



TSCIN**BUNY**

ZYE0011-UNO R4 MINIMA

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 a tutorial for UNO R4 Minima, divided into 13 sections.

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_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

Start by understanding the main control board and creating a development environment. Follow this tutorial to guide you in correctly connecting lines and uploading code to realize project functions. The "Lesson_1_Arduino IDE" folder stores CH340 driver files, library files, etc. It is a must-see. 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 enter the interesting world of Arduino programming!

Table of contents

1. Create a development environment.....	9
1.1. Overview:	9
1.2.Install Arduino IDE	9
Arduino IDE on Mac OS X system.....	17
1.3. Install CH340 driver	20
1.4. Add UNO R4 board in Arduino IDE	23
1.5. Add "Library" in Arduino IDE	25
2.Light up the LED	29
2.1.Overview	29
2.2.Control board resources	29
2.3.Connect the line	33
2. 4. Upload code program	34
2.5. Code analysis	37
3. Button control LED	39
3.1.Overview	39
3.2. Working principle	39

3.2.1.LED	39
3.3.Connect the lines	40
3.4.Upload code	41
3.4.1. Connect the main control board to the computer with a USB cable	41
3.5. Code analysis	43
4. Active buzzer	45
4.1.Overview	45
4.2. Working principle	45
4.3.Connect lines	46
4.4.Upload code	47
4.5. Code analysis	48
5. Traffic lights	49
5.1 Overview	49
5.2. Working principle	49
5.3 Connection lines	50
5.4 Upload code	51
5.5 Code analysis	52

6. Flowing water lamp	55
6.1. Overview	55
6.2. Working principle	55
6.3 Connection lines	55
6.4 Upload code	56
6.4.1 Connect the main control board to the computer using a USB cable	56
6.5 Code analysis	57
7. WS2812B	59
7.1. Overview	59
7.2. Working principle	59
7.3. Connect the lines	60
7.4. Upload code	61
7.5. Code analysis	62
8. Gradient RGB	64
8.1. Overview	64
8.2. Working principle	64
8.3 Connection lines	64

8.4. Upload code program	65
8.5 Code analysis	66
9.Servo control	67
9.1. Overview	67
9.2 Working principle	67
9.3 Connection lines	68
9.4 Upload code program	69
9.5 Code analysis	70
10. Photoresistor	72
10.1 Overview	72
10.2 Working principle	72
10.2.1 Photoresistor	72
10.3 Connection lines	73
10.4 Upload code program	74
10.4.1 Connect the main control board to the computer using a USB cable	74
10.5 Code analysis	77
11. Ultrasonic ranging	78

11.1 Overview	78
11.2 Working principle	78
11.2.1. Ultrasonic sensor	78
11.2.2.OLED	81
11.3 Connection lines	82
11.4 Upload code program	83
11.4.1 Connect the main control board to the computer with a USB cable	83
11.5 Code analysis	85
12. DHT11 temperature and humidity sensor	88
12.1. Overview	88
12.2. Working principle	88
12.3 Connection lines	90
12.4 Upload code program	91
12.5 Code analysis	93
13. Infrared remote control RGB	95
13.1 Overview	95
13.2 Working principle	95

13.3 Connection lines	97
13.4 Upload code program	98
13.5 Code analysis	100

1. Create a development environment

1.1. Overview:

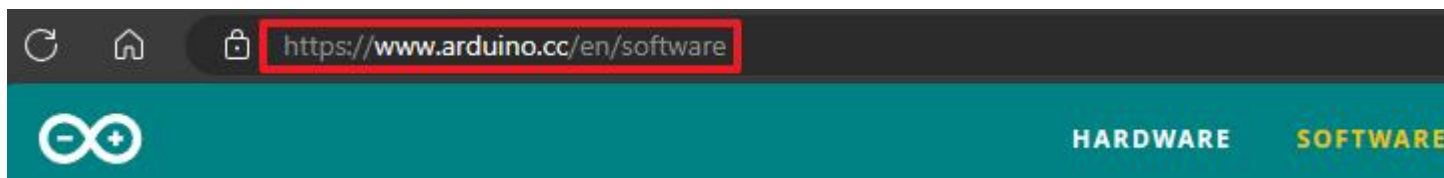
First, let's understand what Arduino is and then create a development environment. Install Arduino IDE as the development editor, and then install the necessary board and library files.

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)

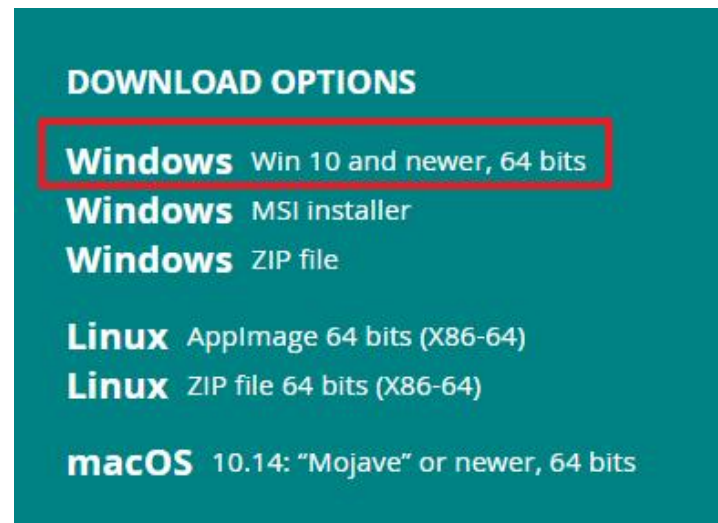
macOS 10.14: "Mojave" or newer, 64 bits

(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

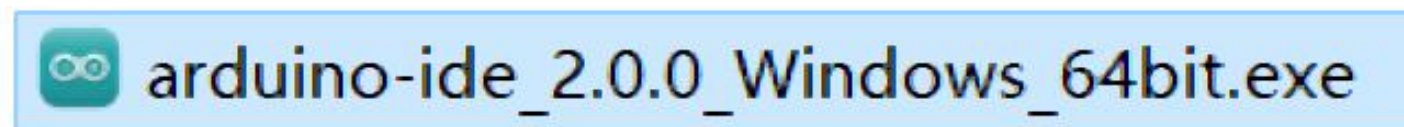
installation.

Click "Windows Win10 and newer"



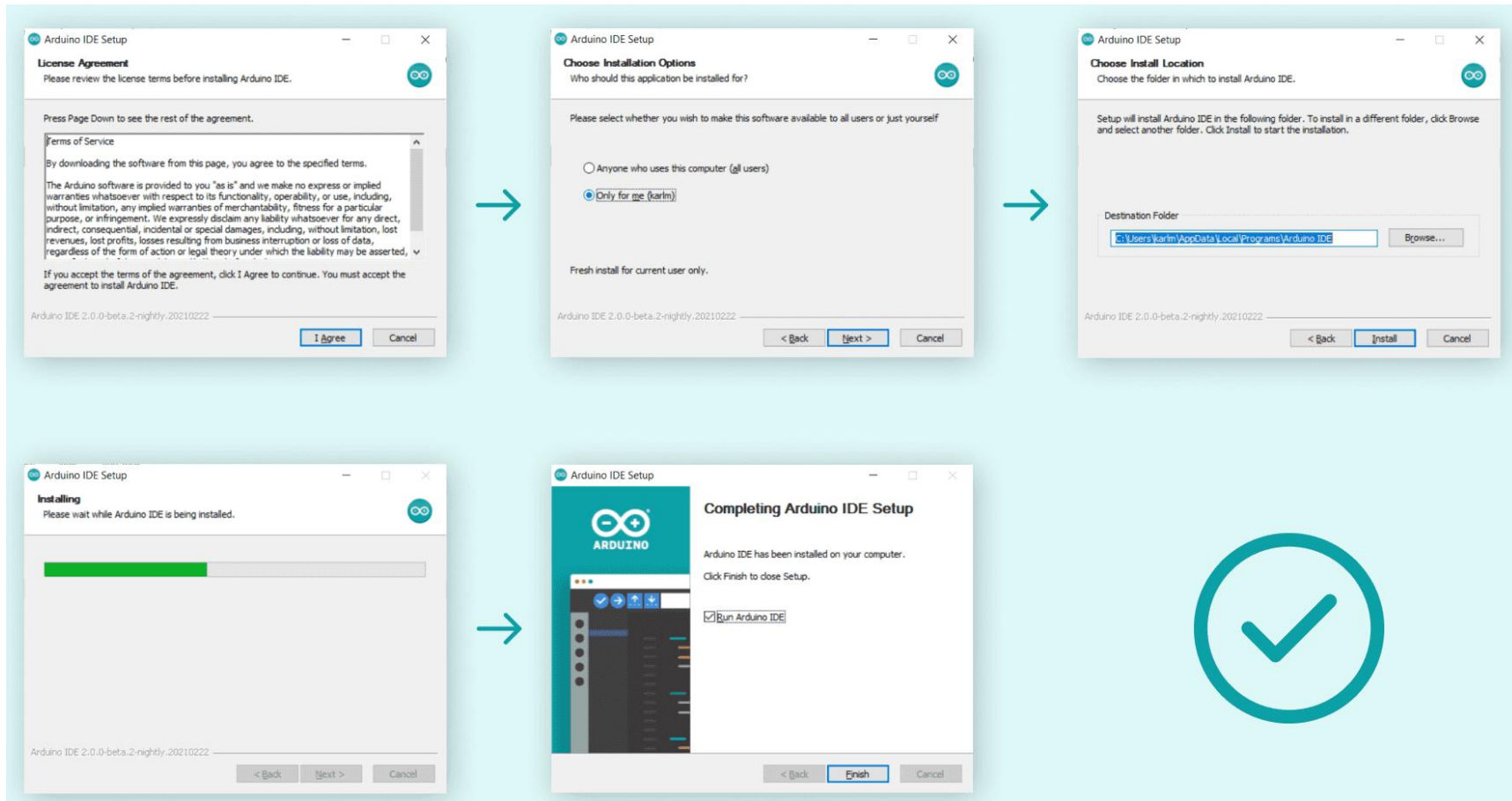
Click "JUST DOWNLOAD".

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

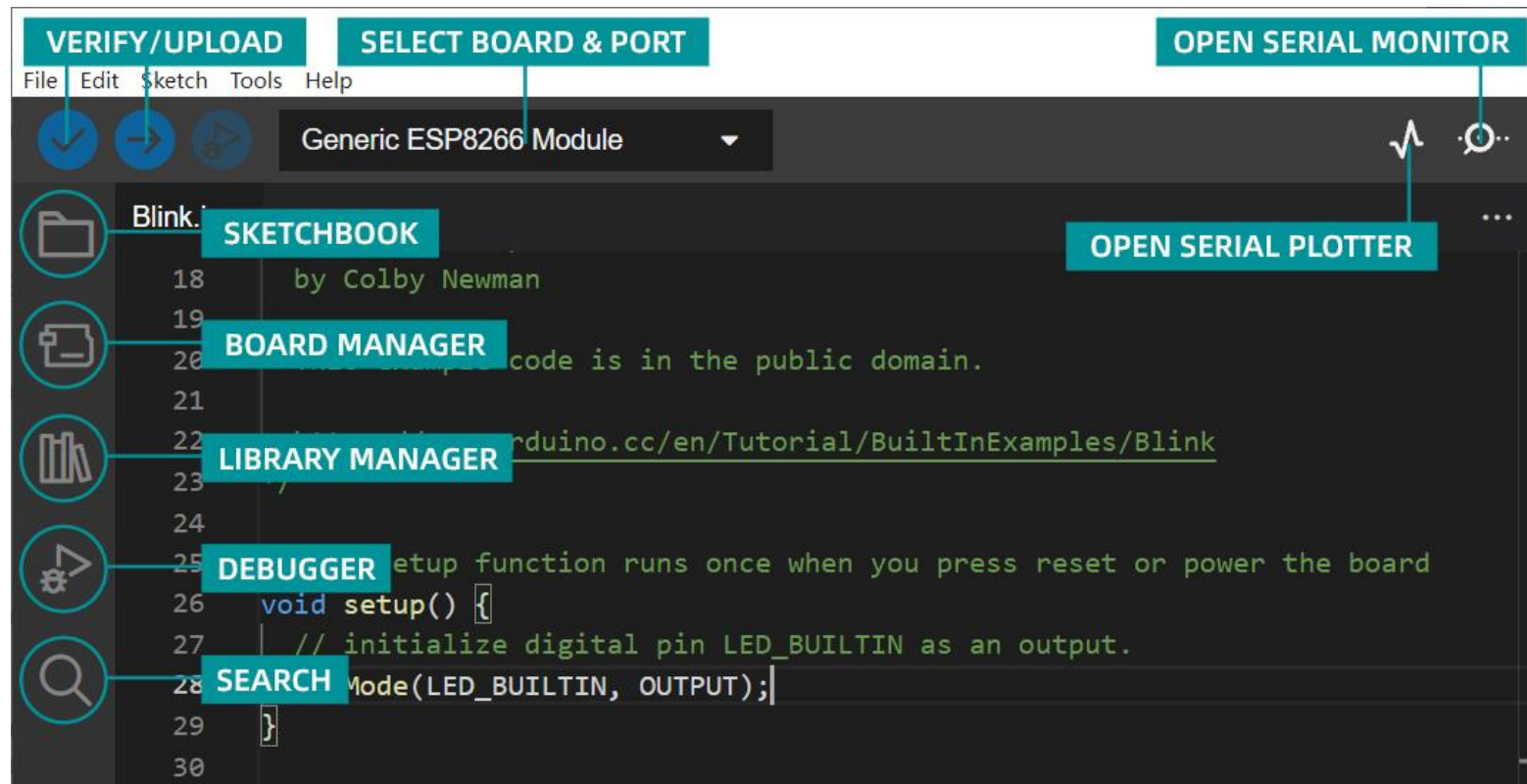


You can press "Browse..." to select the installation path or directly enter the directory you want. Then click "Install" to install. (For Windows users, the driver installation dialog box may pop up during the installation process . When it pops up, please allow the installation)



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

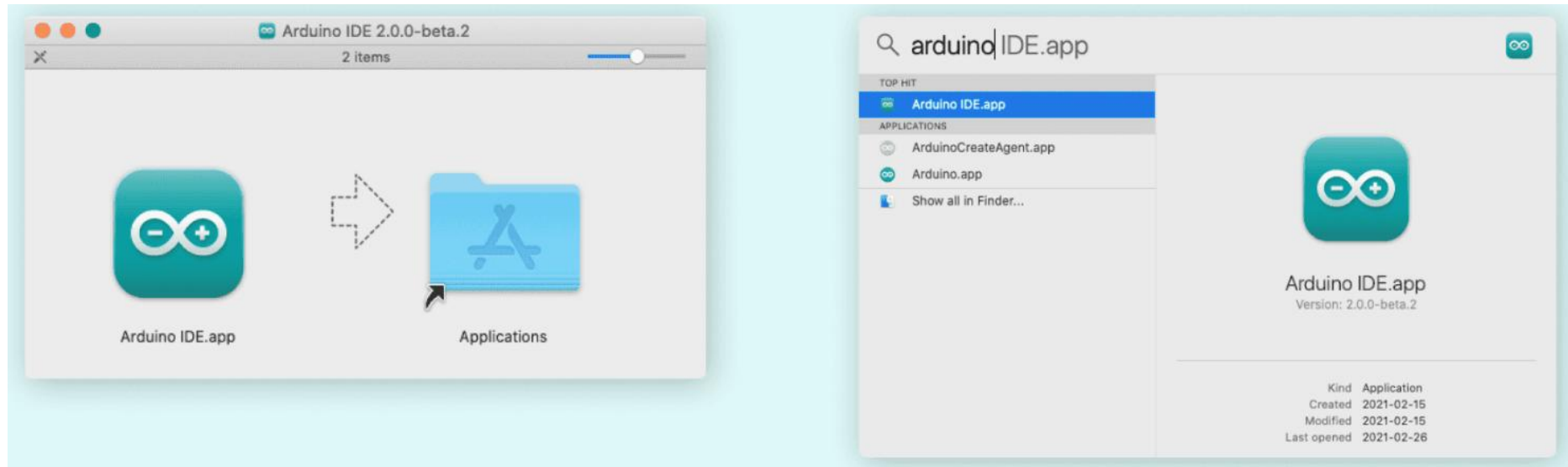
Search - Search for keywords in code

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

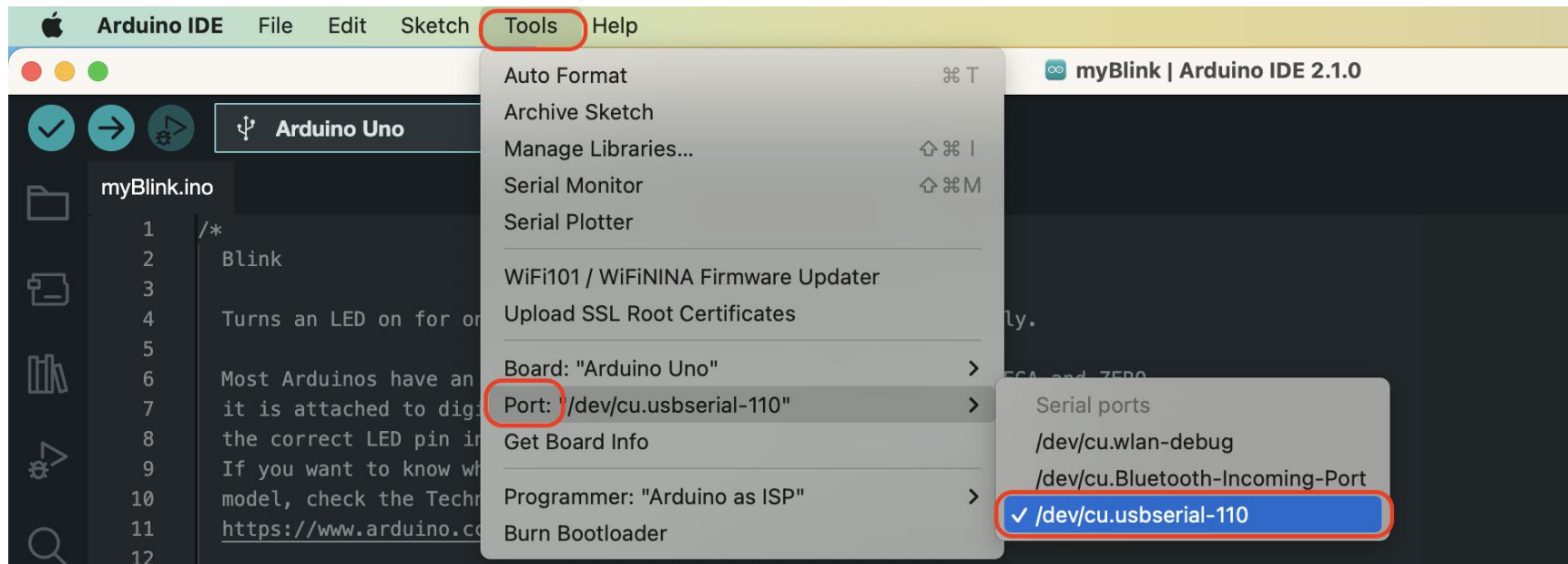
Programs written using the Arduino software (IDE) are called "Sketch". These "Sketch" are written in a text editor and saved with the file extension " .ino ". It is worth noting that the "ino" file must be saved in a folder with the same name. If the program is not opened in a folder with the same name,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



