



TSCIN**BUNY**

ZYE0011-ESP32S3

## Tutorial introduction

This product kit supports UNO or ESP32 as the main control. The tutorial is divided into two parts according to the main control. This tutorial is for ESP32-S3 and is divided into 21 sections.

Starting from understanding the main control board and creating a development environment, and following this tutorial to guide you in correctly connecting lines and uploading code, you can realize rich project functions. The "Lesson\_1\_Arduino IDE" folder stores CH340 driver files, library files, etc. Please be sure to follow the guidance of this tutorial to complete the creation of the development environment. Next, let's start from the first section and enjoy the fun of the IoT Learning Kit!

## Table of contents

1. Create a development environment .....	11
1.1. Overview: .....	11
1.2.Install Arduino IDE .....	11
Arduino IDE on Mac OS X system .....	19
1.3. Install CH340 driver .....	22
1.4. Add ESP32 board in Arduino IDE .....	24
1.5. Add "Library" in Arduino IDE .....	27
2.Light up the LED .....	31
2.1.Overview .....	31
2.2.Control board resources .....	31
2.3.Connect the line .....	35
2. 4. Upload code program .....	36
2.5. Code analysis .....	39
3. Button control LED .....	41
3.1.Overview .....	41
3.2. Working principle .....	41

3.2.1.LED .....	41
3.3.Connect the lines .....	42
3.4.Upload code .....	43
3.5. Code analysis .....	45
4. Active buzzer .....	47
4.1.Overview .....	47
4.2. Working principle .....	47
4.3.Connect lines .....	48
4.4.Upload code .....	49
4.5. Code analysis .....	50
5. Traffic lights .....	52
5.1 Overview .....	52
5.2. Working principle .....	52
5.3 Connection lines .....	53
5.4 Upload code .....	54
5.5 Code analysis .....	55
6. Flowing water lamp .....	58

6.1.Overview .....	58
6.2. Working principle .....	58
6.3 Connection lines .....	58
6.4 Upload code .....	59
6.5 Code analysis .....	60
7. WS2812B .....	62
7.1. Overview .....	62
7.2. Working principle .....	62
7.3.Connect the lines .....	63
7.4.Upload code .....	64
7.5. Code analysis .....	65
8. Gradient RGB .....	67
8.1. Overview .....	67
8.2. Working principle .....	67
8.3 Connection lines .....	67
8.4. Upload code program .....	68
8.5 Code analysis .....	69

9.Servo control .....	70
9.1. Overview .....	70
9.2 Working principle .....	70
9.3 Connection lines .....	71
9.4 Upload code program .....	72
9.5 Code analysis .....	73
10. Photoresistor .....	75
10.1 Overview .....	75
10.2 Working principle .....	75
10.2.1 Photoresistor .....	75
10.3 Connection lines .....	76
10.4 Upload code program .....	77
10.5 Code analysis .....	79
11. Ultrasonic ranging .....	80
11.1 Overview .....	80
11.2 Working principle .....	80
11.2.1. Ultrasonic sensor .....	80

11.2.2.OLED .....	83
11.3 Connection lines .....	84
11.4 Upload code program .....	85
11.5 Code analysis .....	87
12. DHT11 temperature and humidity sensor .....	90
12.1. Overview .....	90
12.2. Working principle .....	90
12.3 Connection lines .....	91
12.4 Upload code program .....	92
12.5 Code analysis .....	94
13. Infrared remote control RGB .....	96
13.1 Overview .....	96
13.2 Working principle .....	96
13.3 Connection lines .....	98
13.4 Upload code program .....	99
13.5 Code analysis .....	101
14. Web switch control LED .....	104

14.1 Overview .....	104
14.2 Working principle .....	104
14.3 Connection lines .....	105
14.4 Upload code program .....	106
14.5 Code analysis .....	110
15. Web slider adjusts LED .....	113
15.1 Overview .....	113
15.2 Working principle .....	113
15.3 Connection lines .....	114
15.4 Upload code program .....	115
15.5 Code analysis .....	119
16. WebControlRGB .....	122
16.1 Overview .....	122
16.2 Working principle .....	122
16.3 Connection lines .....	123
16.4 Upload code program .....	124
16.5 Code analysis .....	127



17. Web display temperature and humidity .....	130
17.1 Overview .....	130
17.2 Working principle .....	130
17.3 Connection lines .....	131
17.4 Upload code program .....	132
17.5 Code analysis .....	135
18. Web control buzzer .....	139
18.1 Overview .....	139
18.2 Working principle .....	139
18.3 Connection lines .....	140
18.4 Upload code program .....	141
18.5 Code analysis .....	144
19. Photoresistor Controlled LED .....	146
19.1 Overview .....	146
19.2 Working principle .....	146
19.3 Connection lines .....	147
19.4 Upload code program .....	148

---

19.5 Code analysis .....	151
20. Web control servo .....	153
20.1 Overview .....	153
20.2 Working principle .....	153
20.3 Connection lines .....	154
20.4 Upload code procedure .....	155
20.5 Code analysis .....	158
21. WebDisplayUltrasound .....	162
21.1 Overview .....	162
21.2 Working principle .....	162
21.3 Connection lines .....	163
21.4 Upload code program .....	164
21.5 Code analysis .....	167

# 1. Create a development environment

## 1.1. Overview:

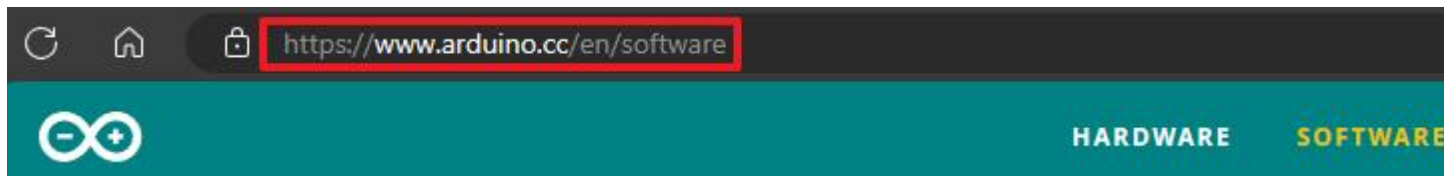
First, let's understand what Arduino is and then create a development environment. Install Arduino IDE as the development editor, and then install the necessary board and library files.

### Arduino

Arduino is an open source electronics platform based on easy-to-use hardware and software. Suitable for anyone working on interactive projects . Generally speaking, an Arduino project consists of hardware circuits and software code. The Arduino Integrated Development Environment (IDE) is the software side of the Arduino platform and is used to write and upload code to the control board . Let 's follow the tutorial to install the Arduino software (IDE) .

## 1.2.Install Arduino IDE

Enter it in the browser and click to go to <https://www.arduino.cc/en/software> web page



You can see the following web page locations:

HARDWARE

SOFTWARE

CLOUD

DOCUMENTATION ▼

COMMUNITY ▼

BLOG

ABOUT

# Downloads



## Arduino IDE 2.0.0

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

For more details, please refer to the [Arduino IDE 2.0 documentation](#).

### DOWNLOAD OPTIONS

**Windows** Win 10 and newer, 64 bits

**Windows** MSI installer

**Windows** ZIP file

**Linux** AppImage 64 bits (X86-64)

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

**macOS** 10.14: "Mojave" or newer, 64 bits

(Here we take the installation of version 2.0.0 IDE on win10 system as an example. For lower systems, please slide the web page below to install version 1.8.X software. At the same time, when you see this tutorial, there may be a newer version on the website!)

## Select the system version to which the software is adapted

Select the development software that is compatible with your computer system to download. Here we take Windows 10 as an example.



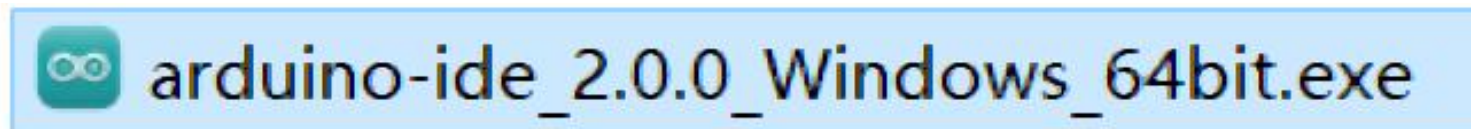
You can choose between installer (.exe) and Zip package. We recommend that you use the first "Windows Win10 and newer" to directly install everything you need to use the Arduino software (IDE), including drivers. While using Zip package, you need to install the driver manually. Of course Zip files are also useful if you want to create a portable installation.

Click "Windows Win10 and newer"



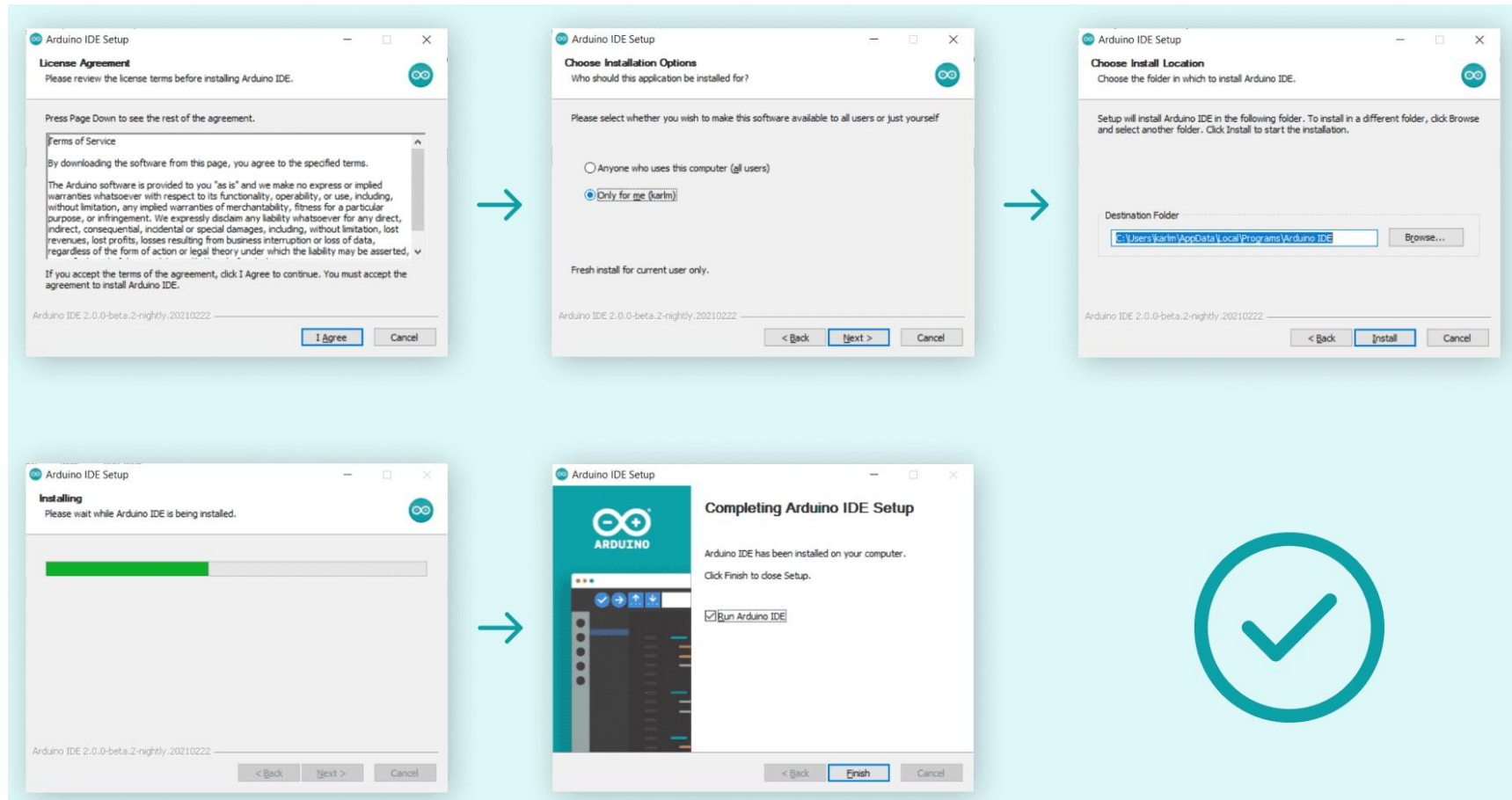
Click "JUST DOWNLOAD".

After the download is complete, you will get the installation package file with the “exe” suffix.



# Install Arduino IDE

Double-click to run the installer



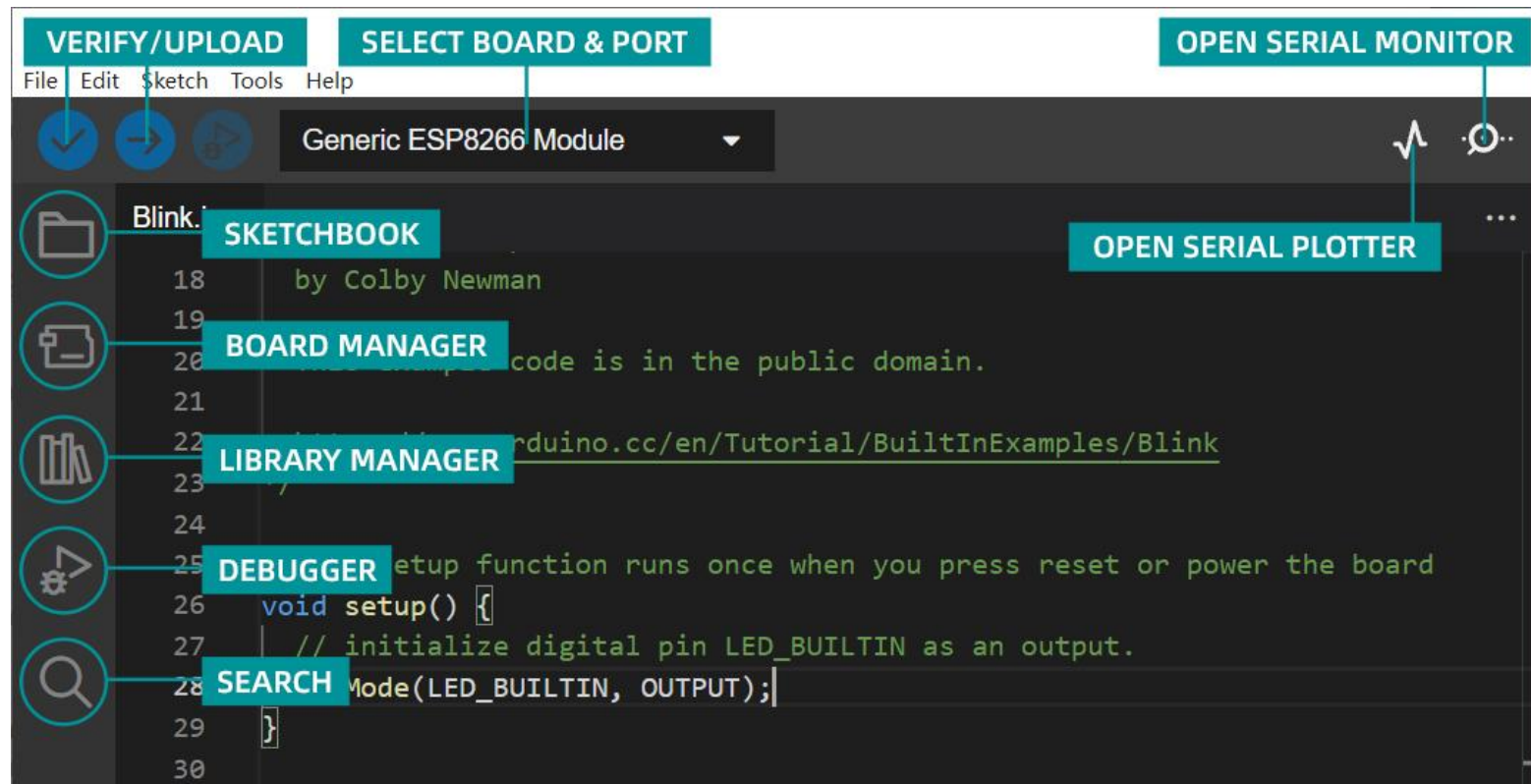
You can press "Browse..." to select the installation path or directly enter the directory you want. Then click "Install" to install. ( For Windows users, the driver installation dialog box may pop up during the installation process . When it pops up, please allow the installation )



After the installation is complete, an Arduino IDE software shortcut will be generated on the desktop. double-click to enter the Arduino software platform environment .

After the installation is complete, open the software and you will see the software platform interface as shown below (the interface will be different in different versions):





**Compile /Upload** - Compile and upload your code to your Arduino board

**Select board type and port number** - The detected Arduino board and port number will automatically show up here

**Project Sketch** - Here you will find all your sketches stored locally on your computer. Additionally, you can sync with

the Arduino Cloud or get your sketches from the online environment

**Board Manager** - Browse Arduino and third-party software packages that can be installed. For example, using the MKR WiFi 1010 board requires installing the Arduino SAMD Boards package

**Library Manager** - Browse thousands of Arduino libraries contributed by Arduino and its community

**Debugging** - Live testing and debugging of programs

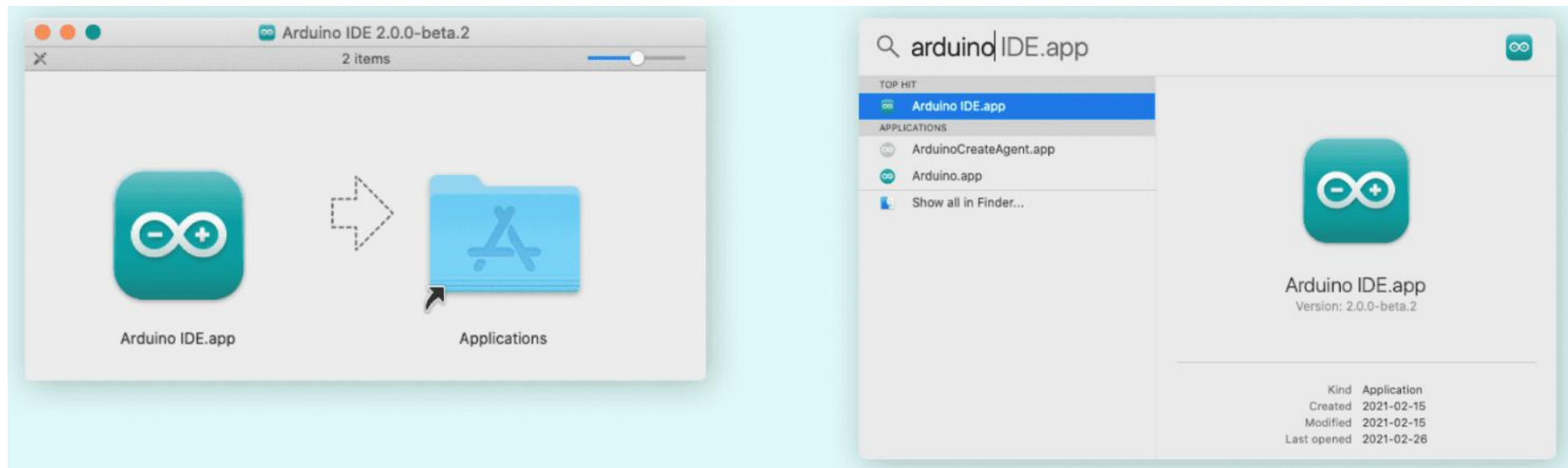
**Search** - Search for keywords in code

**Open Serial Monitor** - Opens the Serial Monitor tool as a new tab in the console

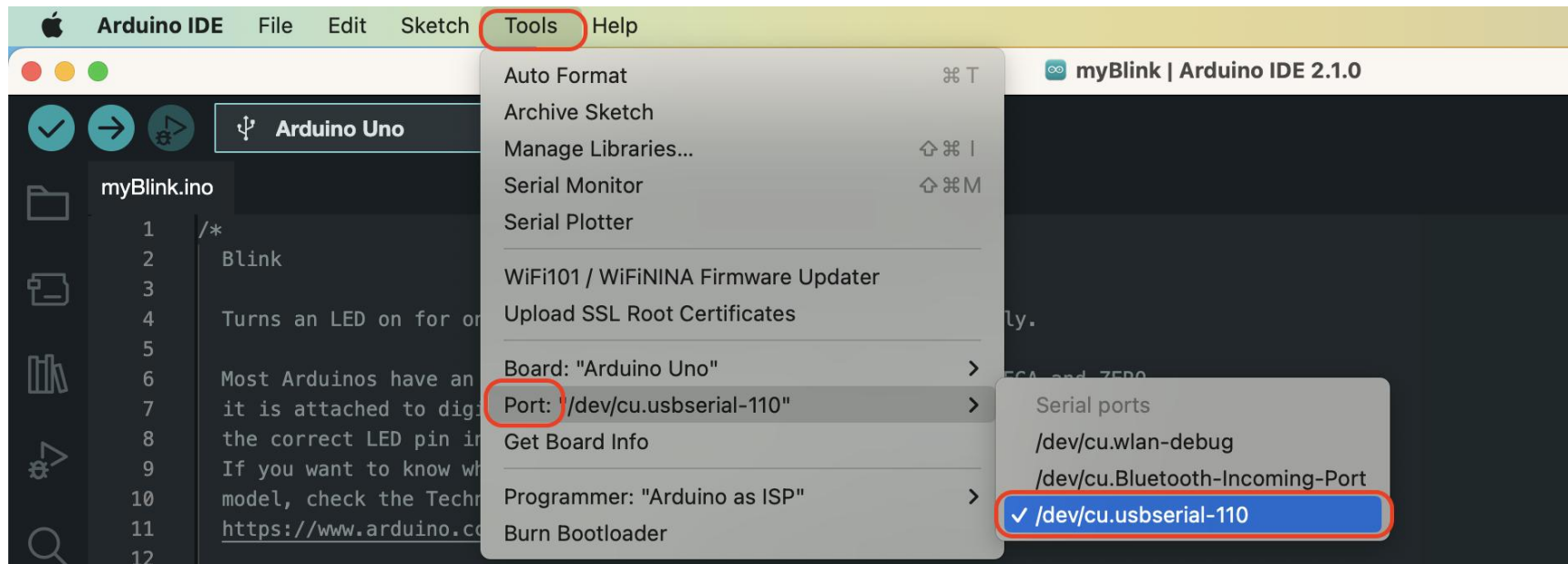
Programs written using the Arduino software (IDE) are called "Sketch". These "Sketch" are written in a text editor and saved with the file extension " .ino ". It is worth noting that the "ino" file must be saved in a folder with the same name. If the program is not opened in a folder with the same name, it will be forced to automatically create a file with the same name .

## Arduino IDE on Mac OS X system

Download and unzip the zip file, double-click Arduino.app to install; if there is no Java runtime library in your computer, the system will ask you to install it. After the installation is complete, you can run the Arduino IDE.



Similarly, when you connect the main control board to the computer with a USB cable, you find that the software recognizes "USBserial" as shown below



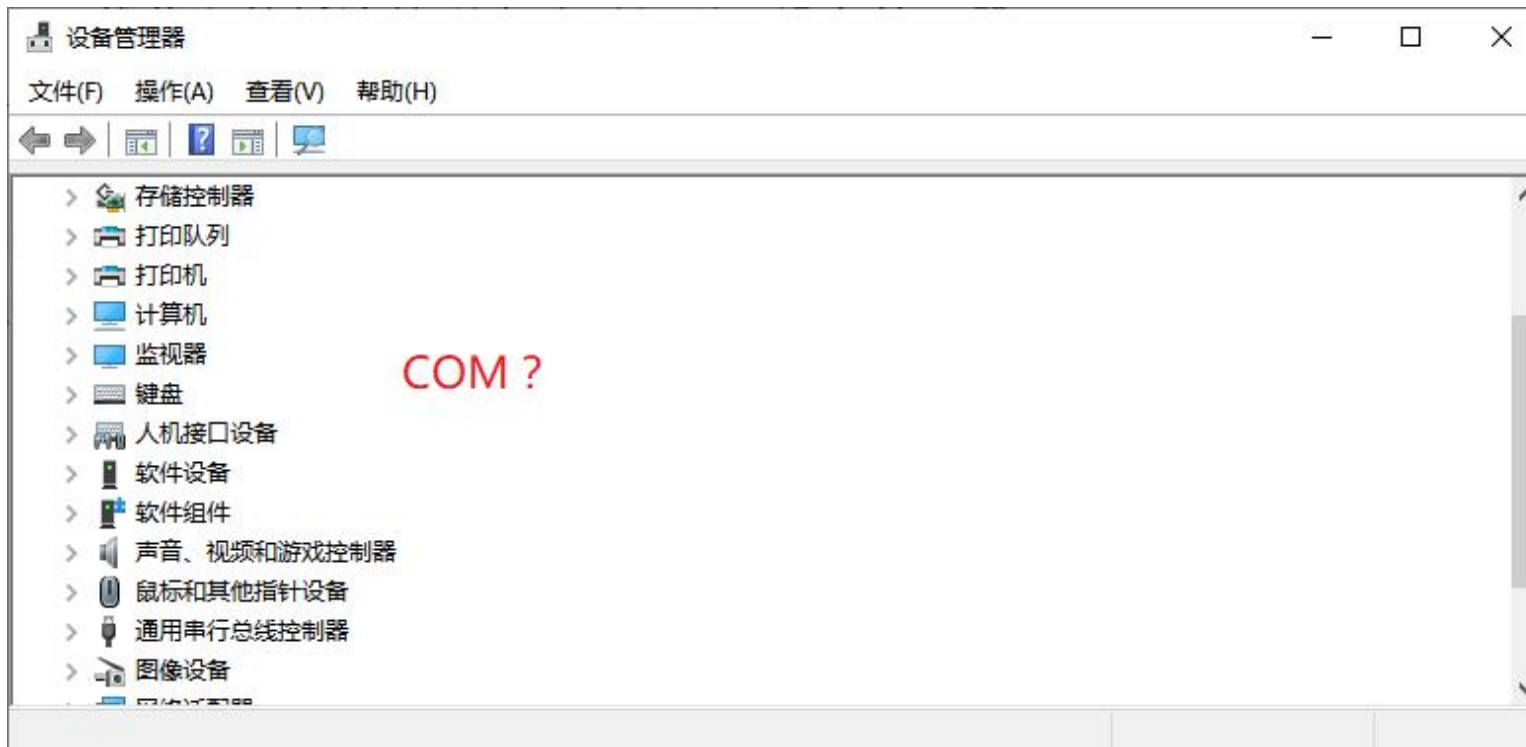
If you don't see "usbserial", you need to install the CH340 driver, please see the next section.

During the installation process, if the computer prompts that installation permission is required, you need to go to the "Security and Privacy" settings to allow the APP to come from any source.



### 1.3. Install CH340 driver

Sometimes the computer lacks the CH340 serial port driver. Use a USB cable to connect the main control board to the computer , then search and open "Device Manager". (If you can see CH340 under COM and LPT, you don't need to install it, just skip it)

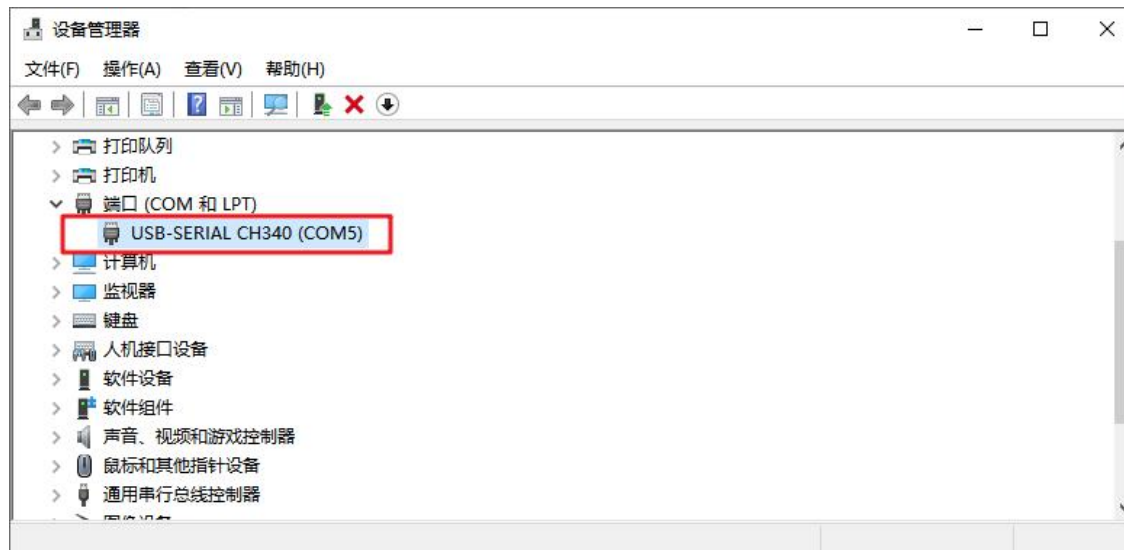


If you do not see the CH340 serial port in the picture above, you need to follow the following steps to install the driver.

Open the folder 2\_CH340 Driver and double-click the CH340 exe program installation package to start the installation.

0_Arduino software	2024/3/14 11:19	文件夹
1_Libraries	2024/3/14 11:19	文件夹
2_CH340 Driver	2024/3/14 11:19	文件夹
CH340 Driver File-Windows	2024/3/14 11:19	文件夹
CH340 Driver File-MAC	2024/3/14 11:19	文件夹

After the installation is completed, you can see that the driver has been displayed in the device manager (make sure the main control board is properly connected to the computer)



When the prompt "Installation failed" appears, connect the main control board to the computer with a USB cable, then uninstall and reinstall the driver.

### Install CH340 driver under Mac OS system

Open the folder 2\_CH340 Driver\CH340 Driver File-MAC and double-click to install the pkg file. If you want to update the version, you can check how to download the updated driver through the FAQ.

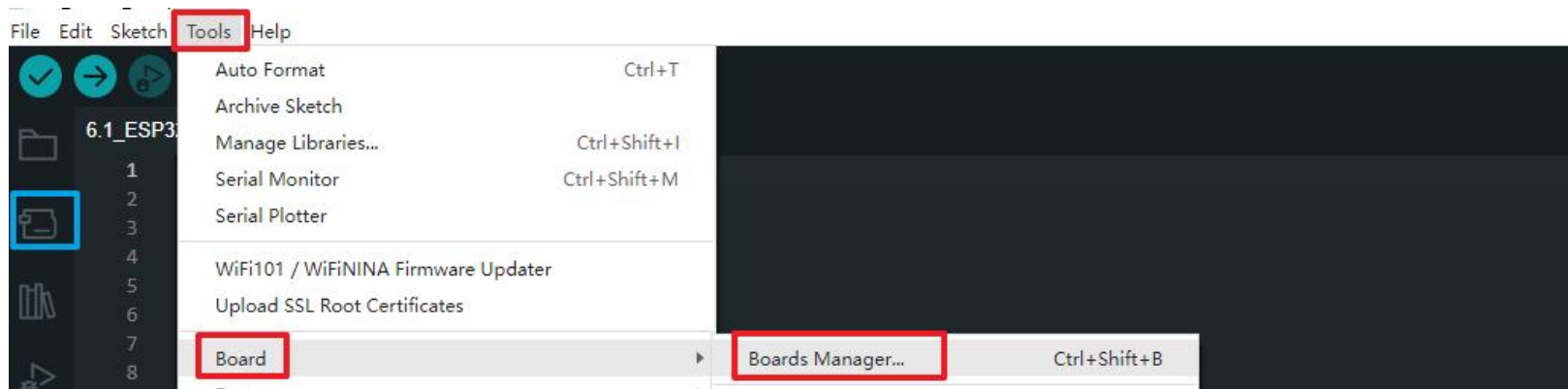
FAQ-EN.pdf	2024/3/14 11:33	Microsoft Edge ...	158 KB
CH340 Driver File-Windows	2024/3/14 11:19	文件夹	
CH340 Driver File-MAC	2024/3/14 11:19	文件夹	
名称	修改日期	类型	大小
CH34X_DRV_INSTALL_INSTRUCTIONS.pdf	2022/1/18 0:00	Microsoft Edge ...	533 KB
CH34xVCPDriver.pkg	2022/1/20 0:00	PKG 文件	1,905 KB

## 1.4. Add ESP32 board in Arduino IDE

### 1.4.1. Search and install ESP32 board

Open "Board Management" (the new version of IDE opens directly from the "Board Management" icon on the left)





Enter "ESP32" in the search bar, select the official board library and click "INSTALL" to install.



Downloading the ESP32 offline package can also install the ESP32 board: (double-click to run the exe installation)

External dropbox:

[https://www.dropbox.com/scl/fi/3j2sozoiuyg0cgmn1i75i/esp32\\_package\\_2.0.12\\_arduino.exe?rlkey=iml3j6zksbcz02iyvhuzxx7a9&dl=0](https://www.dropbox.com/scl/fi/3j2sozoiuyg0cgmn1i75i/esp32_package_2.0.12_arduino.exe?rlkey=iml3j6zksbcz02iyvhuzxx7a9&dl=0)

Intranet: [http://www.zhiyi.ltd/xiazai/APP/esp32\\_2.0.9.rar](http://www.zhiyi.ltd/xiazai/APP/esp32_2.0.9.rar)

#### 1.4.2. Check whether the board is installed successfully

You can see "ESP32" appearing under "Board"



## **1.5. Add "Library" in Arduino IDE**

### **1.5.1. How to install other libraries in Arduino IDE**

Once you are familiar with the Arduino software and using the built-in features, you may want to extend the capabilities of the Arduino with additional libraries.

### **1.5.2. What are Libraries?**

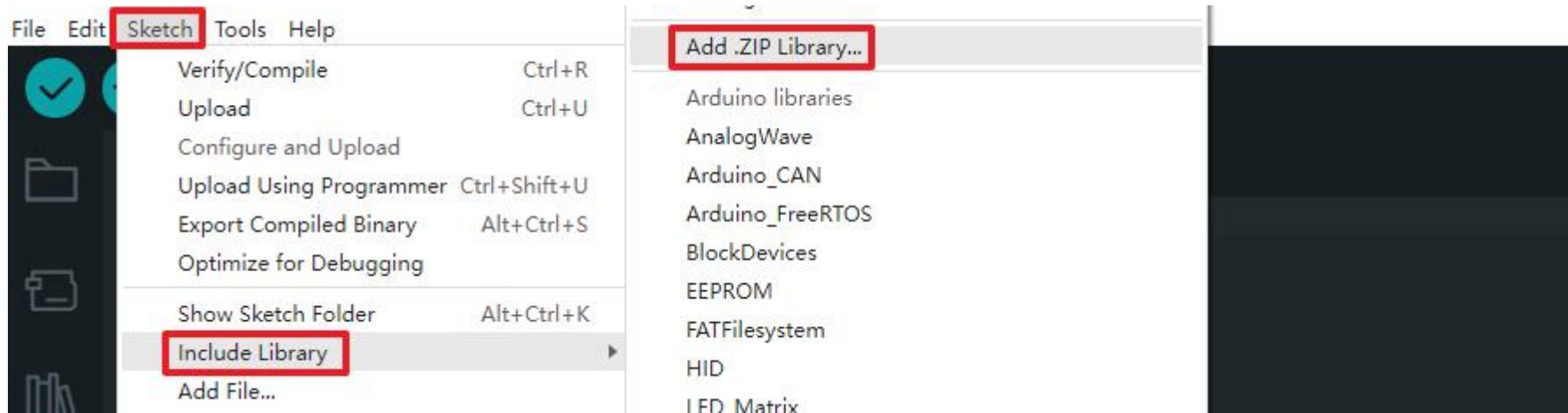
A library is a set of code that allows you to easily connect to sensors, displays, modules, and more. For example, the LiquidCrystal library allows you to easily interact with character LCD displays. There are thousands of libraries available for download directly through the Arduino IDE, all of which you can find in the Library Manager.

### **1.5.3. Method of adding library: import .zip library**

Libraries are usually distributed as ZIP files or folders. The name of the folder is the name of the library. Inside this folder will be a .cpp file, a .h file, usually a keywords.txt file, the examples folder and other files required by the library. Starting with version 1.0.5, you can install 3rd party libraries in the IDE. Do not unzip the downloaded library, leave it as is.

In the Arduino IDE, navigate to Sketch > Include Library > Add .ZIP Library and at the top of the drop-down list, select

the "Add .ZIP Library" option.

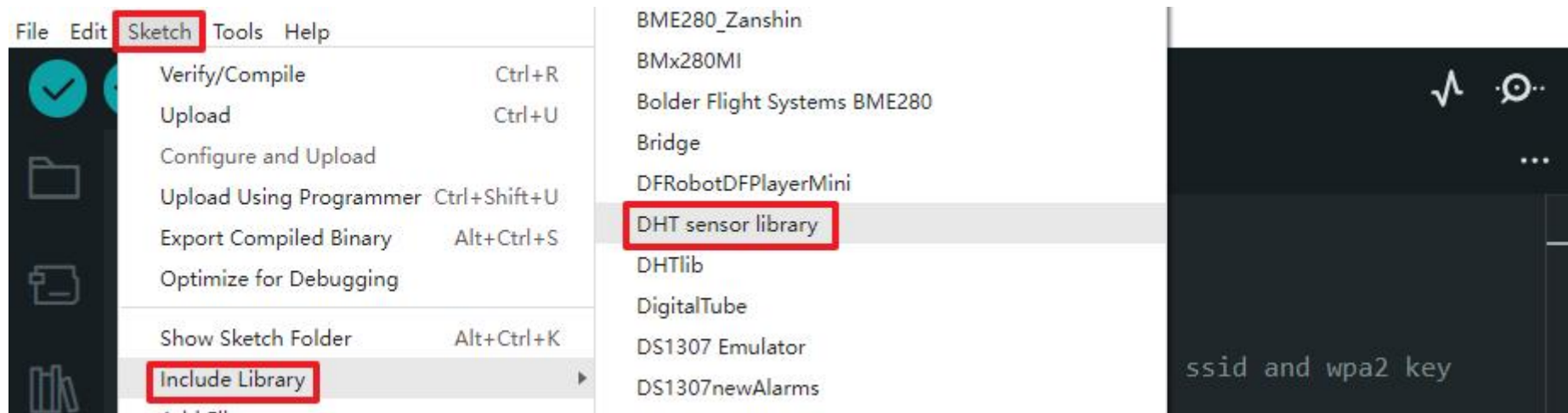


The system will prompt you to select the library to be added . Navigate to the path location of the *DHT\_sensor\_library.zip* file saved in your computer ( *Lesson\_1\_Arduino IDE\1\_Libraries\DHT\_sensor\_library.zip* ) as shown below and open it .














名称	修改日期	类型	大小
 DHT_sensor_library.zip	2023/12/19 13:57	WinRAR ZIP 压缩...	18 KB

Open the Sketch > Include Library menu. You should now see Libraries at the bottom of the drop-down menu. It's ready to use in your sketches .



Use this method to add all the required libraries to the Arduino IDE.

 Adafruit_BusIO.zip	2024/3/13 16:11	WinRAR ZIP 压缩文件	24 KB
 Adafruit_GFX_Library.zip	2024/3/13 15:53	WinRAR ZIP 压缩文件	343 KB
 Adafruit_SSD1306.zip	2024/3/13 15:45	WinRAR ZIP 压缩文件	36 KB
 Adafruit_Unified_Sensor.zip	2023/12/21 10:14	WinRAR ZIP 压缩文件	15 KB
 DHT_sensor_library.zip	2023/12/19 13:57	WinRAR ZIP 压缩文件	18 KB
 ESP32Servo.zip	2024/3/25 18:06	WinRAR ZIP 压缩文件	50 KB
 ESPAsyncTCP.zip	2023/12/21 10:14	WinRAR ZIP 压缩文件	44 KB
 FastLED.zip	2024/2/28 17:49	WinRAR ZIP 压缩文件	525 KB
 IRremote.zip	2024/3/26 11:25	WinRAR ZIP 压缩文件	1,010 KB
 IRremoteESP8266.zip	2024/3/26 11:22	WinRAR ZIP 压缩文件	6,248 KB
 Servo.zip	2024/3/13 15:02	WinRAR ZIP 压缩文件	120 KB

## 2.Light up the LED

### 2.1.Overview

This section mainly focuses on understanding the characteristics of the main control board and expansion board, learning how to burn code and lighting the green LED on the expansion board through the program.

