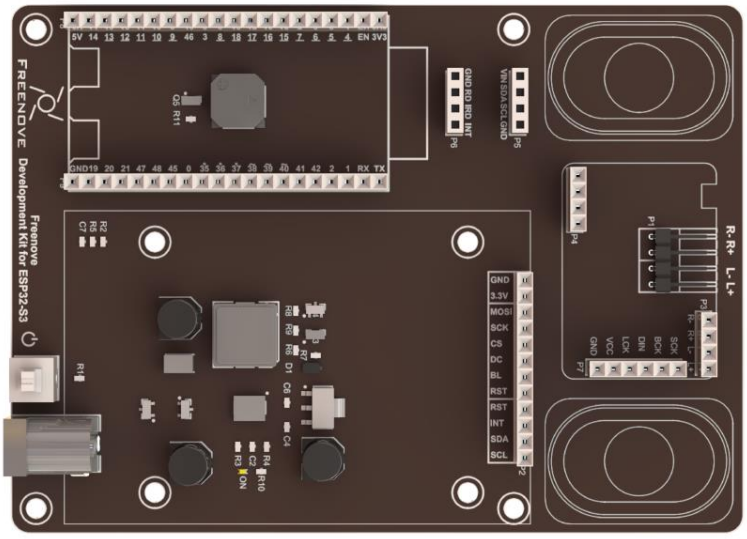
Chapter 5 Play SD card music

In the previous study, we have learned how to use the SD card. Now we are going to learn to play the music in the SD card.

Project 5.1 SDMMC Music

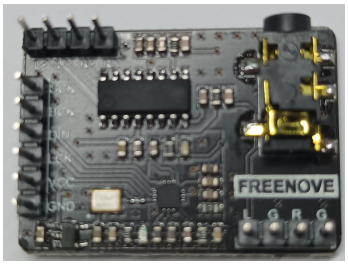
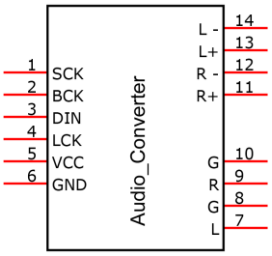
In this project, we will read files in mp3 format from SD card, decode them through ESP32-S3, and use Audio Converter & Amplifier module to transcode into stereo output.

Component List

ESP32-S3 WROOM x1 	USB cable x1 	SDcard x1 
Card reader x1 (random color) 	Audio Converter & Amplifier 	Speaker 
ESP32-S3 WROOM Shield x1 	9V battery x1 (Not included in the kit, prepared by yourself) 	
	9V battery cable x1 	

Component knowledge

Here are the front and back of Audio Converter & Amplifier module.

front view	reverse view	schematic diagram
		

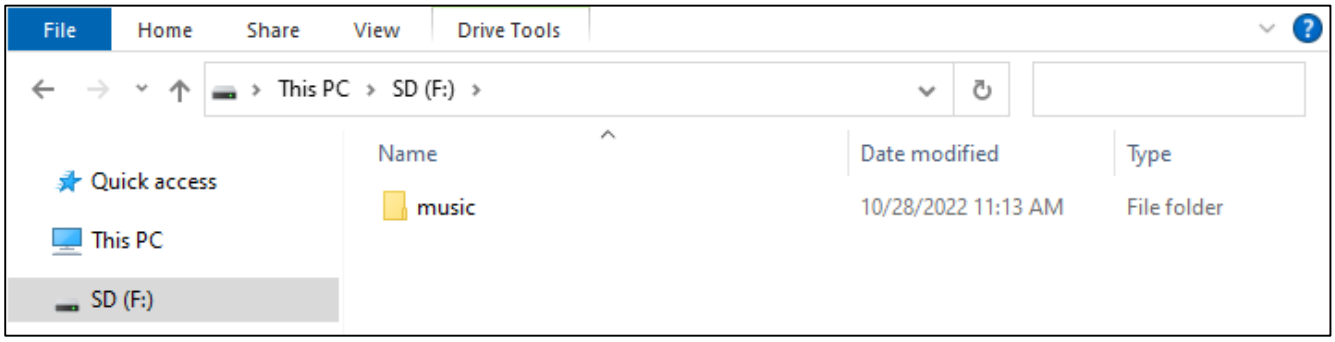
Interface description for Audio Converter & Amplifier module

Pin	Name	Introductions
1	SCK	System clock input
2	BCK	Audio data bit clock input
3	DIN	Audio data input
4	LCK	Audio data word clock input
5	VCC	Power input，3.3V~5.0V
6	GND	Power Ground
7	L	External audio left channel input
8	G	Power Ground
9	R	External audio right channel input
10	G	Power Ground
11	R+	Positive pole of right channel horn
12	R-	Negative pole of right channel horn
13	L+	Positive pole of left channel horn
14	L-	Negative pole of left channel horn

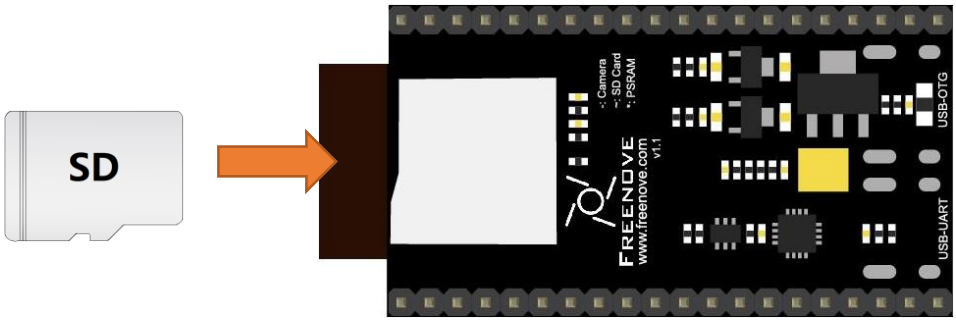
Speaker interface: The board features two speaker groups, namely Group L (L+ & L-) and Group R (R+ & R-), for connecting the left and right channel speakers. Each speaker interface can be connected to either Group L or Group R. However, it is important to note that connecting one interface to Group L will render the other interface incompatible with Group R. This could cause the module to malfunction, and therefore should be avoided.
Headphone interface: the interface to connect the headphones.
I2S input interface: connect to a device with I2S, transcoding audio data into DAC audio signals.
External audio input interface: connect to external audio equipment. amplifying externally input audio signals.
Power interface: connect to external power supply. The external power supply selected should fall within the range of 3.3V-5.0V.

Circuit

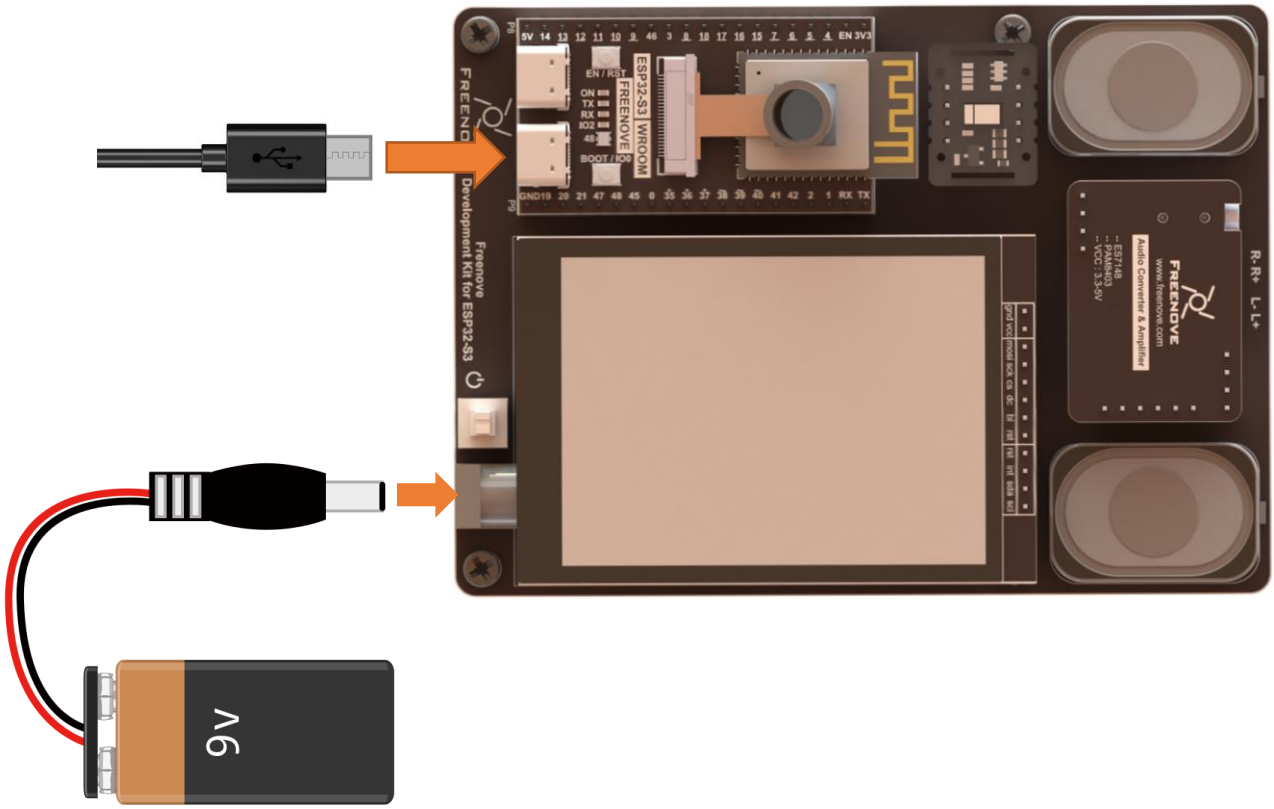
Before compling code, we should copy music to SD card first, as illustrated below.
We have placed a folder called "music" in:
Freenove-Development-Kit-for-ESP32-S3\Sketch\Sketch_05_I2S_Audio,
please copy this folder to the SD card.



Before connecting the USB cable, insert the SD card into the SD card slot on the back of the ESP32-S3.



Plug the audio module to the board. Connect Freenove ESP32-S3 to the computer with the USB cable. Hardware connection. If you need any support, please feel free to contact us via: support@freenove.com

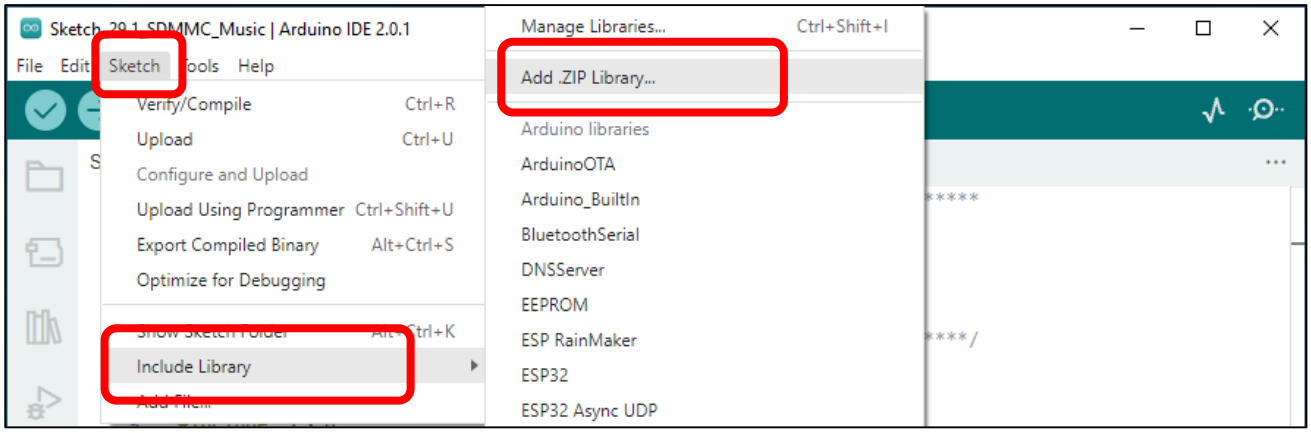


Sketch

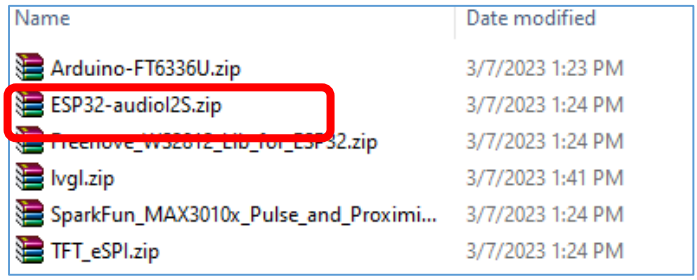
How to install the library

In this project, we will use the ESP32-audioI2S.zip library to decode the audio files in the SD card, and then output the audio signal through IIS. If you have not installed this library, please follow the steps below to install it.

Open arduino->Sketch->Include library-> Add .ZIP Library.

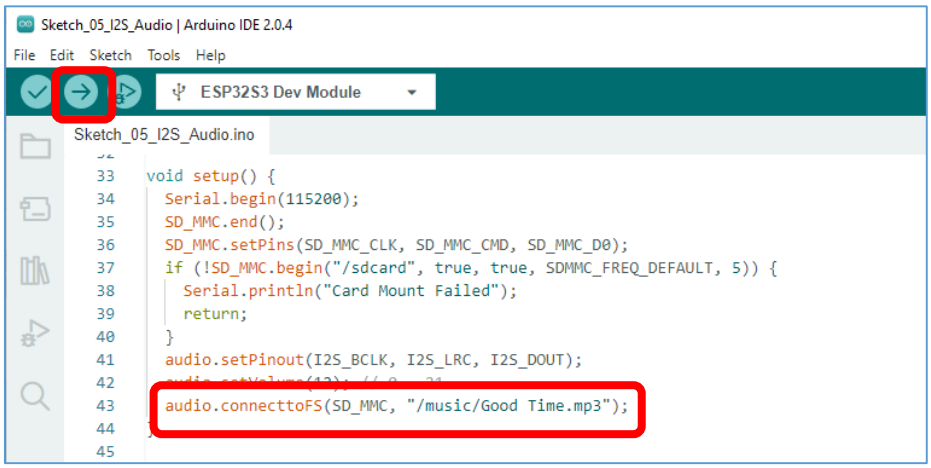


In the newly opened window, select "**Freenove-Development-Kit-for-ESP32-S3Libraries\ESP32-audioI2S.zip**" and click "Open".



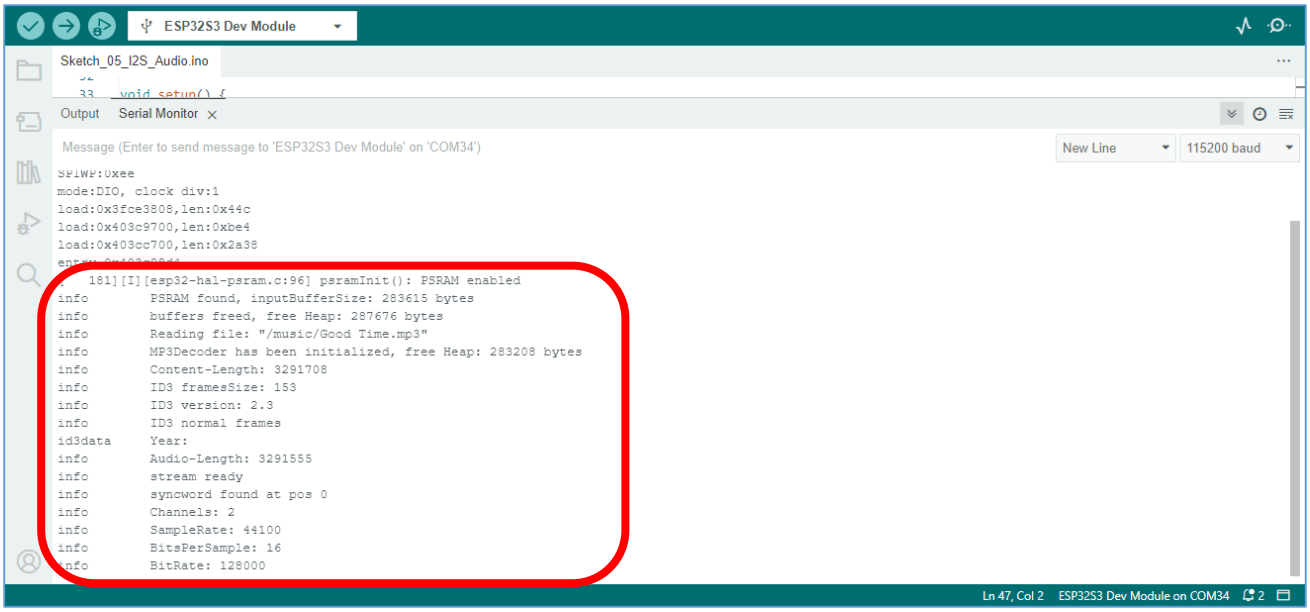
Sketch\Sketch_05_I2S_Audio

Click the upload button to upload the code to ESP32-S3.



Note: If the music fails to play, please check whether the name and path of the music is correct.

Compile and upload the code to the ESP32-S3 WROOM and open the serial monitor. It takes a few seconds to initialize the program. When you see the messages as below, it means that ESP32-S3 has started parsing the mp3 in sd card and started playing music through iis.



The following is the program code:

```
#include "Arduino.h"
#include "Audio.h"
#include "FS.h"
#include "SD_MMC.h"

#define SD_MMC_CMD 38
#define SD_MMC_CLK 39
#define SD_MMC_D0  40
#define I2S_BCLK   42
#define I2S_DOUT   41
#define I2S_LRC    14

Audio audio;

void setup() {
  Serial.begin(115200);
  SD_MMC.setPins(SD_MMC_CLK, SD_MMC_CMD, SD_MMC_D0);
  if (!SD_MMC.begin("/sdcard", true, true, SDMMC_FREQ_DEFAULT, 5)) {
    Serial.println("Card Mount Failed");
    return;
  }
  audio.setPinout(I2S_BCLK, I2S_LRC, I2S_DOUT);
  audio.setVolume(12);  // 0...21
  audio.connecttoFS(SD_MMC, "/music/Good Time.mp3");
}

void loop() {
  audio.loop();
  if (Serial.available()) {  // put streamURL in serial monitor
    audio.stopSong();
    String r = Serial.readString();
    r.trim();
    if (r.length() > 5) audio.connecttoFS(SD_MMC, r.c_str());
    log_i("free heap=%i", ESP.getFreeHeap());
  }
}

// optional
void audio_info(const char *info) {
  Serial.print("info        "); Serial.println(info);
}
void audio_id3data(const char *info) {  //id3 metadata
  Serial.print("id3data     "); Serial.println(info);
```

```
}
void audio_eof_mp3(const char *info) {  //end of file
  Serial.print("eof_mp3     "); Serial.println(info);
}
void audio_showstation(const char *info) {
  Serial.print("station     "); Serial.println(info);
}
void audio_showstreamtitle(const char *info) {
  Serial.print("streamtitle "); Serial.println(info);
}
void audio_bitrate(const char *info) {
  Serial.print("bitrate     "); Serial.println(info);
}
void audio_commercial(const char *info) {  //duration in sec
  Serial.print("commercial  "); Serial.println(info);
}
void audio_icyurl(const char *info) {  //homepage
  Serial.print("icyurl      "); Serial.println(info);
}
void audio_lasthost(const char *info) {  //stream URL played
  Serial.print("lasthost    "); Serial.println(info);
}
```

Add music decoding header files and SD card drive files.

```
#include "Arduino.h"
#include "Audio.h"
#include "FS.h"
#include "SD_MMC.h"
```

Define the drive pins for SD card and IIS. Note that the SD card driver pins cannot be modified, but the IIS drive pins can be modified.

```
#define SD_MMC_CMD 38
#define SD_MMC_CLK 39
#define SD_MMC_D0  40
#define I2S_BCLK   42
#define I2S_DOUT   41
#define I2S_LRC    14
```

Declare an audio decoding object, associate it with the pin, set the volume, and set the decoding object.

```
  Audio audio;
  ...
  audio.setPinout(I2S_BCLK, I2S_LRC, I2S_DOUT);
  audio.setVolume(12);  // 0...21
  audio.connecttoFS(SD_MMC, "/music/Good Time.mp3");
```

Continuously play music until the current track finishes. Upon receiving data through the serial port, remove any leading or trailing spaces and use the audio object to decode the data.

```
45    audio.loop();
46    if (Serial.available()) {  // put streamURL in serial monitor
47      audio.stopSong();
48      String r = Serial.readString();
49      r.trim();
50      if (r.length() > 5) audio.connecttoFS(SD_MMC, r.c_str());
51      log_i("free heap=%i", ESP.getFreeHeap());
52    }
```

In other words, if you want to switch the music in the SD card, you can directly input the song through the serial port.

These functions are used to print information about audio decoding. If you don't want to see this information in the serial port, you can simply comment out these functions."

```
67    void audio_info(const char *info);
71    void audio_id3data(const char *info);
75    void audio_eof_mp3(const char *info);
79    void audio_showstation(const char *info);
83    void audio_showstreamtitle(const char *info);
87    void audio_bitrate(const char *info);
91    void audio_commercial(const char *info);
95    void audio_icyurl(const char *info);
99    void audio_lasthost(const char *info);
```

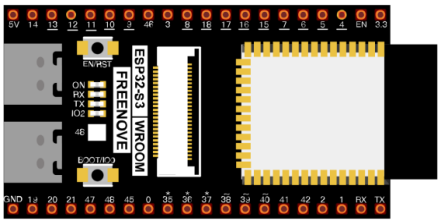
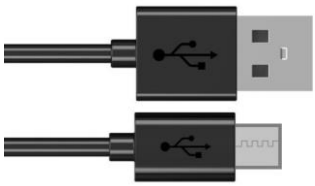
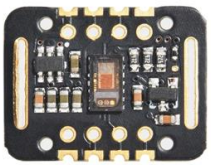
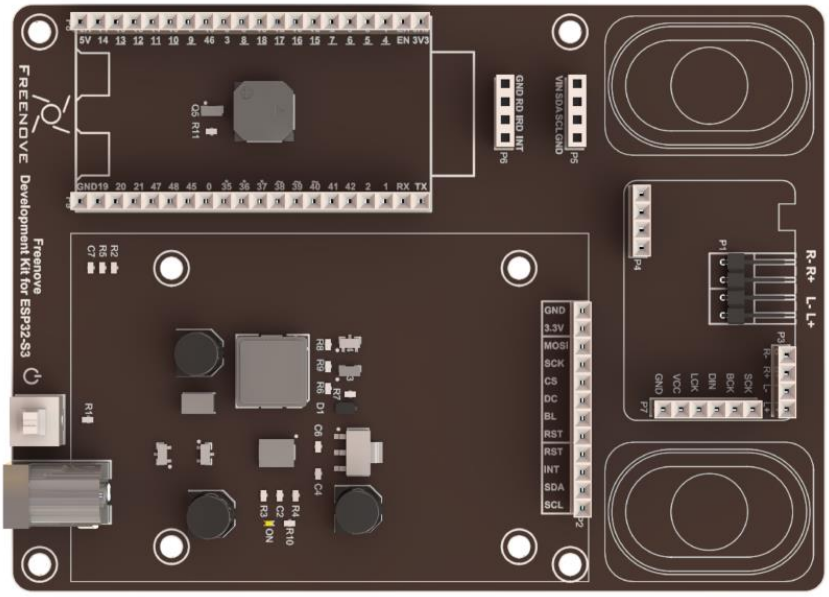
Chapter 6 MAX30102

In this section, we will explore how to utilize the MAX30102 module to measure human heart rate and blood oxygen saturation.

Project 6.1 MAX30102

In this project, we will read data from MAX30102, process the data, and print it out via serial communication.

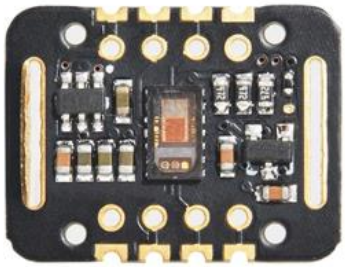
Component List

ESP32-S3 WROOM x1 	USB cable x1 	Max30102 
ESP32-S3 WROOM Shield x1 		

Component knowledge

MAX30102 is a module that integrates a pulse oximeter and biometric sensor for monitoring. It includes a red LED and an infrared LED, a photodetector, optical components, and low-noise electronic circuitry with ambient light suppression. It is typically used for heart rate and blood oxygen sensing in wearable devices, and can be worn on the fingertip, earlobe, and wrist.

The MAX30102 module uses the I2C communication interface to transmit the collected data to devices such as Arduino and ESP32 for heart rate and blood oxygen calculations.

front view	reverse view
	

When the MAX30102 is operating, it emits red and infrared light. As these lights pass through human tissue, the varying levels of light transmission caused by pulsating blood vessels can be detected by the module's photodetectors. These signals are then converted into electrical signals, amplified, and output to the main control chip.

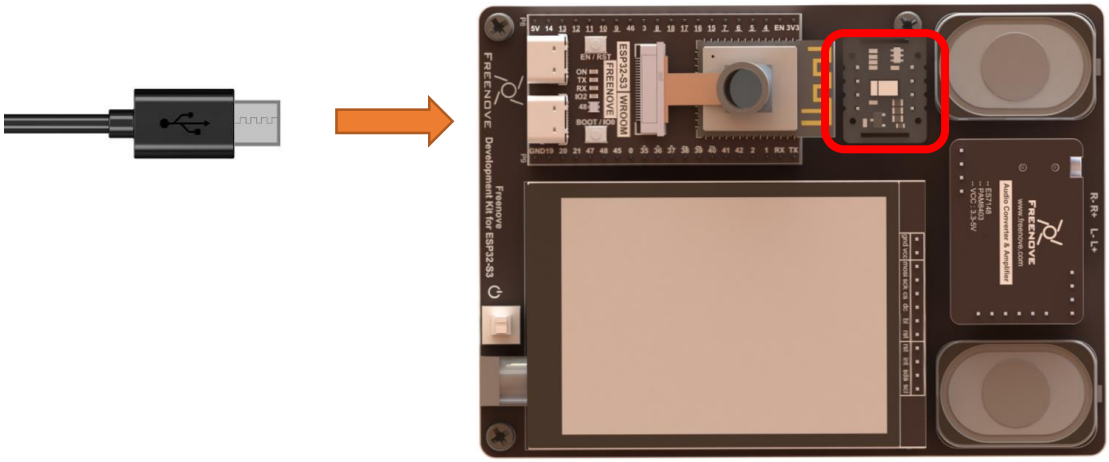
To obtain the data, we only need to access the I2C address of the blood oxygen module using the I2C protocol. The default I2C address for MAX30102 is 0x57.

Introduction to the pins of Max30102 module

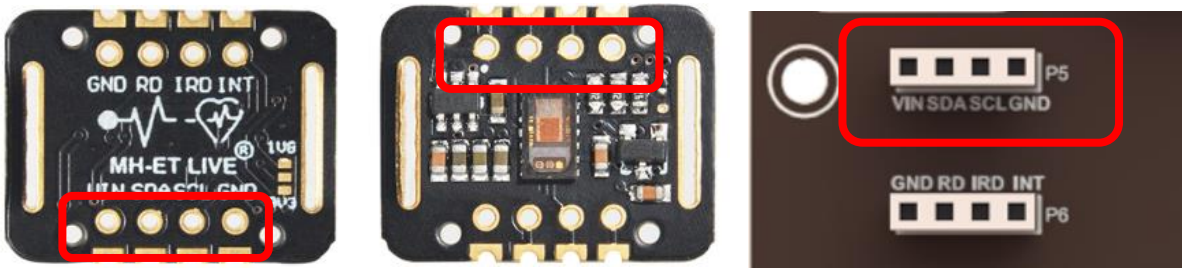
Pin	Name	Introductions
1	Vin	Power input，3.3V~5.0V
2	SDA	I2C data pin, connect to I2C data line
3	SCL	I2C clock pin, connect to I2C clock line
4	GND	Power Ground
5	INT	Interrupt signal pin of MAX30102 chip
6	IRD	Ground of RED LED, not connected be default
7	RD	Ground of IR LED, not connected be default
8	GND	Power Ground

Circuit

Connect Freenove ESP32-S3 to the computer using the USB cable.



Please note that when installing the MAX30102 module, it is crucial not to connect it reversely. Doing so may cause adverse effects on the circuit or even damage it. Please follow the marks on the board strictly and make sure that the pins are installed PIN to PIN.

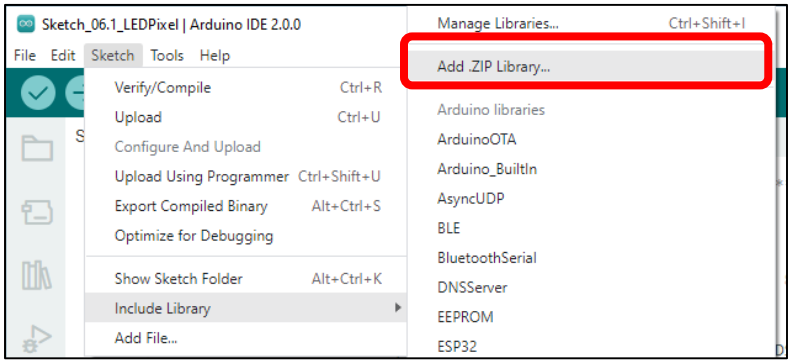


Sketch

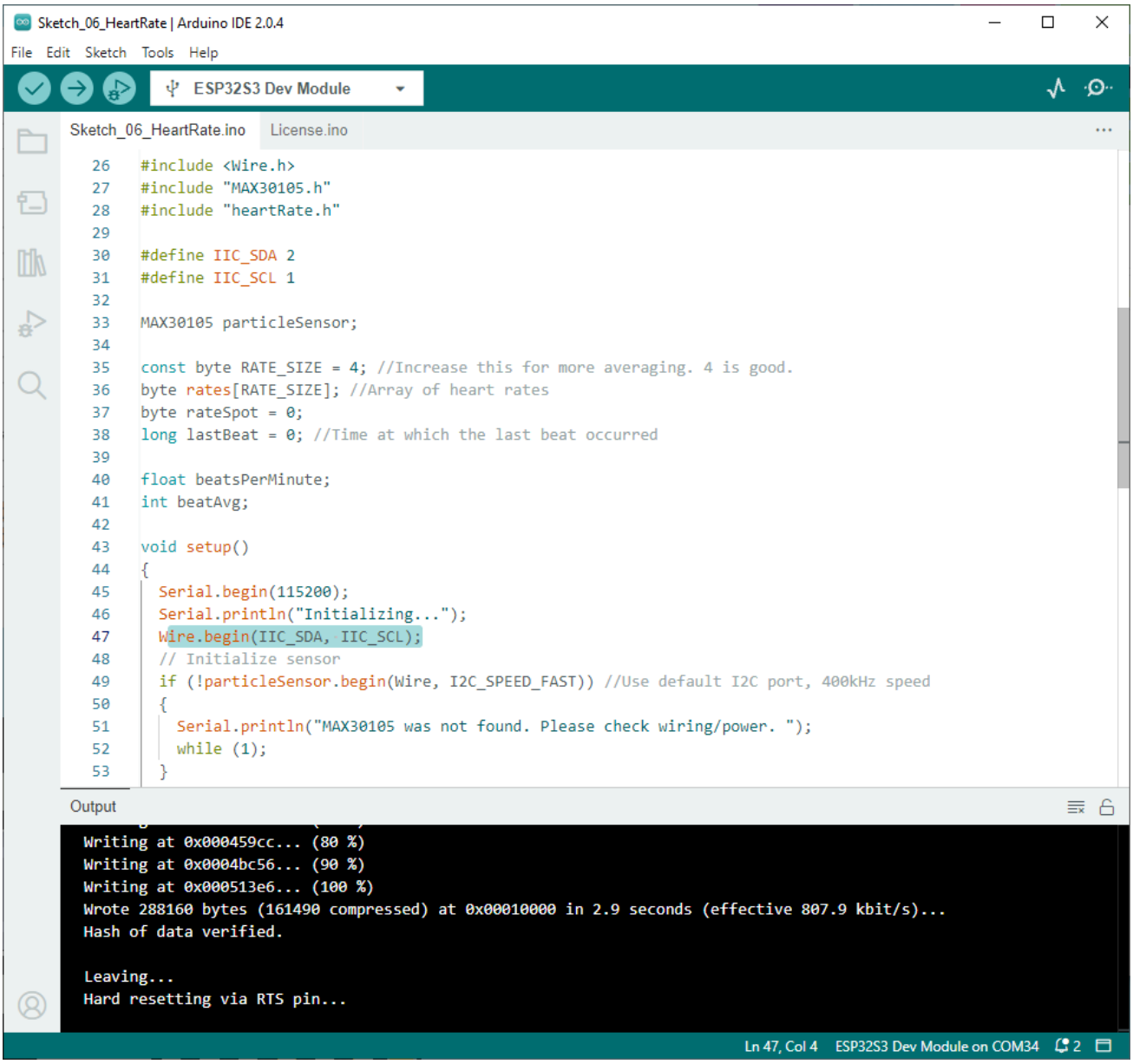
This code uses a library named "SparkFun_MAX3010x_Pulse_and_Proximity_Sensor_Library". If you have not installed it, please do so first.

How to install the library

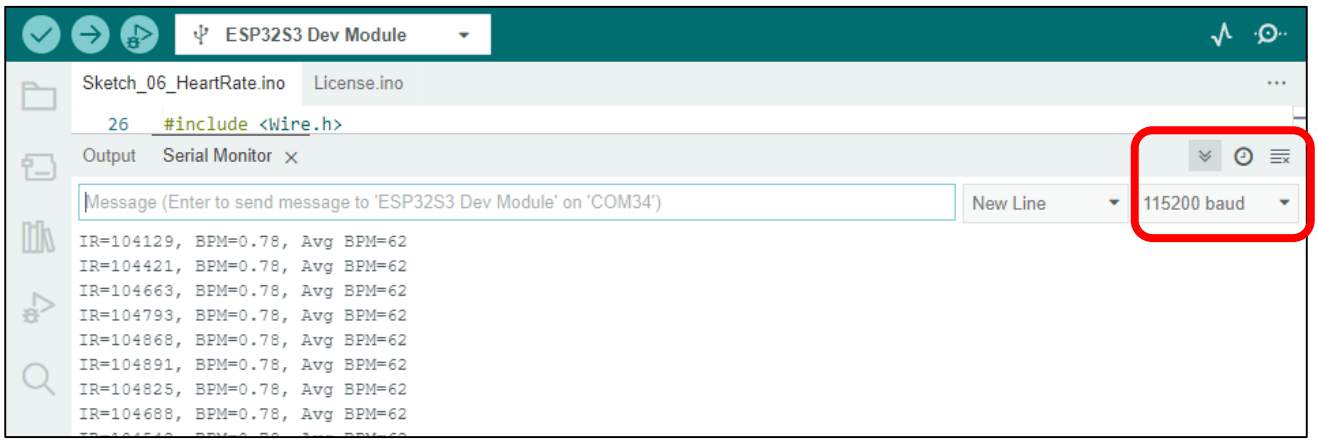
Open Arduino IDE, click Sketch→Include Library→Add .ZIP Library. In the pop-up window, find the file named "**Freenove-Development-Kit-for-ESP32-S3\Libraries\SparkFun_MAX3010x_Pulse_and_Proximity_Sensor_Library.Zip**" which locates in this directory, and click OPEN.



Sketch_06_HeartRate



Click Upload to upload the code to ESP32-S3, and then open the serial port monitor. Set the baud rate to 115200 and gently place the tip of your finger on the Max30102 chip. As shown in the picture below.



The following is the program code:

```
#include <Wire.h>
#include "MAX30105.h"
#include "heartRate.h"

#define IIC_SDA 2
#define IIC_SCL 1

MAX30105 particleSensor;

const byte RATE_SIZE = 4; //Increase this for more averaging. 4 is good.
byte rates[RATE_SIZE]; //Array of heart rates
byte rateSpot = 0;
long lastBeat = 0; //Time at which the last beat occurred

float beatsPerMinute;
int beatAvg;

void setup(){
  Serial.begin(115200);
  Serial.println("Initializing...");
  Wire.begin(IIC_SDA, IIC_SCL);
  // Initialize sensor
  if (!particleSensor.begin(Wire, I2C_SPEED_FAST)) {//Use default I2C port, 400kHz speed
    Serial.println("MAX30105 was not found. Please check wiring/power. ");
    while (1);
  }
  Serial.println("Place your index finger on the sensor with steady pressure.");

  particleSensor.setup(); //Configure sensor with default settings
  particleSensor.setPulseAmplitudeRed(0x0A); //Turn Red LED to low to indicate sensor is
running
  particleSensor.setPulseAmplitudeGreen(0); //Turn off Green LED
```

```
}

void loop(){
  long irValue = particleSensor.getIR();
  if (checkForBeat(irValue) == true){
    //We sensed a beat!
    long delta = millis() - lastBeat;
    lastBeat = millis();
    beatsPerMinute = 60 / (delta / 1000.0);
    if (beatsPerMinute < 255 && beatsPerMinute > 20){
      rates[rateSpot++] = (byte)beatsPerMinute; //Store this reading in the array
      rateSpot %= RATE_SIZE; //Wrap variable
      //Take average of readings
      beatAvg = 0;
      for (byte x = 0 ; x < RATE_SIZE ; x++)
        beatAvg += rates[x];
      beatAvg /= RATE_SIZE;
    }
  }
  Serial.print("IR=");
  Serial.print(irValue);
  Serial.print(", BPM=");
  Serial.print(beatsPerMinute);
  Serial.print(", Avg BPM=");
  Serial.print(beatAvg);

  if (irValue < 50000)
    Serial.print(" No finger?");
  Serial.println();
}
```

To use some libraries, first you need to include their header file.

```
#include <Wire.h>
#include "MAX30105.h"
#include "heartRate.h"
```

Define I2C pins to communicate with MAX30102 module.

```
#define IIC_SDA 2
#define IIC_SCL 1
```

Declare an object to manipulate the data that I2C reads and writes to Max30102.

```
MAX30105 particleSensor;
```

Initialize I2C and call I2C to initialize the Max30102 module.

```
  Wire.begin(IIC_SDA, IIC_SCL);
  // Initialize sensor
  if (!particleSensor.begin(Wire, I2C_SPEED_FAST)) {//Use default I2C port, 400kHz speed
    Serial.println("MAX30105 was not found. Please check wiring/power. ");
```

```
25      while (1);
26    }
```

Configure Max30102 module.

```
29    particleSensor.setup(); //Configure sensor with default settings
30    particleSensor.setPulseAmplitudeRed(0x0A); //Turn Red LED to low to indicate sensor is
      running
31    particleSensor.setPulseAmplitudeGreen(0); //Turn off Green LED
```

Obtain the infrared raw data of Max30102 module.

```
35  long irValue = particleSensor.getIR();
```

Calculate the number of heartbeats per minute and the average number of heartbeats.

We need to consider the fluctuation of data caused by factors such as the location and area of our finger that is in contact with the MAX30102 during each measurement. Therefore, it is more accurate to use the method of summing up and averaging the data to obtain a person's heart rate

```
36    if (checkForBeat(irValue) == true){
37      //We sensed a beat!
38      long delta = millis() - lastBeat;
39      lastBeat = millis();
40      beatsPerMinute = 60 / (delta / 1000.0);
41      if (beatsPerMinute < 255 && beatsPerMinute > 20){
42        rates[rateSpot++] = (byte)beatsPerMinute; //Store this reading in the array
43        rateSpot %= RATE_SIZE; //Wrap variable
44        //Take average of readings
45        beatAvg = 0;
46        for (byte x = 0 ; x < RATE_SIZE ; x++)
47          beatAvg += rates[x];
48        beatAvg /= RATE_SIZE;
49      }
50    }
```

Print out the processed data via serial port.

```
51    Serial.print("IR=");
52    Serial.print(irValue);
53    Serial.print(", BPM=");
54    Serial.print(beatsPerMinute);
55    Serial.print(", Avg BPM=");
56    Serial.print(beatAvg);
```

When the raw data value is smaller than 50,000, it is generally considered that the finger has been removed from the MAX30102 module.

```
58    if (irValue < 50000)
59      Serial.print(" No finger?");
```

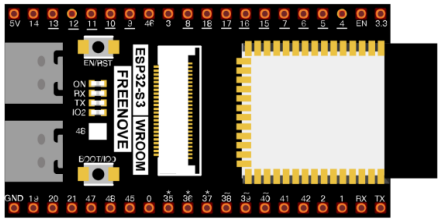
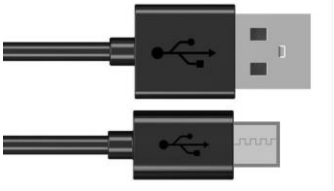
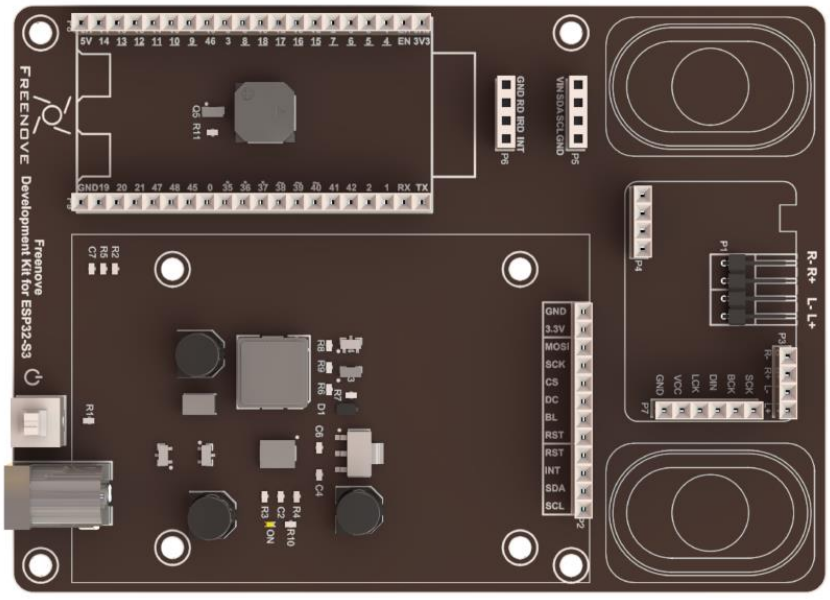
Chapter 7 Drving Freenove 2.8-Inch Screen

From this chapter, we start to learn the use of screens.