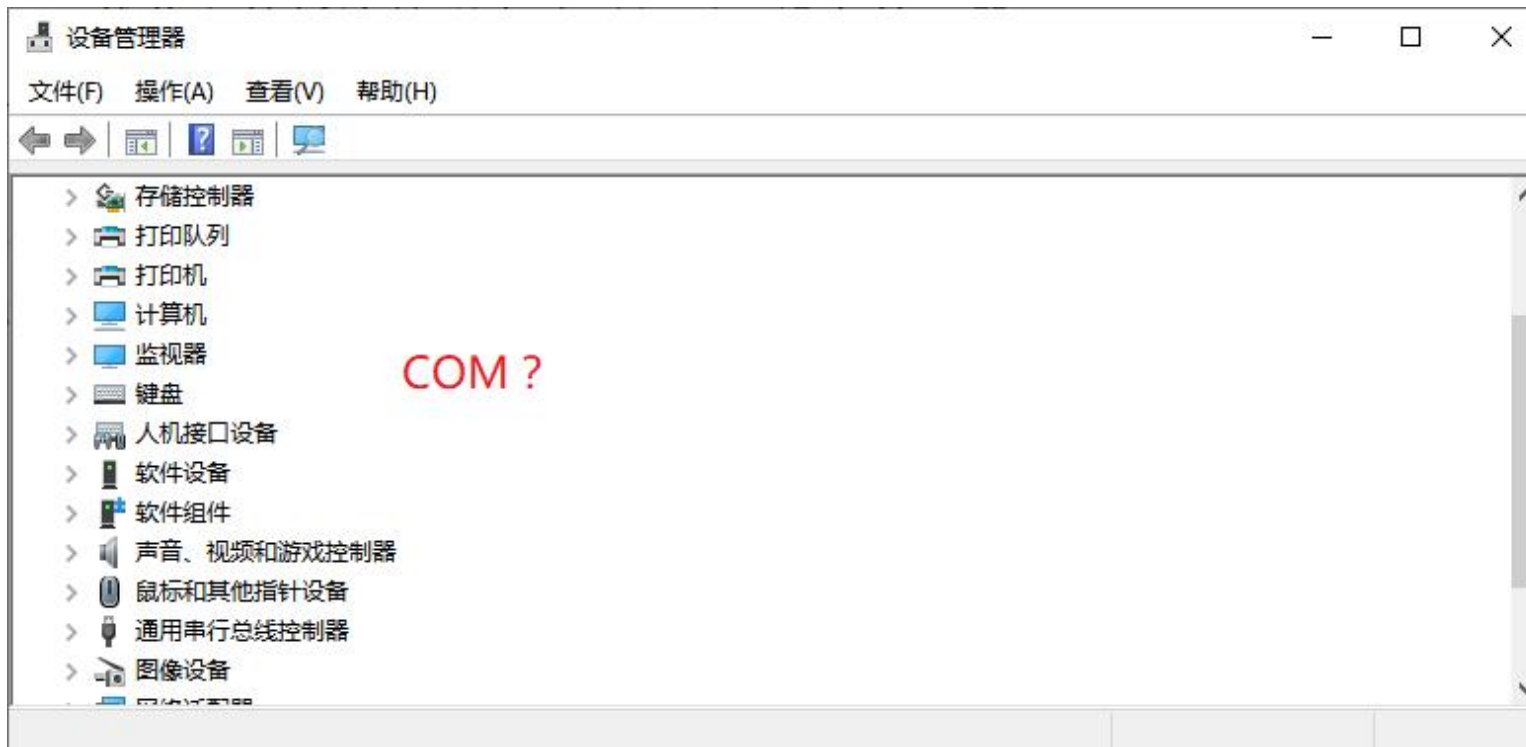
If you don't see "usbserial", you need to install the CH340 driver, please see the next section.

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)

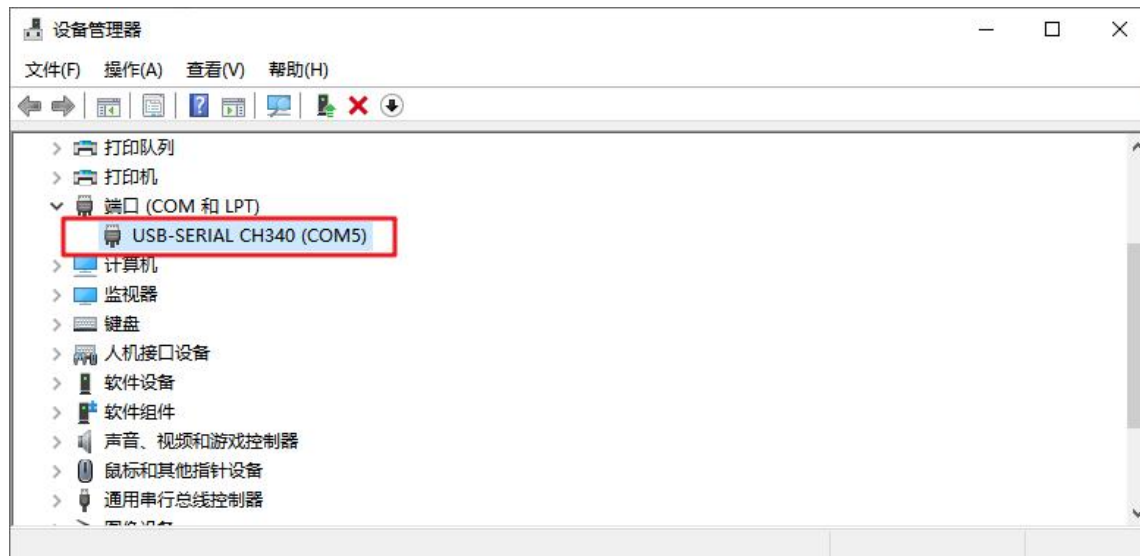


If you do not see the CH340 serial port in the picture above, you need to follow the following steps to install the driver.

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

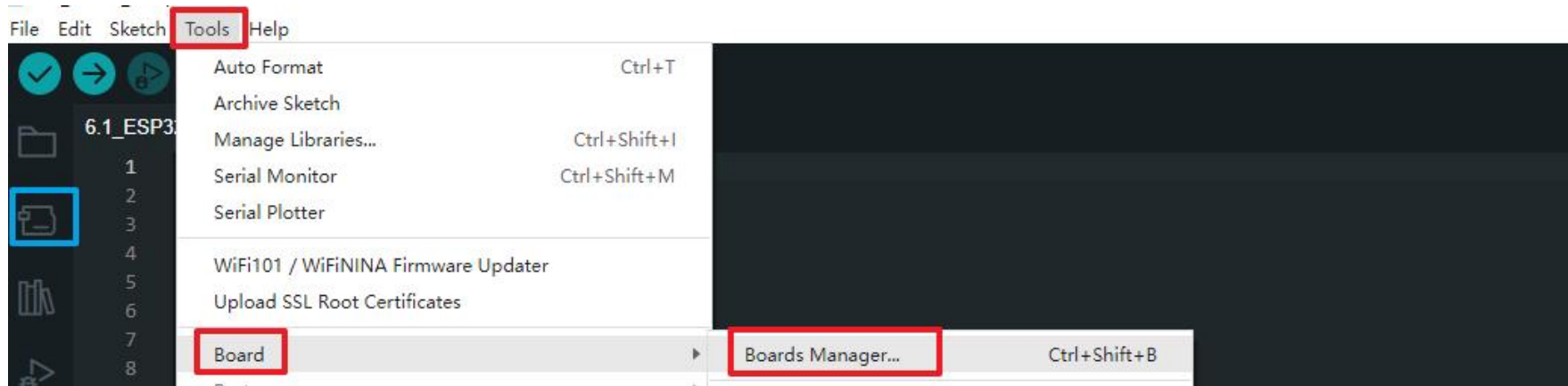
Open the folder 2_CH340 Driver\CH340 Driver File-MAC and double-click to install the pkg file. If you want to update the version, you can check how to download the updated driver through the FAQ.

 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 ^			
 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

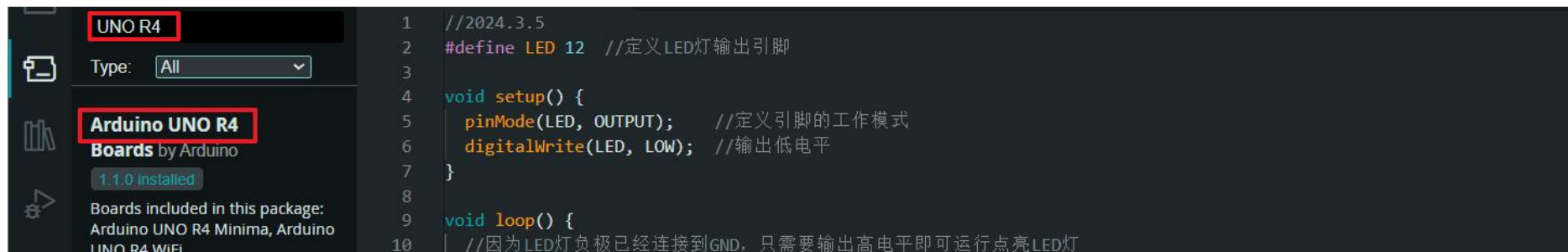
1.4. Add UNO R4 board in Arduino IDE

1.4.1. Search and install UNO R4

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

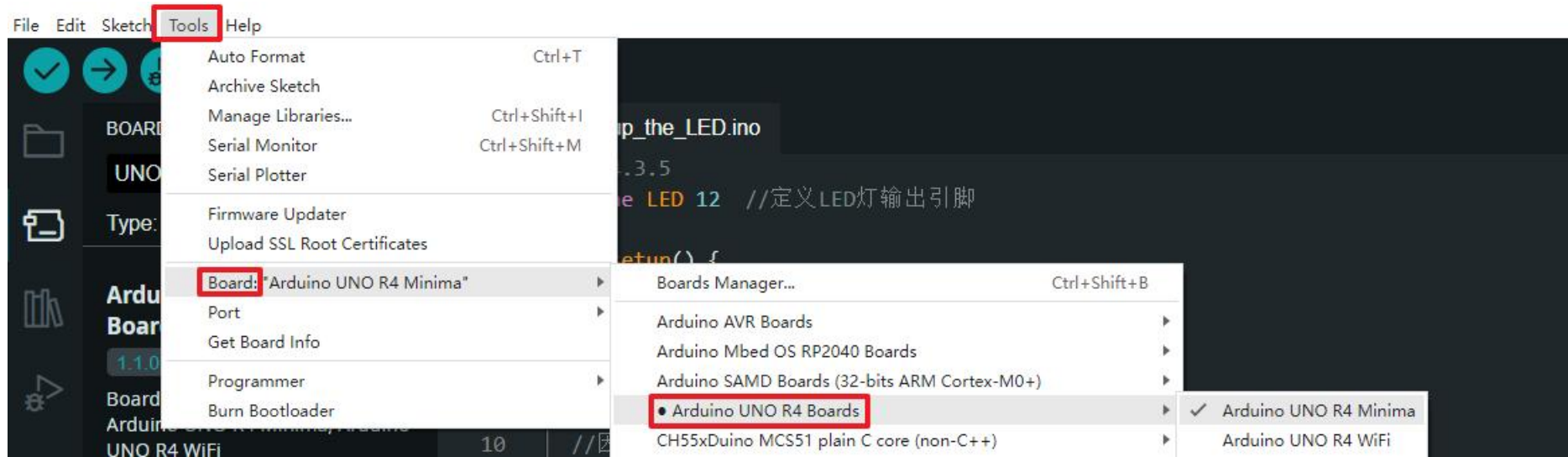


Enter "UNO R4" in the search bar and click "install" to install (make sure the network is open and wait for the installation to complete)



1.4.2. Check whether the board is installed successfully

You can see "UNO R4" 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?

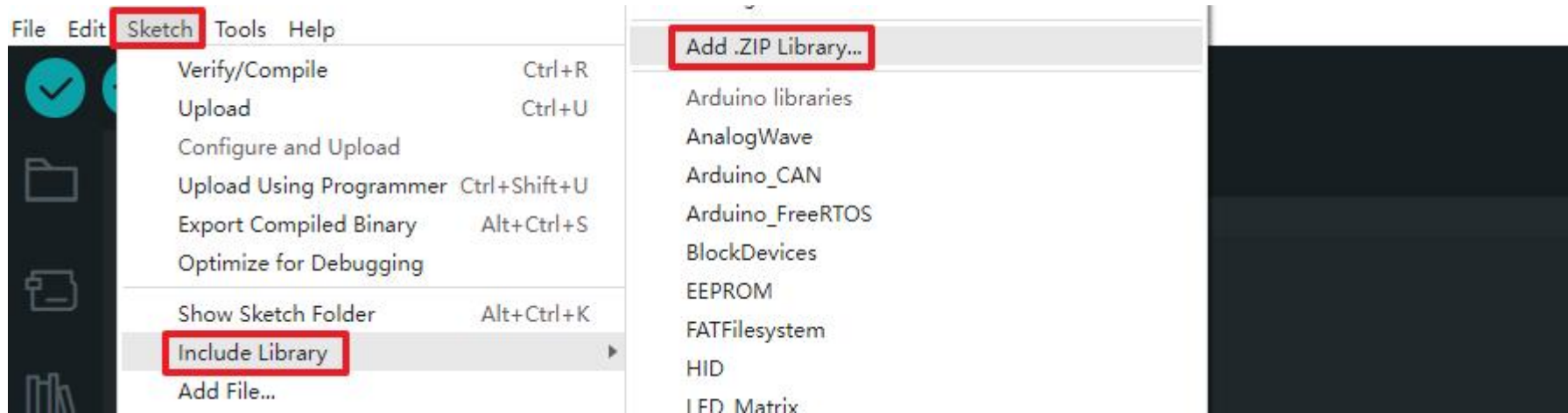
A library is a set of code that allows you to easily connect to sensors, displays, modules, and more. For example, the LiquidCrystal library allows you to easily interact with character LCD displays. There are thousands of libraries available for download directly through the Arduino IDE, all of which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

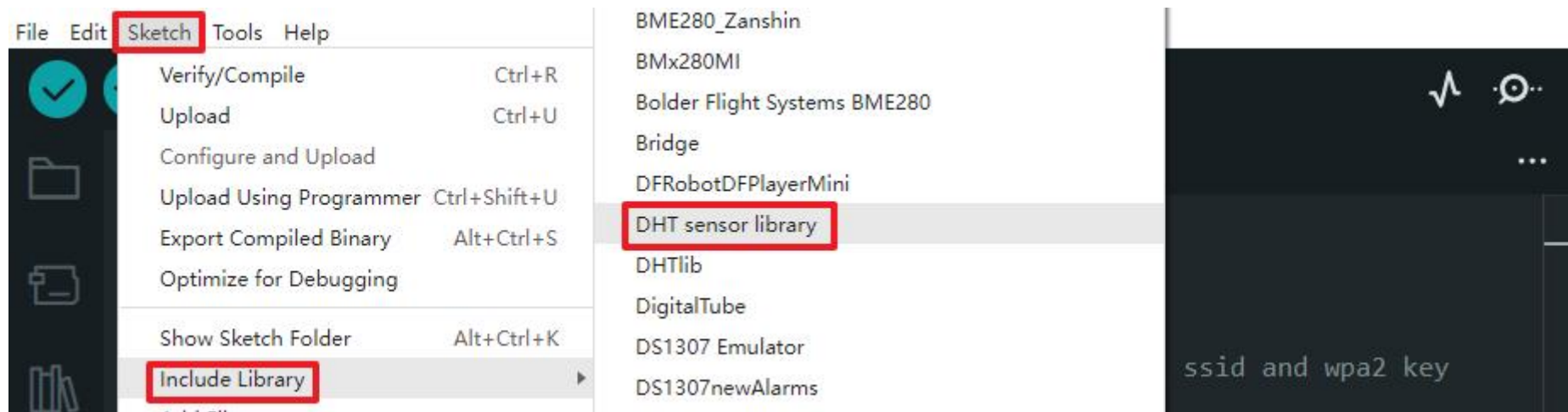


The system will prompt you to select the library to be added . Navigate to the path location of the *DHT_sensor_library.zip* file saved in your computer (*Lesson_1_Arduino IDE\1_Libraries\DHT_sensor_library.zip*) as shown below and open it .