### 2.2.Control board resources

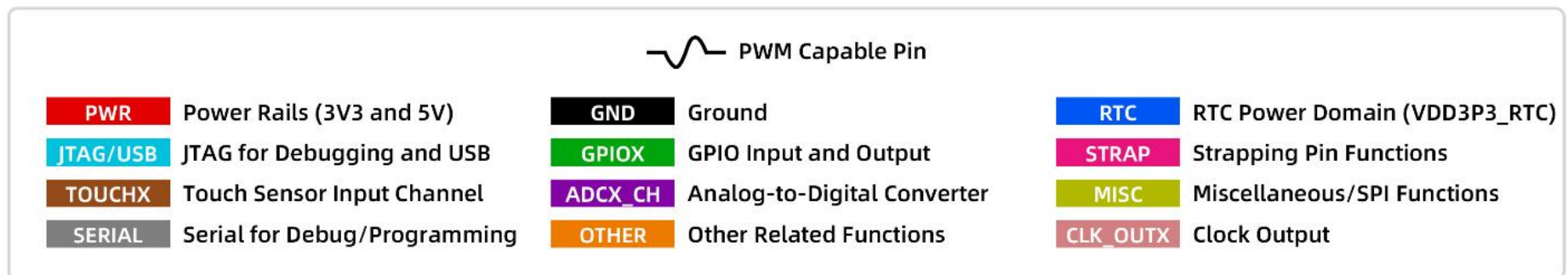
#### 2.2.1.ESP32-S3-PLUS main control board

All I/O ports provided by the main control board support PWM;

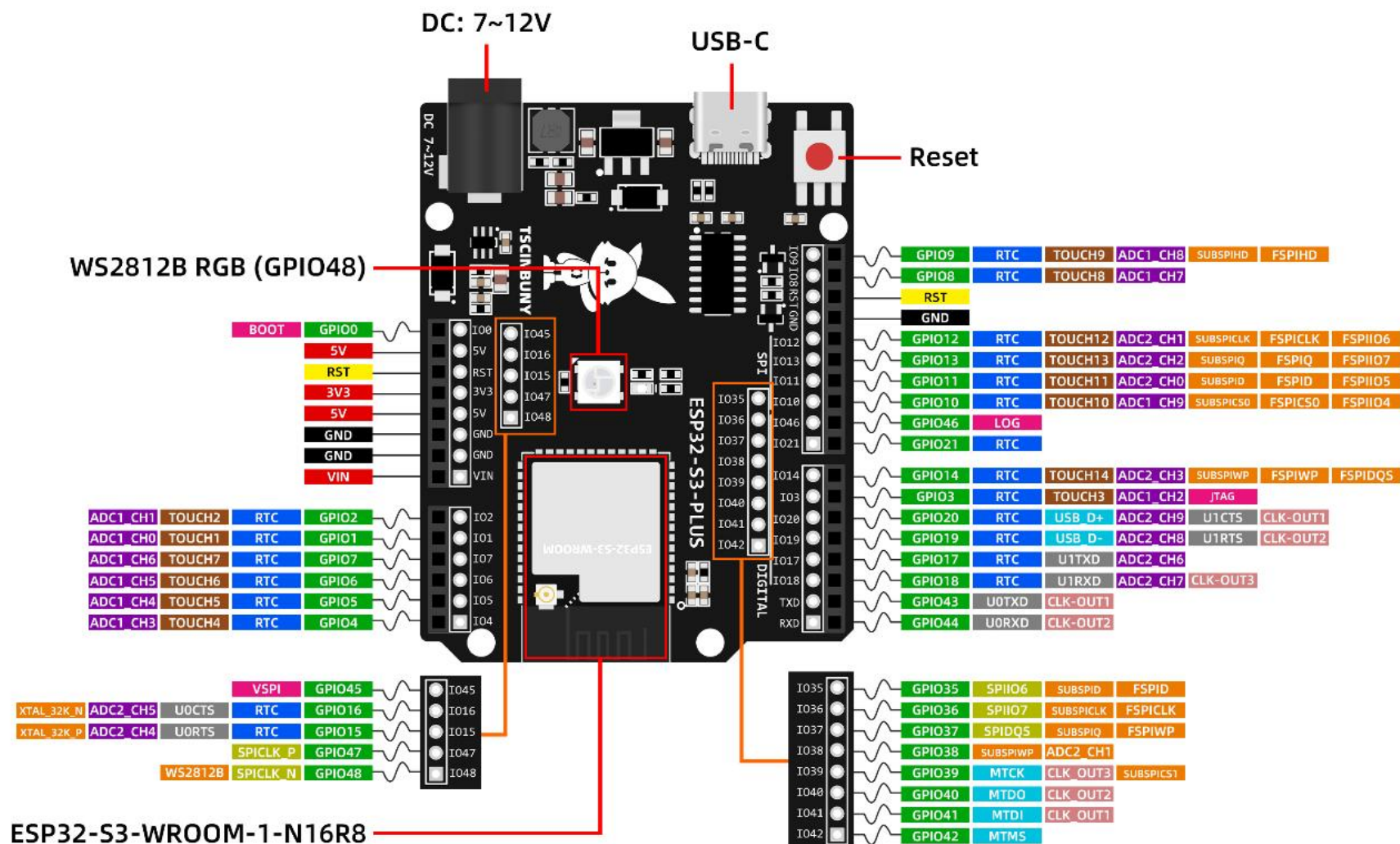
Some I/O second functions also support SPI/IIC/clock output and other functions;

Type-C input voltage is 5V;

DC input voltage 7V~9V;

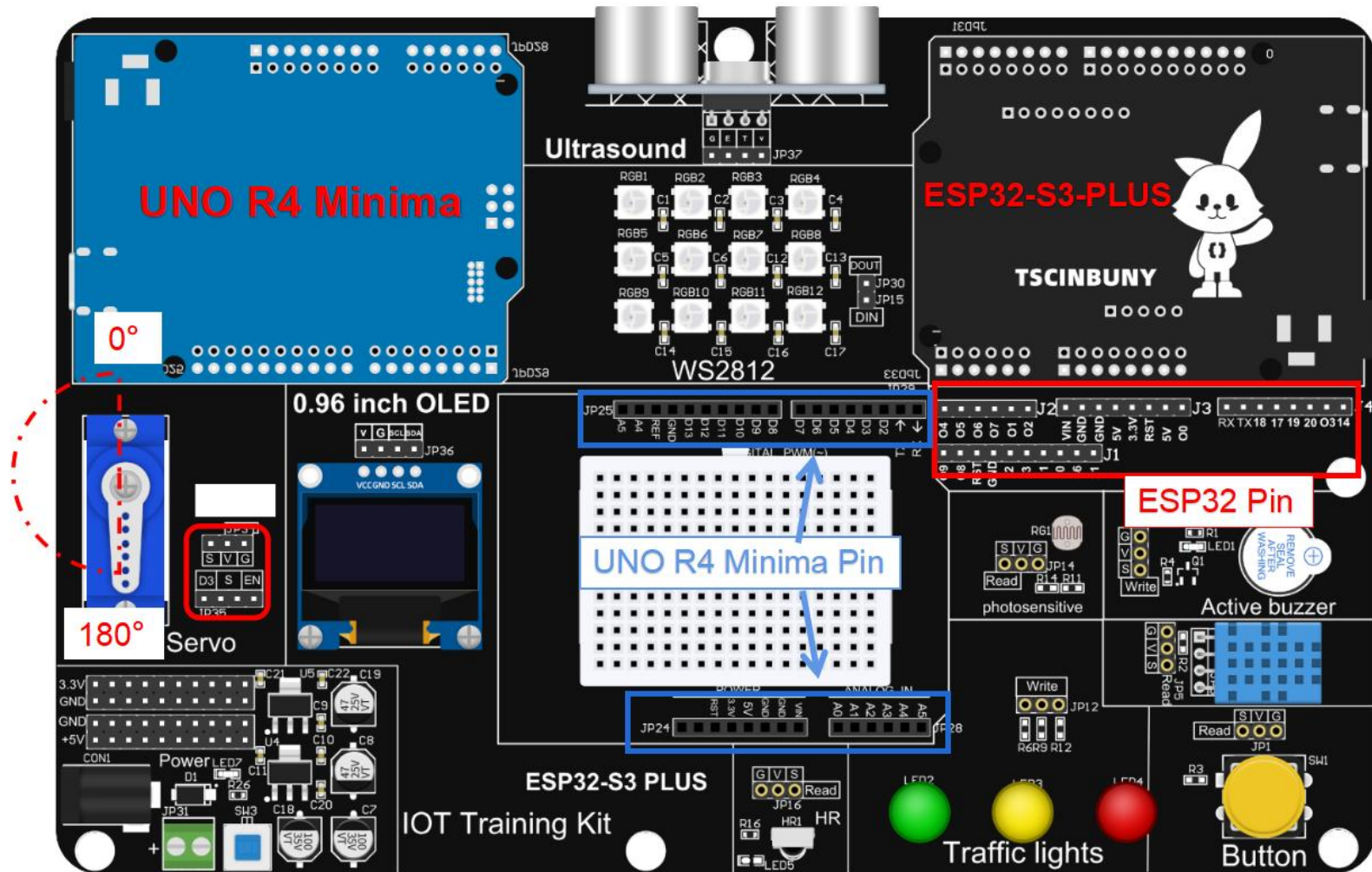








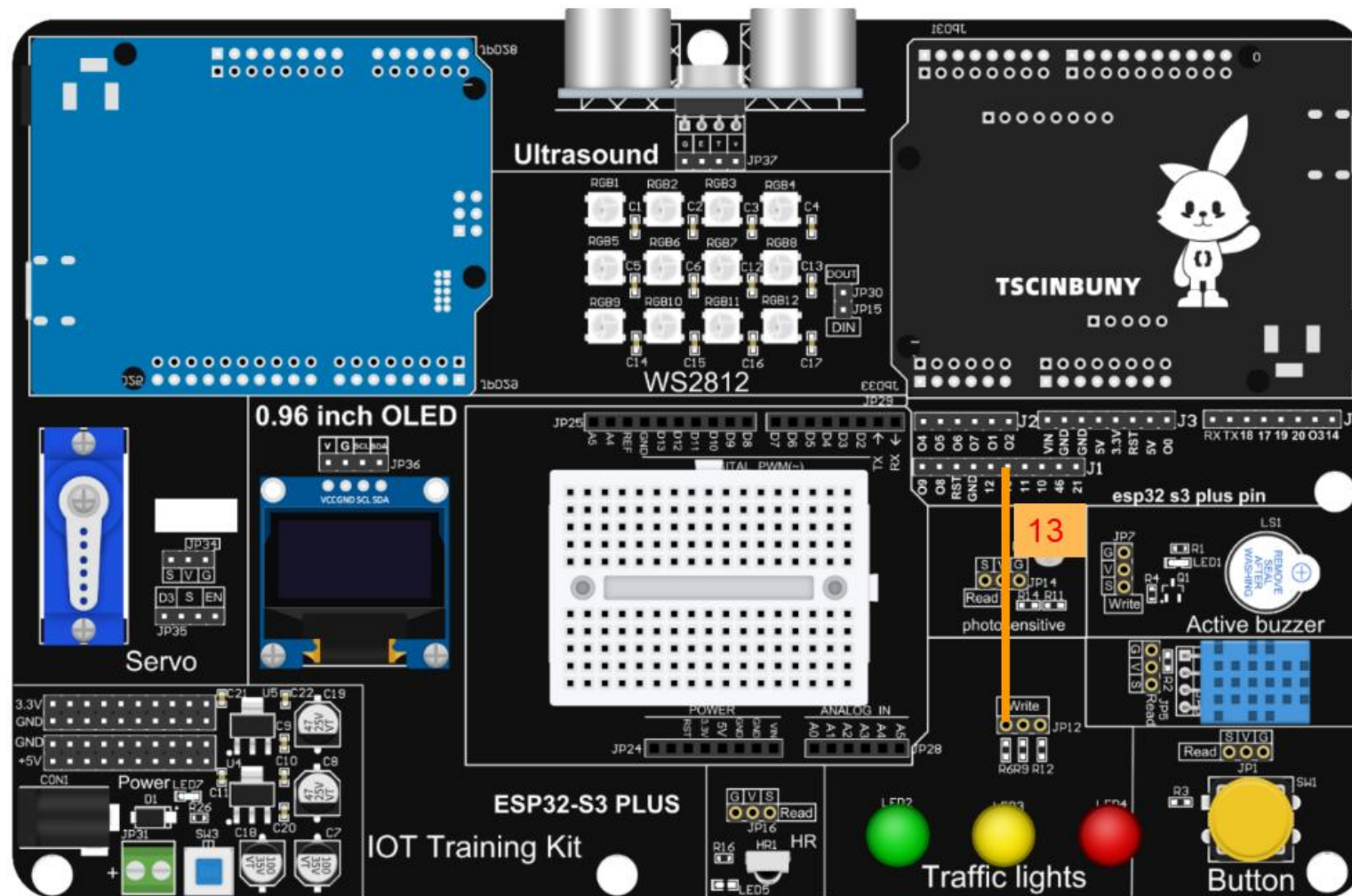
## 2.2.2.Expansion board



**Expansion board overview:**

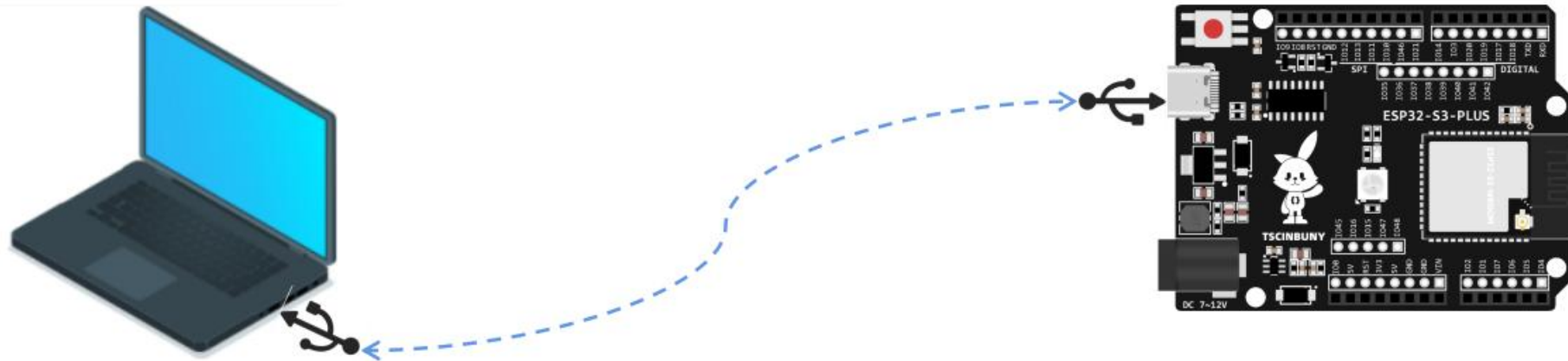
1. The expansion board design supports two main control boards, the UNO R4 Minima on the left and the ESP32 on the right, which should be installed correctly during use;
2. The onboard resources also include ultrasonic sensor, ws2812, servo, OLED, photoresistor, buzzer, DHT11, infrared receiver, LED and buttons;
3. The UNOR4 pin is led to the female header;
4. The ESP32 pins are led out to the pin header;
5. Before using the servo crank, burn the code to reset it and then fix the screws so that the swing range rotates back and forth between  $0^{\circ}$  and  $180^{\circ}$  as shown in the figure to avoid touching the pin header next to it.
6. The internal circuit of the board has already connected the GND of all devices to the common ground and VCC to 5V, so there is no need to connect the VCC and GND lines when using the sensor.

## 2.3.Connect the line



## 2. 4. Upload code program

### 2.4.1. Connect the main control board to the computer with a USB cable

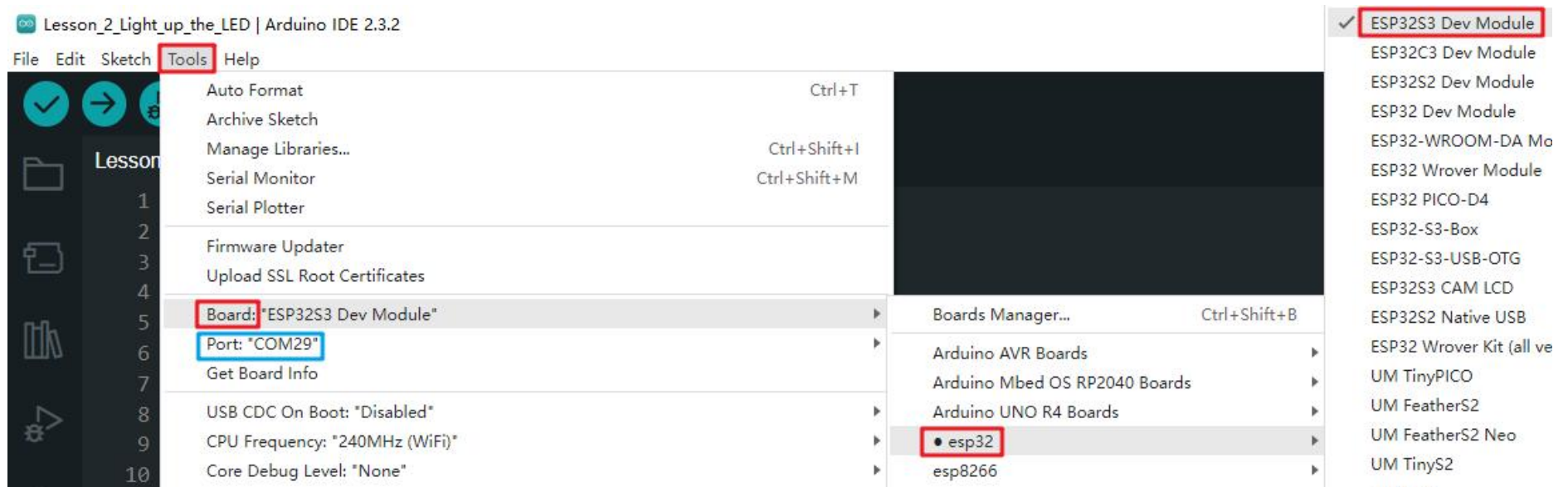


### 2.4.2. Open the " 2\_ESP32\_S3\_PLUS \Lesson\_2\_Light\_up\_the\_LED" code file

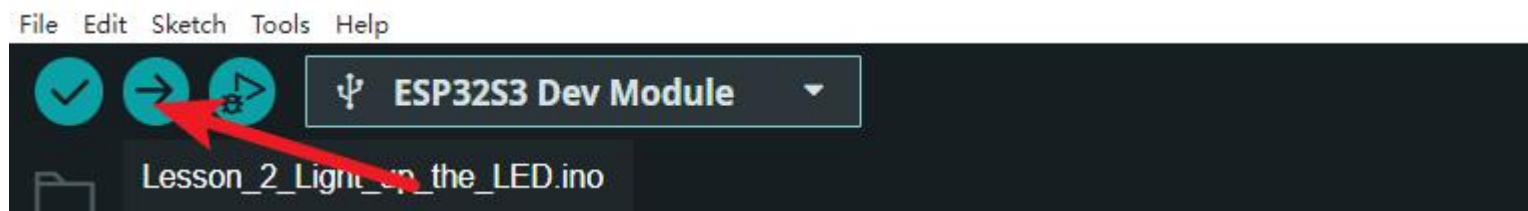
Lesson_1_Arduino IDE	2024/3/13 14:32	文件夹
Lesson_2_Light_up_the_LED	2024/3/13 13:43	文件夹
Lesson_3_Button_control_LED	2024/3/5 12:00	文件夹
Lesson_4_active_buzzer	2024/3/13 13:43	文件夹
Lesson_5_Traffic_light	2024/3/13 13:43	文件夹
Lesson_6_Flow_light	2024/3/13 13:43	文件夹
Lesson_7_WS2812B	2024/3/6 11:10	文件夹
Lesson_8_Gradient_RGB_light	2024/3/6 11:09	文件夹

Select the board type as ESP32S3 Dev Module. When plugging in the USB, a new COM number will be displayed. Select it.

Here it is COM29, but the actual COM number will be different for everyone.

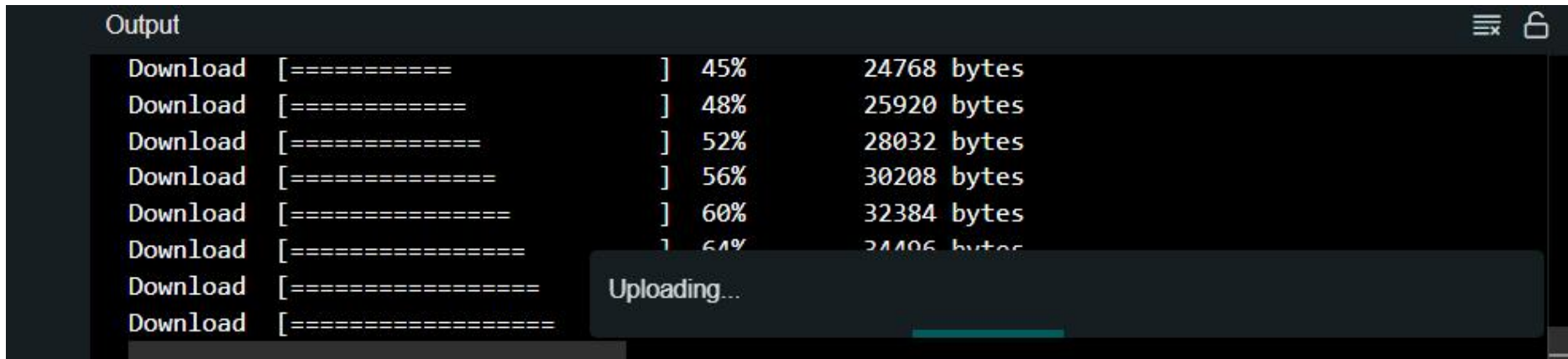


Click "Upload" to start compiling and uploading the program to the main control board.



Waiting for the program to upload;





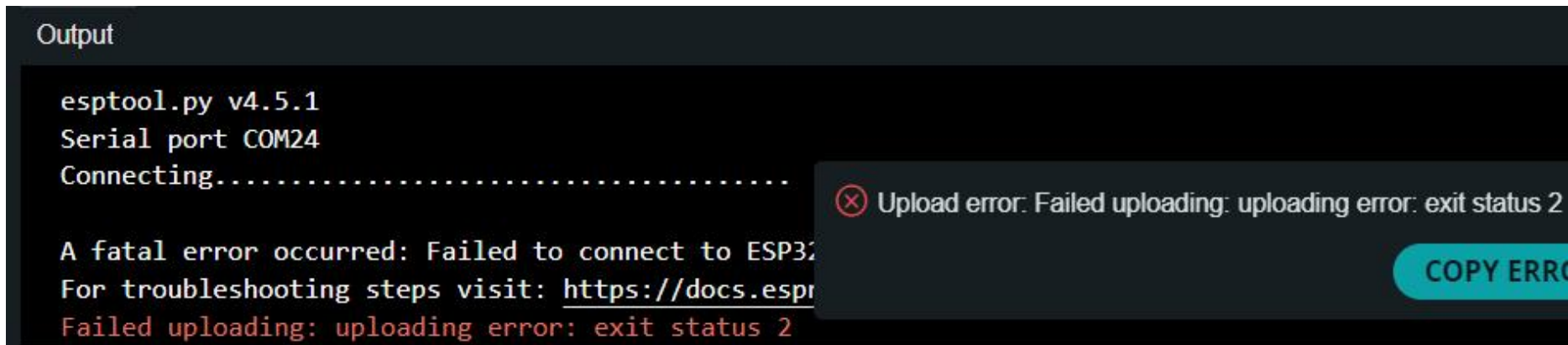
```

Output
Download [=====] 45% 24768 bytes
Download [=====] 48% 25920 bytes
Download [=====] 52% 28032 bytes
Download [=====] 56% 30208 bytes
Download [=====] 60% 32384 bytes
Download [=====] 64% 34560 bytes
Download [=====] Uploading...
Download [=====] Uploading...

```

### pay attention:

When an error occurs, the "Connecting..." screen will always appear:



```

Output
esptool.py v4.5.1
Serial port COM24
Connecting.....
A fatal error occurred: Failed to connect to ESP32:
For troubleshooting steps visit: https://docs.espressosystems.com/en/latest/esp32/faq.html
Failed uploading: uploading error: exit status 2

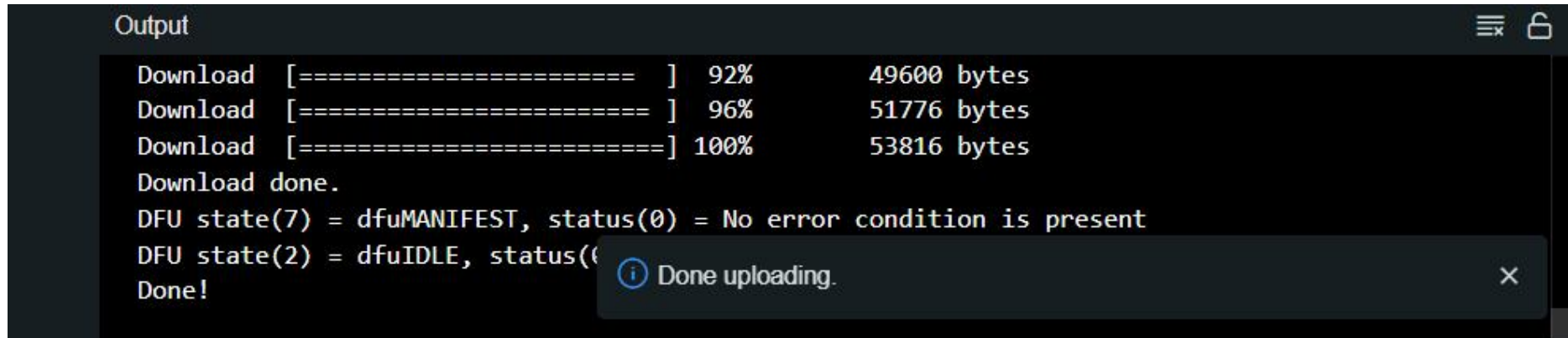
```

Upload error: Failed uploading: uploading error: exit status 2

COPY ERROR

Please remove the ESP32S3 board from the expansion board, connect it to the computer separately, and then upload it again.

After the program upload is completed , you can see that the green light of LED2 is on.



The screenshot shows the 'Output' window of the Arduino IDE. It displays the progress of the program upload in three rows, each with a progress bar, a percentage, and the number of bytes transferred. The progress bars are filled with '=' characters. The first row shows 92% completion (49600 bytes), the second row shows 96% completion (51776 bytes), and the third row shows 100% completion (53816 bytes). Below the progress bars, the text 'Download done.' is displayed. This is followed by two lines of DFU status information: 'DFU state(7) = dfuMANIFEST, status(0) = No error condition is present' and 'DFU state(2) = dfuIDLE, status(0) = No error condition is present'. The final line of the output is 'Done!'. A small notification box with a blue 'i' icon and the text 'Done uploading.' is overlaid on the bottom right of the output window.

```

Output
Download [===== ] 92%      49600 bytes
Download [===== ] 96%      51776 bytes
Download [=====] 100%     53816 bytes
Download done.
DFU state(7) = dfuMANIFEST, status(0) = No error condition is present
DFU state(2) = dfuIDLE, status(0) = No error condition is present
Done!
Done uploading.
  
```

## 2.5. Code analysis

this " sketch " code consists of comments. These are not actual program instructions; instead, they simply explain how to make the program work. They are there for your ease of reading . Everything between " /\* " and " \*/ " at the top of the sketch is a block comment that explains the purpose of the sketch.

Single-line comments begin with "//" and everything up to the end of the line is considered a comment.

The first part of the code is:

```
11  #define LED 13  //定义LED灯输出引脚 Define the LED lamp output pin
12  void setup() {
13      pinMode(LED, OUTPUT);    //定义引脚的工作模式 Define the operation mode of the pin
14      digitalWrite(LED, LOW);  //输出低电平 Output low level
15  }
```

Every sketch requires a "set" function , which is a "Void setup()" function, this is executed when the reset button is pressed. It is executed whenever the board resets for any reason, such as first power-up or after uploading a sketch.

The next step is to name the pin and set the output. Here, set " LED " as the output port, and digitalWrite(LED,LOW) controls the pin to output a low level, which is to turn off.

The sketch must also have a " loop " function. Unlike the "Set " function, which only runs once , after a reset, the "Loop " function will start again immediately after completing the command run.

```
8  void loop() {
9      //因为LED灯负极已经连接到GND, 只需要输出高电平即可运行点亮LED灯
10     //Because the negative electrode of the LED lamp is already connected to the GND,
11     digitalWrite(LED, HIGH);
12 }
```

Inside the loop function, the command turns on the LED ( high ).



## 3. Button control LED

### 3.1.Overview

This section focuses on learning how to use buttons to control LEDs to implement the delay control function.

### 3.2. Working principle

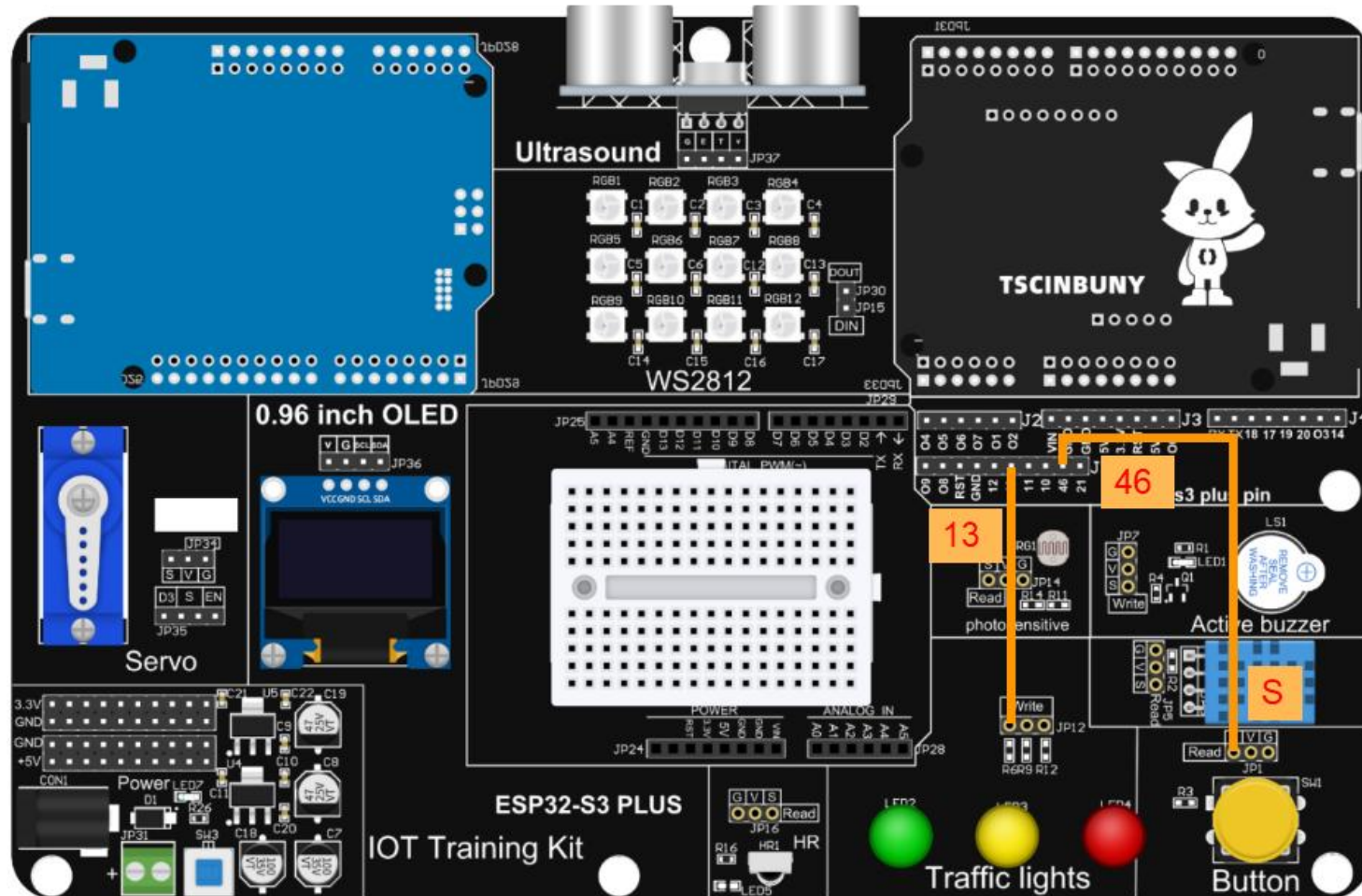
The key signal pin S has a pull-up resistor by default, so it is high level and changes to low level when the key is pressed. At this time, the input of pin 46 of the main control board is low level. By judging the pin status as a condition, the LED pin level is flipped, thereby controlling the LED to turn on and off.

#### 3.2.1.LED



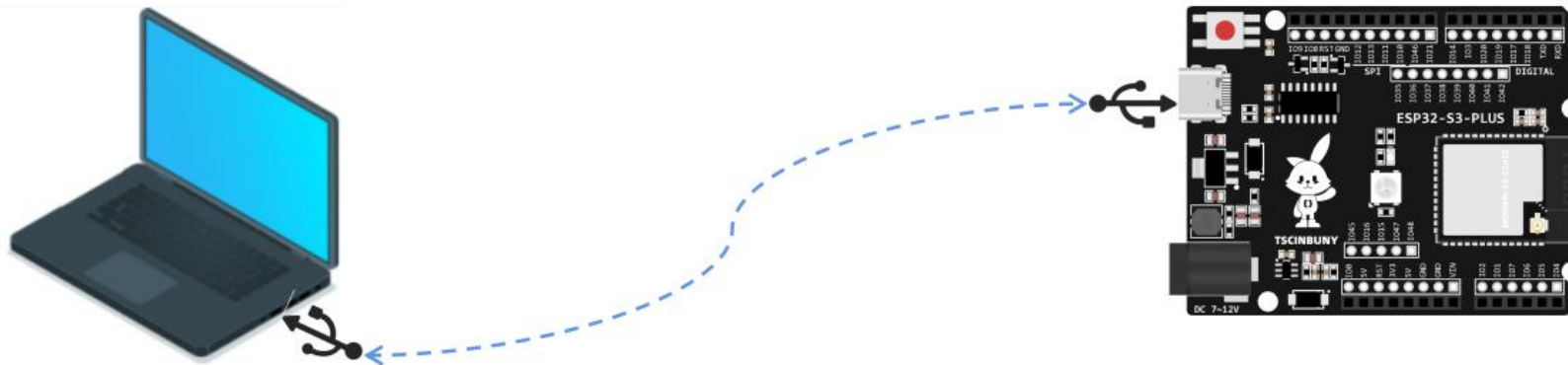
LED (Light Emitting Diode), which converts electrical energy into light energy, also has one-way conductivity and a reverse breakdown voltage of about 5v. Its forward volt-ampere characteristic curve is very steep. In the development board, the negative electrodes of the LEDs are all common to the ground. When the signal pin is set to high level , the LED is on, and when it is set to low level , the LED is turned off.

### 3.3. Connect the lines



### 3.4.Upload code

#### 3.4.1. Connect the main control board to the computer with a USB cable

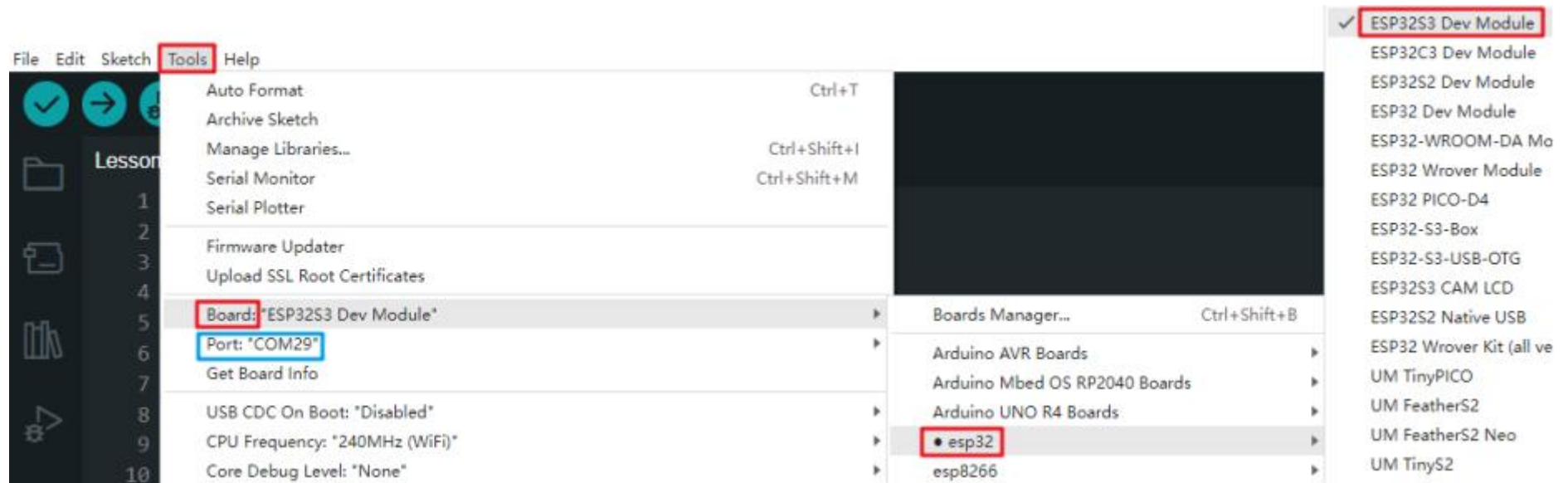


#### 3.4.2. Open the program file (path: 2\_ESP32\_S3\_PLUS \ Lesson\_3\_Button\_control\_LED )

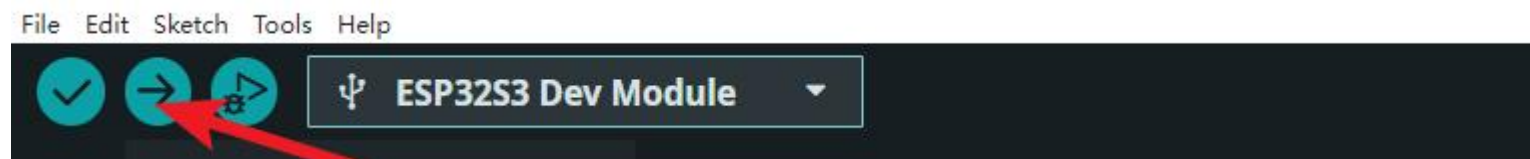
Lesson_1_Arduino IDE	2024/3/13 14:32	文件夹
Lesson_2_Light_up_the_LED	2024/3/13 13:43	文件夹
Lesson_3_Button_control_LED	2024/3/5 12:00	文件夹
Lesson_4_active_buzzer	2024/3/13 13:43	文件夹
Lesson_5_Traffic_light	2024/3/13 13:43	文件夹
Lesson_6_Flow_light	2024/3/13 13:43	文件夹
Lesson_7_WS2812B	2024/3/6 11:10	文件夹
Lesson_8_Gradient_RGB_light	2024/3/6 11:09	文件夹
Lesson_9_Steering_wheel_control	2024/3/5 14:15	文件夹

Also select the board type as ESP32S3 Dev Module, and select the COM number that is newly displayed when the USB is

plugged in. In this case, it is COM29, but the actual COM number will be different for everyone.



Click "Upload" to start compiling and uploading the program to the main control board.



Wait for the program upload to complete .

```

Output
Writing at 0x00024000... (83 %)
Writing at 0x00028000... (91 %)
Writing at 0x0002c000... (100 %)
Wrote 265616 bytes (195725 compressed) at 0x00000000 in 17.2 seconds (effective 123.2 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
Done uploading.

```

Press the button to light up LED2, press it again to turn off LED2 , and repeat the above effect.