Project 7.1 Screen Display Function

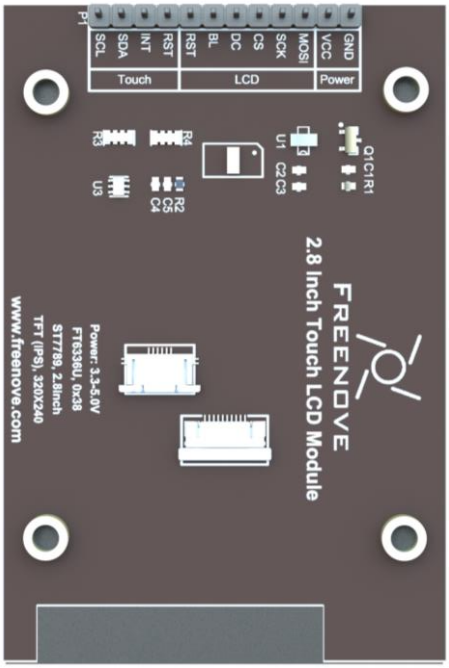
In this project, we will learn how to drive a screen and display information on it.

Component List

ESP32-S3 WROOM x1 	USB cable x1 	2.8-inch Screen 
ESP32-S3 WROOM Shield x1 		

Component knowledge

Below are the front and back photos of the Freenove 2.8-inch screen.

Front view	Back view
	

Introduction to the pins of Freenove 2.8-inch screen

Pin	Name	Introductions
1	GND	Power Ground
2	VCC	Power input，3.3V~5.0V
3	MOSI	Connecting to MOSI of SPI line, used to receive data of mcu.
4	SCK	Connecting to SCK of SPI line, used to receive clock signal of mcu.
5	CS	Connecting to CS of SPI line, used to enable/disable SPI communication.
6	DC	Command and data control pin, differentiate commands and data via high and low levels.
7	BL	Screen backlight control pin, enabled with low level by default
8	RST	Screen driver reset pin
9	RST	Touch driver reset pin
10	INT	Touch interrupt pin
11	SDA	Connect to I2C data line
12	SCL	Connect to I2C clock line

Freenove screen uses SPI protocol to drive screen display content and uses I2C protocol to obtain touch information from the screen.。

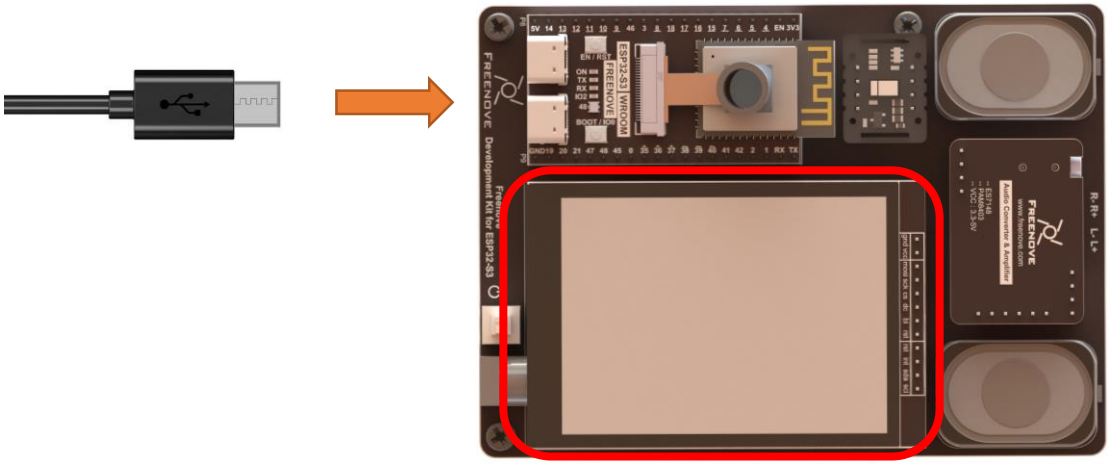
For more details, please refer to

Freenove-Development-Kit-for-ESP32-S3\Datasheet\ST7789V_SPEC_V1.0.pdf

Freenove-Development-Kit-for-ESP32-S3\Datasheet\FocalTech-FT6336U.pdf

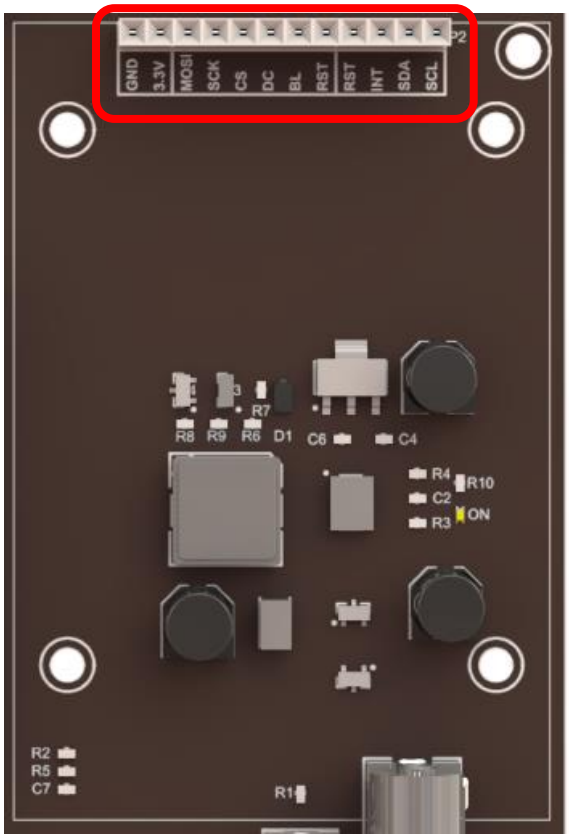
Circuit

Connect Freenove ESP32-S3 to the computer using the USB cable.



Note:

When connecting the screen to the ESP32-S3 WROOM Shield, please ensure that the pins are aligned with the marks before applying power to avoid damaging the screen.

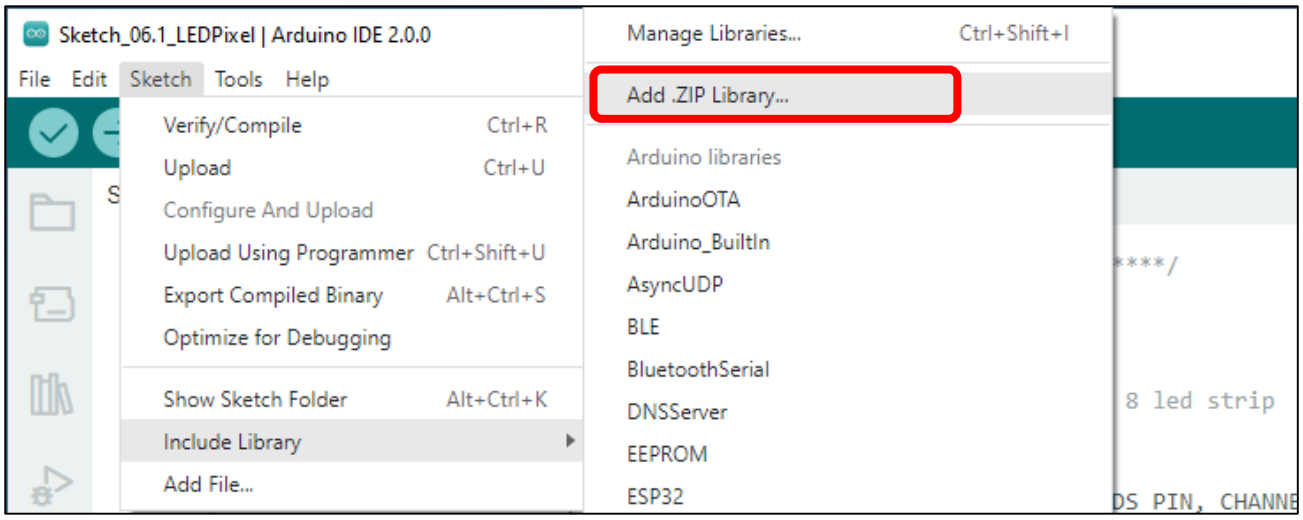


Sketch

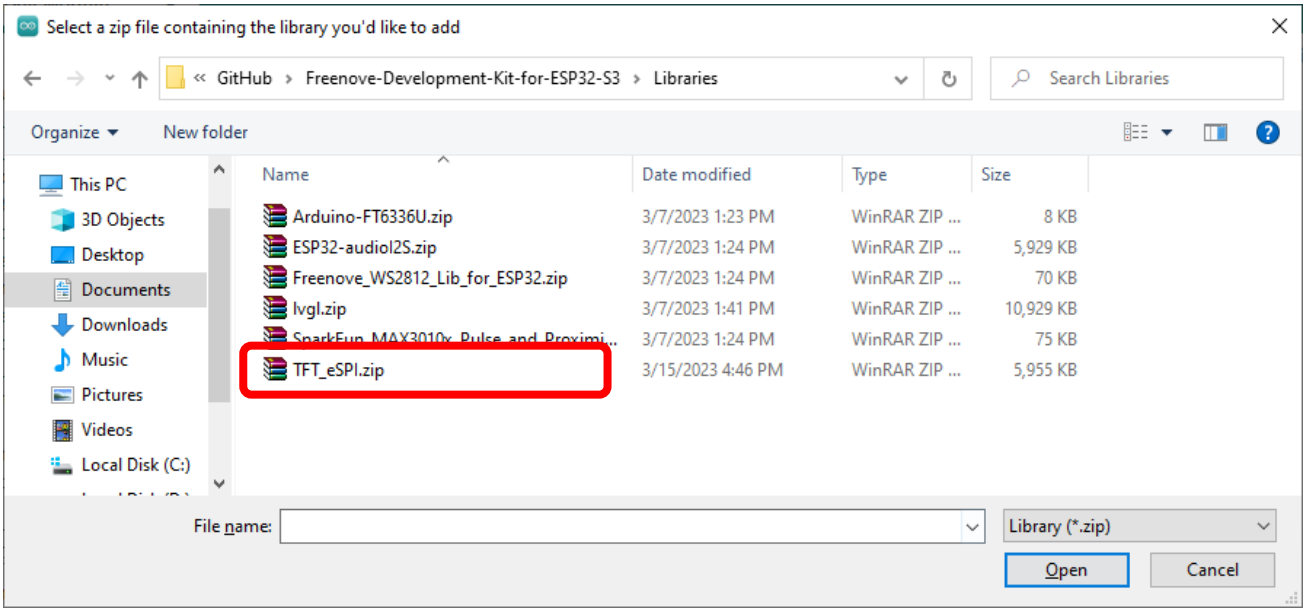
This code uses a library named "TFT_eSPI". If you have not installed it, please do so first.

How to install the library

Open Arduino IDE, click Sketch→Include Library→Add .ZIP Library. In the pop-up window, find the file named "**Freenove-Development-Kit-for-ESP32-S3\Libraries\TFT_eSPI.Zip**", which locates in this directory, and click OPEN.



Select TFT_eSPI.zip and click Open.



Sketch_07_TFT_Clock


```
#include <SPI.h>
#include <TFT_eSPI.h> // Hardware-specific library

#define TFT_GREY 0x5AEB

TFT_eSPI tft = TFT_eSPI();       // Invoke custom library

float sx = 0, sy = 1, mx = 1, my = 0, hx = -1, hy = 0;    // Saved H, M, S x & y multipliers
float sdeg=0, mdeg=0, hdeg=0;
uint16_t osx=120, osy=120, omx=120, omy=120, ohx=120, ohy=120;  // Saved H, M, S x & y coords
uint16_t x0=0, x1=0, yy0=0, yy1=0;
uint32_t targetTime = 0;                    // for next 1 second timeout

static uint8_t conv2d(const char* p); // Forward declaration needed for IDE 1.6.x
uint8_t hh=conv2d(__TIME__), mm=conv2d(__TIME__+3), ss=conv2d(__TIME__+6);  // Get H, M, S from compile time

bool initial = 1;

void setup(void) {
  tft.init();
  tft.setRotation(0);

  //tft.fillScreen(TFT_BLACK);
  //tft.fillScreen(TFT_RED);
```

Click "Upload" to upload the code to the ESP32-S3. Wait for the code to finish uploading, and the screen will display a clock as shown in the following figure.

The following is the program code:

```
#include <SPI.h>
#include <TFT_eSPI.h> // Hardware-specific library

#define TFT_GREY 0x5AEB

TFT_eSPI tft = TFT_eSPI();       // Invoke custom library

float sx = 0, sy = 1, mx = 1, my = 0, hx = -1, hy = 0;    // Saved H, M, S x & y multipliers
```

```
9   float sdeg=0, mdeg=0, hdeg=0;
10  uint16_t osx=120, osy=120, omx=120, omy=120, ohx=120, ohy=120;  // Saved H, M, S x & y coords
11  uint16_t x0=0, x1=0, yy0=0, yy1=0;
12  uint32_t targetTime = 0;                    // for next 1 second timeout
13
14  static uint8_t conv2d(const char* p); // Forward declaration needed for IDE 1.6.x
15  uint8_t hh=conv2d(__TIME__), mm=conv2d(__TIME__+3), ss=conv2d(__TIME__+6);  // Get H, M, S
    from compile time
16
17  bool initial = 1;
18
19  void setup(void) {
20    tft.init();
21    tft.setRotation(0);
22
23    //tft.fillScreen(TFT_BLACK);
24    //tft.fillScreen(TFT_RED);
25    //tft.fillScreen(TFT_GREEN);
26    //tft.fillScreen(TFT_BLUE);
27    //tft.fillScreen(TFT_BLACK);
28    tft.fillScreen(TFT_GREY);
29
30    tft.setTextColor(TFT_WHITE, TFT_GREY);  // Adding a background colour erases previous text
    automatically
31
32    // Draw clock face
33    tft.fillCircle(120, 120, 118, TFT_GREEN);
34    tft.fillCircle(120, 120, 110, TFT_BLACK);
35
36    // Draw 12 lines
37    for(int i = 0; i<360; i+= 30) {
38      sx = cos((i-90)*0.0174532925);
39      sy = sin((i-90)*0.0174532925);
40      x0 = sx*114+120;
41      yy0 = sy*114+120;
42      x1 = sx*100+120;
43      yy1 = sy*100+120;
44
45      tft.drawLine(x0, yy0, x1, yy1, TFT_GREEN);
46    }
47
48    // Draw 60 dots
49    for(int i = 0; i<360; i+= 6) {
50      sx = cos((i-90)*0.0174532925);
```

```
    sy = sin((i-90)*0.0174532925);
    x0 = sx*102+120;
    yy0 = sy*102+120;
    // Draw minute markers
    tft.drawPixel(x0, yy0, TFT_WHITE);

    // Draw main quadrant dots
    if(i==0 || i==180) tft.fillCircle(x0, yy0, 2, TFT_WHITE);
    if(i==90 || i==270) tft.fillCircle(x0, yy0, 2, TFT_WHITE);
  }

  tft.fillCircle(120, 121, 3, TFT_WHITE);

  tft.drawCentreString("Freenove",70,260,4);
  tft.drawString("V1.0",200,280,2);
  targetTime = millis() + 1000;
}

void loop() {
  if (targetTime < millis()) {
    targetTime += 1000;
    ss++;              // Advance second
    if (ss==60) {
      ss=0;
      mm++;            // Advance minute
      if(mm>59) {
        mm=0;
        hh++;          // Advance hour
        if (hh>23) {
          hh=0;
        }
      }
    }

    // Pre-compute hand degrees, x & y coords for a fast screen update
    sdeg = ss*6;                  // 0-59 -> 0-354
    mdeg = mm*6+sdeg*0.01666667;  // 0-59 -> 0-360 - includes seconds
    hdeg = hh*30+mdeg*0.0833333;  // 0-11 -> 0-360 - includes minutes and seconds
    hx = cos((hdeg-90)*0.0174532925);
    hy = sin((hdeg-90)*0.0174532925);
    mx = cos((mdeg-90)*0.0174532925);
    my = sin((mdeg-90)*0.0174532925);
    sx = cos((sdeg-90)*0.0174532925);
    sy = sin((sdeg-90)*0.0174532925);
```

```
95
96      if (ss==0 || initial) {
97        initial = 0;
98        // Erase hour and minute hand positions every minute
99        tft.drawLine(ohx, ohy, 120, 121, TFT_BLACK);
100       ohx = hx*62+121;
101       ohy = hy*62+121;
102       tft.drawLine(omx, omy, 120, 121, TFT_BLACK);
103       omx = mx*84+120;
104       omy = my*84+121;
105     }
106
107       // Redraw new hand positions, hour and minute hands not erased here to avoid flicker
108       tft.drawLine(osx, osy, 120, 121, TFT_BLACK);
109       osx = sx*90+121;
110       osy = sy*90+121;
111       tft.drawLine(osx, osy, 120, 121, TFT_RED);
112       tft.drawLine(ohx, ohy, 120, 121, TFT_WHITE);
113       tft.drawLine(omx, omy, 120, 121, TFT_WHITE);
114       tft.drawLine(osx, osy, 120, 121, TFT_RED);
115
116     tft.fillCircle(120, 121, 3, TFT_RED);
117   }
118 }
119
120 static uint8_t conv2d(const char* p) {
121   uint8_t v = 0;
122   if ('0' <= *p && *p <= '9')
123     v = *p - '0';
124   return 10 * v + *++p - '0';
125 }
```

To use some libraries, first you need to include their header files.

```
1  #include <SPI.h>
2  #include <TFT_eSPI.h> // Hardware-specific library
```

Declare a tft object for operating the screen.

```
6  TFT_eSPI tft = TFT_eSPI();       // Invoke custom library
```

TIME represents the current time at which the code was uploaded. The function conv2d is used to convert a string to a number.

```
14 static uint8_t conv2d(const char* p); // Forward declaration needed for IDE 1.6.x
15 uint8_t hh=conv2d(__TIME__), mm=conv2d(__TIME__+3), ss=conv2d(__TIME__+6);  // Get H, M, S
   from compile time
```

Initialize the screen and set not to rotate. The values 0-3 represent screen rotations of 0, 90, 180, and 270 degrees, respectively.

```
20   tft.init();
```

```
21    tft.setRotation(0);
```

Screen filling function. It can be used to set the background color of the screen.

```
28    tft.fillScreen(TFT_GREY);
```

Before using a font, first use the setTextColor function to set the font and its background color.

```
30    tft.setTextColor(TFT_WHITE, TFT_GREY);  // Adding a background colour erases previous text
      automatically
```

Draw a circle and fill it with color. Use two circles of different sizes to overlap and create the framework for the clock.

```
33    tft.fillCircle(120, 120, 118, TFT_GREEN);//(x, y, r, color)
34    tft.fillCircle(120, 120, 110, TFT_BLACK);
```

Draw a line. The starting and ending coordinates, as well as the desired color are needed.

```
45    tft.drawLine(x0, yy0, x1, yy1, TFT_GREEN); //(xs, ys, xe, ye, color)
```

Draw a dot, the coordinate and color of which need to be input.

```
45    tft.drawPixel(x0, yy0, TFT_WHITE);
```

Display strings on the screen. The size of text can be set to 2 or 4.

```
64    tft.drawString("Freenove",70,260,4);//(string, x, y, font)
```

For more detailed usage, please refer to: https://github.com/Bodmer/TFT_eSPI

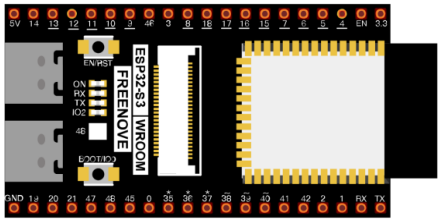
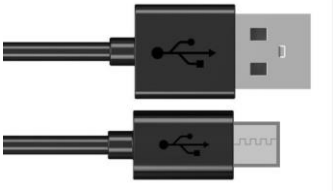
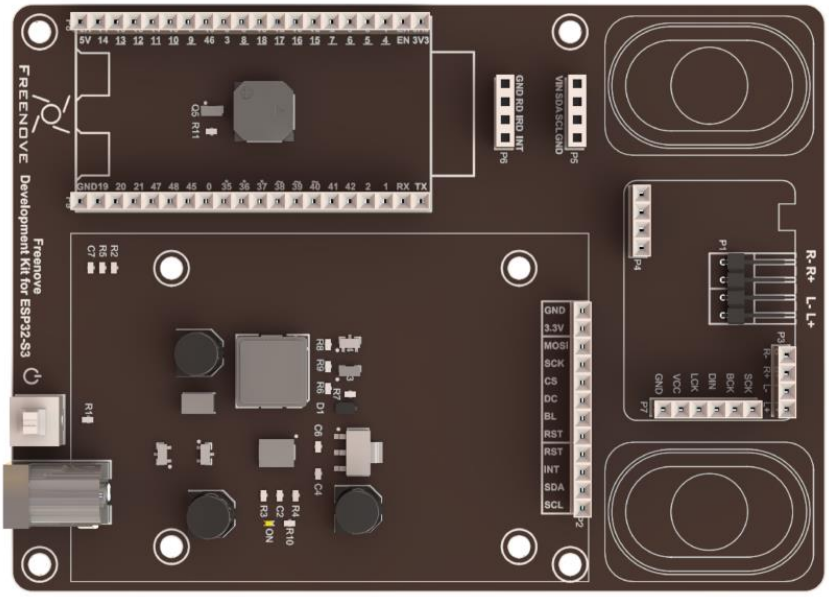
Chapter 8 Touch of Freenove 2.8-Inch Screen

In this chapter, we will learn the usage of screen touch function.

Project 8.1 Touch Function of the Screen

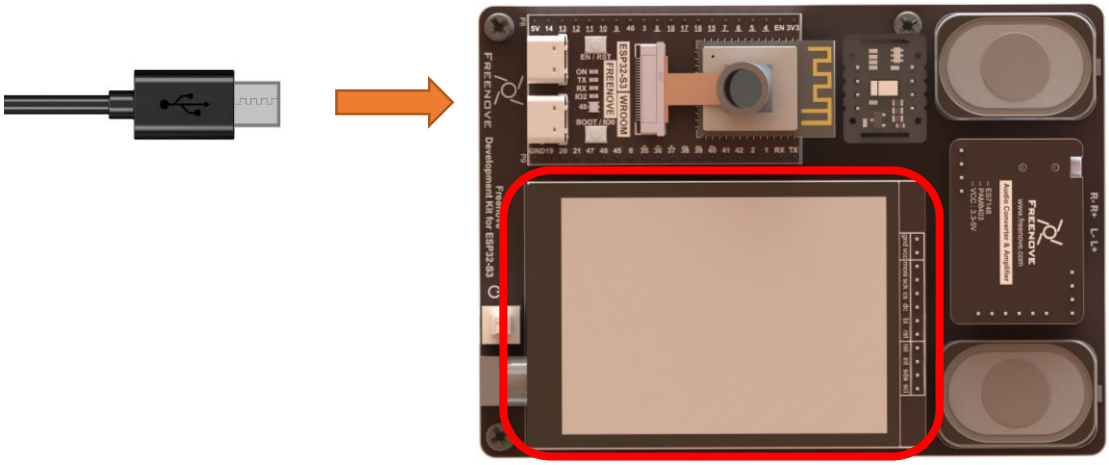
In this project, we will learn to obtain touch data from a screen and print them out through serial port.

Component List

ESP32-S3 WROOM x1 	USB cable x1 	2.8-inch screen 
ESP32-S3 WROOM Shield x1 		

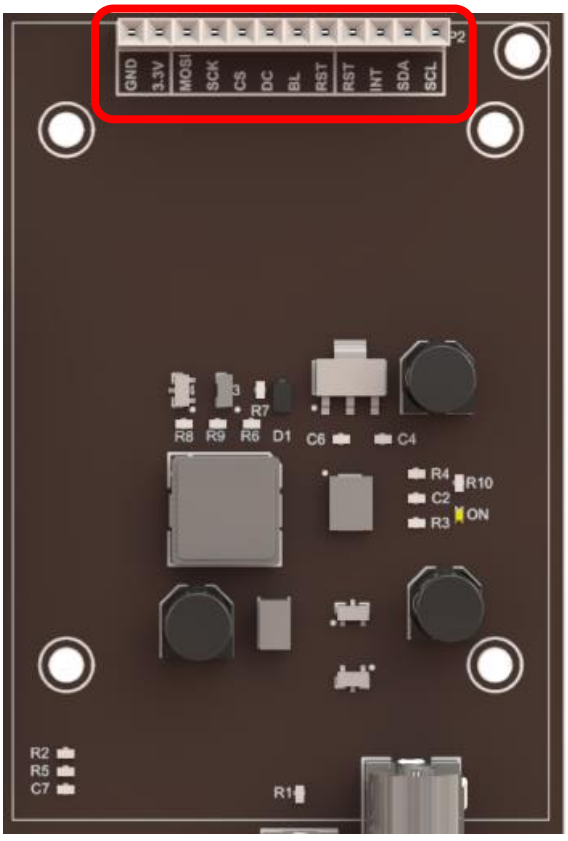
Circuit

Connect Freenove ESP32-S3 to the computer using the USB cable.



Note:

When connecting the screen to the ESP32-S3 WROOM Shield, please ensure that the pins are aligned with the marks before applying power to avoid damaging the screen.

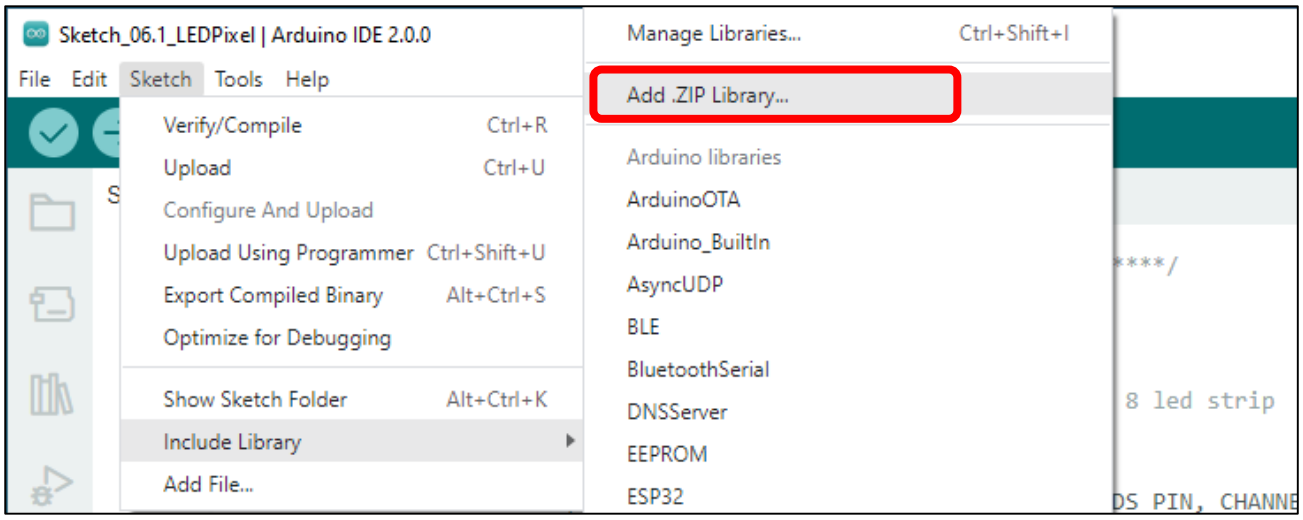


Sketch

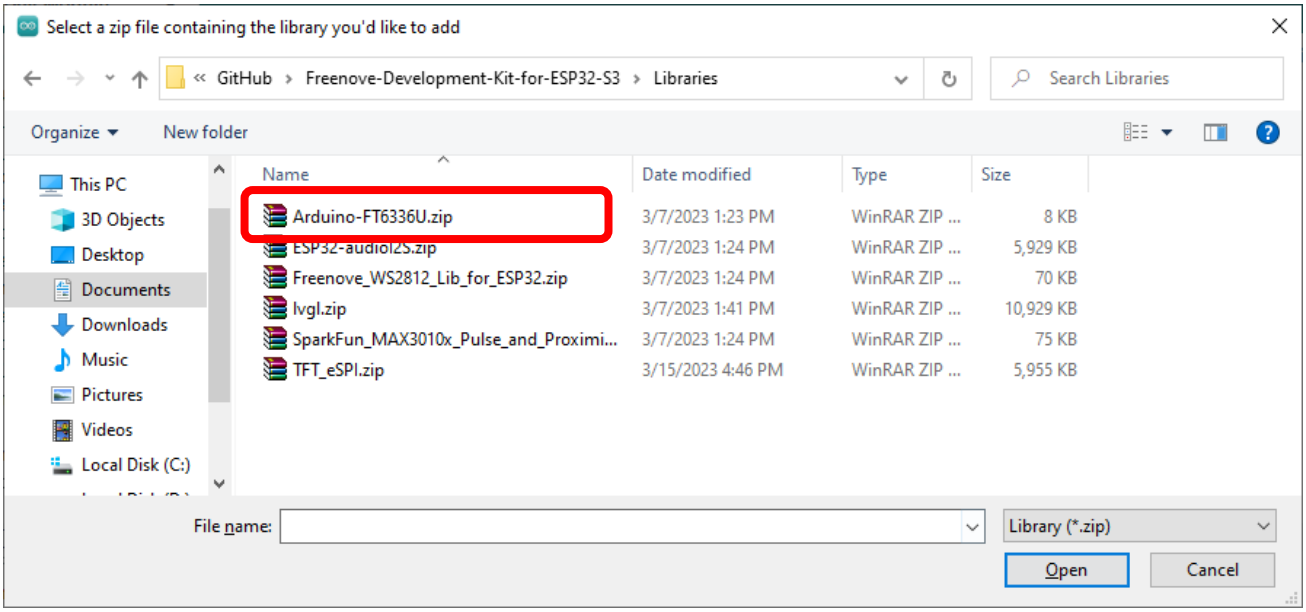
This code uses a library named "**Arduino-FT6336U**". If you have not installed it, please do so first.

How to install the library

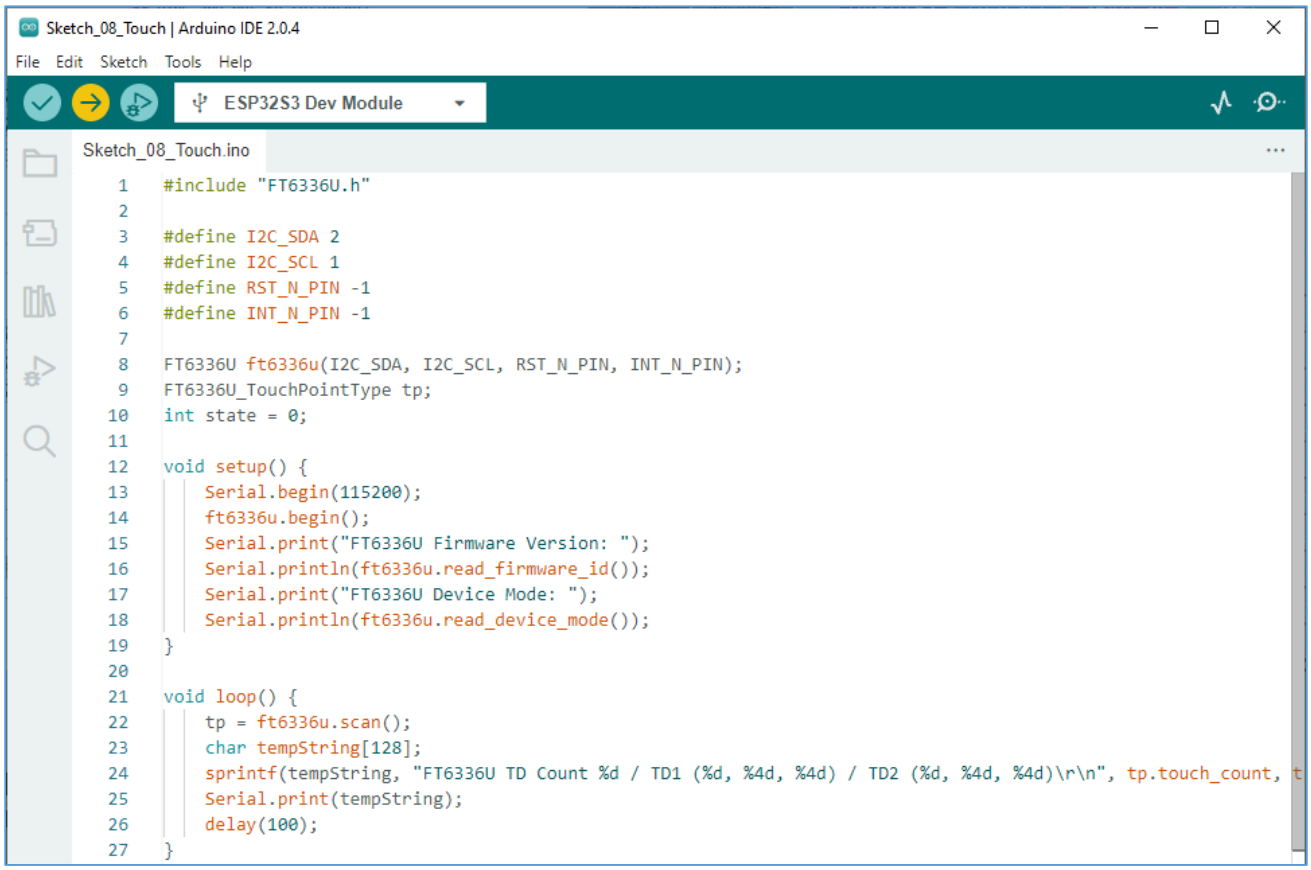
Open Arduino IDE, click Sketch→Include Library→Add .ZIP Library. In the pop-up window, find the file named "**Freenove-Development-Kit-for-ESP32-S3\Libraries\Arduino-FT6336U.Zip**", which locates in this directory, and click OPEN.



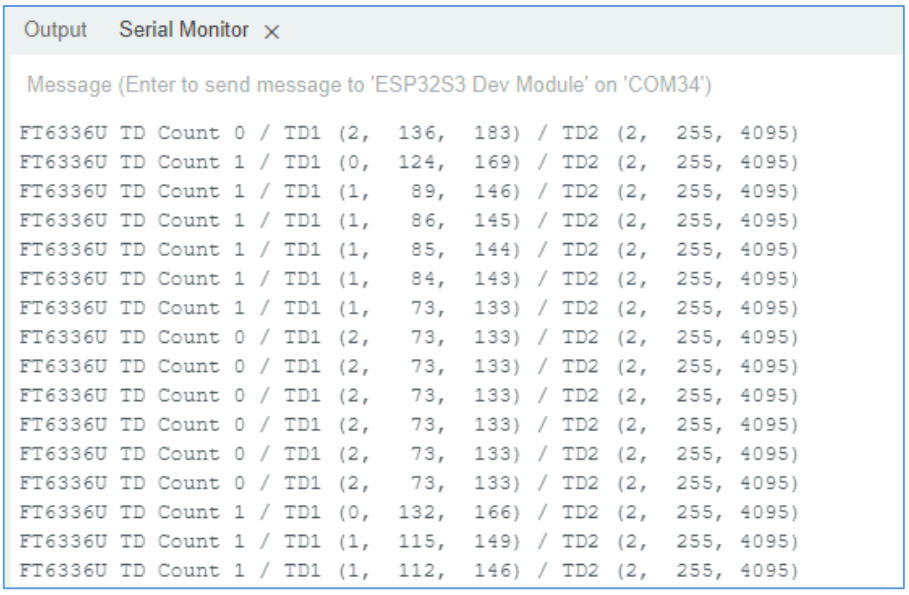
Select Arduino-FT6336U.zip and click Open.



Sketch_08_Touch



Click Upload to upload the code to ESP32-S3. After code completes uploading, open the serial monitor, long press or tap the screen, and you can see the messages as below:



The following is the program code:

```
#include "FT6336U.h"

#define I2C_SDA 2
#define I2C_SCL 1
#define RST_N_PIN -1
#define INT_N_PIN -1

FT6336U ft6336u(I2C_SDA, I2C_SCL, RST_N_PIN, INT_N_PIN);
FT6336U_TouchPointType tp;
int state = 0;

void setup() {
    Serial.begin(115200);
    ft6336u.begin();
    Serial.print("FT6336U Firmware Version: ");
    Serial.println(ft6336u.read_firmware_id());
    Serial.print("FT6336U Device Mode: ");
    Serial.println(ft6336u.read_device_mode());
}
void loop() {
    tp = ft6336u.scan();
    char tempString[128];
    sprintf(tempString, "FT6336U TD Count %d / TD1 (%d, %4d, %4d) / TD2 (%d, %4d, %4d)\r\n",
tp.touch_count, tp.tp[0].status, tp.tp[0].x, tp.tp[0].y, tp.tp[1].status, tp.tp[1].x,
tp.tp[1].y);
    Serial.print(tempString);
    delay(100);
}
```

To use some libraries, first you need to include their header files.

```
#include "FT6336U.h"
```

Declare an ft6336u object to read the touch chip data. Define a touch data type to receive data.

```
FT6336U ft6336u(I2C_SDA, I2C_SCL, RST_N_PIN, INT_N_PIN);
FT6336U_TouchPointType tp;
```

Initialize the touch chip.

```
    ft6336u.begin();
```

Non-blocking scan function, store scan results in tp.

```
  tp = ft6336u.scan();
```

Combine and print the captured touchdata.

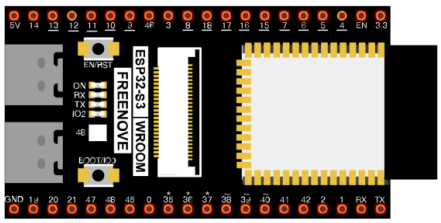
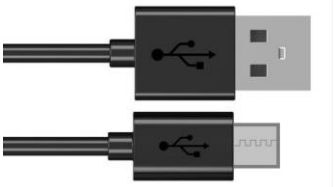
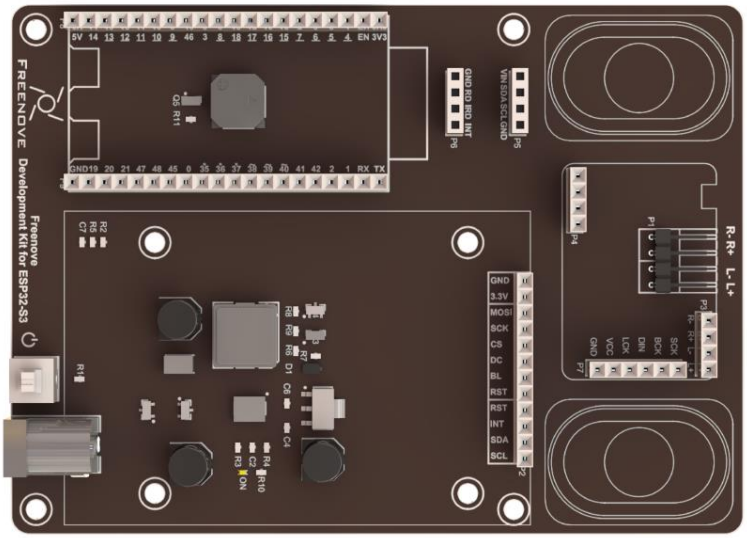
```
    sprintf(tempString, "FT6336U TD Count %d / TD1 (%d, %4d, %4d) / TD2 (%d, %4d, %4d)\r\n",
tp.touch_count, tp.tp[0].status, tp.tp[0].x, tp.tp[0].y, tp.tp[1].status, tp.tp[1].x,
tp.tp[1].y);
    Serial.print(tempString);
```

Chapter 9 LVGL

In this section, we will learn how to use the lvgl library.