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...	24 KB
 Adafruit_GFX_Library.zip	2024/3/13 15:53	WinRAR ZIP 压缩...	343 KB
 Adafruit_SSD1306.zip	2024/3/13 15:45	WinRAR ZIP 压缩...	36 KB
 Adafruit_Unified_Sensor.zip	2023/12/21 10:14	WinRAR ZIP 压缩...	15 KB
 DHT_sensor_library.zip	2023/12/19 13:57	WinRAR ZIP 压缩...	18 KB
 ESPAsyncTCP.zip	2023/12/21 10:14	WinRAR ZIP 压缩...	44 KB
 ESPAsyncWebServer-master.zip	2023/12/21 10:14	WinRAR ZIP 压缩...	281 KB
 FastLED.zip	2024/2/28 17:49	WinRAR ZIP 压缩...	525 KB
 Servo.zip	2024/3/13 15:02	WinRAR ZIP 压缩...	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.UNO R4 main control board

The main control board provides a total of 14 digital pins D0~D13, which can perform input/output functions;

6 analog input pins A0~A5;

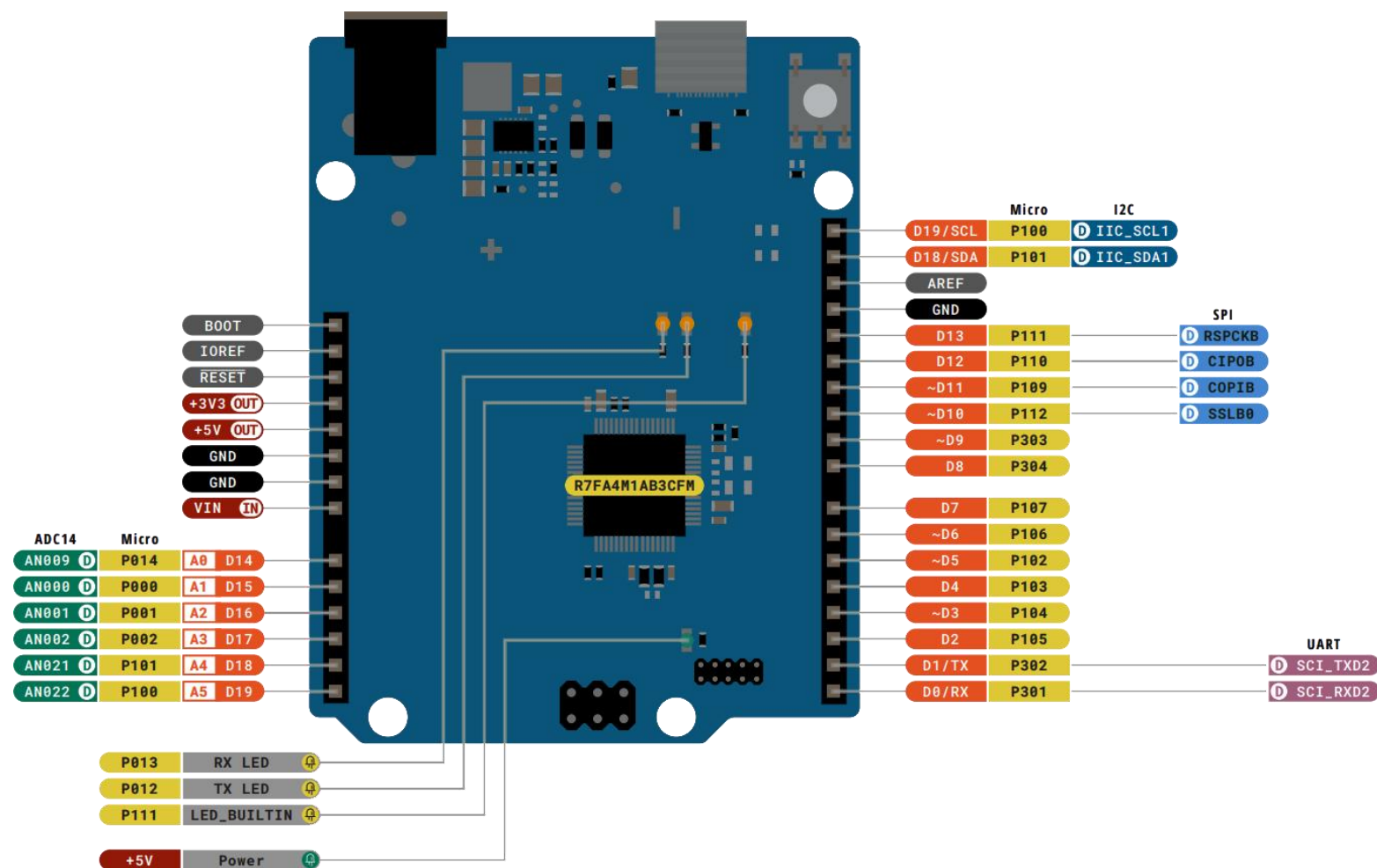
UART communication: D0/D1 pin;

SPI communication: D10~D13 pins;

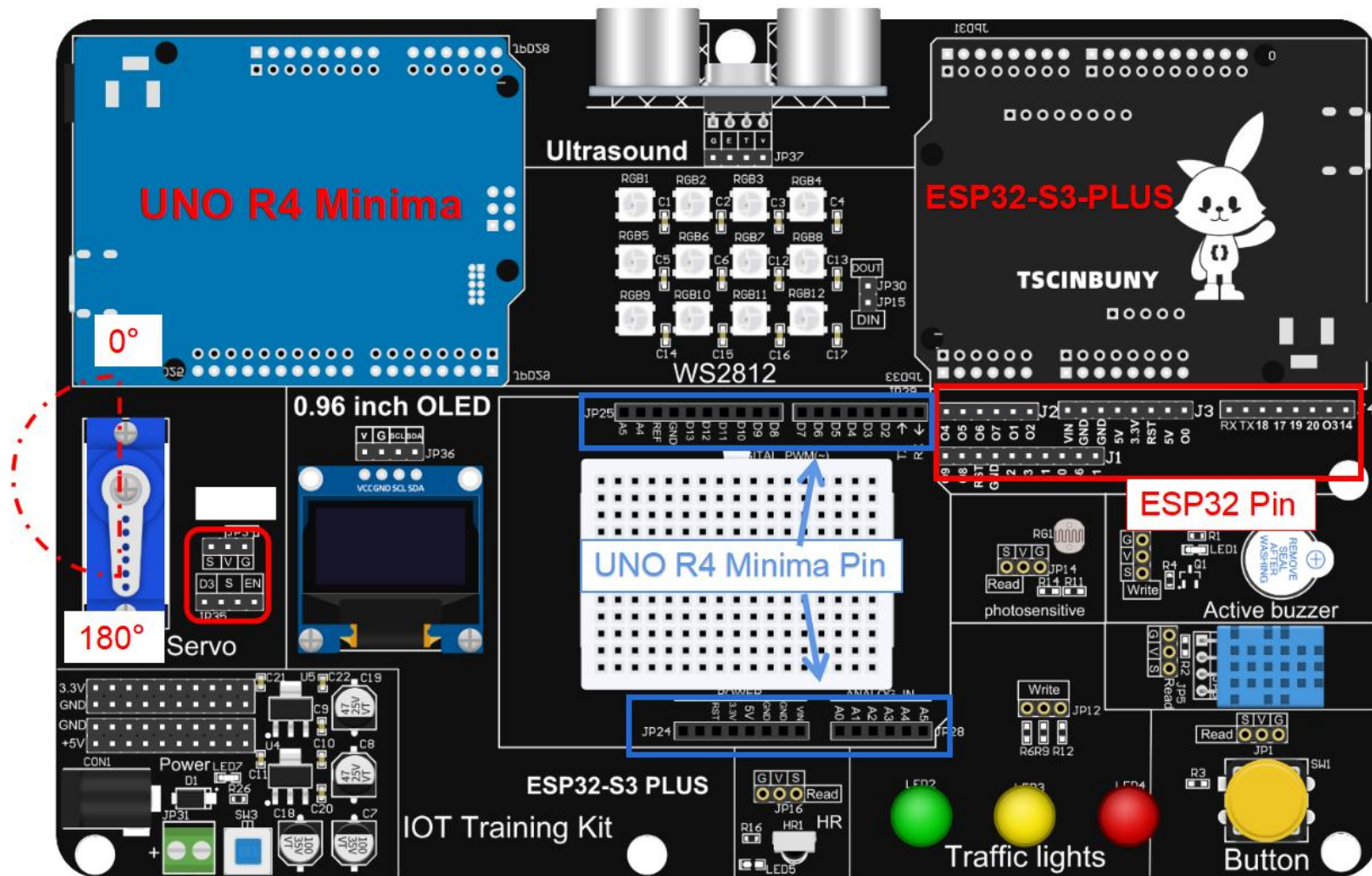
I2C communication: A4/A5(SDA/SCL)

Type-C input voltage is 5V;

DC input voltage 7V~9V;



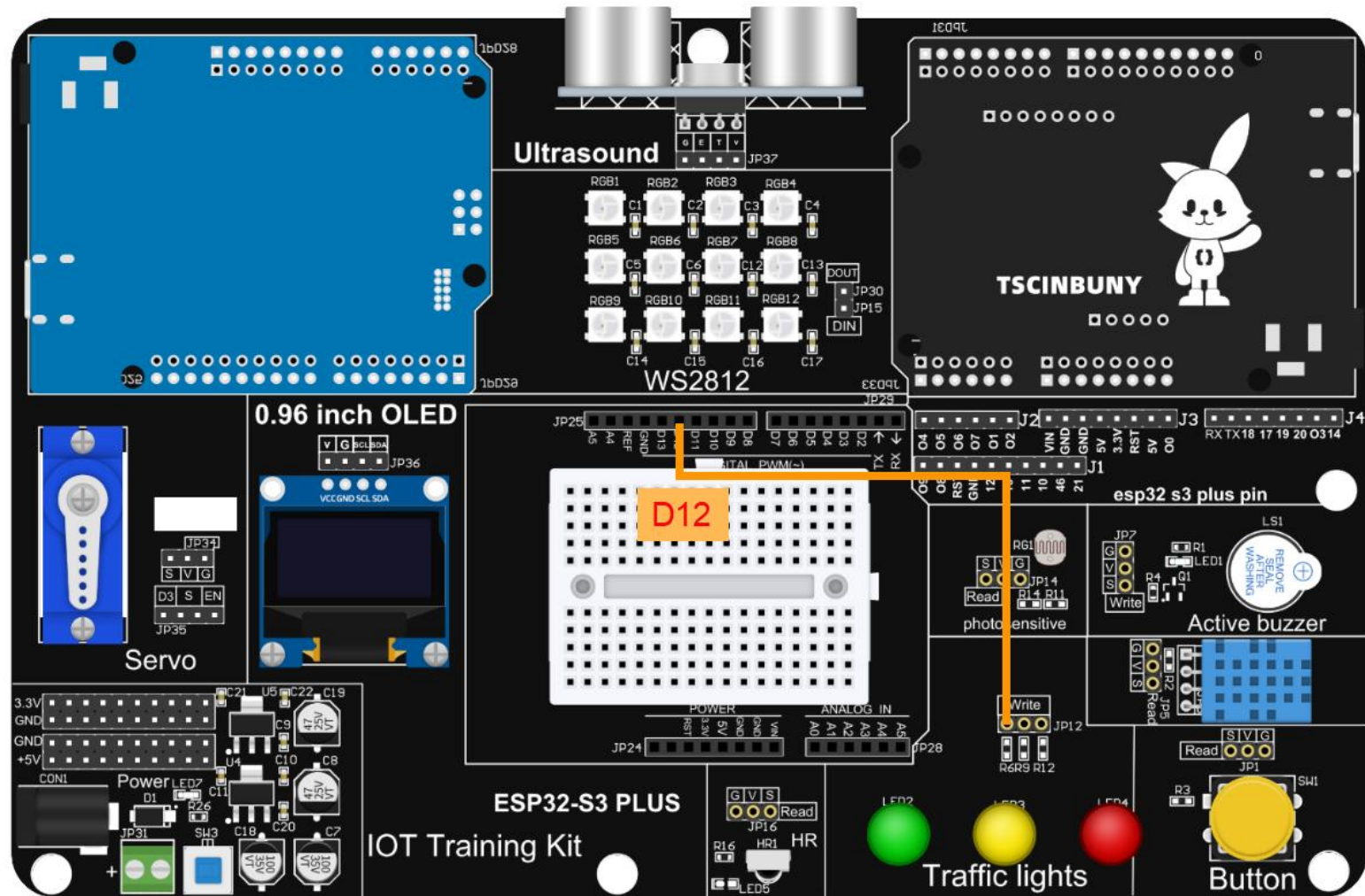
2.2.2.Expansion board



Board overview:

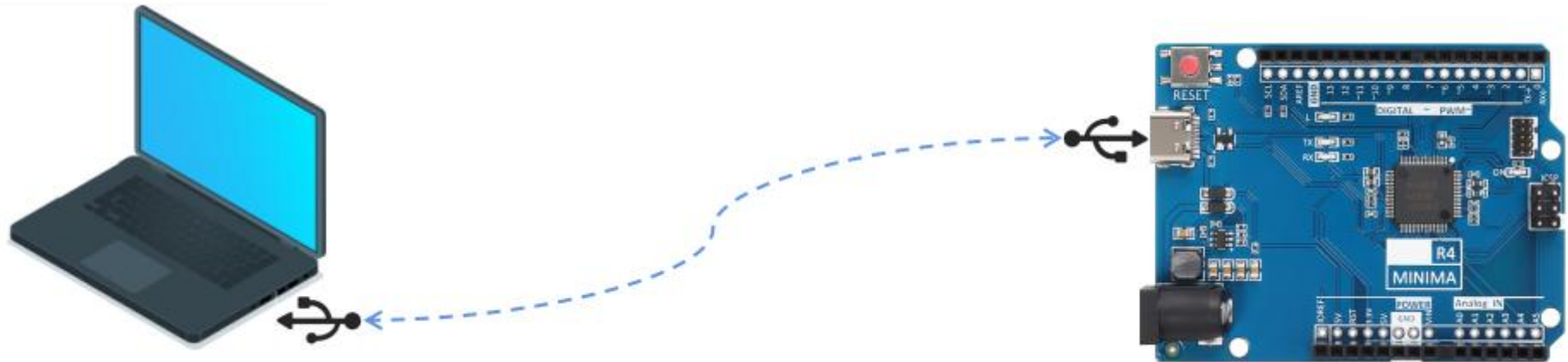
1. The expansion board design supports two main control boards, the UNO R4 Minima on the left and the ESP32 on the right, which should be installed correctly during use;
2. The onboard resources also include ultrasonic sensor, ws2812, servo, OLED, photoresistor, buzzer, DHT11, infrared receiver, LED 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° and 180° 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 has been connected 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

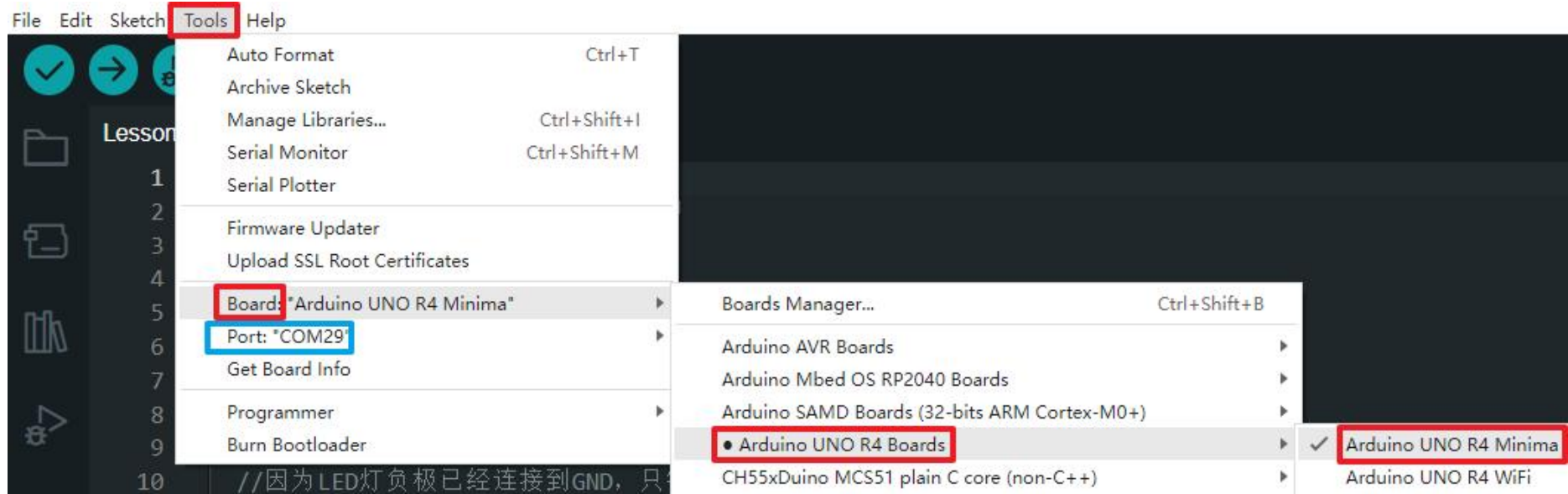
2.4.1. Connect the main control board to the computer with a USB cable