### 3.5. Code analysis

First define the led pin , button pin and define two variables

```

2  #define LED 13
3  #define button 46

4  int key_ok = 0;    //定义项目所需的数据变量
5  int LED_en = 0;

```

Set the button pin as input and the LED pin as output

```

6  void setup() {
7      pinMode(button, INPUT);    //设置按键引脚为输入 Set the key pin as input
8      pinMode(LED, OUTPUT);      //设置LED引脚为输出 Set the LED pin as the output
9  }

```

Obtain the button status within the loop structure, use the if...else... statement to determine whether the button is pressed or released, implement the debounce function for the button status variable key\_ok, and reverse the LED\_en state when the button is released, thus affecting the LED Write high and low levels to make the LED turn on or off.

```
11 void loop() {  
12     // 判断是否有按钮被按下，读取按钮的电平 Determine if a button has been pressed and read the button level  
13     if (digitalRead(button)) {  
14         if (key_ok)  
15         {  
16             key_ok = 0;  
17             if (LED_en) LED_en = 0;    //确定最后一个标志位是否建立 Determines whether the last flag bit is established  
18             else LED_en = 1;  
19         }  
20     } else {  
21         delay(20);    //延时20毫秒  
22         if (!digitalRead(button)) key_ok = 1;  
23     }  
24     //当按下按键时，LED灯输出引脚的电平反转 When the key is pressed, the level of the LED light output pin is reversed  
25     if (LED_en) digitalWrite(LED, HIGH);  
26     else digitalWrite(LED, LOW);  
27 }
```



## 4. Active buzzer

### 4.1. Overview

The electronic buzzer is DC powered and equipped with an integrated circuit. They are widely used in computers, printers, copiers, alarms, electronic toys, automotive electronic equipment, telephones, timers and other electronic products for voice equipment. Buzzers can be divided into active buzzers and passive buzzers. Turn the two buzzer pins upward. The one with the green circuit board is the passive buzzer, and the other one sealed with black tape is the active buzzer. In this section you will learn how to use an active buzzer to generate a sound that sounds for half a second and then stops for half a second.

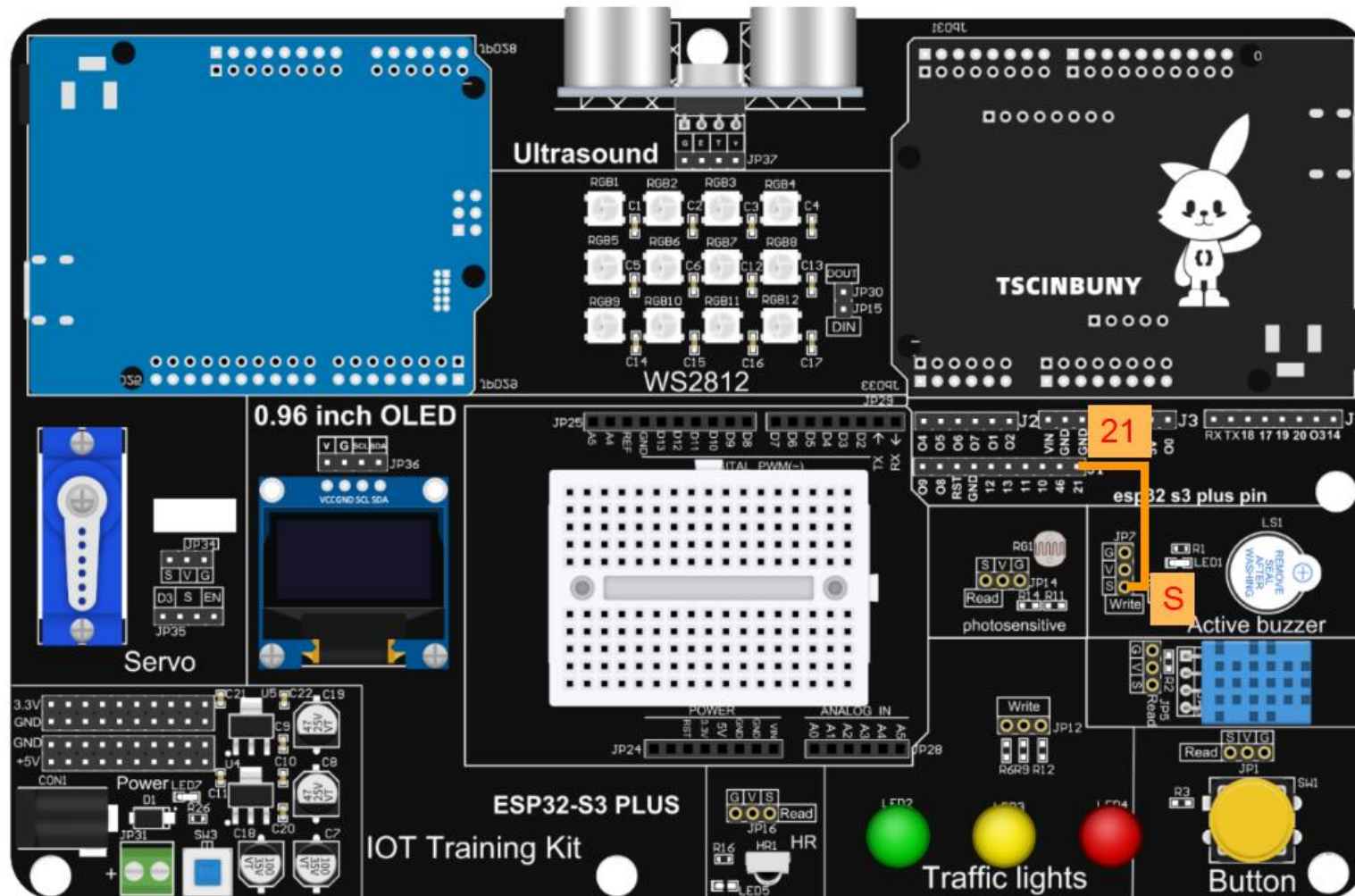
### 4.2. Working principle

#### 4.2.1. Active buzzer

An active buzzer has an internal oscillation source, and it can sound as long as it is given a high level. Use the delay function to make the buzzer sound regularly.



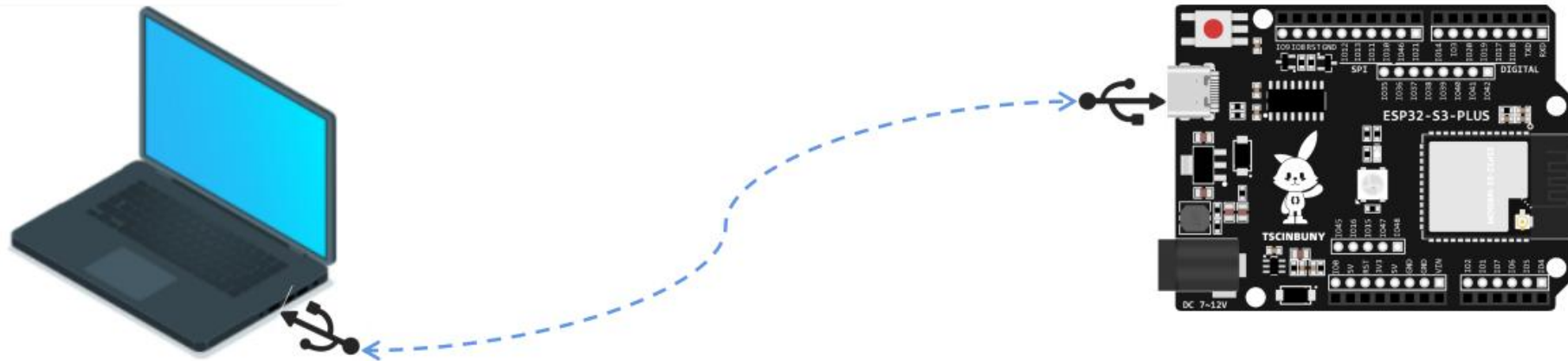
### 4.3.Connect lines





## 4.4.Upload code

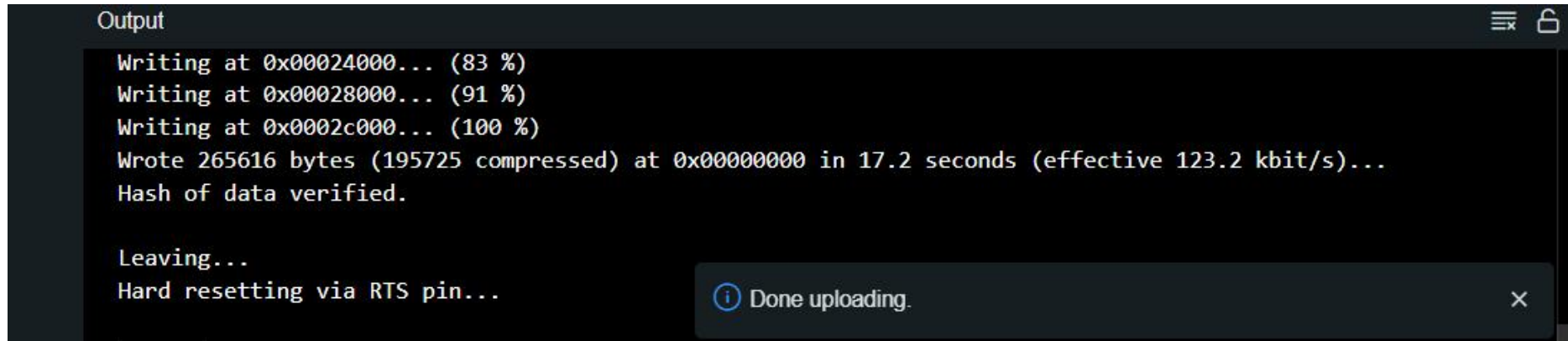
### 4.4.1. Connect the main control board to the computer using a USB cable



### 4.4.2. Open the program file (path: 2\_ESP32\_S3\_PLUS\ Lesson\_4\_active\_buzzer )

Lesson_1_Arduino IDE	2024/3/13 14:32	文件夹
Lesson_2_Light_up_the_LED	2024/3/13 13:43	文件夹
Lesson_3_Button_control_LED	2024/3/5 12:00	文件夹
<b>Lesson_4_active_buzzer</b>	2024/3/13 13:43	文件夹
Lesson_5_Traffic_light	2024/3/13 13:43	文件夹
Lesson_6_Flow_light	2024/3/13 13:43	文件夹
Lesson_7_WS2812B	2024/3/6 11:10	文件夹
Lesson_8_Gradient_RGB_light	2024/3/6 11:09	文件夹
Lesson_9_Steering_gear_control	2024/3/5 14:15	文件夹

Also select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board.



The screenshot shows the 'Output' window of an IDE. The text inside the window is as follows:

```
Writing at 0x00024000... (83 %)
Writing at 0x00028000... (91 %)
Writing at 0x0002c000... (100 %)
Wrote 265616 bytes (195725 compressed) at 0x00000000 in 17.2 seconds (effective 123.2 kbit/s)...
Hash of data verified.

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

At the bottom right of the window, there is a notification box that says "Done uploading." with an information icon on the left and a close icon on the right.

## 4.5. Code analysis

Define the buzzer pin and set the buzzer as an output

```
2  #define Buzzer 21           //定义有源蜂鸣器引脚 Define the active buzzer pin
3  void setup() {
4      pinMode(Buzzer, OUTPUT); //定义引脚为输出工作模式 Define the pin as the output working mode
5  }
```

The buzzer pin is at a high level for 500 milliseconds and then at a low level for 500 milliseconds to achieve a "beep" sound effect at intervals.

```
7 void loop() {  
8   digitalWrite(Buzzer, HIGH); //输出高电平, 蜂鸣器响 Output high level, buzzer goes off  
9   delay(500);  
10  digitalWrite(Buzzer, LOW); //输出低电平, 蜂鸣器不响 The output is low and the buzzer  
11  delay(500);  
12 }
```

## 5. Traffic lights

### 5.1 Overview

In this section, you will learn to light multiple LEDs and control the green, yellow, and red lights to light up at intervals through a delay function to achieve the effect of a traffic light.

### 5.2. Working principle



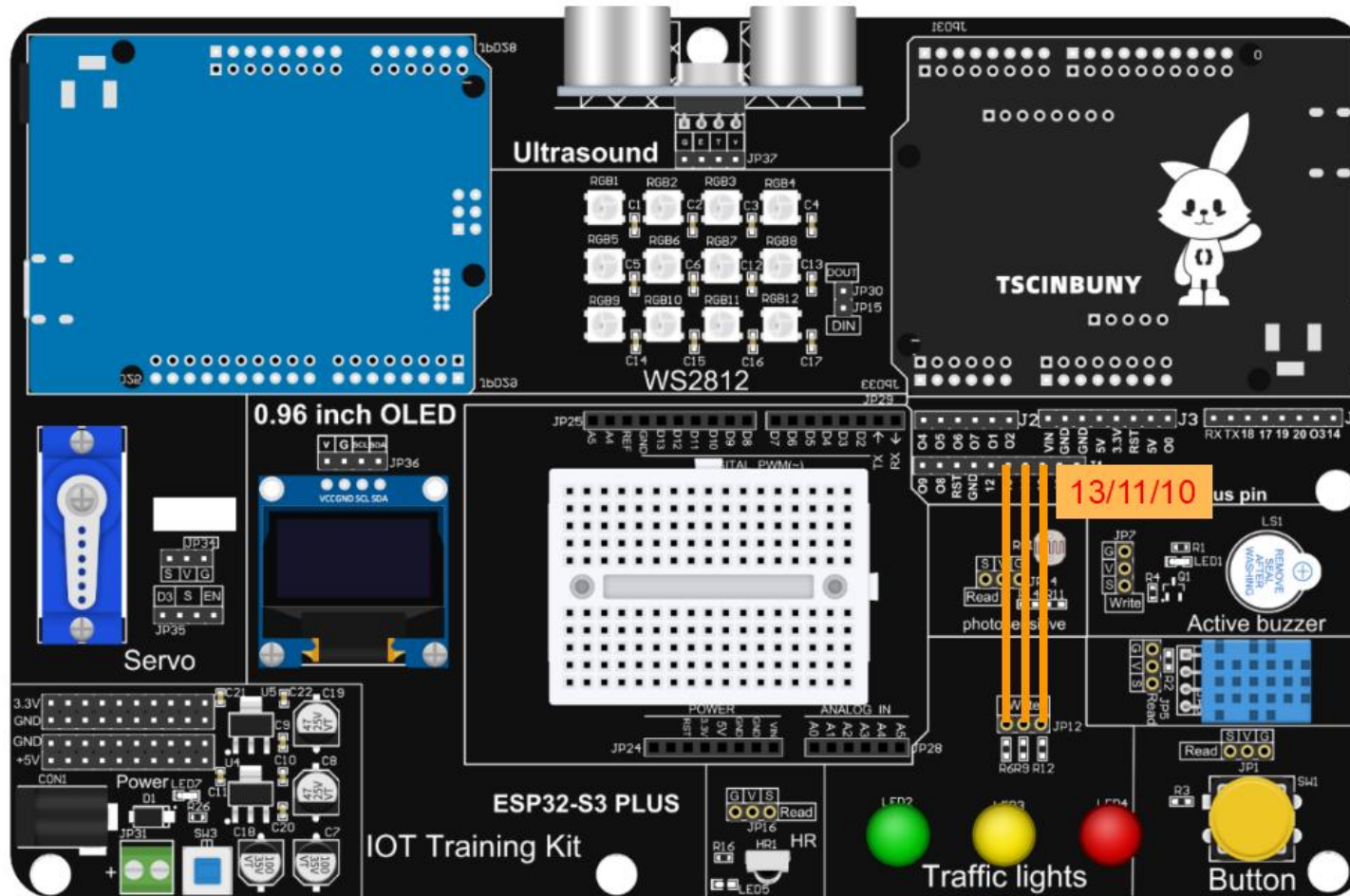
There are three colors of traffic lights, green light, yellow light and red light. The pins connected to the main control board are 13/11/10. By controlling the high level of the three pins, the corresponding lights can be lit. A single light cycle is as follows: first let the green light be on for a period of time, which means that the traffic is in a passable state, then flash a reminder before the green light ends and switch to the yellow light, then switch to the yellow light, and wait for a short period of time before switching to the yellow light. Switch to red light. The red light also waits for a period of time, indicating that traffic is prohibited from passing, and flashes as a reminder before the red light ends and is ready to switch to the green light.

So the single-cycle process is roughly as follows:

Leave the green light on for five seconds , the green light flashing every 500 milliseconds, the yellow light on for 1 second,

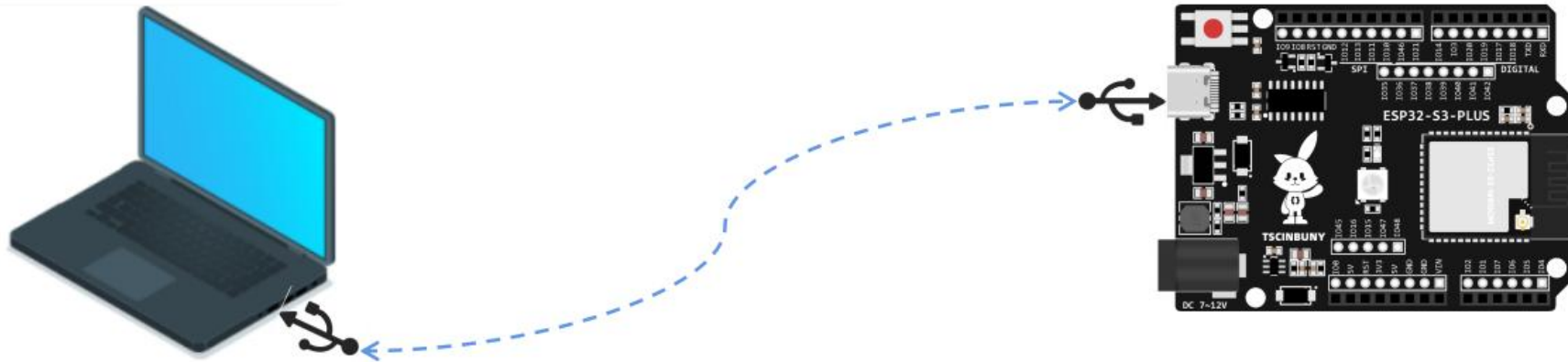
the red light on for 5 seconds, the red light flashing every 500 milliseconds, and so on.

### 5.3 Connection lines



## 5.4 Upload code

### 5.4.1 Connect the main control board to the computer using a USB cable



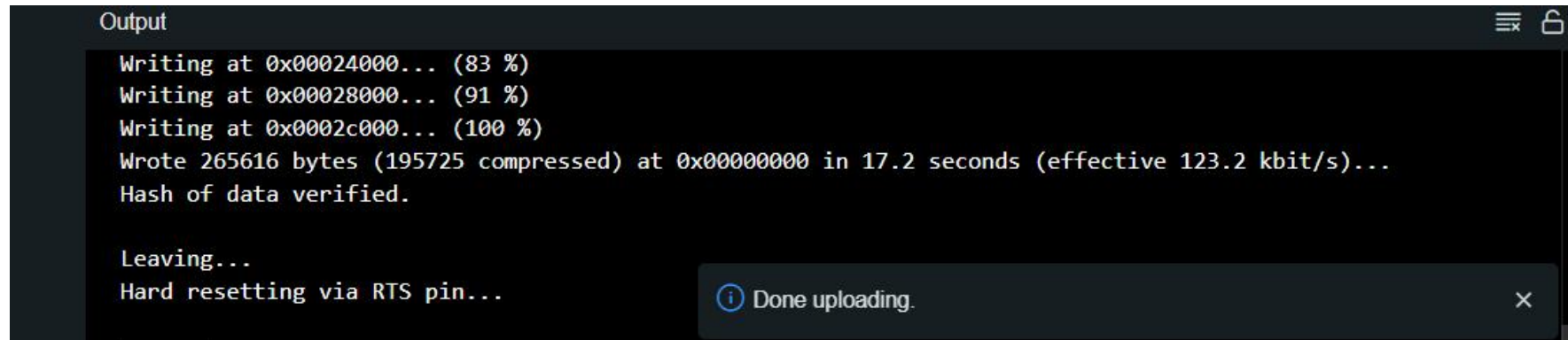
### 5.4.2 Open the program file (path: 2\_ESP32\_S3\_PLUS\ Lesson\_5\_Traffic\_light )

Lesson_1_Arduino IDE	2024/3/13 14:32	文件夹
Lesson_2_Light_up_the_LED	2024/3/13 13:43	文件夹
Lesson_3_Button_control_LED	2024/3/5 12:00	文件夹
Lesson_4_active_buzzer	2024/3/13 13:43	文件夹
<b>Lesson_5_Traffic_light</b>	2024/3/13 13:43	文件夹
Lesson_6_Flow_light	2024/3/13 13:43	文件夹
Lesson_7_WS2812B	2024/3/6 11:10	文件夹
Lesson_8_Gradient_RGB_light	2024/3/6 11:09	文件夹
Lesson_9_Steering_gear_control	2024/3/5 14:15	文件夹

Also select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged



in . Then click "Upload" to start compiling and uploading the program to the main control board.



```

Output
Writing at 0x00024000... (83 %)
Writing at 0x00028000... (91 %)
Writing at 0x0002c000... (100 %)
Wrote 265616 bytes (195725 compressed) at 0x00000000 in 17.2 seconds (effective 123.2 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
Done uploading.
  
```

## 5.5 Code analysis

Define three LED pins

```

3  #define Green 13    //定义绿色灯引脚 Define the green light pin
4  #define Yellow 11   //定义黄色灯引脚 Define the yellow light pin
5  #define red 10      //定义绿色灯引脚 Define the green light pin
  
```

Set pin as output

```

7  void setup() {
8    // 定义引脚为输出工作模式 Define the pin as the output working mode
9    pinMode(Green, OUTPUT);
10   pinMode(Yellow, OUTPUT);
11   pinMode(red, OUTPUT);
  
```

The loop function executes a single light cycle.

The red light first lights up for 5 seconds and then flashes every 500 milliseconds.

```
14 void loop() {  
15     digitalWrite(Green, HIGH); //点亮绿色灯 Turn on the green light  
16     digitalWrite(Yellow, LOW); //熄灭黄色灯 Put out the yellow light  
17     digitalWrite(red, LOW); //熄灭红色灯 Turn off the red light  
18     delay(5000); //让绿灯亮五秒钟 Turn on the green light for five seconds  
19  
20     //绿灯每500毫秒闪烁一次 The green light flashes every 500 milliseconds  
21     digitalWrite(Green, HIGH);  
22     delay(500);  
23     digitalWrite(Green, LOW);  
24     delay(500);  
25     digitalWrite(Green, HIGH);  
26     delay(500);  
27     digitalWrite(Green, LOW);  
28     delay(500);  
29     digitalWrite(Green, HIGH);  
30     delay(500);
```

Yellow light on for 1 second

```
32     //黄灯亮1秒 The yellow light is on for 1 second  
33     digitalWrite(Green, LOW);  
34     digitalWrite(Yellow, HIGH);  
35     digitalWrite(red, LOW);  
36     delay(1200);
```

Finally, the red light turns on for 5 seconds and then flashes every 500 milliseconds.



```
38 //红灯亮5秒钟 The red light stays on for 5 seconds
39 digitalWrite(Green, LOW);
40 digitalWrite(Yellow, LOW);
41 digitalWrite(red, HIGH);
42 delay(5000);
43
44 //红灯每500毫秒闪烁一次 The red light flashes every 500 milliseconds
45 digitalWrite(red, HIGH);|
46 delay(500);
47 digitalWrite(red, LOW);
48 delay(500);
49 digitalWrite(red, HIGH);
50 delay(500);
51 digitalWrite(red, LOW);
52 delay(500);
53 digitalWrite(red, HIGH);
54 delay(500);
```

## 6. Flowing water lamp

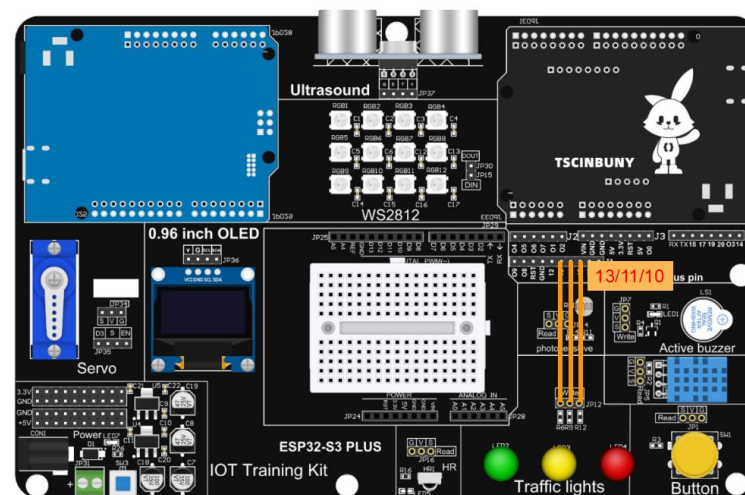
### 6.1. Overview

In this section, you will learn how to use three LED lights to achieve a running water effect.

### 6.2. Working principle

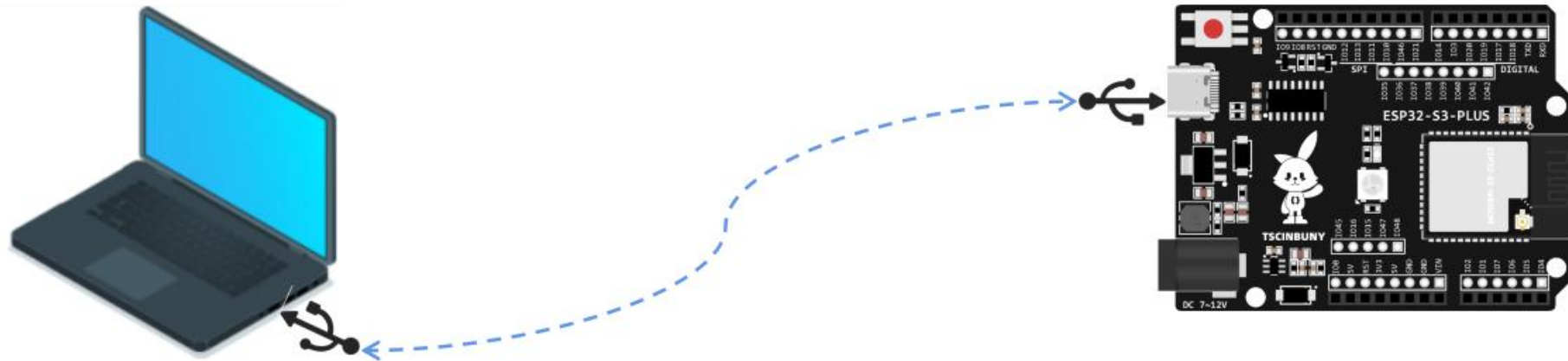
As in the previous section "Traffic Light Project", the LED on/off is controlled by controlling the high/low level of the three LED light pins. After setting the waiting time for on and off, you can see the lighting process of the three LEDs one by one, simulating the effect of running water lamps.

### 6.3 Connection lines



## 6.4 Upload code

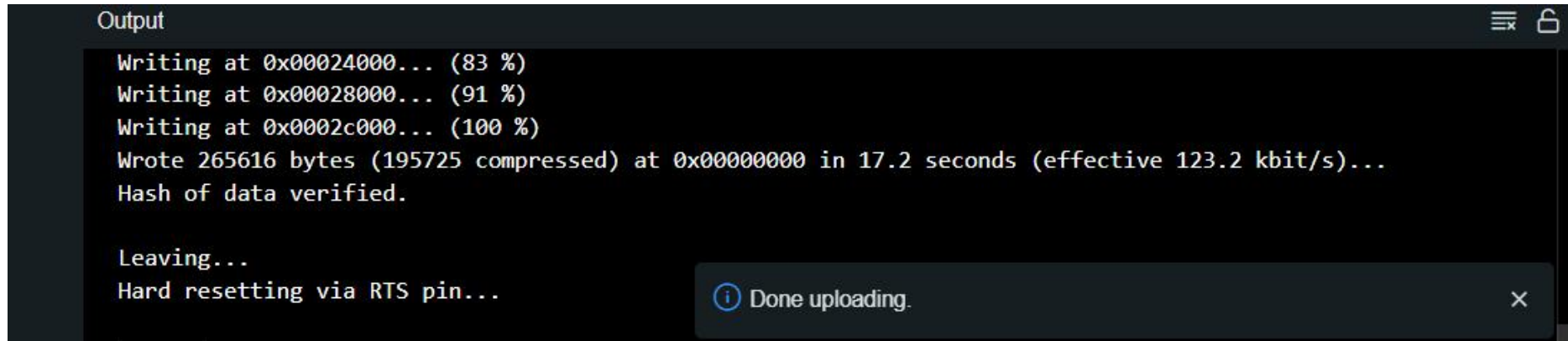
### 6.4.1 Connect the main control board to the computer using a USB cable



### 6.4.2 Open the program file (path: 2\_ESP32\_S3\_PLUS\ Lesson\_6\_Flow\_light )

Lesson_1_Arduino IDE	2024/3/13 14:32	文件夹
Lesson_2_Light_up_the_LED	2024/3/13 13:43	文件夹
Lesson_3_Button_control_LED	2024/3/5 12:00	文件夹
Lesson_4_active_buzzer	2024/3/13 13:43	文件夹
Lesson_5_Traffic_light	2024/3/13 13:43	文件夹
<b>Lesson_6_Flow_light</b>	2024/3/13 13:43	文件夹
Lesson_7_WS2812B	2024/3/6 11:10	文件夹
Lesson_8_Gradient_RGB_light	2024/3/6 11:09	文件夹
Lesson_9_Steering_gear_control	2024/3/5 14:15	文件夹

Also select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board.



The screenshot shows the 'Output' window of an IDE. The text inside the window is as follows:

```
Writing at 0x00024000... (83 %)
Writing at 0x00028000... (91 %)
Writing at 0x0002c000... (100 %)
Wrote 265616 bytes (195725 compressed) at 0x00000000 in 17.2 seconds (effective 123.2 kbit/s)...
Hash of data verified.

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

At the bottom right of the window, there is a notification box that says "Done uploading." with an information icon on the left and a close icon on the right.

## 6.5 Code analysis

Define three LED pins and operating modes

```
3  #define Green 13    //定义绿色灯引脚
4  #define Yellow 11   //定义黄色灯引脚
5  #define red 10      //定义绿色灯引脚
6
7  void setup() {
8      // 定义引脚为输出工作模式
9      pinMode(Green, OUTPUT);    //绿色灯
10     pinMode(Yellow, OUTPUT);    //黄色灯
11     pinMode(red, OUTPUT);       //红色灯
```

Within the loop function, each LED light is lit individually at intervals of 200 milliseconds.

```
13 void loop() {  
14     digitalWrite(Green, HIGH); //点亮绿色灯  
15     digitalWrite(Yellow, LOW); //熄灭黄色灯  
16     digitalWrite(red, LOW); //熄灭红色灯  
17     delay(200); //让绿灯亮200毫秒  
18  
19     digitalWrite(Green, LOW); //熄灭绿色灯  
20     digitalWrite(Yellow, HIGH); //点亮黄色灯  
21     digitalWrite(red, LOW); //熄灭红色灯  
22     delay(200); //让黄灯亮200毫秒  
23  
24     digitalWrite(Green, LOW); //熄灭绿色灯  
25     digitalWrite(Yellow, LOW); //熄灭黄色灯  
26     digitalWrite(red, HIGH); //点亮红色灯  
27     delay(200); //让红灯亮200毫秒  
28 }
```

## 7. WS2812B

### 7.1. Overview

This section focuses on understanding ws2812 and learning how to declare a library and instantiate objects through the library to light up RGB lights one by one in turn. We will use a library designed specifically for these sensors, which will keep our code short and easy to write. ( Please see the previous tutorial [for how to install the library](#) )

### 7.2. Working principle

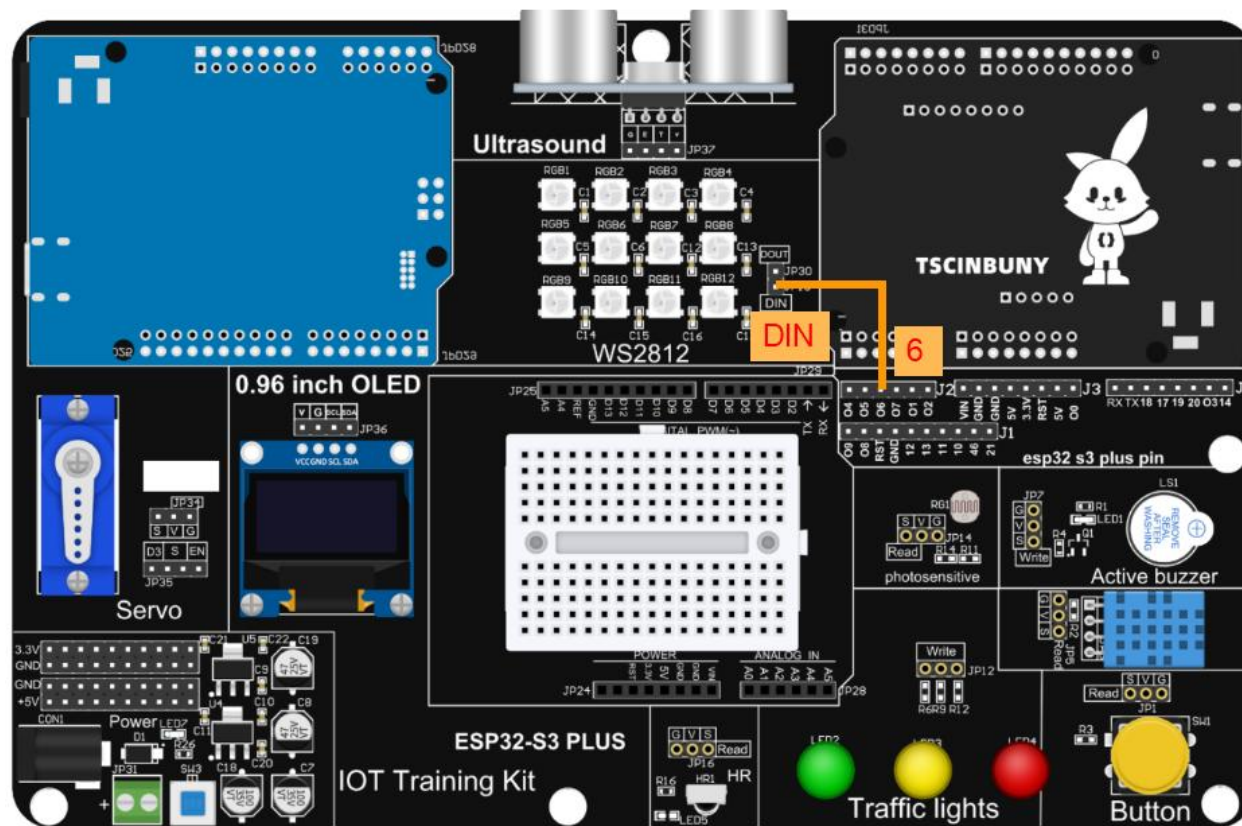


WS2812B is an intelligent externally controlled LED light source that integrates control circuit and light-emitting circuit. Its appearance is the same as a 5050LED lamp bead, and each component is a pixel. There is an intelligent digital interface data latch signal shaping amplification drive circuit, a high-precision internal oscillator and a 12V high-voltage programmable fixed current control part inside the pixel, which effectively ensures that the pixel light color is highly consistent.

The data protocol adopts single-line zero return code communication method. After the pixel is powered on and reset, the DIN client receives the data from the controller and first sends out the 24-bit data. After extracting the first pixel, it is sent to the pixel data latch. The remaining data is shaped and amplified by the internal shaping processing circuit. . It starts to be

forwarded to the next cascade pixel through the DO output port. After each pixel is transmitted, the signal is reduced by 24 bits. The pixels adopt automatic shaping and forwarding technology, so that the number of cascades of pixels is not limited by signal transmission, but only limited by the signal transmission speed requirements.

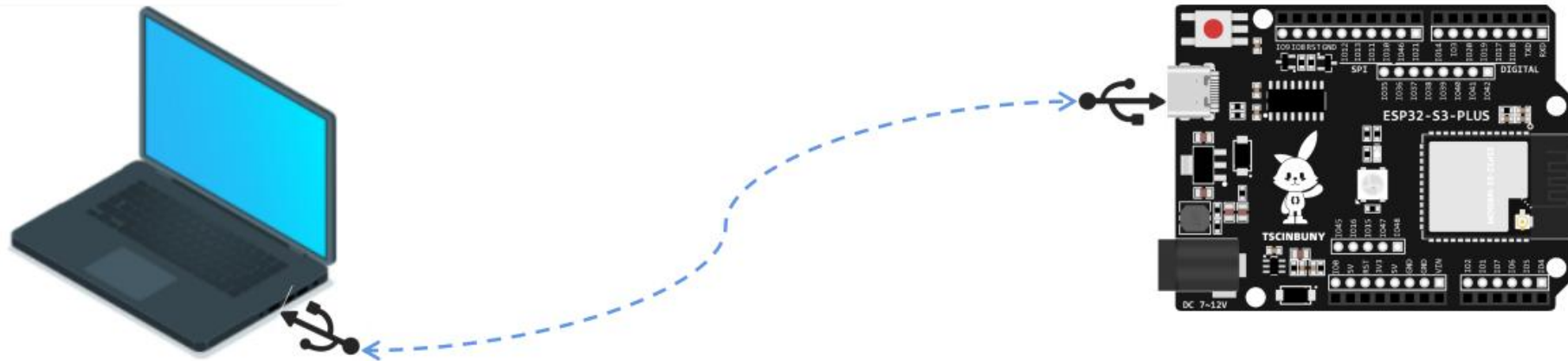
### 7.3. Connect the lines





## 7.4.Upload code

### 7.4.1. Connect the main control board to the computer using a USB cable

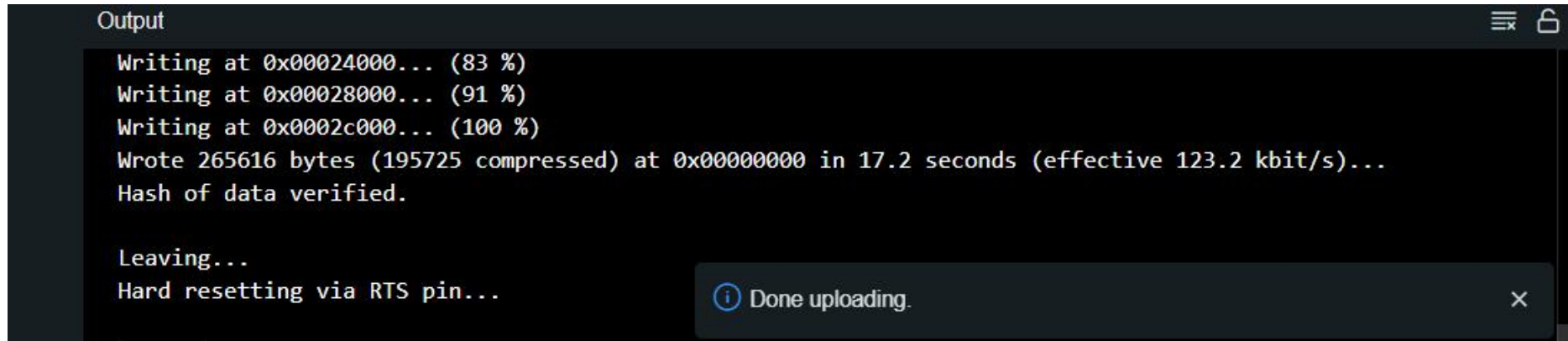


### 7.4.2 Open the program file (path: 2\_ESP32\_S3\_PLUS\ Lesson\_7\_WS2812B )

Lesson_1_Arduino IDE	2024/3/13 14:32	文件夹
Lesson_2_Light_up_the_LED	2024/3/13 13:43	文件夹
Lesson_3_Button_control_LED	2024/3/5 12:00	文件夹
Lesson_4_active_buzzer	2024/3/13 13:43	文件夹
Lesson_5_Traffic_light	2024/3/13 13:43	文件夹
Lesson_6_Flow_light	2024/3/13 13:43	文件夹
<b>Lesson_7_WS2812B</b>	2024/3/6 11:10	文件夹
Lesson_8_Gradient_RGB_light	2024/3/6 11:09	文件夹
Lesson_9_Steering_gear_control	2024/3/5 14:15	文件夹



Also select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board.



```

Output
Writing at 0x00024000... (83 %)
Writing at 0x00028000... (91 %)
Writing at 0x0002c000... (100 %)
Wrote 265616 bytes (195725 compressed) at 0x00000000 in 17.2 seconds (effective 123.2 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
Done uploading.
  