Project 9.1 LVGL Test

Component List

ESP32-S3 WROOM x1 	USB cable x1 	2.8-inch Screen 
ESP32-S3 WROOM Shield x1 	9VBattery x1 (Not included in the kit, prepared by yourself) 	
	9V battery cable x1 	

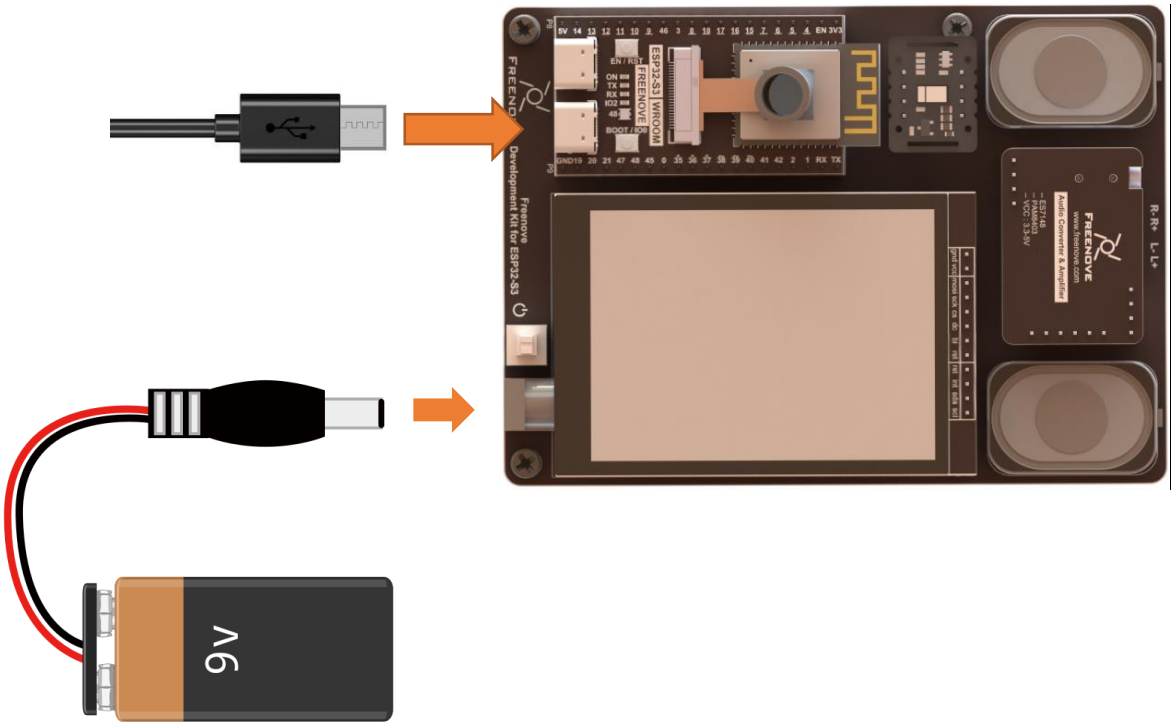
Component knowledge

LVGL is a widely-used embedded GUI library that is implemented in pure C, making it highly portable and performant. It offers rich features and content, supporting both display and input devices such as touchscreens and keyboards.

	Features supported by LVGL
1	Powerful building blocks such as buttons, charts, lists, sliders, images, and more.
2	Advanced graphics with animation, anti-aliasing, opacity, and smooth scrolling.
3	Various input devices, such as touchpads, mice, keyboards, encoders, and more.
4	Multiple languages with UTF-8 encoding.
5	Multiple display types, including TFT and monochrome displays.
6	Fully customizable graphical elements.
7	LVGL can be used independently of any microcontroller or display hardware.
8	Highly extensible and can be configured to use very little memory (e.g. 64 kB of flash and 16 kB of RAM)
9	It can be used with or without an operating system, and supports external memory and GPUs as optional features.
10	Single-frame buffer operation, even with advanced graphics effects.
11	Written in C language to achieve maximum compatibility (compatible with C++ as well).
12	LVGL has a simulator that allows for embedded GUI design on a PC without any embedded hardware.
13	Resources to help developers quickly get started with the library, including tutorials, examples, and themes.
14	A wide range of resources.

Circuit

Connect Freenove ESP32-S3 to the computer using the USB cable.
Hardware connection. If you need any support, please feel free to contact us via: support@freenove.com

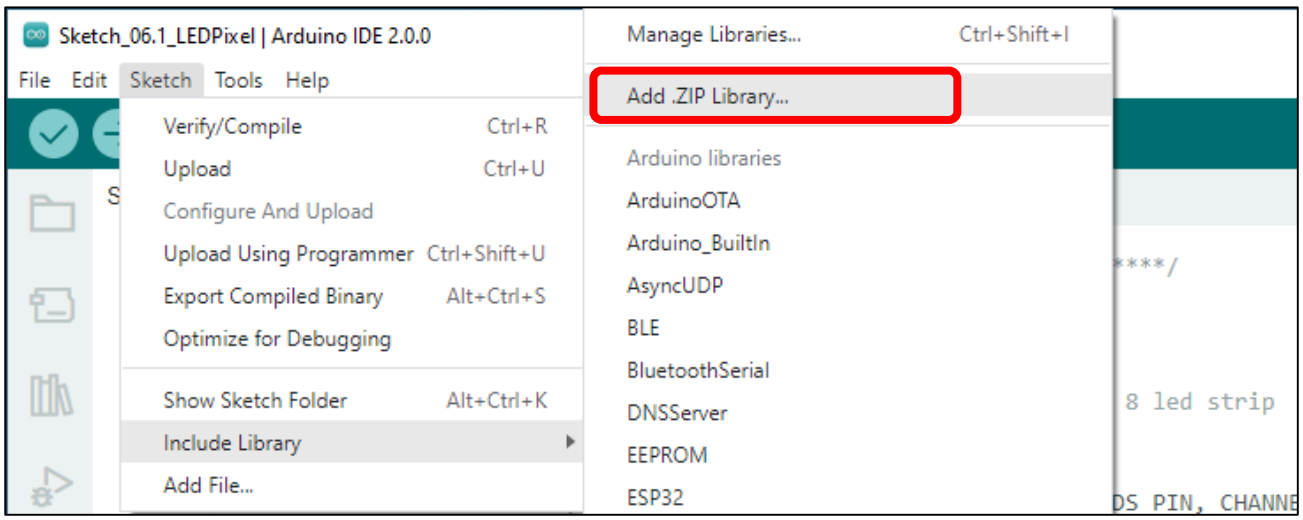


Sketch

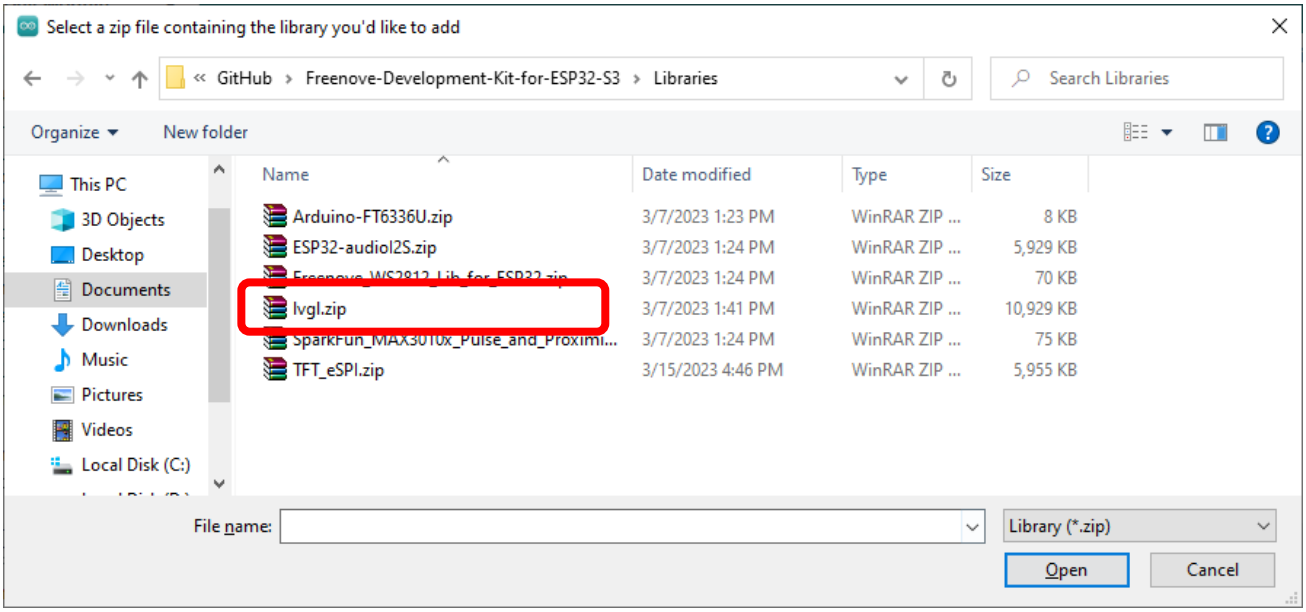
This code uses a library named "**lvgl**". If you have not installed it, please do so first.

How to install the library

Open Arduino IDE, click Sketch→Include Library→Add .ZIP Library. In the pop-up window, find the file named "**Freenove-Development-Kit-for-ESP32-S3\Libraries\lvgl.Zip**", which locates in this directory, and click OPEN.

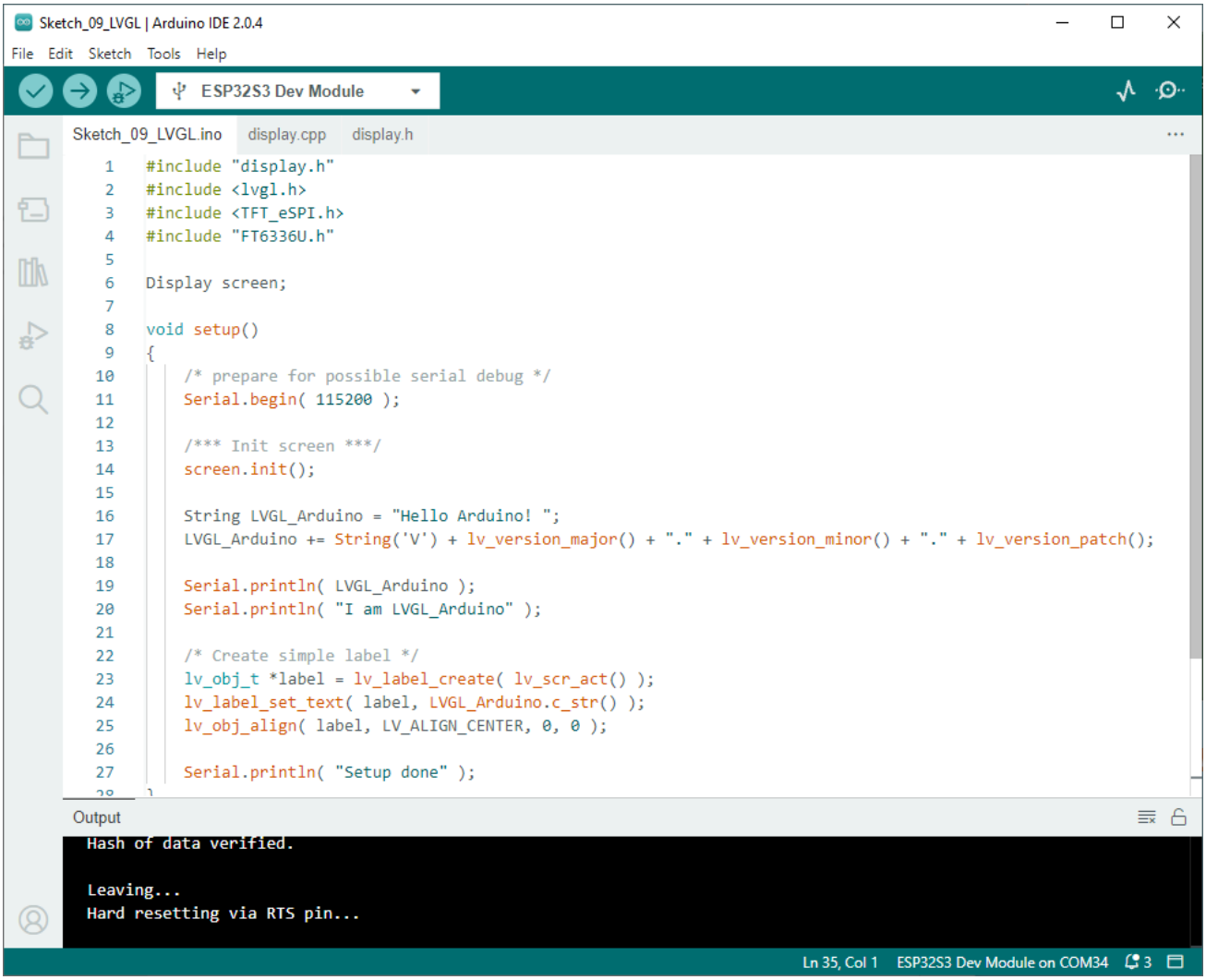


Select lvgl.zip and click Open.

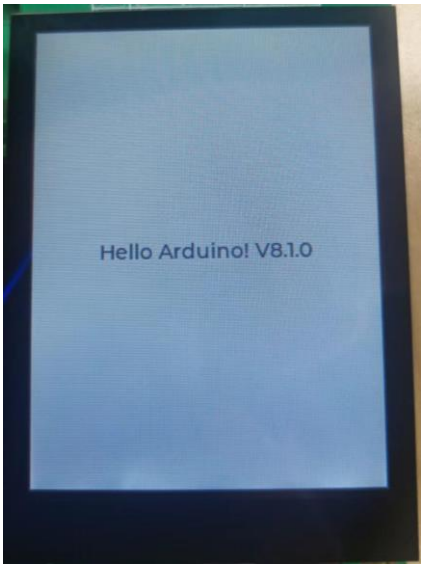


Please be aware that the lvgl.zip file is a pre-configured file library that is specifically designed for our product. Using the online "add library" function to add the lvgl library may cause compilation errors and render our product unusable.

Sketch_09_LVGL



Click on Upload to upload the code to the ESP32-S3. Once the code has completed uploading, if you can see the screen displaying as shown below, it means that you have successfully configured the library and can now begin learning LVGL.



The following is the program code:

```
#include "display.h"
#include <lvgl.h>
#include <TFT_eSPI.h>
#include "FT6336U.h"

Display screen;
void setup(){
    /* prepare for possible serial debug */
    Serial.begin( 115200 );
    /*** Init screen ***/
    screen.init();
    String LVGL_Arduino = "Hello Arduino! ";
    LVGL_Arduino += String('V') + lv_version_major() + "." + lv_version_minor() + "." +
lv_version_patch();
    Serial.println( LVGL_Arduino );
    Serial.println( "I am LVGL_Arduino" );
    /* Create simple label */
    lv_obj_t *label = lv_label_create( lv_scr_act() );
    lv_label_set_text( label, LVGL_Arduino.c_str() );
    lv_obj_align( label, LV_ALIGN_CENTER, 0, 0 );
    Serial.println( "Setup done" );
}

void loop(){
    screen.routine(); /* let the GUI do its work */
    delay( 5 );
}
```

To use some libraries, first you need to include their header files.

```
#include "display.h"
#include <lvgl.h>
#include <TFT_eSPI.h>
#include "FT6336U.h"
```

Request a screen operation object to operate the screen.

```
Display screen;
```

Initialize and configure the screen.

```
    screen.init();
```

Screen refresh function, by continuously calling this function, the interface can work properly.

```
    screen.routine(); /* let the GUI do its work */
```

Place a label in the screen, centered and showing LVGL information.

```
    /* Create simple label */
    lv_obj_t *label = lv_label_create( lv_scr_act() );
    lv_label_set_text( label, LVGL_Arduino.c_str() );
    lv_obj_align( label, LV_ALIGN_CENTER, 0, 0 );
```

We placed the LVGL configuration code for the screen in the display file. Below is an introduction to the display file.

Add header files.

```
#include "display.h"
#include "TFT_eSPI.h"
#include "FT6336U.h"
```

Define two variables to record the resolution of the screen.

```
static const uint16_t screenWidth  = 240;
static const uint16_t screenHeight = 320;
```

Create a screen display object tft, a touch screen object ft6336u, and a touch screen data type variable tp.

```
TFT_eSPI tft = TFT_eSPI(screenWidth, screenHeight); /* TFT instance */
FT6336U ft6336u(I2C_SDA, I2C_SCL, RST_N_PIN, INT_N_PIN);
FT6336U_TouchPointType tp;
```

Every once in a while, lvgl will check for any changes in the displayed content. If changes are detected, the function will be called to refresh the screen.

```
void my_disp_flush( lv_disp_drv_t *disp, const lv_area_t *area, lv_color_t *color_p ){
    uint32_t w = ( area->x2 - area->x1 + 1 );
    uint32_t h = ( area->y2 - area->y1 + 1 );
    tft.startWrite();
    tft.setAddrWindow( area->x1, area->y1, w, h );
    tft.pushColors((uint16_t*)&color_p->full, w * h, true );
    tft.endWrite();
    lv_disp_flush_ready( disp );
}
```

Every so often, LVGL will call this function to check if there have been any touch events on the screen.

```
void my_touchpad_read( lv_indev_drv_t * indev_driver, lv_indev_data_t * data ){
    uint16_t touchX, touchY;
    //bool touched = tft.getTouch( &touchX, &touchY, 600 );
    tp = ft6336u.scan();
    bool touched = tp.touch_count;
    if( !touched ){
        data->state = LV_INDEV_STATE_REL;
    }
    else{
        int x = tp.tp[0].x;
        int y = tp.tp[0].y;
        if(x >= 0 && x < screenWidth && y >= 0 && y < screenHeight) {
            data->state = LV_INDEV_STATE_PR;
            data->point.x = tp.tp[0].x;
            data->point.y = tp.tp[0].y;
        }
    }
}
```

Configuration function of LVGL adapted to the screen. With the following function, we can write any content based on LVGL and display it on the screen. We can also execute corresponding events based on touch events.

```
46  void Display::init(void)
47  {
48      ft6336u.begin();
49      lv_init();
50      tft.begin();          /* TFT init */
51      tft.setRotation( TFT_DIRECTION ); /* Landscape orientation, flipped */
52      lv_disp_draw_buf_init( &draw_buf, buf, NULL, screenWidth * 10 );
53
54      /*Initialize the display*/
55      static lv_disp_drv_t disp_drv;
56      lv_disp_drv_init( &disp_drv );
57      /*Change the following line to your display resolution*/
58      disp_drv.hor_res = screenWidth;
59      disp_drv.ver_res = screenHeight;
60      disp_drv.flush_cb = my_disp_flush;
61      disp_drv.draw_buf = &draw_buf;
62      lv_disp_drv_register( &disp_drv );
63
64      /*Initialize the (dummy) input device driver*/
65      static lv_indev_drv_t indev_drv;
66      lv_indev_drv_init( &indev_drv );
67      indev_drv.type = LV_INDEV_TYPE_POINTER;
68      indev_drv.read_cb = my_touchpad_read;
69      lv_indev_drv_register( &indev_drv );
70  }
```

Call the following function to keep refreshing the screen, and to get the state and data of the touch screen.

```
72  void Display::routine(void)
73  {
74      lv_task_handler();
75  }
```

For more information about LVGL, you can click on the link below to check it out.

https://docs.lvgl.io/8.1/

https://github.com/lvgl/lvgl/tree/release/v8.1

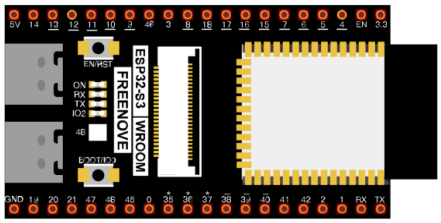
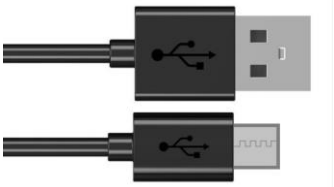
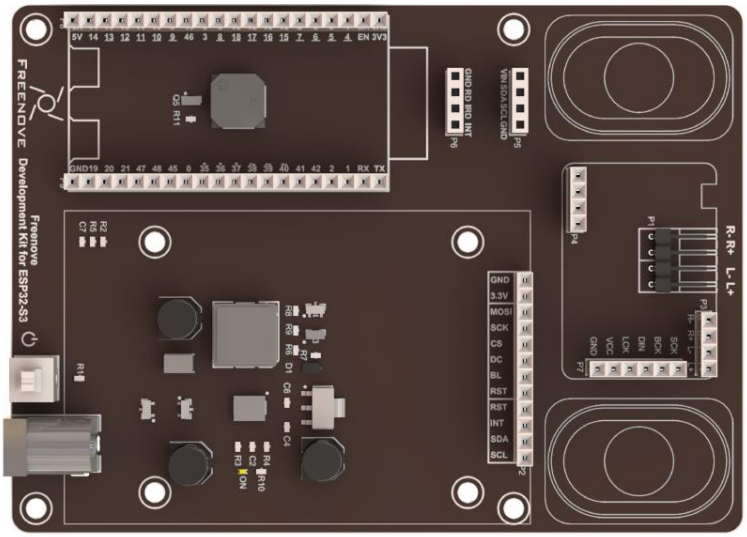
Chapter 10 LVGL Lable

In this chapter, we will learn how to use the label component on the screen.

Project 10.1 LVGL Lable

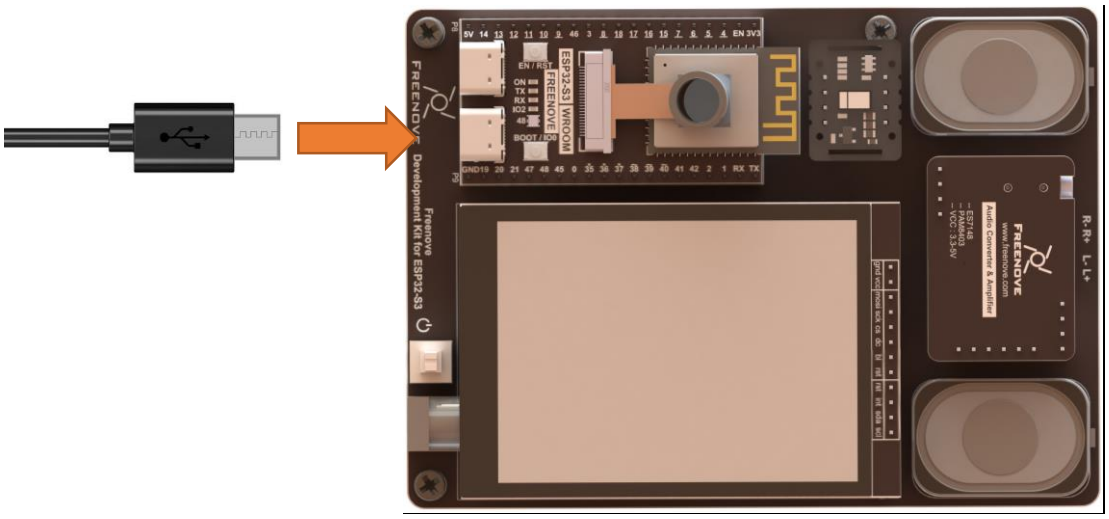
We have prepared two examples in the code, through which we will learn how to use the label component, how to display text in different colors, and how to add shadow effects to the text.

Component List

ESP32-S3 WROOM x1 	USB cable x1 	2.8-inch screen 
ESP32-S3 WROOM Shield x1 		

Circuit

Connect Freenove ESP32-S3 to the computer using the USB cable.
Hardware connection. If you need any support, please feel free to contact us via: support@freenove.com



Sketch

Sketch_10_LVGL_Lable

Sketch_10_LVGL_Label | Arduino IDE 2.0.4

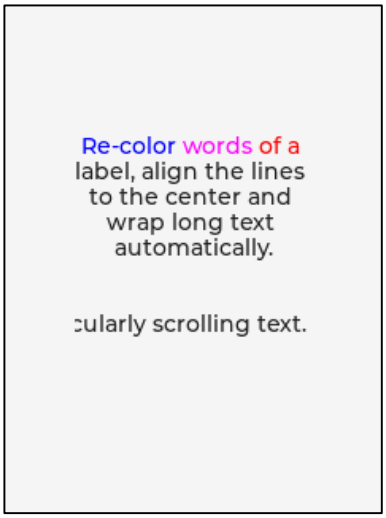
File Edit Sketch Tools Help

ESP32S3 Dev Module

Sketch_10_LVGL_Label.ino | display.cpp | display.h | lv_example_label.cpp | lv_example_label.h

```
#include <lvgl.h>
#include "Arduino.h"
#include "display.h"
#include "lv_example_label.h"

Display screen;

void setup() {
  Serial.begin(115200);

  /*** Init screen ***/
  screen.init();

  /***  Print lvgl version ***/
  String LVGL_Arduino = "Hello Arduino! ";
  LVGL_Arduino += String('V') + lv_version_major() + "." + lv_version_minor() + "." + lv_version_patch();
  Serial.println(LVGL_Arduino);
  Serial.println("I am LVGL_Arduino");
  Serial.println("Setup done");

  /*** The custom code ***/
  lv_example_label_1();//Recolor the text
  //lv_example_label_2();//Add a shadow to the text
}

void loop() {
  screen.routine();
  delay(5);
}
```

In the code, we have prepared two examples. After compiling and uploading the code with commented lines, the screen will display different content based on each example.
The following images show the display of the two examples.

lv_example_label_1	lv_example_label_2
	

The following is the program code:
Sketch_10_LVGL_Lable.ino

```
#include <lvgl.h>
#include "Arduino.h"
#include "display.h"
#include "lv_example_label.h"
Display screen;
void setup() {
  Serial.begin(115200);
  /*** Init screen ***/
  screen.init();
  /***  Print lvgl version ***/
  String LVGL_Arduino = "Hello Arduino! ";
  LVGL_Arduino += String('V') + lv_version_major() + "." + lv_version_minor() + "." +
lv_version_patch();
  Serial.println(LVGL_Arduino);
  Serial.println("I am LVGL_Arduino");
  Serial.println("Setup done");
  /*** The custom code ***/
  lv_example_label_1();//Recolor the text
  //lv_example_label_2();//Add a shadow to the text
}
void loop() {
  screen.routine();
  delay(5);
}
```

To use some libraries, first you need to include their header files.

```
#include <lvgl.h>
#include "Arduino.h"
#include "display.h"
#include "lv_example_label.h"
```

Request a screen operation object to operate the screen.

```
Display screen;
```

Initialize and configure the screen.

```
    screen.init();
```

Screen refresh function, by continuously calling this function, the interface can work properly.

```
    screen.routine(); /* let the GUI do its work */
```

Here are two examples of lables. We can comment out one of them to display the other. For the specific code, please refer to lv_example_lable.cpp and lv_example_lable.h files.

```
  /*** The custom code ***/
  lv_example_label_1();//Recolor the text
  //lv_example_label_2();//Add a shadow to the text
```

lv_example_lable.h

Declare two example functions so that they can be called in the ino file.

```
#ifndef __LV_EXAMPLE_LABLE_H
#define __LV_EXAMPLE_LABLE_H
void lv_example_label_1(void);
void lv_example_label_2(void);
#endif
```

lv_example_lable.cpp

Here is the complete code.

```
#include "lvgl.h"
#include "lv_example_label.h"

//Show line wrap, re-color, line align and text scrolling.
void lv_example_label_1(void)
{
    lv_obj_t * label1 = lv_label_create(lv_scr_act());
    /*
    LV_LABEL_LONG_WRAP,             Keep the width of the object, wrap the line if the text is too long, and enlarge the height of the object
    LV_LABEL_LONG_DOT,              Keep the text size, and if it's too long, put dots at the end
    LV_LABEL_LONG_SCROLL,           Keep the text size and scroll back and forth
    LV_LABEL_LONG_SCROLL_CIRCULAR,  Keep the text size and scroll the text
    LV_LABEL_LONG_CLIP,             Keep the size, and clip the text (keep only the front part of the display)
    */
```

```
18      /*Support long text, if the width is not enough to display, then continue to display the
19  newline*/
20      lv_label_set_long_mode(label1, LV_LABEL_LONG_WRAP);
21      lv_label_set_recolor(label1, true); /*Enable recolor through commands in the text*/
22      lv_label_set_text(label1, "#0000ff Re-color# #ff00ff words# #ff0000 of a # label, align
23  the lines to the center and wrap long text automatically.");
24      lv_obj_set_width(label1, 150); /*Set a small width to wrap lines automatically*/
25      lv_obj_set_style_text_align(label1, LV_TEXT_ALIGN_CENTER, 0); /*Set the label width to
26  center*/
27      lv_obj_align(label1, LV_ALIGN_CENTER, 0, -40); /*Set the label height to a distance of
28  between mid-40 pixels*/
29
30      lv_obj_t * label2 = lv_label_create(lv_scr_act());
31      lv_label_set_long_mode(label2, LV_LABEL_LONG_SCROLL);     /*Circular scroll*/
32      lv_obj_set_width(label2, 150);
33      lv_label_set_text(label2, "It is a circularly scrolling text. ");
34      lv_obj_align(label2, LV_ALIGN_CENTER, 0, 40);
35  }
36
37  //Create a fake text shadow
38  void lv_example_label_2(void)
39  {
40      /*Create a style for the shadow*/
41      static lv_style_t style_shadow;//Apply for a style object
42      lv_style_init(&style_shadow);  //Initialize the object
43      lv_style_set_text_opa(&style_shadow, LV_OPA_30);//Set the opacity to 30
44      lv_style_set_text_color(&style_shadow, lv_color_black());//Set the text color to black
45
46      /*Create a label for the shadow first (it's in the background)*/
47      lv_obj_t * shadow_label = lv_label_create(lv_scr_act());
48      lv_obj_add_style(shadow_label, &style_shadow, 0);
49      /*Create the main label*/
50      lv_obj_t * main_label = lv_label_create(lv_scr_act());
51      lv_label_set_text(main_label, "A simple method to create\n"
52                        "shadows on a text.\n"
53                        "It even works with\n\n"
54                        "newlines     and spaces.");
55      /*Set the same text for the shadow label*/
56      lv_label_set_text(shadow_label, lv_label_get_text(main_label));
57      /*Position the main label*/
58      lv_obj_align(main_label, LV_ALIGN_CENTER, 0, 0);
59      /*Shift the second label down and to the right by 2 pixel*/
60      lv_obj_align_to(shadow_label, main_label, LV_ALIGN_TOP_LEFT, 2, 2);
61  }
```

Create a label component in the current interface and assign it to a component type pointer variable label1.

```
7   lv_obj_t * label1 = lv_label_create(lv_scr_act());
```

Sets the mode of this label component.

```
20      lv_label_set_long_mode(label1, LV_LABEL_LONG_WRAP);
```

To re-customize the color of the font, please refer to the following two sentences.

```
21      lv_label_set_recolor(label1, true); /*Enable recolor through commands in the text*/
22      lv_label_set_text(label1, "#0000ff Re-color# #ff00ff words# #ff0000 of a # label, align
    the lines to the center and wrap long text automatically.");
```

Set the display width of the label component. If the content to be displayed exceeds this length, the remaining text will be displayed according to the default settings of the label component.

```
24      lv_obj_set_width(label1, 150); /*Set a small width to wrap lines automatically*/
```

Set the text content of the label component to be displayed in the center.

```
25      lv_obj_set_style_text_align(label1, LV_TEXT_ALIGN_CENTER, 0);
```

Set the label component to be located 40 pixels down from the center of the screen as a reference point.

```
27      lv_obj_align(label1, LV_ALIGN_CENTER, 0, -40);
```

Declare a variable of type "style" and name it "style_shadow", then initialize this variable.

```
41      static lv_style_t style_shadow;//Apply for a style object
42      lv_style_init(&style_shadow);  //Initialize the object
```

Set the opacity of style_shadow to 30% and the color to black.

```
43      lv_style_set_text_opa(&style_shadow, LV_OPA_30);//Set the opacity to 30
44      lv_style_set_text_color(&style_shadow, lv_color_black());//Set the text color to black
```

Create a label component to serve as the shadow effect with the style set to style_shadow.

```
47      lv_obj_t * shadow_label = lv_label_create(lv_scr_act());
48      lv_obj_add_style(shadow_label, &style_shadow, 0);
```

Define a label component to display the main text and set its content.

```
50      lv_obj_t * main_label = lv_label_create(lv_scr_act());
51      lv_label_set_text(main_label, "A simple method to create\n"
52                        "shadows on a text.\n"
53                        "It even works with\n\n"
54                        "newlines     and spaces.");
```

Read the content of the main_lable1 component and copy the content to the shadow_label component

```
56      lv_label_set_text(shadow_label, lv_label_get_text(main_label));
```

Display the content of the main_lable1 component in the center of the screen, and use main_lable1 as a reference to display the content of shadow_label1 2 pixels below and 2 pixels to the right of main_lable1, creating a shadow effect.

```
57      /*Position the main label*/
58      lv_obj_align(main_label, LV_ALIGN_CENTER, 0, 0);
59      /*Shift the second label down and to the right by 2 pixel*/
60      lv_obj_align_to(shadow_label, main_label, LV_ALIGN_TOP_LEFT, 2, 2);
```

To learn more abou LVGL, please refer to the link below:
https://docs.lvgl.io/8.1/widgets/core/label.html

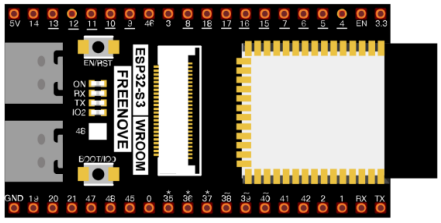
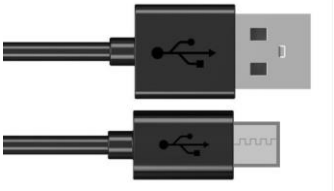
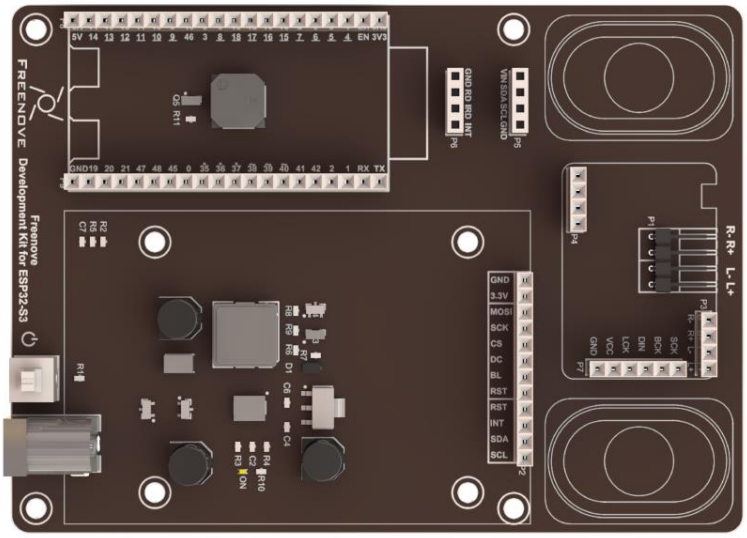
Chapter 11 LVGL Button

In this Chapter, we will learn how to use the button component on the screen

Project 11.1 LVGL Button

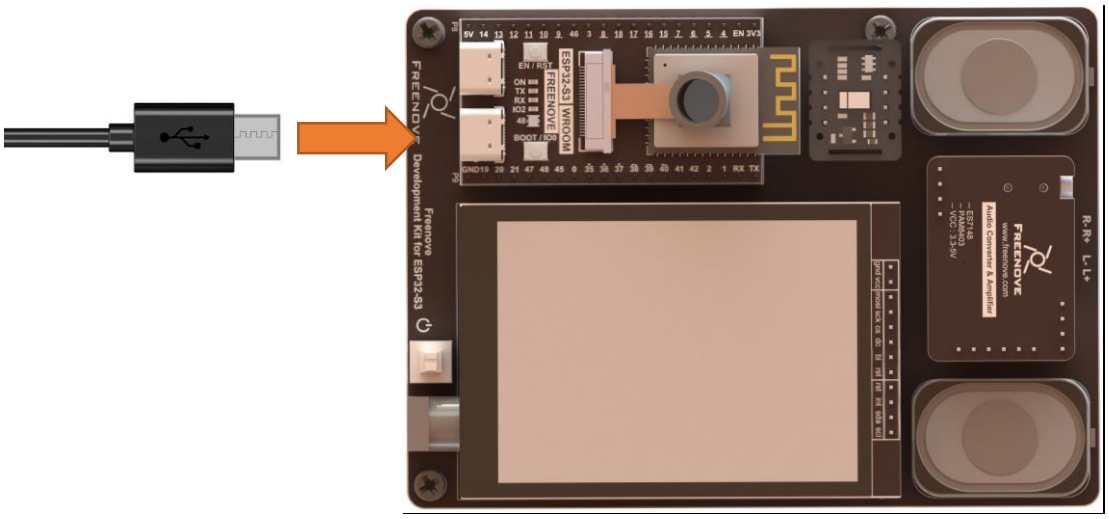
We have prepared three examples in the code to learn how to use the button component.

Component List

ESP32-S3 WROOM x1 	USB cable x1 	2.8-inch screen 
ESP32-S3 WROOM Shield x1 		

Circuit

Connect Freenove ESP32-S3 to the computer using the USB cable.
Hardware connection. If you need any support, please feel free to contact us via: support@freenove.com



Sketch

Sketch_11_LVGL_Btn

Sketch_11_LVGL_Btn | Arduino IDE 2.0.4

File Edit Sketch Tools Help

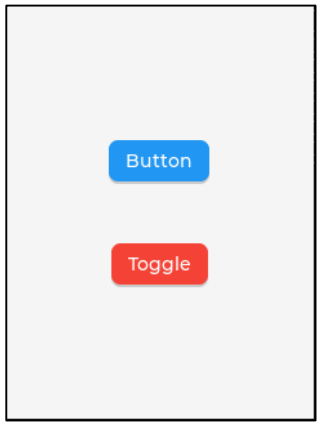
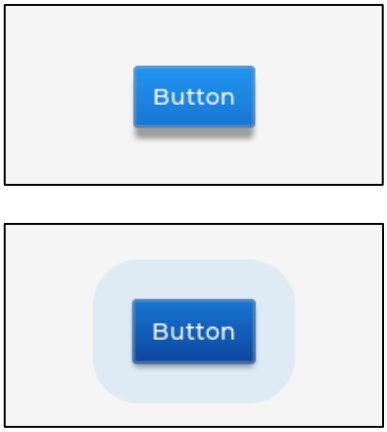
ESP32S3 Dev Module

Sketch_11_LVGL_Btn.ino | display.cpp | display.h | lv_example_btn.cpp | lv_example_btn.h

```
#include <lvgl.h>
#include "Arduino.h"
#include "display.h"
#include "lv_example_btn.h"

Display screen;

void setup() {
  Serial.begin(115200);

  /*** Init screen ***/
  screen.init();

  /***  Print lvgl version ***/
  String LVGL_Arduino = "Hello Arduino! ";
  LVGL_Arduino += String('V') + lv_version_major() + "." + lv_version_minor() + "." + lv_version_patch();
  Serial.println(LVGL_Arduino);
  Serial.println("I am LVGL_Arduino");
  Serial.println("Setup done");

  /*** The custom code ***/
  lv_example_btn_1();//Two types of button trigger events
  //lv_example_btn_2();//Style a button from scratch
  //lv_example_btn_3();//Create a style transition on a button to act like a gum when clicked

}
```

After commenting out the code, compiling and uploading it, different content can be displayed on the screen.