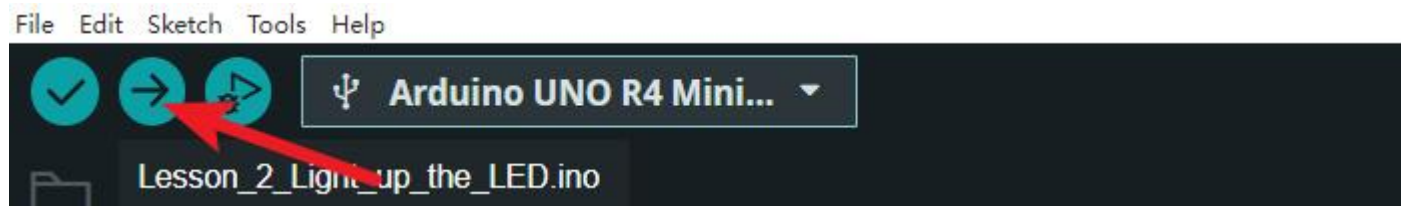
2.4.2. Open the "1_UNO_R4_MINIMA\Lesson_2_Light_up_the_LED" code file

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

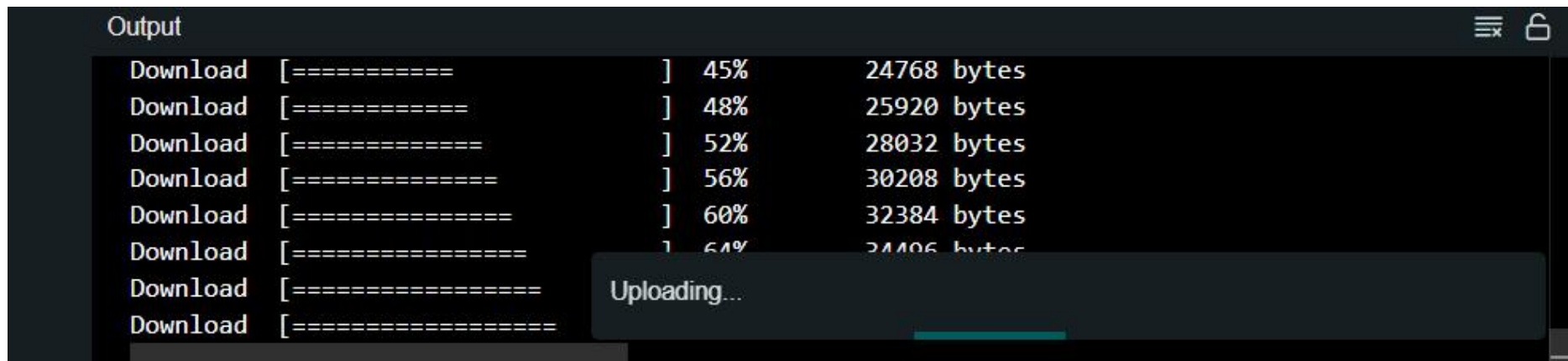
Select the board type as UNO R4 Minima. When plugging in the USB, a new COM number will be displayed. Select it. Here it is COM29, but the actual COM number will be different for everyone.



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



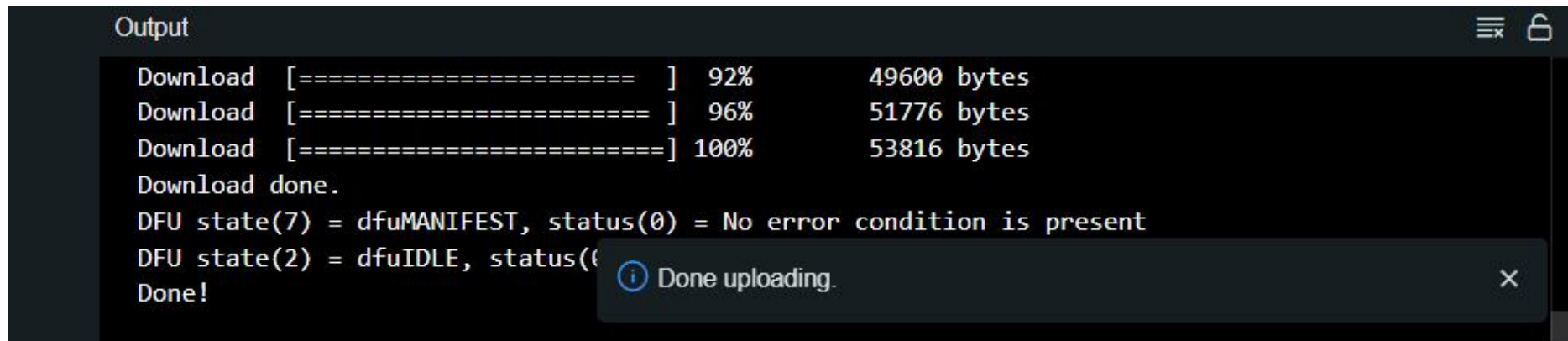
Waiting for the program to upload;



```

Output
Download [===== ] 45%      24768 bytes
Download [===== ] 48%      25920 bytes
Download [===== ] 52%      28032 bytes
Download [===== ] 56%      30208 bytes
Download [===== ] 60%      32384 bytes
Download [===== ] 64%      34496 bytes
Download [===== ]
Download [===== ]
  
```

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



```

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!
  
```

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 12  //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.

```
17 void loop() {  
18     //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  
19     digitalWrite(LED, HIGH);  
20 }
```

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

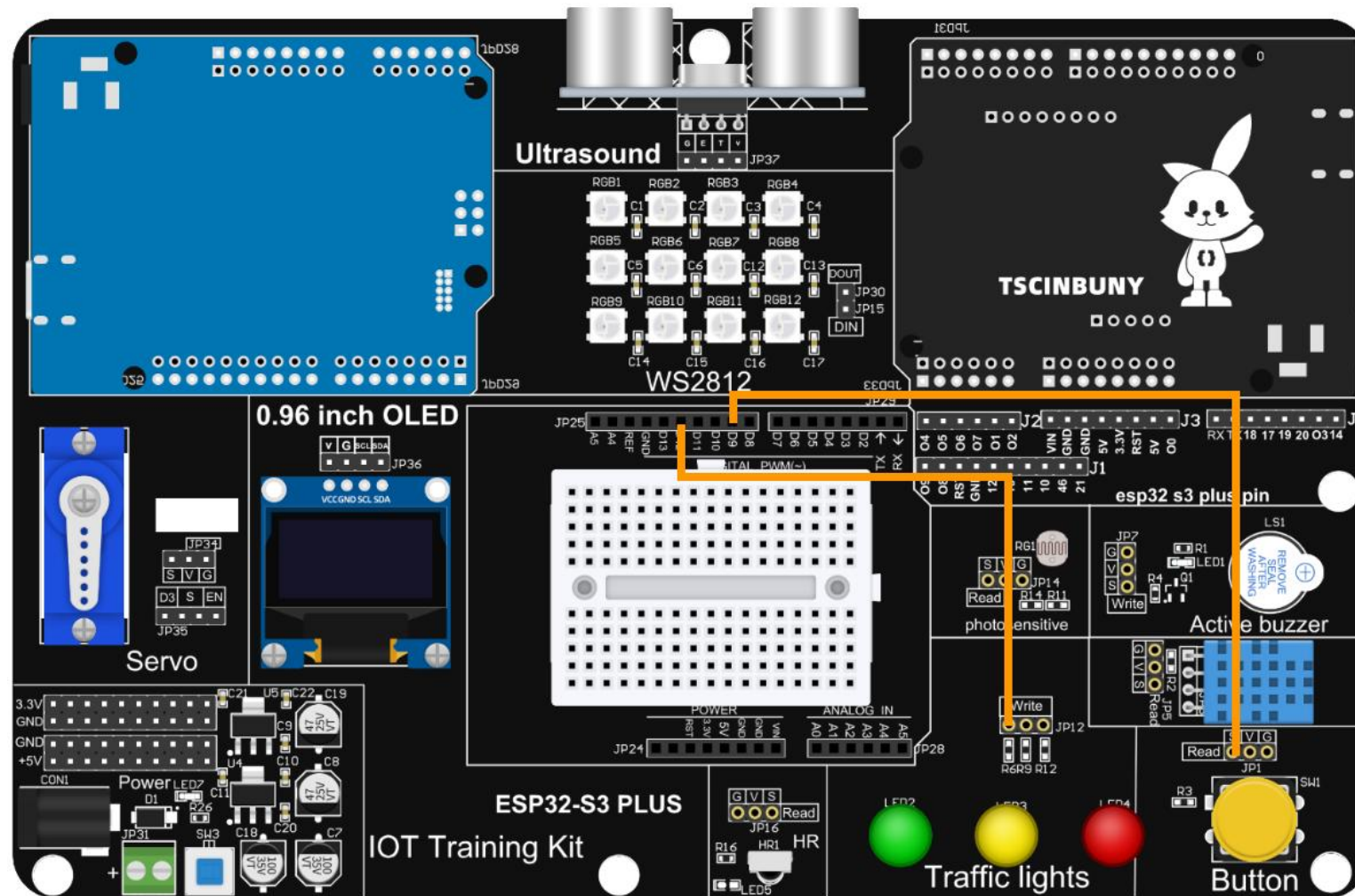
The key signal pin S has a pull-up resistor by default, so it is high level and changes to low level when the key is pressed. At this time, the D9 input of the main control board is low level, and the LED pin level is flipped by judging the pin status as a condition, thereby controlling the LED to turn on and off.

3.2.1. LED



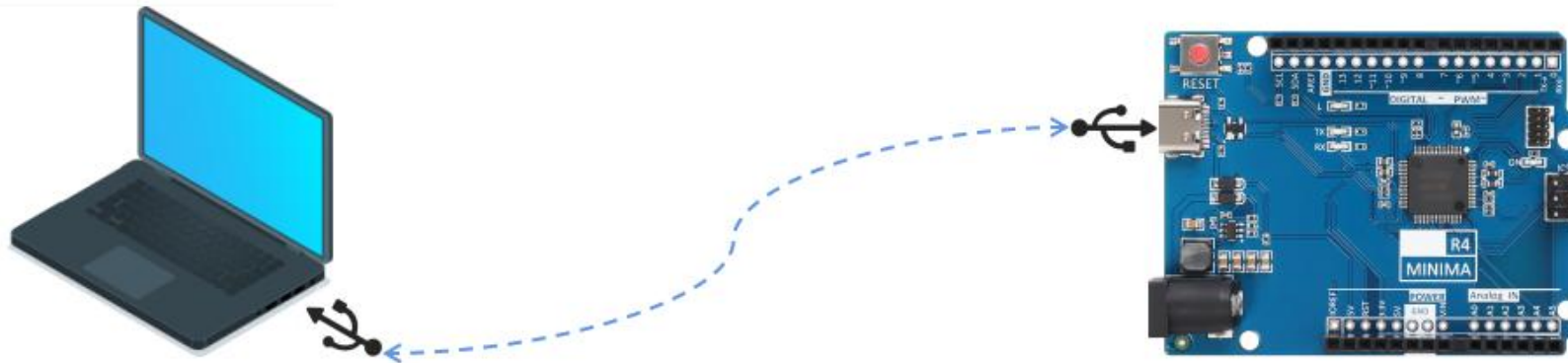
LED (Light Emitting Diode), which converts electrical energy into light energy, also has one-way conductivity and a reverse breakdown voltage of about 5v. Its forward volt-ampere characteristic curve is very steep. In the development board, the negative electrodes of the LEDs are all common to the ground.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

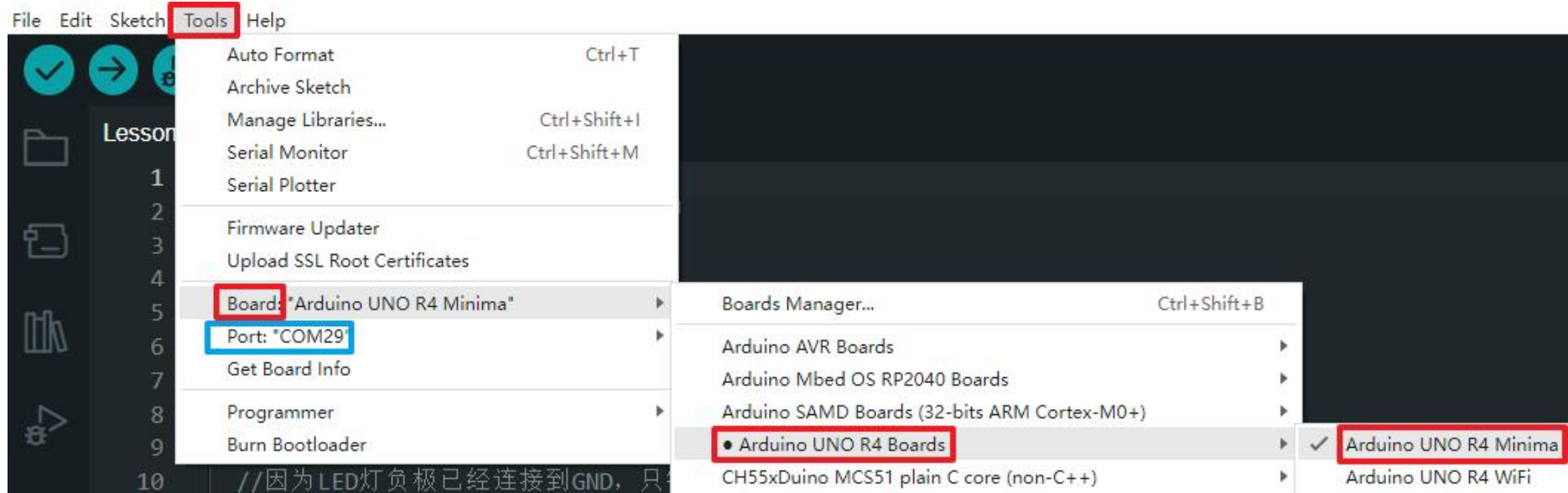


3.4.2. Open the program file (path: 1_UNO_R4_MINIMA \ Lesson_3_Button_control_LED)

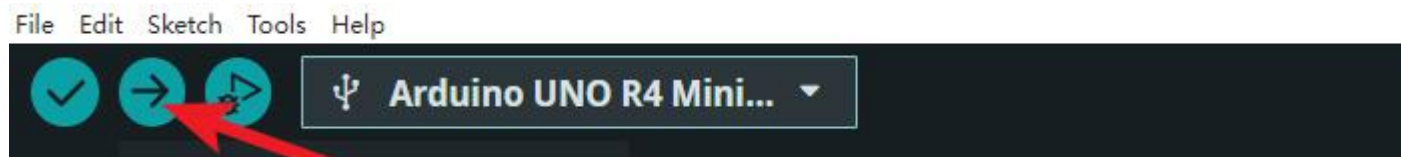
Lesson_1_Arduino IDE	2024/3/13 14:32	文件夹
Lesson_2_Light_up_the_LED	2024/3/13 13:43	文件夹
Lesson_3_Button_control_LED	2024/3/5 12:00	文件夹
Lesson_4_active_buzzer	2024/3/13 13:43	文件夹
Lesson_5_Traffic_light	2024/3/13 13:43	文件夹
Lesson_6_Flow_light	2024/3/13 13:43	文件夹
Lesson_7_WS2812B	2024/3/6 11:10	文件夹
Lesson_8_Gradient_RGB_light	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 UNO R4 Minima, and select the COM number newly displayed when plugging in the USB. 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 12    //定义LED2引脚 Define the LED2 pin
3  #define button 9  //定义按键信号引脚 Define the key signal pin
4  int key_ok = 0;   //定义项目所需的数据变量 De
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.