```

## 7.5. Code analysis

Declare the FastLED library, define the ws2812 pin as 6, the number of lamp beads as 12 and the brightness value as 64.

```

1  #include <FastLED.h>    //声明FastLED库 Declare the FastLED library
2
3  #define LED_PIN 6        //定义WS2812 RGB灯引脚 Define the WS2812 RGB lamp pin
4  #define NUM_LEDS 12     //定义灯珠数量 Define the number of beads
5  #define BRIGHTNESS 64   //设定灯光亮度, 范围0~255,数字越大越亮 Set the light level from 0 to 255,
  
```

Instantiate ws2812 as LEDs, set the NEOPIXEL type and initialize the light brightness.

```
6  CRGB leds[NUM_LEDS];    //实例化一个长度为 NUM_LEDS 的 CRGB 类型数组，用于控制
7  void setup() {
8      FastLED.addLeds<NEOPIXEL, LED_PIN>(leds, NUM_LEDS); //使用NEOPIXEL类型
9      FastLED.setBrightness(BRIGHTNESS);                //初始化灯光亮度
10 }
```

The loop function lights up the LEDs one by one and displays green

```
12 void loop() {
13     //逐个点亮LED显示绿色 Light the leds one by one to show green
14     for (int i = 0; i < NUM_LEDS; i++) {
15         fill_solid(leds, NUM_LEDS, CRGB::Black); //将所有LED关闭 Turn off all leds
16         leds[i] = CRGB::Green;                   //仅点亮当前LED Light only the current LED
17         FastLED.show();
18         delay(100);
19     }
20 }
```

## 8. Gradient RGB

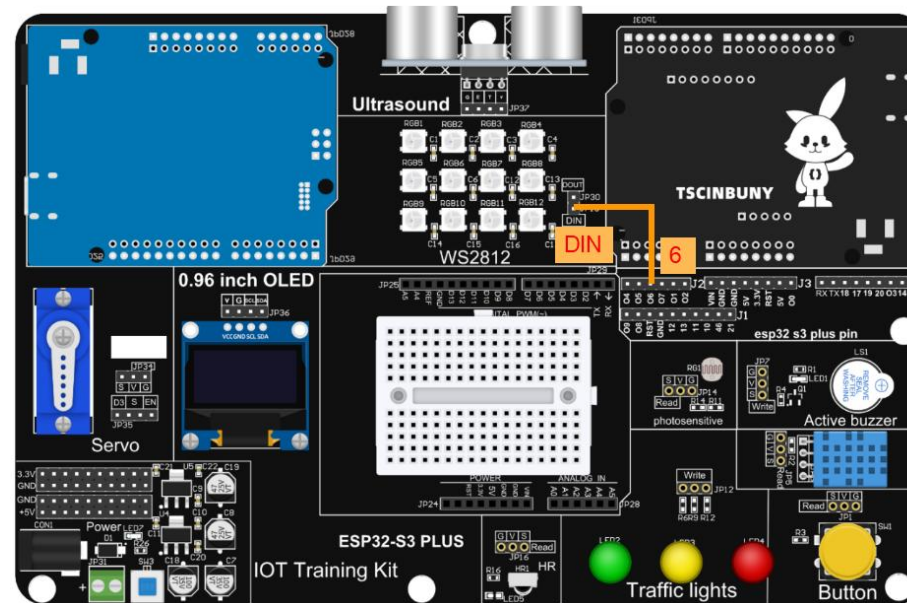
### 8.1. Overview

In this section , you will learn the control of WS2812 in an advanced way to achieve the effect of RGB light gradient .

### 8.2. Working principle

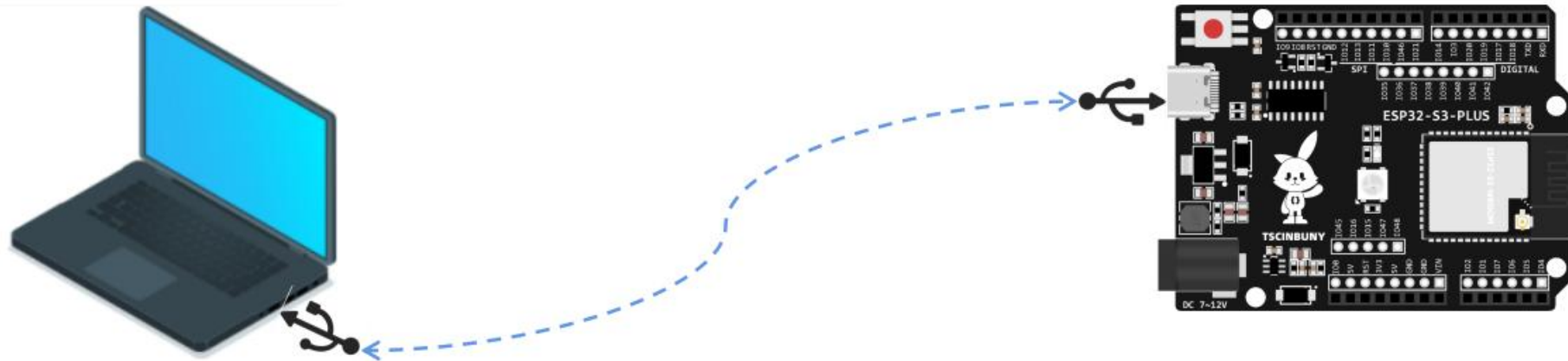
Use the library function `fill_rainbow(leds, NUM_LEDS, hue, 8);` to change the gradient display of RGB lights.

### 8.3 Connection lines



## 8.4. Upload code program

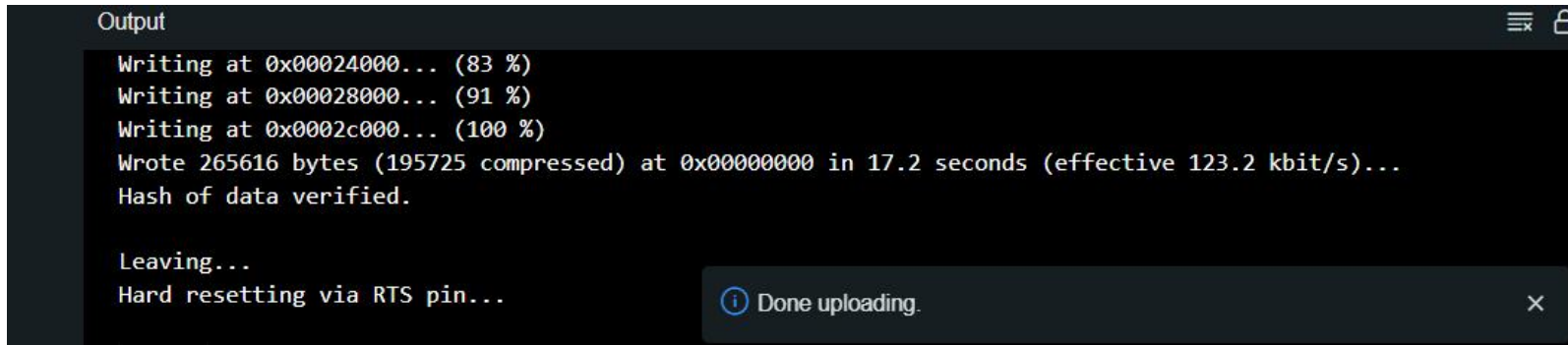
### 8.4.1. Connect the main control board to the computer using a USB cable



### 8.4.2 Open the program file (path: 2\_ESP32\_S3\_PLUS\Lesson\_8\_Gradient\_RGB\_light)

Lesson_1_Arduino IDE	2024/3/13 14:32	文件夹
Lesson_2_Light_up_the_LED	2024/3/13 13:43	文件夹
Lesson_3_Button_control_LED	2024/3/5 12:00	文件夹
Lesson_4_active_buzzer	2024/3/13 13:43	文件夹
Lesson_5_Traffic_light	2024/3/13 13:43	文件夹
Lesson_6_Flow_light	2024/3/13 13:43	文件夹
Lesson_7_WS2812B	2024/3/6 11:10	文件夹
<b>Lesson_8_Gradient_RGB_light</b>	2024/3/6 11:09	文件夹
Lesson_9_Steering_gear_control	2024/3/5 14:15	文件夹

Also select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board, and wait for the program upload to be completed .



```
Output
Writing at 0x00024000... (83 %)
Writing at 0x00028000... (91 %)
Writing at 0x0002c000... (100 %)
Wrote 265616 bytes (195725 compressed) at 0x00000000 in 17.2 seconds (effective 123.2 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
Done uploading.
```

## 8.5 Code analysis

The declaration library, pin definition and setting part of the code are consistent with the previous section, and then use the `fill_rainbow` function in the loop function to implement the rainbow color gradient.

```
14 void loop() {
15     //渐变显示不同颜色 Gradients display different colors
16     for (int hue = 0; hue < 255; hue++) {
17         fill_rainbow(leds, NUM_LEDS, hue, 8);
18         FastLED.show();
19         delay(20);
20     }
21 }
```

## 9.Servo control

### 9.1. Overview

In this section, you will learn how to drive the servo to achieve 0~180° rotation.

### 9.2 Working principle

#### 9.2.1. Steering gear



The steering gear ( servo motor ) control pulse signal period is a 20MS pulse width modulation signal (PWM), the pulse width is from 0.5ms to 2.5ms, and the corresponding steering position changes linearly from 0 to 180 degrees.

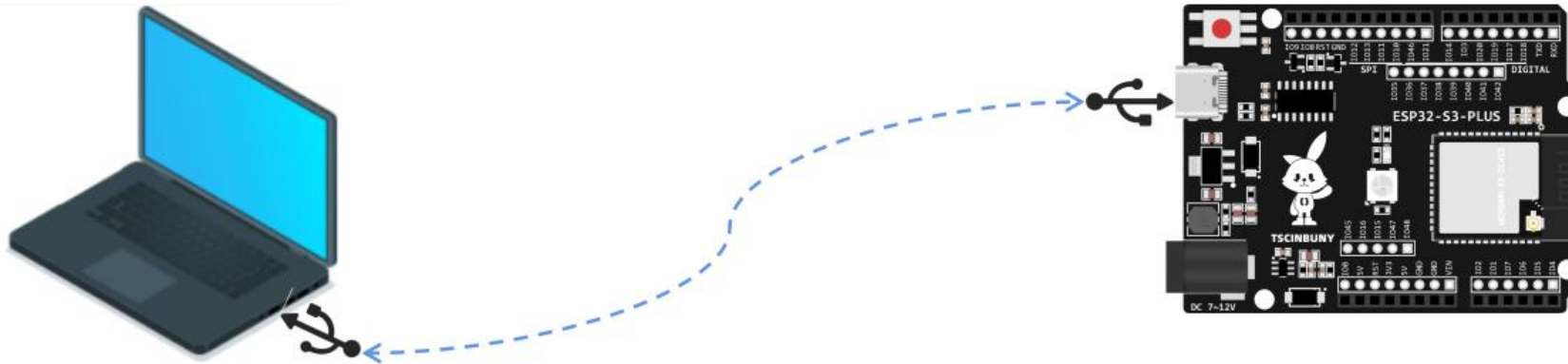
There is a reference circuit inside the steering gear, which generates a pulse signal with a period of 20ms and a width of 1.5ms. There is a comparator that compares the external signal with the reference signal to determine the direction and size, thereby generating a motor rotation signal.





## 9.4 Upload code program

### 9.4.1 Connect the main control SD board to the computer using a USB cable



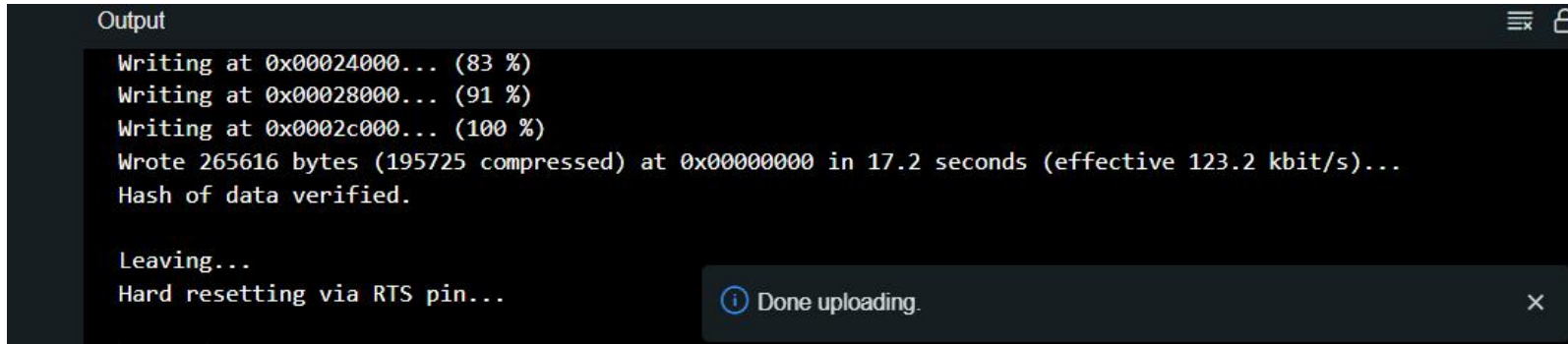
### 9.4.2 Open the program file (path: 2\_ESP32\_S3\_PLUS\Lesson\_9\_Steering\_gear\_control)

Lesson_7_WS2812B	2024/3/6 11:10	文件夹
Lesson_8_Gradient_RGB_light	2024/3/6 11:09	文件夹
<b>Lesson_9_Steering_gear_control</b>	2024/3/5 14:15	文件夹
Lesson_10_Photorresistor	2024/3/5 14:30	文件夹
Lesson11_Ultrasonic_ranging_OLED_display	2024/3/5 14:49	文件夹
Lesson12_DHT11_OLED_display	2024/3/6 11:44	文件夹
Lesson13_Infrared_change_RGB	2024/3/14 9:48	文件夹

Also select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board, and wait for the program



upload to be completed .



```

Output
Writing at 0x00024000... (83 %)
Writing at 0x00028000... (91 %)
Writing at 0x0002c000... (100 %)
Wrote 265616 bytes (195725 compressed) at 0x00000000 in 17.2 seconds (effective 123.2 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
Done uploading.
  
```

## 9.5 Code analysis

Declare the servo library Servo, define the servo pin A0 and instantiate the servo object myServo.

```

1  #include <Servo.h>
2  #define SERVO_PIN A0 //将舵机信号线连接到UNOR4 MINIMA的A0 Connect the servo s
3  Servo myServo;       //实例化一个舵机对象 Instantiate a steering gear object
  
```

Set the servo pin to output mode and initialize the servo.

```

5  void setup() {
6      pinMode(SERVO_PIN, OUTPUT); //将舵机连接的引脚设为输出模式 Set the servo c
7      myServo.attach(SERVO_PIN);  //设置舵机对象 Set the steering gear object
8  }
  
```

Let the servo rotate back and forth between 0~180° in the loop function

```
10 void loop() {  
11     //让舵机从180度转到0度 Let the steering gear turn from 180 to 0 degrees  
12     for (int angle = 180; angle >= 0; angle--) {  
13         myServo.write(angle);  
14         delay(10);  
15     }  
16  
17     //让舵机从0度转到180度 Let the steering gear turn from 0 to 180 degrees  
18     for (int angle = 0; angle <= 180; angle++) {  
19         myServo.write(angle);  
20         delay(10); //等待一小段时间，使舵机有足够的时间运动到下一个角度  
21     }
```

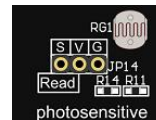
# 10. Photoresistor

## 10.1 Overview

In this section , you will learn how to use a light intensity detection module also called a photoresistor. The photoresistor detects the ambient light intensity and prints it to the serial monitor.

## 10.2 Working principle

### 10.2.1 Photoresistor

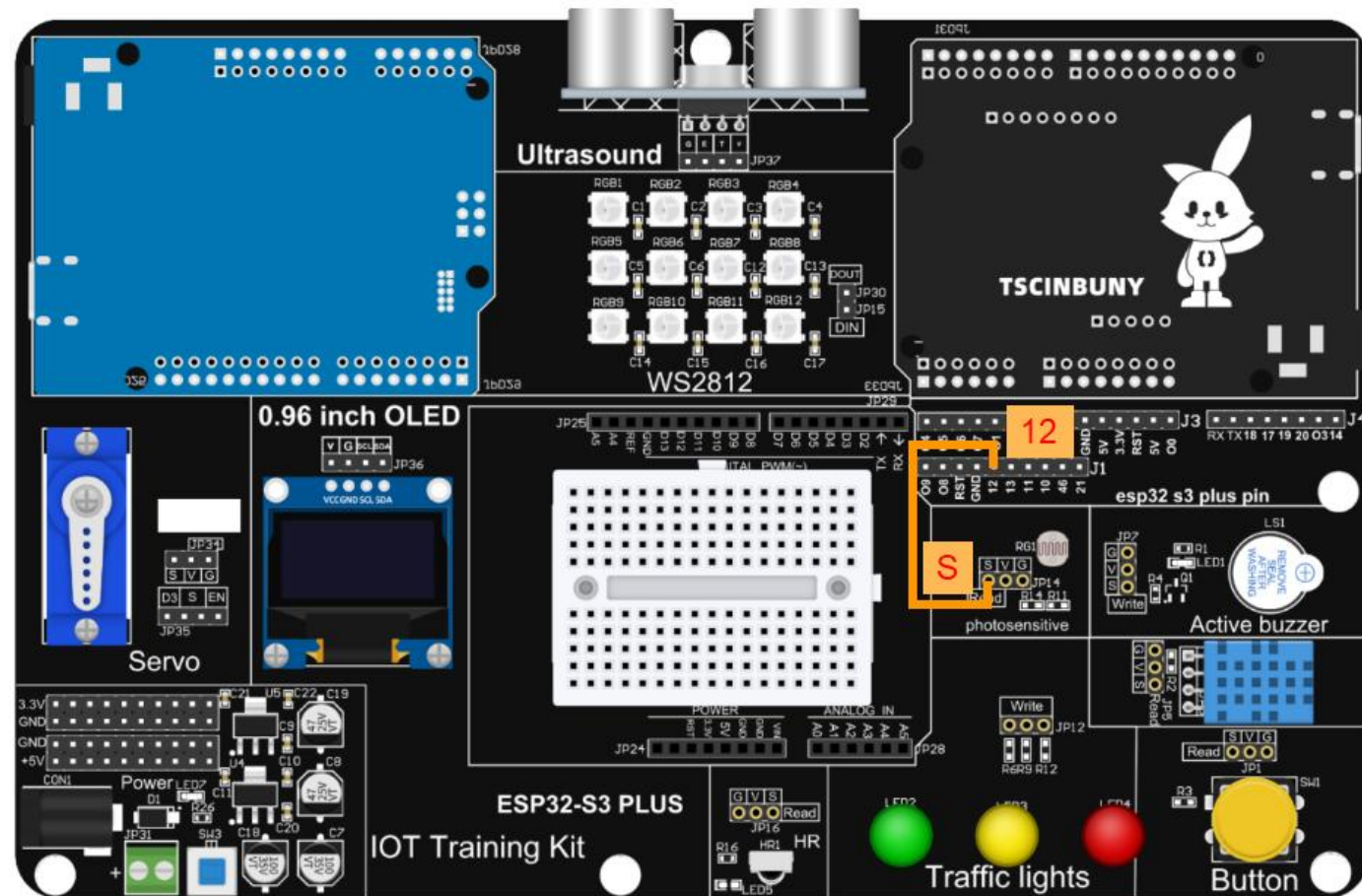


A photoresistor is a resistor made by utilizing the photoelectric effect of semiconductors. The resistance value changes with the intensity of incident light. It is also called a photodetector. When the incident light is strong, the resistance decreases. When the incident light is weak, the resistance increases. . There is another type where the resistance decreases when the incident light is weak and when the incident light is strong, the resistance increases. According to this characteristic, photoresistors with different shapes and irradiation areas can be manufactured.

Working principle: Since the carriers generated by illumination participate in conduction and drift under the action of the external electric field, the electrons rush to the positive electrode of the power supply and the holes rush to the negative electrode of the power supply, thus causing the resistance of the photoresistor to drop rapidly. Photoresistors are generally

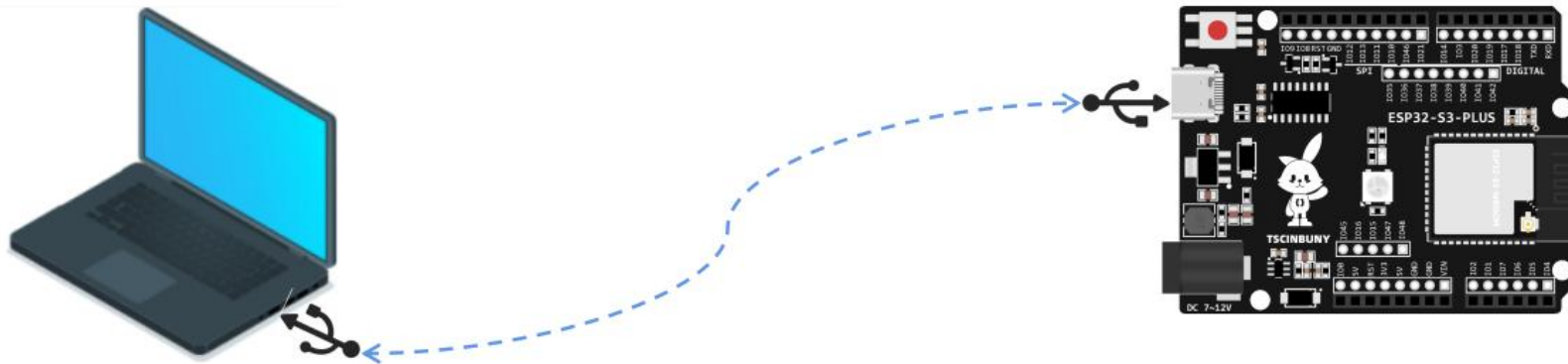
used for light measurement, light control and photoelectric conversion .

### 10.3 Connection lines



## 10.4 Upload code program

### 1 0 .4.1 Connect the main control board to the computer using a USB cable



### 1 0 .4.2 Open the program file (path: 2\_ESP32\_S3\_PLUS \ Lesson\_10\_Photoresistor )

Lesson_7_WS2812B	2024/3/6 11:10	文件夹
Lesson_8_Gradient_RGB_light	2024/3/6 11:09	文件夹
Lesson_9_Steering_gear_control	2024/3/5 14:15	文件夹
<b>Lesson_10_Photoresistor</b>	2024/3/5 14:30	文件夹
Lesson11_Ultrasonic_ranging_OLED_display	2024/3/5 14:49	文件夹
Lesson12_DHT11_OLED_display	2024/3/6 11:44	文件夹
Lesson13_Infrared_change_RGB	2024/3/14 9:48	文件夹

Also select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board, and wait for the program

upload to be completed .

After the program is uploaded, open the IDE serial port monitor and you can see the printed output light intensity value.

Under normal lighting, the value is a maximum of 4095



The detection value is  $<300$  when covering the photoresistor with objects.





## 10.5 Code analysis

The variable Lightvalue that defines the analog input pin analogInPin of the photoresistor detection module and the light intensity value.

```
2  const int analogInPin = 12; //光敏电阻模拟输入引脚 The photoresistor mimics the input pin
3  int Lightvalue = 0;         //定义变量 Defining variables
```

Set the photosensitive detection module pin as input, baud rate 9600

```
5  void setup() {
6      Serial.begin(9600);      //初始化串口波特率 Initialize the serial port baud rate
7      pinMode(analogInPin, INPUT);
8  }
```

The loop function first obtains the simulation value of the light detection module, saves it to a variable and prints it to the serial monitor.

```
10 void loop() {
11     Lightvalue = analogRead(analogInPin); //读取光敏度的值 Read the value of light sensitivity
12     Serial.print("Lightvalue = ");       //通过串口打印光敏度的值 Print the value of light ser
13     Serial.println(Lightvalue);
14     delay(200);
15 }
```



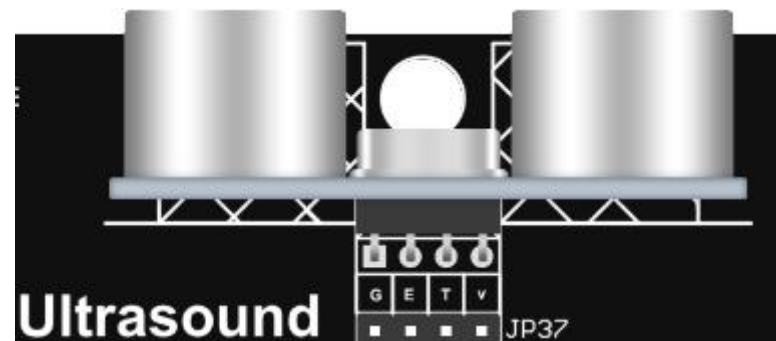
# 11. Ultrasonic ranging

## 11.1 Overview

In this section you will learn how to use the Ultrasonic Module. The ultrasonic sensor is used to measure distance and the value is displayed on the OLED screen in real time.

## 11.2 Working principle

### 11.2.1. Ultrasonic sensor



Sound waves are produced by vibrations and can travel at different speeds in different media. Ultrasonic waves have the advantages of strong directivity, slow energy loss, and long propagation distance in media, and are often used for distance measurement. For example, distance meters, liquid level measuring instruments, etc. can all be realized through ultrasonic waves.

Electrical parameters	HC-SR04 Ultrasonic module
Working voltage	DC-5V
Working current	15mA
Working frequency	40KHz
Maximum range	4m
Minimum range	2cm
Measuring angle	15 °
Input trigger signal	10 US TTL pulse
Output echo signal	Output TTL level signal, proportional to the range

Ultrasonic ranging is a non-contact detection method , especially used in airborne ranging. Because the wave speed in the air is slow, the echo signal contained along the direction of structural information propagation is easy to detect and has very high resolution, so it The accuracy is higher than other methods; the ultrasonic sensor has the characteristics of simple structure, small size, and reliable signal processing. The use of ultrasonic detection is often faster, more convenient, simpler to calculate, easier to achieve real-time control, and can meet industrial practical requirements in terms of measurement accuracy.

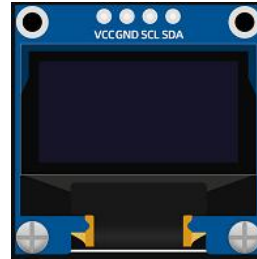
There are many methods of ultrasonic ranging. The principle of this system in ultrasonic measurement is: the trigger signal input terminal ( TRIG ) will input a high-level signal of more than 10 microseconds. After receiving the signal, the ultrasonic transmitter will automatically send 8 A 40Hz square wave. At the same time, the timer will start. When the sensor receives the echo, it stops timing and outputs the echo signal. Detect the ultrasonic wave from the ultrasonic transmitter, the transmission time through the gas medium to the receiver , multiply this time by the speed of sound in the gas, and get the distance of sound propagation. That is, the ultrasonic transmitter emits ultrasonic waves in a certain direction, and the MCU starts timing at the same time. The ultrasonic waves are launched in the air and return immediately when encountering obstacles on the way. The ultrasonic receiver stops timing immediately after receiving the reflected waves.

T recorded by the timer , the distance ( s ) from the launch point to the obstacle can be calculated .

$$\text{Formula: } S = VT/2$$

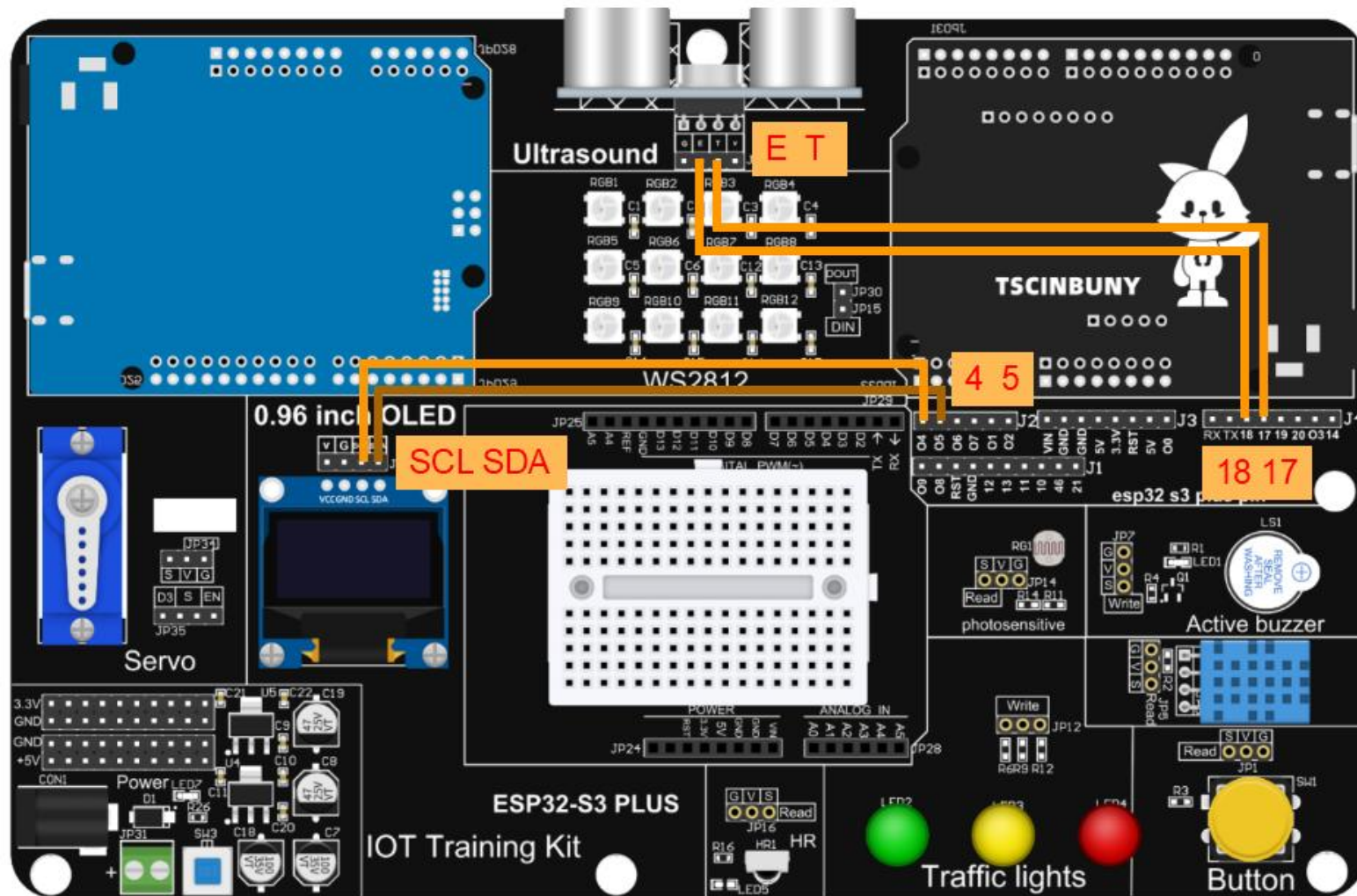
Four factors limit the maximum measurable distance of an ultrasound system: the amplitude of the ultrasound wave, the texture of the reflector, the angle between the reflected and incident sound waves, and the sensitivity of the receiving transducer. The ability of the receiving transducer to directly receive the acoustic pulse will determine the minimum measurable distance.

### 11.2.2.OLED



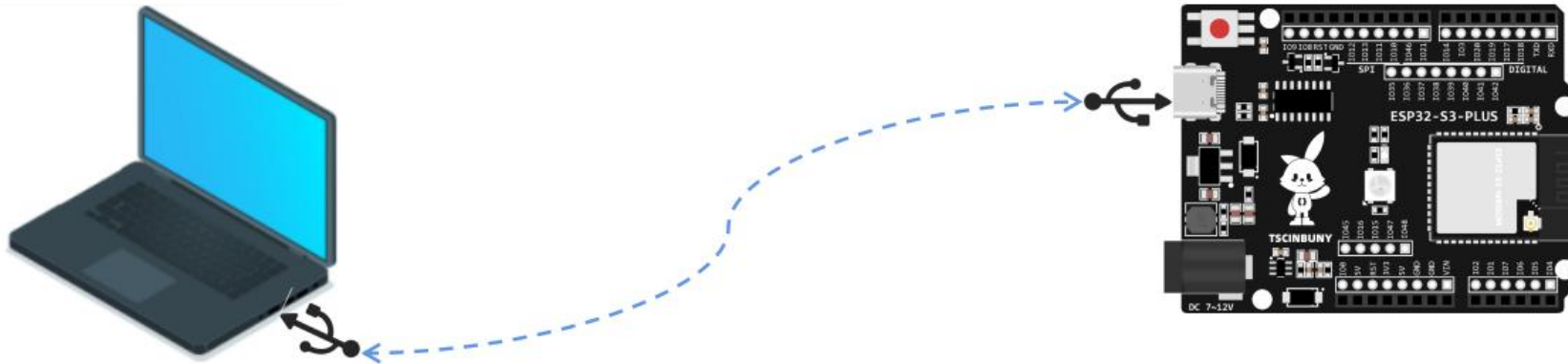
1. Resolution: 128\*64
2. Super wide viewing angle: greater than 160°
3. Communication method: IIC
4. Working voltage: 3.3V~5V

## 11.3 Connection lines



## 11.4 Upload code program

### 11.4.1 Connect the main control board to the computer with a USB cable



### 11.4.2 Open the program file ( path : 2\_ESP32\_S3\_PLUS \ Lesson\_11\_Ultrasonic\_ranging\_OLED\_display )

Lesson_7_WS2812B	2024/3/6 11:10	文件夹
Lesson_8_Gradient_RGB_light	2024/3/6 11:09	文件夹
Lesson_9_Steering_gear_control	2024/3/5 14:15	文件夹
Lesson_10_Photorresistor	2024/3/5 14:30	文件夹
Lesson_11_Ultrasonic_ranging_OLED_display	2024/3/21 18:04	文件夹
Lesson_12_DHT11_OLED_display	2024/3/21 18:05	文件夹
Lesson_13_Infrared_change_RGB	2024/3/21 18:05	文件夹

Also select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board, and wait for the program

upload to be completed .

```
Output
Writing at 0x00024000... (83 %)
Writing at 0x00028000... (91 %)
Writing at 0x0002c000... (100 %)
Wrote 265616 bytes (195725 compressed) at 0x00000000 in 17.2 seconds (effective 123.2 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
Done uploading.
```

After the program is uploaded, use an object to block the ultrasonic wave and see the measured distance value ( **when the screen does not display correctly, you need to check the SCL/SDA wiring, or press the motherboard reset button to reset the program** ).





## 11.5 Code analysis

Declare the required libraries. If the corresponding libraries are not added, please go back to [the installation library](#) to see how to add the libraries.

```
1  #include <Wire.h>
2  #include <Adafruit_GFX.h>
3  #include <Adafruit_SSD1306.h>
```

Define the pixel parameters of the OLED screen and instantiate a screen object display

```
5  #define SCREEN_WIDTH 128 // 定义屏幕宽度为128像素 Define the screen width to be 128 pixels
6  #define SCREEN_HEIGHT 64 // 定义屏幕高度为64像素 Define the screen height to be 64 pixels
7  #define OLED_RESET -1    // 定义 OLED 复位引脚为 -1（如果不使用复位功能，则设置为 -1） Define t
8  // 创建 Adafruit_SSD1306 对象，用于控制 OLED 屏幕，指定屏幕宽度、高度、I2C 总线对象和复位引脚
9  // Create an Adafruit_SSD1306 object that controls the OLED screen, specifying the screen wid
10 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
```

Define ultrasonic sensor pins and distance variables

```
12 #define Echo 18          //定义超声波Echo引脚 Define the ultrasonic Echo pin
13 #define Trig 17         //定义超声波Trig引脚 Define the ultrasonic Trig pin
14 int Udistance = 0;      //定义超声波距离变量 Define the ultrasonic distance variable
```

Set the ultrasonic pin Echo as input, Trig as output, initialize IIC bus and initialization screen, baud rate 9600

```

16 void setup() {
17     Serial.begin(9600);
18     pinMode(Echo, INPUT); //定义引脚工作模式为输入 Define the pin operation mode as the input
19     pinMode(Trig, OUTPUT); //定义引脚工作模式为输出 Define the pin operation mode as output
20     Wire.begin(); //初始化 I2C 总线 SDA引脚设置为A4, SCL引脚设置为A5 The initial I2C bus SDA pin is
21
22     // 初始化屏幕 Initializing the screen
23     if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
24         Serial.println(F("SSD1306 initialization failed"));
25         for (;;)
26             ;
27     }
28 }

```

### Ultrasonic acquisition distance function

```

47 float GetDistance() //获取超声波传感器数值 Get ultrasonic sensor values
48     float distance;
49     digitalWrite(Trig, LOW); //发送一个低电平到Trig触发测距 Sending a low level to the
50     delayMicroseconds(2);
51     digitalWrite(Trig, HIGH);
52     delayMicroseconds(10);
53     digitalWrite(Trig, LOW);
54     distance = pulseIn(Echo, HIGH) / 58.00; //输出距离转换 Output distance conversion
55     //delay(10);
56     return distance;

```

The loop function first obtains the distance value of the ultrasonic detection module and saves it to the variable Udistance. The value is output to the screen display through the display method of the screen object.

```
30 void loop() {  
31     Udistance = GetDistance(); // 获取超声波距离 Obtaining ultrasonic distance  
32     display.clearDisplay();    // 清空屏幕缓冲区 Clear the screen buffer  
33     //显示超声波距离 Display ultrasonic distance  
34     display.setTextSize(2);  
35     display.setTextColor(SSD1306_WHITE);  
36     display.setCursor(16, 8);  
37     display.println("Distance");  
38     display.setCursor(32, 40);  
39     display.print(Udistance);  
40     display.print(" cm");  
41     display.display();
```

## 12. DHT11 temperature and humidity sensor

### 12.1. Overview

In this tutorial, we will learn how to use the DHT11 temperature and humidity sensor. It's accurate enough for most projects where you need to detect humidity and temperature readings. Again, we will use a library specifically designed for these sensors, which will keep our code short and easy to write.