Below are displaying examples

lv_example_btn_1	lv_example_btn_2	lv_example_btn_3
		

The following is the program code:

Sketch_11_LVGL_Btn.ino

```
#include <lvgl.h>
#include "Arduino.h"
#include "display.h"
#include "lv_example_label.h"
Display screen;
void setup() {
  Serial.begin(115200);
  /*** Init screen ***/
  screen.init();
  /***  Print lvgl version ***/
  String LVGL_Arduino = "Hello Arduino! ";
  LVGL_Arduino += String('V') + lv_version_major() + "." + lv_version_minor() + "." +
lv_version_patch();
  Serial.println(LVGL_Arduino);
  Serial.println("I am LVGL_Arduino");
  Serial.println("Setup done");
  /*** The custom code ***/
  lv_example_btn_1();//Two types of button trigger events
  //lv_example_btn_2();//Style a button from scratch
  //lv_example_btn_3();//Create a style transition on a button to act like a gum when clicked
}
void loop() {
  screen.routine();
  delay(5);
}
```

There are three examples of button components. We can comment out two of them and remain one to display on the screen. The specific code can be found in the lv_example_btn.cpp and lv_example_btn.h files.

```
  /*** The custom code ***/
  lv_example_btn_1();//Two types of button trigger events
  //lv_example_btn_2();//Style a button from scratch
  //lv_example_btn_3();//Create a style transition on a button to act like a gum when clicked
```

lv_example_btn.h

Declare threee functions so that they can be called in the ino file.

```
#ifndef __LV_EXAMPLE_BTN_H
#define __LV_EXAMPLE_BTN_H
void lv_example_btn_1(void);
void lv_example_btn_2(void);
void lv_example_btn_3(void);
#endif
```

lv_example_btn.cpp

Here is the complete code.

```
#include "lvgl.h"
#include "lv_example_btn.h"

/*************** Example 1 ***************************/
static void event_handler(lv_event_t * e){
  lv_event_code_t code = lv_event_get_code(e);
  if (code == LV_EVENT_CLICKED) {
    LV_LOG_USER("Clicked");
  }
  else if (code == LV_EVENT_VALUE_CHANGED) {
    LV_LOG_USER("Toggled");
  }
}
//Two types of button trigger events
void lv_example_btn_1(void){
  lv_obj_t * label;
  lv_obj_t * btn1 = lv_btn_create(lv_scr_act());
  lv_obj_add_event_cb(btn1, event_handler, LV_EVENT_ALL, NULL);
  lv_obj_align(btn1, LV_ALIGN_CENTER, 0, -40);
  label = lv_label_create(btn1);
  lv_label_set_text(label, "Button");
  lv_obj_center(label);

  lv_obj_t * btn2 = lv_btn_create(lv_scr_act());
  lv_obj_add_event_cb(btn2, event_handler, LV_EVENT_ALL, NULL);
  lv_obj_align(btn2, LV_ALIGN_CENTER, 0, 40);
```

```
27    lv_obj_add_flag(btn2, LV_OBJ_FLAG_CHECKABLE);
28    lv_obj_set_height(btn2, LV_SIZE_CONTENT);
29    label = lv_label_create(btn2);
30    lv_label_set_text(label, "Toggle");
31    lv_obj_center(label);
32  }
33
34  /********** Example 2 **************************/
35  //Style a button from scratch
36  void lv_example_btn_2(void){
37    /*Init the style for the default state*/
38    static lv_style_t style;
39    lv_style_init(&style);
40    lv_style_set_radius(&style, 3);
41    lv_style_set_bg_opa(&style, LV_OPA_100);
42    lv_style_set_bg_color(&style, lv_palette_main(LV_PALETTE_BLUE));
43    lv_style_set_bg_grad_color(&style, lv_palette_darken(LV_PALETTE_BLUE, 2));
44    lv_style_set_bg_grad_dir(&style, LV_GRAD_DIR_VER);
45    lv_style_set_border_opa(&style, LV_OPA_40);
46    lv_style_set_border_width(&style, 2);
47    lv_style_set_border_color(&style, lv_palette_main(LV_PALETTE_GREY));
48    lv_style_set_shadow_width(&style, 8);
49    lv_style_set_shadow_color(&style, lv_palette_main(LV_PALETTE_GREY));
50    lv_style_set_shadow_ofs_y(&style, 8);
51    lv_style_set_outline_opa(&style, LV_OPA_COVER);
52    lv_style_set_outline_color(&style, lv_palette_main(LV_PALETTE_BLUE));
53    lv_style_set_text_color(&style, lv_color_white());
54    lv_style_set_pad_all(&style, 10);
55
56    /*Init the pressed style*/
57    static lv_style_t style_pr;
58    lv_style_init(&style_pr);
59    /*Add a large outline when pressed*/
60    lv_style_set_outline_width(&style_pr, 30);
61    lv_style_set_outline_opa(&style_pr, LV_OPA_TRANSP);
62    lv_style_set_translate_y(&style_pr, 5);
63    lv_style_set_shadow_ofs_y(&style_pr, 3);
64    lv_style_set_bg_color(&style_pr, lv_palette_darken(LV_PALETTE_BLUE, 2));
65    lv_style_set_bg_grad_color(&style_pr, lv_palette_darken(LV_PALETTE_BLUE, 4));
66
67    /*Add a transition to the outline*/
68    static lv_style_transition_dsc_t trans;
69    static lv_style_prop_t props[] = {LV_STYLE_OUTLINE_WIDTH, LV_STYLE_OUTLINE_OPA,
70  (lv_style_prop_t)0};
```

```
71    lv_style_transition_dsc_init(&trans, props, lv_anim_path_linear, 300, 0, NULL);
72    lv_style_set_transition(&style_pr, &trans);
73
74    lv_obj_t * btn1 = lv_btn_create(lv_scr_act());
75    lv_obj_remove_style_all(btn1);  /*Remove the style coming from the theme*/
76    lv_obj_add_style(btn1, &style, LV_STATE_DEFAULT);
77    lv_obj_add_style(btn1, &style_pr, LV_STATE_PRESSED);
78    lv_obj_set_size(btn1, LV_SIZE_CONTENT, LV_SIZE_CONTENT);
79    lv_obj_center(btn1);
80
81    lv_obj_t * label = lv_label_create(btn1);
82    lv_label_set_text(label, "Button");
83    lv_obj_center(label);
84  }
85
86  /*************** Example 3 ***************************/
87  //Create a style transition on a button to act like a gum when clicked
88  void lv_example_btn_3(void){
89    /*Properties to transition*/
90    static lv_style_prop_t props[] = {LV_STYLE_TRANSFORM_WIDTH, LV_STYLE_TRANSFORM_HEIGHT,
91  LV_STYLE_TEXT_LETTER_SPACE, (lv_style_prop_t)0};
92    /*Transition descriptor when going back to the default state.
93      Add some delay to be sure the press transition is visible even if the press was very
94  short*/
95    static lv_style_transition_dsc_t transition_dsc_def;
96    lv_style_transition_dsc_init(&transition_dsc_def, props, lv_anim_path_overshoot, 250, 100,
97  NULL);
98    /*Transition descriptor when going to pressed state.
99      No delay, go to presses state immediately*/
100   static lv_style_transition_dsc_t transition_dsc_pr;
101   lv_style_transition_dsc_init(&transition_dsc_pr, props, lv_anim_path_ease_in_out, 250, 0,
102 NULL);
103   /*Add only the new transition to he default state*/
104   static lv_style_t style_def;
105   lv_style_init(&style_def);
106   lv_style_set_transition(&style_def, &transition_dsc_def);
107
108   /*Add the transition and some transformation to the presses state.*/
109   static lv_style_t style_pr;
110   lv_style_init(&style_pr);
111   lv_style_set_transform_width(&style_pr, 10);
112   lv_style_set_transform_height(&style_pr, -10);
113   lv_style_set_text_letter_space(&style_pr, 10);
114   lv_style_set_transition(&style_pr, &transition_dsc_pr);
```

```
115
116     lv_obj_t * btn1 = lv_btn_create(lv_scr_act());
117     lv_obj_align(btn1, LV_ALIGN_CENTER, 0, -20);
118     lv_obj_add_style(btn1, &style_pr, LV_STATE_PRESSED);
119     lv_obj_add_style(btn1, &style_def, 0);
120
121     lv_obj_t * label = lv_label_create(btn1);
122     lv_label_set_text(label, "Gum");
123 }
```

Create a button component in the current interface and assign it to a component type pointer variable btn1.

```
17    lv_obj_t * btn1 = lv_btn_create(lv_scr_act());
```

Associate the button btn1 with a callback function event_handler to be triggered when the button is pressed.

```
18    lv_obj_add_event_cb(btn1, event_handler, LV_EVENT_ALL, NULL);
```

Display the button btn1 40 pixels above the center of the screen.

```
19    lv_obj_align(btn1, LV_ALIGN_CENTER, 0, -40);
```

Add a label component to button btn1 with the text "Button" displayed at the center of the button btn1.

```
20    label = lv_label_create(btn1);
21    lv_label_set_text(label, "Button");
22    lv_obj_center(label);
```

Set the button btn2 to have a selected toggle state effect, and set it to default height and width.

```
27    lv_obj_add_flag(btn2, LV_OBJ_FLAG_CHECKABLE);
28    lv_obj_set_height(btn2, LV_SIZE_CONTENT);
```

Create a style object and use it to set the component to have a background color gradient from blue to dark blue, with a 3-pixel rounded corner and 100% opacity.

```
37    /*Init the style for the default state*/
38    static lv_style_t style;
39    lv_style_init(&style);
40    lv_style_set_radius(&style, 3);
41    lv_style_set_bg_opa(&style, LV_OPA_100);
42    lv_style_set_bg_color(&style, lv_palette_main(LV_PALETTE_BLUE));
43    lv_style_set_bg_grad_color(&style, lv_palette_darken(LV_PALETTE_BLUE, 2));
44    lv_style_set_bg_grad_dir(&style, LV_GRAD_DIR_VER);
```

Apply a style to the component with 40% non-transparent border opacity, 2-pixel border width, and gray border color.

```
45    lv_style_set_border_opa(&style, LV_OPA_40);
46    lv_style_set_border_width(&style, 2);
47    lv_style_set_border_color(&style, lv_palette_main(LV_PALETTE_GREY));
```

Set the shadow of the component style to occupy 8 pixel rows, with the color gray, and the shadow offset relative to the main body is 8 pixel rows downward.

```
48    lv_style_set_shadow_width(&style, 8);
49    lv_style_set_shadow_color(&style, lv_palette_main(LV_PALETTE_GREY));
50    lv_style_set_shadow_ofs_y(&style, 8);
```

Set the outline of the component style to be fully opaque, and set the outline color to blue.

```
51    lv_style_set_outline_opa(&style, LV_OPA_COVER);
52    lv_style_set_outline_color(&style, lv_palette_main(LV_PALETTE_BLUE));
```

Set the component style font color to white and set the font distance from the component's border to 10 pixels.

```
53    lv_style_set_text_color(&style, lv_color_white());
54    lv_style_set_pad_all(&style, 10);
```

Set the outline of component style_pr to 30 pixels and the outline opacity to fully transparent.

```
60    lv_style_set_outline_width(&style_pr, 30);
61    lv_style_set_outline_opa(&style_pr, LV_OPA_TRANSP);
```

Set the component style_pr to have a downward offset of 5 pixels for the main body and 3 pixels for the shadow.

```
62    lv_style_set_translate_y(&style_pr, 5);
63    lv_style_set_shadow_ofs_y(&style_pr, 3);
```

Set the background color to be 2 shades darker than the normal blue color. Set the background gradient color to be 4 shades darker than the normal blue color.

```
64    lv_style_set_bg_color(&style_pr, lv_palette_darken(LV_PALETTE_BLUE, 2));
65    lv_style_set_bg_grad_color(&style_pr, lv_palette_darken(LV_PALETTE_BLUE, 4));
```

Apply for a transition object, set the transition time to 300ms, and associate it with the style_pr component.

```
68    static lv_style_transition_dsc_t trans;
69    static lv_style_prop_t props[] = {LV_STYLE_OUTLINE_WIDTH, LV_STYLE_OUTLINE_OPA,
      (lv_style_prop_t)0};
70    lv_style_transition_dsc_init(&trans, props, lv_anim_path_linear, 300, 0, NULL);
71    lv_style_set_transition(&style_pr, &trans);
```

Create a button object btn1 and remove all default styles. Set the style of the button in its normal state to style. When the button is pressed, trigger a transition effect from style to style_pr. Set the button to default width and height, and display it in the center of the screen.

```
73    lv_obj_t * btn1 = lv_btn_create(lv_scr_act());
74    lv_obj_remove_style_all(btn1);  /*Remove the style coming from the theme*/
75    lv_obj_add_style(btn1, &style, LV_STATE_DEFAULT);
76    lv_obj_add_style(btn1, &style_pr, LV_STATE_PRESSED);
77    lv_obj_set_size(btn1, LV_SIZE_CONTENT, LV_SIZE_CONTENT);
78    lv_obj_center(btn1);
```

Set the width of the style_pr component to double its original size, decrease its height to half of its original size, and increase the font spacing by double.

```
106   lv_style_set_transform_width(&style_pr, 10);
107   lv_style_set_transform_height(&style_pr, -10);
108   lv_style_set_text_letter_space(&style_pr, 10);
```

To learn more about LVGL, please refer to
https://docs.lvgl.io/8.1/widgets/core/btn.html

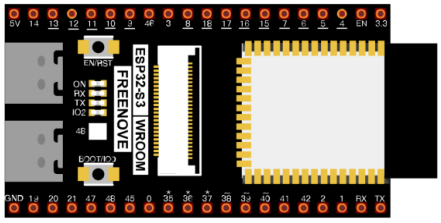
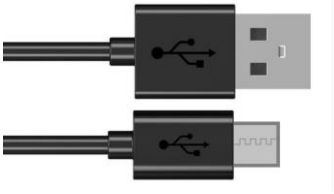
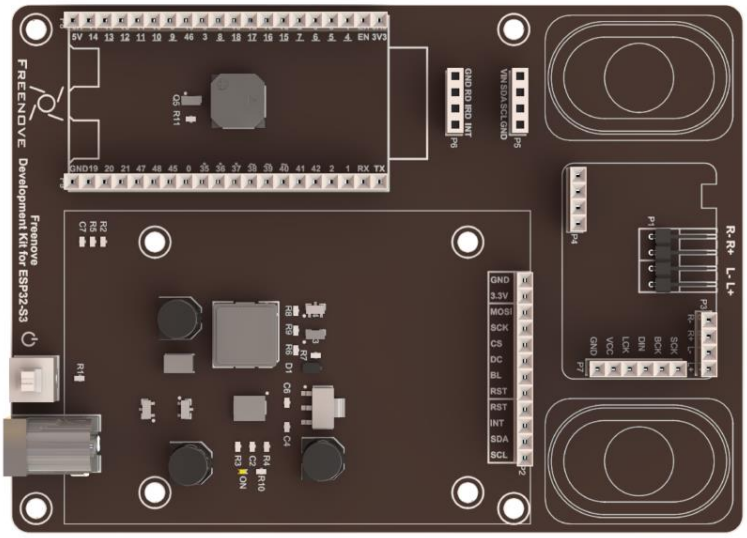
Chapter 12 LVGL Slider

In this Chapter, we will learn how to use the Slider component on the screen.

Project 12.1 LVGL Slider

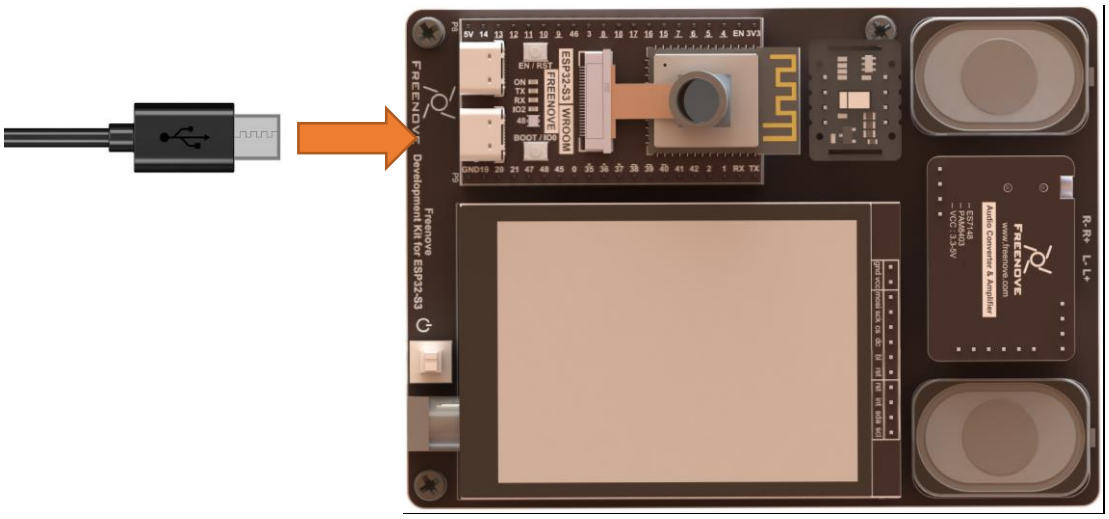
We have prepared three examples in the code, through which we will learn how to use the Slider component.

Component List

ESP32-S3 WROOM x1 	USB cable x1 	2.8-inch screen 
ESP32-S3 WROOM Shield x1 		

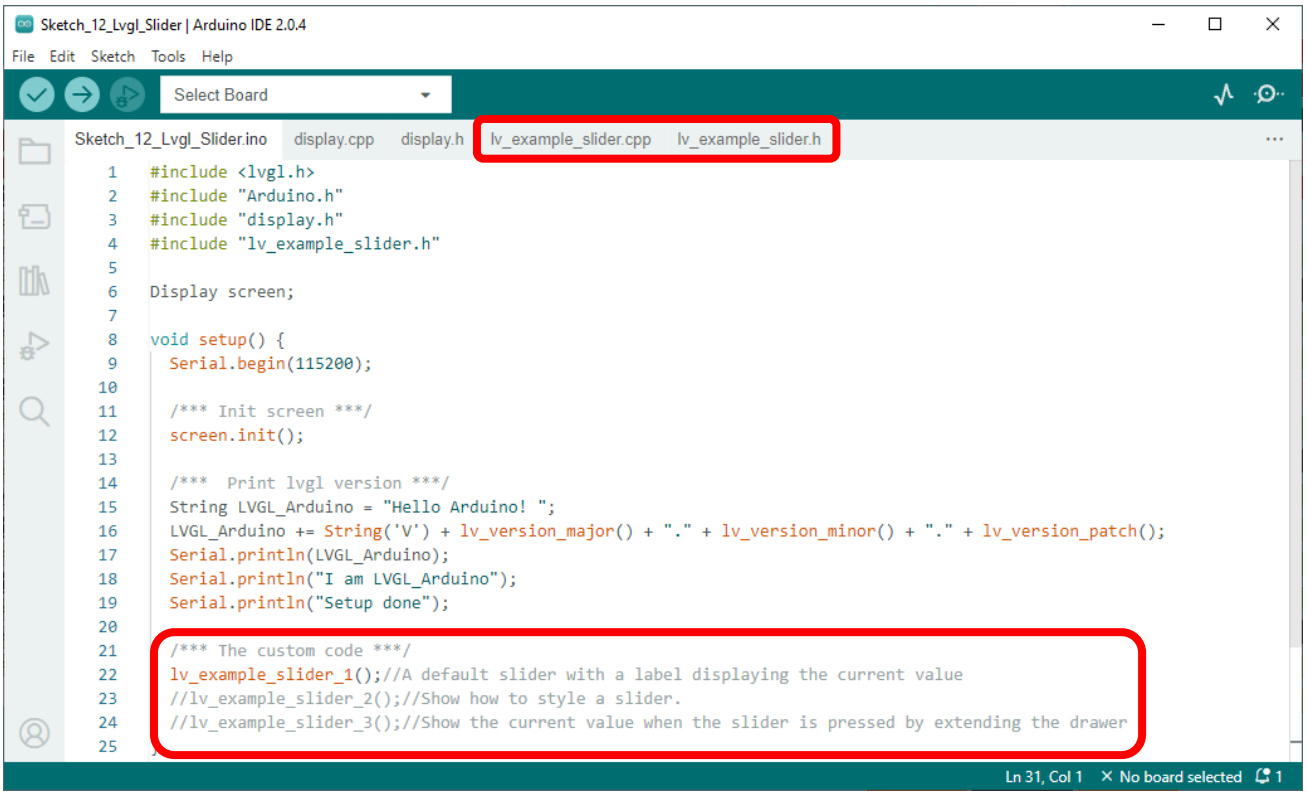
Circuit

Connect Freenove ESP32-S3 to the computer using the USB cable.
Hardware connection. If you need any support, please feel free to contact us via: support@freenove.com



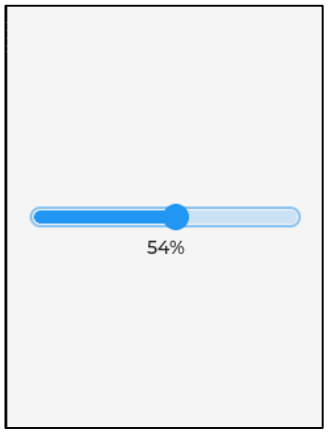
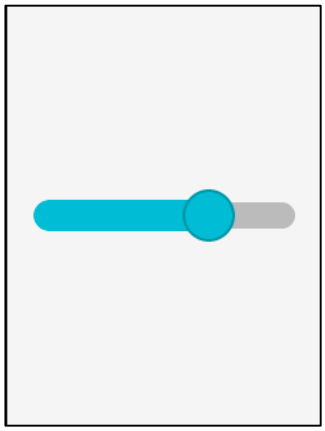
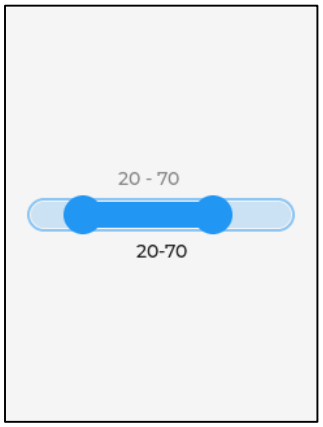
Sketch

Sketch_12_LVGL_Slider



After commenting out the code, compiling and uploading, the screen will display different contents depending on which example is uncommented.

Here are illustrations of the examples.

lv_example_btn_1	lv_example_btn_2	lv_example_btn_3
		

The following is the program code:

Sketch_12_LVGL_Slider.ino

```
#include <lvgl.h>
#include "Arduino.h"
#include "display.h"
#include "lv_example_label.h"
Display screen;
void setup() {
  Serial.begin(115200);
  /*** Init screen ***/
  screen.init();
  /***  Print lvgl version ***/
  String LVGL_Arduino = "Hello Arduino! ";
  LVGL_Arduino += String('V') + lv_version_major() + "." + lv_version_minor() + "." +
lv_version_patch();
  Serial.println(LVGL_Arduino);
  Serial.println("I am LVGL_Arduino");
  Serial.println("Setup done");
  /*** The custom code ***/
  lv_example_slider_1();//A default slider with a label displaying the current value
  //lv_example_slider_2();//Show how to style a slider.
  //lv_example_slider_3();//Show the current value when the slider is pressed by extending the
drawer
}
void loop() {
  screen.routine();
  delay(5);
}
```

Here are three examples of Slider. We can comment out two of them to display the remaining one. You can refer to the lv_example_slider.cpp and lv_example_slider.h files for the specific code.

```
17    lv_example_slider_1();//A default slider with a label displaying the current value
18    //lv_example_slider_2();//Show how to style a slider.
19    //lv_example_slider_3();//Show the current value when the slider is pressed by extending the
20  drawer
```

lv_example_slider.h

Declare three functions so that they can be called in the ino file.

```
1   #ifndef __LV_EXAMPLE_SLIDER_H
2   #define __LV_EXAMPLE_SLIDER_H
3   void lv_example_slider_1(void);
4   void lv_example_slider_2(void);
5   void lv_example_slider_3(void);
6   #endif
```

lv_example_slider.cpp

Here is the complete code.

```
1   #include "lvgl.h"
2   #include "lv_example_slider.h"
3
4   /*************** Example 1 ***************************/
5   static lv_obj_t * slider_label;
6   static void slider_event_cb_1(lv_event_t * e){
7       lv_obj_t * slider = lv_event_get_target(e);
8       char buf[8];
9       lv_snprintf(buf, sizeof(buf), "%d%%", (int)lv_slider_get_value(slider));
10      lv_label_set_text(slider_label, buf);
11      lv_obj_align_to(slider_label, slider, LV_ALIGN_OUT_BOTTOM_MID, 0, 10);
12  }
13
14  //A default slider with a label displaying the current value
15  void lv_example_slider_1(void){
16      /*Create a slider in the center of the display*/
17      lv_obj_t * slider = lv_slider_create(lv_scr_act());
18      lv_obj_center(slider);
19      lv_obj_set_size(slider, 200, 10);
20      lv_obj_add_event_cb(slider, slider_event_cb_1, LV_EVENT_VALUE_CHANGED, NULL);
21
22      /*Create a label below the slider*/
23      slider_label = lv_label_create(lv_scr_act());
24      lv_label_set_text(slider_label, "0%");
25
26      lv_obj_align_to(slider_label, slider, LV_ALIGN_OUT_BOTTOM_MID, 0, 10);
```

```
}

//Show how to style a slider.
void lv_example_slider_2(void){
    /*Create a transition*/
    static const lv_style_prop_t props[] = {LV_STYLE_BG_COLOR, (lv_style_prop_t)0};
    static lv_style_transition_dsc_t transition_dsc;
    lv_style_transition_dsc_init(&transition_dsc, props, lv_anim_path_linear, 300, 0, NULL);

    static lv_style_t style_main;
    static lv_style_t style_indicator;
    static lv_style_t style_knob;
    static lv_style_t style_pressed_color;
    lv_style_init(&style_main);
    lv_style_set_bg_opa(&style_main, LV_OPA_COVER);
    lv_style_set_bg_color(&style_main, lv_color_hex3(0xbbb));
    lv_style_set_radius(&style_main, LV_RADIUS_CIRCLE);
    lv_style_set_pad_ver(&style_main, -2); /*Makes the indicator larger*/

    lv_style_init(&style_indicator);
    lv_style_set_bg_opa(&style_indicator, LV_OPA_COVER);
    lv_style_set_bg_color(&style_indicator, lv_palette_main(LV_PALETTE_BLUE));
    lv_style_set_radius(&style_indicator, LV_RADIUS_CIRCLE);
    lv_style_set_transition(&style_indicator, &transition_dsc);

    lv_style_init(&style_knob);
    lv_style_set_bg_opa(&style_knob, LV_OPA_COVER);
    lv_style_set_bg_color(&style_knob, lv_palette_main(LV_PALETTE_CYAN));
    lv_style_set_border_color(&style_knob, lv_palette_darken(LV_PALETTE_CYAN, 2));
    lv_style_set_border_width(&style_knob, 2);
    lv_style_set_radius(&style_knob, LV_RADIUS_CIRCLE);
    lv_style_set_pad_all(&style_knob, 10); /*Makes the knob larger*/
    lv_style_set_transition(&style_knob, &transition_dsc);

    lv_style_init(&style_pressed_color);
    lv_style_set_bg_color(&style_pressed_color, lv_palette_darken(LV_PALETTE_CYAN, 2));
    /*Create a slider and add the style*/
    lv_obj_t * slider = lv_slider_create(lv_scr_act());
    lv_obj_remove_style_all(slider);        /*Remove the styles coming from the theme*/
    lv_obj_add_style(slider, &style_main, LV_PART_MAIN);//Set the slider base color to gray
    lv_obj_add_style(slider, &style_indicator, LV_PART_INDICATOR);//Set the slider color to
cyan
    lv_obj_add_style(slider, &style_knob, LV_PART_KNOB);//Set the slider ball handle larger
and more prominent
```

```
    lv_obj_add_style(slider, &style_pressed_color, LV_PART_INDICATOR | LV_PART_KNOB | LV_STATE_PRESSED);// When pressed, the slider color is 2 color numbers darker

    lv_obj_set_size(slider, 200, 20);
    lv_obj_center(slider);
}

/*************** Example 3 ***************************/
static void slider_event_cb_3(lv_event_t * e){
    lv_event_code_t code = lv_event_get_code(e);
    lv_obj_t *slider = lv_event_get_target(e);

    /*Provide some extra space for the value*/
    if(code == LV_EVENT_REFR_EXT_DRAW_SIZE) {
        lv_event_set_ext_draw_size(e, 50);
    }
    else if(code == LV_EVENT_DRAW_PART_END) {
        lv_obj_draw_part_dsc_t * dsc = lv_event_get_draw_part_dsc(e);
        if(dsc->part == LV_PART_INDICATOR) {
            char buf[16];
            lv_snprintf(buf, sizeof(buf), "%d - %d", (int)lv_slider_get_left_value(slider), (int)lv_slider_get_value(slider));
            LV_LOG_USER("Slider Value Range: %s",buf);
            lv_label_set_text(slider_label, buf);
            lv_obj_align_to(slider_label, slider, LV_ALIGN_OUT_BOTTOM_MID, 0, 10);
        }
    }
}

//Show the current value when the slider is pressed by extending the drawer
void lv_example_slider_3(void){
    /*Create a slider in the center of the display*/
    lv_obj_t * slider;
    slider = lv_slider_create(lv_scr_act());

    lv_obj_set_size(slider, 200, 20);
    lv_obj_center(slider);

    lv_slider_set_mode(slider, LV_SLIDER_MODE_RANGE);
    lv_slider_set_value(slider, 70, LV_ANIM_OFF);
    lv_slider_set_left_value(slider, 20, LV_ANIM_OFF);

    lv_obj_add_event_cb(slider, slider_event_cb_3, LV_EVENT_ALL, NULL);
    lv_obj_refresh_ext_draw_size(slider);
```

```
112
113     slider_label = lv_label_create(lv_scr_act());
114     lv_label_set_text(slider_label, "20-70");
115
116     lv_obj_align_to(slider_label, slider, LV_ALIGN_OUT_BOTTOM_MID, 0, 10);
117 }
```

Create a slider component with a width of 200 pixels and a height of 10 pixels, center it on the screen, and associate it with the callback function slider_event_cb_1.

```
14  //A default slider with a label displaying the current value
15  void lv_example_slider_1(void){
16      /*Create a slider in the center of the display*/
17      lv_obj_t * slider = lv_slider_create(lv_scr_act());
18      lv_obj_center(slider);
19      lv_obj_set_size(slider, 200, 10);
20      lv_obj_add_event_cb(slider, slider_event_cb_1, LV_EVENT_VALUE_CHANGED, NULL);
21
22      /*Create a label below the slider*/
23      slider_label = lv_label_create(lv_scr_act());
24      lv_label_set_text(slider_label, "0%");
25
26      lv_obj_align_to(slider_label, slider, LV_ALIGN_OUT_BOTTOM_MID, 0, 10);
27  }
```

Get the object responsible for triggering the slider event and assign it to the variable "slider".

```
7       lv_obj_t * slider = lv_event_get_target(e);
```

Get the value of the slider, use the lv_snprintf() function to concatenate the data together, and display the content on a label.

```
8       char buf[8];
9       lv_snprintf(buf, sizeof(buf), "%d%%", (int)lv_slider_get_value(slider));
10      lv_label_set_text(slider_label, buf);
11      lv_obj_align_to(slider_label, slider, LV_ALIGN_OUT_BOTTOM_MID, 0, 10);
```

Create a background color transition object with a transition time of 300ms.

```
32      static const lv_style_prop_t props[] = {LV_STYLE_BG_COLOR, (lv_style_prop_t)0};
33      static lv_style_transition_dsc_t transition_dsc;
34      lv_style_transition_dsc_init(&transition_dsc, props, lv_anim_path_linear, 300, 0, NULL);
```

Set the "style_main" component's background to fully opaque, the background color to gray, and to rounded cornered. Also, make the indicator two pixels larger than the slider on the top and bottom.

```
40      lv_style_init(&style_main);
41      lv_style_set_bg_opa(&style_main, LV_OPA_COVER);
42      lv_style_set_bg_color(&style_main, lv_color_hex3(0xbbb));
43      lv_style_set_radius(&style_main, LV_RADIUS_CIRCLE);
44      lv_style_set_pad_ver(&style_main, -2); /*Makes the indicator larger*/
```

Set the background opacity of style_indicator component to fully opaque, set the background color to blue, set the component to rounded corner, and add a transition display for background color.

```
46      lv_style_init(&style_indicator);
47      lv_style_set_bg_opa(&style_indicator, LV_OPA_COVER);
48      lv_style_set_bg_color(&style_indicator, lv_palette_main(LV_PALETTE_BLUE));
49      lv_style_set_radius(&style_indicator, LV_RADIUS_CIRCLE);
50      lv_style_set_transition(&style_indicator, &transition_dsc);
```

Set the background opacity of the style_knob component to be completely opaque, set the background color to cyan, set the edge color to 2 shades darker than cyan. Set the edge width of the component to 2 pixels, set the component as rounded, and set the knob 10 pixels larger than the slider.

```
52      lv_style_init(&style_knob);
53      lv_style_set_bg_opa(&style_knob, LV_OPA_COVER);
54      lv_style_set_bg_color(&style_knob, lv_palette_main(LV_PALETTE_CYAN));
55      lv_style_set_border_color(&style_knob, lv_palette_darken(LV_PALETTE_CYAN, 2));
56      lv_style_set_border_width(&style_knob, 2);
57      lv_style_set_radius(&style_knob, LV_RADIUS_CIRCLE);
58      lv_style_set_pad_all(&style_knob, 10); /*Makes the knob larger*/
59      lv_style_set_transition(&style_knob, &transition_dsc);
```

Set the background color of the style_pressed_color component to 2 shades darker than cyan.

```
61      lv_style_init(&style_pressed_color);
62      lv_style_set_bg_color(&style_pressed_color, lv_palette_darken(LV_PALETTE_CYAN, 2));
```

Create a slider component, remove the default style attribute, add new attributes to the slider, and set the slider width to 200 pixels, height to 20 pixels, and display it in the center of the screen.

```
64      /*Create a slider and add the style*/
65      lv_obj_t * slider = lv_slider_create(lv_scr_act());
66      lv_obj_remove_style_all(slider);        /*Remove the styles coming from the theme*/
67      lv_obj_add_style(slider, &style_main, LV_PART_MAIN);//Set the slider base color to gray
68      lv_obj_add_style(slider, &style_indicator, LV_PART_INDICATOR);//Set the slider color to
    cyan
69      lv_obj_add_style(slider, &style_knob, LV_PART_KNOB);//Set the slider ball handle larger
    and more prominent
70      lv_obj_add_style(slider, &style_pressed_color, LV_PART_INDICATOR | LV_PART_KNOB |
    LV_STATE_PRESSED);// When pressed, the slider color is 2 color numbers darker
71
72      lv_obj_set_size(slider, 200, 20);
73      lv_obj_center(slider);
```

Provide some extra space to draw the slider.

```
81      /*Provide some extra space for the value*/
82      if(code == LV_EVENT_REFR_EXT_DRAW_SIZE) {
83          lv_event_set_ext_draw_size(e, 50);
84      }
```

If the drawing area of the indicator part changes, get the value of the slider and display it.

```
        lv_obj_draw_part_dsc_t * dsc = lv_event_get_draw_part_dsc(e);
        if(dsc->part == LV_PART_INDICATOR) {
            char buf[16];
            lv_snprintf(buf, sizeof(buf), "%d - %d", (int)lv_slider_get_left_value(slider),
(int)lv_slider_get_value(slider));
            LV_LOG_USER("Slider Value Range: %s",buf);
            lv_label_set_text(slider_label, buf);
            lv_obj_align_to(slider_label, slider, LV_ALIGN_OUT_BOTTOM_MID, 0, 10);
        }
```

Create a slider component with a width of 200 pixels, a height of 20 pixels, and display it in the center of the screen.

```
    /*Create a slider in the center of the display*/
    lv_obj_t * slider;
    slider = lv_slider_create(lv_scr_act());

    lv_obj_set_size(slider, 200, 20);
    lv_obj_center(slider);
```

Change the slider mode from normal mode to range mode. Set the starting value of the slider to 20 and the ending value to 70.

```
    lv_slider_set_mode(slider, LV_SLIDER_MODE_RANGE);
    lv_slider_set_value(slider, 70, LV_ANIM_OFF);
    lv_slider_set_left_value(slider, 20, LV_ANIM_OFF);
```

Associate the slider with the callback function slider_event_cb_3. Refresh the drawing size.

```
    lv_obj_add_event_cb(slider, slider_event_cb_3, LV_EVENT_ALL, NULL);
    lv_obj_refresh_ext_draw_size(slider);
```

Create a label component, which displays the content of the slider 10 pixels below the slider.

```
    slider_label = lv_label_create(lv_scr_act());
    lv_label_set_text(slider_label, "20-70");

    lv_obj_align_to(slider_label, slider, LV_ALIGN_OUT_BOTTOM_MID, 0, 10);
```

For more information about LVGL, please refer to the link below:
https://docs.lvgl.io/8.1/widgets/core/slider.html

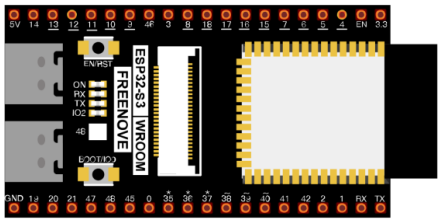
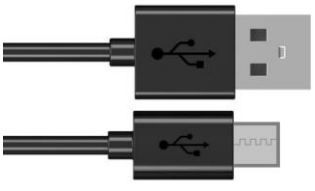
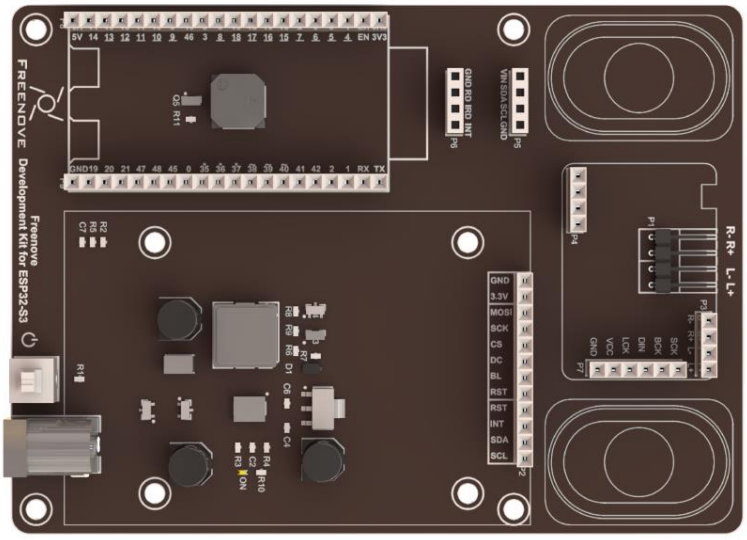
Chapter 13 LVGL Img

In this chapter, we will explore how to use the img component to display images on the screen in LVGL.

Project 13.1 LVGL Img

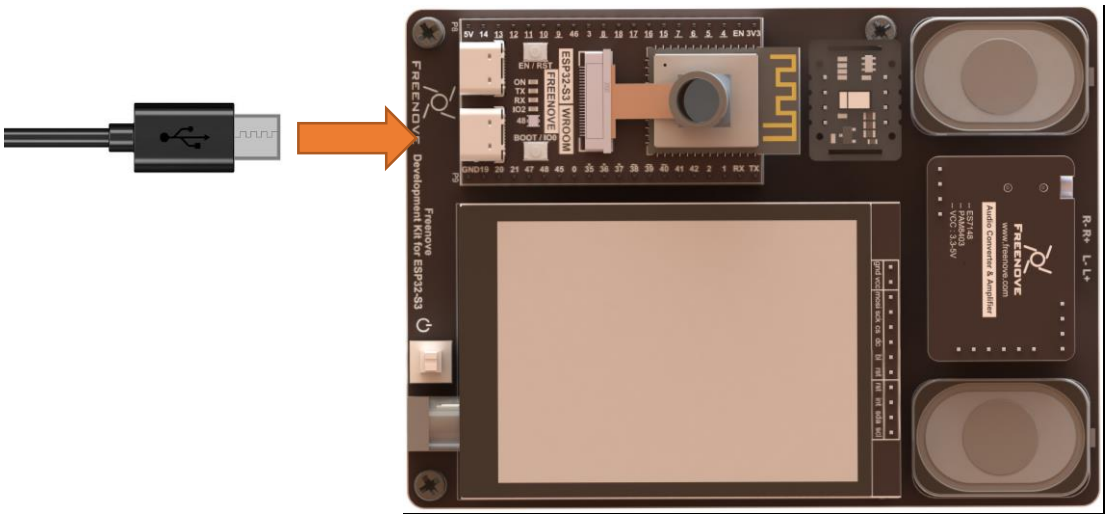
In this section, we will cover three examples that demonstrate how to use the img component in LVGL. Through these examples, you will learn how to effectively display images on the screen using the img component.

Component List

ESP32-S3 WROOM x1 	USB cable x1 	2.8-inch screen 
ESP32-S3 WROOM Shield x1 		

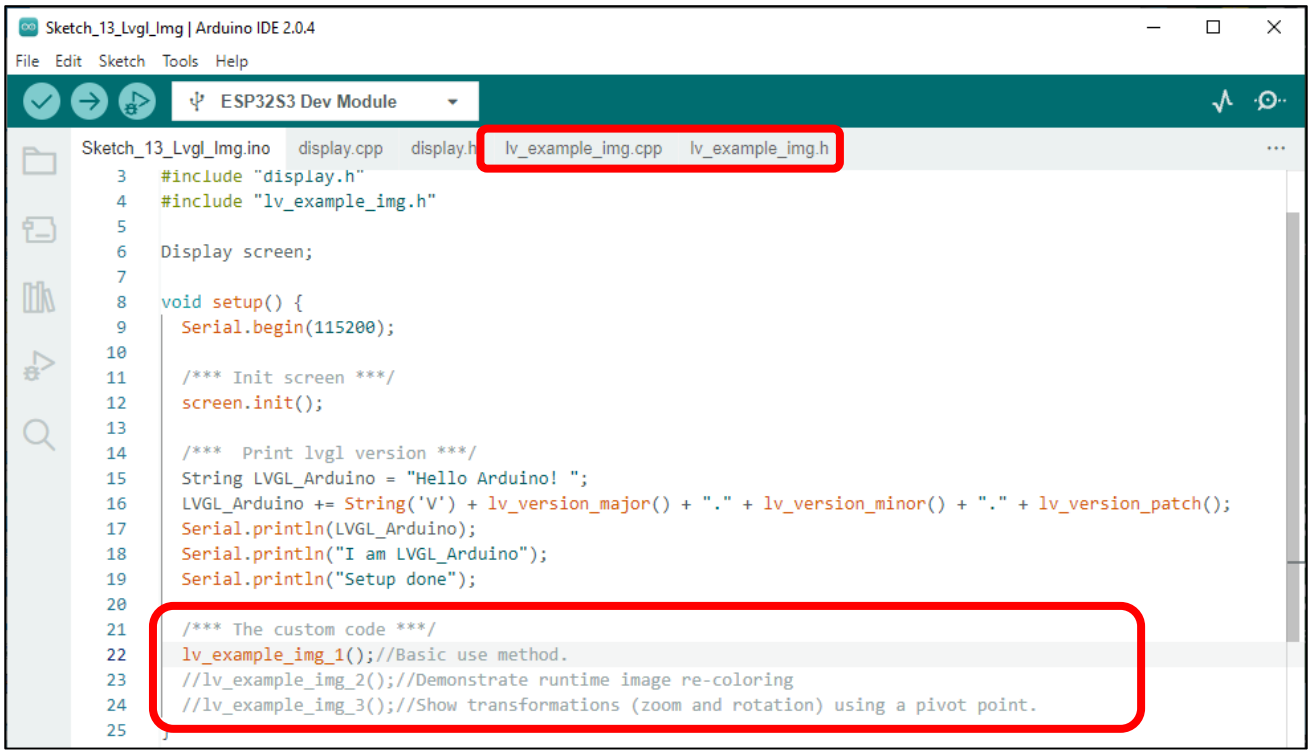
Circuit

Connect Freenove ESP32-S3 to the computer using the USB cable.
Hardware connection. If you need any support, please feel free to contact us via: support@freenove.com



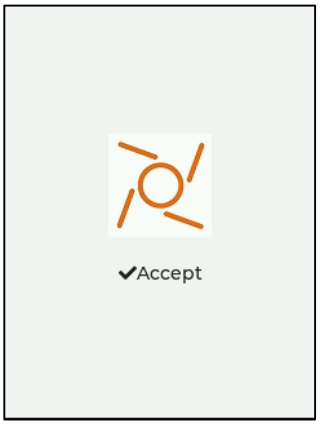
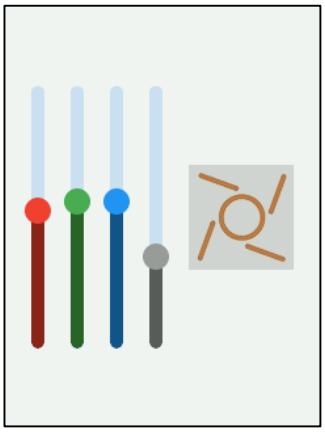
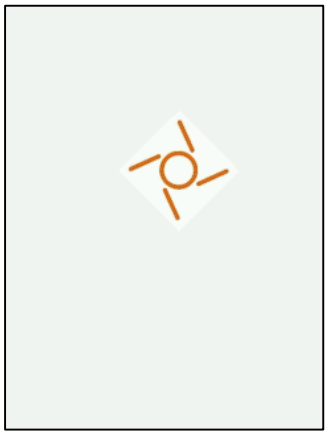
Sketch

Sketch_13_LVGL_Img



By commenting out different sections of the code, compiling and uploading, you can display different content on the screen.