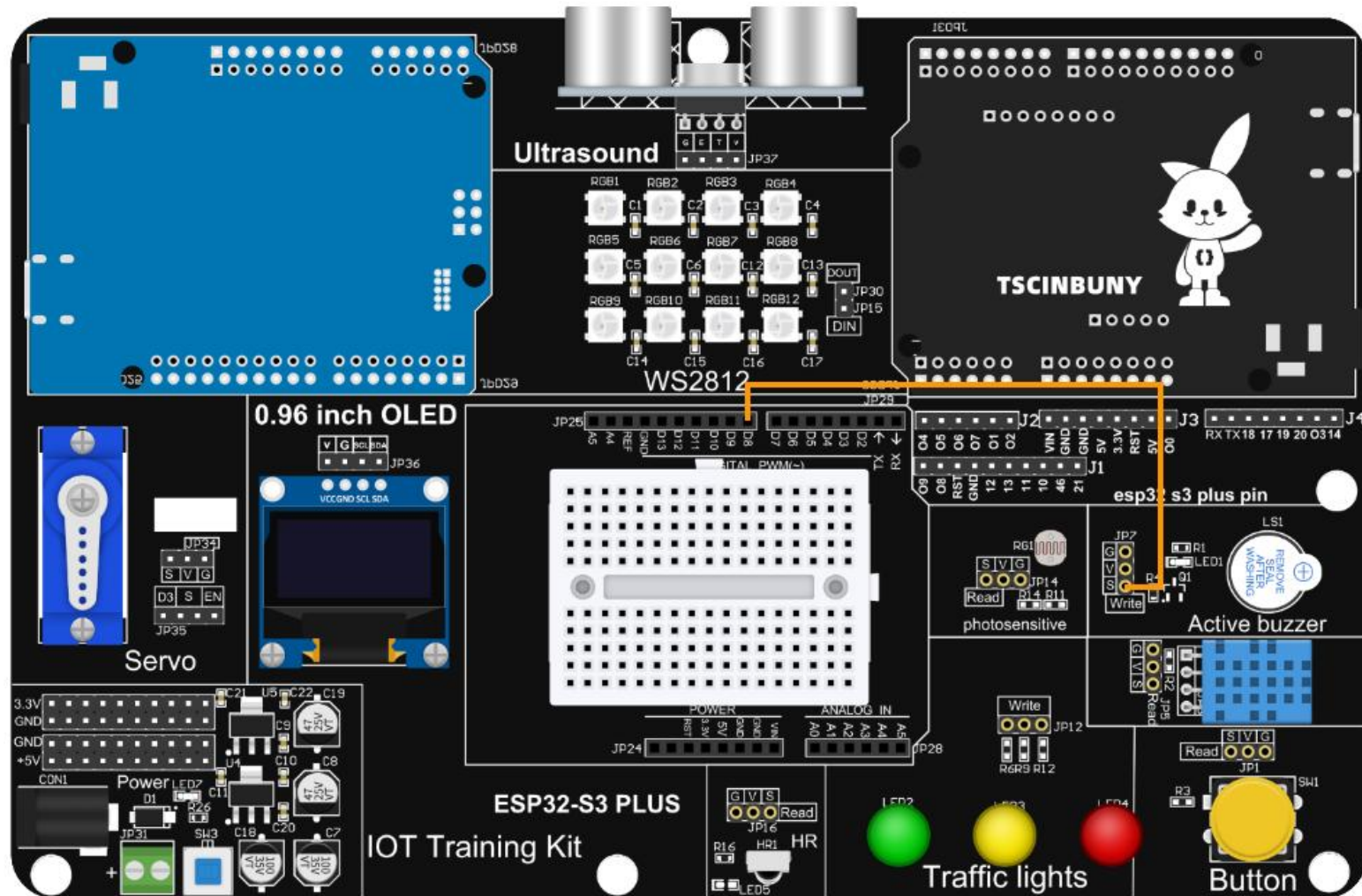
4.2. Working principle

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.

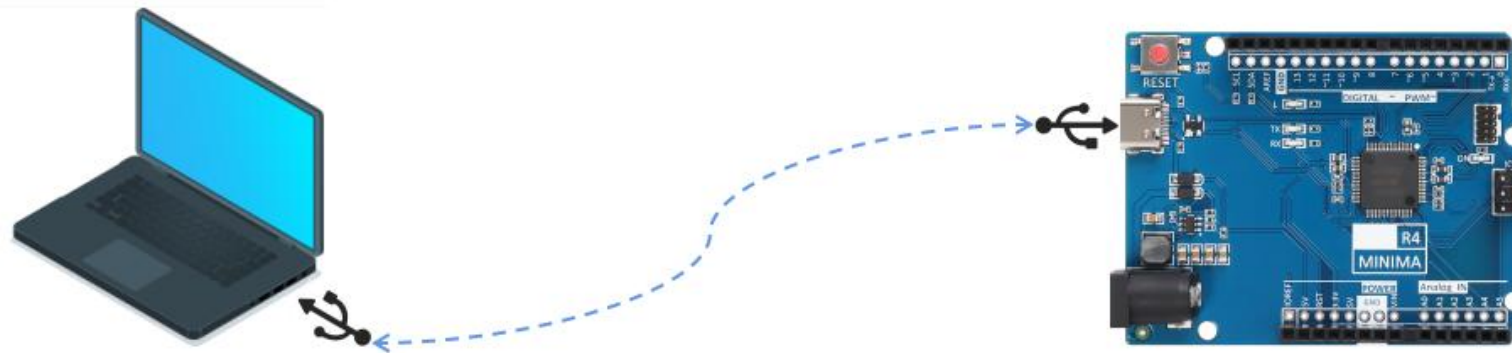


4.3.Connect lines



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: 1_UNO_R4_MINIMA \ 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
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_gear_control	2024/3/5 14:15	文件夹

Also select the board type as UNO R4 Minima, 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.

```

4.5. Code analysis

Define the buzzer pin and set the buzzer as an output

```

2  #define Buzzer 8           //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 o
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

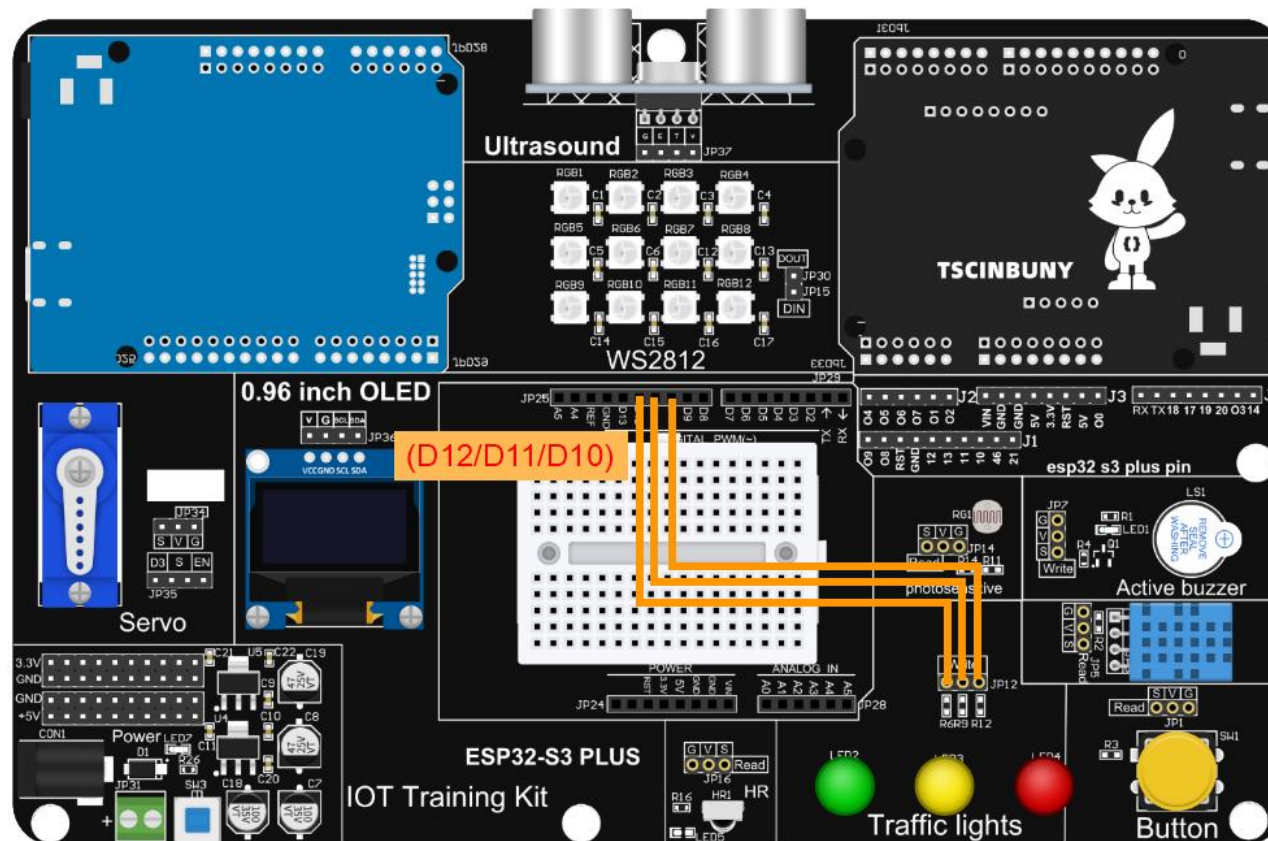


There are three colors of traffic lights, green light, yellow light and red light. The pins connected to the main control board are 12/11/10. By controlling the high level of the three pins, the corresponding lights can be lit. A single light cycle is as follows: first let the green light be on for a 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:

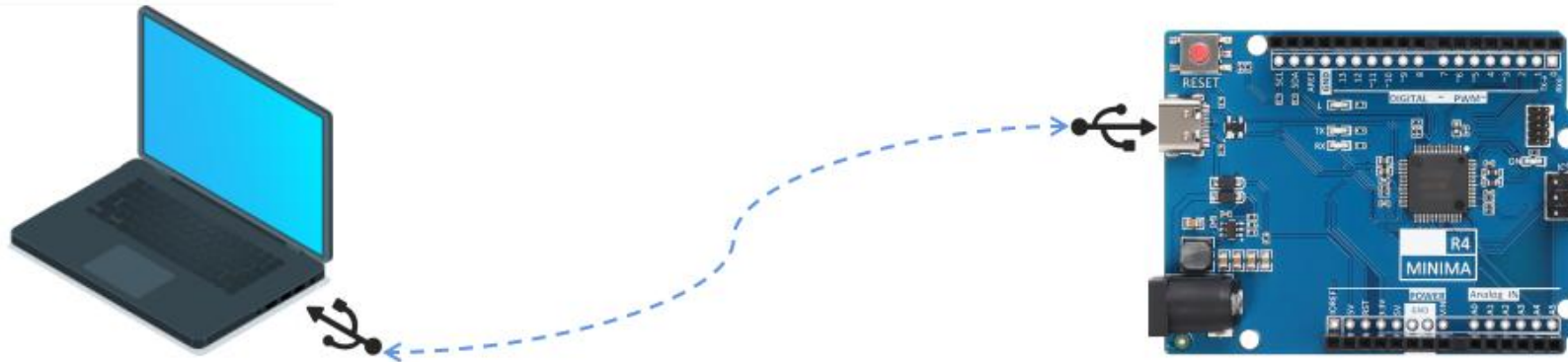
Leave the green light on for five seconds , the green light flashing every 500 milliseconds, the yellow light on for 1 second, the red light on for 5 seconds, the red light flashing every 500 milliseconds, and so on.

5.3 Connection lines



5.4 Upload code

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

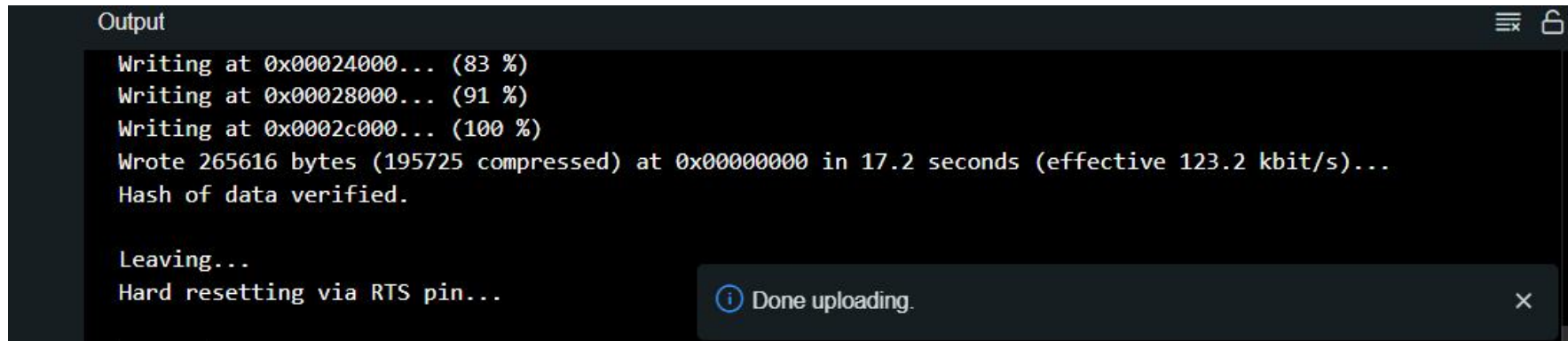


5.4.2 Open the program file (path: 1_UNO_R4_MINIMA\Lesson_5_Traffic_light)

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
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 UNO R4 Minima, 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 12  //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 the buzzer 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

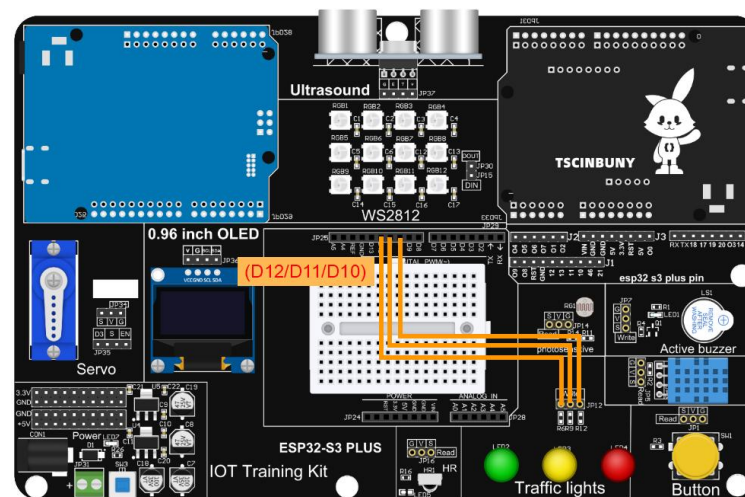
6.1. Overview

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

6.2. Working principle

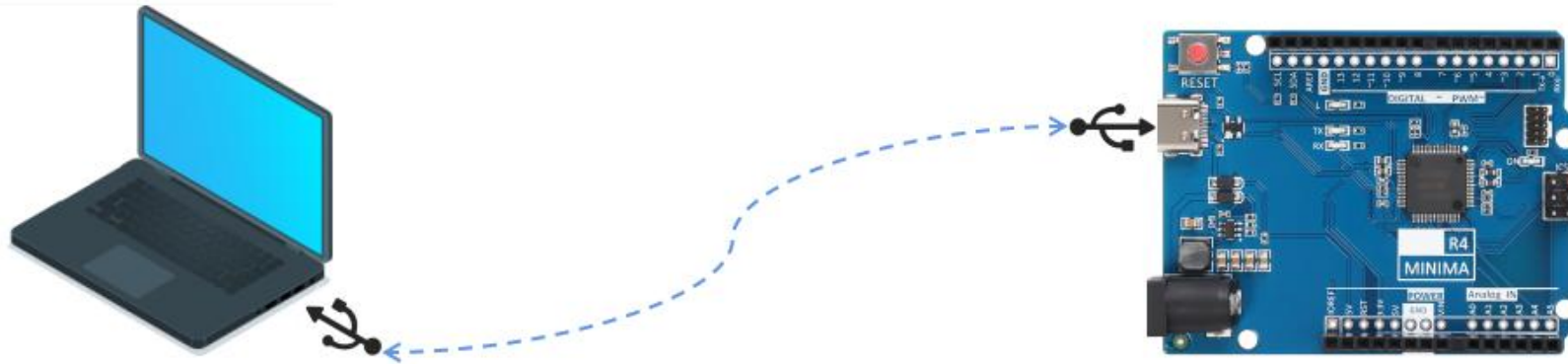
As in the previous section "Traffic Light Project", the LED on/off is controlled by controlling the high/low level of the three LED light pins. After setting the waiting time for on and off, you can see the lighting process of the three LEDs one by one, simulating the effect of running water lamps.