### 12.2. Working principle

#### 12.2.1. DHT11 temperature and humidity sensor

**humidity:**

**Resolution:** 16Bit

**Resolution:**  $\pm 1\%$  RH

**Accuracy:**  $\pm 5\%$  RH at  $25^{\circ}\text{C}$

**Interchangeability:** Interchangeable

**Response time:** 6S under  $1/e(63\%)$   $25^{\circ}\text{C}$ , 1m/s air condition

**Hysteresis:**  $< \pm 0.3\%$  RH

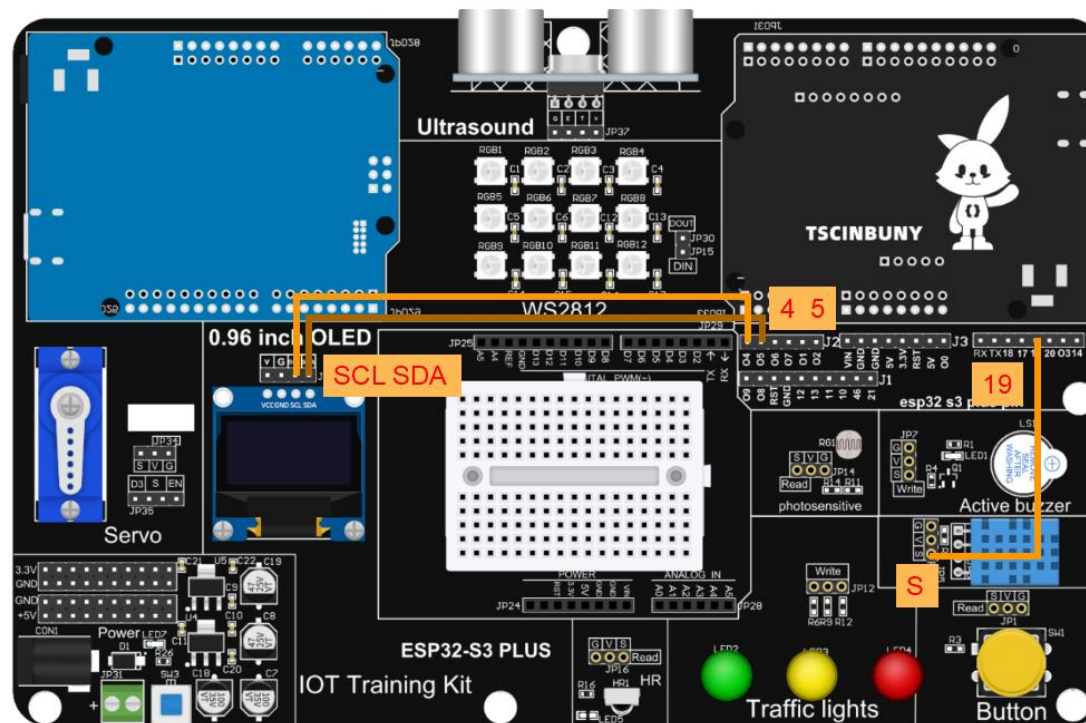
**Long-term stability:**  $< \pm 0.5\%$  RH/year



temperature:

<b>Resolution:</b>	$\pm 2^{\circ}\text{C}$	<b>Lag:</b>	$< \pm 0.3\% \text{ RH}$
<b>Repeatability:</b>	$\pm 0.2^{\circ}\text{C}$	<b>Response time:</b>	10S at 1/e (63%) condition
<b>Interchangeability:</b>	Interchangeable		

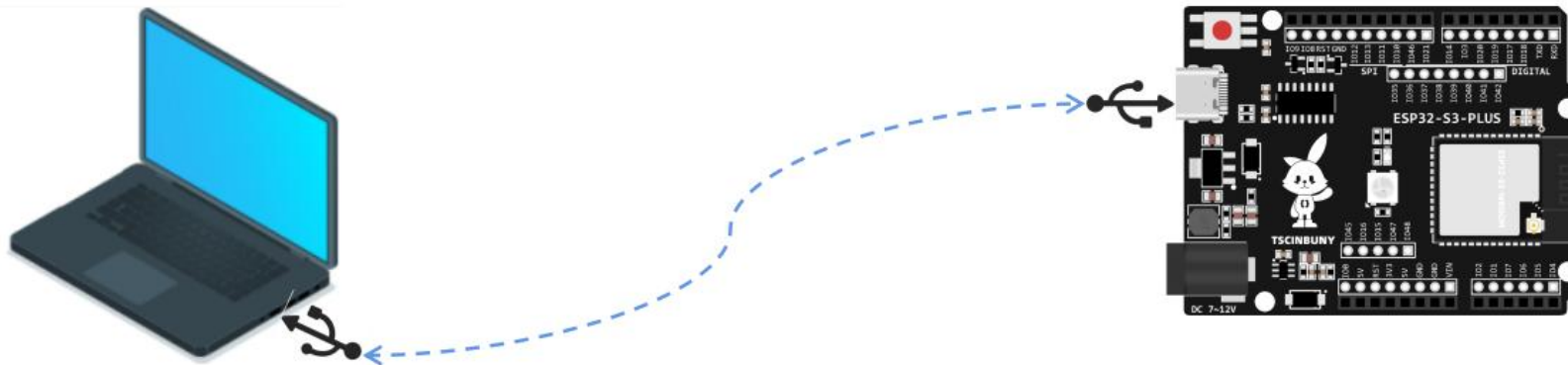
## 12.3 Connection lines





## 12.4 Upload code program

### 12.4.1 Connect the main control board to the computer using a USB cable



### 12.4.2 Open the program file (path: 2\_ESP32\_S3\_PLUS\ Lesson \_ 12\_DHT11\_OLED\_display )

Lesson_7_WS2812B	2024/3/6 11:10	文件夹
Lesson_8_Gradient_RGB_light	2024/3/6 11:09	文件夹
Lesson_9_Steering_gear_control	2024/3/5 14:15	文件夹
Lesson_10_Photorresistor	2024/3/5 14:30	文件夹
Lesson_11_Ultrasonic_ranging_OLED_display	2024/3/21 18:04	文件夹
Lesson_12_DHT11_OLED_display	2024/3/21 18:05	文件夹
Lesson_13_Infrared_change_RGB	2024/3/21 18:05	文件夹

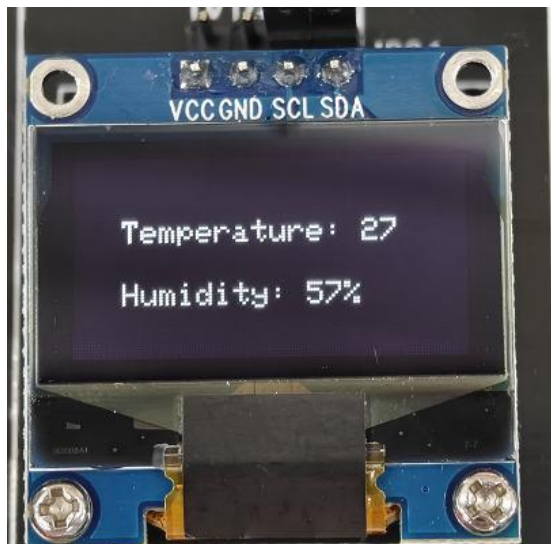
Also select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board, and wait for the program

upload to be completed .

```
Output
Writing at 0x00024000... (83 %)
Writing at 0x00028000... (91 %)
Writing at 0x0002c000... (100 %)
Wrote 265616 bytes (195725 compressed) at 0x00000000 in 17.2 seconds (effective 123.2 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
Done uploading.
```

After the program is uploaded, you will see the temperature and humidity values ( when the screen does not display correctly, you need to check the SCL/SDA wiring, or press the motherboard reset button to reset the program ).





## 12.5 Code analysis

Declare the required library files. If the corresponding library is not added, please go back to [the installation library](#) to see how to add the library.

```
1  #include <Wire.h>
2  #include <Adafruit_GFX.h>
3  #include <Adafruit_SSD1306.h>
4  #include <DHT.h>
```

Define the pixel parameters of the OLED screen and instantiate a screen object display

```
5  #define SCREEN_WIDTH 128 // 定义屏幕宽度为128像素 Define the screen width to be 128 pixels
6  #define SCREEN_HEIGHT 64 // 定义屏幕高度为64像素 Define the screen height to be 64 pixels
7  #define OLED_RESET -1 // 定义 OLED 复位引脚为 -1 (如果不使用复位功能, 则设置为 -1) Define t
8  // 创建 Adafruit_SSD1306 对象, 用于控制 OLED 屏幕, 指定屏幕宽度、高度、I2C 总线对象和复位引脚
9  // Create an Adafruit_SSD1306 object that controls the OLED screen, specifying the screen wid
10 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
```

Define DHT11 sensor pins and instantiate DHT objects as dht

```
13 #define DHTPIN 19 // DHT11传感器连接到Arduino的数字引脚4 The DH
14 #define DHTTYPE DHT11 // 使用DHT11传感器 The DHT11 sensor was used
15 DHT dht(DHTPIN, DHTTYPE); // 创建 DHT 对象, 用于连接 DHT 传感器, 指定传感
```

Set up the initialization bus, OLED screen and DHT11

```

17 void setup() {
18     Serial.begin(9600);
19     Wire.begin(); //初始化 I2C 总线 SDA引脚设置为A4, SCL引脚设置为A5 The initial I2C bus SDA pin is set
20
21     // 初始化 OLED 屏幕 Initializing the screen
22     if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
23         Serial.println(F("SSD1306 initialization failed"));
24         for (;;)
25             ;
26     }
27     display.clearDisplay(); // 清空屏幕缓冲区 Clear the screen buffer
28     dht.begin();           // 初始化 DHT11 传感器 Initialize the DHT11 sensor

```

Read the temperature and humidity values in the loop function, save them to variables, and then display them on the OLED screen.

```

31 void loop() {
32     // 读取温湿度数据 Read the temperature and humidity data
33     int humidity = dht.readHumidity();
34     int temperature = dht.readTemperature();
35     display.clearDisplay(); // 清空屏幕缓冲区 Clear the screen buffer
36     // 显示温湿度数据 Display temperature and humidity data
37     display.setTextSize(1);
38     display.setTextColor(SSD1306_WHITE);
39     display.setCursor(16, 20);
40     display.print("Temperature: ");
41     display.println(temperature);
42     display.setCursor(16, 40);
43     display.print("Humidity: ");
44     display.print(humidity);
45     display.println("%");
46     display.display(); // 更新显示
47     delay(2000); // 延迟2秒再进行下一次读取

```

## 13. Infrared remote control RGB

### 13.1 Overview

In this section , you will learn how to use an IR remote control with an IR receiver . Complete a project to switch RGB light colors via infrared remote control.

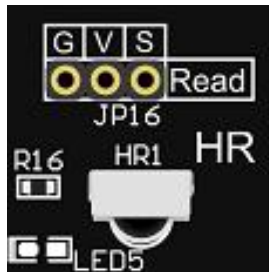
### 13.2 Working principle

The universal infrared remote control system consists of two parts: sending and receiving. The sending part is composed of infrared remote control, and the receiving part is composed of infrared receiving tube. The signal sent by the infrared remote control is a series of binary pulse codes. In order to avoid interference from other infrared signals during wireless transmission, it is generally necessary to modulate at a given carrier frequency and then transmit through an infrared emitting phototransistor. The infrared receiving tube filters out other noise waves, receives only the signal of a given frequency, and restores it to a demodulated binary pulse code. The built-in receiving tube converts the light signal sent by the infrared light-emitting diode, amplifies the signal through the amplifier in the IC, and restores the original code sent by the remote control through automatic gain control, band-pass filtering, demodulation, and wave formation, and outputs the signal through the infrared receiving module Pins identify the circuits that enter an appliance.

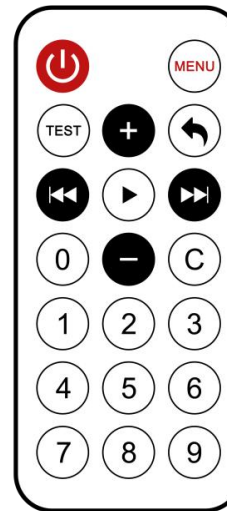
The encoding scheme that matches the infrared remote control protocol is: NEC protocol. Next, let us understand what the NEC protocol is.

- (1) 8 address bits, 8 sequence bits address bits and sequence bits are transmitted twice to ensure reliability
- (3) Pulse position modulation
- (4) The carrier frequency is 38 kHz
- (5) The time for each bit is 1.125 ms or 2.25 ms

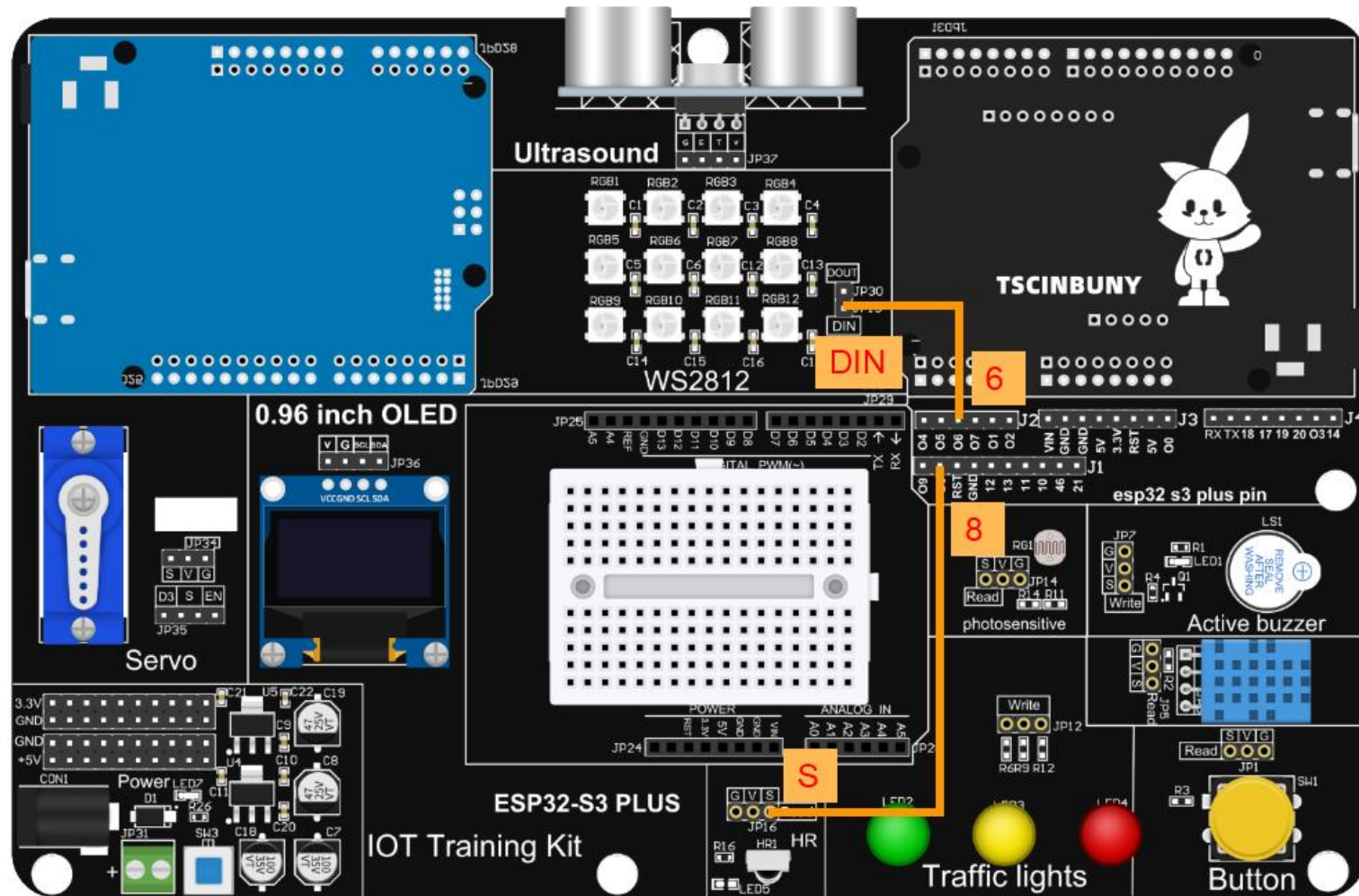
#### infrared receiver



#### infrared transmitter



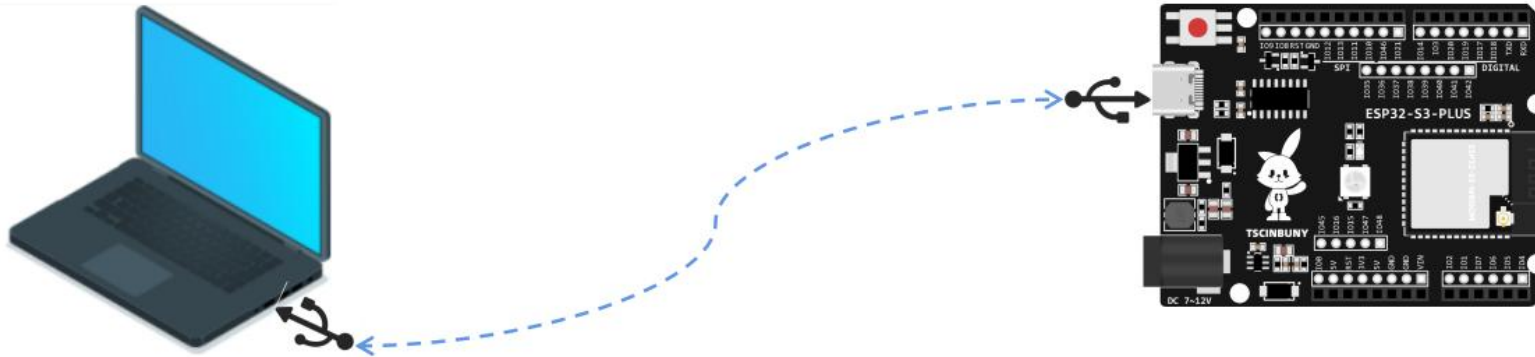
### 13.3 Connection lines





## 13.4 Upload code program

### 1 3 .4.1 Connect the main control board to the computer using a USB cable



### 1 3 .4.2 Open the program file ( Path : 2\_ESP32\_S3\_PLUS \ Lesson\_13\_Infrared\_change\_RGB )

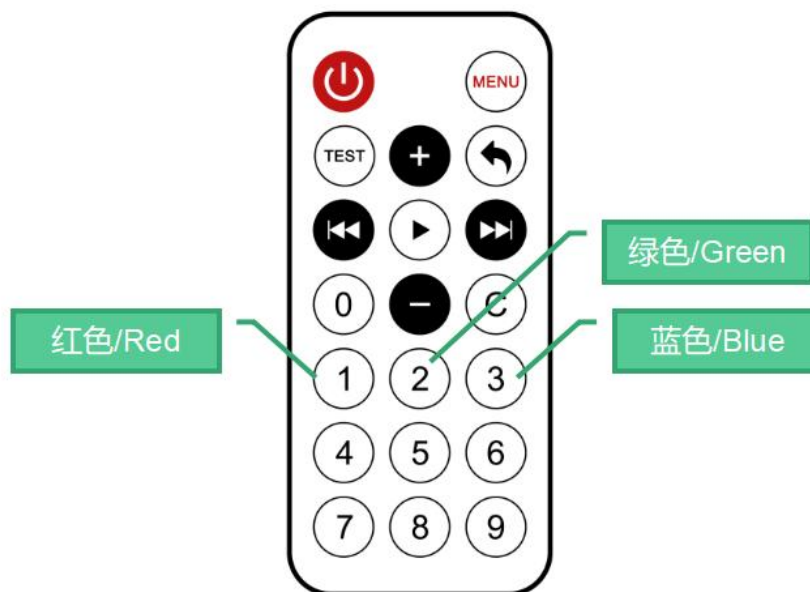
Lesson_7_WS2812B	2024/3/6 11:10	文件夹
Lesson_8_Gradient_RGB_light	2024/3/6 11:09	文件夹
Lesson_9_Steering_gear_control	2024/3/5 14:15	文件夹
Lesson_10_Photorresistor	2024/3/5 14:30	文件夹
Lesson_11_Ultrasonic_ranging_OLED_display	2024/3/21 18:04	文件夹
Lesson_12_DHT11_OLED_display	2024/3/21 18:05	文件夹
<b>Lesson_13_Infrared_change_RGB</b>	2024/3/21 18:05	文件夹

Also select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board, and wait for the program upload to be completed .

```
Output
Writing at 0x00024000... (83 %)
Writing at 0x00028000... (91 %)
Writing at 0x0002c000... (100 %)
Wrote 265616 bytes (195725 compressed) at 0x00000000 in 17.2 seconds (effective 123.2 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
Done uploading.
```

After the program is uploaded, point the infrared remote control at the infrared receiver and press the number keys 1/2/3 to control the WS2812 to switch lights of different colors.





## 13.5 Code analysis

Declare the required library files. If the corresponding library is not added, please go back to [the installation library](#) to see how to add the library.

```
1  #include <IRremoteESP8266.h>
2  #include <IRrecv.h>
3  #include <IRutils.h>
4  #include <FastLED.h>
```

Define ws2812 pins, number of lamp beads, brightness and other parameters and variables hue, Color

```
10  #define LED_PIN 7           //定义WS2812B RGB灯引脚 Define the WS2812B RGB lamp pin
11  #define NUM_LEDS 12        //定义灯珠数量 Define the number of beads
12  #define BRIGHTNESS 5       //定义灯光亮度 Defining the brightness of the light
13  #define FRAMES_PER_SECOND 120 //定义每秒帧数 Defines frames per second
14  CRGB leds[NUM_LEDS];       //定义一个长度为 NUM_LEDS 的 CRGB 类型数组，用于控制 WS2812B
15  static uint8_t hue = 0;     //设置颜色的值 Sets the value of the color
16  int Color = 1;             //定义灯光颜色模式的变量 A variable that defines the color mode
```

Define the infrared receiver pin as 8, instantiate the infrared receiver object as irReceiver, and receive the infrared information object pointer results.

```
14  const int IR_RECEIVE_PIN = 8; //红外接收器连接的GPIO8引脚 Infrared receiver pin
15  IRrecv irReceiver(IR_RECEIVE_PIN);
16  decode_results results;
```

Initialize the infrared receiver and ws2812, set the baud rate to 9600

```

18 void setup() {
19     Serial.begin(9600);
20     irReceiver.enableIRIn();           //启用红外接收功能 Enable infrared receiver function
21     FastLED.addLeds<WS2812B, LED_PIN, GRB>(leds, NUM_LEDS); //使用FastLED库将LED条上的LED添加到控制器中 The LED
22     FastLED.setBrightness(BRIGHTNESS); //设置灯光亮度 Set the light brightness

```

The loop function determines whether an infrared signal is received, and compares the received infrared signal code with the three instruction code values to obtain three color modes.

```

26 void loop() {
27     if (irReceiver.decode(&results)) { // 如果接收到红外信号 If you receive an infrared signal
28         // Serial.println(results.value, HEX); // 打印红外信号的数值 Print the numerical value of the infrared signal
29
30         if (results.value == 0xFF30CF) {
31             Color = 1; //如果按下按键1, 颜色模式为红 If key 1 is pressed, the color mode is red
32         }else if(results.value == 0xFF18E7){
33             Color = 2; //如果按下按键2, 颜色模式为绿 If key 1 is pressed, the color mode is green
34         }else if(results.value == 0xFF7A85){
35             Color = 3; //如果按下按键3, 颜色模式为蓝 If key 1 is pressed, the color mode is blue
36         }
37         irReceiver.resume(); // 继续接收下一个信号 Continue to receive the next signal
38     }

```

Three color modes are obtained according to the received infrared encoding information, and the corresponding color can be obtained by judging the value of the color mode variable Color. Let ws2812 display this color again.

```
42   for (int i = 0; i < NUM_LEDS; i++) {  
43       // 根据颜色值设置LED的颜色  
44       if (Color == 1) {  
45           leds[i] = CHSV(hue, 255, 255);           // 设置为红色 set it to red  
46       } else if (Color == 2) {  
47           leds[i] = CHSV(hue + 85, 255, 255);      // 设置为绿色 set it to green  
48       } else if (Color == 3) {  
49           leds[i] = CHSV(hue + 170, 255, 255);     // 设置为蓝色 set it to blue  
50       }  
51       FastLED.show(); // 显示LED的颜色  
52       FastLED.delay(2 / FRAMES_PER_SECOND); // 延迟以控制帧率  
53   }
```

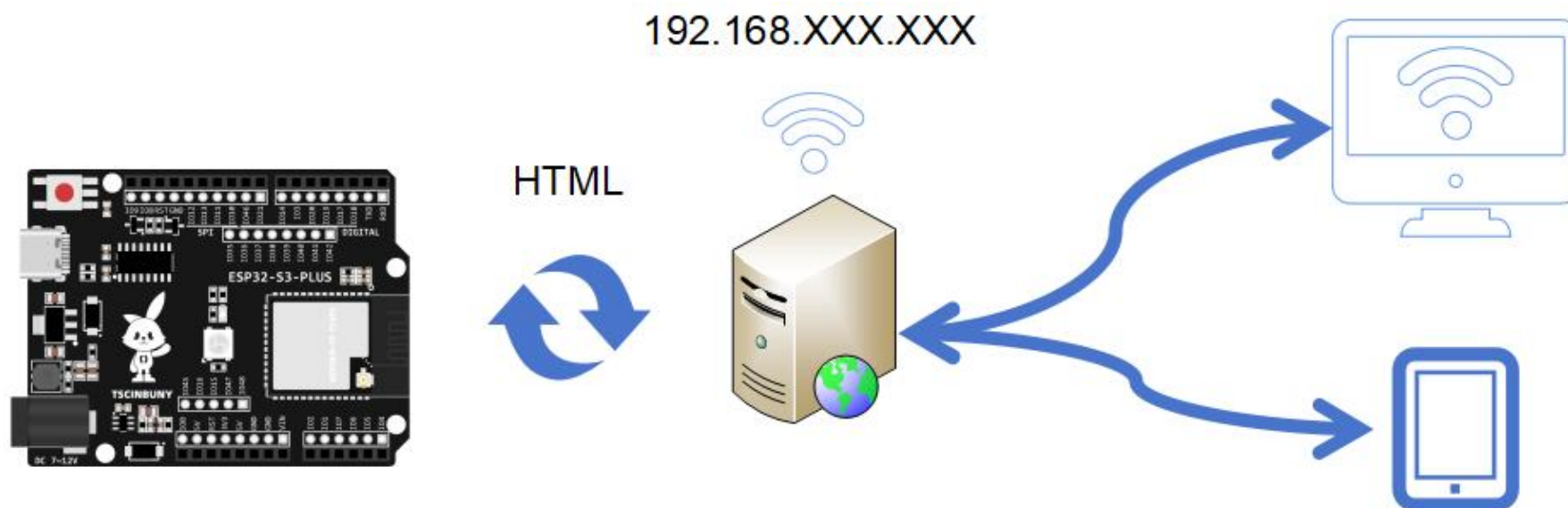
## 14. Web switch control LED

### 14.1 Overview

This section focuses on learning the WiFi function of ESP32S3 and controlling the LED on the web page.

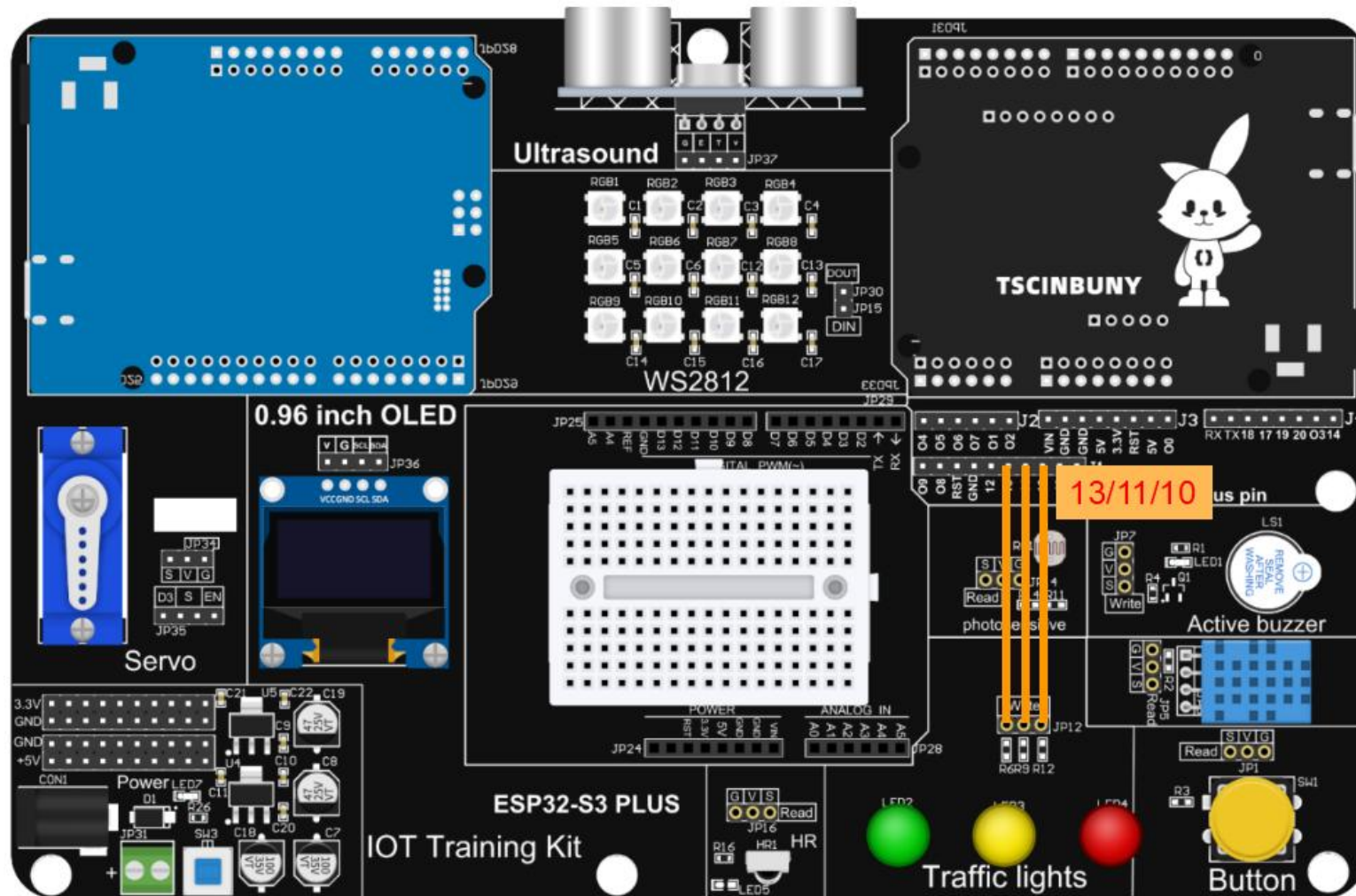
### 14.2 Working principle

Connect to a WiFi hotspot with smooth network through ESP32, and then use other devices such as computers or mobile phones to connect to the same WiFi. Enter the IP in the browser address bar to open the web interface created by ESP32, and control it through the web page containing the LED switch. LED.



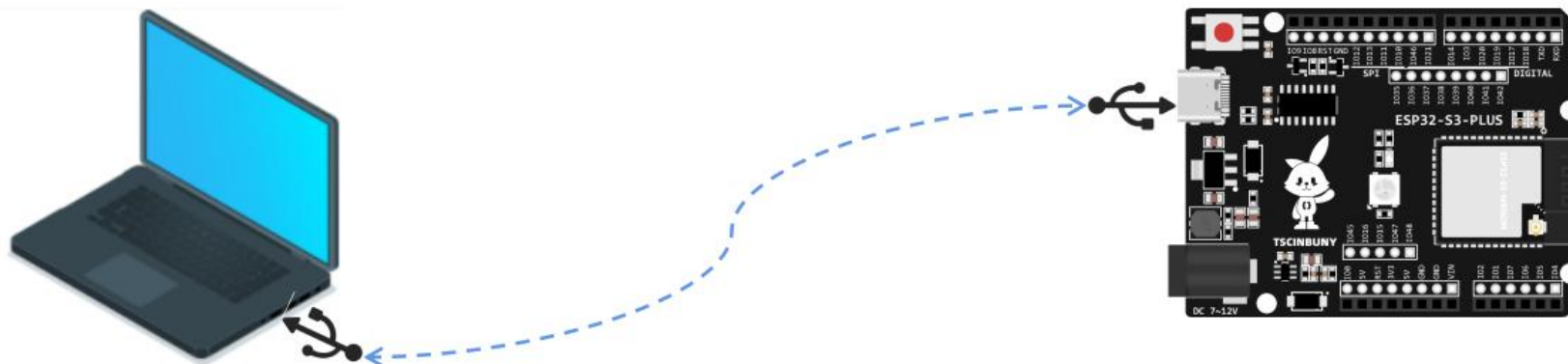


## 14.3 Connection lines



## 14.4 Upload code program

### 1 4 .4.1 Connect the main control board to the computer using a USB cable



### 1 4 .4.2 Open the program file (path: 2\_ESP32\_S3\_PLUS \ Lesson\_14\_Web\_key\_controls\_LED )

Lesson_14_Web_key_controls_LED	2024/3/27 11:23	文件夹
Lesson_15_Web_server_control_LED	2024/3/27 11:24	文件夹
Lesson_16_Web_server_control_RGB	2024/3/21 18:25	文件夹
Lesson_17_Web_Display_temperature_and_humidity	2024/3/21 18:25	文件夹
Lesson_18_Web_server_control_buzzer_alarm	2024/3/21 18:23	文件夹
Lesson_19_Photoresistor_controls_LED	2024/3/21 18:24	文件夹
Lesson_20_Web_control_steering_gear_Angle_display	2024/3/21 18:26	文件夹
Lesson_21_Web_control_ultrasonic_ranging_display	2024/3/21 18:26	文件夹

Modify the WiFi account and password to which ESP32 is connected in the code. (This WiFi can be the router WiFi at home)

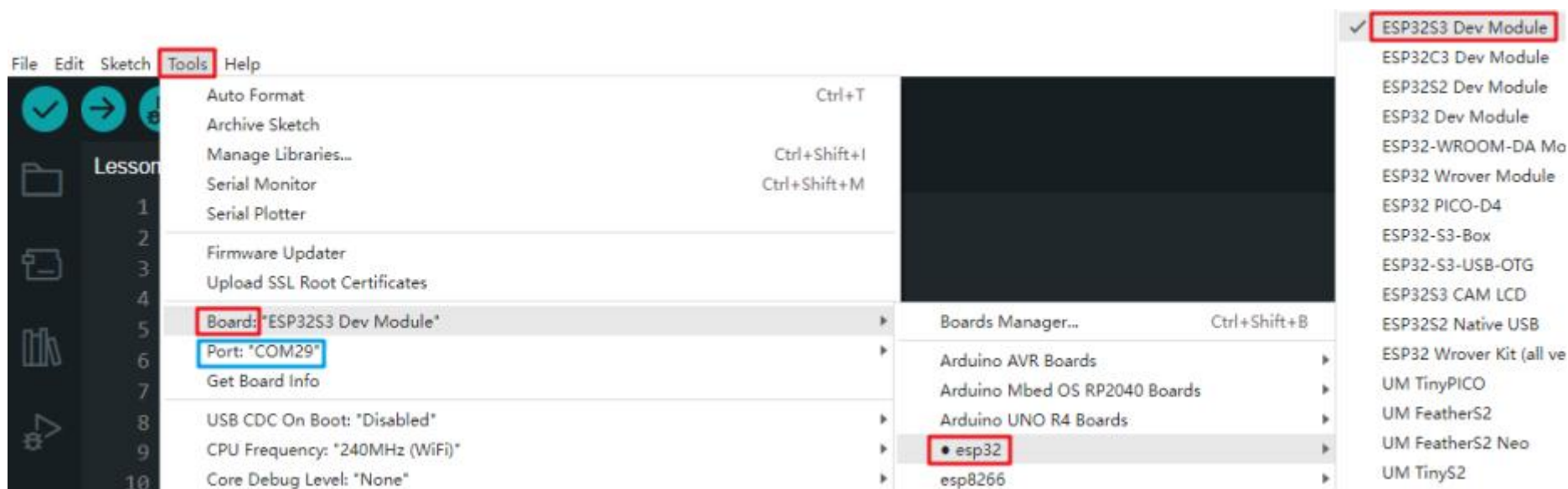
or the WiFi transmitted by the mobile hotspot, but make sure it is 2.4G and not 5G)

```

6  const char *ssid = "xxxxxx"; //输入ESP32要连接的WiFi账号 Enter the
7  const char *password = "xxxxxxxx"; //设置WiFi密码 Set WiFi password

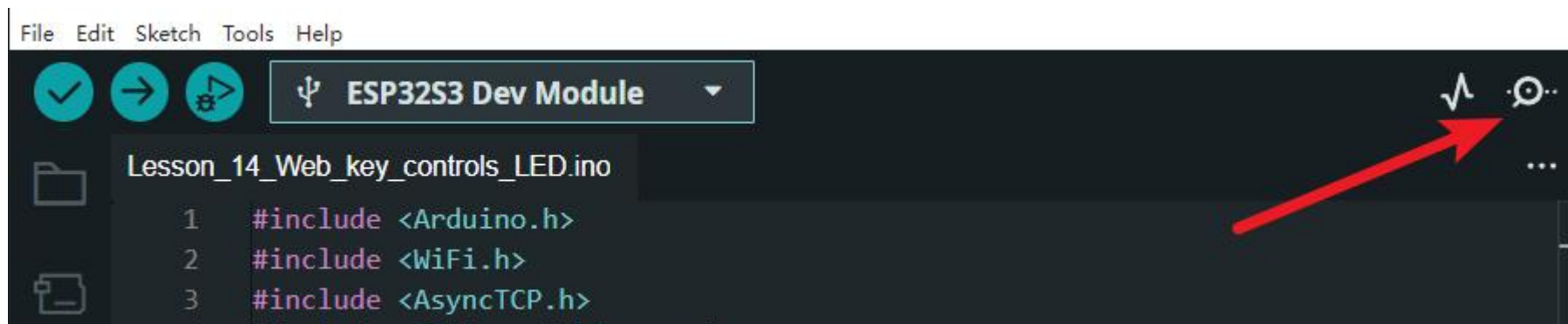
```

Then select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board, and wait for the program upload to be completed .

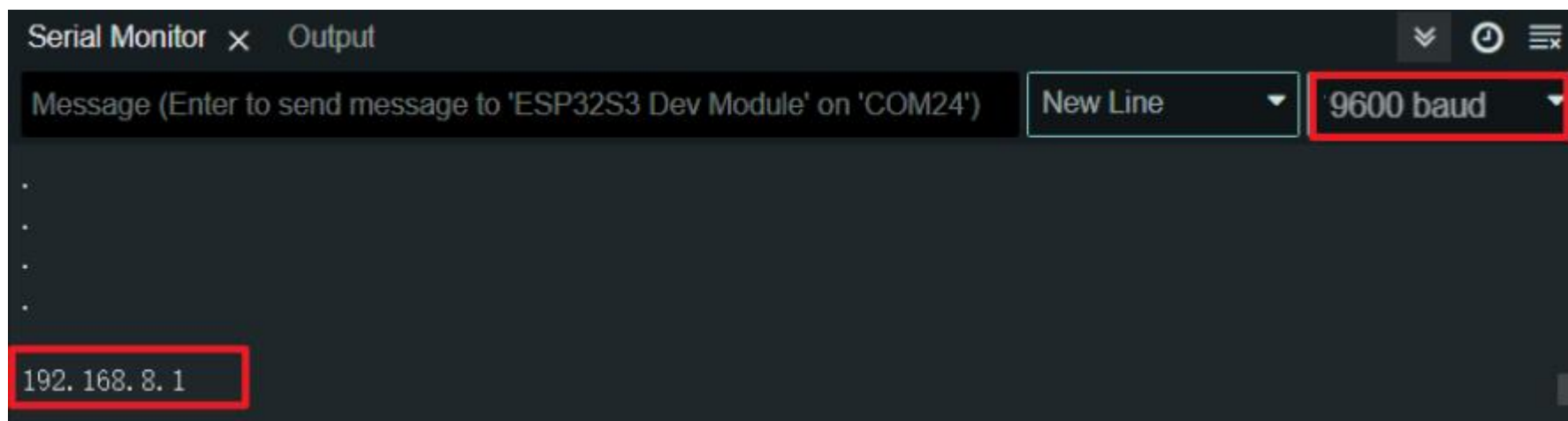




After the program upload is completed, open the serial monitor to view the IP address.



Here the IP is 192.168.8.1, but everyone will get a different IP.



Connect to the same WiFi with your mobile phone, then open the browser, enter the IP address above and enter the page.

Click the icon to control the corresponding LED switch.