Below are the illustrations of the the examples.

lv_example_img_1	lv_example_img_2	lv_example_img_3
		

The following is the program code:

Sketch_13_LVGL_Img.ino

```
#include <lvgl.h>
#include "Arduino.h"
#include "display.h"
#include "lv_example_label.h"
Display screen;
void setup() {
  Serial.begin(115200);
  /*** Init screen ***/
  screen.init();
  /***  Print lvgl version ***/
  String LVGL_Arduino = "Hello Arduino! ";
  LVGL_Arduino += String('V') + lv_version_major() + "." + lv_version_minor() + "." +
lv_version_patch();
  Serial.println(LVGL_Arduino);
  Serial.println("I am LVGL_Arduino");
  Serial.println("Setup done");

  /*** The custom code ***/
  lv_example_img_1();//Basic use method.
  //lv_example_img_2();//Demonstrate runtime image re-coloring
  //lv_example_img_3();//Show transformations (zoom and rotation) using a pivot point.
}
void loop() {
  screen.routine();
  delay(5);
}
```

Here are three examples. We can comment out two of them to display the remaining one. Please refer to the lv_example_img.cpp and lv_example_img.h files for the specific code.

```
18    lv_example_img_1();//Basic use method.
19    //lv_example_img_2();//Demonstrate runtime image re-coloring
20    //lv_example_img_3();//Show transformations (zoom and rotation) using a pivot point.
```

lv_example_img.h

Declare three functions so that they can be called in the ino file.

```
1  #ifndef __LV_EXAMPLE_IMG_H
2  #define __LV_EXAMPLE_IMG_H
3  void lv_example_img_1(void);
4  void lv_example_img_2(void);
5  void lv_example_img_3(void);
6  #endif
```

Here is the complete code.

lv_example_img.cpp

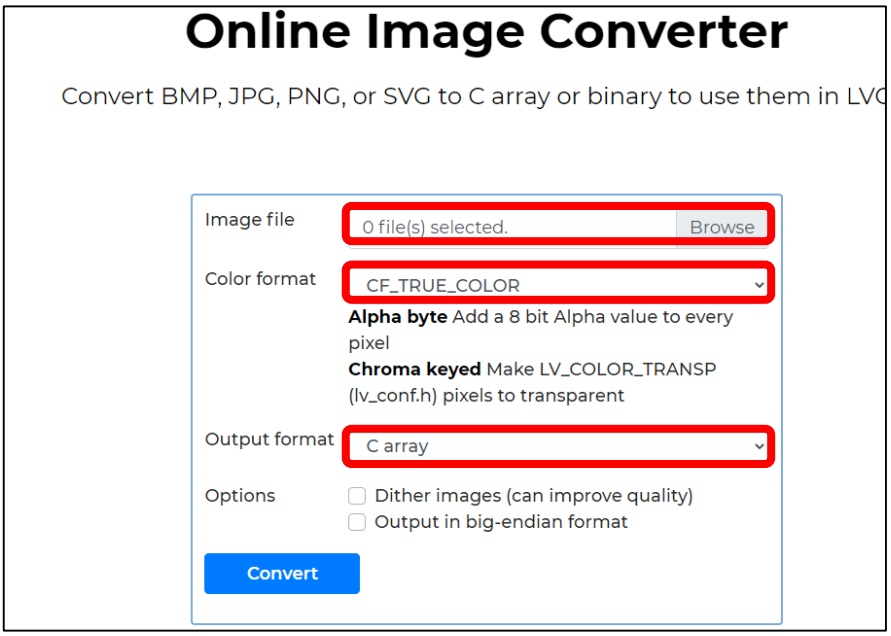
```
1   #include "lvgl.h"
2   #include "lv_example_img.h"
3
4   const uint8_t img_freenove_map[] = {…
85  };
86
87  /*************** Example 1 ***************************/
88  lv_img_dsc_t img_freenove;
89  void lv_img_freenove_init(void){
90    lv_img_header_t header;
91    header.always_zero = 0;
92    header.w = 80;
93    header.h = 80;
94    header.cf = LV_IMG_CF_TRUE_COLOR;
95    img_freenove.header = header;
96    img_freenove.data_size = 6400 * LV_IMG_PX_SIZE_ALPHA_BYTE;
97    img_freenove.data = img_freenove_map;
98  };
99
100 void lv_example_img_1(void){
101     lv_img_freenove_init();
102     lv_obj_t * img1 = lv_img_create(lv_scr_act());//Apply for an img variable
103     lv_img_set_src(img1, &img_freenove);      //Add image information to img
104     lv_obj_align(img1, LV_ALIGN_CENTER, 0, -20);  //It is displayed 20 pixels above the middle
    of the screen
105     lv_obj_set_size(img1, 80, 80);              //Set the size of the image
106
```

```
107     lv_obj_t * img2 = lv_img_create(lv_scr_act());
108     lv_img_set_src(img2, LV_SYMBOL_OK "Accept");
109     lv_obj_align_to(img2, img1, LV_ALIGN_OUT_BOTTOM_MID, 0, 20);//img2 is displayed 20 pixels
        below img1
110 }
111
112 /*************** Example 2 ***************************/
113 static lv_obj_t * red_slider, * green_slider, * blue_slider, * intense_slider;
114 static lv_obj_t * img1;
115
116 static void slider_event_cb(lv_event_t * e){
117     LV_UNUSED(e);
118     /*Recolor the image based on the sliders' values*/
119     lv_color_t color  = lv_color_make(lv_slider_get_value(red_slider),
    lv_slider_get_value(green_slider),lv_slider_get_value(blue_slider));
120     lv_opa_t intense = lv_slider_get_value(intense_slider);
121     lv_obj_set_style_img_recolor_opa(img1, intense, 0);
122     lv_obj_set_style_img_recolor(img1, color, 0);
123 }
124
125 static lv_obj_t * create_slider(lv_color_t color){
126     lv_obj_t * slider = lv_slider_create(lv_scr_act());
127     lv_slider_set_range(slider, 0, 255);
128     lv_obj_set_size(slider, 10, 200);
129     lv_obj_set_style_bg_color(slider, color, LV_PART_KNOB);
130     lv_obj_set_style_bg_color(slider, lv_color_darken(color, LV_OPA_40), LV_PART_INDICATOR);
131     lv_obj_add_event_cb(slider, slider_event_cb, LV_EVENT_VALUE_CHANGED, NULL);
132     return slider;
133 }
134
135 //Demonstrate runtime image re-coloring
136 void lv_example_img_2(void){
137     /*Create 4 sliders to adjust RGB color and re-color intensity*/
138     red_slider = create_slider(lv_palette_main(LV_PALETTE_RED));
139     green_slider = create_slider(lv_palette_main(LV_PALETTE_GREEN));
140     blue_slider = create_slider(lv_palette_main(LV_PALETTE_BLUE));
141     intense_slider = create_slider(lv_palette_main(LV_PALETTE_GREY));
142
143     lv_slider_set_value(red_slider, LV_OPA_20, LV_ANIM_OFF);
144     lv_slider_set_value(green_slider, LV_OPA_90, LV_ANIM_OFF);
145     lv_slider_set_value(blue_slider, LV_OPA_60, LV_ANIM_OFF);
146     lv_slider_set_value(intense_slider, LV_OPA_50, LV_ANIM_OFF);
147
148     lv_obj_align(red_slider, LV_ALIGN_LEFT_MID, 20, 0);
```

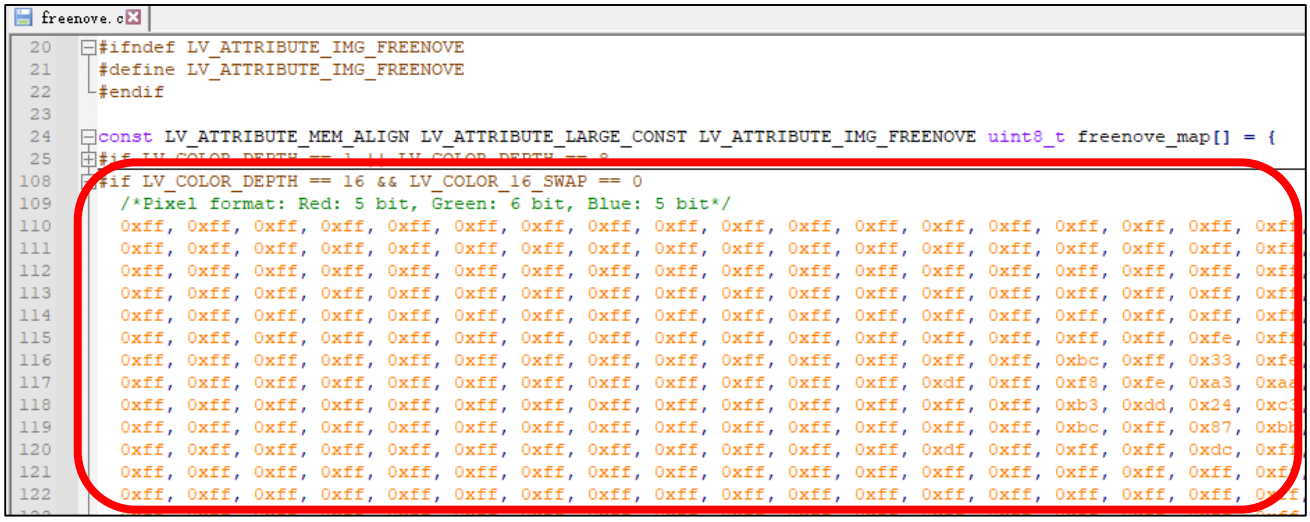
```
149     lv_obj_align_to(green_slider, red_slider, LV_ALIGN_OUT_RIGHT_MID, 20, 0);
150     lv_obj_align_to(blue_slider, green_slider, LV_ALIGN_OUT_RIGHT_MID, 20, 0);
151     lv_obj_align_to(intense_slider, blue_slider, LV_ALIGN_OUT_RIGHT_MID, 20, 0);
152
153     /*Now create the actual image*/
154     lv_img_freenove_init();
155     img1 = lv_img_create(lv_scr_act());
156     lv_img_set_src(img1, &img_freenove);
157     lv_obj_align_to(img1, intense_slider, LV_ALIGN_OUT_RIGHT_MID, 20, 0);
158     lv_event_send(intense_slider, LV_EVENT_VALUE_CHANGED, NULL);
159 }
160
161 /*************** Example 3 ***************************/
162 static void set_angle(void * img, int32_t v){
163     lv_img_set_angle((lv_obj_t*)img, v);
164 }
165 static void set_zoom(void * img, int32_t v){
166     lv_img_set_zoom((lv_obj_t*)img, v);
167 }
168
169 //Show transformations (zoom and rotation) using a pivot point.
170 void lv_example_img_3(void){
171     lv_img_freenove_init();
172     /*Now create the actual image*/
173     lv_obj_t * img = lv_img_create(lv_scr_act());
174     lv_img_set_src(img, &img_freenove);
175     lv_obj_align(img, LV_ALIGN_CENTER, 0, 0);
176     lv_img_set_pivot(img, 0, 0);    /*Rotate around the top left corner*/
177
178     lv_anim_t a;
179     lv_anim_init(&a);
180     lv_anim_set_var(&a, img);
181     lv_anim_set_time(&a, 3000);          //Animation execution time
182     lv_anim_set_playback_time(&a, 3000);//Animation playback time
183     lv_anim_set_repeat_count(&a, LV_ANIM_REPEAT_INFINITE);//Repetitive animation
184
185     lv_anim_set_exec_cb(&a, set_angle); //Sets the animation callback function
186     lv_anim_set_values(&a, 0, 3600);    //Sets the parameter range of the callback function
187     lv_anim_start(&a);                  //Start animation
188
189     lv_anim_set_exec_cb(&a, set_zoom);  //Sets the animation callback function
190     lv_anim_set_values(&a, 128, 512);   //Sets the parameter range of the callback function
191     lv_anim_start(&a);                  //Start animation
192 }
```

Here we provide a website: https://lvgl.io/tools/imageconverter

It is an official website that allows you to convert images into arrays. You can click on the "Browse" button to select an image and convert it into an array file using the following configuration. Here we take the Freenove logo as an example.



Click "Convert" and the website will generate and download the corresponding array file. Open this file and find the definition shown below.



Define an array img_freenove_map and copy the content of the previously generated array to it.

```
4    const uint8_t img_freenove_map[] = {
...  ...
85   };
```

Define an image type variable img_freenove and write a function lv_img_freenove_init() to initialize it. Please note that the resolution of the original image must be filled in correctly. “w” and “h” represent the width and height of the original image resolution, respectively. “data_size” represents the size of the image, while “data” represents the actual image data. It is recommended not to modify other options.

```
88  lv_img_dsc_t img_freenove;
89  void lv_img_freenove_init(void){
90    lv_img_header_t header;
91    header.always_zero = 0;
92    header.w = 80;
93    header.h = 80;
94    header.cf = LV_IMG_CF_TRUE_COLOR;
95    img_freenove.header = header;
96    img_freenove.data_size = 6400 * LV_IMG_PX_SIZE_ALPHA_BYTE;
97    img_freenove.data = img_freenove_map;
98  };
```

Call the lv_img_freenove_init() function to configure img_freenove.

```
101     lv_img_freenove_init();
```

Create a new image component and assign it to img1, set the image content to img_freenove, and display the image 20 pixels above the center of the screen. Set the size of the component to 80 pixels wide and 80 pixels high.

```
102     lv_obj_t * img1 = lv_img_create(lv_scr_act());//Apply for an img variable
103     lv_img_set_src(img1, &img_freenove);      //Add image information to img
104     lv_obj_align(img1, LV_ALIGN_CENTER, 0, -20);  //It is displayed 20 pixels above the middle
105 of the screen
106     lv_obj_set_size(img1, 80, 80);                //Set the size of the image
```

Create a new image component and assign it to img2. Display an internal symbol and text on img2. Set img2 to be positioned 20 pixels below img1.

```
107     lv_obj_t * img2 = lv_img_create(lv_scr_act());
108     lv_img_set_src(img2, LV_SYMBOL_OK "Accept");
109     lv_obj_align_to(img2, img1, LV_ALIGN_OUT_BOTTOM_MID, 0, 20);//img2 is displayed 20 pixels
    below img1
```

Write a slider creation function, set the range, size, background color, and associated function of the slider.

```
126 static lv_obj_t * create_slider(lv_color_t color){
127     lv_obj_t * slider = lv_slider_create(lv_scr_act());
128     lv_slider_set_range(slider, 0, 255);
129     lv_obj_set_size(slider, 10, 200);
130     lv_obj_set_style_bg_color(slider, color, LV_PART_KNOB);
131     lv_obj_set_style_bg_color(slider, lv_color_darken(color, LV_OPA_40), LV_PART_INDICATOR);
132     lv_obj_add_event_cb(slider, slider_event_cb, LV_EVENT_VALUE_CHANGED, NULL);
133     return slider;
134 }
```

Create 4 sliders, and set them to different colors.

```
139     red_slider = create_slider(lv_palette_main(LV_PALETTE_RED));
140     green_slider = create_slider(lv_palette_main(LV_PALETTE_GREEN));
```

```
    blue_slider = create_slider(lv_palette_main(LV_PALETTE_BLUE));
    intense_slider = create_slider(lv_palette_main(LV_PALETTE_GREY));
```

Use the values of three sliders to redefine the color of the image, and use the value of one slider to modify the opacity of the image background.

```
static void slider_event_cb(lv_event_t * e){
    LV_UNUSED(e);
    /*Recolor the image based on the sliders' values*/
    lv_color_t color  = lv_color_make(lv_slider_get_value(red_slider),
lv_slider_get_value(green_slider),lv_slider_get_value(blue_slider));
    lv_opa_t intense = lv_slider_get_value(intense_slider);
    lv_obj_set_style_img_recolor_opa(img1, intense, 0);
    lv_obj_set_style_img_recolor(img1, color, 0);
}
```

Trigger a slider event manually.

```
    lv_event_send(intense_slider, LV_EVENT_VALUE_CHANGED, NULL);
```

Image rotation function.

```
    lv_img_set_angle((lv_obj_t*)img, v);
```

Image scaling function.

```
    lv_img_set_zoom((lv_obj_t*)img, v);
```

Create an image component, set the display content and display position. Set the origin position of the image.

```
    lv_img_freenove_init();
    /*Now create the actual image*/
    lv_obj_t * img = lv_img_create(lv_scr_act());
    lv_img_set_src(img, &img_freenove);
    lv_obj_align(img, LV_ALIGN_CENTER, 0, 0);
    lv_img_set_pivot(img, 0, 0);    /*Rotate around the top left corner*/
```

Create an animation object, associate the animation object with the img component, set the animation time to 3000ms, and the animation playback time to 3000ms, and play the animation repeatedly.

```
    lv_anim_t a;
    lv_anim_init(&a);
    lv_anim_set_var(&a, img);
    lv_anim_set_time(&a, 3000);          //Animation execution time
    lv_anim_set_playback_time(&a, 3000);//Animation playback time
    lv_anim_set_repeat_count(&a, LV_ANIM_REPEAT_INFINITE);//Repetitive animation
```

Set the animation object associated with the set_angle function, with a precision of 0.1 degrees and 360 degrees corresponding to 3600. Start the animation.

```
    lv_anim_set_exec_cb(&a, set_angle); //Sets the animation callback function
    lv_anim_set_values(&a, 0, 3600);    //Sets the parameter range of the callback function
    lv_anim_start(&a);                  //Start animation
```

Associate the animation object with the set_zoom function, with 256 as the original image size, 128 representing the image being reduced to half its size, and 512 representing the image being enlarged to twice its size. Start the animation.

```
    lv_anim_set_exec_cb(&a, set_zoom);  //Sets the animation callback function
```

```
191     lv_anim_set_values(&a, 128, 512);    //Sets the parameter range of the callback function
192     lv_anim_start(&a);                   //Start animation
```

For more information about LVGL, please refer to the link below:
https://docs.lvgl.io/8.1/widgets/core/img.html

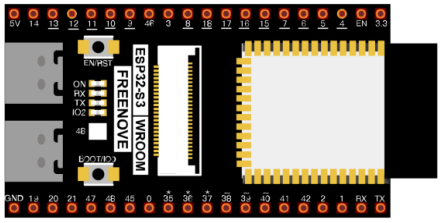
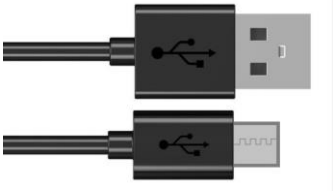
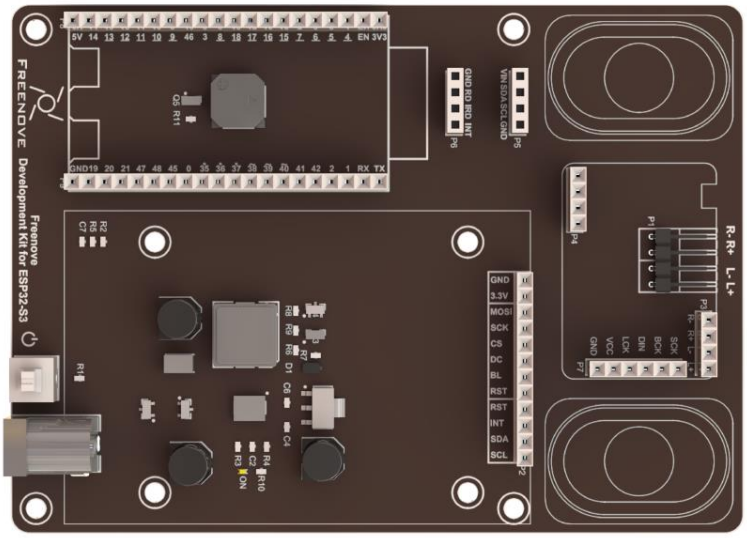
Chapter 14 LVGL Imgbtn

In this chapter, we will learn the usage of imgbtn component on the screen.

Project 14.1 LVGL Imgbtn

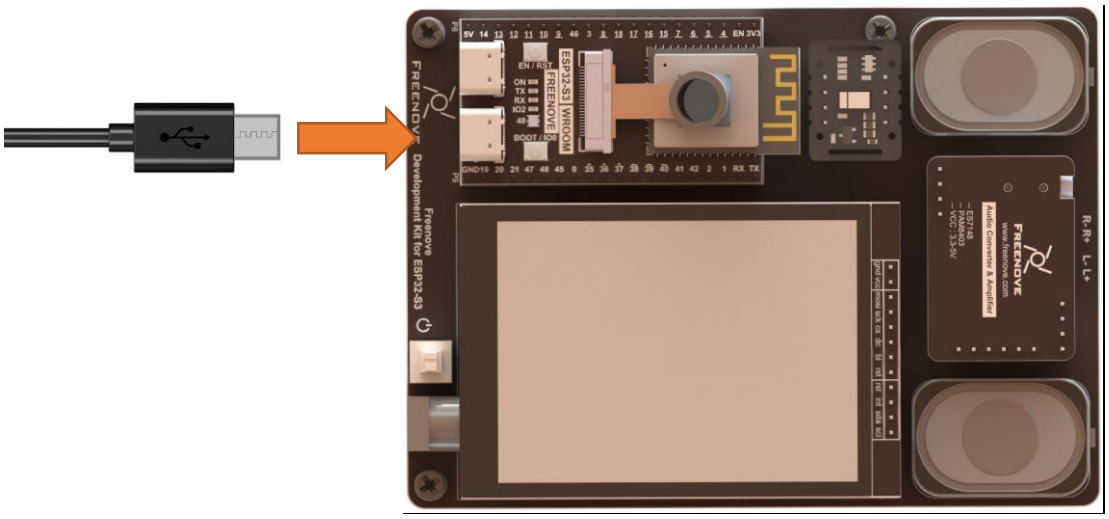
We have provided an example in the code. With the example, we will learn how to use the imgbtn component.

Component List

ESP32-S3 WROOM x1 	USB cable x1 	2.8-inch screen 
ESP32-S3 WROOM Shield x1 		

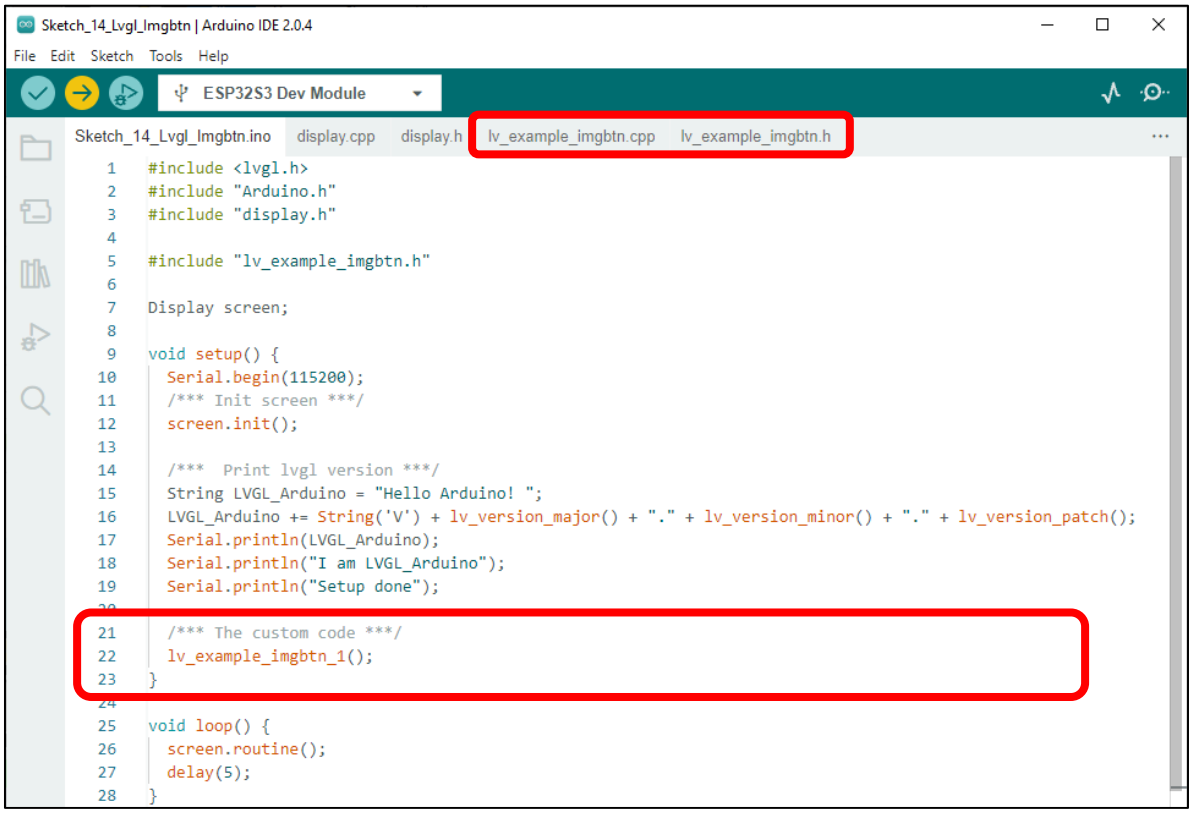
Circuit

Connect Freenove ESP32-S3 to the computer using the USB cable.
Hardware connection. If you need any support, please feel free to contact us via: support@freenove.com



Sketch

Sketch_14_LVGL_Imgbtn



Here is an illustration of the example:

lv_example_imgbtn_1

The following is the program code:

Sketch_14_LVGL_Imgbtn.ino

```
#include <lvgl.h>
#include "Arduino.h"
#include "display.h"
#include "lv_example_label.h"
Display screen;
void setup() {
  Serial.begin(115200);
  /*** Init screen ***/
  screen.init();
  /***  Print lvgl version ***/
  String LVGL_Arduino = "Hello Arduino! ";
  LVGL_Arduino += String('V') + lv_version_major() + "." + lv_version_minor() + "." +
lv_version_patch();
  Serial.println(LVGL_Arduino);
  Serial.println("I am LVGL_Arduino");
  Serial.println("Setup done");

  /*** The custom code ***/
  lv_example_imgbtn_1();//Basic use method.
}
void loop() {
  screen.routine();
  delay(5);
}
```

lv_example_imgbtn.h

We can define the three example functions and call them in the ino file.

```
#ifndef __LV_EXAMPLE_IMGBTN_H
#define __LV_EXAMPLE_IMGBTN_H
void lv_example_imgbtn_1(void);
#endif
```

Here is the complete code:

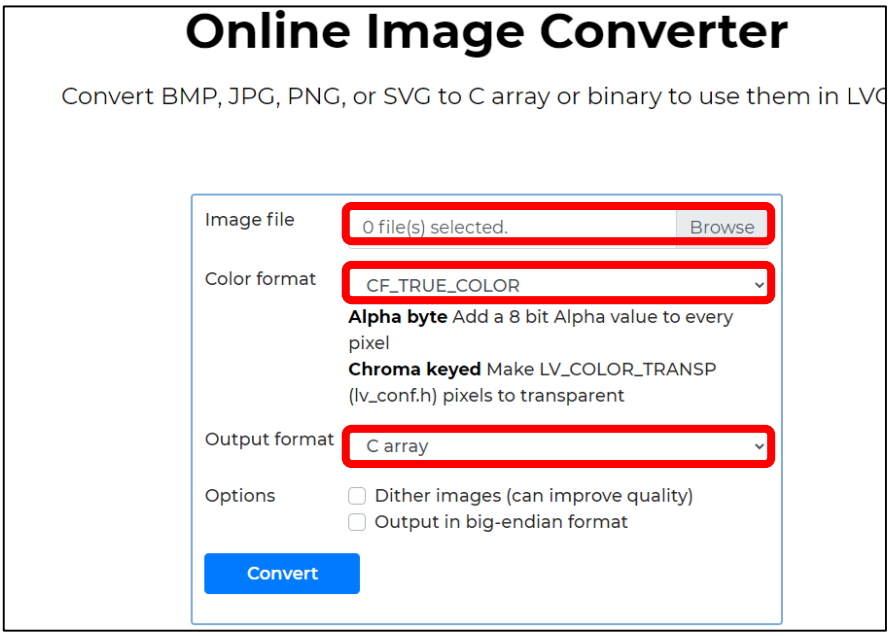
lv_example_imgbtn.cpp

```
#include "lvgl.h"
#include "lv_example_imgbtn.h"

const uint8_t img_freenove_map[] = {…
};

/*************** Example 1 ***************************/
lv_img_dsc_t img_freenove;
void lv_img_freenove_init(void){
  lv_img_header_t header;
  header.always_zero = 0;
  header.w = 80;
  header.h = 80;
  header.cf = LV_IMG_CF_TRUE_COLOR;
  img_freenove.header = header;
  img_freenove.data_size = 6400 * LV_IMG_PX_SIZE_ALPHA_BYTE;
  img_freenove.data = img_freenove_map;
};

void lv_example_imgbtn_1(void) {
  lv_img_freenove_init();
  /*Init the pressed style*/
  static lv_style_t style_pr;//Apply for a style
  lv_style_init(&style_pr);  //Initialize it
  lv_style_set_translate_y(&style_pr, 5);//Style: Every time you trigger, move down 5 pixels

  /*Create an image button*/
  lv_obj_t *imgbtn1 = lv_imgbtn_create(lv_scr_act());    //Apply for a picture button
  lv_obj_add_style(imgbtn1, &style_pr, LV_STATE_PRESSED);//Triggered when the button is
pressed

  lv_obj_set_size(imgbtn1, 80, 80);                //Sets the size of the picture button
  lv_obj_align(imgbtn1, LV_ALIGN_CENTER, 0, 0);   //Set the position of the picture button
  lv_imgbtn_set_src(imgbtn1, LV_IMGBTN_STATE_RELEASED, NULL, &img_freenove, NULL);

```

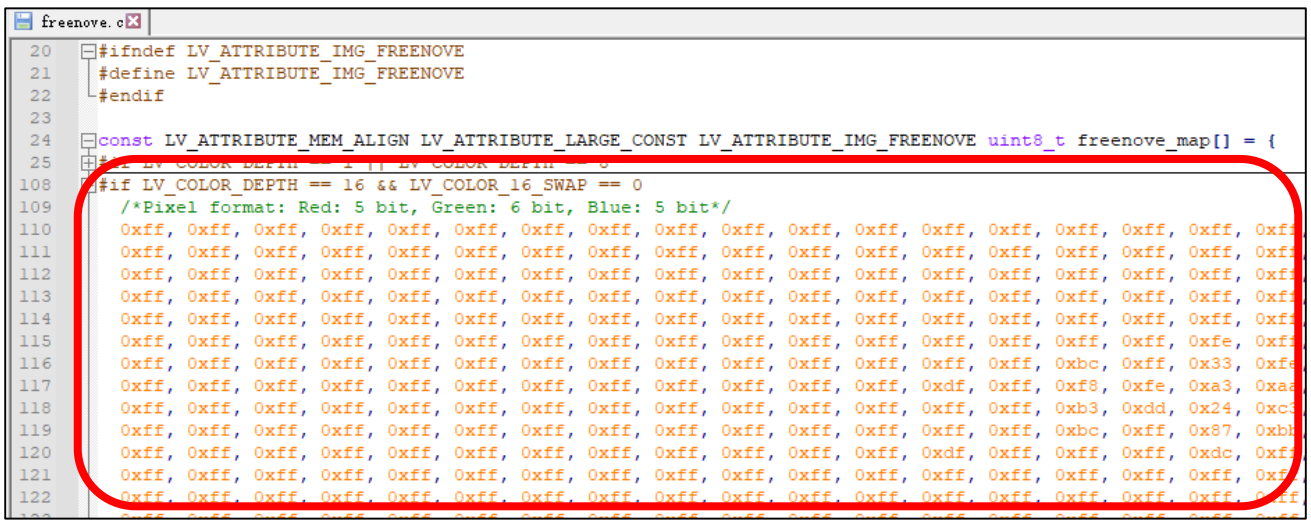
```
115     /*Create a label on the image button*/
116     lv_obj_t *label = lv_label_create(lv_scr_act());
117     lv_label_set_text(label, "Freenove");
118     lv_obj_align_to(label, imgbtn1, LV_ALIGN_OUT_BOTTOM_MID, 0, 20);
119 }
```

Here we provide a website: https://lvgl.io/tools/imageconverter

You can use the official website to convert an image to an array. Click "Browse" to select an image and convert it to an array file with the following configuration. Here we take the Freenove logo as an example.



Click "Convert" and the website will generate and download the corresponding array file. Open this file and find the definition shown below.



Define an array img_freenove_map and copy the content of the previously generated array to it.。

```
4   const uint8_t img_freenove_map[] = {
…   …
85  };
```

Define a variable img_freenove of type lv_img_dsc_t and write a function lv_img_freenove_init() to initialize it.

Please note that the original image resolution must be correctly filled in. “w” and “h” represent the width and height of the original image resolution, respectively. “data_size” represents the size of the image, while “data” represents the actual image data. It is recommended not to modify other options.

```
88  lv_img_dsc_t img_freenove;
89  void lv_img_freenove_init(void){
90    lv_img_header_t header;
91    header.always_zero = 0;
92    header.w = 80;
93    header.h = 80;
94    header.cf = LV_IMG_CF_TRUE_COLOR;
95    img_freenove.header = header;
96    img_freenove.data_size = 6400 * LV_IMG_PX_SIZE_ALPHA_BYTE;
97    img_freenove.data = img_freenove_map;
98  };
```

Call the lv_img_freenove_init() function to configure img_freenove.

```
101     lv_img_freenove_init();
```

Create a style component, when this component is triggered, move the component down 5 pixels.

```
102   /*Init the pressed style*/
103   static lv_style_t style_pr;//Apply for a style
104   lv_style_init(&style_pr);  //Initialize it
105   lv_style_set_translate_y(&style_pr, 5);//Style: Every time you trigger, move down 5 pixels
```

Create an image button component. Trigger the effect of the style when the component is pressed.

```
107   /*Create an image button*/
108   lv_obj_t *imgbtn1 = lv_imgbtn_create(lv_scr_act());    //Apply for a picture button
109   lv_obj_add_style(imgbtn1, &style_pr, LV_STATE_PRESSED);//Triggered when the button is
      pressed
```

Set the size and position of the image button component, and display a picture on it.

```
111   lv_obj_set_size(imgbtn1, 80, 80);                //Sets the size of the picture button
112   lv_obj_align(imgbtn1, LV_ALIGN_CENTER, 0, 0);    //Set the position of the picture button
113   lv_imgbtn_set_src(imgbtn1, LV_IMGBTN_STATE_RELEASED, NULL, &img_freenove, NULL);
```

Create a label and display the content of the label 20 pixels below the image button component.

```
115   /*Create a label on the image button*/
116   lv_obj_t *label = lv_label_create(lv_scr_act());
117   lv_label_set_text(label, "Freenove");
118   lv_obj_align_to(label, imgbtn1, LV_ALIGN_OUT_BOTTOM_MID, 0, 20);
```

For more information about LVGL, please refer to the link below:

https://docs.lvgl.io/8.1/widgets/extra/imgbtn.html

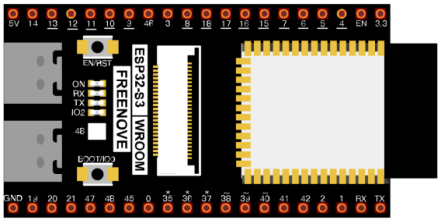
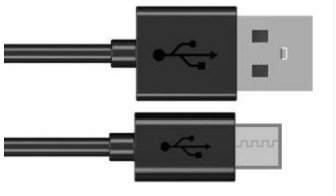
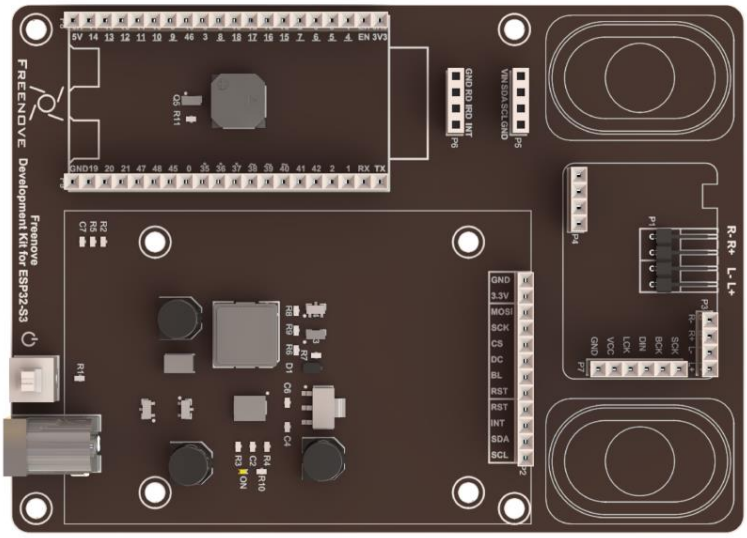
Chapter 15 LVGL Camera

In this chapter, we will learn how to create a camera example.

Project 15.1 LVGL Camera

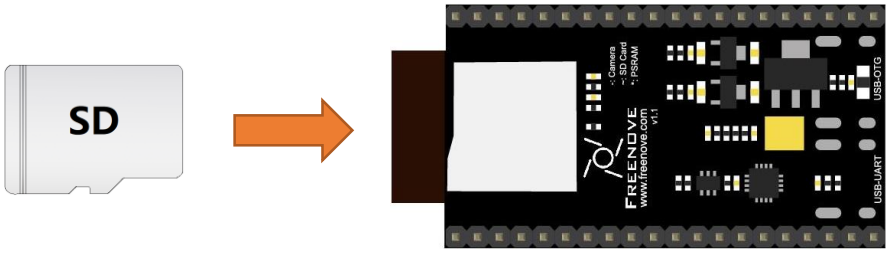
In this project, we will learn how to display the images captured by the camera on the screen.

Component List

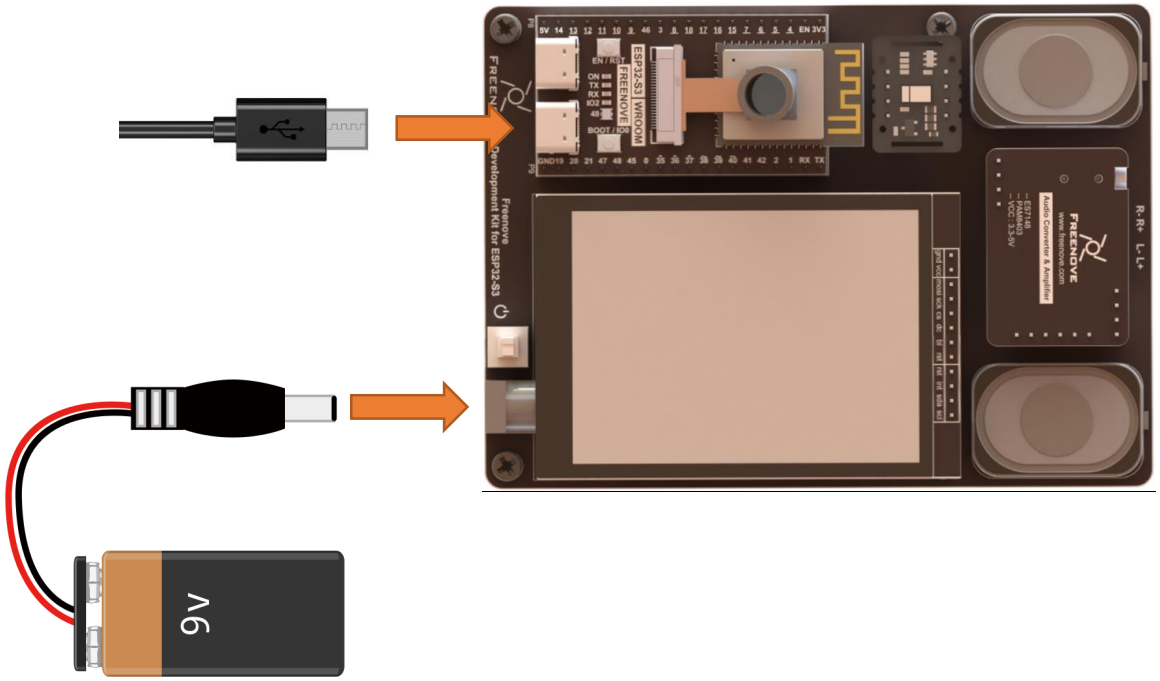
ESP32-S3 WROOM x1 	USB cable x1 	2.8-inch screen 
ESP32-S3 WROOM Shield x1 		Card reader x1 (random color) 
9V battery cable x1 	9V battery x1 (Not included, prepared by yourself) 	SD card x1 

Circuit

If you have not yet used the SD card, please refer to Chapter 4. Click here to navigate back to Chapter 4.
Before connecting the USB cable, insert the SD card into the SD card slot on the back of the ESP32-S3.