6.3 Connection lines



6.4 Upload code

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

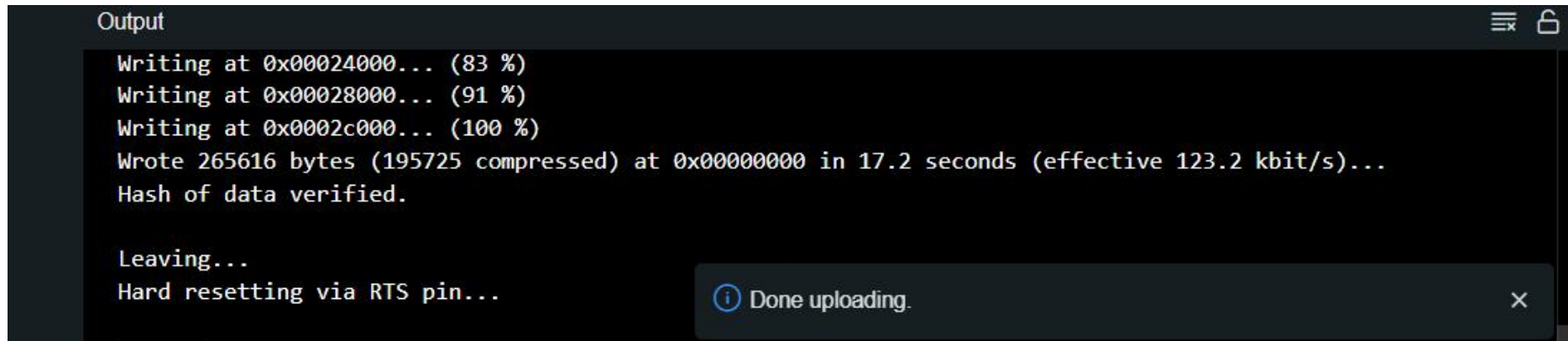


6.4.2 Open the program file (path: 1_UNO_R4_MINIMA\ Lesson_6_Flow_light)

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
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 UNO R4 Minima, 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 an IDE's output window with a dark background. The title bar says "Output" and has icons for a list and a lock. The text in the window shows the progress of uploading a program to a microcontroller. It includes status updates like "Writing at 0x00024000... (83 %)", "Writing at 0x00028000... (91 %)", and "Writing at 0x0002c000... (100 %)". It also reports "Wrote 265616 bytes (195725 compressed) at 0x00000000 in 17.2 seconds (effective 123.2 kbit/s)..." and "Hash of data verified." Below this, it says "Leaving..." and "Hard resetting via RTS pin...". At the bottom right, there is a grey notification box with a blue information icon and the text "Done uploading.", with a close button (X) on the right.

```
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.
```

6.5 Code analysis

Define three LED pins and operating modes

```
3  #define Green 12  //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  void setup() {
7      //定义引脚为输出工作模式
8      pinMode(Green, OUTPUT); //绿色灯
9      pinMode(Yellow, OUTPUT); //黄色灯
10     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

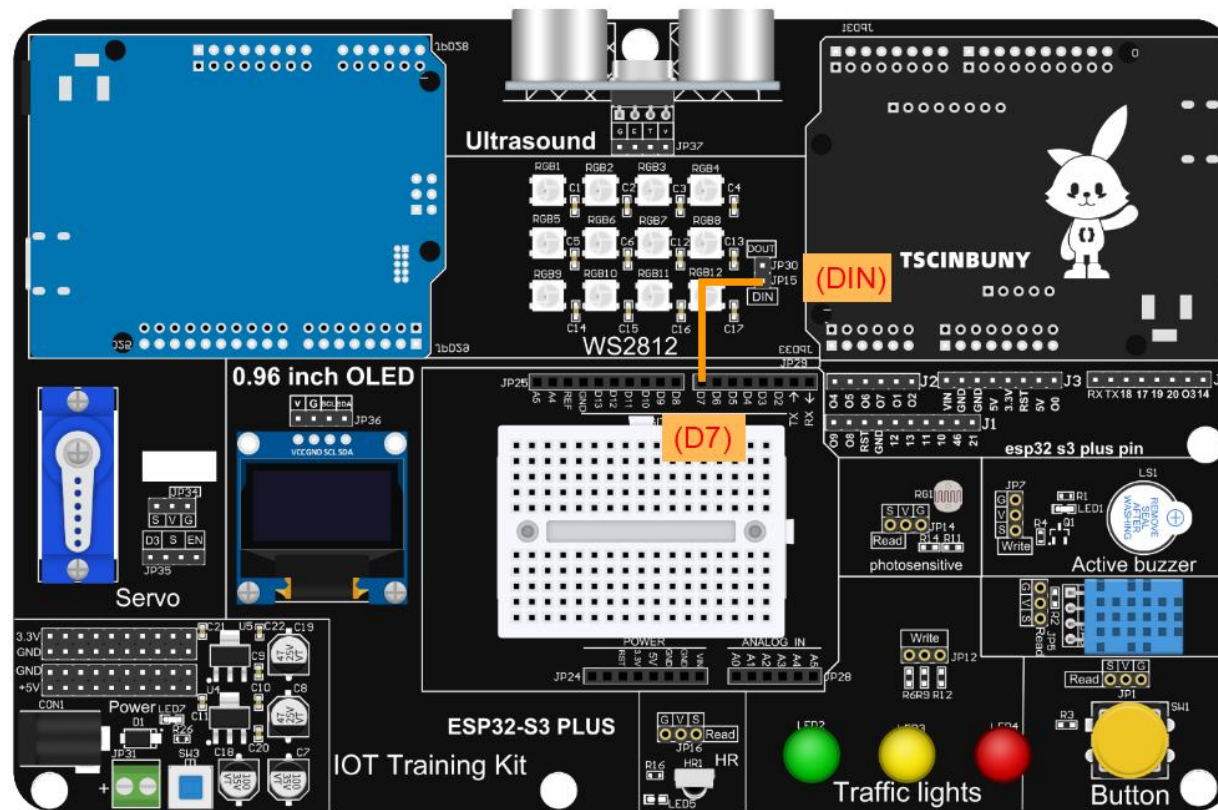


WS2812B is an intelligent externally controlled LED light source that integrates control circuit and light-emitting circuit. Its appearance is the same as a 5050LED lamp bead, and each component is a pixel. There is an intelligent digital interface data latch signal shaping amplification drive circuit, 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

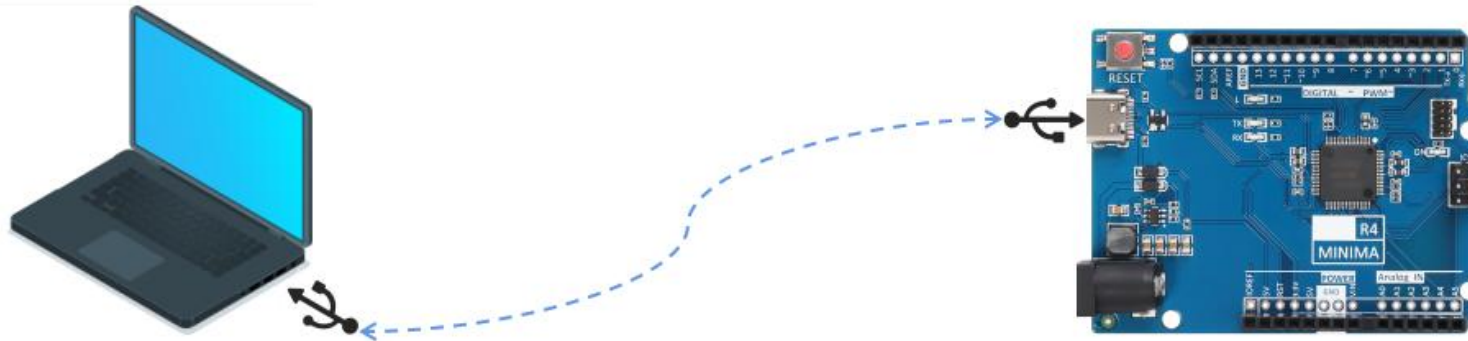
forwarded to the next cascade pixel through the DO output port. After each pixel is transmitted, the signal is reduced by 24 bits. The pixels adopt automatic shaping and forwarding technology, so that the number of cascades of pixels is not limited by signal transmission, but only limited by the signal 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: 1_UNO_R4_MINIMA\ 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
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 UNO R4 Minima, 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 7, 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 7        //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

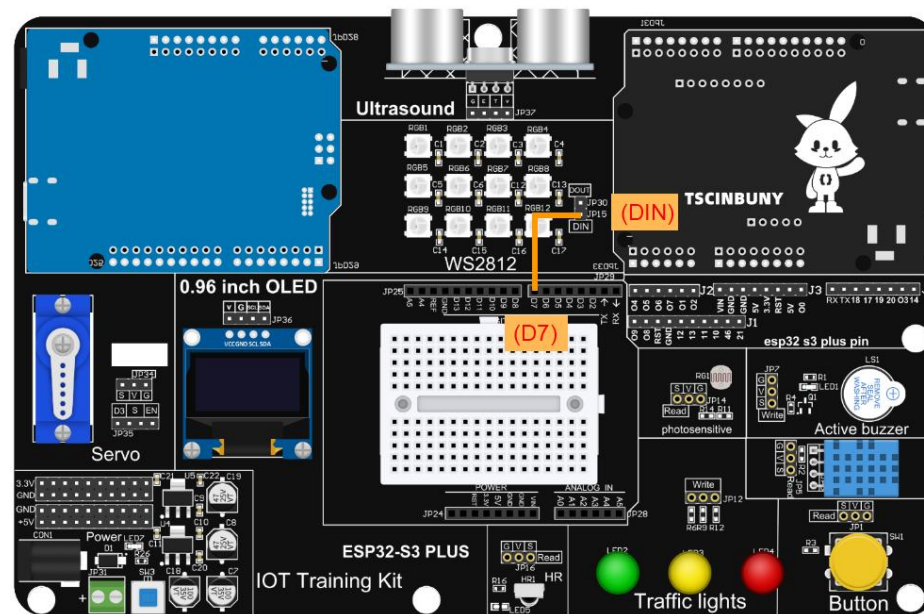
8.1. Overview

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

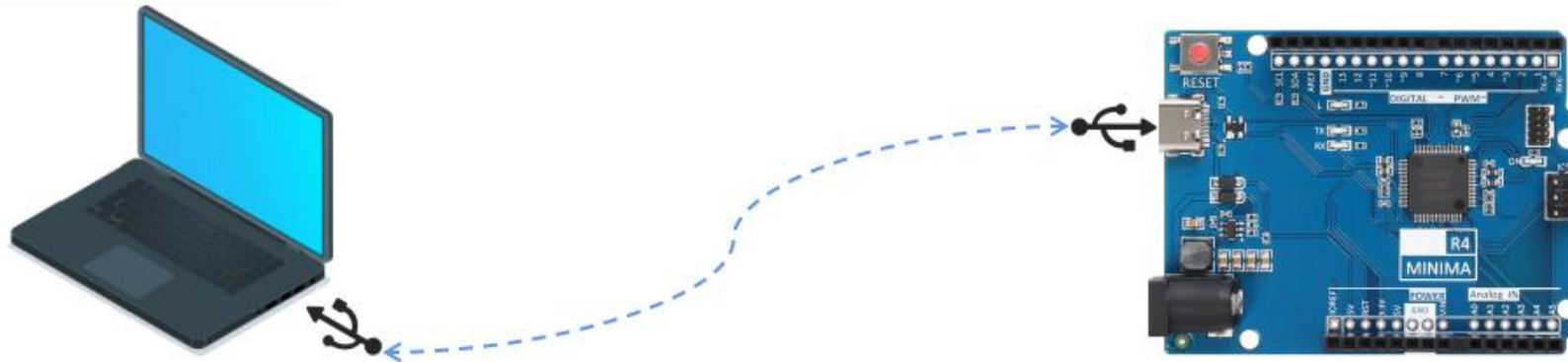
Use the library function `fill_rainbow(leds, NUM_LEDS, hue, 8);` to change the gradient display of RGB lights.

8.3 Connection lines



8.4. Upload code program

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

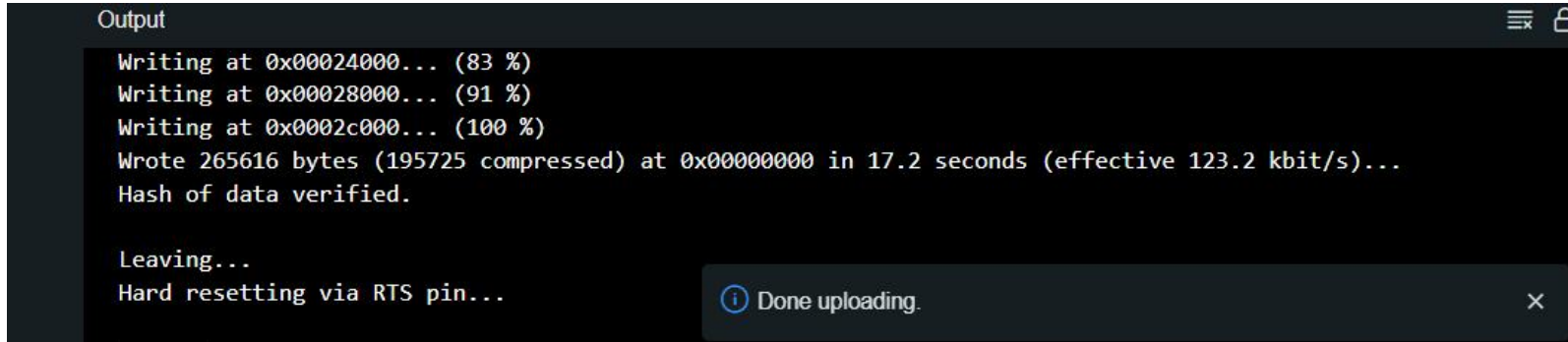


8.4.2 Open the program file (path: 1_UNO_R4_MINIMA\Lesson_8_Gradient_RGB_light)

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
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 UNO R4 Minima, 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. The `fill_rainbow` function is used 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

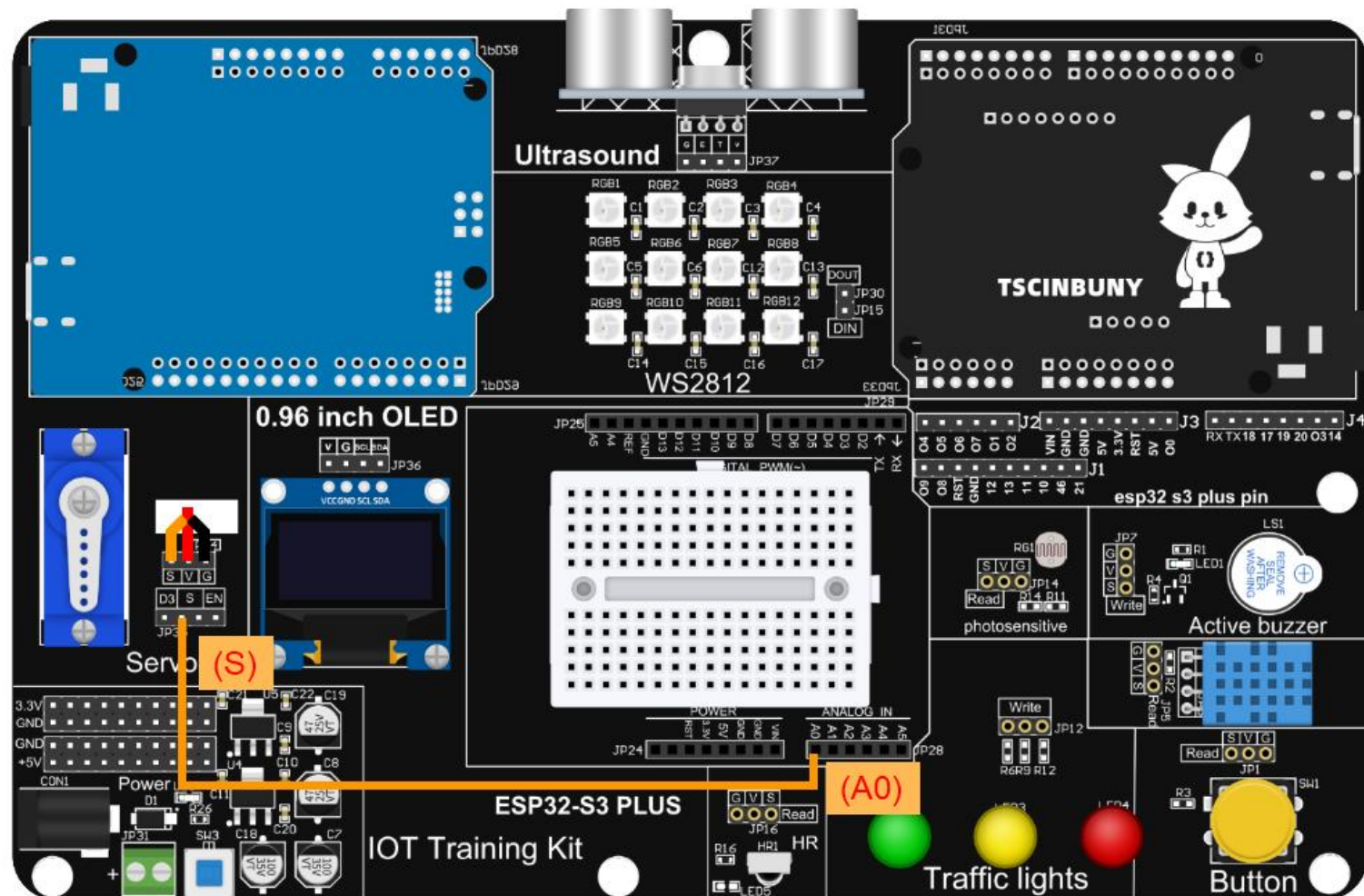
9.2.1. Steering gear



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.

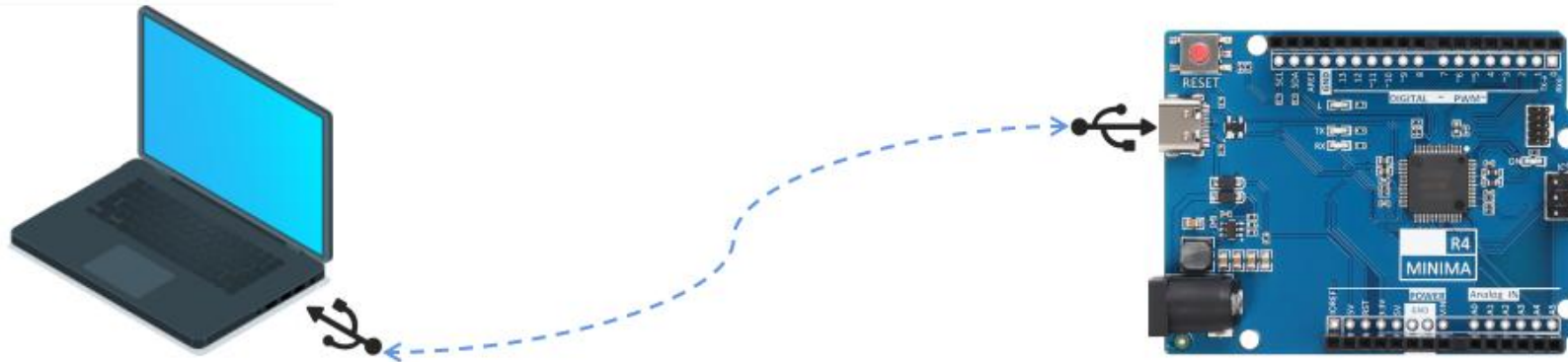
There is a reference circuit inside the steering gear, which generates a pulse signal with a period of 20ms and a width of 1.5ms. There is a comparator that compares the external signal with the reference signal to determine the direction and size, thereby generating a motor rotation signal.

9.3 Connection lines



9.4 Upload code program

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

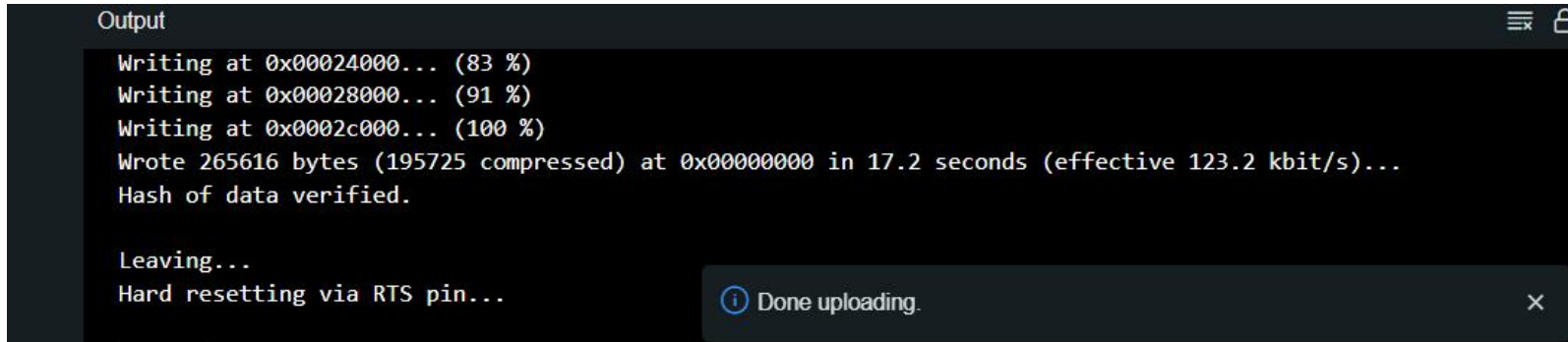


9.4.2 Open the program file (path: 1_UNO_R4_MINIMA\Lesson_9_Steering_gear_control)

Lesson_7_WS2812B	2024/3/6 11:10	文件夹
Lesson_8_Gradient_RGB_light	2024/3/6 11:09	文件夹
Lesson_9_Steering_gear_control	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 UNO R4 Minima, 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   }
```