## ESP32 LED Control



## 14.5 Code analysis

Declare the required libraries primarily for creating web services and handling asynchronous event-driven models

```
1  #include <Arduino.h>
2  #include <WiFi.h>
3  #include <AsyncTCP.h>
4  #include <ESPAsyncWebServer.h>
```

Initialize the WiFi to be connected to ESP32, set the three LED pins and the initial state is off

```
6  const char *ssid = "XXXXXX";      //输入ESP32要连接的WiFi账号 Enter the WiFi account to which you
7  const char *password = "XXXXXXXX"; //设置WiFi密码 Set WiFi password
8
9  const int greenLEDPin = 13;        //绿色LED控制引脚 Green LED control pin
10 const int yellowLEDPin = 11;       //黄色LED控制引脚 Yellow LED control pin
11 const int redLEDPin = 10;          //红色LED控制引脚 Red LED control pin
12
13 int greenLEDState = LOW;            //绿色LED初始状态（关闭） Green LED initial state (off)
14 int yellowLEDState = LOW;          //黄色LED初始状态（关闭） Yellow LED initial state (off)
15 int redLEDState = LOW;              //红色LED初始状态（关闭） Red LED initial state (off)
```

Create AsyncWebServer object

```
17 // 创建AsyncWebServer对象 Create an AsyncWebServer object
18 AsyncWebServer server(80);
```

Generate the HTML code of the web page, in which the web page icon is obtained from an external link, so the connected WiFi must be a valid WiFi to obtain the icon normally.

```

20 // 生成网页的HTML代码 Generates the HTML code for the web page
21 const char index_html[] PROGMEM = R"rawliteral(
22 <!DOCTYPE HTML><html>
23 <head>
24   <meta name="viewport" content="width=device-width, initial-scale=1">
25   <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.2/css/all.css" integrity="sha
26   <style>
27     html {
28       font-family: Arial;
29       display: inline-block;
30       margin: 0px auto;
31       text-align: center;
32     }
33     h2 { font-size: 3.0rem; }

```

The GetLEDState function implements web page update operations. When the icon is clicked, it updates the icon status accordingly and issues instructions to the ESP32.

```

90 String getLEDState() {
91   String json;
92   json += "{\"greenColor\": \"" + String(greenLEDState == HIGH ? "#66CC33" : "#808080") + "\", "; // 绿色LED LOGO的颜色
93   json += "\"yellowColor\": \"" + String(yellowLEDState == HIGH ? "#FFCC33" : "#808080") + "\", "; // 黄色LED LOGO的颜色
94   json += "\"redColor\": \"" + String(redLEDState == HIGH ? "#FF0000" : "#808080") + "\", "; // 红色LED LOGO的颜色
95   json += "\"greenText\": \"" + String(greenLEDState == HIGH ? "LED ON" : "LED OFF") + "\", "; // 绿色LED的文本信息
96   json += "\"yellowText\": \"" + String(yellowLEDState == HIGH ? "LED ON" : "LED OFF") + "\", "; // 黄色LED的文本信息
97   json += "\"redText\": \"" + String(redLEDState == HIGH ? "LED ON" : "LED OFF") + "\"}"; // 红色LED的文本信息
98   return json;
99 }

```

Set the request processing function in the step

```
124 // 设置请求处理函数 Set up the request handler function
125 server.on("/", HTTP_GET, [](AsyncWebServerRequest *request) {
126     request->send_P(200, "text/html", index_html);
127 });
128 server.on("/toggle", HTTP_GET, [](AsyncWebServerRequest *request) {
129     String color = request->getParam("color")->value();
130     if (color == "green") {
131         greenLEDState = !greenLEDState;
132         digitalWrite(greenLEDPin, greenLEDState);
133     } else if (color == "yellow") {
134         yellowLEDState = !yellowLEDState;
135         digitalWrite(yellowLEDPin, yellowLEDState);
136     } else if (color == "red") {
137         redLEDState = !redLEDState;
138         digitalWrite(redLEDPin, redLEDState);
139     }
140     request->send(200, "application/json", getLEDState());
141 });
142 server.on("/ledstate", HTTP_GET, [](AsyncWebServerRequest *request) {
143     request->send(200, "application/json", getLEDState());
144 });
```

Finally start the server

```
146 // 启动服务器 Starting the server
147 server.begin();
```

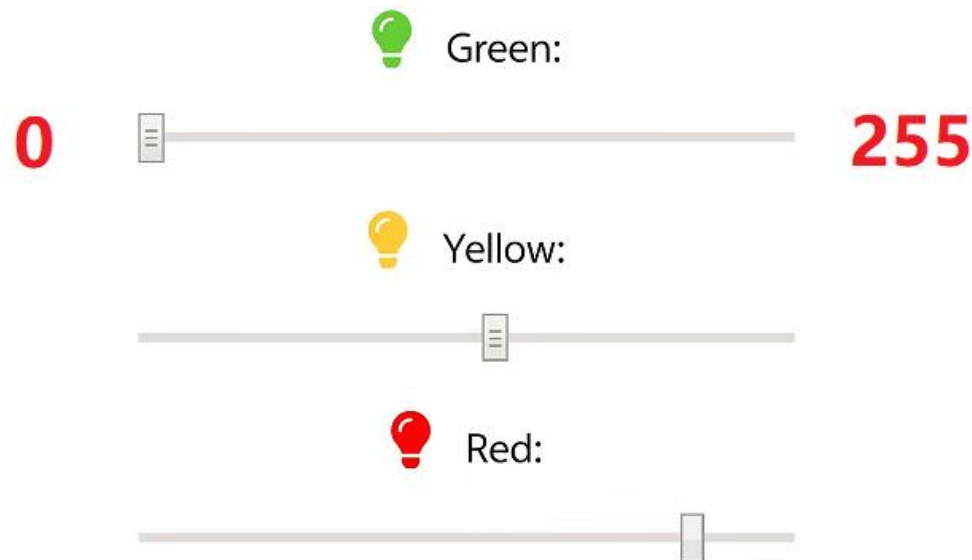
## 15. Web slider adjusts LED

### 15.1 Overview

This section focuses on learning the WiFi function of ESP32S3 and adjusting the LED brightness on the web page.

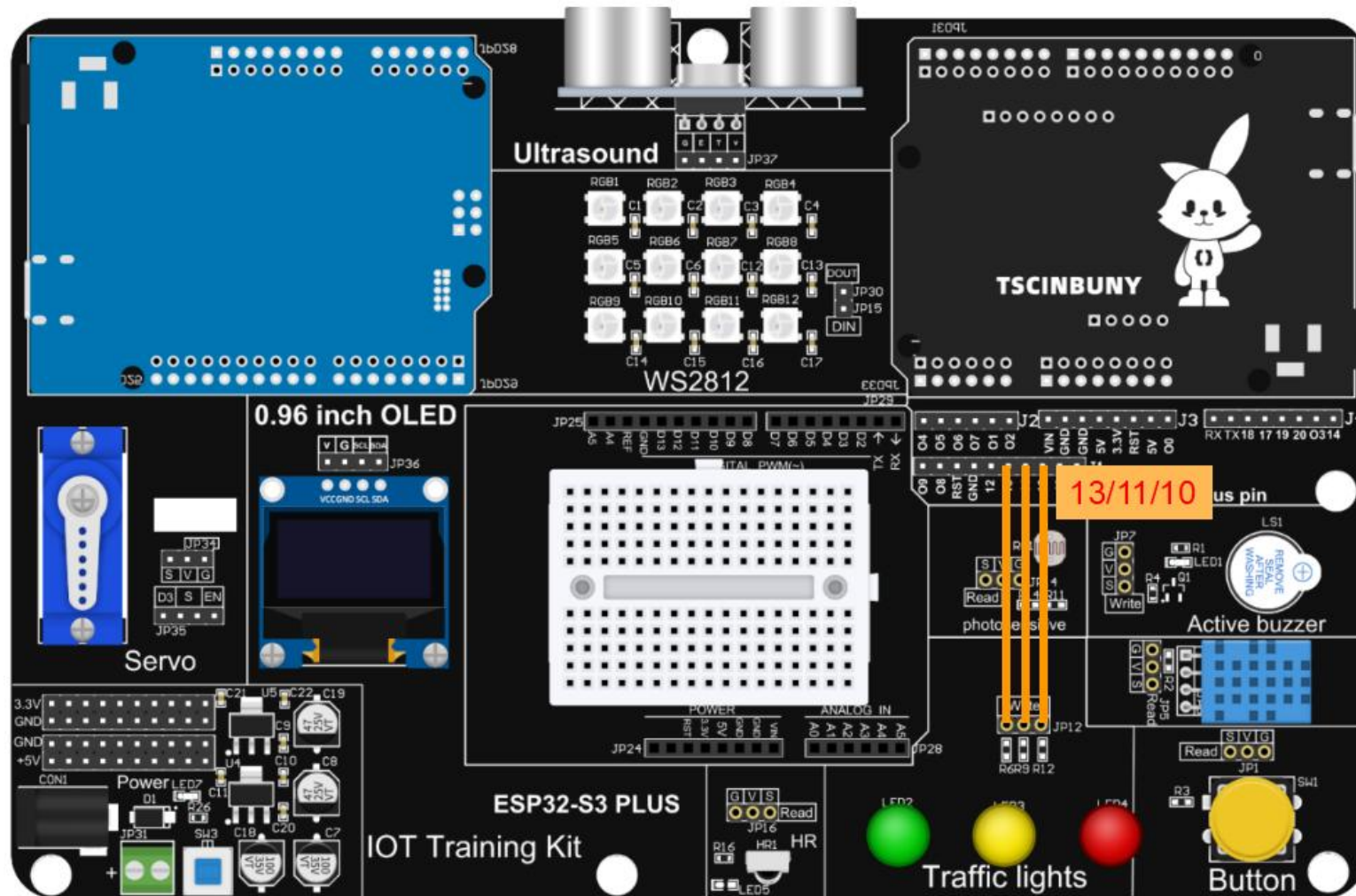
### 15.2 Working principle

Based on the content in the previous section, successfully connect to WiFi, and then adjust the LED brightness through web controls. Different from the previous section, this time the controls on the web page are sliders instead of buttons. The slider divides the LED brightness into 255 equal parts. The longer the slider, the higher the brightness of the LED.





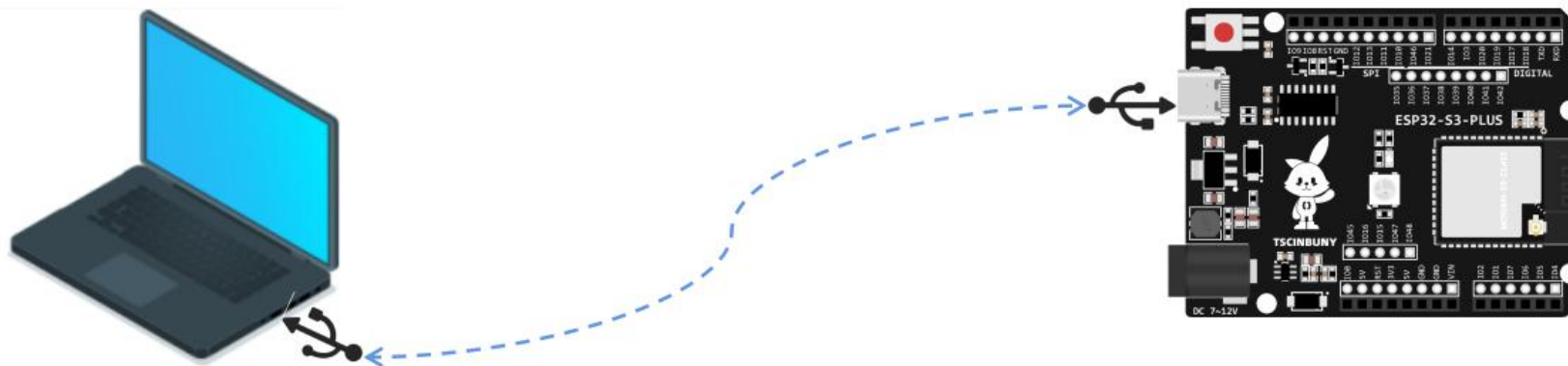
## 15.3 Connection lines





## 15.4 Upload code program

### 1 5 .4.1 Connect the main control board to the computer using a USB cable



### 1 5 .4.2 Open the program file (path: 2\_ESP32\_S3\_PLUS \ Lesson\_15\_Web\_server\_control\_LED )

Lesson_14_Web_key_controls_LED	2024/3/27 11:23	文件夹
Lesson_15_Web_server_control_LED	2024/3/27 11:24	文件夹
Lesson_16_Web_server_control_RGB	2024/3/21 18:25	文件夹
Lesson_17_Web_Display_temperature_and_humidity	2024/3/21 18:25	文件夹
Lesson_18_Web_server_control_buzzer_alarm	2024/3/21 18:23	文件夹
Lesson_19_Photorresistor_controls_LED	2024/3/21 18:24	文件夹
Lesson_20_Web_control_steering_gear_Angle_display	2024/3/21 18:26	文件夹
Lesson_21_Web_control_ultrasonic_ranging_display	2024/3/21 18:26	文件夹

Modify the WiFi account and password to which ESP32 is connected in the code. (This WiFi can be the router WiFi at home)

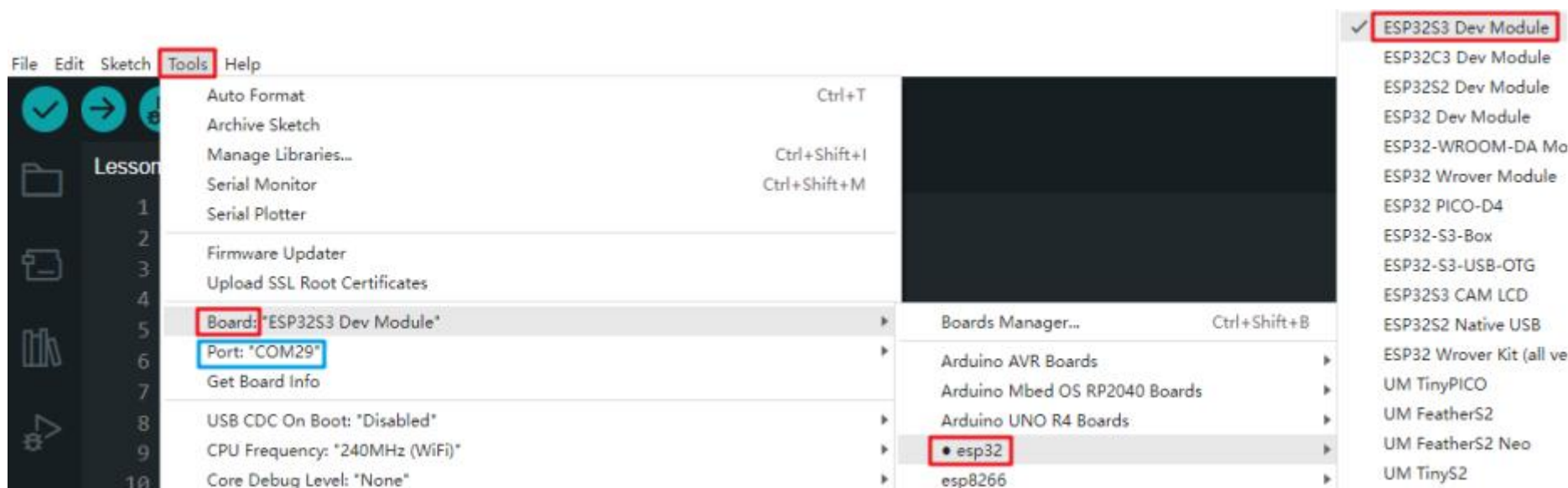
or the WiFi transmitted by the mobile hotspot, but make sure it is 2.4G and not 5G)

```

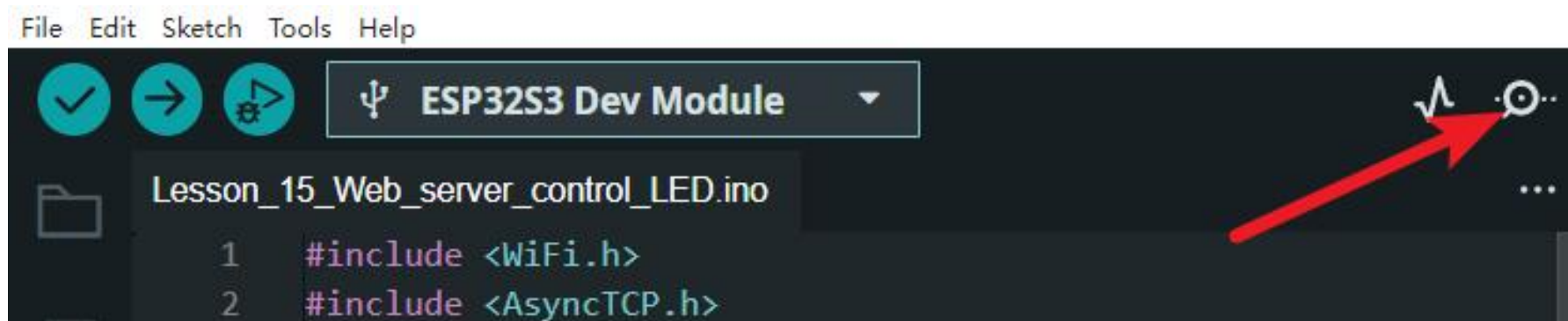
6  const char *ssid = "xxxxxx"; //输入ESP32要连接的WiFi账号 Enter the
7  const char *password = "xxxxxxxx"; //设置WiFi密码 Set WiFi password

```

Then select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board, and wait for the program upload to be completed .



After the program upload is completed, open the serial monitor to view the IP address.

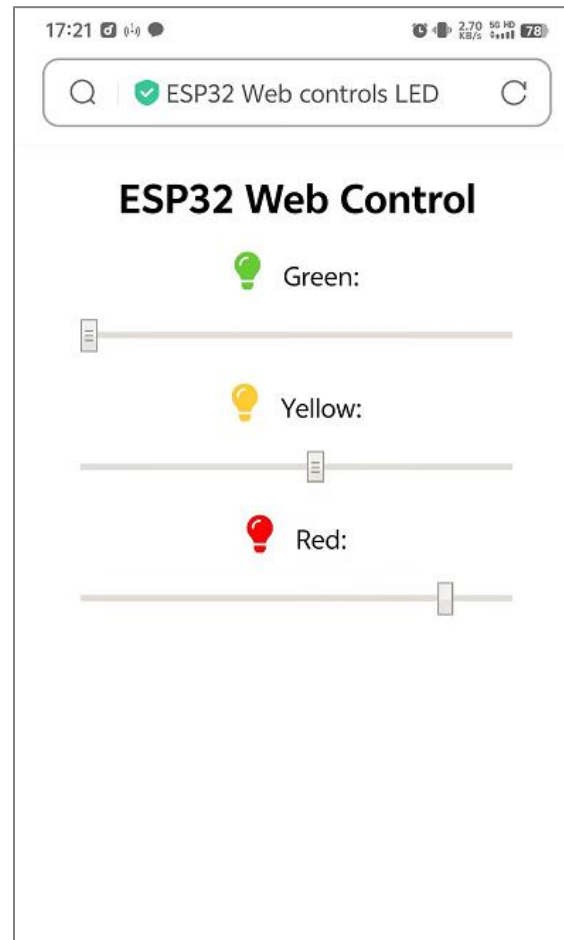


Here the IP is 192.168.8.1, but everyone will get a different IP.



Connect to the same WiFi with your mobile phone, then open the browser, enter the IP address above and enter the page.

Slide the slider to the right to control the corresponding LED brightness.



## 15.5 Code analysis

Declare the required libraries mainly used to create web services and handle asynchronous event drivers

```
1  #include <WiFi.h>
2  #include <AsyncTCP.h>
3  #include <ESPAsyncWebServer.h>
4  #include <ESP32PWM.h>
```

Initialize the WiFi to be connected to ESP32 and set three LED pins

```
6  const char *ssid = "XXXXXX";           //设置WiFi账号 Set up WiFi account
7  const char *password = "XXXXXXXXX";     //设置WiFi密码 Set WiFi password
8
9  #define GREEN_LED_PIN 13 // 定义绿色LED控制引脚 Define the green LED control pin
10 #define YELLOW_LED_PIN 11 // 定义黄色LED控制引脚 Define the yellow LED control pin
11 #define RED_LED_PIN 10 // 定义红色LED控制引脚 Define the red LED control pin
```

Create AsyncWebServer object and handle root path function

```
13 AsyncWebServer server(80); // 创建AsyncWebServer对象 Create an AsyncWebServer object
14
15 // 处理根路径的函数 A function to handle the root path
16 void handleRoot(AsyncWebServerRequest *request) {
17     request->send(200, "text/html", getIndexHtml());
18 }
```

Function to handle slider request



```

20 // 处理滑动条请求的函数 A function to handle slider requests
21 void handleSlider(AsyncWebServerRequest *request) {
22     // 读取参数值
23     if (request->hasParam("red")) {
24         int redValue = request->getParam("red")->value().toInt();
25         ledcWrite(0, redValue); // 设置红色LED亮度 Set the red LED brightness
26     }
27     if (request->hasParam("yellow")) {
28         int yellowValue = request->getParam("yellow")->value().toInt();
29         ledcWrite(1, yellowValue); // 设置黄色LED亮度 Set the yellow LED brightness
30     }
31     if (request->hasParam("green")) {
32         int greenValue = request->getParam("green")->value().toInt();
33         ledcWrite(2, greenValue); // 设置绿色LED亮度 Set the green LED brightness
34     }
35     request->send(200, "text/plain", "OK");

```

Generate the HTML code of the web page , in which the web page icon is obtained from an external link, so the connected WiFi must be a valid WiFi to obtain the icon normally.

```

38 // 生成网页的HTML代码 Generates the HTML code for the web page
39 String getIndexHtml() {
40     String html = "<!DOCTYPE html>";
41     html += "<html><head><title>ESP32 Web controls LED</title>";
42     html += "<meta name='viewport' content='width=device-width, initial-scale=1'>";
43     html += "<link rel='stylesheet' href='https://use.fontawesome.com/releases/v5.7.2/css/all.css'>";
44     html += "<style>";
45     html += "input[type=range] { width: 80%; }";
46     html += ".led-icon { font-size: 24px; margin-right: 10px; }";
47     html += "</style>";
48     html += "</head><body style='text-align:center;'>";
49     html += "<h2>ESP32 Web Control</h2>";

```

Initialize the PWM channel in the step

```
70 // 初始化PWM通道
71 ledcSetup(0, 5000, 8); // 8位分辨率, 5000Hz频率 8-bit resolution, 5000Hz frequency
72 ledcSetup(1, 5000, 8);
73 ledcSetup(2, 5000, 8);
74
75 // 绑定PWM通道到引脚
76 ledcAttachPin(RED_LED_PIN, 0); // 红色LED Red LED
77 ledcAttachPin(YELLOW_LED_PIN, 1); // 黄色LED Yellow LED
78 ledcAttachPin(GREEN_LED_PIN, 2); // 绿色LED Green LED
```

Set up WiFi connection and set request handler function

```
80 // 设置WiFi连接 Setting up WiFi connection
81 WiFi.begin(ssid, password);
82 while (WiFi.status() != WL_CONNECTED) {
83     delay(1000);
84     Serial.println("Connecting to WiFi...");
85 }
86 Serial.println("Connected to the WiFi network");
87 Serial.print("IP Address: ");
88 Serial.println(WiFi.localIP()); //打印IP地址 Printing IP addresses
89
90 // 设置请求处理函数 Set up the request handler function
91 server.on("/", HTTP_GET, handleRoot);
92 server.on("/slider", HTTP_GET, handleSlider);
93 server.begin(); // 启动服务器 Starting the server
94 }
```



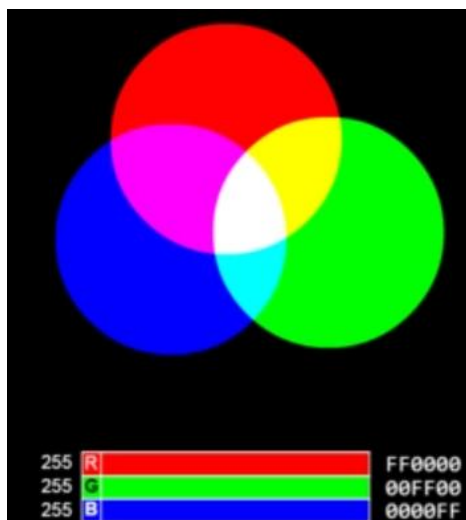
## 16. WebControlRGB

### 16.1 Overview

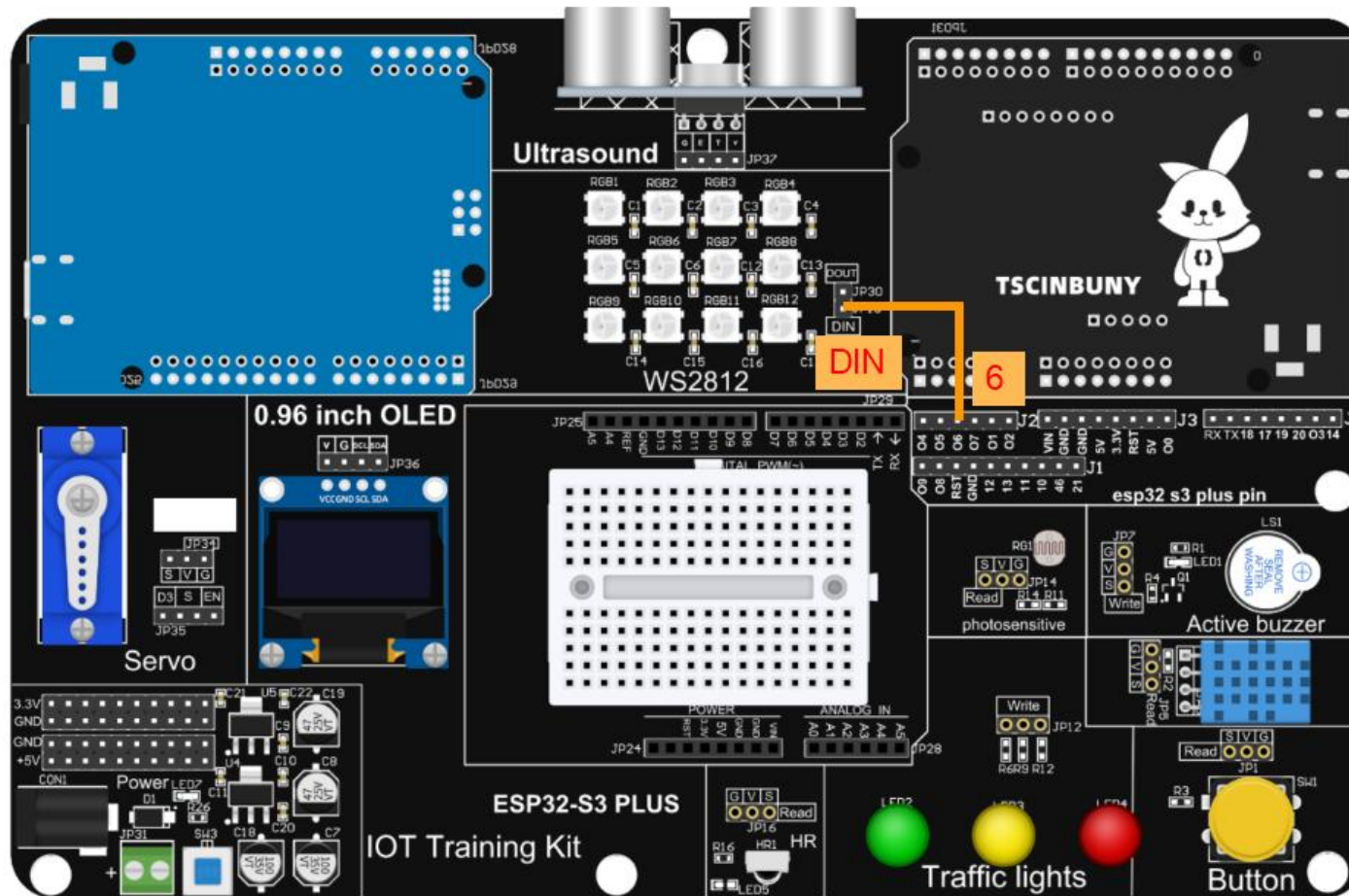
This section focuses on learning how to control WS2812 on the web through the WiFi function of ESP32S3.

### 16.2 Working principle

When ESP32 is successfully connected to WiFi, the combination of the three color channels of WS2812 can be adjusted through a new web page to obtain rich colored lights. This time the control in the web page is still a slider. The slider divides the three color channels of R/G/B into 255 equal parts respectively. The longer the slider is, the higher the color brightness ratio of that channel will be. The final superimposed value is the final color light value.

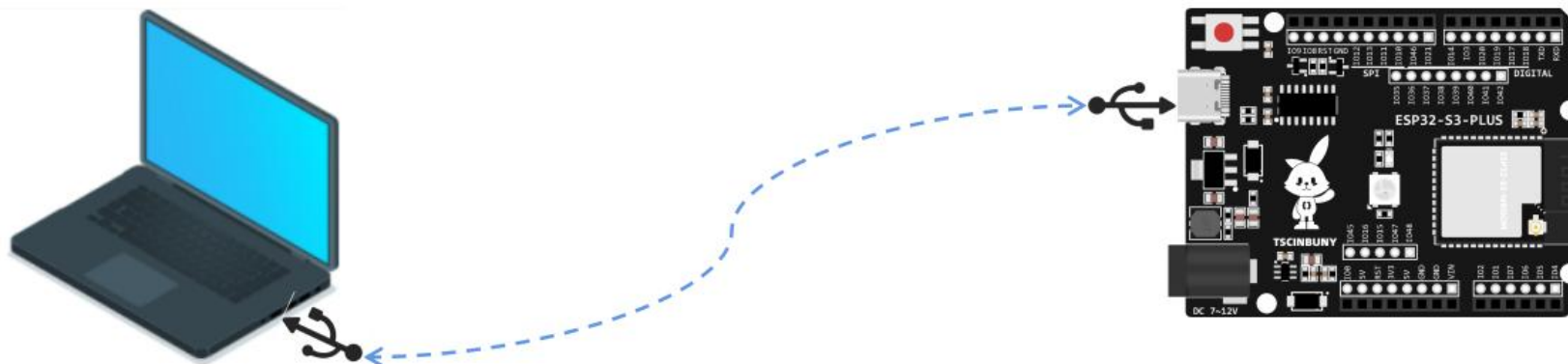


## 16.3 Connection lines



## 16.4 Upload code program

### 1 6 .4.1 Connect the main control board to the computer with a USB cable



### 1 6 .4.2 Open the program file (path: 2\_ESP32\_S3\_PLUS \ Lesson\_16\_Web\_server\_control\_RGB )

Lesson_14_Web_key_controls_LED	2024/3/27 11:23	文件夹
Lesson_15_Web_server_control_LED	2024/3/27 11:24	文件夹
Lesson_16_Web_server_control_RGB	2024/3/21 18:25	文件夹
Lesson_17_Web_Display_temperature_and_humidity	2024/3/21 18:25	文件夹
Lesson_18_Web_server_control_buzzer_alarm	2024/3/21 18:23	文件夹
Lesson_19_Photoresistor_controls_LED	2024/3/21 18:24	文件夹
Lesson_20_Web_control_steering_gear_Angle_display	2024/3/21 18:26	文件夹
Lesson_21_Web_control_ultrasonic_ranging_display	2024/3/21 18:26	文件夹

Modify the WiFi account and password to which ESP32 is connected in the code. (This WiFi can be the router WiFi at home)

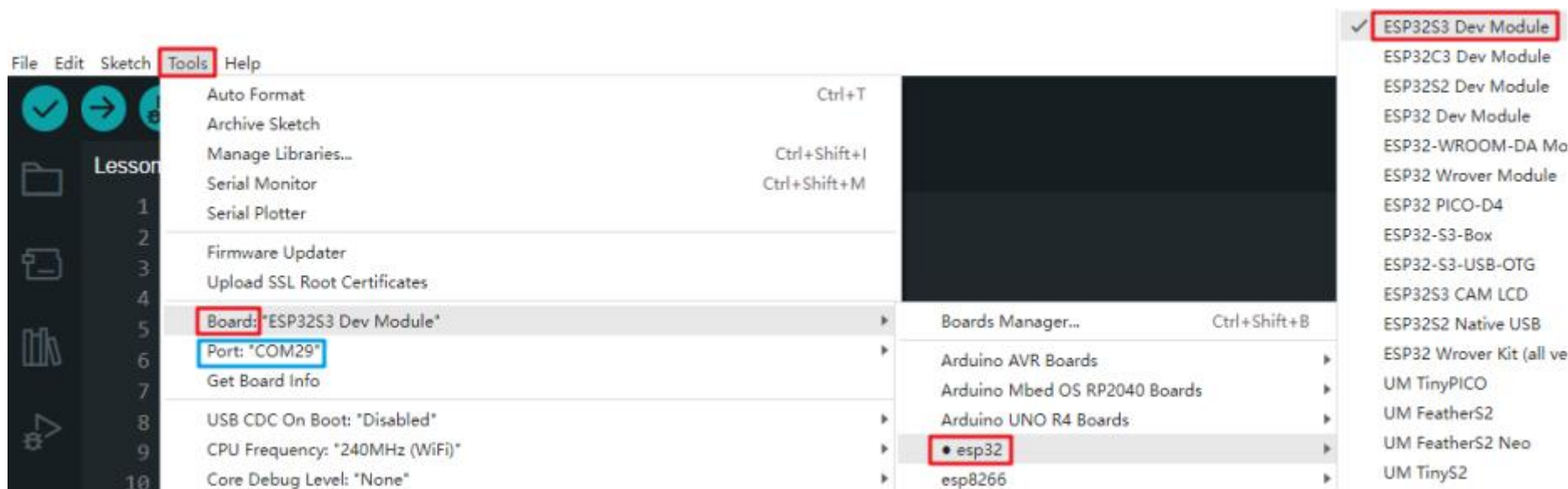
or the WiFi transmitted by the mobile hotspot, but make sure it is 2.4G and not 5G)

```

6  const char *ssid = "xxxxxx"; //输入ESP32要连接的WiFi账号 Enter the
7  const char *password = "xxxxxxxx"; //设置WiFi密码 Set WiFi password

```

Then select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board, and wait for the program upload to be completed .

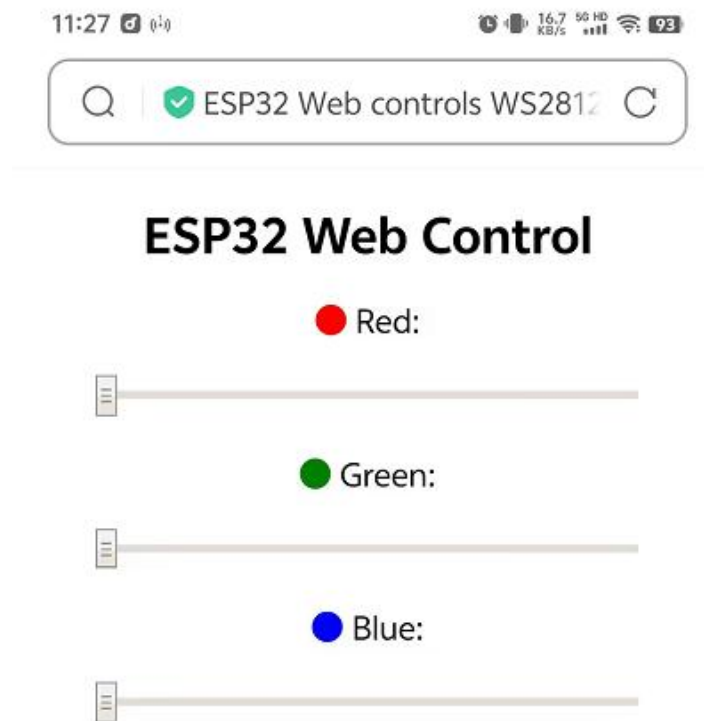


After the program upload is completed, open the serial monitor to view the IP address. Here the IP is 192.168.8.1, but

everyone will get a different IP.

Connect to the same WiFi with your mobile phone, then open the browser, enter the IP address above and enter the page.

Slide the slider to the right to control the intensity of the corresponding three channel colors.



## 16.5 Code analysis

Declare the required libraries mainly used to create web services and handle asynchronous event drivers

```
1  #include <WiFi.h>
2  #include <AsyncTCP.h>
3  #include <ESPAsyncWebServer.h>
4  #include <FastLED.h>
```

Initialize the WiFi to be connected to ESP32, set the WS2812 pin and create the RGB light array

```
6  const char *ssid = "XXXXXX"; //设置WiFi账号 Set up WiFi account
7  const char *password = "XXXXXXXX"; //设置WiFi密码 Set WiFi password
8
9  #define LED_PIN 6 // WS2812B连接的引脚 The pin for WS2812B connection
10 #define NUM_LEDS 12 // WS2812B上的灯数量 Number of lights on WS2812B
11
12 CRGB leds[NUM_LEDS]; // 创建RGB灯数组 Create an array of RGB lights
```

Create AsyncWebServer object and handle root path function

```
13 AsyncWebServer server(80); // 创建AsyncWebServer对象 Create an AsyncWebServer object
14
15 // 处理根路径的函数 A function to handle the root path
16 void handleRoot(AsyncWebServerRequest *request) {
17     request->send(200, "text/html", getIndexHtml());
18 }
```



## Function to handle slider request

```

22 // 处理颜色控制请求的函数 A function that handles requests for color control
23 void handleColor(AsyncWebServerRequest *request) {
24     // 读取参数值 Reading parameter values
25     if (request->hasParam("red") && request->hasParam("green") && request->hasParam("blue")) {
26         int red = request->getParam("red")->value().toInt();
27         int green = request->getParam("green")->value().toInt();
28         int blue = request->getParam("blue")->value().toInt();
29
30         // 设置ws2812颜色 Set the ws2812 color
31         for (int i = 0; i < NUM_LEDS; i++) {
32             leds[i] = CRGB(red, green, blue);
33         }
34         FastLED.show(); // 更新ws2812颜色 Update ws2812 color
35     }
36     request->send(200, "text/plain", "OK");

```

## Generate the HTML code of the web page .

```

38 // 生成网页的HTML代码 Generates the HTML code for the web page
39 String getIndexHtml() {
40     String html = "<!DOCTYPE html>";
41     html += "<html><head><title>ESP32 Web controls LED</title>";
42     html += "<meta name='viewport' content='width=device-width, initial-scale=1'>";
43     html += "<link rel='stylesheet' href='https://use.fontawesome.com/releases/v5.7.2/css/all.css'>";
44     html += "<style>";
45     html += "input[type=range] { width: 80%; }";
46     html += ".led-icon { font-size: 24px; margin-right: 10px; }";
47     html += "</style>";
48     html += "</head><body style='text-align:center;'>";
49     html += "<h2>ESP32 Web Control</h2>";

```

Initialize the WS2812 light strip and WiFi connection in the step

```
67 void setup() {  
68     Serial.begin(9600); // 设置串口波特率 Set the serial port baud rate  
69     FastLED.addLeds<WS2812B, LED_PIN, GRB>(leds, NUM_LEDS); // 初始化ws2812灯条 Initialize the ws2812 light bar  
70  
71     // 设置 WiFi 连接 setting up WiFi connection  
72     WiFi.begin(ssid, password);  
73     while (WiFi.status() != WL_CONNECTED) {  
74         delay(1000);  
75         Serial.println("Connecting to WiFi...");  
76     }
```

Set request handling function

```
81     // 设置请求处理函数 Set up the request handler function  
82     server.on("/", HTTP_GET, handleRoot);  
83     server.on("/color", HTTP_GET, handleColor);  
84  
85     // 启动服务器 Starting the server  
86     server.begin();  
87 }
```

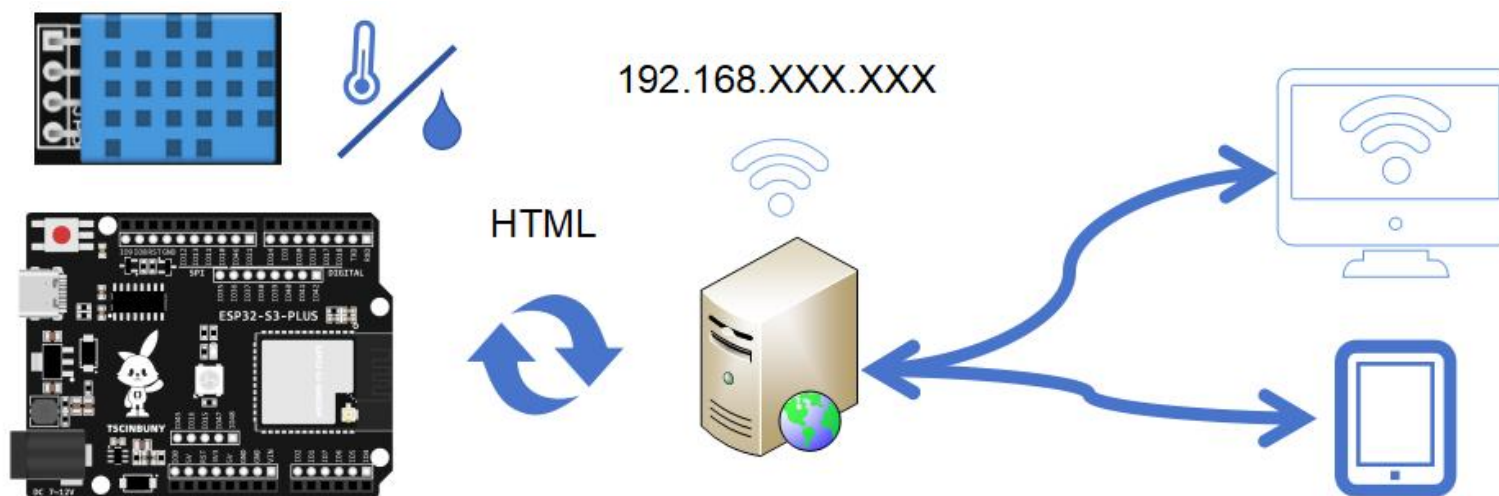
## 17. Web display temperature and humidity

### 17.1 Overview

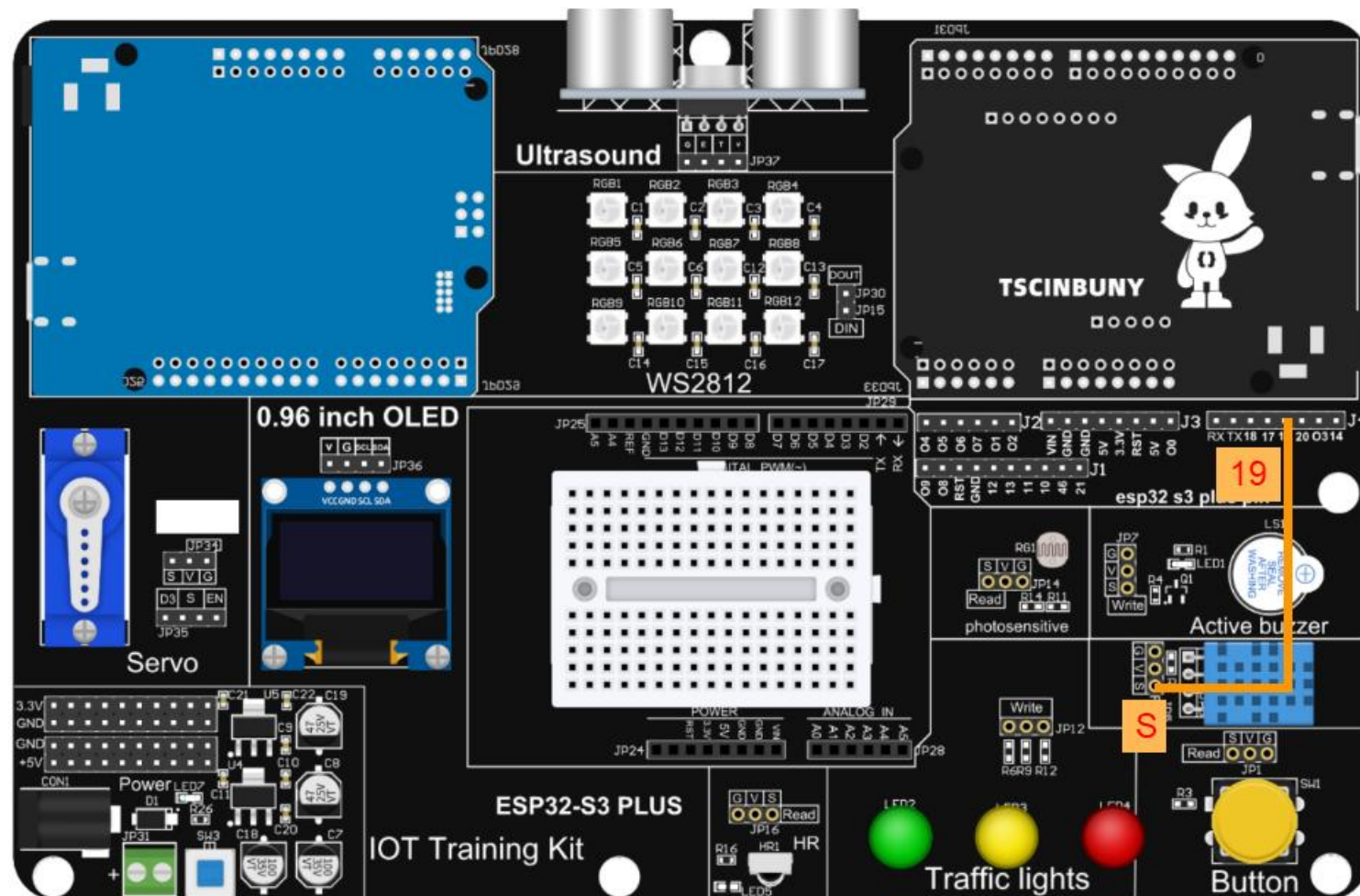
This section focuses on learning how to display temperature and humidity on the web page through the WiFi function of ESP32S3.