Connect Freenove ESP32-S3 to the computer using the USB cable.

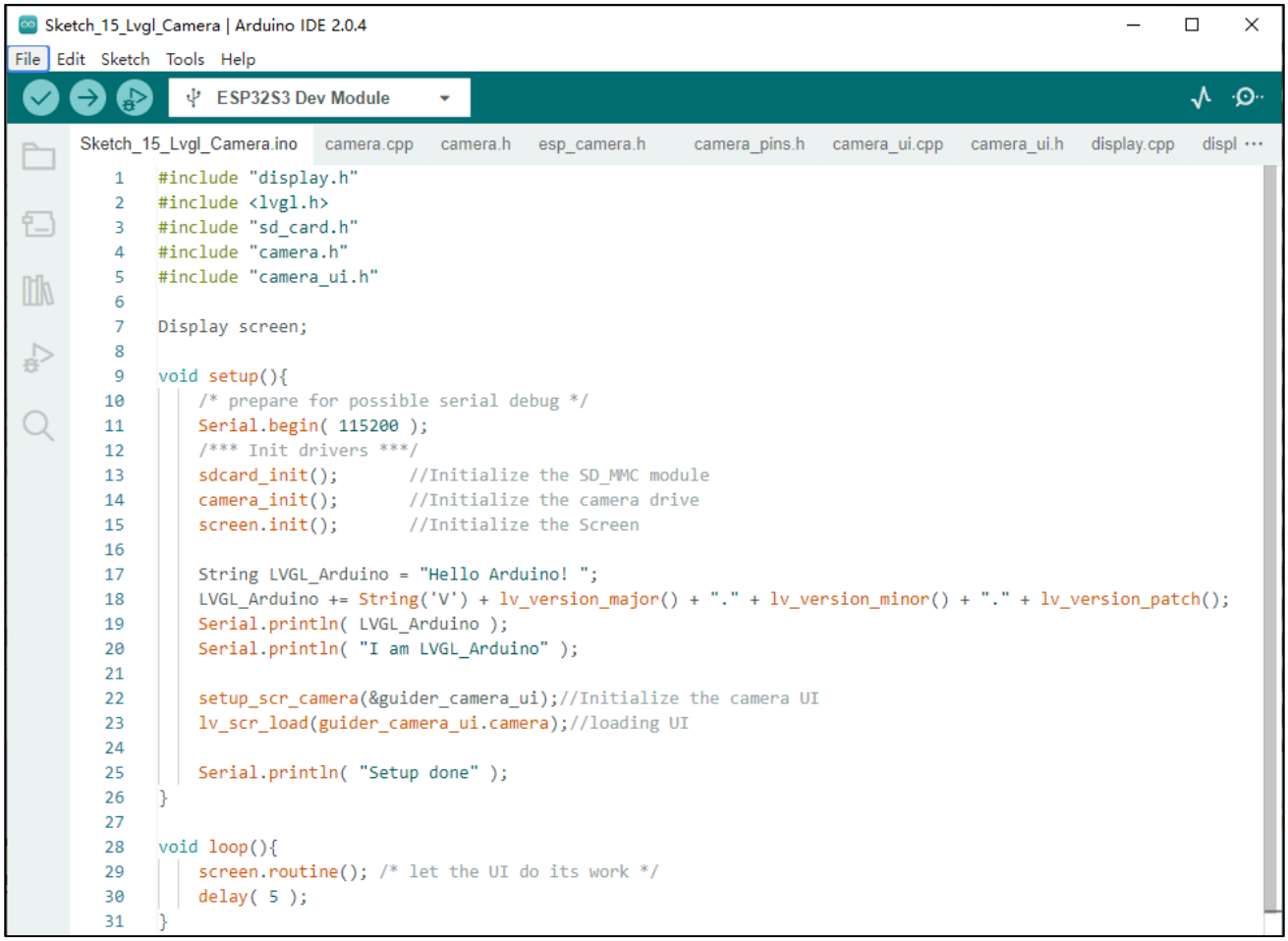


Sketch

Note: If you press the button to take a picture in the project, the picture will be saved to the picture folder of the sd card. You can use a card reader to access these pictures.

If you want to flip the image of the display, just swipe the screen slightly from left to right.

Sketch_15_LVGL_Camera



The following is the program code:

```
#include "display.h"
#include <lvgl.h>
#include "sd_card.h"
#include "camera.h"
#include "camera_ui.h"

Display screen;

void setup(){
    /* prepare for possible serial debug */
    Serial.begin( 115200 );
    /*** Init drivers ***/
    sdcard_init();       //Initialize the SD_MMC module
```

```
14      camera_init();        //Initialize the camera drive
15      screen.init();        //Initialize the Screen
16
17      String LVGL_Arduino = "Hello Arduino! ";
18      LVGL_Arduino += String('V') + lv_version_major() + "." + lv_version_minor() + "." +
    lv_version_patch();
19      Serial.println( LVGL_Arduino );
20      Serial.println( "I am LVGL_Arduino" );
21
22      setup_scr_camera(&guider_camera_ui);
23      lv_scr_load(guider_camera_ui.camera);
24
25      Serial.println( "Setup done" );
26  }
27
28  void loop(){
29      screen.routine(); /* let the GUI do its work */
30      delay( 5 );
31  }
```

Initialize the SD card, camera, and screen. Note that the LVGL library file system has already been configured to include the SD card by default. Therefore, in order to use the file system to manipulate the SD card in LVGL, SD card initialization must be performed before screen initialization.

```
13      sdcard_init();        //Initialize the SD_MMC module
14      camera_init();        //Initialize the camera drive
15      screen.init();        //Initialize the Screen
```

Configure the camera interface and load this interface.

```
22      setup_scr_camera(&guider_camera_ui);
23      lv_scr_load(guider_camera_ui.camera);
```

camera_ui.h

Declare the functions so that they can be called in the INO file.

```
1   #ifndef __CAMERA_UI_H
2   #define __CAMERA_UI_H
3
4   #include "lvgl.h"
5   #include "Arduino.h"
6   #include "esp_camera.h"
7   #include "lv_img.h"
8
9   extern camera_fb_t *fb;
10
11  typedef struct lvgl_camera{
12    lv_obj_t *camera;
13    lv_obj_t *camera_video;
14    lv_obj_t *camera_imgbtn_photo;
```

```
  lv_obj_t *camera_imgbtn_home;
}lvgl_camera_ui;

extern lv_img_dsc_t photo_show;              //Apply an lvgl image variable
extern lvgl_camera_ui guider_camera_ui;      //Camera ui structure

void ui_set_photo_show(void);                //Initialize an lvgl image variable
void setup_scr_camera(lvgl_camera_ui *ui);   //Parameter configuration function on the camera screen

void create_camera_task(void);               //Create camera task thread
void stop_camera_task(void);                 //Close the camera thread
void loopTask_camera(void *pvParameters);    //Camera task thread

#endif
```

camera_ui.cpp

```
#include "camera_ui.h"
#include "camera.h"
#include "sd_card.h"

lv_img_dsc_t photo_show;          //apply an lvgl image variable
lvgl_camera_ui guider_camera_ui; //camera ui structure
camera_fb_t *fb = NULL;           //data structure of camera frame buffer
camera_fb_t *fb_buf = NULL;
TaskHandle_t cameraTaskHandle;    //camera thread task handle
static int camera_task_flag=0;    //camera thread running flag

//Create camera task thread
void create_camera_task(void){
  if(camera_task_flag==0){
    camera_task_flag=1;
    ui_set_photo_show();
    xTaskCreate(loopTask_camera, "loopTask_camera", 8192, NULL, 1, &cameraTaskHandle);
  }
  else{
    Serial.println("loopTask_camera is running...");
  }
}

//Close the camera thread
void stop_camera_task(void){
  if(camera_task_flag==1){
    camera_task_flag=0;
    while(1){
```

```
29        if (eTaskGetState(cameraTaskHandle) == eDeleted){
30            break;
31        }
32        vTaskDelay(10);
33      }
34      Serial.println("loopTask_camera deleted!");
35    }
36  }
37
38  //camera thread
39  void loopTask_camera(void *pvParameters) {
40    Serial.println("loopTask_camera start...");
41    while (camera_task_flag) {
42      fb = esp_camera_fb_get();
43      fb_buf = fb;
44      esp_camera_fb_return(fb);
45      if(fb_buf!=NULL){
46        for(int i=0;i<fb_buf->len;i+=2){
47          uint8_t temp=0;
48          temp=fb_buf->buf[i];
49          fb_buf->buf[i]=fb_buf->buf[i+1];
50          fb_buf->buf[i+1]=temp;
51        }
52        photo_show.data = fb_buf->buf;
53        lv_img_set_src(guider_camera_ui.camera_video,&photo_show);
54      }
55    }
56    vTaskDelete(cameraTaskHandle);
57  }
58
59  //Initialize an lvgl image variable
60  void ui_set_photo_show(void){
61      lv_img_header_t header;
62      header.always_zero = 0;
63      header.w = 240;
64      header.h = 240;
65      header.cf = LV_IMG_CF_TRUE_COLOR;
66      photo_show.header = header;
67      photo_show.data_size = 240 * 240 * 2;
68      photo_show.data = NULL;
69  }
70
71  //Click the photo icon, callback function: goes to the main ui interface
72  static void camera_imgbtn_photo_event_handler(lv_event_t *e) {
```

```
73    lv_event_code_t code = lv_event_get_code(e);
74    if (code == LV_EVENT_CLICKED) {
75      Serial.println("Clicked the camera button.");
76      if(camera_task_flag==1){
77        stop_camera_task();
78        fb = esp_camera_fb_get();
79        if (fb != NULL) {
80          for (int i = 0; i < fb->len; i += 2) {
81            uint8_t temp = 0;
82            temp = fb->buf[i];
83            fb->buf[i] = fb->buf[i + 1];
84            fb->buf[i + 1] = temp;
85          }
86          int photo_index = list_count_number(list_picture);
87          Serial.printf("photo_index: %d\r\n",photo_index);
88          if(photo_index!=-1){
89            String path = String(PICTURE_FOLDER) + "/" + String(++photo_index) +".bmp";//You can
    view it directly from your computer
90            write_rgb565_to_bmp((char *)path.c_str(), fb->buf, fb->len, fb->height, fb->width);
91            list_insert_tail(list_picture,(char *)path.c_str());
92          }
93        }
94        else {
95          Serial.println("Camera capture failed.");
96        }
97        esp_camera_fb_return(fb);
98      }
99      create_camera_task();
100   }
101 }
102
103 //Click the home icon, callback function: goes to the main ui interface
104 static void camera_imgbtn_home_event_handler(lv_event_t *e) {
105   lv_event_code_t code = lv_event_get_code(e);
106
107   switch (code) {
108     case LV_EVENT_CLICKED:
109       {
110         Serial.println("Clicked the home button.");
111       }
112       break;
113     case LV_EVENT_RELEASED:
114       {
115         /*
```

```
116         stop_camera_task();
117         if (!lv_obj_is_valid(guider_main_ui.main))
118           setup_scr_main(&guider_main_ui);
119         lv_disp_t *d = lv_obj_get_disp(lv_scr_act());
120         if (d->prev_scr == NULL && d->scr_to_load == NULL)
121           lv_scr_load(guider_main_ui.main);
122         lv_obj_del(guider_camera_ui.camera);
123         */
124       }
125       break;
126     default:
127       break;
128   }
129 }
130
131 //Slide the screen to flip the screen
132 static void camera_screen_gesture_event_handler(lv_event_t *e) {
133   lv_event_code_t code = lv_event_get_code(e);
134   if (code == LV_EVENT_GESTURE) {
135     lv_dir_t dir = lv_indev_get_gesture_dir(lv_indev_get_act());
136     bool state=0;
137     switch (dir) {
138       case LV_DIR_LEFT:
139       case LV_DIR_RIGHT:
140         state = camera_get_mirror_horizontal();
141         state = !state;
142         camera_set_mirror_horizontal(state);
143         break;
144       case LV_DIR_TOP:
145       case LV_DIR_BOTTOM:
146         state = camera_get_flip_vertical();
147         state = !state;
148         camera_set_flip_vertical(state);
149         break;
150     }
151   }
152 }
153
154 //Parameter configuration function on the camera screen
155 void setup_scr_camera(lvgl_camera_ui *ui) {
156   //Write the camera interface
157   ui->camera = lv_obj_create(NULL);
158   setup_list_head_picture();    //Generate a linked list based on the SD card's picture folder
159   lv_img_home_init();
```

```
160   lv_img_camera_init();
161
162   /*Init the pressed style*/
163   static lv_style_t style_pr;//Apply for a style
164   lv_style_init(&style_pr);  //Initialize it
165   lv_style_set_translate_y(&style_pr, 5);//Style: Every time you trigger, move down 5 pixels
166
167   //Write codes for camera_video
168   ui->camera_video = lv_img_create(ui->camera);
169   lv_obj_set_pos(ui->camera_video, 0, 0);
170   lv_obj_set_size(ui->camera_video, 240, 240);
171
172   //Write codes for camera_photo
173   ui->camera_imgbtn_photo = lv_imgbtn_create(ui->camera);
174   lv_obj_set_pos(ui->camera_imgbtn_photo, 20, 240);
175   lv_obj_set_size(ui->camera_imgbtn_photo, 80, 80);
176   lv_img_set_src(ui->camera_imgbtn_photo, &img_camera);
177   lv_obj_add_style(ui->camera_imgbtn_photo, &style_pr, LV_STATE_PRESSED);
178   //Write codes for camera_return
179   ui->camera_imgbtn_home = lv_imgbtn_create(ui->camera);
180   lv_obj_set_pos(ui->camera_imgbtn_home, 140, 240);
181   lv_obj_set_size(ui->camera_imgbtn_home, 80, 80);
182   lv_img_set_src(ui->camera_imgbtn_home, &img_home);
183   lv_obj_add_style(ui->camera_imgbtn_home, &style_pr, LV_STATE_PRESSED);//Triggered when the
184 button is pressed
185
186   lv_obj_add_event_cb(ui->camera_imgbtn_photo, camera_imgbtn_photo_event_handler,
187 LV_EVENT_ALL, NULL);
188   lv_obj_add_event_cb(ui->camera_imgbtn_home, camera_imgbtn_home_event_handler, LV_EVENT_ALL,
189 NULL);
190   lv_obj_add_event_cb(ui->camera, camera_screen_gesture_event_handler, LV_EVENT_ALL, NULL);
191   create_camera_task();
192 }
```

Obtain the raw data of the camera through esp_camera_fb_get(), store the data in fb_buf, and release the camera for the next acquisition of camera data.

```
42      fb = esp_camera_fb_get();
43      fb_buf = fb;
44      esp_camera_fb_return(fb);
```

Obtain the raw data from the camera using esp_camera_fb_get() function, store the data in fb_buf, and release the camera for the next data acquisition.

```
46      for(int i=0;i<fb_buf->len;i+=2){
47          uint8_t temp=0;
48          temp=fb_buf->buf[i];
49          fb_buf->buf[i]=fb_buf->buf[i+1];
```

```
50          fb_buf->buf[i+1]=temp;
51      }
```

Assign the processed data to the lvgl img component and display it on the screen.

```
52      photo_show.data = fb_buf->buf;
53      lv_img_set_src(guider_camera_ui.camera_video,&photo_show);
```

Create a video streaming thread to have the ESP32-S3 collect camera data and display it on the screen.

```
12  //Create camera task thread
13  void create_camera_task(void){
14    if(camera_task_flag==0){
15      camera_task_flag=1;
16      ui_set_photo_show();
17      xTaskCreate(loopTask_camera, "loopTask_camera", 8192, NULL, 1, &cameraTaskHandle);
18    }
19    else{
20      Serial.println("loopTask_camera is running...");
21    }
22  }
```

When camera_task_flag is set to 0, the video thread will exit and delete the handle cameraTaskHandle. Therefore, when calling the stop_camera_task() function, just set camera_task_flag to 0, and then determine whether cameraTaskHandle has been deleted.

```
24  //Close the camera thread
25  void stop_camera_task(void){
26    if(camera_task_flag==1){
27      camera_task_flag=0;
28      while(1){
29        if (eTaskGetState(cameraTaskHandle) == eDeleted){
30            break;
31        }
32        vTaskDelay(10);
33      }
34      Serial.println("loopTask_camera deleted!");
35    }
36  }
```

Call the function to calculate the number of members in the current linked list,

```
86          int photo_index = list_count_number(list_picture);
87          Serial.printf("photo_index: %d\r\n",photo_index);
```

Save the obtained camera raw image data as a 16-bit deep bmp file.

```
90          write_rgb565_to_bmp((char *)path.c_str(), fb->buf, fb->len, fb->height, fb->width);
```

Call the tail insertion method to add the file name to the linked list.

```
91          list_insert_tail(list_picture,(char *)path.c_str());
```

Picture button home, nothing is executed here, only the button trigger information is printed.

```
103 //Click the home icon, callback function: goes to the main ui interface
104 static void camera_imgbtn_home_event_handler(lv_event_t *e)
```

Create an operation object and assign it to the camera member in the ui structure.

```
133    //Write the camera interface
134    ui->camera = lv_obj_create(NULL);
```

Generate a linked list according to the picture folder in the sd card.

```
135    setup_list_head_picture();    //Generate a linked list based on the SD card's picture folder
```

Configure the image data for the two image buttons used in the camera interface.

```
136    lv_img_home_init();
137    lv_img_camera_init();
```

Apply a style to the image button to make it move down 5 pixels every time it is pressed.

```
139    /*Init the pressed style*/
140    static lv_style_t style_pr;//Apply for a style
141    lv_style_init(&style_pr);  //Initialize it
142    lv_style_set_translate_y(&style_pr, 5);//Style: Every time you trigger, move down 5 pixels
```

Add an img component to the ui structure's camera member to display the image.

```
167    //Write codes for camera_video
168    ui->camera_video = lv_img_create(ui->camera);
169    lv_obj_set_pos(ui->camera_video, 0, 0);
170    lv_obj_set_size(ui->camera_video, 240, 240);
```

Add an image button with the camera icon as its image content. Set the position and size of the image. Add a downshift effect to the button when it is pressed.

```
172    //Write codes for camera_photo
173    ui->camera_imgbtn_photo = lv_imgbtn_create(ui->camera);
174    lv_obj_set_pos(ui->camera_imgbtn_photo, 20, 240);
175    lv_obj_set_size(ui->camera_imgbtn_photo, 80, 80);
176    lv_img_set_src(ui->camera_imgbtn_photo, &img_camera);
177    lv_obj_add_style(ui->camera_imgbtn_photo, &style_pr, LV_STATE_PRESSED);//Triggered when the
       button is pressed
```

Add a picture button, and set the picture content as home. Set the picture position and size, and add a pressed effect that moves the button down.

```
178    //Write codes for camera_return
179    ui->camera_imgbtn_home = lv_imgbtn_create(ui->camera);
180    lv_obj_set_pos(ui->camera_imgbtn_home, 140, 240);
181    lv_obj_set_size(ui->camera_imgbtn_home, 80, 80);
182    lv_img_set_src(ui->camera_imgbtn_home, &img_home);
183    lv_obj_add_style(ui->camera_imgbtn_home, &style_pr, LV_STATE_PRESSED);//Triggered when the
       button is pressed
```

Associate each of the two image buttons with their respective callback functions.

```
186    lv_obj_add_event_cb(ui->camera_imgbtn_photo, camera_imgbtn_photo_event_handler,
       LV_EVENT_ALL, NULL);
187    lv_obj_add_event_cb(ui->camera_imgbtn_home, camera_imgbtn_home_event_handler, LV_EVENT_ALL,
       NULL);
188    lv_obj_add_event_cb(ui->camera, camera_screen_gesture_event_handler, LV_EVENT_ALL, NULL);
```

Call the function to create a camera thread to display the camera image on the screen.

```
189    create_camera_task();
```

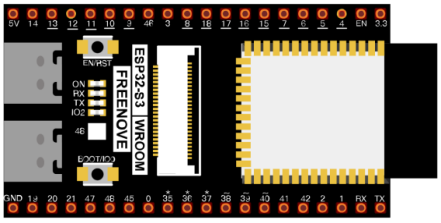
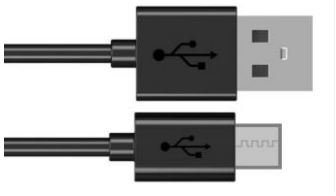
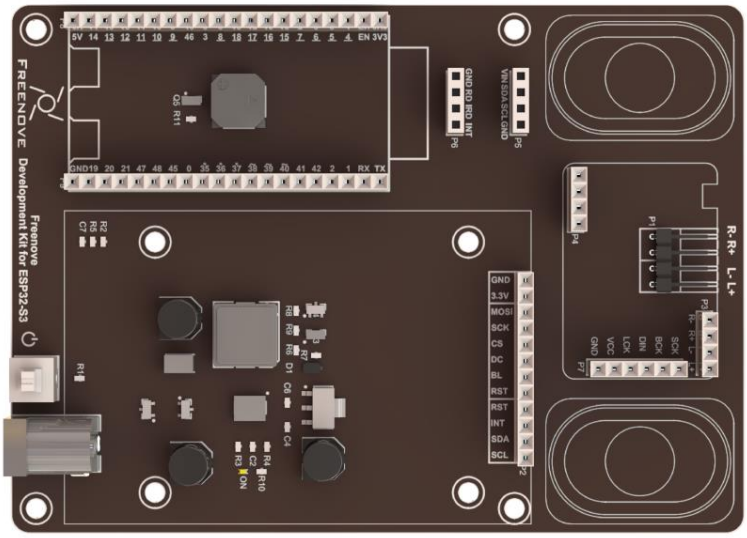
Chapter 16 LVGL Picture

In this chapter, we will learn how to create an image viewer.

Project 16.1 LVGL Picture

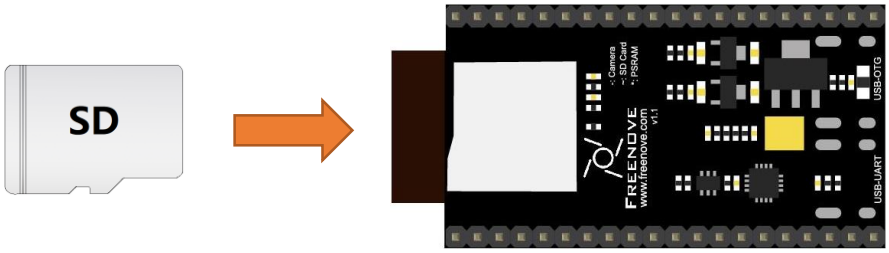
In this project, we will learn how to view pictures saved in SD card.

Component List

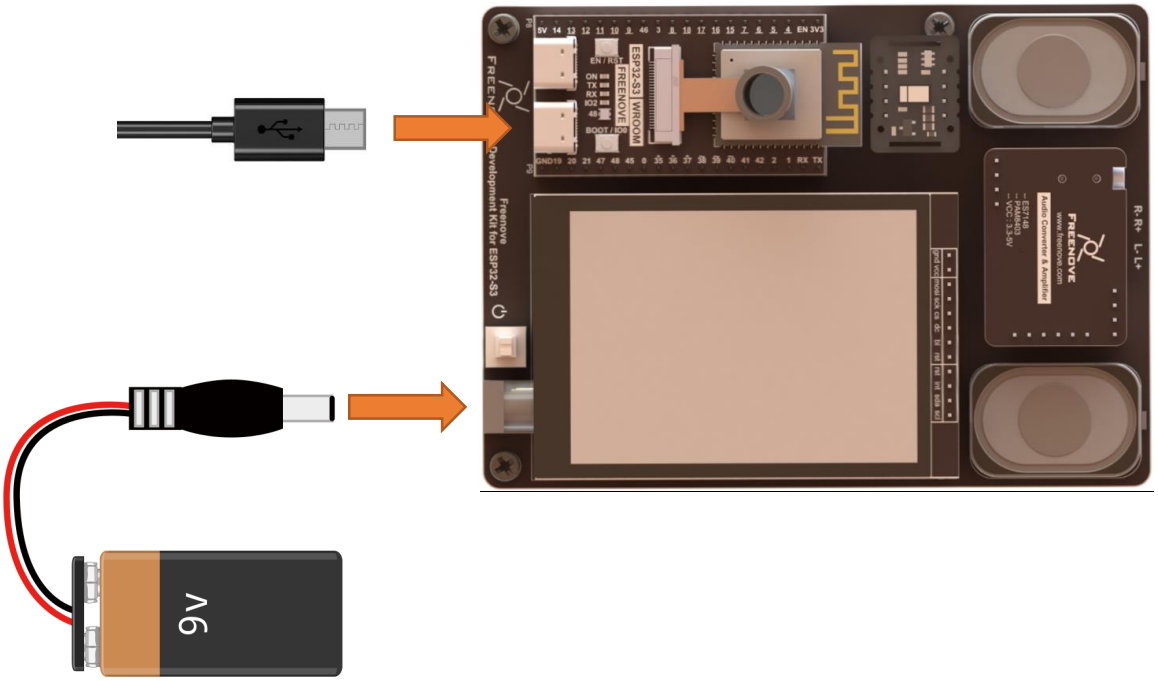
ESP32-S3 WROOM x1 	USB cable x1 	2.8-inch screen 
ESP32-S3 WROOM Shield x1 		Card reader x1 (random color) 
9V battery cable x1 	9V battery x1 (Not included, prepared by yourself) 	SD card x1 

Circuit

If you have not yet used the SD card, please refer to Chapter 4. Click here to navigate back to Chapter 4.
Before connecting the USB cable, insert the SD card into the SD card slot on the back of the ESP32-S3.



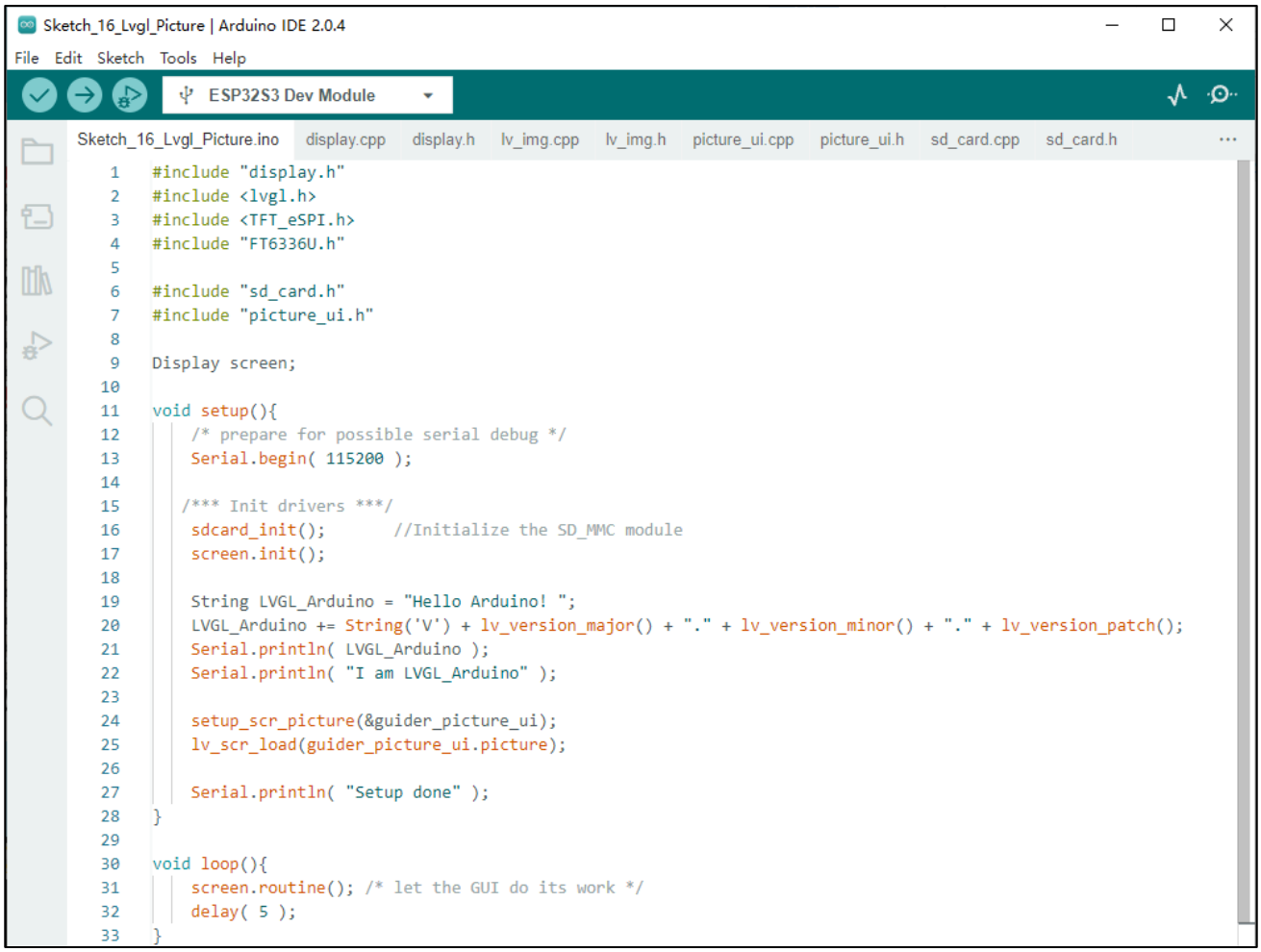
Connect Freenove ESP32-S3 to the computer using the USB cable.



Sketch

Before starting the project, please make sure that there is a folder named picture in your sd card, and the pictures inside are of bmp format, and their resolution is 240*240.

Sketch_16_LVGL_Picture



The following is the program code:

```
#include "display.h"
#include <lvgl.h>
#include <TFT_eSPI.h>
#include "FT6336U.h"

#include "sd_card.h"
#include "picture_ui.h"

Display screen;

void setup(){
    /* prepare for possible serial debug */
    Serial.begin( 115200 );
```

```
14
15      /*** Init drivers ***/
16      sdcard_init();       //Initialize the SD_MMC module
17      screen.init();       //Initialize the Screen
18
19      String LVGL_Arduino = "Hello Arduino! ";
20      LVGL_Arduino += String('V') + lv_version_major() + "." + lv_version_minor() + "." +
    lv_version_patch();
21      Serial.println( LVGL_Arduino );
22      Serial.println( "I am LVGL_Arduino" );
23
24      setup_scr_picture(&guider_picture_ui);
25      lv_scr_load(guider_picture_ui.picture);
26
27      Serial.println( "Setup done" );
28  }
29
30  void loop(){
31      screen.routine(); /* let the GUI do its work */
32      delay( 5 );
33  }
```

Configure the image viewing interface and load this interface.

```
22      setup_scr_picture(&guider_picture_ui);
23      lv_scr_load(guider_picture_ui.picture);
```

picture_ui.h

Declare the functions so that they can be called in the ino file.

```
1   #ifndef __PICTURE_UI_H
2   #define __PICTURE_UI_H
3
4   #include "lvgl.h"
5   #include "Arduino.h"
6
7   typedef struct lvgl_picture{
8     lv_obj_t *picture;
9     lv_obj_t *picture_left;
10    lv_obj_t *picture_right;
11    lv_obj_t *picture_show;
12    lv_obj_t *picture_home;
13  }lvgl_picture_ui;
14
15  extern lvgl_picture_ui guider_picture_ui;    //picture ui structure
16  void setup_scr_picture(lvgl_picture_ui *ui); //Parameter configuration function on the picture
    screen
17  void picture_imgbtn_display(char *name);     //Read the image file and display it
```

```
18
19  #endif
```

picture_ui.cpp

```
1   #include "picture_ui.h"
2   #include "sd_card.h"
3   #include "lv_img.h"
4
5   lv_img_dsc_t pic_show;              //apply an lvgl image variable
6   lvgl_picture_ui guider_picture_ui;  //picture ui structure
7   static int picture_index_num=1;     //index number of the picture
8
9   //Initialize an lvgl image variable
10  void ui_set_pic_show(void){
11      lv_img_header_t header;
12      header.always_zero = 0;
13      header.w = 240;
14      header.h = 240;
15      header.cf = LV_IMG_CF_TRUE_COLOR;
16      pic_show.header = header;
17      pic_show.data_size = 240 * 240 * 2;
18      pic_show.data = NULL;
19  }
20
21  //Click the left icon, callback function: show the last picture
22  static void picture_imgbtn_left_event_handler(lv_event_t *e) {
23    lv_event_code_t code = lv_event_get_code(e);
24    if (code == LV_EVENT_CLICKED) {
25      Serial.println("Clicked the left button.");
26      picture_index_num--;
27      if(picture_index_num < 1)
28        picture_index_num=list_count_number(list_picture);
29      picture_imgbtn_display(list_find_node(list_picture,picture_index_num));
30    }
31  }
32
33  //Click the right icon, callback function: show the next picture
34  static void picture_imgbtn_right_event_handler(lv_event_t *e) {
35    lv_event_code_t code = lv_event_get_code(e);
36    if (code == LV_EVENT_CLICKED) {
37      Serial.println("Clicked the right button.");
38      picture_index_num++;
39      if(picture_index_num > list_count_number(list_picture))
40        picture_index_num=1;
41      picture_imgbtn_display(list_find_node(list_picture,picture_index_num));
```

```
  }
}

//Click the home icon, callback function: goes to the main ui interface
static void picture_imgbtn_home_event_handler(lv_event_t *e) {
  lv_event_code_t code = lv_event_get_code(e);
  switch (code) {
    case LV_EVENT_CLICKED:
      {
        Serial.println("Clicked the return button.");
      }
      break;
    case LV_EVENT_RELEASED:
      {
        /*
        if (!lv_obj_is_valid(guider_main_ui.main))
          setup_scr_main(&guider_main_ui);
        lv_disp_t *d = lv_obj_get_disp(lv_scr_act());
        if (d->prev_scr == NULL && d->scr_to_load == NULL)
          lv_scr_load(guider_main_ui.main);
        lv_obj_del(guider_picture_ui.picture);
        */
      }
      break;
    default:
      break;
  }
}

//Parameter configuration function on the picture screen
void setup_scr_picture(lvgl_picture_ui *ui){
  //Write codes picture
  ui->picture = lv_obj_create(NULL);
  lv_img_home_init();
  lv_img_left_init();
  lv_img_right_init();

  /*Init the pressed style*/
  static lv_style_t style_pr;//Apply for a style
  lv_style_init(&style_pr);  //Initialize it
  lv_style_set_translate_y(&style_pr, 5);//Style: Every time you trigger, move down 5 pixels

  //Write codes picture_left
  ui->picture_left = lv_imgbtn_create(ui->picture);
```

```
86    lv_obj_set_pos(ui->picture_left, 10, 250);
87    lv_obj_set_size(ui->picture_left, 60, 60);
88    lv_img_set_src(ui->picture_left, &img_left);
89    lv_obj_add_style(ui->picture_left, &style_pr, LV_STATE_PRESSED);
90
91    //Write codes picture_right
92    ui->picture_right = lv_imgbtn_create(ui->picture);
93    lv_obj_set_pos(ui->picture_right, 170, 250);
94    lv_obj_set_size(ui->picture_right, 60, 60);
95    lv_img_set_src(ui->picture_right, &img_right);
96    lv_obj_add_style(ui->picture_right, &style_pr, LV_STATE_PRESSED);
97
98    //Write codes picture_home
99    ui->picture_home = lv_imgbtn_create(ui->picture);
100   lv_obj_remove_style_all(ui->picture_home);
101   lv_obj_set_pos(ui->picture_home, 80, 240);
102   lv_obj_set_size(ui->picture_home, 80, 80);
103   lv_img_set_src(ui->picture_home, &img_home);
104   lv_obj_add_style(ui->picture_home, &style_pr, LV_STATE_PRESSED);
105
106   //Write codes picture_show
107   ui->picture_show = lv_img_create(ui->picture);
108   lv_obj_set_pos(ui->picture_show, 0, 0);
109   lv_obj_set_size(ui->picture_show, 240, 240);
110
111   ui_set_pic_show();
112   setup_list_head_picture();
113   picture_index_num = list_count_number(list_picture);
114   picture_imgbtn_display(list_find_node(list_picture,picture_index_num));
115
116   lv_obj_add_event_cb(ui->picture_left, picture_imgbtn_left_event_handler, LV_EVENT_ALL,
      NULL);
117   lv_obj_add_event_cb(ui->picture_right, picture_imgbtn_right_event_handler, LV_EVENT_ALL,
      NULL);
118   lv_obj_add_event_cb(ui->picture_home, picture_imgbtn_home_event_handler, LV_EVENT_ALL,
      NULL);
119 }
120
121 //Read the image file and display it
122 void picture_imgbtn_display(char *name){
123   if(name!=NULL){
124     char buf_picture_name[100]={"S:"};
125     strcat(buf_picture_name,PICTURE_FOLDER);
126     strcat(buf_picture_name,"/");
```

```
127     strcat(buf_picture_name,name);
128     lv_img_set_src(guider_picture_ui.picture_show, buf_picture_name);
129   }
130   else{
131     lv_img_set_src(guider_picture_ui.picture_show, LV_SYMBOL_DUMMY "The picture folder has no
      files.");
132   }
133 }
```

Declare the header files.

```
1 #include "picture_ui.h"
2 #include "sd_card.h"
3 #include "lv_img.h"
```

“pic_show” is used to store image data. “guider_picture_ui” is used to configure the interface. “picture_index_num” is used to record the position of the picture.

```
5 lv_img_dsc_t pic_show;              //apply an lvgl image variable
6 lvgl_picture_ui guider_picture_ui;  //picture ui structure
7 static int picture_index_num=1;     //index number of the picture
```

Configure the parameters of the variable pic_show.

```
9  //Initialize an lvgl image variable
10 void ui_set_pic_show(void){
11     lv_img_header_t header;
12     header.always_zero = 0;
13     header.w = 240;
14     header.h = 240;
15     header.cf = LV_IMG_CF_TRUE_COLOR;
16     pic_show.header = header;
17     pic_show.data_size = 240 * 240 * 2;
18     pic_show.data = NULL;
19 }
```

In the image display interface, we will use 3 image buttons. Here, we need to configure the image data.

```
75   lv_img_home_init();
76   lv_img_left_init();
77   lv_img_right_init();
```

Set a style to configure the image button so that it moves down 5 pixels each time it is pressed.

```
80   static lv_style_t style_pr;//Apply for a style
81   lv_style_init(&style_pr);  //Initialize it
82   lv_style_set_translate_y(&style_pr, 5);//Style: Every time you trigger, move down 5 pixels
```

Apply for a picture button component, set its position, size, displayed picture content, and add the effect of the button being pressed.

```
84 //Write codes picture_left
85   ui->picture_left = lv_imgbtn_create(ui->picture);
86   lv_obj_set_pos(ui->picture_left, 10, 250);
87   lv_obj_set_size(ui->picture_left, 60, 60);
88   lv_img_set_src(ui->picture_left, &img_left);
```

```
89    lv_obj_add_style(ui->picture_left, &style_pr, LV_STATE_PRESSED);
```

Call the ui_set_pic_show() function to configure pic_show.

```
111   ui_set_pic_show();
```

Search the "picture" folder on the SD card and add all the BMP file names to the linked list "list_picture". Please note that you can directly put custom pictures in the SD card and display them on the screen, but the resolution of the pictures must be 240x240. Resolutions that are too large or too small may not be displayed well.

```
112   setup_list_head_picture();
```

Calculate the number of members in the linked list list_picture, and assign the value back to picture_index_num.

```
113   picture_index_num = list_count_number(list_picture);
```

Call the list_find_node() function to obtain the file name of the node specified in the linked list list_picture. Call the picture_imgbtn_display() function to load and display the picture on the screen.

```
114   picture_imgbtn_display(list_find_node(list_picture,picture_index_num));
```

Associate the three picture buttons with their respective callback functions.

```
116   lv_obj_add_event_cb(ui->picture_left, picture_imgbtn_left_event_handler, LV_EVENT_ALL,
      NULL);
117   lv_obj_add_event_cb(ui->picture_right, picture_imgbtn_right_event_handler, LV_EVENT_ALL,
      NULL);
118   lv_obj_add_event_cb(ui->picture_home, picture_imgbtn_home_event_handler, LV_EVENT_ALL,
      NULL);
```

Image button display function. When the file name is empty, print a prompt message. When the file name is not empty, generate the correct image path according to the file name, and load this file to the image component.

```
121 //Read the image file and display it
122 void picture_imgbtn_display(char *name){
123   if(name!=NULL){
124     char buf_picture_name[100]={"S:"};
125     strcat(buf_picture_name,PICTURE_FOLDER);
126     strcat(buf_picture_name,"/");
127     strcat(buf_picture_name,name);
128     lv_img_set_src(guider_picture_ui.picture_show, buf_picture_name);
129   }
130   else{
131     lv_img_set_src(guider_picture_ui.picture_show, LV_SYMBOL_DUMMY "The picture folder has no
      files.");
132   }
133 }
```

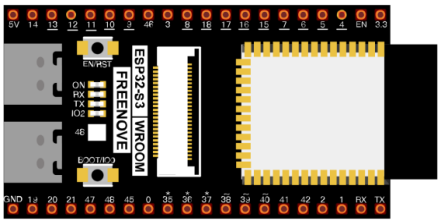
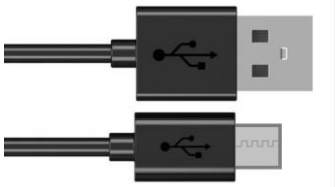
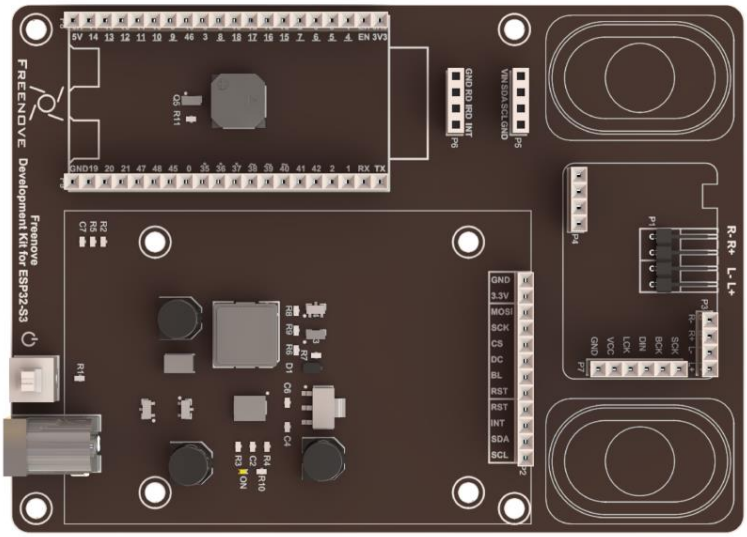
Chapter 17 LVGL Music

In this chapter, we will learn how to create a simple music player.