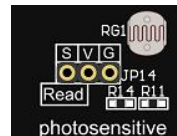
10. Photoresistor

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

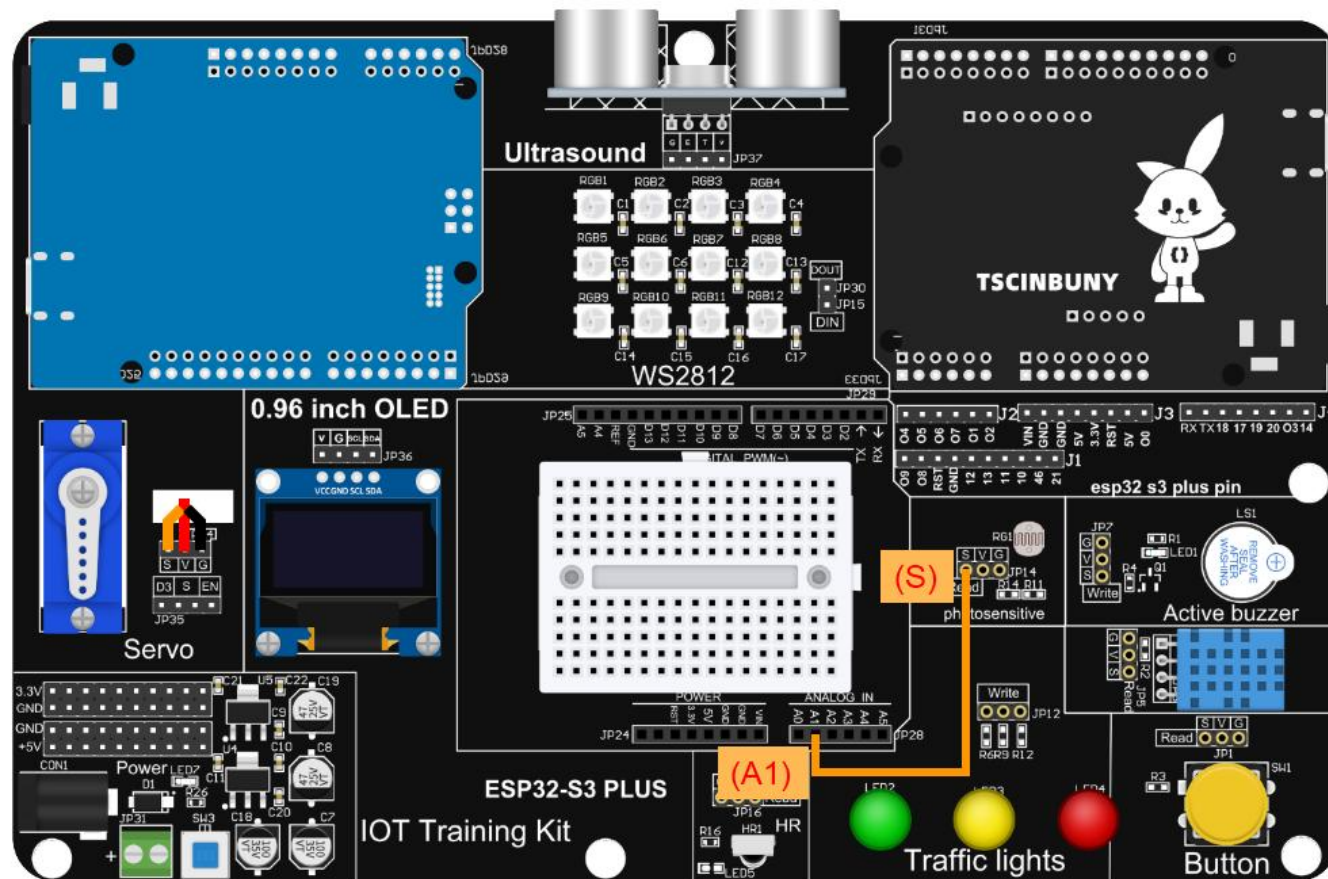


A photoresistor is a resistor made by utilizing the photoelectric effect of semiconductors. The resistance value changes with the intensity of incident light. It is also called a photodetector. When the incident light is strong, the resistance decreases. When the incident light is weak, the resistance 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

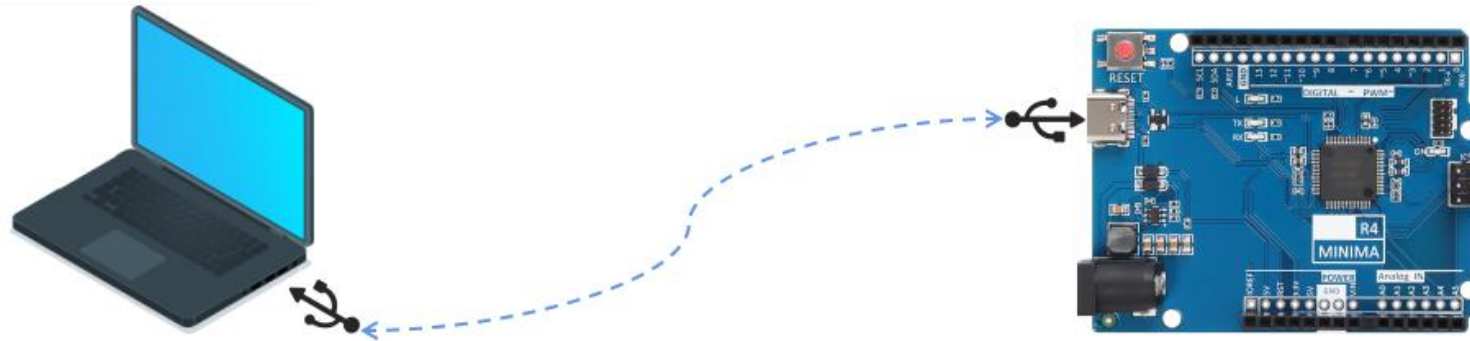
electrode of the power supply, thus causing the resistance of the photoresistor to drop rapidly. Photoresistors are generally used for light measurement, light control and photoelectric conversion.

10.3 Connection lines



10.4 Upload code program

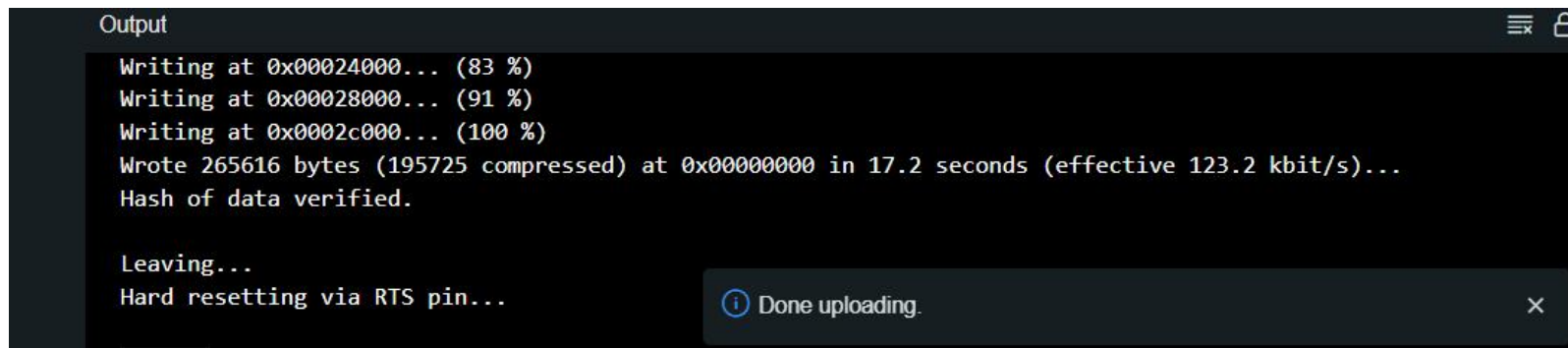
1 0 .4.1 Connect the main control board to the computer using a USB cable



1 0 .4.2 Open the program file (path: 1_UNO_R4_MINIMA\ Lesson_10_Photorresistor)

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
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 UNO R4 Minima, 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, open the IDE serial port monitor and you can see the printed output light intensity value. Generally, the indoor light detection value is 740



Lesson_10_Photoresistor.ino

Serial Monitor

Message (Enter to send message to 'Arduino UN... New Line 9600 baud

```
Lightvalue = 742
Lightvalue = 743
Lightvalue = 743
Lightvalue = 739
Lightvalue = 742
Lightvalue = 738
```

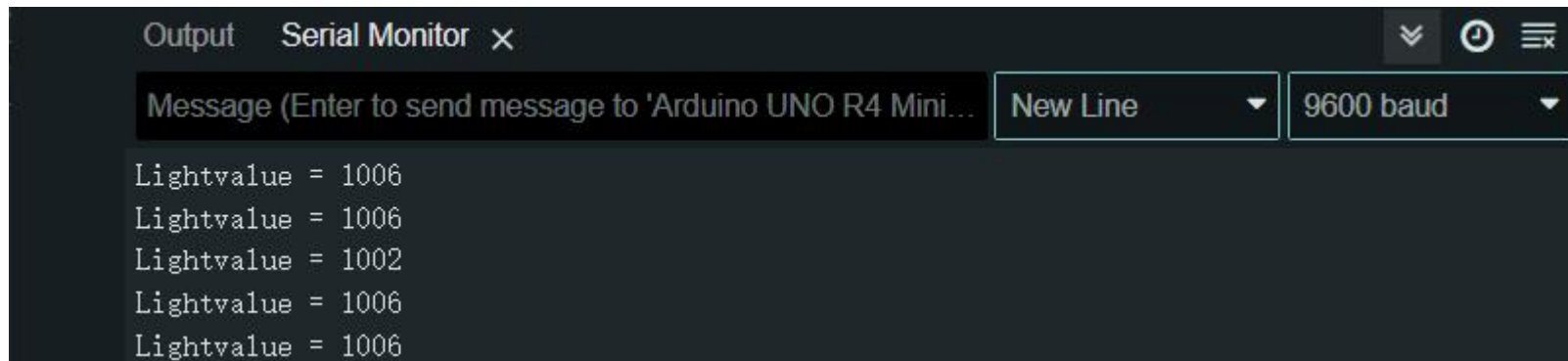
The detection value is <50 when covering the photoresistor with objects.



The screenshot shows the Arduino IDE Serial Monitor window. The title bar reads "Output Serial Monitor" with a close button. The input field contains the text "Message (Enter to send message to 'Arduino UNO R4 Mini...'", and the dropdown menu is set to "New Line". The baud rate is set to "9600 baud". The output area displays the following text:

```
Lightvalue = 33  
Lightvalue = 30  
Lightvalue = 33  
Lightvalue = 34  
Lightvalue = 35  
Lightvalue = 30
```

When the photoresistor is illuminated with a flashlight, the detection value is >1000 .



The screenshot shows the Arduino IDE Serial Monitor window. The title bar reads "Output Serial Monitor" with a close button. The input field contains the text "Message (Enter to send message to 'Arduino UNO R4 Mini...'", and the dropdown menu is set to "New Line". The baud rate is set to "9600 baud". The output area displays the following text:

```
Lightvalue = 1006  
Lightvalue = 1006  
Lightvalue = 1002  
Lightvalue = 1006  
Lightvalue = 1006
```