### 17.2 Working principle

When ESP32 successfully connects to WiFi, push the temperature and humidity values obtained by DHT11 to the web service web page through a new web page. Connect to the same WiFi with a mobile device and access the same IP address in the browser to get the web service. temperature and humidity data.



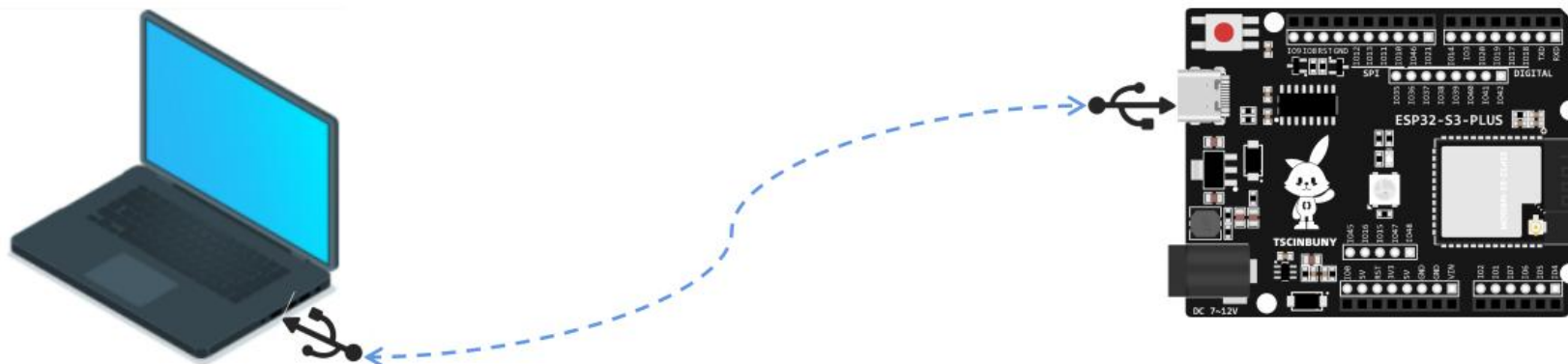
## 17.3 Connection lines





## 17.4 Upload code program

### 1 7 .4.1 Connect the main control board to the computer with a USB cable



### 1 7 .4.2 Open the program file (path: 2\_ESP32\_S3\_PLUS \ Lesson\_17\_Web\_Display\_temperature\_and\_humidity )

Lesson_14_Web_key_controls_LED	2024/3/27 11:23	文件夹
Lesson_15_Web_server_control_LED	2024/3/27 11:24	文件夹
Lesson_16_Web_server_control_RGB	2024/3/21 18:25	文件夹
Lesson_17_Web_Display_temperature_and_humidity	2024/3/21 18:25	文件夹
Lesson_18_Web_server_control_buzzer_alarm	2024/3/21 18:23	文件夹
Lesson_19_Photorresistor_controls_LED	2024/3/21 18:24	文件夹
Lesson_20_Web_control_steering_gear_Angle_display	2024/3/21 18:26	文件夹
Lesson_21_Web_control_ultrasonic_ranging_display	2024/3/21 18:26	文件夹

Modify the WiFi account and password to which ESP32 is connected in the code. (This WiFi can be the router WiFi at home )

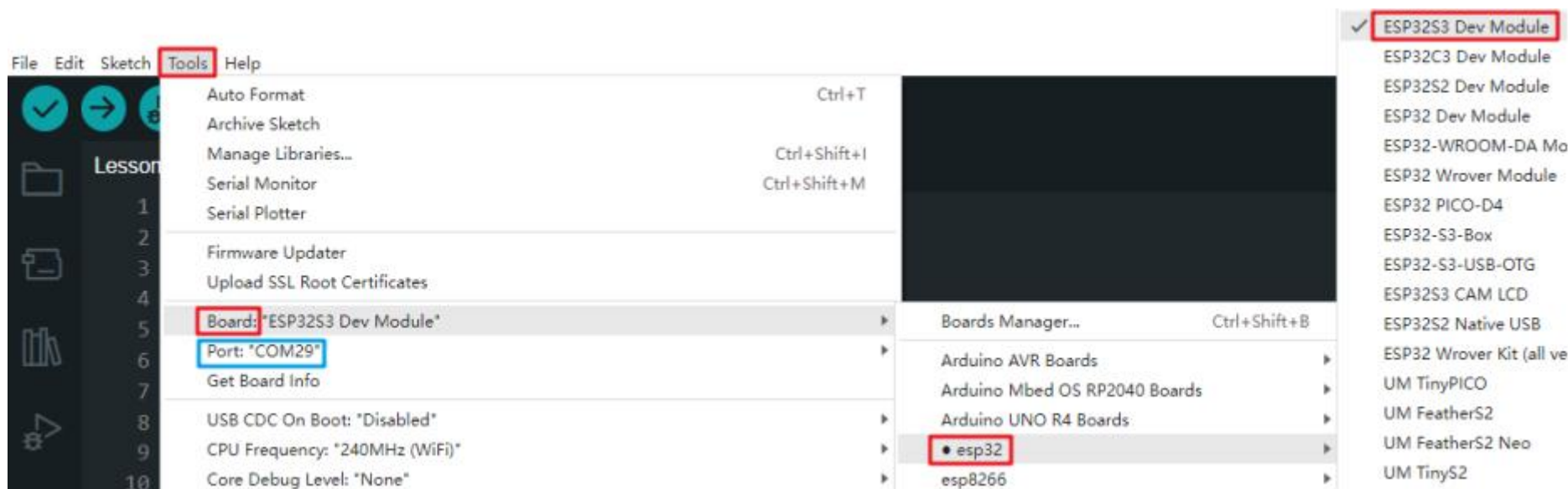
or the WiFi transmitted by the mobile hotspot, but make sure it is 2.4G and not 5G)

```

6  const char *ssid = "xxxxxx"; //输入ESP32要连接的WiFi账号 Enter the
7  const char *password = "xxxxxxxx"; //设置WiFi密码 Set WiFi password

```

Then select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board, and wait for the program upload to be completed .

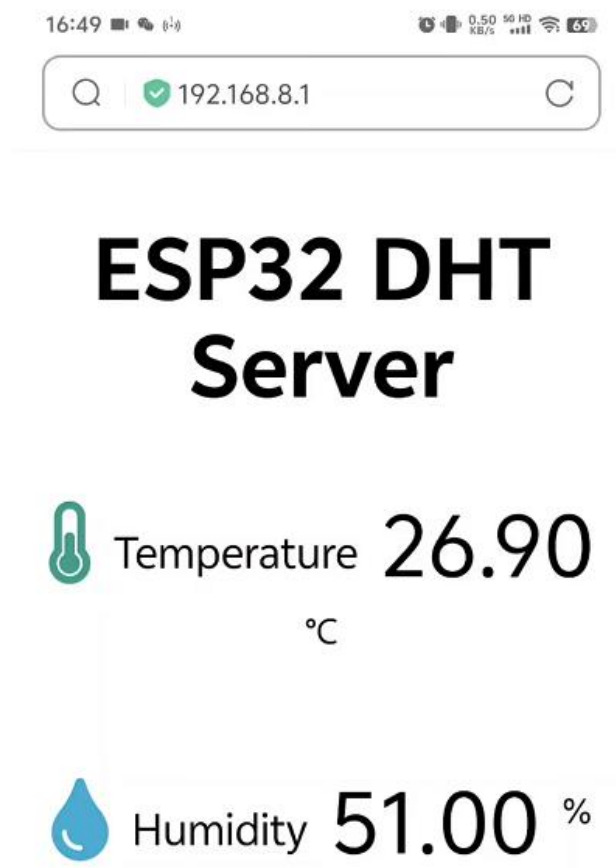


After the program upload is completed, open the serial monitor to view the IP address. Here the IP is 192.168.8.1, but



everyone will get a different IP.

Use your mobile phone to connect to the same WiFi, then open the browser, enter the IP address above and enter the page to see the temperature and humidity values.



## 17.5 Code analysis

Declare the required libraries, mainly used to create web services and DHT11 to obtain temperature and humidity

```
1  #include <Arduino.h>
2  #include <WiFi.h>
3  #include <AsyncTCP.h>
4  #include <ESPAsyncWebServer.h>
5  #include <Adafruit_Sensor.h>
6  #include <DHT.h>
```

Initialize the WiFi to be connected to ESP32, set the DHT11 pin and create the dht object and server object

```
8  const char *ssid = "XXXXXX";      //设置WiFi账号 Set up WiFi account
9  const char *password = "XXXXXXXX"; //设置WiFi密码 Set WiFi password
10
11 #define DHTPIN 19                  // 19引脚连接到DHT传感器 The 19 pins are connected to the DHT sensor
12 #define DHTTYPE DHT11              // 定义传感器类型为DHT11 The sensor type is defined as DHT11
13 DHT dht(DHTPIN, DHTTYPE);          // 创建一个DHT对象，使用指定的引脚和传感器类型 Creates a DHT object with
14 // 创建 AsyncWebServer 对象
15 AsyncWebServer server(80);
```

Define temperature, humidity variables and time variables

```
17 float t = 0.0; // 定义一个浮点型变量t，用于存储温度值 Define a floating-point variable t to store the temperature
18 float h = 0.0; // 定义一个浮点型变量h，用于存储湿度值 Define a floating point variable h to store the humidity value
19 unsigned long previousMillis = 0; // 将存储上次DHT更新的时间 The time of the last DHT update is stored
20 const long interval = 10000;      // 每10秒更新DHT读数 The DHT readings are updated every 10 seconds
```

Generate the HTML code of the web page .

```

22 // 生成网页的 HTML 代码 Generates the HTML code for the web page
23 const char index_html[] PROGMEM = R"rawliteral(
24 <!DOCTYPE HTML><html>
25 <head>
26   <meta name="viewport" content="width=device-width, initial-scale=1">
27   <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.2/css/all.css" integrity="sha384-fnmOCqbTlWIlj8LyTjo
28   <style>
29     html {
30       font-family: Arial;
31       display: inline-block;
32       margin: 0px auto;
33       text-align: center;

```

Function processor requests to obtain temperature and humidity

```

85 String processor(const String &var) {
86   if (var == "TEMPERATURE") { // 如果请求的变量是"TEMPERATURE" If the request variable is "TEMPERATURE"
87     return String(t); // 返回当前温度值的字符串形式 Returns the current temperature as a string
88   } else if (var == "HUMIDITY") { // 如果请求的变量是"HUMIDITY" If the requested variable is "HUMIDITY"
89     return String(h); // 返回当前湿度值的字符串形式 Returns the current humidity value as a string
90   }
91   return String(); // 如果请求的变量不是"TEMPERATURE"或"HUMIDITY", 则返回空字符串 If the requested variable is no
92 }

```

Initialize dht and set up WiFi connection in step

```

94 void setup() {
95     Serial.begin(9600);
96     dht.begin();          // 初始化DHT11 Initialize DHT11
97
98     // 设置 WiFi 连接 Setting up WiFi connection
99     WiFi.begin(ssid, password);
100    Serial.println("Connecting to WiFi");
101    while (WiFi.status() != WL_CONNECTED) {
102        delay(1000);
103        Serial.println(".");
104    }
105
106    Serial.println(WiFi.localIP()); //打印IP地址 Printing IP addresses

```

### Set request handling function

```

108 server.on("/", HTTP_GET, [](AsyncWebServerRequest *request) {
109     request->send_P(200, "text/html", index_html, processor);
110 });
111 server.on("/temperature", HTTP_GET, [](AsyncWebServerRequest *request) {
112     float newT = dht.readTemperature();
113     if (isnan(newT)) {
114         request->send(500, "text/plain", "Failed to read temperature from DHT sensor!");
115     } else {
116         request->send_P(200, "text/plain", String(newT).c_str());
117     }
118 });
119 server.on("/humidity", HTTP_GET, [](AsyncWebServerRequest *request) {
120     float newH = dht.readHumidity();
121     if (isnan(newH)) {
122         request->send(500, "text/plain", "Failed to read humidity from DHT sensor!");
123     } else {
124         request->send_P(200, "text/plain", String(newH).c_str());

```

Regularly refresh temperature and humidity data within the loop function

```

132 void loop() {
133     unsigned long currentMillis = millis();           //获取当前的毫秒数，用于记录当前时间。 Gets the current num
134     if (currentMillis - previousMillis >= interval) { //检查是否已经到达了指定的时间间隔，如果是则执行下面的代码。
135         previousMillis = currentMillis;               //更新上一次执行的时间，以便下一次判断时间间隔。 Update the
136         float newT = dht.readTemperature();           //读取温度传感器的数据，并将读取到的值存储在newT变量中。 Th
137         if (isnan(newT)) {                             //检查读取的温度值是否为非数字（NaN），如果是则输出错误信息。
138             Serial.println("Failed to read temperature from DHT sensor!");
139         } else {
140             t = newT;                                  //将读取到的温度值赋给t变量 Assign the temperature to the t variable
141             Serial.println(t);                          //打印输出温度 Print output temperature
142         }
143         float newH = dht.readHumidity();               //读取湿度传感器的数据，并将读取到的值存储在newH变量中 Read the humidity s
144         if (isnan(newH)) {                             //检查读取的湿度值是否为非数字（NaN），如果是则输出错误信息 Check if the h
145             Serial.println("Failed to read humidity from DHT sensor!");
146         } else {
147             h = newH;                                  //将读取到的湿度值赋给h变量 The read humidity value is assigned to the h variable
148             Serial.println(h);                          //打印输出湿度 Print output humidity
149         }
150     }

```



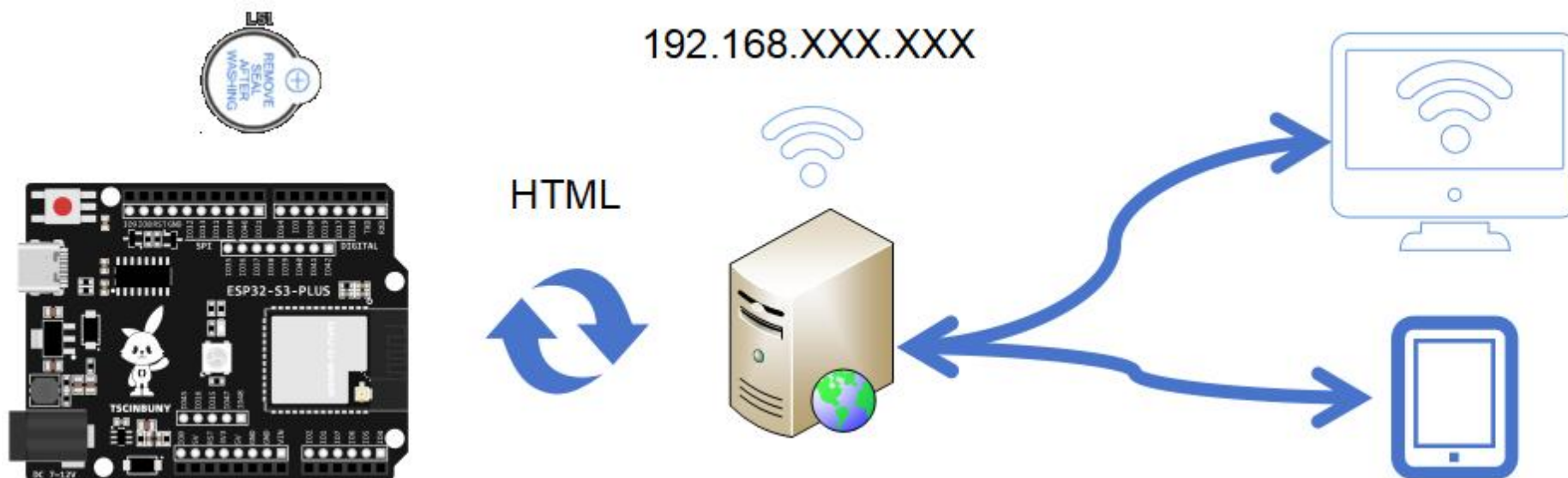
## 18. Web control buzzer

### 18.1 Overview

This section focuses on learning the WiFi function of ESP32S3 and controlling the buzzer on the web page.

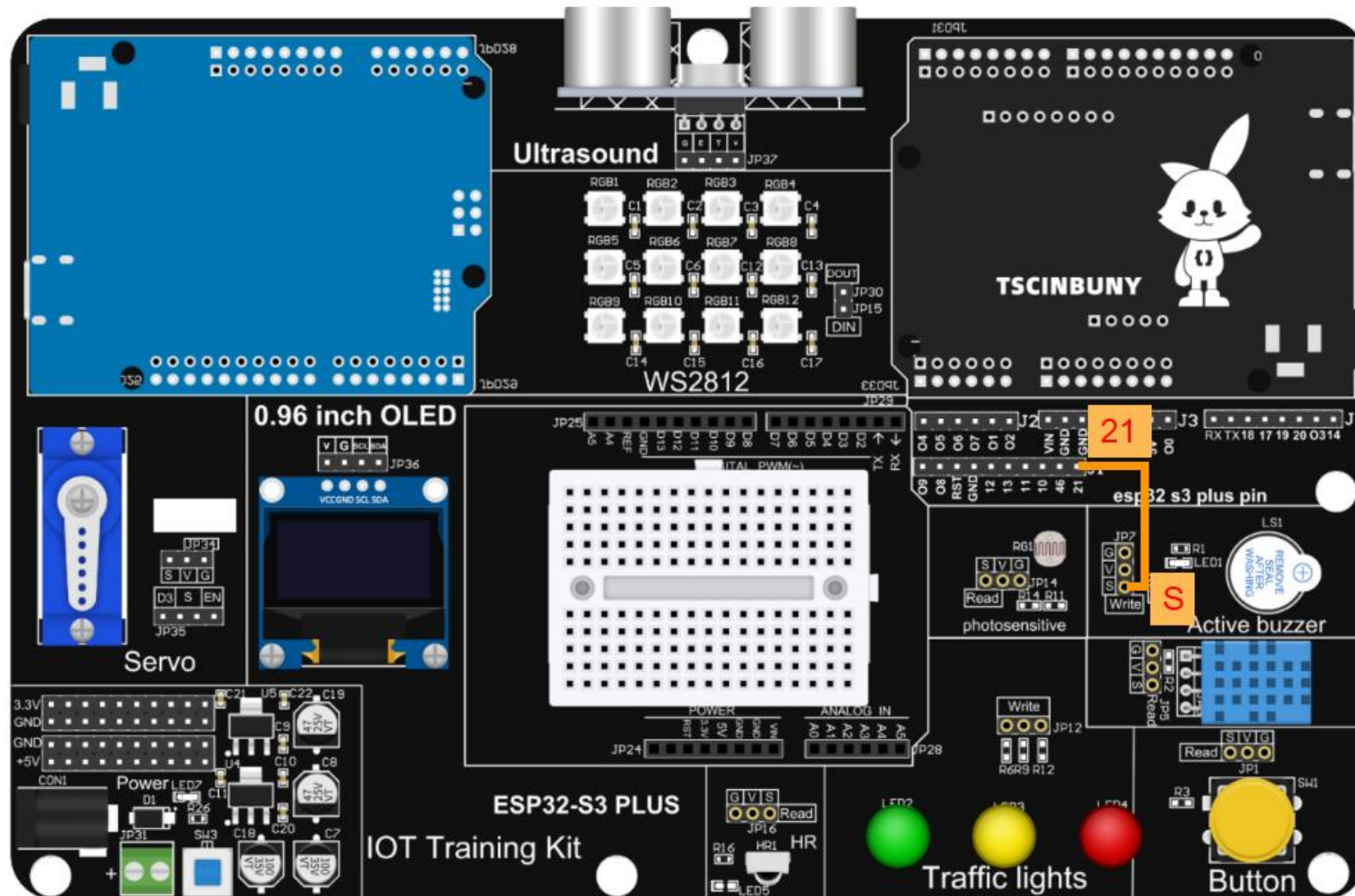
### 18.2 Working principle

When ESP32 successfully connects to WiFi, push the web page that generates the button control to the web service, connect to the same WiFi with a mobile device, and access the same IP address in the browser, you can control the buzzer through the web control.



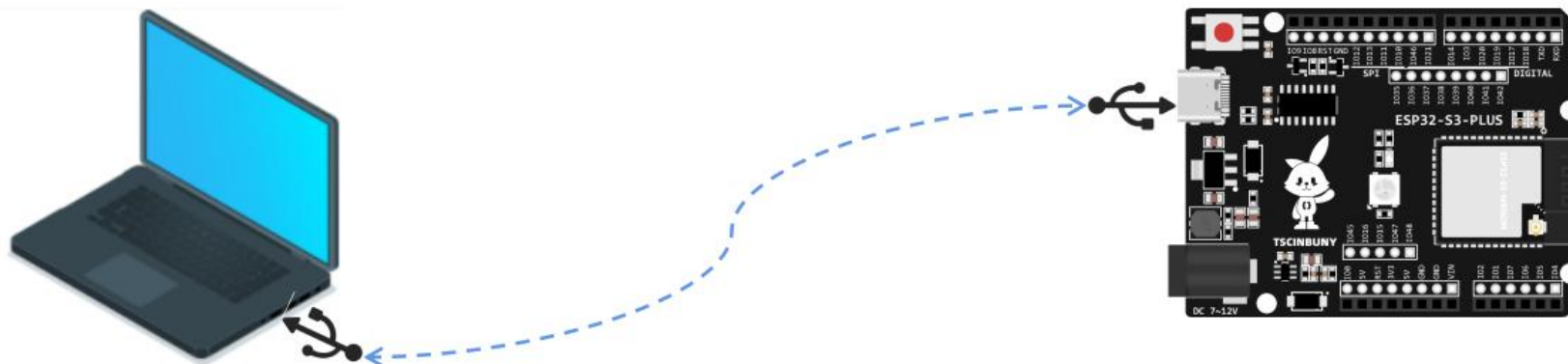


## 18.3 Connection lines



## 18.4 Upload code program

### 1 8 .4.1 Connect the main control board to the computer using USB cable



### 1 8 .4.2 Open the program file (path: 2\_ESP32\_S3\_PLUS \ Lesson\_18\_Web\_server\_control\_buzzer\_alarm )

Lesson_14_Web_key_controls_LED	2024/3/27 11:23	文件夹
Lesson_15_Web_server_control_LED	2024/3/27 11:24	文件夹
Lesson_16_Web_server_control_RGB	2024/3/21 18:25	文件夹
Lesson_17_Web_Display_temperature_and_humidity	2024/3/21 18:25	文件夹
Lesson_18_Web_server_control_buzzer_alarm	2024/3/21 18:23	文件夹
Lesson_19_Photorresistor_controls_LED	2024/3/21 18:24	文件夹
Lesson_20_Web_control_steering_gear_Angle_display	2024/3/21 18:26	文件夹
Lesson_21_Web_control_ultrasonic_ranging_display	2024/3/21 18:26	文件夹

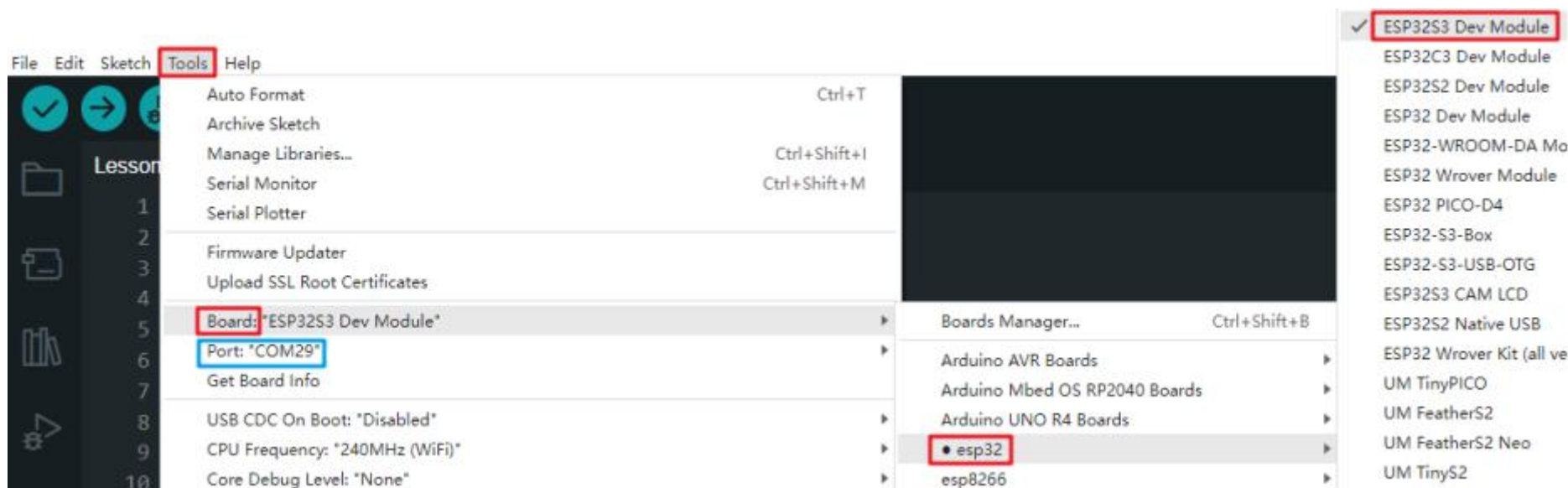
Modify the WiFi account and password to which ESP32 is connected in the code. (This WiFi can be the router WiFi at home or the WiFi transmitted by the mobile hotspot, but make sure it is 2.4G and not 5G)

```

6  const char *ssid = "xxxxxx"; //输入ESP32要连接的WiFi账号 Enter th
7  const char *password = "xxxxxxxx"; //设置WiFi密码 Set WiFi password

```

Then select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board, and wait for the program upload to be completed .



After the program upload is completed, open the serial monitor to view the IP address. Here the IP is 192.168.8.1, but everyone will get a different IP.

Use your mobile phone to connect to the same WiFi, then open the browser, enter the IP address above and enter the page to see the page for controlling the buzzer.



# ESP32 Buzzer Control



## 18.5 Code analysis

Declare the required libraries to create web services

```
1  #include <Arduino.h>
2  #include <WiFi.h>
3  #include <AsyncTCP.h>
4  #include <ESPAsyncWebServer.h>
```

Initialize the WiFi to be connected to ESP32, set the buzzer pin, status variable and create the server object

```
6  const char *ssid = "XXXXXX";           //设置WiFi账号 Set up WiFi account
7  const char *password = "XXXXXXXX";      //设置WiFi密码 Set WiFi password
8  const int buzzerPin = 21;               // 21引脚连接到有源蜂鸣器 The 21 pins are connected to the active buzzer
9  bool isBuzzerOn = false;               // 跟踪蜂鸣器的状态 Track the state of the buzzer
10 AsyncWebServer server(80);             // 创建 AsyncWebServer 对象
```

Generate the HTML code of the web page .

```
13 // 生成网页的 HTML 代码 Generate HTML code for the web page
14 const char index_html[] PROGMEM = R"rawliteral(
15 <!DOCTYPE HTML><html>
16 <head>
17   <meta name="viewport" content="width=device-width, initial-scale=1">
18   <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.2/css/all.css" integrity="sha384-fnmOCqbTlWIlj8LyTjo7m0UStjsKC4p
19   <style>
20     html {
21       font-family: Arial;
22       display: inline-block;
23       margin: 0px auto;
```

Set the buzzer pin working mode and set up WiFi connection in the step



```

62 void setup() {
63     Serial.begin(9600);
64     pinMode(buzzerPin, OUTPUT); // 设置引脚工作模式为输出 Set the pin operation mode to output
65     digitalWrite(buzzerPin, LOW); // 蜂鸣器初始设置关闭 The initial setting of the buzzer is off
66
67     // 设置 WiFi 连接 Setting up WiFi connection
68     WiFi.begin(ssid, password);
69     Serial.println("Connecting to WiFi");
70     while (WiFi.status() != WL_CONNECTED) {
71         delay(1000);
72         Serial.println(".");
73     }
74
75     Serial.println(WiFi.localIP()); //打印IP地址 Printing IP addresses

```

Set the request processing function server.on

```

77 server.on("/", HTTP_GET, [](AsyncWebServerRequest *request) {
78     request->send_P(200, "text/html", index_html);
79 });
80
81 server.on("/toggleBuzzer", HTTP_GET, [](AsyncWebServerRequest *request) {
82     if (isBuzzerOn) {
83         digitalWrite(buzzerPin, LOW); // 关闭蜂鸣器 Turn off the buzzer
84         isBuzzerOn = false;
85     } else {
86         digitalWrite(buzzerPin, HIGH); // 打开蜂鸣器 Turn on the buzzer
87         isBuzzerOn = true;
88     }
89     request->send(200, "text/plain", "Buzzer toggled");

```



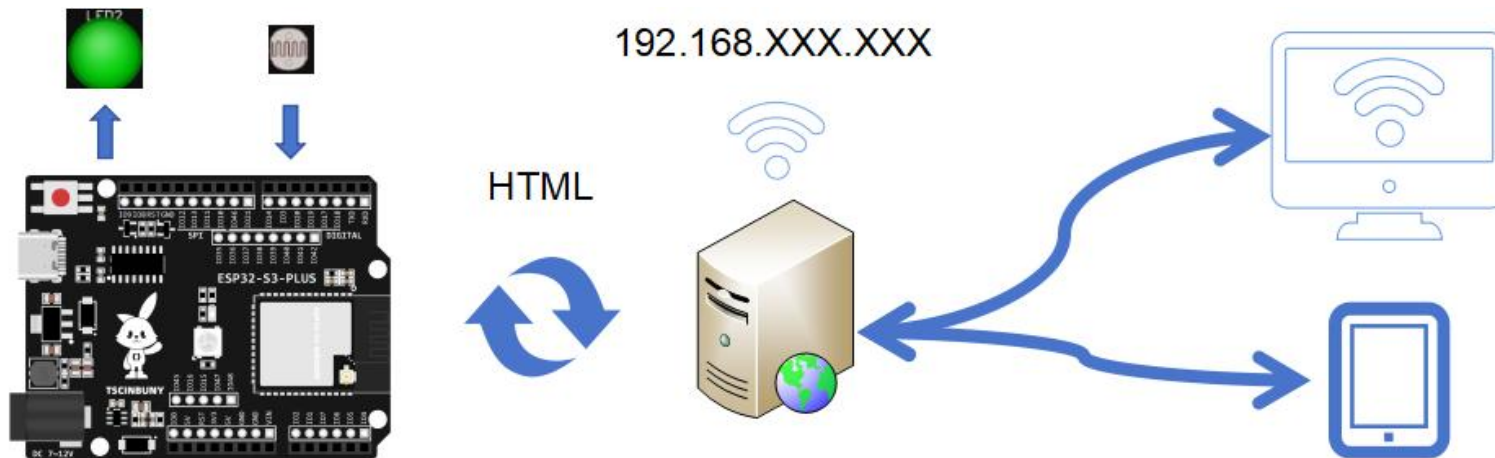
## 19. Photoresistor Controlled LED

### 19.1 Overview

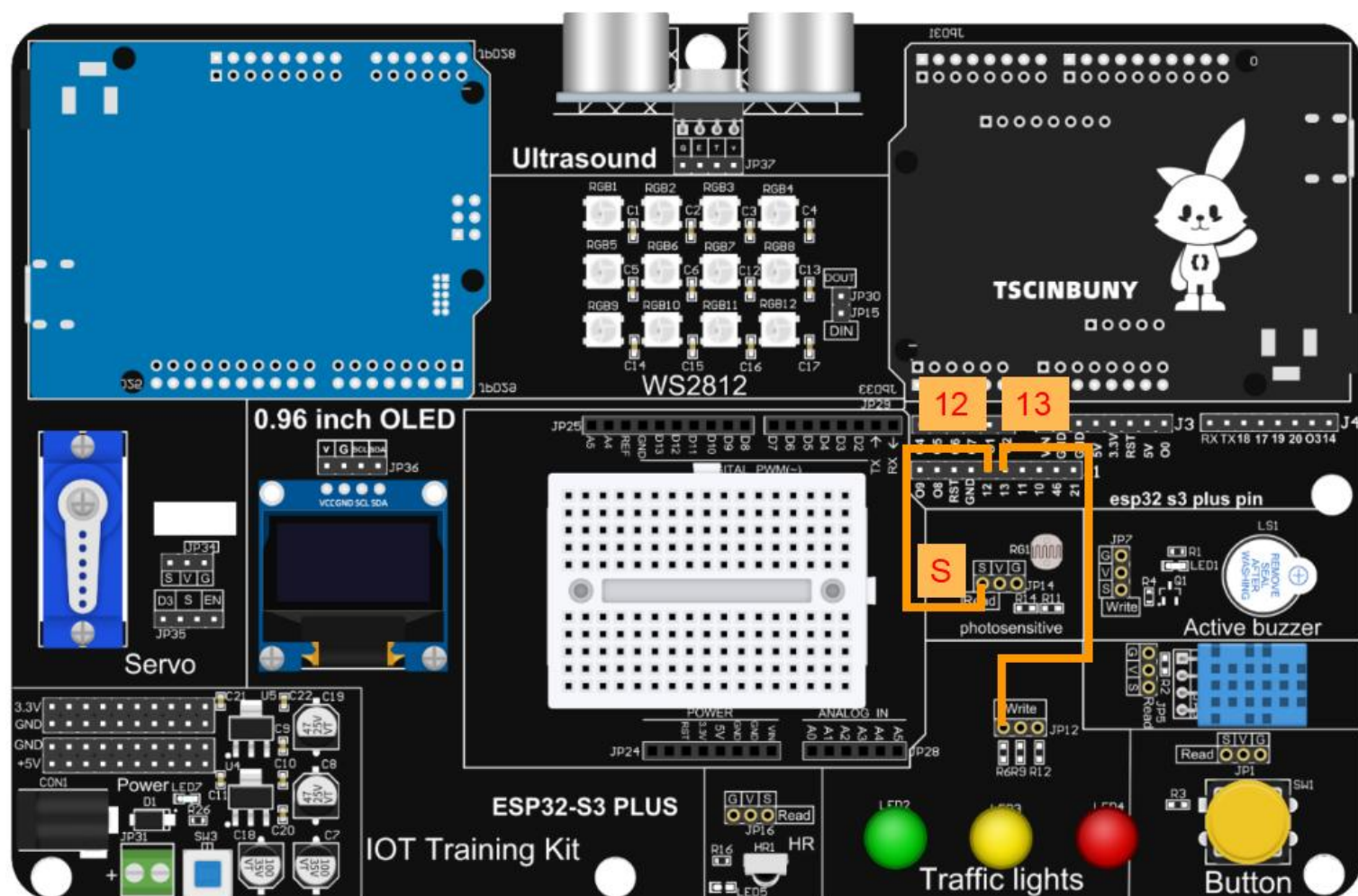
This section focuses on learning the WiFi function of ESP32S3 and monitoring the lighting conditions and LED status in real time on the web page.

### 19.2 Working principle

When ESP32 successfully connects to WiFi, push the web page that generates LED status and light intensity value data to the web service. Use a mobile device to connect to the same WiFi and access the same IP address in the browser to monitor the lighting in real time through the web page. In this case, when the illumination is lower than 70%, the LED will be triggered to light up.

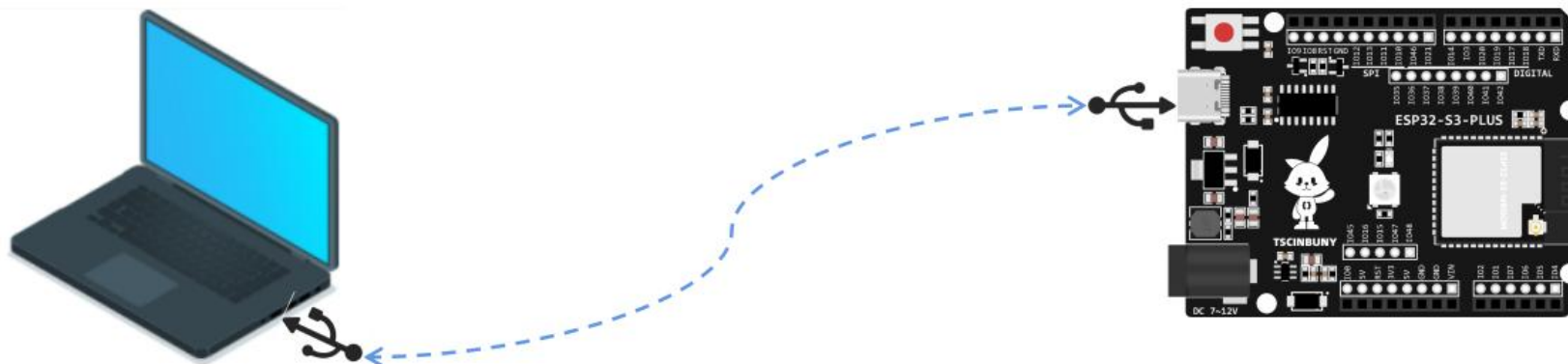


### 19.3 Connection lines



## 19.4 Upload code program

### 19.4.1 Connect the main control board to the computer using a USB cable



### 19.4.2 Open the program file (path: 2\_ESP32\_S3\_PLUS \ Lesson\_19\_Photoresistor\_controls\_LED )

Lesson_14_Web_key_controls_LED	2024/3/27 11:23	文件夹
Lesson_15_Web_server_control_LED	2024/3/27 11:24	文件夹
Lesson_16_Web_server_control_RGB	2024/3/21 18:25	文件夹
Lesson_17_Web_Display_temperature_and_humidity	2024/3/21 18:25	文件夹
Lesson_18_Web_server_control_buzzer_alarm	2024/3/21 18:23	文件夹
Lesson_19_Photoresistor_controls_LED	2024/3/21 18:24	文件夹
Lesson_20_Web_control_steering_gear_Angle_display	2024/3/21 18:26	文件夹
Lesson_21_Web_control_ultrasonic_ranging_display	2024/3/21 18:26	文件夹

Modify the WiFi account and password to which ESP32 is connected in the code. (This WiFi can be the router WiFi at home)

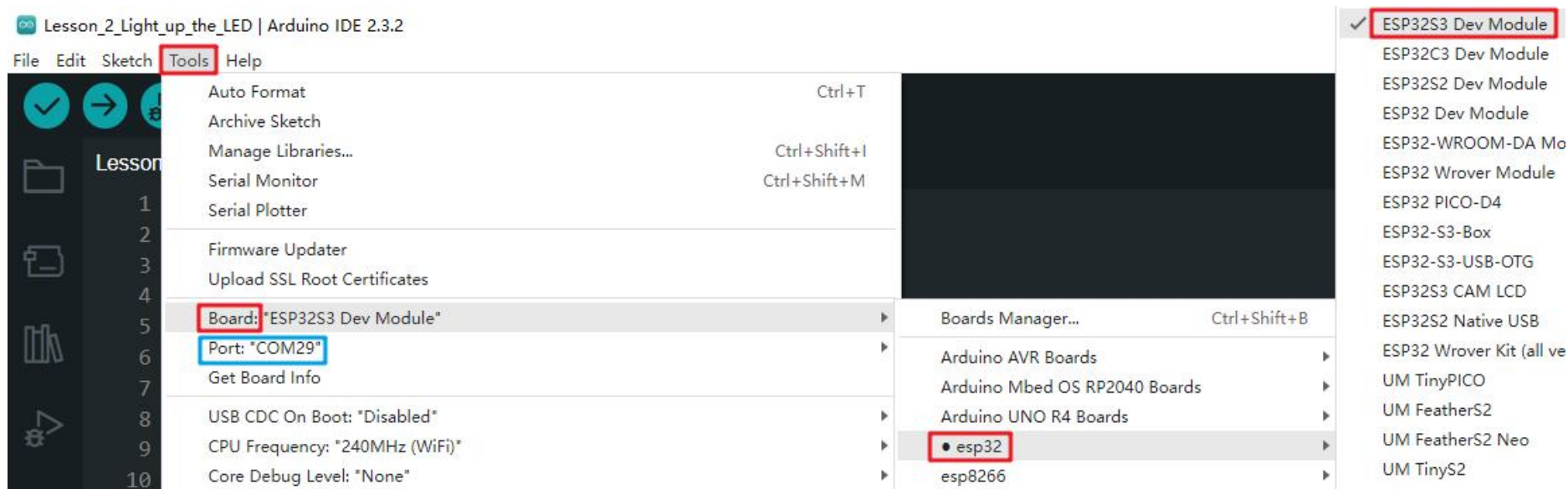
or the WiFi transmitted by the mobile hotspot, but make sure it is 2.4G and not 5G)

```

6  const char *ssid = "xxxxxx"; //输入ESP32要连接的WiFi账号 Enter the
7  const char *password = "xxxxxxxx"; //设置WiFi密码 Set WiFi password

```

Then select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board, and wait for the program upload to be completed .