Project 17.1 LVGL Music

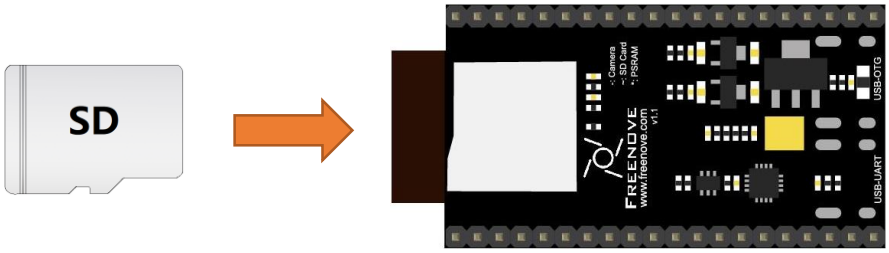
In this project, we will learn how to make a simple mp3 player and play the music of mp3 format in the SD card through the speaker.

Component List

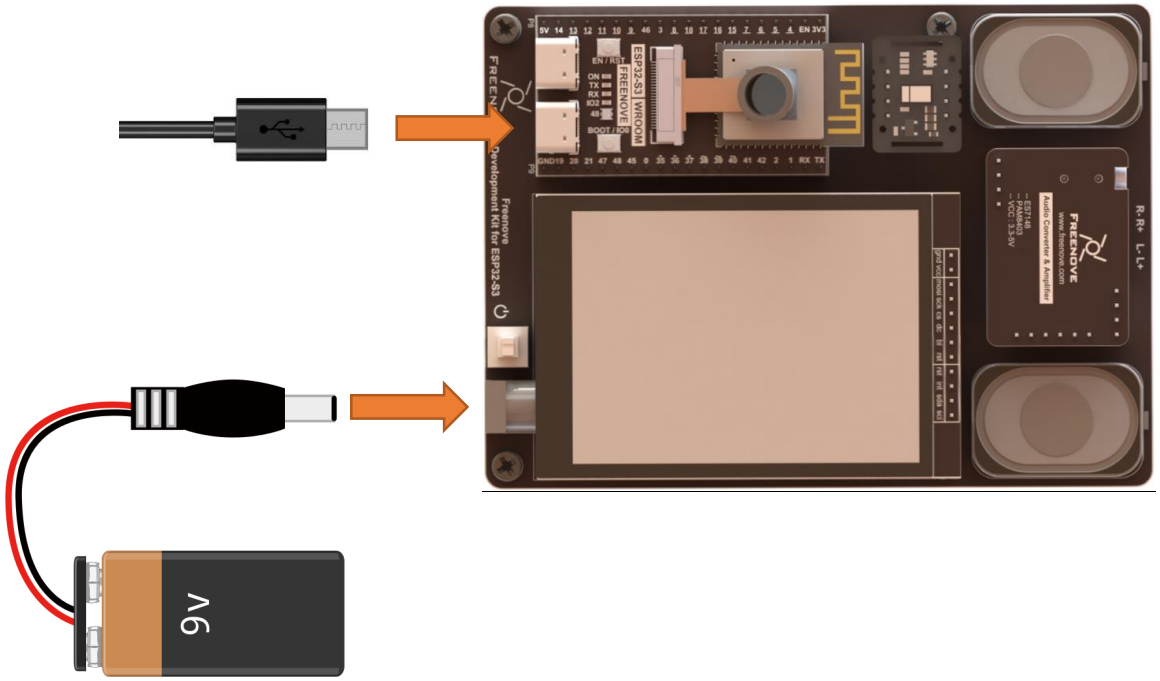
ESP32-S3 WROOM x1 	USB cable x1 	2.8-inch screen 
ESP32-S3 WROOM Shield x1 		Card reader x1 (random color) 
9V battery cable x1 	9V battery x1 (Not included, prepared by yourself) 	SD card x1 

Circuit

If you have not yet used the SD card, please refer to Chapter 4. Click here to navigate back to Chapter 4.
Before connecting the USB cable, insert the SD card into the SD card slot on the back of the ESP32-S3.



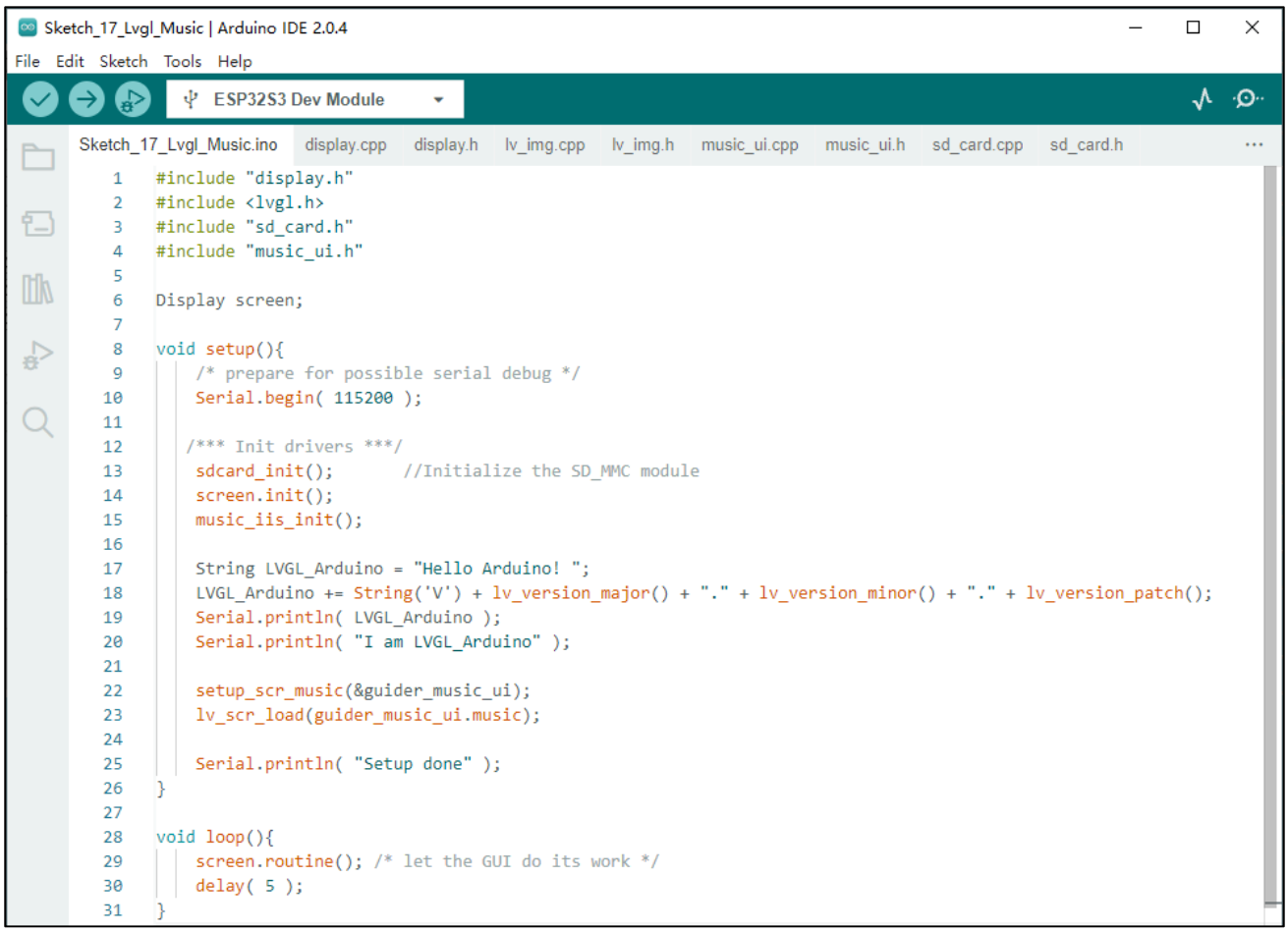
Connect Freenove ESP32-S3 to the computer using the USB cable.



Sketch

Before starting the project, please make sure that there is a folder named music in your sd card, and the songs inside are in mp3 format.

Sketch_17_LVGL_Music



The following is the program code:

```
#include "display.h"
#include <lvgl.h>
#include "sd_card.h"
#include "music_ui.h"

Display screen;

void setup(){
    /* prepare for possible serial debug */
    Serial.begin( 115200 );

   /*** Init drivers ***/
    sdcard_init();       //Initialize the SD_MMC module
    screen.init();
```

```
    music_iis_init();

    String LVGL_Arduino = "Hello Arduino! ";
    LVGL_Arduino += String('V') + lv_version_major() + "." + lv_version_minor() + "." +
lv_version_patch();
    Serial.println( LVGL_Arduino );
    Serial.println( "I am LVGL_Arduino" );

    setup_scr_music(&guider_music_ui);
    lv_scr_load(guider_music_ui.music);

    Serial.println( "Setup done" );
}

void loop(){
    screen.routine(); /* let the GUI do its work */
    delay( 5 );
}
```

Configure the music player interface and load this interface.

```
    setup_scr_music(&guider_music_ui);
    lv_scr_load(guider_music_ui.music);
```

music_ui.h

Declare the functions so that they can be called in the ino file.

```
#ifndef __MUSIC_H
#define __MUSIC_H

#include "lvgl.h"
#include "Arduino.h"

#define I2S_BCLK    42
#define I2S_DOUT    41
#define I2S_LRC     14

typedef struct lvgl_music{
  lv_obj_t *music;
  lv_obj_t *music_imgbtn_home;
  lv_obj_t *music_imgbtn_left;
  lv_obj_t *music_imgbtn_play;
  lv_obj_t *music_imgbtn_stop;
  lv_obj_t *music_imgbtn_right;
  lv_obj_t *music_label;
  lv_obj_t *music_slider_valume;
  lv_obj_t *music_slider_label;
  lv_obj_t *music_bar_time;
```

```
}lvgl_music_ui;

extern lvgl_music_ui guider_music_ui;//music ui structure
void setup_scr_music(lvgl_music_ui *ui);//Parameter configuration function on the music screen

void music_set_label_text(char *text);      //Set the label display content
void music_iis_init(void);                  //Initialize the audio interface
void music_set_volume(int volume);          //Set the volume: 0-21
int music_read_volume(void);                //Query volume
void music_load_mp3(char* name);            //load the mp3
void music_pause_resume(void);              //Pause/play the music
void music_stop(void);                      //Stop the music
bool music_is_running(void);                //Whether the music is running
long music_get_total_playing_time(void);    //Gets how long the music player has been playing
long music_get_file_duration(void);         //Obtain the playing time of the music file
bool music_set_play_position(int second);   //Set play position
long music_read_play_position(void);        //Gets the current playing time of the music
void music_loop(void);                      //Non-blocking audio execution function
void loopTask_music(void *pvParameters);    //music player task thread
void start_music_task(void);                //Create music task thread
void stop_music_task(void);                 //Close the music thread
int music_task_is_running(void);            //Check whether the thread is running
//optional
void audio_info(const char *info);
void audio_eof_mp3(const char *info);

#endif
```

music_ui.cpp

```
#include "music_ui.h"
#include "sd_card.h"
#include "stdlib.h"
#include "string.h"
#include "Audio.h"
#include "FS.h"
#include "SD_MMC.h"
#include "lv_img.h"

lvgl_music_ui guider_music_ui;//music ui structure
int music_button_state = 0;    //UI Button status
int music_index_num = 1;       //index number of the music

Audio audio;
int music_task_flag = 0;        //music thread running flag
TaskHandle_t musicTaskHandle;  //music thread task handle
```

```
17
18  //Click the home icon, callback function: goes to the main ui interface
19  static void music_imgbtn_home_event_handler(lv_event_t *e) {
20    lv_event_code_t code = lv_event_get_code(e);
21    switch (code) {
22      case LV_EVENT_CLICKED:
23        {
24          Serial.println("Clicked the home button.");
25        }
26        break;
27      case LV_EVENT_RELEASED:
28        {
29          /*
30          stop_music_task();//Close the thread when exiting the musci ui interface
31          if (!lv_obj_is_valid(guider_main_ui.main))
32            setup_scr_main(&guider_main_ui);
33          lv_disp_t *d = lv_obj_get_disp(lv_scr_act());
34          if (d->prev_scr == NULL && d->scr_to_load == NULL)
35            lv_scr_load(guider_main_ui.main);
36          //After exiting the interface, delete the interface to ensure that the next time you
    enter the initial interface
37          lv_obj_del(guider_music_ui.music);
38          */
39        }
40        break;
41      default:
42        break;
43    }
44  }
45
46  //Click the left icon, callback function: play the last song
47  static void music_imgbtn_left_event_handler(lv_event_t *e) {
48    lv_event_code_t code = lv_event_get_code(e);
49    switch (code) {
50      case LV_EVENT_CLICKED:
51        {
52          Serial.println("Clicked the left button.");
53        }
54        break;
55      case LV_EVENT_RELEASED:
56        {
57          Serial.println("Play the last song.");
58          music_index_num--;
59          if (music_index_num < 1)
```

```
            music_index_num = list_count_number(list_music);
          stop_music_task();
          lv_img_set_src(guider_music_ui.music_imgbtn_play, &img_pause);
          music_button_state = 0;
          char *buf_music_name = list_find_node(list_music, music_index_num);
          music_set_label_text(buf_music_name);
          if(buf_music_name!=NULL){
            char buf_music_path[255] = {MUSIC_FOLDER};
            strcat(buf_music_path, "/");
            strcat(buf_music_path, buf_music_name);
            Serial.println(buf_music_path);
            music_load_mp3(buf_music_path);
            start_music_task();
            lv_img_set_src(guider_music_ui.music_imgbtn_play, &img_playing);
            music_button_state = 1;
          }
        }
        break;
      default:
        break;
    }
}

//Click the right icon, callback function: play the next song
static void music_imgbtn_right_event_handler(lv_event_t *e) {
  lv_event_code_t code = lv_event_get_code(e);
  switch (code) {
    case LV_EVENT_CLICKED:
      {
        Serial.println("Clicked the right button.");
      }
      break;
    case LV_EVENT_RELEASED:
      {
        Serial.println("Play the next song.");
        music_index_num++;
        if (music_index_num > list_count_number(list_music))
          music_index_num = 1;

        stop_music_task();
        lv_img_set_src(guider_music_ui.music_imgbtn_play, &img_pause);
        music_button_state = 0;
        char *buf_music_name = list_find_node(list_music, music_index_num);
        music_set_label_text(buf_music_name);
```

```
104         if(buf_music_name!=NULL){
105           char buf_music_path[255] = {MUSIC_FOLDER};
106           strcat(buf_music_path, "/");
107           strcat(buf_music_path, buf_music_name);
108           Serial.println(buf_music_path);
109           music_load_mp3(buf_music_path);
110           start_music_task();
111           lv_img_set_src(guider_music_ui.music_imgbtn_play, &img_playing);
112           music_button_state = 1;
113         }
114       }
115       break;
116     default:
117       break;
118   }
119 }
120
121 //Click the play icon, callback function: play or pause the music
122 static void music_imgbtn_play_event_handler(lv_event_t *e) {
123   lv_event_code_t code = lv_event_get_code(e);
124   switch (code) {
125     case LV_EVENT_CLICKED:
126       {
127         Serial.println("Clicked the play button.");
128       }
129       break;
130     case LV_EVENT_RELEASED:
131       {
132         music_button_state = !music_button_state;
133         if (music_button_state == 1)
134           lv_img_set_src(guider_music_ui.music_imgbtn_play, &img_playing);
135         else
136           lv_img_set_src(guider_music_ui.music_imgbtn_play, &img_pause);
137
138         //Gets whether a music task currently exists
139         int is_task_running = music_task_is_running();
140         if (is_task_running == 1) {  //If so, pause or play it
141           music_pause_resume();
142         } else {
143           /*If there is no music thread currently, load the music name first, and then create
    the audio thread*/
144           char *buf_music_name = list_find_node(list_music, music_index_num);
145           if(buf_music_name!=NULL){
146             char buf_music_path[255] = {MUSIC_FOLDER};
```

```
147                 strcat(buf_music_path, "/");
148                 strcat(buf_music_path, buf_music_name);
149                 Serial.println(buf_music_path);
150                 music_load_mp3(buf_music_path);
151                 start_music_task();
152               }
153             }
154           }
155           break;
156         default:
157           break;
158       }
159     }
160
161     //Click the stop icon, callback function: stop the music
162     static void music_imgbtn_stop_event_handler(lv_event_t *e) {
163       lv_event_code_t code = lv_event_get_code(e);
164       switch (code) {
165         case LV_EVENT_CLICKED:
166           {
167             Serial.println("Clicked the stop button.");
168           }
169           break;
170         case LV_EVENT_RELEASED:
171           {
172             lv_img_set_src(guider_music_ui.music_imgbtn_play, &img_pause);
173             stop_music_task();
174             lv_bar_set_value(guider_music_ui.music_bar_time, 0, LV_ANIM_OFF);
175             music_button_state = 0;
176             Serial.println("The music has stop.");
177           }
178           break;
179         default:
180           break;
181       }
182     }
183
184     static void music_slider_change_event_handler(lv_event_t * e){
185         lv_obj_t *slider = lv_event_get_target(e);
186         char buf[16];
187         int volume = (int)lv_slider_get_value(slider);
188         lv_snprintf(buf, sizeof(buf), "Volume:%d", volume);
189         lv_label_set_text(guider_music_ui.music_slider_label, buf);
190         int last_volume = music_read_volume();
```

```
    if(volume != last_volume)
      music_set_volume(volume);
}


//Parameter configuration function on the music screen
void setup_scr_music(lvgl_music_ui *ui) {
  ui->music = lv_obj_create(NULL);
  lv_img_home_init();
  lv_img_left_init();
  lv_img_right_init();
  lv_img_pause_init();
  lv_img_playing_init();
  lv_img_stop_init();
  setup_list_head_music();

  /*Init the pressed style*/
  static lv_style_t style_pr;//Apply for a style
  lv_style_init(&style_pr);  //Initialize it
  lv_style_set_translate_y(&style_pr, 5);//Style: Every time you trigger, move down 5 pixels

  ui->music_imgbtn_home = lv_imgbtn_create(ui->music);
  lv_obj_set_pos(ui->music_imgbtn_home, 80, 20);
  lv_obj_set_size(ui->music_imgbtn_home, 80, 80);
  lv_img_set_src(ui->music_imgbtn_home, &img_home);
  lv_obj_add_style(ui->music_imgbtn_home, &style_pr, LV_STATE_PRESSED);//Triggered when the
button is pressed

  ui->music_slider_label = lv_label_create(ui->music);
  lv_obj_set_size(ui->music_slider_label, 160, 20);
  lv_obj_align(ui->music_slider_label, LV_ALIGN_CENTER, 40, -30);

  ui->music_slider_valume = lv_slider_create(ui->music);
  lv_obj_set_size(ui->music_slider_valume, 180, 10);
  lv_obj_set_pos(ui->music_slider_valume, 30, 150);
  lv_slider_set_mode(ui->music_slider_valume, LV_SLIDER_MODE_NORMAL);
  lv_slider_set_range(ui->music_slider_valume, 0, 21);
  lv_slider_set_value(ui->music_slider_valume, 10, LV_ANIM_OFF);

  ui->music_label = lv_label_create(ui->music);
  lv_obj_set_pos(ui->music_label, 40, 180);
  lv_obj_set_size(ui->music_label, 160, 20);
  music_set_label_text(list_find_node(list_music, music_index_num));
  lv_label_set_long_mode(ui->music_label, LV_LABEL_LONG_SCROLL_CIRCULAR );
```

```
234     lv_obj_set_style_text_align(ui->music_label, LV_TEXT_ALIGN_CENTER, 0);
235
236     ui->music_imgbtn_left = lv_imgbtn_create(ui->music);
237     lv_obj_set_pos(ui->music_imgbtn_left, 0, 220);
238     lv_obj_set_size(ui->music_imgbtn_left, 60, 60);
239     lv_img_set_src(ui->music_imgbtn_left, &img_left);
240     lv_obj_add_style(ui->music_imgbtn_left, &style_pr, LV_STATE_PRESSED);
241
242     ui->music_imgbtn_play = lv_imgbtn_create(ui->music);
243     lv_obj_set_pos(ui->music_imgbtn_play, 60, 220);
244     lv_obj_set_size(ui->music_imgbtn_play, 60, 60);
245     lv_img_set_src(ui->music_imgbtn_play, &img_pause);
246     lv_obj_add_style(ui->music_imgbtn_play, &style_pr, LV_STATE_PRESSED);
247
248     ui->music_imgbtn_stop = lv_imgbtn_create(ui->music);
249     lv_obj_set_pos(ui->music_imgbtn_stop, 120, 220);
250     lv_obj_set_size(ui->music_imgbtn_stop, 60, 60);
251     lv_img_set_src(ui->music_imgbtn_stop, &img_stop);
252     lv_obj_add_style(ui->music_imgbtn_stop, &style_pr, LV_STATE_PRESSED);
253
254     ui->music_imgbtn_right = lv_imgbtn_create(ui->music);
255     lv_obj_set_pos(ui->music_imgbtn_right, 180, 220);
256     lv_obj_set_size(ui->music_imgbtn_right, 60, 60);
257     lv_img_set_src(ui->music_imgbtn_right, &img_right);
258     lv_obj_add_style(ui->music_imgbtn_right, &style_pr, LV_STATE_PRESSED);
259
260     ui->music_bar_time = lv_bar_create(ui->music);
261     lv_obj_set_size(ui->music_bar_time, 240, 5);
262     lv_obj_set_pos(ui->music_bar_time, 0, 290);
263     lv_slider_set_mode(ui->music_bar_time, LV_SLIDER_MODE_NORMAL);
264     lv_bar_set_range(ui->music_bar_time, 0, 100);
265     lv_bar_set_value(ui->music_bar_time, 0, LV_ANIM_OFF);
266
267     lv_obj_add_event_cb(ui->music_imgbtn_home, music_imgbtn_home_event_handler, LV_EVENT_ALL,
    NULL);
268     lv_obj_add_event_cb(ui->music_imgbtn_left, music_imgbtn_left_event_handler, LV_EVENT_ALL,
    NULL);
269     lv_obj_add_event_cb(ui->music_imgbtn_right, music_imgbtn_right_event_handler, LV_EVENT_ALL,
    NULL);
270     lv_obj_add_event_cb(ui->music_imgbtn_play, music_imgbtn_play_event_handler, LV_EVENT_ALL,
    NULL);
271     lv_obj_add_event_cb(ui->music_imgbtn_stop, music_imgbtn_stop_event_handler, LV_EVENT_ALL,
    NULL);
272
```

```
    lv_obj_add_event_cb(ui->music_slider_valume, music_slider_change_event_handler,
LV_EVENT_ALL, NULL);
}

//Set the label display content
void music_set_label_text(char *text){
  if(text!=NULL){
    lv_label_set_text(guider_music_ui.music_label, text);
  }
  else{
    lv_label_set_text(guider_music_ui.music_label, "The music folder has no files.");
  }
}

//Initialize the audio interface
void music_iis_init(void) {
  audio.setPinout(I2S_BCLK, I2S_LRC, I2S_DOUT);
}
//Set the volume: 0-21
void music_set_volume(int volume) {
  audio.setVolume(volume);
}
//Query volume
int music_read_volume(void) {
  return audio.getVolume();
}
//load the mp3
void music_load_mp3(char *name) {
  audio.connecttoFS(SD_MMC, name);
}
//Pause/play the music
void music_pause_resume(void) {
  audio.pauseResume();
}
//Stop the music
void music_stop(void) {
  audio.stopSong();
}
//Whether the music is running
bool music_is_running(void) {
  return audio.isRunning();
}
//Gets how long the music player has been playing
long music_get_total_playing_time(void) {
```

```
315     return (long)audio.getTotalPlayingTime() / 1000;
316   }
317   //Obtain the playing time of the music file
318   long music_get_file_duration(void) {
319     return (long)audio.getAudioFileDuration();
320   }
321   //Set play position
322   bool music_set_play_position(int second) {
323     return audio.setAudioPlayPosition((uint16_t)second);
324   }
325   //Gets the current playing time of the music
326   long music_read_play_position(void) {
327     return audio.getAudioCurrentTime();
328   }
329   //Non-blocking music execution function
330   void music_loop(void) {
331     audio.loop();
332   }
333
334   //music player thread
335   void loopTask_music(void *pvParameters) {
336     Serial.println("loopTask_music start...");
337     int temp = 0;
338     while (music_task_flag == 1) {
339       music_loop();
340       int t1 = music_get_total_playing_time();//Gets how long the music player has been playing
341       int t2 = music_get_file_duration();     //Gets the playing time of the music file
342       int t3 = music_read_play_position();    //Gets the current playing time of the music
343       if(temp==1){
344         int t4 = map(t3, 0, t2, 0, 100);
345         lv_bar_set_value(guider_music_ui.music_bar_time, t4, LV_ANIM_OFF);
346         //Serial.printf("t1: %d\t t2: %d\t t3: %d\t t4: %d\r\n", t1, t2, t3, t4);
347       }
348       if ((t1 < t2) && (t2 > 0) && (temp == 0)) { //The music starts to play
349         lv_bar_set_value(guider_music_ui.music_bar_time, 0, LV_ANIM_OFF);
350         temp = 1;
351       } else if ((t2 == 0) && (temp == 1)) {      //The music stop to play
352         temp = 0;
353         music_task_flag = 0;
354         lv_img_set_src(guider_music_ui.music_imgbtn_play, &img_pause);
355         music_button_state = 0;
356         break;
357       }
358     }
```

```
359   music_stop();
360   Serial.println("loopTask_music stop...");
361   vTaskDelete(musicTaskHandle);
362 }
363
364 //Create music task thread
365 void start_music_task(void) {
366   if (music_task_flag == 0) {
367     music_task_flag = 1;
368     xTaskCreate(loopTask_music, "loopTask_music", 8192, NULL, 1, &musicTaskHandle);
369   } else {
370     Serial.println("loopTask_music is running...");
371   }
372 }
373
374 //Close the music thread
375 void stop_music_task(void) {
376   if (music_task_flag == 1) {
377     music_task_flag = 0;
378     while (1) {
379       if (eTaskGetState(musicTaskHandle) == eDeleted) {
380         break;
381       }
382       vTaskDelay(10);
383     }
384     Serial.println("loopTask_music deleted!");
385   }
386 }
387
388 //Check whether the thread is running
389 int music_task_is_running(void) {
390   return music_task_flag;
391 }
392
393 // optional
394 void audio_info(const char *info) {
395   Serial.print("info        ");
396   Serial.println(info);
397 }
398 void audio_eof_mp3(const char *info) {
399   Serial.print("eof_mp3     ");
400   Serial.println(info);
   }
```

Callback functions, which are called when the corresponding screen events are triggered.

```
19  static void music_imgbtn_home_event_handler(lv_event_t *e)
47  static void music_imgbtn_left_event_handler(lv_event_t *e)
84  static void music_imgbtn_right_event_handler(lv_event_t *e)
122 static void music_imgbtn_play_event_handler(lv_event_t *e)
162 static void music_imgbtn_stop_event_handler(lv_event_t *e)
184 static void music_slider_change_event_handler(lv_event_t * e)
```

Music player interface setting function.

```
196 void setup_scr_music(lvgl_music_ui *ui)
```

The display function of the label component. If the text is empty, the prompt information will be printed. If not empty, the content is displayed.

```
275 void music_set_label_text(char *text){
276   if(text!=NULL){
277     lv_label_set_text(guider_music_ui.music_label, text);
278   }
279   else{
280     lv_label_set_text(guider_music_ui.music_label, "The music folder has no files.");
281   }
282 }
```

Some audio driver functions, which can be used to control music playback, pause, playback position view, etc.

```
285 void music_iis_init(void)
289 void music_set_volume(int volume)
293 int music_read_volume(void)
297 void music_load_mp3(char *name)
301 void music_pause_resume(void)
305 void music_stop(void)
309 bool music_is_running(void)
313 long music_get_total_playing_time(void)
317 long music_get_file_duration(void)
321 bool music_set_play_position(int second)
325 long music_read_play_position(void)
```

When the music parameters are properly configured, the audio task execution function should continuously call the music_loop() function to enable uninterrupted music playback. Failure to call this function in a timely manner can lead to issues such as frozen or failed mp3 playback.

```
329 void music_loop(void)
```

Music playback thread. When music_task_flag is 1, the thread content is executed, the music_loop() function is continuously called to play music, and the music playback time is obtained and set on the progress bar. If the music has finished playing, close the music playback thread.

```
333 //music player thread
334 void loopTask_music(void *pvParameters) {
335   Serial.println("loopTask_music start...");
336   int temp = 0;
337   while (music_task_flag == 1) {
```

```
338     music_loop();
339     int t1 = music_get_total_playing_time();//Gets how long the music player has been playing
340     int t2 = music_get_file_duration();     //Gets the playing time of the music file
341     int t3 = music_read_play_position();    //Gets the current playing time of the music
342     if(temp==1){
343       int t4 = map(t3, 0, t2, 0, 100);
344       lv_bar_set_value(guider_music_ui.music_bar_time, t4, LV_ANIM_OFF);
345       //Serial.printf("t1: %d\t t2: %d\t t3: %d\t t4: %d\r\n", t1, t2, t3, t4);
346     }
347     if ((t1 < t2) && (t2 > 0) && (temp == 0)) { //The music starts to play
348       lv_bar_set_value(guider_music_ui.music_bar_time, 0, LV_ANIM_OFF);
349       temp = 1;
350     } else if ((t2 == 0) && (temp == 1)) {      //The music stop to play
351       temp = 0;
352       music_task_flag = 0;
353       lv_img_set_src(guider_music_ui.music_imgbtn_play, &img_pause);
354       music_button_state = 0;
355       break;
356     }
357   }
258   music_stop();
359   Serial.println("loopTask_music stop...");
360   vTaskDelete(musicTaskHandle);
361 }
```

Thread creation function. Determine whether the thread flag music_task_flag has been configured to 1. If the value is 1, the thread already exists and there is no need to create a thread again. If the value is 0, a thread needs to be created.

```
363 //Create music task thread
364 void start_music_task(void) {
365   if (music_task_flag == 0) {
366     music_task_flag = 1;
367     xTaskCreate(loopTask_music, "loopTask_music", 8192, NULL, 1, &musicTaskHandle);
368   } else {
369     Serial.println("loopTask_music is running...");
370   }
371 }
```

Thread shutdown function. Set the thread flag music_task_flag to zero, the thread will end and delete the handle musicTaskHandle. Use the eTaskGetState() function to determine whether the handle musicTaskHandle has been deleted. If it has been deleted, the thread has been closed; Otherwise, wait until the thread is deleted.

```
373 //Close the music thread
374 void stop_music_task(void) {
375   if (music_task_flag == 1) {
376     music_task_flag = 0;
```

```
377     while (1) {
378       if (eTaskGetState(musicTaskHandle) == eDeleted) {
379         break;
380       }
381       vTaskDelay(10);
382     }
383     Serial.println("loopTask_music deleted!");
384   }
385 }
```

If music_task_flag is 1, it means that the thread already exists. If it is 0, it means that the thread has been closed.

```
387 //Check whether the thread is running
388 int music_task_is_running(void) {
389   return music_task_flag;
390 }
```

Every time the music information is loaded, the basic information of the music will be printed out.

```
393 void audio_info(const char *info) {
394   Serial.print("info        ");
395   Serial.println(info);
396 }
```

Every time when playing music ends, print the end prompt message.

```
397 void audio_eof_mp3(const char *info) {
398   Serial.print("eof_mp3     ");
399   Serial.println(info);
400 }
```

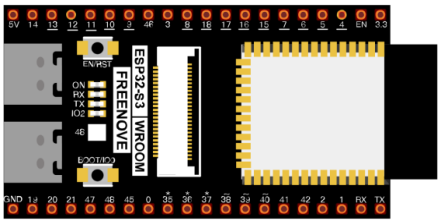
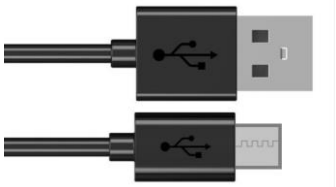
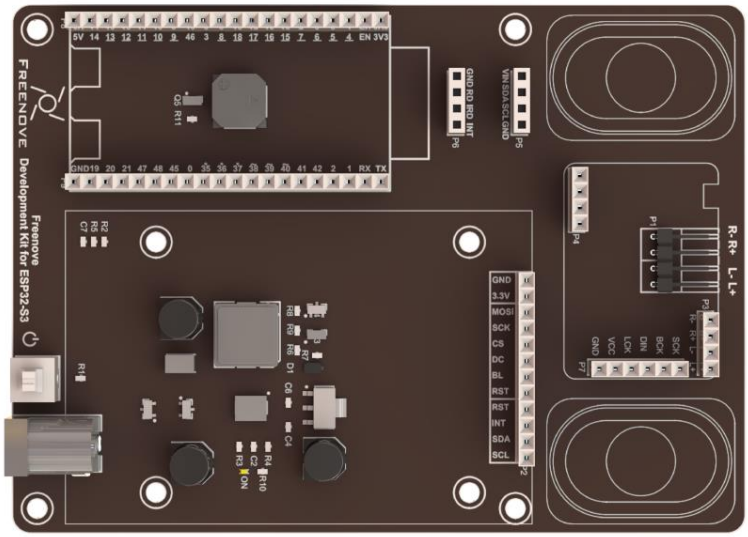
Chapter 18 LVGL Heartrate

In this chapter, we will learn how to create a heartrate monitor.

Project 18.1 LVGL Heartrate

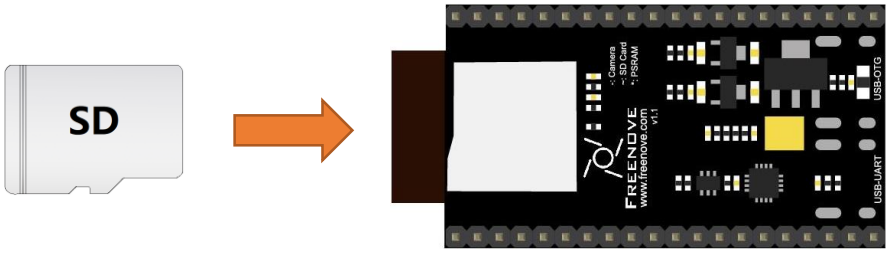
In this project, we will learn how to obtain the raw data and heartrate data of the MAX30102 module and display them on the screen.

Component List

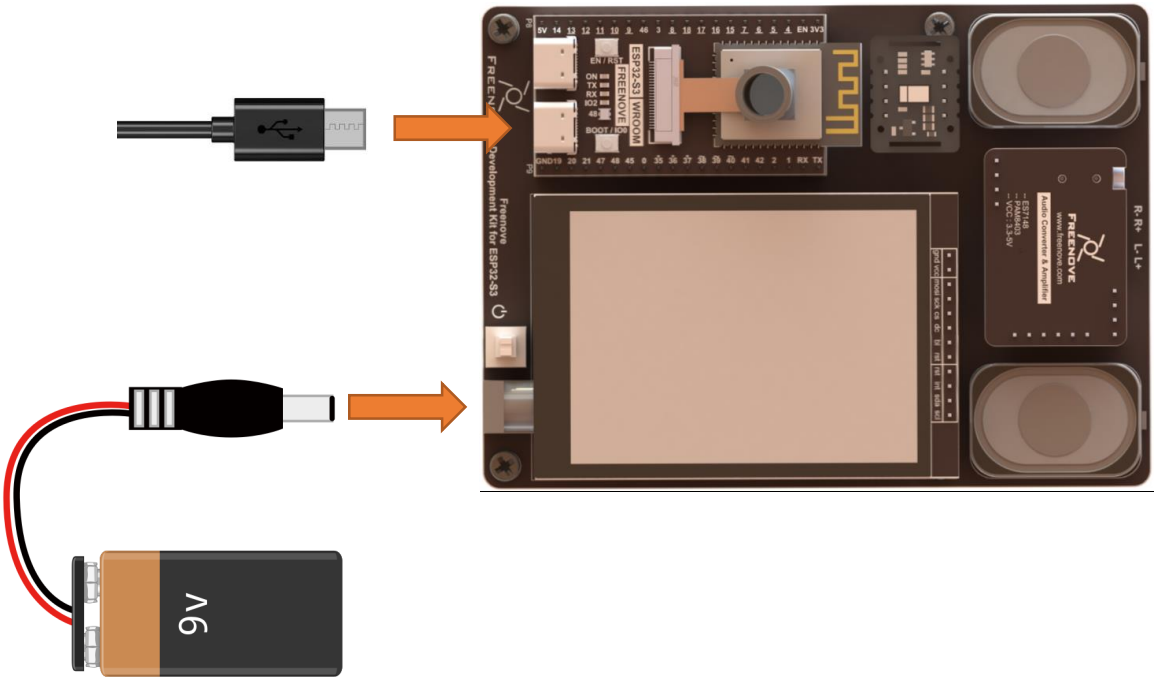
ESP32-S3 WROOM x1 	USB cable x1 	2.8-inch screen 
ESP32-S3 WROOM Shield x1 		Card reader x1 (random color) 
9V battery cable x1 	9V battery x1 (Not included, prepared by yourself) 	SD card x1 

Circuit

If you have not yet used the SD card, please refer to Chapter 4. Click here to navigate back to Chapter 4.
Before connecting the USB cable, insert the SD card into the SD card slot on the back of the ESP32-S3.

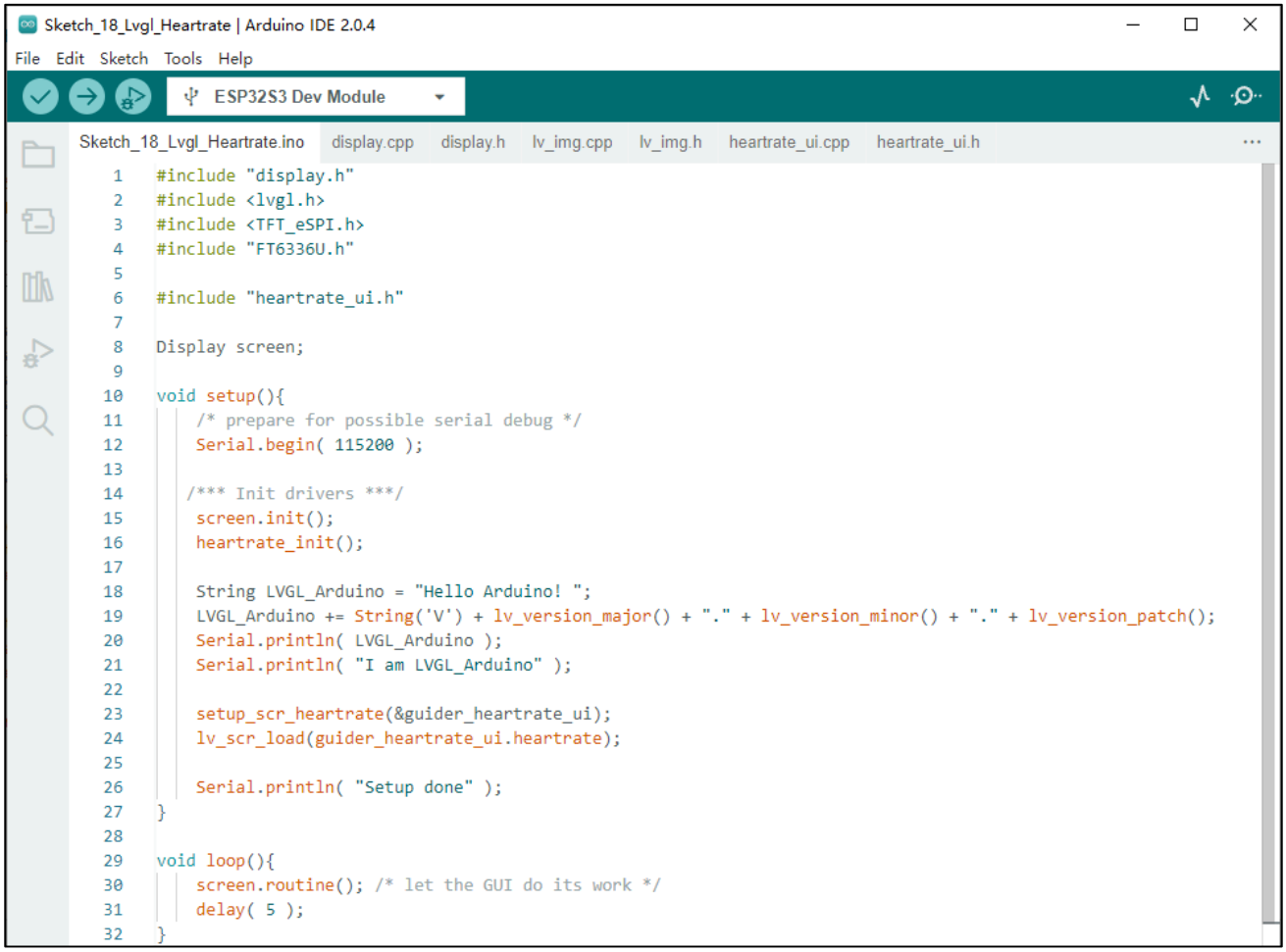


Connect Freenove ESP32-S3 to the computer using the USB cable.