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 = A1;  //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 sen
13     Serial.println(Lightvalue);
14     delay(200);
15 }
```

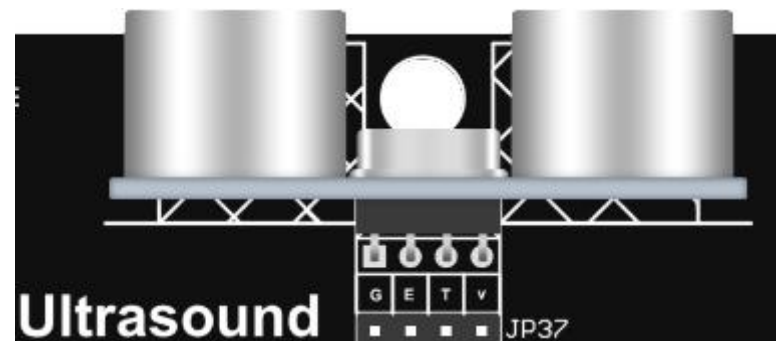
11. Ultrasonic ranging

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.

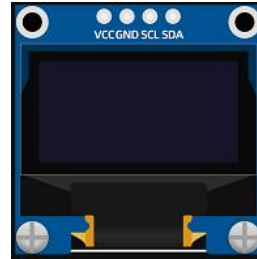
There are many methods of ultrasonic ranging. The principle of this system in ultrasonic measurement is: the trigger signal input terminal (TRIG) will input a high-level signal of more than 10 microseconds. After receiving the signal, the ultrasonic transmitter will automatically send 8 A 40Hz square 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 .

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

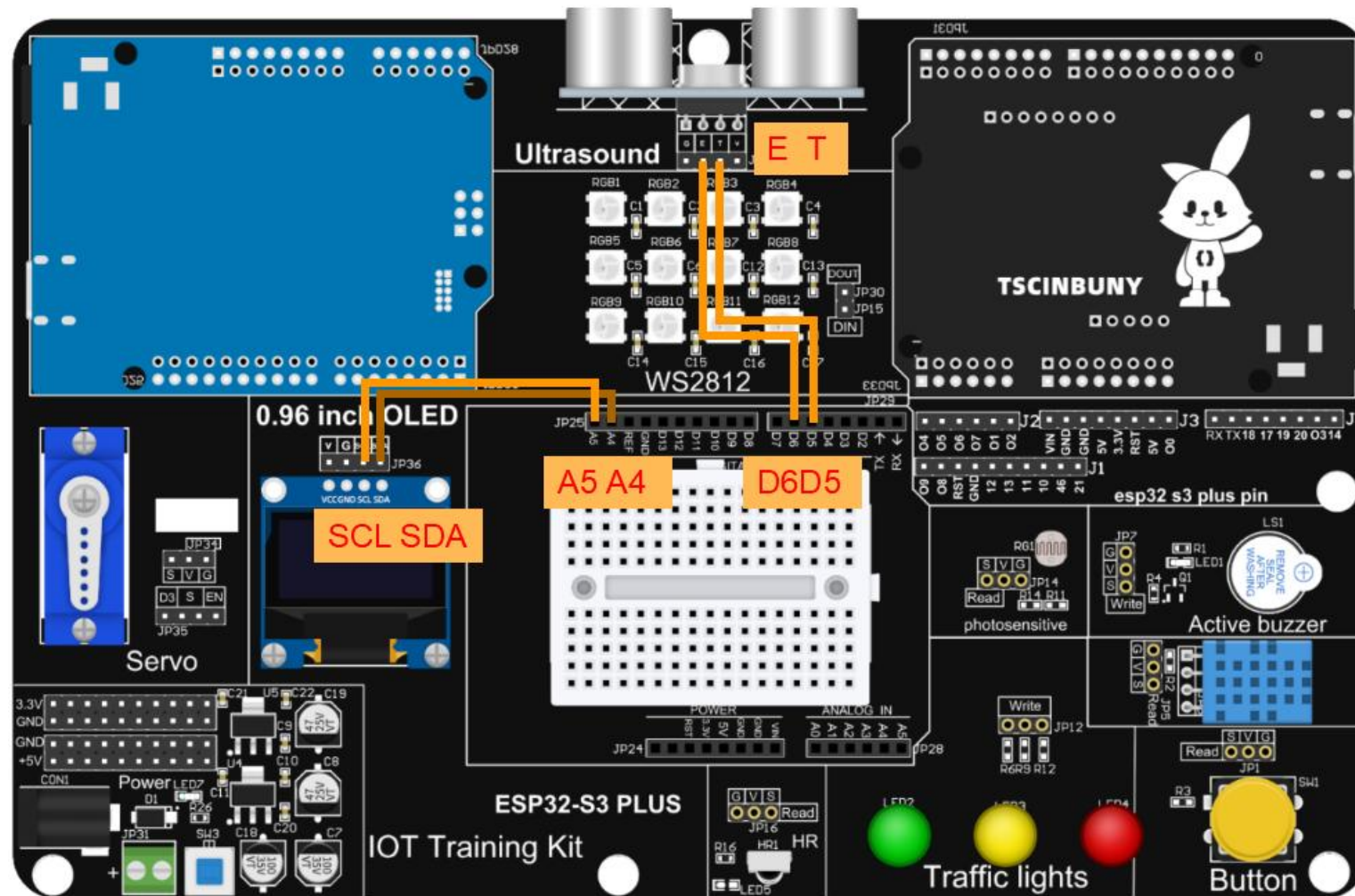
Four factors limit the maximum measurable distance of an ultrasound system: the amplitude of the ultrasound wave, the texture of the reflector, the angle between the reflected and incident sound waves, and the sensitivity of the receiving transducer. The ability of the receiving transducer to directly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

11.2.2.OLED



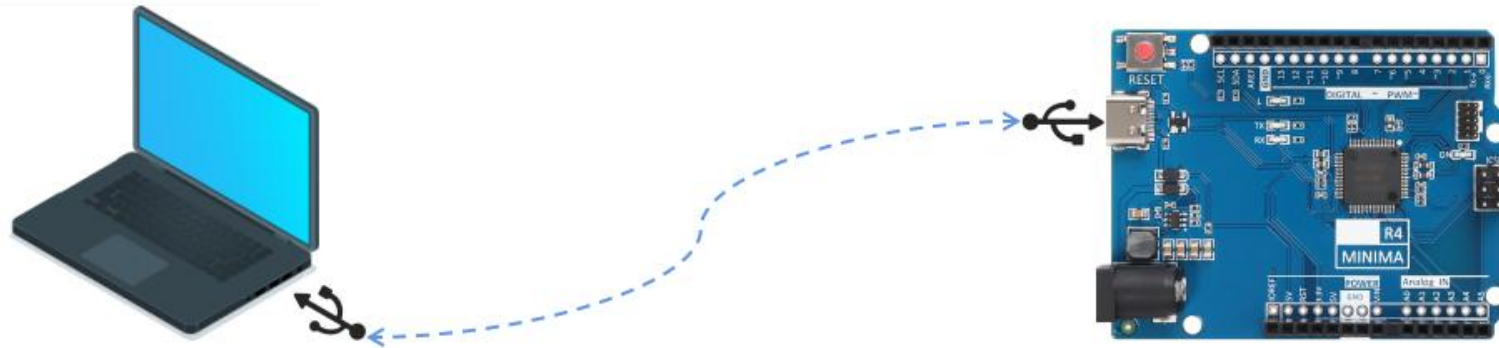
1. Resolution: 128*64
2. Super wide viewing angle: greater than 160
3. Communication method: IIC
4. Working voltage: 3.3V~5V

11.3 Connection lines



11.4 Upload code program

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



1 1 .4.2 Open the program file (path: 1_UNO_R4_MINIMA\ 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_Photoresistor	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 UNO R4 Minima, 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 [Section 3](#) to see how to add 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 6          //定义超声波Echo引脚 Define the ultrasonic Echo pin
13 #define Trig 5          //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°C

Interchangeability: Interchangeable

Response time: 6S under 1/e(63%) 25°C, 1m/s air condition

Hysteresis: $<\pm 0.3\%$ RH

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

temperature:

Resolution: $\pm 2^\circ\text{C}$

Repeatability: $\pm 0.2^\circ\text{C}$

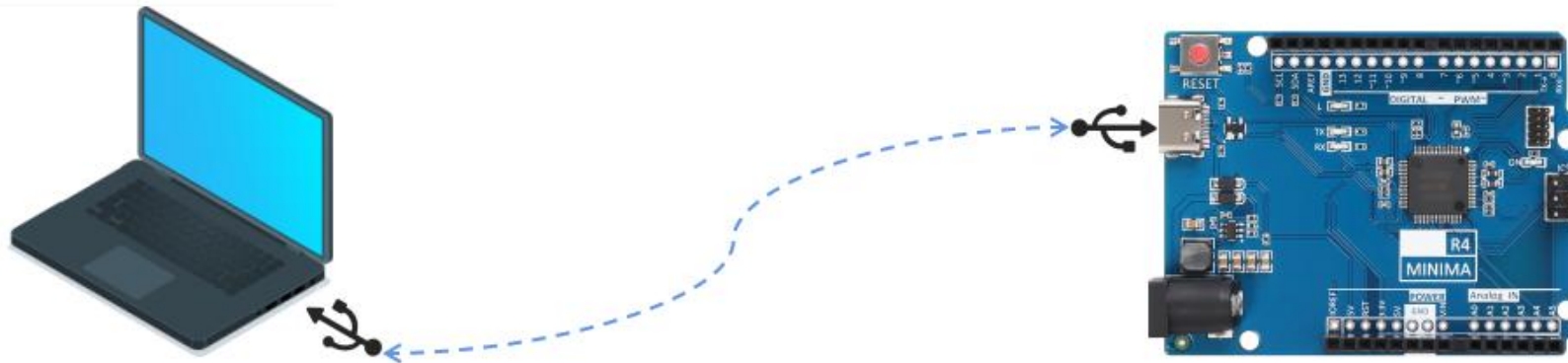
Interchangeability: Interchangeable

Hysteresis: $<\pm 0.3\%$ RH

Response time: 10S under 1/e (63%) conditions

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: 1_UNO_R4_MINIMA\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 UNO R4 Minima, 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

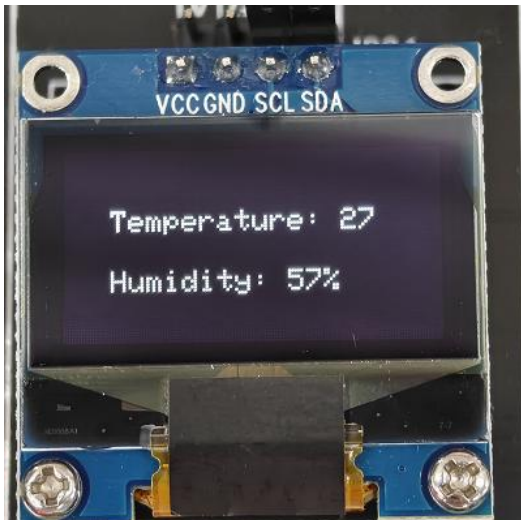
upload to be completed .

```
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 [Section 3](#) 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 4          // DHT11传感器连接到Arduino的数字引脚4 The DHT
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
```

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.

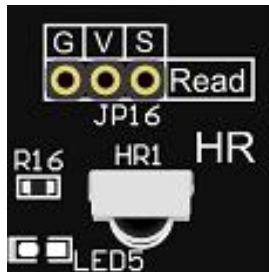
13.2 Working principle

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.

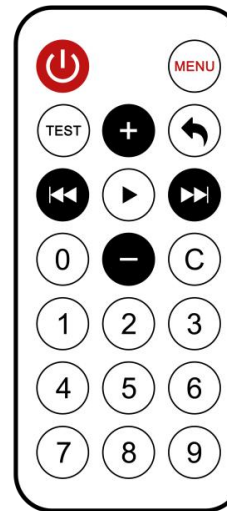
The encoding scheme that matches the infrared remote control protocol is: NEC protocol. Next, let us understand what the NEC protocol is.

- (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

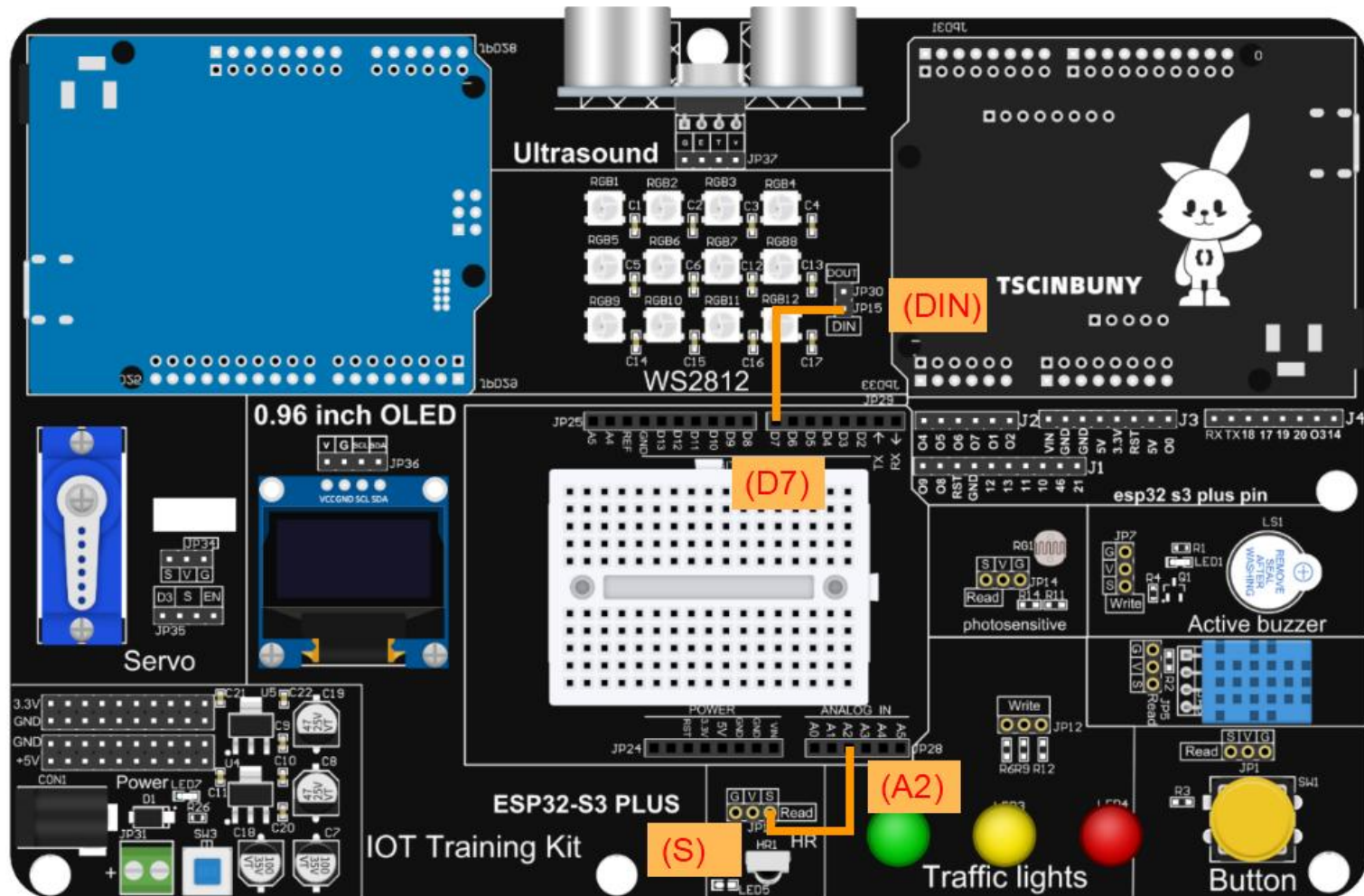
infrared receiver



infrared transmitter

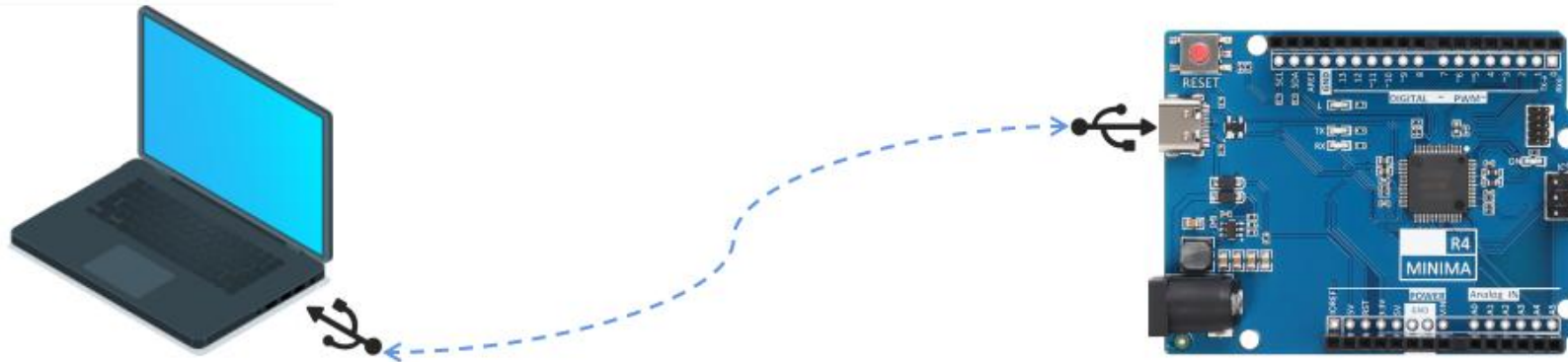


13.3 Connection lines



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: 1_UNO_R4_MINIMA\ Lesson _ 13_Infrared_change_RGB)

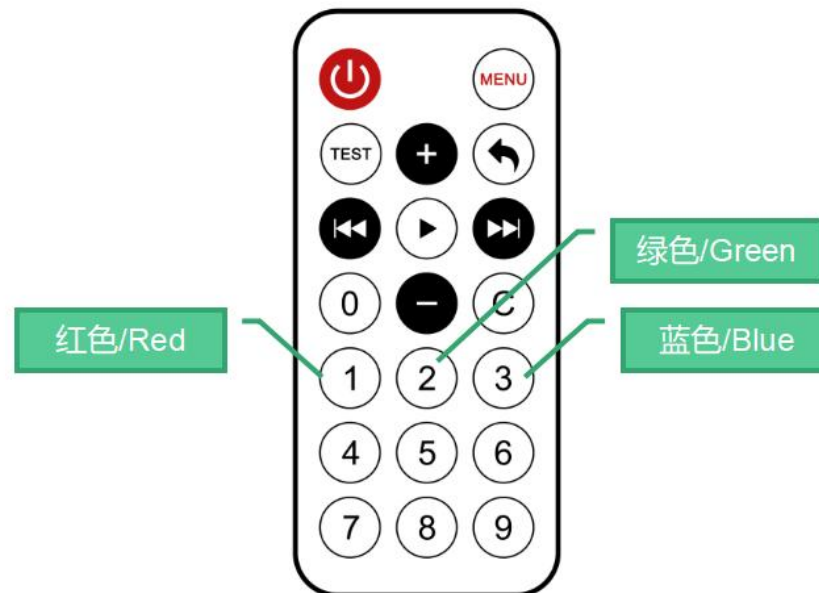
Lesson_7_WS2812B	2024/3/6 11:10	文件夹
Lesson_8_Gradient_RGB_light	2024/3/6 11:09	文件夹
Lesson_9_Steering_gear_control	2024/3/5 14:15	文件夹
Lesson_10_Photorresistor	2024/3/5 14:30	文件夹
Lesson_11_Ultrasonic_ranging_OLED_display	2024/3/21 18:04	文件夹
Lesson_12_DHT11_OLED_display	2024/3/21 18:05	文件夹
Lesson_13_Infrared_change_RGB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

Also select the board type as UNO R4 Minima, 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 [Section 3](#) to see how to add the library.

```
1  #include <IRremote.hpp>
2  #include <FastLED.h>
```

Define the coding variables and pins corresponding to the infrared receiver buttons.

```
4  //设定三个按键的红外信号值 Set the infrared signal value of the three keys
5  #define NUM1 0xF30CFF00 //number 1
6  #define NUM2 0xE718FF00 //number 2
7  #define NUM3 0xA15EFF00 //number 3
8  #define IR_RECEIVE_PIN A2 //定义红外接收器的引脚 Define the pins of the
```

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 WS2812
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
```

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


```

18 void setup() {
19     Serial.begin(9600);
20     IrReceiver.begin(IR_RECEIVE_PIN, ENABLE_LED_FEEDBACK); //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 on the
22     FastLED.setBrightness(BRIGHTNESS); //设置灯光亮度 Set the light brightness
23     pinMode(LED_PIN, OUTPUT); //定义7引脚为输出模式 Pin 7 is defined as the output mode
24     digitalWrite(LED_PIN, LOW); //设定7引脚为低电平输出 Set pin 7 for low output
25 }

```

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

```

27 void loop() {
28     if (IrReceiver.decode()) {
29         // Serial.println(IrReceiver.decodedIRData.decodedRawData, HEX); //打印“旧的”原始数据 P
30         // IrReceiver.printIRResultShort(&Serial); //在一行中打印完整的接收
31         // IrReceiver.printIRSendUsage(&Serial); //打印发送此数据所需的语
32         if (IrReceiver.decodedIRData.decodedRawData == NUM1) {
33             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
34         } else if (IrReceiver.decodedIRData.decodedRawData == NUM2) {
35             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
36         } else if (IrReceiver.decodedIRData.decodedRawData == NUM3) {
37             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
38         }
39         IrReceiver.resume(); // 等下接收下一个值 Wait to receive the next value
40     }
}

```

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  }
```