After the program upload is completed, open the serial monitor to view the IP address. Here the IP is 192.168.246.1, but the

IP everyone gets will be different.

Use your mobile phone to connect to the same WiFi, then open the browser, enter the IP address above and enter the page to see the lighting conditions and LED conditions in real time.



## ESP32 Light Control



 Brightness **77** %

When the light intensity is below 70%, the LED will be turned on.



## ESP32 Light Control



 Brightness **47** %

When the light intensity is below 70%, the LED will be turned on.



## 19.5 Code analysis

Declare the required libraries, mainly used to create web services.

```
1  #include <Arduino.h>
2  #include <WiFi.h>
3  #include <AsyncTCP.h>
4  #include <ESPAsyncWebServer.h>
```

Initialize the WiFi to be connected to ESP32, define the photosensitive and LED pins; define the threshold and create the server object

```
6  const char *ssid = "XXXXXX";      // 设置WiFi账号 Set up WiFi account
7  const char *password = "XXXXXXXX"; // 设置WiFi密码 Set WiFi password
8  const int ledPin = 13;             // LED控制引脚 LED control pin
9  const int ldrPin = 12;             // 光敏电阻连接的引脚 Photoresistor connected pin
10 int threshold = 4095 * 0.7;        // 如果sensorValue小于70%的阈值 If sensorValue is less than the 70% threshold
11 AsyncWebServer server(80);        // 创建AsyncWebServer对象 Create an AsyncWebServer object
```

Generate the HTML code of the web page .

```
13 // 生成网页的HTML代码 Generates the HTML code for the web page
14 const char index_html[] PROGMEM = R"rawliteral(
15 <!DOCTYPE HTML><html>
16 <head>
17   <meta name="viewport" content="width=device-width, initial-scale=1">
18   <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.2/css/all.css" integrity="sha384-fnmOCqbTlWIl
19   <style>
20     html {
21       font-family: Arial;
22       display: inline-block;
23       margin: 0px auto;
24       text-align: center;
```



Set the LED and photoresistor working modes in the step; set up WiFi connection

```

75 void setup() {
76   Serial.begin(9600);
77   pinMode(ledPin, OUTPUT); // 定义引脚工作模式 Define the pin operation mode
78   digitalWrite(ledPin, LOW); // 确保LED初始状态为关闭 Ensure that the LED is initially off
79   pinMode(ldrPin, INPUT); // 设置光敏电阻引脚为输入模式 Set the photoresistor pin to input mode
80
81   // 设置 WiFi 连接 Setting up WiFi connection
82   WiFi.begin(ssid, password);
83   Serial.println("Connecting to WiFi");
84   while (WiFi.status() != WL_CONNECTED) {
85     delay(1000);
86     Serial.println(".");

```

Set request handling function

```

91 server.on("/", HTTP_GET, [](AsyncWebServerRequest *request) {
92   request->send_P(200, "text/html", index_html);
93 });
94
95 server.on("/brightness", HTTP_GET, [](AsyncWebServerRequest *request) {
96   int sensorValue = analogRead(ldrPin); // 读取光敏电阻的值 (0-4095之间) Read photoresistor value
97   int brightnessPercentage = map(sensorValue, 0, 4095, 0, 100); // 将值映射到百分比范围 (0-100) Map values to a percentage
98   String brightnessString = String(brightnessPercentage); // 将百分比转换为字符串 Convert percentages to strings
99   request->send(200, "text/plain", brightnessString); // 发送亮度百分比到客户端 Send the brightness percentage
100
101   if (sensorValue < threshold) { //如果检测到亮度低于70% If the brightness is detected below 70%
102     digitalWrite(ledPin, HIGH); //点亮LED灯 Light up the LED
103   } else { //否则 otherwise
104     digitalWrite(ledPin, LOW); //熄灭LED灯 Turn off the LED lights

```

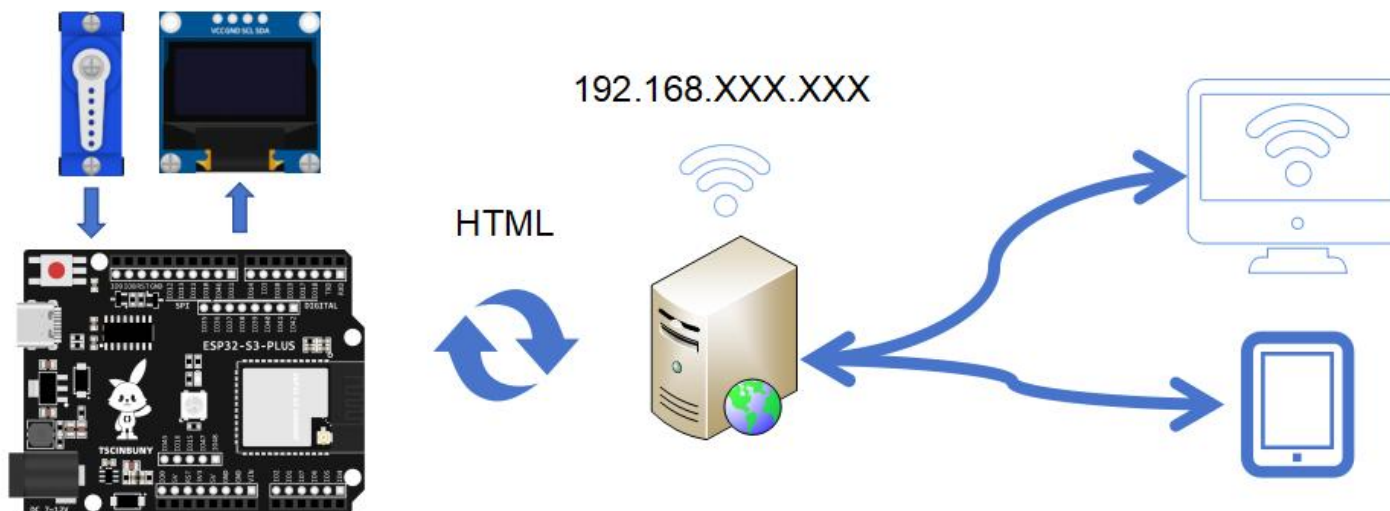
## 20. Web control servo

### 20.1 Overview

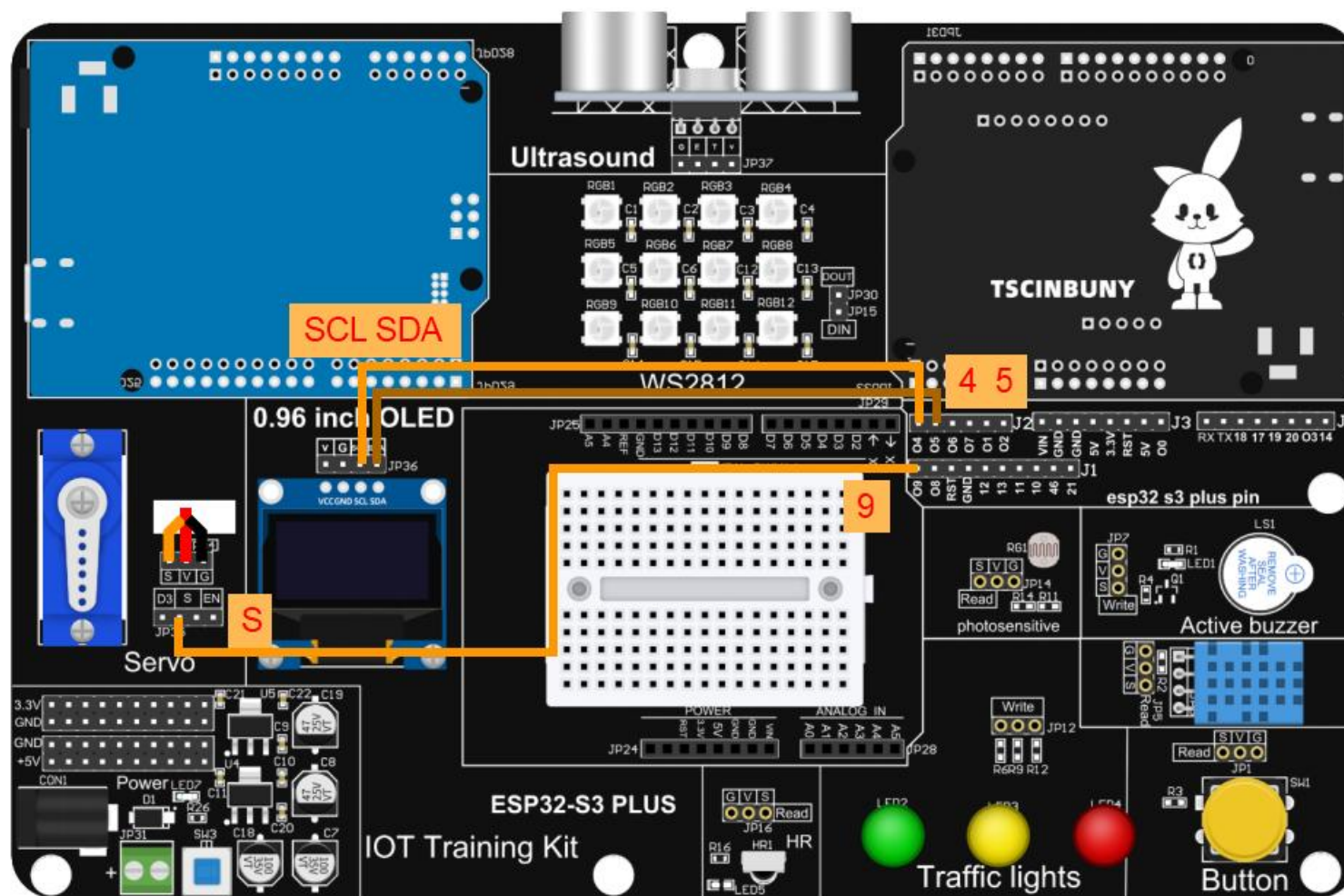
This section focuses on learning the WiFi function of ESP32S3, controlling the servo and displaying the servo angle in real time on the web page.

### 20.2 Working principle

When ESP32 successfully connects to WiFi, push the web page that generates the servo angle and servo control slider to the web service. Use a mobile device to connect to the same WiFi and access the same IP address in the browser. You can then use the web page and The screen obtains the servo angle and controls the servo in real time.

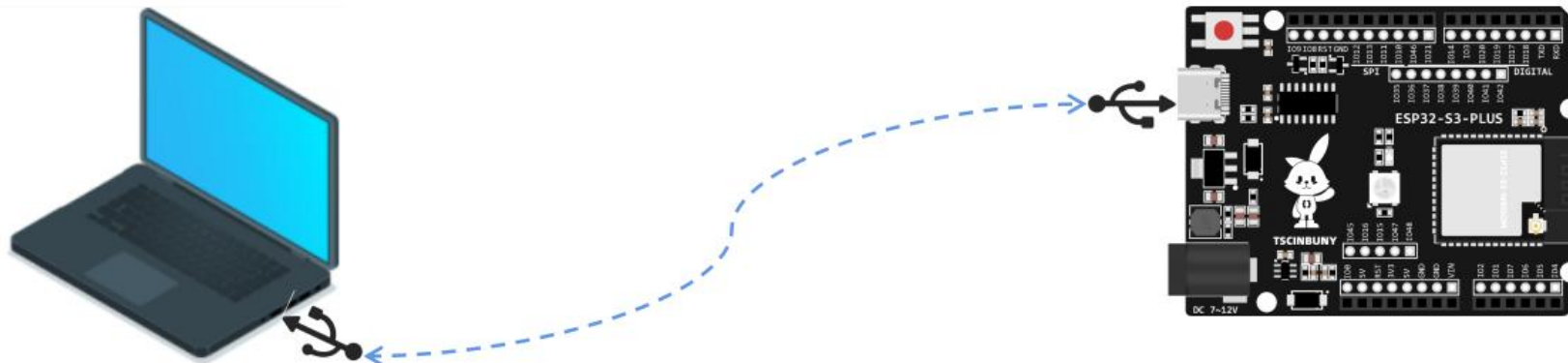


## 20.3 Connection lines



## 20.4 Upload code procedure

### 20.4.1 Connect the main control board to the computer using a USB cable



### 20.4.2 Open the program file (path: 2\_ESP32\_S3\_PLUS \ Lesson\_20\_Web\_control\_steering\_gear\_Angle\_display )

Lesson_14_Web_key_controls_LED	2024/3/27 11:23	文件夹
Lesson_15_Web_server_control_LED	2024/3/27 11:24	文件夹
Lesson_16_Web_server_control_RGB	2024/3/21 18:25	文件夹
Lesson_17_Web_Display_temperature_and_humidity	2024/3/21 18:25	文件夹
Lesson_18_Web_server_control_buzzer_alarm	2024/3/21 18:23	文件夹
Lesson_19_Photoresistor_controls_LED	2024/3/21 18:24	文件夹
Lesson_20_Web_control_steering_gear_Angle_display	2024/3/21 18:26	文件夹
Lesson_21_Web_control_ultrasonic_ranging_display	2024/3/21 18:26	文件夹

Modify the WiFi account and password to which ESP32 is connected in the code. (This WiFi can be the router WiFi at home)



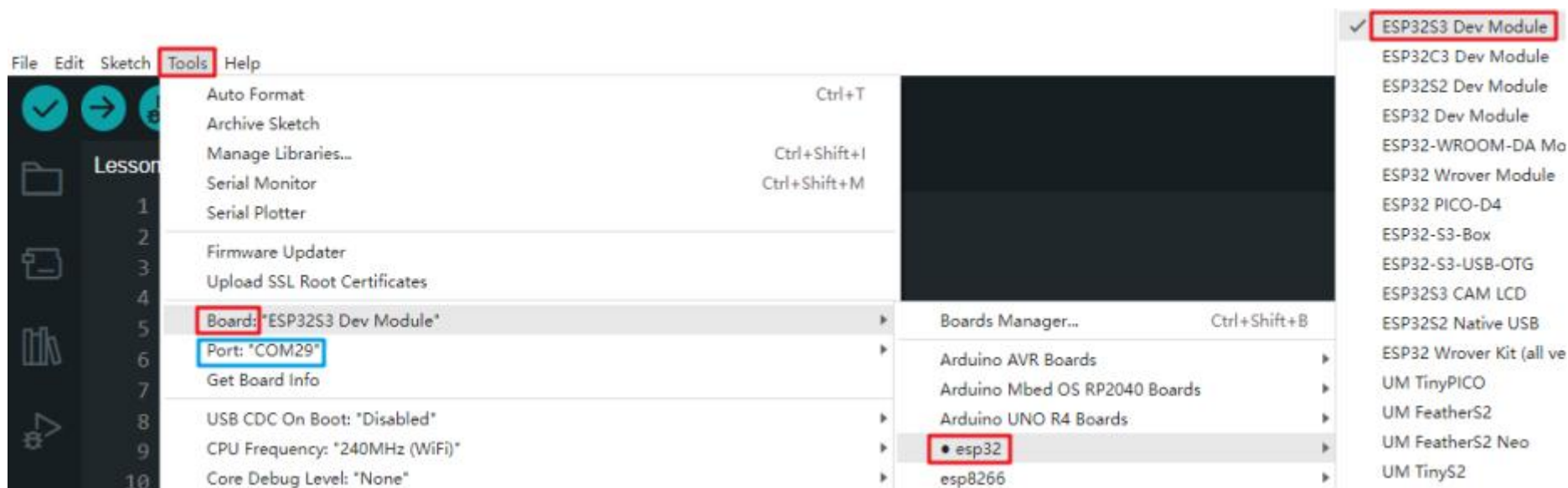
or the WiFi transmitted by the mobile hotspot, but make sure it is 2.4G and not 5G)

```

6  const char *ssid = "xxxxxx"; //输入ESP32要连接的WiFi账号 Enter the
7  const char *password = "xxxxxxxx"; //设置WiFi密码 Set WiFi password

```

Then select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board, and wait for the program upload to be completed .



After the program upload is completed, open the serial monitor to view the IP address. Here the IP is 192.168.8.1, but

everyone will get a different IP.

Use your mobile phone to connect to the same WiFi, then open the browser, enter the IP address above and enter the page to see the lighting conditions and LED conditions in real time.



# ESP32 Servo Control



Servo Angle: 90





## 20.5 Code analysis

Declare the required libraries, which are mainly used to create web services and instance servos and OLED screens.

```
1  #include <Arduino.h>
2  #include <WiFi.h>
3  #include <AsyncTCP.h>
4  #include <ESPAsyncWebServer.h>
5  #include <ESP32Servo.h>
6  #include <Wire.h>
7  #include <Adafruit_GFX.h>
8  #include <Adafruit_SSD1306.h>
```

Define the pixel parameters of the OLED screen and instantiate a screen object display

```
9
10 #define SCREEN_WIDTH 128 // 定义屏幕宽度为128像素 Define the screen width to be 128 pixels
11 #define SCREEN_HEIGHT 64 // 定义屏幕高度为64像素 Define the screen height to be 64 pixels
12 #define OLED_RESET -1 // 定义 OLED 复位引脚为 -1（如果不使用复位功能，则设置为 -1） Define
13 // 创建 Adafruit_SSD1306 对象，用于控制 OLED 屏幕，指定屏幕宽度、高度、I2C 总线对象和复位引脚
14 // Create an Adafruit_SSD1306 object that controls the OLED screen, specifying the screen wi
15 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
```

Initialize the WiFi to be connected to ESP32, set the servo pins and create servo objects and server objects.

```

17 const char *ssid = "ZHIYI";           // 设置WiFi账号 Set up WiFi account
18 const char *password = "zy12345678"; // 设置WiFi密码 Set WiFi password
19 const int servoPin = 9;               // 舵机控制引脚 Servo control pin
20 Servo servo;                         // 实例化舵机对象 Instantiate the steering gear object
21 AsyncWebServer server(80);           // 创建 AsyncWebServer 对象

```

Generate the HTML code of the web page .

```

23 // 生成网页的 HTML 代码 Generates the HTML code for the web page
24 const char index_html[] PROGMEM = R"rawliteral(
25 <!DOCTYPE HTML><html>
26 <head>
27   <meta name="viewport" content="width=device-width, initial-scale=1">
28   <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.2/css/all.css" integrity="sha384-fnmOCqbTlWIlj8LyTjo7m
29   <style>
30     html {
31       font-family: Arial;
32       display: inline-block;
33       margin: 0px auto;

```

Set the Wire pin in the step and initialize the OLED screen

```

71 void setup() {
72   Serial.begin(9600);
73   Wire.begin(5, 4); // SDA引脚设置为5，SCL引脚设置为4
74
75   // 初始化屏幕
76   if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
77     Serial.println(F("SSD1306 initialization failed"));
78     for (;;)
79       ;

```

Initialize screen clearing and initialize display of servo angle to OLED screen

```
82  display.clearDisplay();
83  // 初始化显示舵机角度 Initialize display steering Angle
84  display.setTextSize(2);
85  display.setTextColor(SSD1306_WHITE);
86  display.setCursor(0, 8);
87  display.println("SG90 Angle");
88  display.setCursor(32, 40);
89  display.print("0");
90  display.display();
```

Associate the servo to the web service and set up the WiFi connection

```
92  servo.attach(servoPin); //关联舵机引脚 Associate servo pin
93  // 设置 WiFi 连接
94  WiFi.begin(ssid, password);
95  Serial.println("Connecting to WiFi");
96  while (WiFi.status() != WL_CONNECTED) {
97      delay(1000);
98      Serial.println(".");
99  }
100 Serial.println(WiFi.localIP()); //打印IP地址
```

Set the request processing function and display the servo angle to the OLED screen at the same time

```
102 server.on("/", HTTP_GET, [](AsyncWebServerRequest *request) {
103     request->send_P(200, "text/html", index_html);
104 });
105
106 server.on("/servo", HTTP_GET, [](AsyncWebServerRequest *request) {
107     if (request->hasParam("angle")) {
108         int angle = request->getParam("angle")->value().toInt();
109         servo.write(angle);
110         display.clearDisplay();
111
112         // 显示舵机角度
113         display.setTextSize(2);
114         display.setTextColor(SSD1306_WHITE);
115         display.setCursor(0, 8);
116         display.println("SG90 Angle");
117         display.setCursor(32, 40);
118         display.print(angle);
119         display.display();
120         request->send(200, "text/plain", "OK");
121     } else {
122         request->send(400, "text/plain", "Missing angle parameter");
123     }
124 });
```

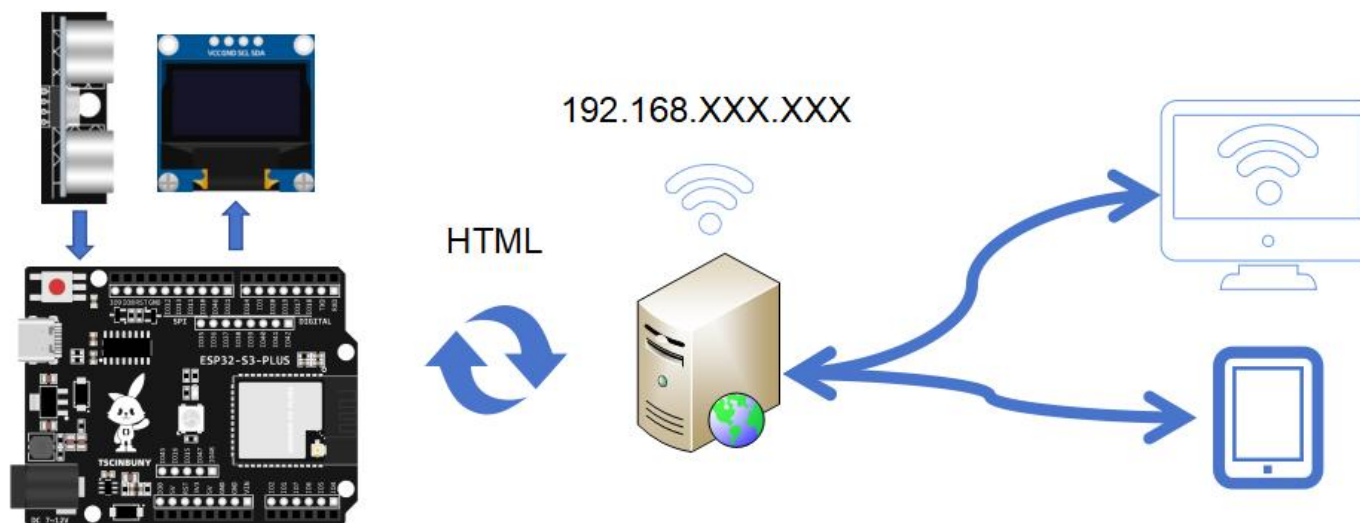
## 21. WebDisplayUltrasound

### 21.1 Overview

This section focuses on learning the WiFi function of ESP32S3 and displaying the distance measured by ultrasonic waves on the web page.

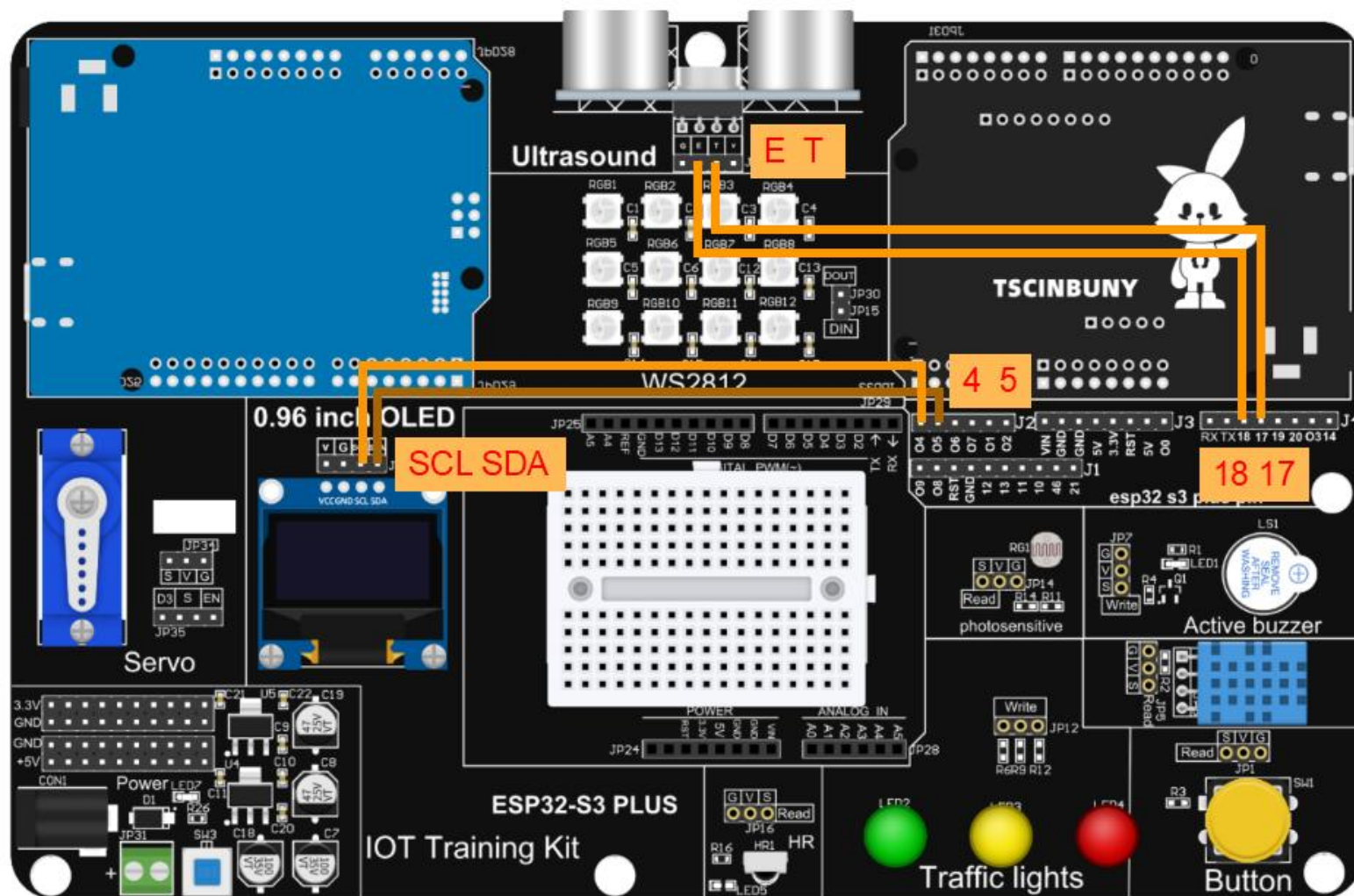
### 21.2 Working principle

When ESP32 successfully connects to WiFi, push the web page that generates the ultrasonic distance measurement to the web service, connect to the same WiFi with a mobile device, and access the same IP address in the browser, you can obtain the ultrasonic measurement results in real time through the web page and screen. distance.





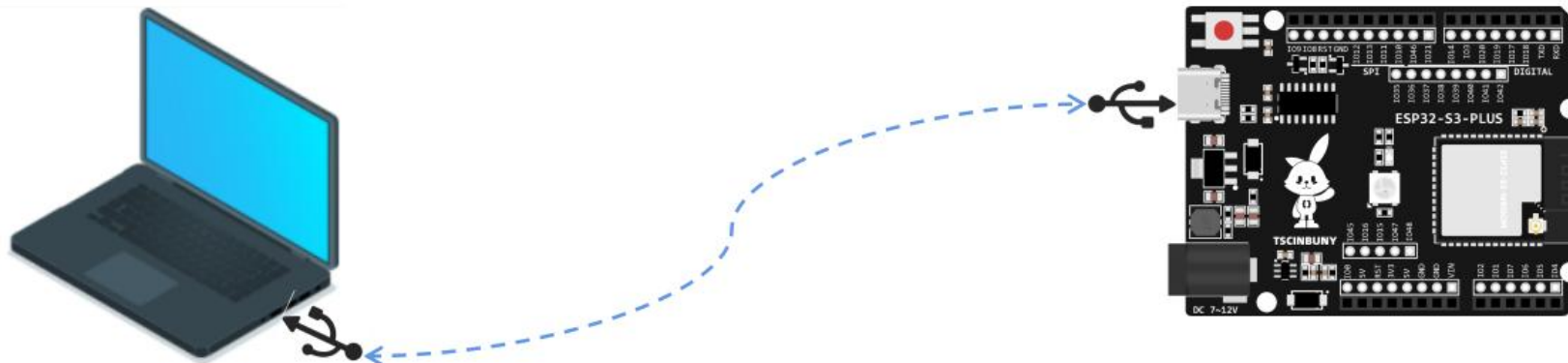
## 21.3 Connection lines





## 21.4 Upload code program

### 21.4.1 Connect the main control board to the computer using USB cable



### 21.4.2 Open the program file (path: 2\_ESP32\_S3\_PLUS \ Lesson\_21\_Web\_control\_ultrasonic\_ranging\_display )

Lesson_14_Web_key_controls_LED	2024/3/27 11:23	文件夹
Lesson_15_Web_server_control_LED	2024/3/27 11:24	文件夹
Lesson_16_Web_server_control_RGB	2024/3/21 18:25	文件夹
Lesson_17_Web_Display_temperature_and_humidity	2024/3/21 18:25	文件夹
Lesson_18_Web_server_control_buzzer_alarm	2024/3/21 18:23	文件夹
Lesson_19_Photoresistor_controls_LED	2024/3/21 18:24	文件夹
Lesson_20_Web_control_steering_gear_Angle_display	2024/3/21 18:26	文件夹
Lesson_21_Web_control_ultrasonic_ranging_display	2024/3/21 18:26	文件夹

Modify the WiFi account and password to which ESP32 is connected in the code. (This WiFi can be the router WiFi at home)

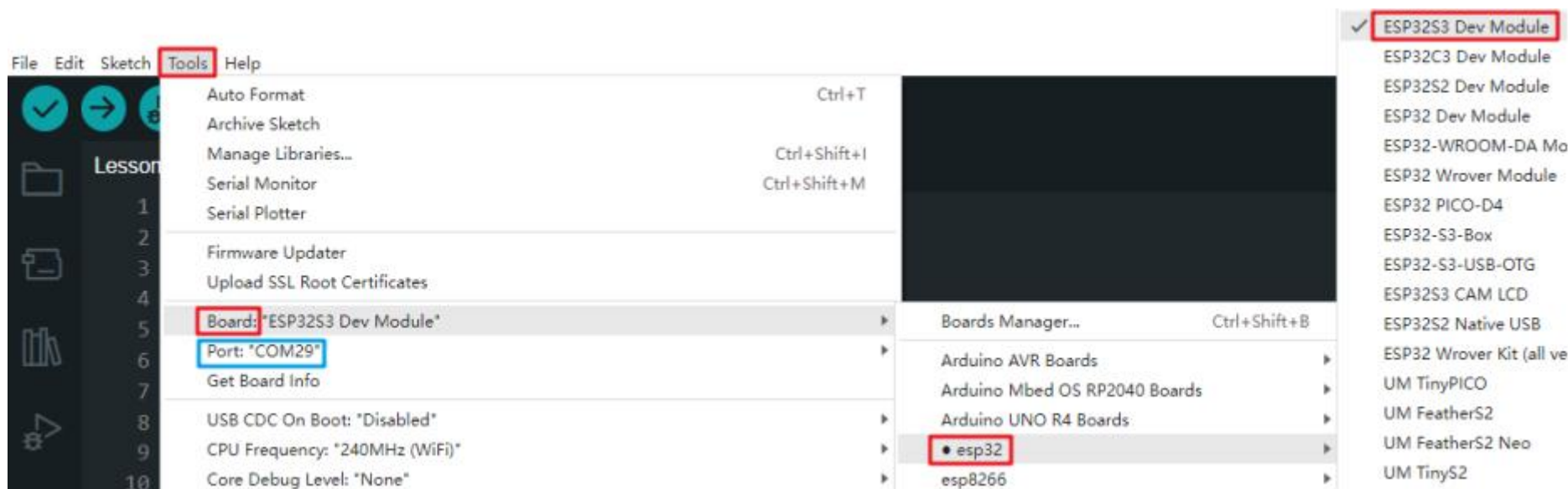
or the WiFi transmitted by the mobile hotspot, but make sure it is 2.4G and not 5G)

```

6  const char *ssid = "xxxxxx"; //输入ESP32要连接的WiFi账号 Enter th
7  const char *password = "xxxxxxxx"; //设置WiFi密码 Set WiFi password

```

Then select the board type as ESP32S3 Dev Module and select the COM number newly displayed when the USB is plugged in . Then click "Upload" to start compiling and uploading the program to the main control board, and wait for the program upload to be completed .



After the program upload is completed, open the serial monitor to view the IP address. Here the IP is 192.168.8.1, but

everyone will get a different IP.

Use your mobile phone to connect to the same WiFi, then open the browser, enter the IP address above and enter the page to see the ultrasonic ranging situation in real time.



# ESP32 Ultrasonic

 Distance: 19 cm

## 21.5 Code analysis

Declare the required libraries, mainly used to create web services and instance OLED screens

```
1  #include <Arduino.h>
2  #include <WiFi.h>
3  #include <AsyncTCP.h>
4  #include <ESPAsyncWebServer.h>
5  #include <Wire.h>
6  #include <Adafruit_GFX.h>
7  #include <Adafruit_SSD1306.h>
```

Define the pixel parameters of the OLED screen and instantiate a screen object display

```
9
10 #define SCREEN_WIDTH 128 // 定义屏幕宽度为128像素 Define the screen width to be 128 pixels
11 #define SCREEN_HEIGHT 64 // 定义屏幕高度为64像素 Define the screen height to be 64 pixels
12 #define OLED_RESET -1 // 定义 OLED 复位引脚为 -1（如果不使用复位功能，则设置为 -1） Define
13 // 创建 Adafruit_SSD1306 对象，用于控制 OLED 屏幕，指定屏幕宽度、高度、I2C 总线对象和复位引脚
14 // Create an Adafruit_SSD1306 object that controls the OLED screen, specifying the screen wi
15 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
```

Initialize the WiFi to be connected to ESP32, set the ultrasonic pin and create the server object

```

17  const char *ssid = "ZHIYI";           // 设置WiFi账号 Set up WiFi account
18  const char *password = "zy12345678";  // 设置WiFi密码 Set WiFi password
19  const int Trig = 17;                  // 超声波传感器的触发引脚 Trigger pin of ultrasonic sensor
20  const int Echo = 18;                  // 超声波传感器的回声引脚 The echo pin of the ultrasonic sensor
21  // 创建 AsyncWebServer 对象
22  AsyncWebServer server(80);

```

Generate the HTML code of the web page .

```

24  // 生成网页的 HTML 代码 Generates the HTML code for the web page
25  const char index_html[] PROGMEM = R"rawliteral(
26  <!DOCTYPE HTML><html>
27  <head>
28    <meta name="viewport" content="width=device-width, initial-scale=1">
29    <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.2/css/all.css" integrity="sha384-
30    <style>
31    html {
32      font-family: Arial;
33      display: inline-block;
34      margin: 0px auto;
35      text-align: center;
36    }
37    h2 { font-size: 3.0rem; }
38    p { font-size: 3.0rem; }
39    .units { font-size: 1.2rem; }

```



The function `GetDistance` returns the distance value measured by ultrasonic wave

```

77 float GetDistance() //获取超声波传感器数值 Get ultrasonic sensor values
78 {
79     float distance;
80     //发送一个低短脉冲到Trig触发测距
81     digitalWrite(Trig, LOW); //发送一个低电平到Trig
82     delayMicroseconds(2);
83     digitalWrite(Trig, HIGH);
84     delayMicroseconds(10);
85     digitalWrite(Trig, LOW);
86     distance = pulseIn(Echo, HIGH) / 58.00;
87     //Serial.print("Distance = ");
88     //Serial.println(distance); //串行输出距离转换成厘米
89     delay(10);
90     return distance;

```

Set the ultrasonic pin working mode and initialize Wire in the step

```

93 void setup() {
94     Serial.begin(9600);
95     pinMode(Trig, OUTPUT); //定义引脚工作模式为输入 Define the pin operation mode as the input
96     pinMode(Echo, INPUT); //定义引脚工作模式为输出 Define the pin operation mode as output
97     Wire.begin(5, 4); //初始化 I2C 总线 SDA引脚设置为5, SCL引脚设置为4 The initial I2C bus SDA

```



Clear screen buffer and display ultrasonic distance

```
106 // 清空屏幕缓冲区 Clear the screen buffer
107 display.clearDisplay();
108
109 // 显示超声波距离 Display ultrasonic distance
110 display.setTextSize(2);
111 display.setTextColor(SSD1306_WHITE);
112 display.setCursor(16, 8);
113 display.println("Distance");
114 display.setCursor(16, 40);
115 display.print("0");
116 display.print(" cm");
117 display.display();
```

Set up WiFi connection

```
119 // 设置 WiFi 连接 Setting up WiFi connection
120 WiFi.begin(ssid, password);
121 Serial.println("Connecting to WiFi");
122 while (WiFi.status() != WL_CONNECTED) {
123     delay(1000);
124     Serial.println(".");
125 }
126
127 Serial.println(WiFi.localIP()); //打印IP地址
```

Set the request processing function and display the ultrasonic measurement distance value to the OLED screen at the same time.

```
133 server.on("/ultrasonic", HTTP_GET, [](AsyncWebServerRequest *request) {
134     float distance = GetDistance();
135     if (distance >= 400 || distance <= 2) {
136         distance = -1; // 超出范围或错误值
137     }
138     request->send(200, "text/plain", String(distance));
139
140     // 清空屏幕缓冲区
141     display.clearDisplay();
142
143     // 显示超声波距离
144     display.setTextSize(2);
145     display.setTextColor(SSD1306_WHITE);
146     display.setCursor(16, 8);
147     display.println("Distance");
148     display.setCursor(16, 40);
149     display.print(distance);
150     display.print(" cm");
151     display.display();
```