Sketch

Sketch_18_LVGL_Heartrate



The following is the program code:

```
#include "display.h"
#include <lvgl.h>
#include <TFT_eSPI.h>
#include "FT6336U.h"
#include "heartrate_ui.h"

Display screen;

void setup(){
    /* prepare for possible serial debug */
    Serial.begin( 115200 );

   /*** Init drivers ***/
    screen.init();
    heartrate_init();
```

```
16
17      String LVGL_Arduino = "Hello Arduino! ";
18      LVGL_Arduino += String('V') + lv_version_major() + "." + lv_version_minor() + "." +
    lv_version_patch();
19      Serial.println( LVGL_Arduino );
20      Serial.println( "I am LVGL_Arduino" );
21
22      setup_scr_heartrate(&guider_heartrate_ui);
23      lv_scr_load(guider_heartrate_ui.heartrate);
24
25      Serial.println( "Setup done" );
26  }
27
28  void loop(){
29      screen.routine(); /* let the GUI do its work */
30      delay( 5 );
31  }
```

Configure the heart rate monitor interface and load this interface.

```
22      setup_scr_heartrate(&guider_heartrate_ui);
23      lv_scr_load(guider_heartrate_ui.heartrate);
```

heartrate_ui.h

Declare the functions so that they can be called in the ino file.

```
1   #ifndef __HEARTRATE_UI_H
2   #define __HEARTRATE_UI_H
3
4   #include "Arduino.h"
5   #include "lvgl.h"
6
7   #define I2C_SDA 2
8   #define I2C_SCL 1
9
10  #define HEARTRATE_SERIAL 0      //Serial print the original ir value, average value and dynamic
11  data.
12  #define CHART_LOW_LIMIT 0       //Lower limit of Chart
13  #define CHART_HIGH_LIMIT 2000   //Upper limit of Chart
14
15  typedef struct lvgl_heartrate
16  {
17    lv_obj_t *heartrate;
18    lv_obj_t *heartrate_chart;
19    lv_obj_t *heartrate_label_heart_rate;
20    lv_obj_t *heartrate_home;
21  }lvgl_heartrate_ui;
22
```

```
23  extern lvgl_heartrate_ui guider_heartrate_ui;     //music ui structure
24
25  int heartrate_init(void);         //Initialization heartrate module
26  void heartrate_shutdown(void);    //The heartrate module is enabled to enter low power mode
27  void heartrate_wake_up(void);     //Wake up the heartrate module
28
29  void loopTask_heartrate(void *pvParameters);     //heartrate task thread
30  void create_heartrate_task(void);                //Create heartrate task thread
31  void stop_heartrate_task(void);                  //Close the heartrate thread
32
33  void setup_scr_heartrate(lvgl_heartrate_ui *ui);  //Parameter configuration function on the
    heartrate screen
34
35  #endif
```

heartrate_ui.cpp

```
1   #include "heartrate_ui.h"
2   #include "lv_img.h"
3   #include <Wire.h>
4   #include "MAX30105.h"
5   #include "heartRate.h"
6
7   MAX30105 heartrate;                    //apply a module object
8   static lv_chart_series_t *series;      //apply an lvgl chart variable
9   lvgl_heartrate_ui guider_heartrate_ui;//heartrate ui structure
10  int heartrate_task_flag = 0;           //heartrate thread running flag
11  TaskHandle_t heartrateTaskHandle;      //heartrate thread task handle
12
13  const byte RATE_SIZE = 4;             //Increase this for more averaging. 4 is good.
14  byte rates[RATE_SIZE];                //Array of heart rates
15  byte rateSpot = 0;                    //Record the number of bits in the heart rate array
16  long lastBeat = 0;                    //Time at which the last beat occurred
17  float beatsPerMinute;                 //store the heartbeat value per minute
18  int beatAvg = 0;                      //Heart rate average
19  int beatAvgLast = 0;                  //Average heart rate from last time
20  byte irvalueSpot = 0;                 //Record the number of bits in the heartrate irvalue array
21  const byte AVERAGE_NUM = 10;          //Increase this for more averaging. 10 is good.
22  long heartrate_irvalue[AVERAGE_NUM]; //Array of heartrate irvalue
23  long show_value = 0;           //The data values of the points in the heart rate line plot
24
25  //Initialization heartrate module
26  int heartrate_init(void) {
27    if (!heartrate.begin(Wire, I2C_SPEED_FAST)) {
28      Serial.println("MAX30105 was not found. Please check wiring/power. ");
29      return -1;
```

```
  }
  heartrate.setup();  //Configure sensor with these settings
  heartrate_shutdown();
}

//The heartrate module is enabled to enter low power mode
void heartrate_shutdown(void) {
  heartrate.shutDown();
}

//Wake up the heartrate module
void heartrate_wake_up(void) {
  heartrate.wakeUp();
}

//Calculate the average value of the array
long heartrate_average(long array[], int num) {
  long average = 0;
  for (int i = 0; i < num; i++) {
    average += array[i];
  }
  average = average / num;
  return average;
}

//heartrate task thread
void loopTask_heartrate(void *pvParameters) {
  Serial.println("loopTask_heartrate start...");

  while (heartrate_task_flag) {
    long irValue = heartrate.getIR();  //Get the raw data
    if (irValue < 50000) {
      //heartrate.setup();
      delay(100);
    }
    else{
      if (checkForBeat(irValue) == true) {
        //We sensed a beat!
        long delta = millis() - lastBeat;
        lastBeat = millis();
        beatsPerMinute = 60.0 / (delta / 1000.0);

        if (beatsPerMinute < 255 && beatsPerMinute > 50) {
          rates[rateSpot++] = (byte)beatsPerMinute;  //Store this reading in the array
```

```
74            rateSpot %= RATE_SIZE;                    //Wrap variable
75            //Take average of readings
76            beatAvg = 0;
77            for (byte x = 0; x < RATE_SIZE; x++)
78              beatAvg += rates[x];
79            beatAvg /= RATE_SIZE;
80          }
81        }
82        heartrate_irvalue[irvalueSpot++] = irValue;  //Put the raw data into an array
83        irvalueSpot %= AVERAGE_NUM;
84        long average = heartrate_average(heartrate_irvalue, AVERAGE_NUM);
85        show_value = irValue - average + (CHART_HIGH_LIMIT / 2);
86
87        #if HEARTRATE_SERIAL
88          Serial.printf("%ld %ld %ld\r\n", irValue, average, show_value);
89        #endif
90
91        if (irValue > 50000 && show_value > CHART_LOW_LIMIT && show_value < CHART_HIGH_LIMIT)
92          lv_chart_set_next_value(guider_heartrate_ui.heartrate_chart, series, show_value);
93        if (beatAvg != beatAvgLast) {
94          beatAvgLast = beatAvg;
95          lv_label_set_text_fmt(guider_heartrate_ui.heartrate_label_heart_rate, "HeartRate: %d",
    beatAvg);
96        }
97      }
98    }
99    heartrate_shutdown();
100   vTaskDelete(heartrateTaskHandle);
101 }
102
103 //Create heartrate task thread
104 void create_heartrate_task(void) {
105   if (heartrate_task_flag == 0) {
106     heartrate_task_flag = 1;
107     xTaskCreate(loopTask_heartrate, "loopTask_heartrate", 8192, NULL, 1,
    &heartrateTaskHandle);
108   } else {
109     Serial.println("loopTask_heartrate is running...");
110   }
111 }
112
113 //Close the heartrate thread
114 void stop_heartrate_task(void) {
115   if (heartrate_task_flag == 1) {
```

```
    heartrate_task_flag = 0;
    while (1) {
      if (eTaskGetState(heartrateTaskHandle) == eDeleted) {
        break;
      }
      vTaskDelay(10);
    }
    Serial.println("loopTask_heartrate deleted!");
  }
}

//Click the home icon, callback function: goes to the main ui interface
static void heartrate_home_event_handler(lv_event_t *e) {
  lv_event_code_t code = lv_event_get_code(e);
  switch (code) {
    case LV_EVENT_CLICKED:
      {
        Serial.println("Clicked the home button.");
      }
      break;
    case LV_EVENT_RELEASED:
      {
        /*
        stop_heartrate_task();
        delay(100);
        if (!lv_obj_is_valid(guider_main_ui.main))
          setup_scr_main(&guider_main_ui);
        lv_disp_t *d = lv_obj_get_disp(lv_scr_act());
        if (d->prev_scr == NULL && d->scr_to_load == NULL)
          lv_scr_load(guider_main_ui.main);
        lv_obj_del(guider_heartrate_ui.heartrate);
        */
      }
      break;
    default:
      break;
  }
}

//Parameter configuration function on the heartrate screen
void setup_scr_heartrate(lvgl_heartrate_ui *ui) {
  ui->heartrate = lv_obj_create(NULL);
  lv_img_home_init();

```

```
160   /*Init the pressed style*/
161   static lv_style_t style_pr;//Apply for a style
162   lv_style_init(&style_pr);  //Initialize it
163   lv_style_set_translate_y(&style_pr, 5);//Style: Every time you trigger, move down 5 pixels
164
165   /*Create a chart1*/
166   ui->heartrate_chart = lv_chart_create(ui->heartrate);
167
168   lv_obj_set_size(ui->heartrate_chart, 180, 200);
169   lv_obj_set_pos(ui->heartrate_chart, 60, 120);
170   lv_chart_set_type(ui->heartrate_chart, LV_CHART_TYPE_LINE); /*Show lines and points too*/
171   lv_chart_set_div_line_count(ui->heartrate_chart, 5, 10);          //Set chart to 5 rows and
      10 columns
172   lv_chart_set_point_count(ui->heartrate_chart, 100);               //chart shows the data for
      100 points
173   lv_obj_set_style_size(ui->heartrate_chart, 0, LV_PART_INDICATOR); //Make each data point
      unsalient
174
175   lv_chart_set_update_mode(ui->heartrate_chart, LV_CHART_UPDATE_MODE_SHIFT);//Move chart to
      update data
176   /*Add two data series*/
177   series = lv_chart_add_series(ui->heartrate_chart, lv_palette_main(LV_PALETTE_BLUE),
      LV_CHART_AXIS_SECONDARY_Y);//Let the data be on the right y axis
178   lv_chart_set_range(ui->heartrate_chart, LV_CHART_AXIS_SECONDARY_Y, CHART_LOW_LIMIT,
      CHART_HIGH_LIMIT);         //Set the value range of right y axis
179
180   lv_chart_set_axis_tick(ui->heartrate_chart, LV_CHART_AXIS_PRIMARY_Y, 3, 1, 11, 2, true, 60);
      //Let the axis be on the left y axis
181   lv_chart_set_range(ui->heartrate_chart, LV_CHART_AXIS_PRIMARY_Y, CHART_LOW_LIMIT,
      CHART_HIGH_LIMIT);           //Set the value range of left y axis
182
183   ui->heartrate_label_heart_rate = lv_label_create(ui->heartrate);
184   lv_obj_set_pos(ui->heartrate_label_heart_rate, 0, 50);
185   lv_obj_set_size(ui->heartrate_label_heart_rate, 140, 40);
186   lv_label_set_text(ui->heartrate_label_heart_rate, "HeartRate:0");
187   lv_label_set_long_mode(ui->heartrate_label_heart_rate, LV_LABEL_LONG_WRAP);
188   lv_obj_set_style_text_align(ui->heartrate_label_heart_rate, LV_TEXT_ALIGN_CENTER, 0);
189
190   ui->heartrate_home = lv_imgbtn_create(ui->heartrate);
191   lv_obj_set_pos(ui->heartrate_home, 150, 20);
192   lv_obj_set_size(ui->heartrate_home, 80, 80);
193   lv_img_set_src(ui->heartrate_home, &img_home);
194   lv_obj_add_style(ui->heartrate_home, &style_pr, LV_STATE_PRESSED);//Triggered when the
      button is pressed
```

```
195
196     lv_obj_add_event_cb(ui->heartrate_home, heartrate_home_event_handler, LV_EVENT_ALL, NULL);
197     create_heartrate_task();
198     heartrate_wake_up();
199 }
```

Initialize the heart rate module and configure it to sleep mode.

```
25  //Initialization heartrate module
26  int heartrate_init(void) {
27    if (!heartrate.begin(Wire, I2C_SPEED_FAST)) {
28      Serial.println("MAX30105 was not found. Please check wiring/power. ");
29      return -1;
30    }
31    heartrate.setup();  //Configure sensor with these settings
32    heartrate_shutdown();
33  }
```

Heartrate module sleep function and module wake-up function.

```
35  //The heartrate module is enabled to enter low power mode
36  void heartrate_shutdown(void) {
37    heartrate.shutDown();
38  }
39
40  //Wake up the heartrate module
41  void heartrate_wake_up(void) {
42    heartrate.wakeUp();
43  }
```

Average the heartrate values obtained multiple times.

```
45  //Calculate the average value of the array
46  long heartrate_average(long array[], int num) {
47    long average = 0;
48    for (int i = 0; i < num; i++) {
49      average += array[i];
50    }
51    average = average / num;
52    return average;
53  }
```

Acquire raw infrared data.

```
60  long irValue = heartrate.getIR();  //Get the raw data
```

Heartrate checking function. Returns true if a mentality spike is detected, otherwise returns false.

```
66  checkForBeat(irValue)
```

If the HEARTRATE_SERIAL macro definition is set to 1, the original data will be printed to the serial port, allowing you to view the heart rate waveform using a serial port drawing function. By default, serial port data is not printed.

```
87  #if HEARTRATE_SERIAL
88          Serial.printf("%ld %ld %ld\r\n", irValue, average, show_value);
89  #endif
```

Label component setup function, which can be used similarly like printf().

```
95  lv_label_set_text_fmt(guider_heartrate_ui.heartrate_label_heart_rate, "HeartRate: %d",
    beatAvg);
```

Create a line chart with 100 data points to make the waveform more closely resemble real data. Set the chart to scroll continuously for optimal viewing.

```
165   /*Create a chart1*/
166   ui->heartrate_chart = lv_chart_create(ui->heartrate);
167
168   lv_obj_set_size(ui->heartrate_chart, 180, 200);
169   lv_obj_set_pos(ui->heartrate_chart, 60, 120);
170   lv_chart_set_type(ui->heartrate_chart, LV_CHART_TYPE_LINE); /*Show lines and points too*/
171   lv_chart_set_div_line_count(ui->heartrate_chart, 5, 10);//Set chart to 5 rows and 10 columns
172   lv_chart_set_point_count(ui->heartrate_chart, 100);    //chart shows the data for 100 points
173   lv_obj_set_style_size(ui->heartrate_chart, 0, LV_PART_INDICATOR); //Make each data point
      unsalient
174
175   lv_chart_set_update_mode(ui->heartrate_chart, LV_CHART_UPDATE_MODE_SHIFT);//Move chart to
      update data
176   /*Add two data series*/
177   series = lv_chart_add_series(ui->heartrate_chart, lv_palette_main(LV_PALETTE_BLUE),
      LV_CHART_AXIS_SECONDARY_Y);//Let the data be on the right y axis
178   lv_chart_set_range(ui->heartrate_chart, LV_CHART_AXIS_SECONDARY_Y, CHART_LOW_LIMIT,
      CHART_HIGH_LIMIT);         //Set the value range of right y axis
179
180   lv_chart_set_axis_tick(ui->heartrate_chart, LV_CHART_AXIS_PRIMARY_Y, 3, 1, 11, 2, true, 60);
      //Let the axis be on the left y axis
181   lv_chart_set_range(ui->heartrate_chart, LV_CHART_AXIS_PRIMARY_Y, CHART_LOW_LIMIT,
      CHART_HIGH_LIMIT);           //Set the value range of left y axis
```

Display heart rate data on a line chart.

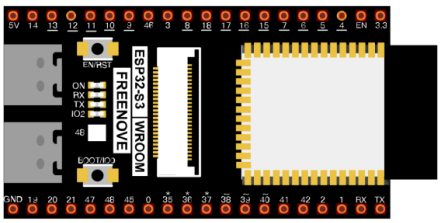
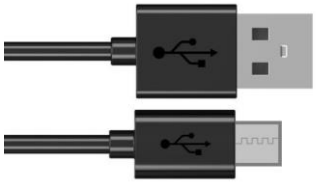
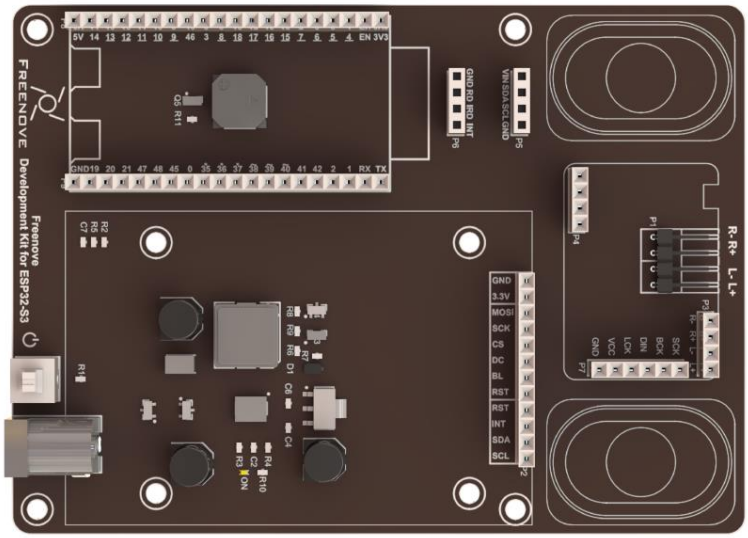
```
92  lv_chart_set_next_value(guider_heartrate_ui.heartrate_chart, series, show_value);
```

Chapter 19 LVGL Multifunctionality

In this chapter, we pull together several of the previous examples as a comprehensive routine.

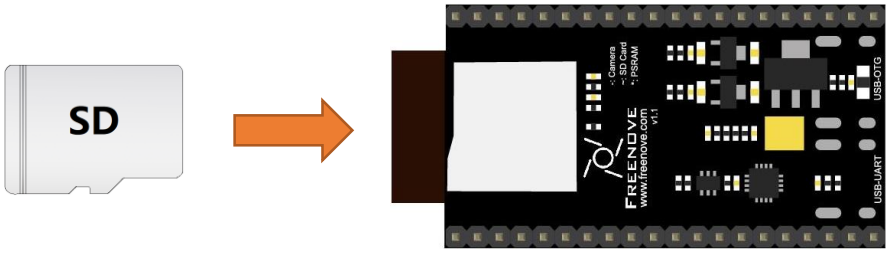
Project 19.1 LVGL Multifunctionality

Component List

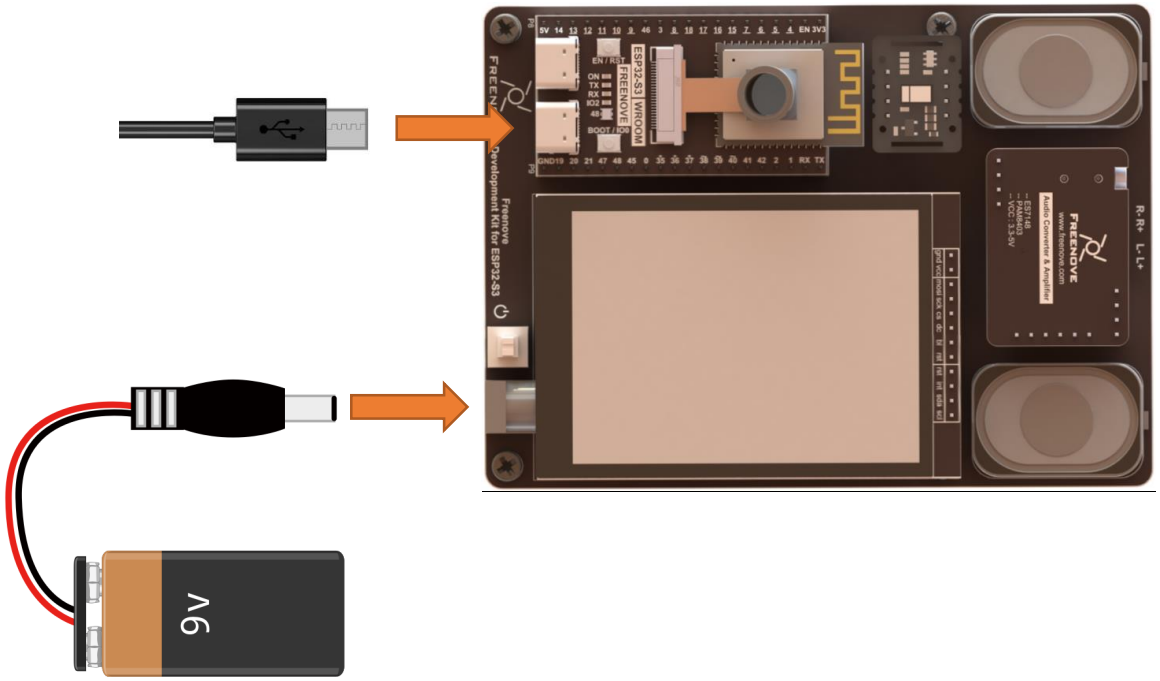
ESP32-S3 WROOM x1 	USB cable x1 	2.8-inch screen 
ESP32-S3 WROOM Shield x1 		Card reader x1 (random color) 
9V battery cable x1 	9V battery x1 (Not included, prepared by yourself) 	SD card x1 

Circuit

If you have not yet used the SD card, please refer to Chapter 4. Click here to navigate back to Chapter 4.
Before connecting the USB cable, insert the SD card into the SD card slot on the back of the ESP32-S3.

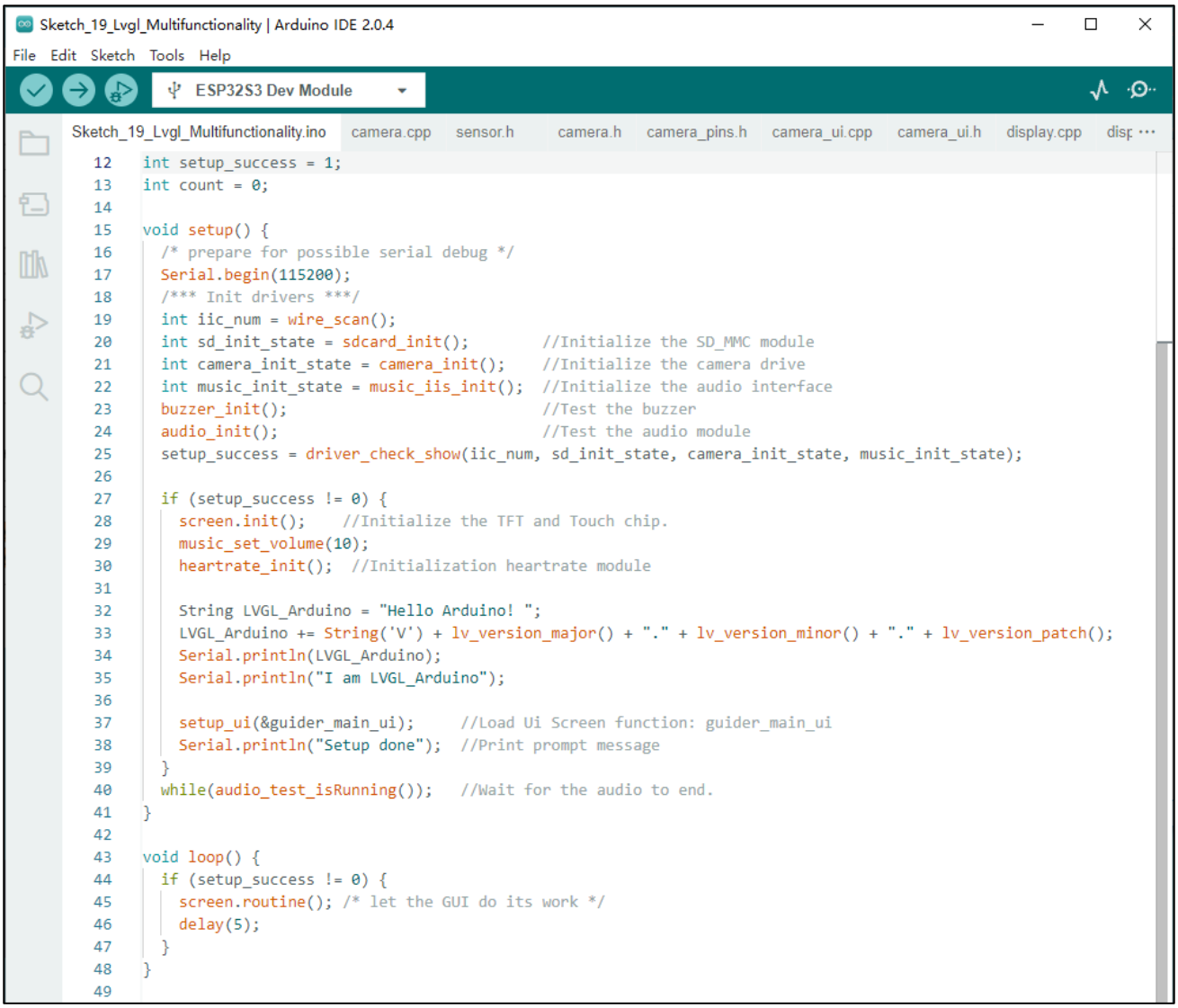


Connect Freenove ESP32-S3 to the computer using the USB cable.



Sketch

Sketch_19_LVGL_Multifunctionality



The following is the program code:

```
#include <Arduino.h>
#include <lvgl.h>
#include <TFT_eSPI.h>
#include "display.h"
#include "FT6336U.h"
#include "sd_card.h"
#include "camera.h"
#include "main_ui.h"
#include "driver_test.h"

Display screen;
int setup_success = 1;
```

```
13
14  void setup() {
15    /* prepare for possible serial debug */
16    Serial.begin(115200);
17    /*** Init drivers ***/
18    int iic_num = wire_scan();
19    int sd_init_state = sdcard_init();         //Initialize the SD_MMC module
20    int camera_init_state = camera_init();     //Initialize the camera drive
21    int music_init_state = music_iis_init();  //Initialize the audio interface
22    buzzer_init();                             //Test the buzzer
23    audio_init();                              //Test the audio module
24    setup_success = driver_check_show(iic_num, sd_init_state, camera_init_state,
      music_init_state);
25    if (setup_success != 0) {
26      screen.init();    //Initialize the TFT and Touch chip.
27      music_set_volume(10);
28      heartrate_init();  //Initialization heartrate module
29
30      String LVGL_Arduino = "Hello Arduino! ";
31      LVGL_Arduino += String('V') + lv_version_major() + "." + lv_version_minor() + "." +
      lv_version_patch();
32      Serial.println(LVGL_Arduino);
33      Serial.println("I am LVGL_Arduino");
34
35      setup_ui(&guider_main_ui);     //Load Ui Screen function: guider_main_ui
36      Serial.println("Setup done");  //Print prompt message
37    }
38    while(audio_test_isRunning());   //Wait for the audio to end.
39  }
40
41  void loop() {
42    if (setup_success != 0) {
43      screen.routine(); /* let the GUI do its work */
44      delay(5);
45    }
46  }
```

Scan for I2C devices and initialize the SD card, camera, and I2S interface. Emit 3 beeps from the buzzer and play a test music on the speaker.

```
18    int iic_num = wire_scan();
19    int sd_init_state = sdcard_init();         //Initialize the SD_MMC module
20    int camera_init_state = camera_init();     //Initialize the camera drive
21    int music_init_state = music_iis_init();  //Initialize the audio interface
22    buzzer_init();                             //Test the buzzer
23    audio_init();                              //Test the audio module
```

Print the initialization information on the screen and return the setup_success value. If the value is 1, it indicates that the device has passed the test. If the value is 0, it means that some device initialization has failed.

```
24    setup_success = driver_check_show(iic_num, sd_init_state, camera_init_state,
      music_init_state);
```

If all hardware initialization succeeds, load the lvgl system, set the music volume, initialize the heart rate module, print a prompt message, and then enter the main control interface.

```
25    if (setup_success != 0) {
26      screen.init();    //Initialize the TFT and Touch chip.
27      music_set_volume(10);
28      heartrate_init();  //Initialization heartrate module
29
30      String LVGL_Arduino = "Hello Arduino! ";
31      LVGL_Arduino += String('V') + lv_version_major() + "." + lv_version_minor() + "." +
      lv_version_patch();
32      Serial.println(LVGL_Arduino);
33      Serial.println("I am LVGL_Arduino");
34
35      setup_ui(&guider_main_ui);      //Load Ui Screen function: guider_main_ui
36      Serial.println("Setup done");  //Print prompt message
37    }
38    while(audio_test_isRunning());    //Wait for the audio to end.
```

main_ui.h

Declare the functions so that they can be called in the ino file.

```
1   #ifndef __MAIN_UI_H
2   #define __MAIN_UI_H
3
4   #include "lvgl.h"
5   #include "Arduino.h"
6
7   #include "camera_ui.h"
8   #include "picture_ui.h"
9   #include "music_ui.h"
10  #include "heartrate_ui.h"
11
12  typedef struct lvgl_main
13  {
14    lv_obj_t *main;
15    lv_obj_t *main_imgbtn_logo;
16    lv_obj_t *main_imgbtn_camera;
17    lv_obj_t *main_imgbtn_picture;
18    lv_obj_t *main_imgbtn_music;
19    lv_obj_t *main_imgbtn_heartrate;
20  }lvgl_main_ui;//main ui struct
```

```
21
22  extern lvgl_main_ui guider_main_ui;   //main ui structure
23  void setup_scr_main(lvgl_main_ui *ui);//Parameter configuration function on the main screen
24  void setup_ui(lvgl_main_ui *ui);      //Load Ui Screen function
25
26  #endif
```

main_ui.cpp

```
1   #include "main_ui.h"
2   #include "lv_img.h"
3
4   lvgl_main_ui guider_main_ui;//main ui structure
5
6   //Click the main interface logo, callback function: do nothing
7   static void main_imgbtn_logo_event_handler(lv_event_t *e) {
8     lv_event_code_t code = lv_event_get_code(e);
9
10    if (code == LV_EVENT_CLICKED) {
11      Serial.println("Clicked the logo button.");
12    }
13  }
14
15  //Click the camera icon, callback function: goes to the camera ui interface
16  static void main_imgbtn_camera_event_handler(lv_event_t *e) {
17    lv_event_code_t code = lv_event_get_code(e);
18    switch (code) {
19      case LV_EVENT_CLICKED:
20        {
21          Serial.println("Clicked the camera button.");
22        }
23        break;
24      case LV_EVENT_RELEASED:
25        {
26          if (!lv_obj_is_valid(guider_camera_ui.camera))
27            setup_scr_camera(&guider_camera_ui);
28          lv_disp_t *d = lv_obj_get_disp(lv_scr_act());
29          if (d->prev_scr == NULL && d->scr_to_load == NULL) {
30            lv_scr_load(guider_camera_ui.camera);
31            lv_obj_del(guider_main_ui.main);
32          }
33        }
34        break;
35      default:
36        break;
37    }
```

```
}

//Click the picture icon, callback function: goes to the picture ui interface
static void main_imgbtn_picture_event_handler(lv_event_t *e) {
  lv_event_code_t code = lv_event_get_code(e);
  switch (code) {
    case LV_EVENT_CLICKED:
      {
        Serial.println("Clicked the picture button.");
      }
      break;
    case LV_EVENT_RELEASED:
      {
        if (!lv_obj_is_valid(guider_picture_ui.picture))
          setup_scr_picture(&guider_picture_ui);
        lv_disp_t *d = lv_obj_get_disp(lv_scr_act());
        if (d->prev_scr == NULL && d->scr_to_load == NULL) {
          lv_scr_load(guider_picture_ui.picture);
          lv_obj_del(guider_main_ui.main);
        }
      }
      break;
    default:
      break;
  }
}

//Click the music icon, callback function: goes to the music ui interface
static void main_imgbtn_music_event_handler(lv_event_t *e) {
  lv_event_code_t code = lv_event_get_code(e);
  switch (code) {
    case LV_EVENT_CLICKED:
      {
        Serial.println("Clicked the music button.");
      }
      break;
    case LV_EVENT_RELEASED:
      {
        if (!lv_obj_is_valid(guider_music_ui.music))
          setup_scr_music(&guider_music_ui);
        lv_disp_t *d = lv_obj_get_disp(lv_scr_act());
        if (d->prev_scr == NULL && d->scr_to_load == NULL) {
          lv_scr_load(guider_music_ui.music);
          lv_obj_del(guider_main_ui.main);
```

```
        }
      }
      break;
    default:
      break;
  }
}

//Click the heartrate icon, callback function: goes to the heartrate ui interface
static void main_imgbtn_heartrate_event_handler(lv_event_t *e) {
  lv_event_code_t code = lv_event_get_code(e);

  switch (code) {
    case LV_EVENT_CLICKED:
      {
        Serial.println("Clicked the heartrate button.");
      }
      break;
    case LV_EVENT_RELEASED:
      {
        if (!lv_obj_is_valid(guider_heartrate_ui.heartrate))
          setup_scr_heartrate(&guider_heartrate_ui);
        lv_disp_t *d = lv_obj_get_disp(lv_scr_act());
        if (d->prev_scr == NULL && d->scr_to_load == NULL) {
          lv_scr_load(guider_heartrate_ui.heartrate);
          lv_obj_del(guider_main_ui.main);
        }
      }
      break;
    default:
      break;
  }
}

//Parameter configuration function on the main screen
void setup_scr_main(lvgl_main_ui *ui) {
  //Write codes main
  ui->main = lv_obj_create(NULL);

  static lv_style_t bg_style;
  lv_style_init(&bg_style);
  lv_style_set_bg_color(&bg_style, lv_color_hex(0xffffff));
  lv_obj_add_style(ui->main, &bg_style, LV_PART_MAIN);

```

```
126     lv_img_freenove_init();
127     lv_img_camera_init();
128     lv_img_music_init();
129     lv_img_heartrate_init();
130     lv_img_picture_init();
131
132     /*Init the pressed style*/
133     static lv_style_t style_pr;//Apply for a style
134     lv_style_init(&style_pr);  //Initialize it
135     lv_style_set_translate_y(&style_pr, 5);//Style: Every time you trigger, move down 5 pixels
136
137     //Write codes main_imgbtn_logo
138     ui->main_imgbtn_logo = lv_imgbtn_create(ui->main);
139     lv_obj_set_pos(ui->main_imgbtn_logo, 40, 20);
140     lv_obj_set_size(ui->main_imgbtn_logo, 160, 90);
141     lv_img_set_src(ui->main_imgbtn_logo, &img_freenove);
142     lv_obj_add_style(ui->main_imgbtn_logo, &style_pr, LV_STATE_PRESSED);//Triggered when the
    button is pressed
143
144     //Write codes main_imgbtn_camera
145     ui->main_imgbtn_camera = lv_imgbtn_create(ui->main);
146     lv_obj_set_pos(ui->main_imgbtn_camera, 20, 135);
147     lv_obj_set_size(ui->main_imgbtn_camera, 80, 80);
148     lv_img_set_src(ui->main_imgbtn_camera, &img_camera);
149     lv_obj_add_style(ui->main_imgbtn_camera, &style_pr, LV_STATE_PRESSED);//Triggered when the
    button is pressed
150
151     //Write codes main_imgbtn_picture
152     ui->main_imgbtn_picture = lv_imgbtn_create(ui->main);
153     lv_obj_set_pos(ui->main_imgbtn_picture, 140, 135);
154     lv_obj_set_size(ui->main_imgbtn_picture, 80, 80);
155     lv_img_set_src(ui->main_imgbtn_picture, &img_picture);
156     lv_obj_add_style(ui->main_imgbtn_picture, &style_pr, LV_STATE_PRESSED);//Triggered when the
    button is pressed
157
158     //Write codes main_imgbtn_music
159     ui->main_imgbtn_music = lv_imgbtn_create(ui->main);
160     lv_obj_set_pos(ui->main_imgbtn_music, 20, 225);
161     lv_obj_set_size(ui->main_imgbtn_music, 80, 80);
162     lv_img_set_src(ui->main_imgbtn_music, &img_music);
163     lv_obj_add_style(ui->main_imgbtn_music, &style_pr, LV_STATE_PRESSED);//Triggered when the
    button is pressed
164
165     //Write codes main_imgbtn_heartrate
```

```
166   ui->main_imgbtn_heartrate = lv_imgbtn_create(ui->main);
167   lv_obj_set_pos(ui->main_imgbtn_heartrate, 140, 225);
168   lv_obj_set_size(ui->main_imgbtn_heartrate, 80, 80);
169   lv_img_set_src(ui->main_imgbtn_heartrate, &img_heartrate);
170   lv_obj_add_style(ui->main_imgbtn_heartrate, &style_pr, LV_STATE_PRESSED);
171
172   lv_obj_add_event_cb(ui->main_imgbtn_logo, main_imgbtn_logo_event_handler, LV_EVENT_ALL,
      NULL);
173   lv_obj_add_event_cb(ui->main_imgbtn_camera, main_imgbtn_camera_event_handler, LV_EVENT_ALL,
      NULL);
174   lv_obj_add_event_cb(ui->main_imgbtn_picture, main_imgbtn_picture_event_handler,
      LV_EVENT_ALL, NULL);
175   lv_obj_add_event_cb(ui->main_imgbtn_music, main_imgbtn_music_event_handler, LV_EVENT_ALL,
      NULL);
176   lv_obj_add_event_cb(ui->main_imgbtn_heartrate, main_imgbtn_heartrate_event_handler,
      LV_EVENT_ALL, NULL);
177 }
178
179 //Load Ui Screen function
180 void setup_ui(lvgl_main_ui *ui) {
181   setup_scr_main(ui);
182   lv_scr_load(ui->main);
183 }
```

The callback function for the main interface image buttons, triggered when the corresponding button is clicked, causing the interface to switch to the sub-function interface.

```
7   static void main_imgbtn_logo_event_handler(lv_event_t *e)
16  static void main_imgbtn_camera_event_handler(lv_event_t *e)
41  static void main_imgbtn_picture_event_handler(lv_event_t *e)
66  static void main_imgbtn_music_event_handler(lv_event_t *e)
91  static void main_imgbtn_heartrate_event_handler(lv_event_t *e)
```

Create an interface component and set the background color of the component.

```
118   //Write codes main
119   ui->main = lv_obj_create(NULL);
120
121   static lv_style_t bg_style;
122   lv_style_init(&bg_style);
123   lv_style_set_bg_color(&bg_style, lv_color_hex(0xffffff));
124   lv_obj_add_style(ui->main, &bg_style, LV_PART_MAIN);
```

Initialize image data.

```
126   lv_img_freenove_init();
127   lv_img_camera_init();
128   lv_img_music_init();
129   lv_img_heartrate_init();
130   lv_img_picture_init();
```

Define a component style. When the component is triggered, the component moves down by 5 pixels.

```
132   /*Init the pressed style*/
133   static lv_style_t style_pr;//Apply for a style
134   lv_style_init(&style_pr);  //Initialize it
135   lv_style_set_translate_y(&style_pr, 5);//Style: Every time you trigger, move down 5 pixels
```

Set the position and size of the main_imgbtn_logo image button, and specify the image to be displayed. Apply a component style that moves the button down by 5 pixels when it is pressed.

```
137   //Write codes main_imgbtn_logo
138   ui->main_imgbtn_logo = lv_imgbtn_create(ui->main);
139   lv_obj_set_pos(ui->main_imgbtn_logo, 40, 20);
140   lv_obj_set_size(ui->main_imgbtn_logo, 160, 90);
141   lv_img_set_src(ui->main_imgbtn_logo, &img_freenove);
142   lv_obj_add_style(ui->main_imgbtn_logo, &style_pr, LV_STATE_PRESSED);
```

Set the callback function when the image button main_imgbtn_logo is pressed.

```
172   lv_obj_add_event_cb(ui->main_imgbtn_logo, main_imgbtn_logo_event_handler, LV_EVENT_ALL,
      NULL);
```

Configure and load the main interface.

```
179 //Load Ui Screen function
180 void setup_ui(lvgl_main_ui *ui) {
181   setup_scr_main(ui);
182   lv_scr_load(ui->main);
183 }
```

What's next?

Thanks for reading! This tutorial has come to an end. If you notice any mistakes, omissions, or have any other questions or ideas related to the contents of this tutorial or the kit, please do not hesitate to contact us.

support@freenove.com

We will check and correct it as soon as possible.

If you want learn more about ESP32-S3, you view our ultimate tutorial:
https://github.com/Freenove/Freenove_Ultimate_Starter_Kit_for_ESP32_S3/archive/master.zip

If you want to learn more about Arduino, Raspberry Pi, smart cars, robots and other interesting products in science and technology, please continue to focus on our website. We will continue to launch cost-effective, innovative and exciting products.

http://www.freenove.com/

End of the Tutorial

Thank you again for choosing Freenove products.